МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1887ВЕ3Т

Руководство пользователя

2011

Содержание

1 Введение

Одной из быстроразвивающихся областей встраиваемых цифровых систем управления являются системы реального времени, строящиеся на базе современных микроконтроллеров. Остро стоит необходимость реализовывать сложные алгоритмы управления, использующие на входе большое количество цифровых и аналоговых сигналов, когда сигналы отклика должны генерироваться с минимально возможной временной задержкой. Встроенные цифровые системы управления часто являются критичными по площади монтажа, рассеиваемой мощности и стоимости системы в целом, поэтому применение современных высокопроизводительных микроконтроллеров позволяет решить эти проблемы.

Для встраиваемых цифровых систем управления необходимо использовать микроконтроллеры, которые обладают следующими свойствами:

- высокой степенью системной интеграции;
- уменьшенной необходимостью в дополнительных периферийных устройствах и, следовательно, накладных расходах программно-аппаратного обеспечения;
- обеспечение стабильности и надежности систем.

Непрекращающееся увеличение сложности встраиваемых цифровых систем управления требует более существенного увеличения производительности центрального процессорного устройства ЦПУ и периферии микроконтроллера по сравнению с традиционными 8-разрядными микроконтроллерами.

16-разрядный однокристальный КМОП микроконтроллер 1887ВЕ3Т был разработан для удовлетворения высоких требований к рабочим характеристикам цифровых систем управления в реальном времени.

Архитектура оптимизирована для высокоэффективного выполнения инструкций и минимального по времени отклика на внешние воздействия (запросы на прерывание). Программируемые периферийные подсистемы, интегрированные в микроконтроллере, позволяют свести к минимуму необходимость использования внешних периферийных устройств и обмена с ними через внешнюю шину. Высокая гибкость такой архитектуры делает возможным применение микроконтроллера 1887ВЕ3Т в автомобильных и промышленных встраиваемых цифровых системах управления, а также в системах связи.

Архитектура ядра микроконтроллера разработана по модульному принципу.

Контроллер обладает набором команд проверки, дополнительной внутренней быстродействующей оперативной памятью, интегрированным CAN модулем, фазовой автоподстройкой частоты, имеет XBUS интерфейс и т. д.

1.1 Назначение и основные характеристики микросхем

Разработанная микросхема представляет собой высокопроизводительный 16-разрядный микроконтроллер. В автомобилестроении получило широкое распространение использование микроконтроллеров для управления интеллектуальными устройствами и системами.

Применение высокоскоростных 16-разрядных микроконтроллеров с расширенной периферией позволяет осуществлять управление любыми системами – везде, где требуются сбор, обработка и обмен данными между электронными системами управления, наблюдения или измерения, обеспечить требуемые тактико-технические данные, указанные выше, и выполнить требования на аппаратуру (комплексы) по назначению и массогабаритным показателям. Это даст возможность обеспечить комплектование комплексов отечественными микросхемами взамен аналогичных импортных изделий.

Функциональным аналогом разрабатываемого изделия является микросхема SAK-XC167CI-32F40F фирмы Infineon Technologies, США, 2005 г.

Основные технические характеристики микросхемы

Микросхема 1887ВЕ3Т – 16-разрядный RISC-микроконтроллер с тактовой частотой 40 МГц, со встроенными АЦП и ОЗУ, встроенной памятью программ 256 Кбайт типа Flash и расширенной периферией. Архитектура микроконтроллера оптимизирована для работы в режиме реального времени и для обеспечения высокой производительности при малом времени реакции на внешние прерывания. Интегрированная интеллектуальная периферия снижает необходимость вмешательства со стороны центрального процессорного устройства. Мощная система команд, встроенные возможности цифровой обработки сигналов в комбинации с развитым набором встроенной периферии и Flash-памятью программ обеспечивают высокую производительность, гибкость и универсальность.

Микроконтроллер имеет сдвоенный интерфейс CAN (TwinCAN), модуль ШИМ CAPCOM6 и DSP-функциональность. Внутренние средства отладки OCDS/CERBERUS (On Chip Debug System) через отладочный интерфейс JTAG способствуют быстрой разработке программного обеспечения и интеграции систем.

Основные технические характеристики микросхем:

16-разрядный ЦПУ с четырехуровневым конвейером команд
тактовая частота, МГц 40;
объём адресуемой памяти, байт 16М;
объём встроенного ОЗУ, байт 15К;
объём встроенной памяти программ Flash, байт 256К;
объём области регистров специальных функций, байт 1К;
число источников прерываний 112;
число линий ввода-вывода 103;
число каналов аналого-цифрового преобразователя 16;
число разрядов аналого-цифрового преобразователя 10 и 8;
число внешних линий модуля генерации ШИМ 6;
число каналов модулей захвата/сравнения 32;
число 16-разрядных таймерных многофункциональных модулей 1;
число последовательных портов для интерфейсов типа USART, SPI 4;
сдвоенный интерфейс протокола обмена данными CAN 1;
блок интерфейса I2C 1;
отладочный интерфейс JTAG 1;
программируемый сторожевой таймер WDT 1;
число режимов пониженного потребления мощности 3;
напряжение питания, В
ядра микросхемы 2,35 – 2,70;
периферии микросхемы 5,00 ± 0,50.

Система команд разрабатываемой микросхемы соответствует системе команд микроконтроллера SAK-XC167CI-32F40F.

Конструктивные характеристики микросхем

Кристалл микросхем выполнен по КМОП технологии.

Микросхемы разработаны в металлокерамическом корпусе 4247.144-1 (CQFP-144) с четырехсторонним планарным расположением выводов.

Условное графическое обозначение ИС приведено на рисунке 1.1.

Функциональное назначение выводов приведено в таблице 1.1.

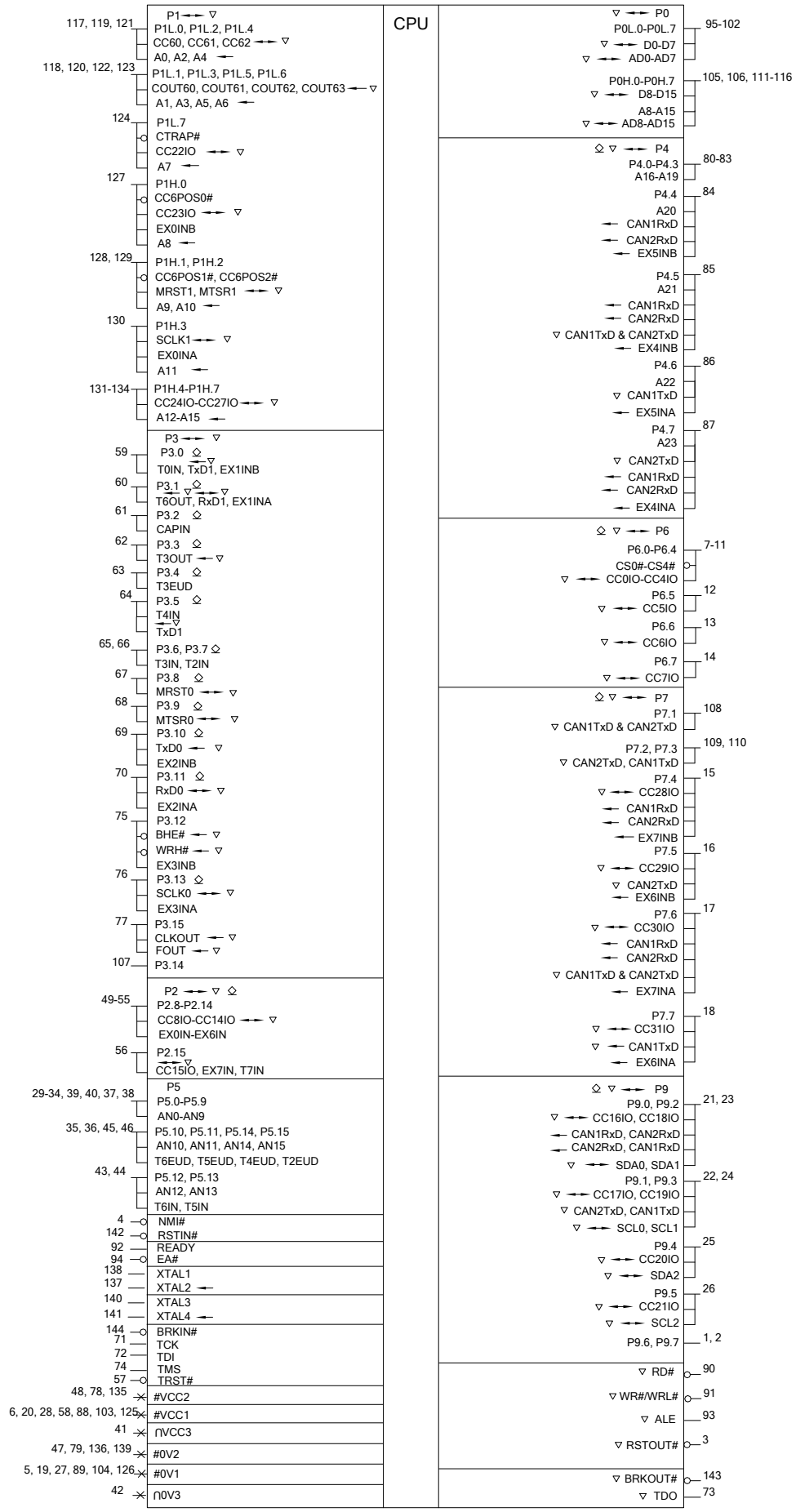


Рисунок 1.1 – Условное графическое изображение микросхемы 1887ВЕ3Т

Таблица 1.1 – Функциональное назначение выводов микросхемы 1887ВЕ3Т

Номер вывода	Обозначение		Функциональное назначение вывода	Тип вывода
	вывода	альтернативной функции вывода		
1	2	3	4	5
4	NMI#		Вход "немаскируемое прерывание"	I
7	P6.0		Вход-выход "порт 6", 0 разряд	I/O/Z/2
		CS0#	Выход "выбор кристалла 0"	O
		CC0IO	Вход "захват"/выход "сравнение", 0 канал	I/O/Z
8	P6.1		Вход-выход "порт 6", 1 разряд	I/O/Z/2
		CS1#	Выход "выбор кристалла 1"	O
		CC1IO	Вход "захват "/выход "сравнение", 1 канал	I/O/Z
9	P6.2		Вход-выход "порт 6", 2 разряд	I/O/Z/2
		CS2#	Выход "выбор кристалла 2"	O
		CC2IO	Вход "захват"/выход "сравнение", 2 канал	I/O/Z
10	P6.3		Вход-выход "порт 6", 3 разряд	I/O/Z/2
		CS3#	Выход "выбор кристалла 3"	O
		CC3IO	Вход "захват"/выход "сравнение", 3 канал	I/O/Z
11	P6.4		Вход-выход "порт 6", 4 разряд	I/O/Z/2
		CS4#	Выход "выбор кристалла 4"	O
		CC4IO	Вход "захват"/выход "сравнение", 4 канал	I/O/Z
12	P6.5		Вход-выход "порт 6", 5 разряд	I/O/Z/2
		CC5IO	Вход "захват"/выход "сравнение", 5 канал	I/O/Z
13	P6.6		Вход-выход "порт 6", 6 разряд	I/O/Z/2
		CC6IO	Вход "захват"/выход "сравнение", 6 канал	I/O/Z
14	P6.7		Вход-выход "порт 6", 7 разряд	I/O/Z/2
		CC7IO	Вход "захват"/выход "сравнение", 7 канал	I/O/Z
108	P7.1		Вход-выход "порт 7", 1 разряд	I/O/Z/2
		CAN1TxD &	Выход 1, 2 данных	
		CAN2TxD	интерфейса CAN	O/Z
109	P7.2		Вход-выход "порт 7", 2 разряд	I/O/Z/2
		CAN2TxD	Выход 2 данных интерфейса CAN	O/Z
110	P7.3		Вход-выход "порт 7", 3 разряд	I/O/Z/2
		CAN1TxD	Выход 1 данных интерфейса CAN	O/Z
15	P7.4		Вход-выход "порт 7", 4 разряд	I/O/Z/2
		CC28IO	Вход "захват"/выход "сравнение", 28 канал	I/O/Z
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		EX7INB	Вход альтернативного сигнала B "прерывание 7"	I
16	P7.5		Вход-выход "порт 7", 5 разряд	I/O/Z/2
		CC29IO	Вход "захват"/выход "сравнение", 29 канал	I/O/Z
		CAN2TxD	Выход 2 данных интерфейса CAN	O/Z
		EX6INB	Вход альтернативного сигнала B "прерывание 6"	I
17	P7.6		Вход-выход "порт 7", 6 разряд	I/O/Z/2
		CC30IO	Вход "захват"/выход "сравнение", 30 канал	I/O/Z
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		CAN1TxD &	Выход 1, 2 данных	
		CAN2TxD	интерфейса CAN	O/Z
		EX7INA	Вход альтернативного сигнала A "прерывание 7"	I
18	P7.7		Вход-выход «порт 7», 7 разряд	I/O/Z/2
		CC31IO	Вход «захват»/выход «сравнение», 31 канал	I/O/Z
		CAN1TxD	Выход 1 данных интерфейса CAN	O/Z
		EX6INA	Вход альтернативного сигнала A «прерывание 6»	I

Продолжение таблицы 1.1

1	2	3	4	5
21	P9.0		Вход-выход «порт 9», 0 разряд	I/O/Z/2
		CC16IO	Вход «захват»/выход «сравнение», 16 канал	I/O/Z
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		SDA0	Вход-выход 0 данных шины I2C	I/O/Z
22	P9.1		Вход-выход «порт 9», 1 разряд	I/O/Z/2
		CC17IO	Вход «захват»/выход «сравнение», 17 канал	I/O/Z
		CAN2TxD	Выход 2 данных интерфейса CAN	O/Z
		SCL0	Вход-выход 0 синхронизации шины I2C	I/O/Z
23	P9.2		Вход-выход «порт 9», 2 разряд	I/O/Z/2
		CC18IO	Вход «захват»/выход «сравнение», 18 канал	I/O/Z
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		SDA1	Вход-выход 1 данных шины I2C	I/O/Z
24	P9.3		Вход-выход «порт 9», 3 разряд	I/O/Z/2
		CC19IO	Вход «захват»/выход «сравнение», 19 канал	I/O/Z
		CAN1TxD	Выход 1 данных интерфейса CAN	O/Z
		SCL1	Вход-выход 1 синхронизации шины I2C	I/O/Z
25	P9.4		Вход-выход «порт 9», 4 разряд	I/O/Z/2
		CC20IO	Вход «захват»/выход «сравнение», 20 канал	I/O/Z
		SDA2	Вход-выход 2 данных шины I2C	I/O/Z
26	P9.5		Вход-выход «порт 9», 5 разряд	I/O/Z/2
		CC21IO	Вход «захват»/выход «сравнение», 21 канал	I/O/Z
		SCL2	Вход-выход 2 синхронизации шины I2C	I/O/Z
1	P9.6		Вход-выход «порт 9», 6 разряд	I/O/Z/2
2	P9.7		Вход-выход «порт 9», 7 разряд	I/O/Z/2
29	P5.0		Вход «порт 5», 0 разряд	I
		AN0	Вход АЦП, 0 канал	I
30	P5.1		Вход «порт 5», 1 разряд	I
		AN1	Вход АЦП, 1 канал	I
31	P5.2		Вход «порт 5», 2 разряд	I
		AN2	Вход АЦП, 2 канал	I
32	P5.3		Вход «порт 5», 3 разряд	I
		AN3	Вход АЦП, 3 канал	I
33	P5.4		Вход «порт 5», 4 разряд	I
		AN4	Вход АЦП, 4 канал	I
34	P5.5		Вход «порт 5», 5 разряд	I
		AN5	Вход АЦП, 5 канал	I
39	P5.6		Вход «порт 5», 6 разряд	I
		AN6	Вход АЦП, 6 канал	I
40	P5.7		Вход «порт 5», 7 разряд	I
		AN7	Вход АЦП, 7 канал	I
37	P5.8		Вход «порт 5», 8 разряд	I
		AN8	Вход АЦП, 8 канал	I
38	P5.9		Вход «порт 5», 9 разряд	I
		AN9	Вход АЦП, 9 канал	I
35	P5.10		Вход «порт 5», 10 разряд	I
		AN10	Вход АЦП, 10 канал	I
		T6EUD	Вход «направление счета таймера 6»	I
36	P5.11		Вход «порт 5», 11 разряд	I
		AN11	Вход АЦП, 11 канал	I
		T5EUD	Вход «направление счета таймера 5»	I

Продолжение таблицы 1.1

1	2	3	4	5
43	P5.12		Вход «порт 5», 12 разряд	I
		AN12	Вход АЦП, 12 канал	I
		T6IN	Вход «счет таймера 6»	I
44	P5.13		Вход «порт 5», 13 разряд	I
		AN13	Вход АЦП, 13 канал	I
		T5IN	Вход «счет таймера 5»	I
45	P5.14		Вход «порт 5», 14 разряд	I
		AN14	Вход АЦП, 14 канал	I
		T4EUD	Вход «направление счета таймера 4»	I
46	P5.15		Вход «порт 5», 15 разряд	I
		AN15	Вход АЦП, 15 канал	I
		T2EUD	Вход «направление счета таймера 2»	I
49	P2.8		Вход-выход «порт 2», 8 разряд	I/O/Z/2
		CC8IO	Вход «захват»/выход «сравнение», 8 канал	I/O/Z
		EX0IN	Вход по умолчанию «прерывание 0»	I
50	P2.9		Вход-выход «порт 2», 9 разряд	I/O/Z/2
		CC9IO	Вход «захват»/выход «сравнение», 9 канал	I/O/Z
		EX1IN	Вход по умолчанию «прерывание 1»	I
51	P2.10		Вход-выход «порт 2», 10 разряд	I/O/Z/2
		CC10IO	Вход «захват»/выход «сравнение», 10 канал	I/O/Z
		EX2IN	Вход по умолчанию «прерывание 2»	I
52	P2.11		Вход-выход «порт 2», 11 разряд	I/O/Z/2
		CC11IO	Вход «захват»/выход «сравнение», 11 канал	I/O/Z
		EX3IN	Вход по умолчанию «прерывание 3»	I
53	P2.12		Вход-выход «порт 2», 12 разряд	I/O/Z/2
		CC12IO	Вход «захват»/выход «сравнение», 12 канал	I/O/Z
		EX4IN	Вход по умолчанию «прерывание 4»	I
54	P2.13		Вход-выход «порт 2», 13 разряд	I/O/Z/2
		CC13IO	Вход «захват»/выход «сравнение», 13 канал	I/O/Z
		EX5IN	Вход по умолчанию «прерывание 5»	I
55	P2.14		Вход-выход «порт 2», 14 разряд	I/O/Z/2
		CC14IO	Вход «захват»/выход «сравнение», 14 канал	I/O/Z
		EX6IN	Вход по умолчанию «прерывание 6»	I
56	P2.15		Вход-выход «порт 2», 15 разряд	I/O/Z/2
		CC15IO	Вход «захват»/выход «сравнение», 15 канал	I/O/Z
		EX7IN	Вход по умолчанию «прерывание 7»	I
		T7IN	Вход «счет таймера 7»	I
57	TRST#		Вход «сброс системы тестирования»	I
59	P3.0		Вход-выход «порт 3», 0 разряд	I/O/Z/2
		T0IN	Вход «счет таймера 0»	I
		TxD1	Выход «такт/данные, асинхронный/синхронный УСАПП1»	O/Z
		EX1INB	Вход альтернативного сигнала B «прерывание 1»	I
60	P3.1		Вход-выход «порт 3», 1 разряд	I/O/Z/2
		T6OUT	Выход «триггер таймера 6»	O/Z
		RxD1	Вход-выход «синхронный/асинхронный ввод/синхронный вывод данных УСАПП1»	I/O/Z
		EX1INA	Вход альтернативного сигнала A «прерывание 1»	I
61	P3.2		Вход-выход "порт 3", 2 разряд	I/O/Z/2
		CAPIN	Вход "загрузка"	I
62	P3.3		Вход-выход "порт 3", 3 разряд	I/O/Z/2
		T3OUT	Выход "триггер таймера 3"	O/Z
63	P3.4		Вход-выход "порт 3", 4 разряд	I/O/Z/2
		T3EUD	Вход "направление счета таймера 3"	I

Продолжение таблицы 1.1

1	2	3	4	5
64	P3.5		Вход-выход "порт 3", 5 разряд	I/O/Z/2
		T4IN	Вход "режим таймера 4	I
		TxD1	Выход «такт/данные, асинхронный/синхронный УСАПП1»	O/Z
65	P3.6		Вход-выход "порт 3", 6 разряд	I/O/Z/2
		T3IN	Вход "режим таймера 3"	I
66	P3.7		Вход-выход "порт 3", 7 разряд	I/O/Z/2
		T2IN	Вход "режим таймера 2"	I
67	P3.8		Вход-выход "порт 3", 8 разряд	I/O/Z/2
		MRST0	Вход-выход "M/S ввод-вывод данных синхронного порта 0"	I/O/Z
68	P3.9		Вход-выход "порт 3", 9 разряд	I/O/Z/2
		MTSR0	Вход-выход "M/S вывод-ввод данных синхронного порта 0"	I/O/Z
69	P3.10		Вход-выход "порт 3", 10 разряд	I/O/Z/2
		TxD0	Выход "такт/данные, асинхронный/синхронный УСАПП0"	O/Z
		EX2INB	Вход альтернативного сигнала B "прерывание 2"	I
70	P3.11		Вход-выход "порт 3", 11 разряд	I/O/Z/2
		RxD0	Вход-выход "синхронный/асинхронный ввод/синхронный вывод данных УСАПП0"	I/O/Z
		EX2INA	Вход альтернативного сигнала A "прерывание 2"	I
75	P3.12		Вход-выход "порт 3", 12 разряд	I/O/Z
		BHE#	Выход "выбор старшего байта"	O/Z
		WRH#	Выход "запись старшего байта"	O/Z
		EX3INB	Вход альтернативного сигнала B "прерывание 3"	I
76	P3.13		Вход-выход "порт 3", 13 разряд	I/O/Z/2
		SCLK0	Вход-выход "M-вывод/S-ввод тактового сигнала синхронного порта 0"	I/O/Z
		EX3INA	Вход альтернативного сигнала A "прерывание 3"	I
107	P3.14		Вход-выход "порт 3", 14 разряд	I/O/Z
77	P3.15		Вход-выход "порт 3", 15 разряд	I/O/Z
		CLKOUT	Выход "системный тактовый сигнал"	O/Z
		FOUT	Выход "программируемая частота"	O/Z
71	TCK		Вход "синхронизация JTAG" отладочной системы	I
72	TDI		Вход "данные JTAG" отладочной системы	I
73	TDO		Выход "данные JTAG" отладочной системы	O/Z
74	TMS		Вход "выбор тестового режима JTAG" отладочной системы	I
80	P4.0		Вход-выход "порт 4", 0 разряд	I/O/Z/2
		A16	Выход адреса, 16 разряд	O
81	P4.1		Вход-выход "порт 4", 1 разряд	I/O/Z/2
		A17	Выход адреса, 17 разряд	O
82	P4.2		Вход-выход "порт 4", 2 разряд	I/O/Z/2
		A18	Выход адреса, 18 разряд	O
83	P4.3		Вход-выход "порт 4", 3 разряд	I/O/Z/2
		A19	Выход адреса, 19 разряд	O
84	P4.4		Вход-выход "порт 4", 4 разряд	I/O/Z/2
		A20	Выход адреса, 20 разряд	O
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		EX5INB	Вход альтернативного сигнала B "прерывание 5"	I
85	P4.5		Вход-выход "порт 4", 5 разряд	I/O/Z/2
		A21	Выход адреса, 21 разряд	O
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		CAN1TxD&	Выход 1, 2 данных	
		CAN2TxD	интерфейса CAN	O/Z
		EX4INB	Вход альтернативного сигнала B "прерывание 4"	I

Продолжение таблицы 1.1

1	2	3	4	5
86	P4.6		Вход-выход "порт 4", 6 разряд	I/O/Z/2
		A22	Выход адреса, 22 разряд	O
		CAN1TxD	Выход 1 данных интерфейса CAN	O/Z
		EX5INA	Вход альтернативного сигнала A "прерывание 5"	I
87	P4.7		Вход-выход "порт 4", 7 разряд	I/O/Z/2
		A23	Выход адреса, 23 разряд	O
		CAN2TxD	Выход 2 данных интерфейса CAN	O/Z
		CAN1RxD	Вход 1 данных интерфейса CAN	I
		CAN2RxD	Вход 2 данных интерфейса CAN	I
		EX4INA	Вход альтернативного сигнала A "прерывание 4"	I
90	RD#		Выход "чтение команд/данных"	O/Z
91	WR#/ WRL#		Выход "запись данных/ запись младшего байта"	O/Z
92	READY		Вход "готовность"	I
93	ALE		Выход "строб адреса"	O/Z
94	EA#		Вход "внешний доступ"	I
3	RSTOUT#		Выход "индикация сброса"	O/Z
95	P0L.0		Вход-выход "порт 0, младший байт", 0 разряд	I/O/Z
		D0	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 0 разряд	I/O/Z
		AD0	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 0 разряд	I/O/Z
96	P0L.1		Вход-выход "порт 0, младший байт", 1 разряд	I/O/Z
		D1	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 1 разряд	I/O/Z
		AD1	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 1 разряд	I/O/Z
97	P0L.2		Вход-выход "порт 0, младший байт", 2 разряд	I/O/Z
		D2	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 2 разряд	I/O/Z
		AD2	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 2 разряд	I/O/Z
98	P0L.3		Вход-выход "порт 0, младший байт", 3 разряд	I/O/Z
		D3	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 3 разряд	I/O/Z
		AD3	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 3 разряд	I/O/Z
99	P0L.4		Вход-выход "порт 0, младший байт", 4 разряд	I/O/Z
		D4	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 4 разряд	I/O/Z
		AD4	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 4 разряд	I/O/Z
100	P0L.5		Вход-выход "порт 0, младший байт", 5 разряд	I/O/Z
		D5	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 5 разряд	I/O/Z
		AD5	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 5 разряд	I/O/Z
101	P0L.6		Вход-выход "порт 0, младший байт", 6 разряд	I/O/Z
		D6	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 6 разряд	I/O/Z
		AD6	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 6 разряд	I/O/Z
102	P0L.7		Вход-выход "порт 0, младший байт", 7 разряд	I/O/Z
		D7	Демультиплексированная шина 8 бит, 16 бит, вход-выход данных, 7 разряд	I/O/Z
		AD7	Мультиплексированная шина 8 бит, 16 бит, вход-выход адреса/данных, 7 разряд	I/O/Z
105	P0H.0		Вход-выход "порт 0, старший байт", 0 разряд	I/O/Z
		D8	Демультиплексированная шина 16 бит, вход-выход данных, 8 разряд	I/O/Z
		A8	Мультиплексированная шина 16 бит, выход адреса, 8 разряд	O
		AD8	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 8 разряд	I/O/Z
106	P0H.1		Вход-выход "порт 0, старший байт", 1 разряд	I/O/Z
		D9	Демультиплексированная шина 16 бит, вход-выход данных, 9 разряд	I/O/Z
		A9	Мультиплексированная шина 16 бит, выход адреса, 9 разряд	O
		AD9	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 9 разряд	I/O/Z
111	P0H.2		Вход-выход "порт 0, старший байт", 2 разряд	I/O/Z
		D10	Демультиплексированная шина 16 бит, вход-выход данных, 10 разряд	I/O/Z
		A10	Мультиплексированная шина 16 бит, выход адреса, 10 разряд	O
		AD10	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 10 разряд	I/O/Z

Продолжение таблицы 1.1

1	2	3	4	5
112	P0H.3		Вход-выход "порт 0, старший байт", 3 разряд	I/O/Z
		D11	Демультиплексированная шина 16 бит, вход-выход данных, 11 разряд	I/O/Z
		A11	Мультиплексированная шина 16 бит, выход адреса, 11 разряд	O
		AD11	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 11 разряд	I/O/Z
113	P0H.4		Вход-выход "порт 0, старший байт", 4 разряд	I/O/Z
		D12	Демультиплексированная шина 16 бит, вход-выход данных, 12 разряд	I/O/Z
		A12	Мультиплексированная шина 16 бит, выход адреса, 12 разряд	O
		AD12	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 12 разряд	I/O/Z
114	P0H.5		Вход-выход "порт 0, старший байт", 5 разряд	I/O/Z
		D13	Демультиплексированная шина 16 бит, вход-выход данных, 13 разряд	I/O/Z
		A13	Мультиплексированная шина 16 бит, выход адреса, 13 разряд	O
		AD13	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 13 разряд	I/O/Z
115	P0H.6		Вход-выход "порт 0, старший байт", 6 разряд	I/O/Z
		D14	Демультиплексированная шина 16 бит, вход-выход данных, 14 разряд	I/O/Z
		A14	Мультиплексированная шина 16 бит, выход адреса, 14 разряд	O
		AD14	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 14 разряд	I/O/Z
116	P0H.7		Вход-выход "порт 0, старший байт", 7 разряд	I/O/Z
		D15	Демультиплексированная шина 16 бит, вход-выход данных, 15 разряд	I/O/Z
		A15	Мультиплексированная шина 16 бит, выход адреса, 15 разряд	O
		AD15	Мультиплексированная шина 16 бит, вход-выход адреса/данных, 15 разряд	I/O/Z
117	P1L.0		Вход-выход "порт 1, младший байт", 0 разряд	I/O/Z
		CC60	Вход-выход модуля захвата/сравнения 6, канал 0	I/O/Z
		A0	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 0 разряд	O
118	P1L.1		Вход-выход "порт 1, младший байт", 1 разряд	I/O/Z
		COUT60	Выход модуля захвата/сравнения 6, канал 0	O/Z
		A1	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 1 разряд	O
119	P1L.2		Вход-выход "порт 1, младший байт", 2 разряд	I/O/Z
		CC61	Вход-выход модуля захвата/сравнения 6, канал 1	I/O/Z
		A2	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 2 разряд	O
120	P1L.3		Вход-выход "порт 1, младший байт", 3 разряд	I/O/Z
		COUT61	Выход модуля захвата/сравнения 6, канал 1	O/Z
		A3	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 3 разряд	O
121	P1L.4		Вход-выход "порт 1, младший байт", 4 разряд	I/O/Z
		CC62	Вход-выход модуля захвата/сравнения 6, канал 2	I/O/Z
		A4	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 4 разряд	O
122	P1L.5		Вход-выход "порт 1, младший байт", 5 разряд	I/O/Z
		COUT62	Выход модуля захвата/сравнения 6, канал 2	O/Z
		A5	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 5 разряд	O
123	P1L.6		Вход-выход "порт 1, младший байт", 6 разряд	I/O/Z
		COUT63	Выход 10-битного канала сравнения	O/Z
		A6	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 6 разряд	O
124	P1L.7		Вход-выход "порт 1, младший байт", 7 разряд	I/O/Z
		CTRAP#	Вход "системное прерывание" модуля захвата/сравнения 6	I
		CC22IO	Вход "захват"/выход "сравнение" 22 канал	I/O/Z
		A7	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 7 разряд	O
127	P1H.0		Вход-выход "порт 1, старший байт", 0 разряд	I/O/Z
		CC6POS0#	Вход "позиция 0" модуля захвата/сравнения 6	I
		CC23IO	Вход "захват"/выход "сравнение", 23 канал	I/O/Z
		EX0INB	Вход альтернативного сигнала B "прерывание 0"	I
		A8	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 8 разряд	O

Окончание таблицы 1.1

1	2	3	4	5
128	P1H.1		Вход-выход "порт 1, старший байт", 1 разряд	I/O/Z
		CC6POS1#	Вход "позиция 1" модуля захвата/сравнения 6	I
		MRST1	Вход-выход "M/S ввод-вывод данных синхронного порта 1"	I/O/Z
		A9	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 9 разряд	O
129	P1H.2		Вход-выход "порт 1, старший байт", 2 разряд	I/O/Z
		CC6POS2#	Вход "позиция 2" модуля захвата/сравнения 6	I
		MTSR1	Вход-выход "M/S вывод-ввод данных синхронного порта 1"	I/O/Z
		A10	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 10 разряд	O
130	P1H.3		Вход-выход "порт 1, старший байт", 3 разряд	I/O/Z
		SCLK1	Вход-выход "M-вывода/S-ввода тактового сигнала синхронного порта 1"	I/O/Z
		EX0INA	Вход альтернативного сигнала A "прерывание 0"	I
		A11	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 11 разряд	O
131	P1H.4		Вход-выход "порт 1, старший байт", 4 разряд	I/O/Z
		CC24IO	Вход "захват"/выход "сравнение", 24 канал	I/O/Z
		A12	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 12 разряд	O
132	P1H.5		Вход-выход "порт 1, старший байт", 5 разряд	I/O/Z
		CC25IO	Вход "захват"/выход "сравнение", 25 канал	I/O/Z
		A13	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 13 разряд	O
133	P1H.6		Вход-выход "порт 1, старший байт", 6 разряд	I/O/Z
		CC26IO	Вход "захват"/выход "сравнение", 26 канал	I/O/Z
		A14	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 14 разряд	O
134	P1H.7		Вход-выход "порт 1, старший байт", 7 разряд	I/O/Z
		CC27IO	Вход "захват"/выход "сравнение", 27 канал	I/O/Z
		A15	Демультиплексированная шина 8 бит, 16 бит, выход адреса, 15 разряд	O
137	XTAL2		Выход усилителя задающего генератора	O
			Вывод для подключения кварцевого резонатора	–
138	XTAL1		Вход внешнего тактового сигнала	I
			Вывод для подключения кварцевого резонатора	–
140	XTAL3		Вход усилителя дополнительного (32 кГц) тактового генератора	I
			Вывод для подключения кварцевого резонатора	–
141	XTAL4		Выход усилителя дополнительного (32 кГц) задающего генератора	O
			Вывод для подключения кварцевого резонатора	–
142	RSTIN#		Вход "сброс"	I
143	BRKOUT#		Выход "останов" отладочной системы	O/Z
144	BRKIN#		Вход "останов" отладочной системы	I
41	∩VCC3		Вывод опорного напряжения АЦП	–
42	∩0V3		Общий вывод АЦП	–
48, 78, 135	#VCC2		Выводы питания ядра микросхемы	–
6, 20, 28, 58, 88, 103, 125	#VCC1		Выводы питания периферийной части микросхемы	–
47, 79, 136, 139	#0V2		Выводы общей шины ядра микросхемы	–
5, 19, 27, 89, 104, 126	#0V1		Выводы общей шины периферийной части микросхемы	–

Примечание – В графе «Тип вывода»: I – вход; O – выход; Z – третье состояние; 2 – режим открытого стока.

1.2 Электрические параметры микросхем

Номинальное значение напряжения питания периферии микросхемы составляет 5,0 В, допустимые отклонения значения напряжения питания от номинального должны быть не более ± 10 %.

Напряжения источника питания ядра микросхемы должно быть от 2,35 до 2,70 В.

Напряжение опорного источника АЦП должно быть от 4,5 до 5,6 В. Допустимые отклонения напряжения опорного источника АЦП от крайних значений – не более плюс 1 % для напряжения 4,5 В и не менее минус 1 % для напряжения 5,6 В. Напряжение питания АЦП $U_{\cap VCC3}$ не должно превышать напряжение питания периферии $U_{\#VCC1}$, оба напряжения должны находиться в одном диапазоне.

Электрические параметры микросхем при приёмке и поставке в диапазоне рабочих температур окружающей среды от минус 60 до 85 °С приведены в таблице 1.2.

Значения предельно допустимых и предельных электрических режимов эксплуатации микросхем в диапазоне рабочих температур среды приведены в таблице 1.3.

Таблица 1.2 – Электрические параметры микросхем при приёмке и поставке в диапазоне рабочих температур окружающей среды от минус 60 до 85 °С

Наименование параметра, единица измерения, режим измерения		Буквенное обозначение параметра	Норма параметра		Температура среды, ºС
			не менее	не более	
1		2	3	4	5
Выходное напряжение низкого уровня на выводах P0 – P4, P6, P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#, В, I_{OL} = 5,0 мА, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,35 В		U_{OL}	–	0,45	–60 ± 3 25 ± 10 85 ± 3
Выходное напряжение высокого уровня на выводах P0 – P4, P6, P7, P9, ALE, RD#, WR#/WRL#, RSTOUT# [1], В, I_{OH} = –5,0 мА, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,35 В		U_{OH}	4,05	–	
Ток утечки на входах P5.0 – P5.15, нА, $U_{\#VCC1}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В	U_{IL} = 0 В, $U_{\cap VCC3}$ = 0 В	I_{ILL1}	–200,0	200,0	
	U_{IH} = 5,5 В, $U_{\cap VCC3}$ = 5,5 В	I_{ILH1}			
Ток утечки на входах NMI#, TRST#, TCK, TDI, TMS, RSTIN#, BRKIN#, EA#, нА, U_{IL} = 0,45 В, U_{IH} = 5,5 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В		I_{ILL2}, I_{ILH2}	–500,0	500,0	
Ток утечки в состоянии «Выключено» по выводам P0 – P4, P6, P7, P9, TDO, нА, U_{IL} = 0,45 В, U_{IH} = 5,5 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В		I_{OZL}, I_{OZH}	–500,0	500,0	

Продолжение таблицы 1.2

1		2	3	4	5
Входной ток низкого уровня по выводам P0, RD#, WR#/WRL#, READY[2], мкА, U_{IL} = 0,8 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{IL1}	–100,0	–	
Входной ток высокого уровня по выводам P0, RD#, WR#/WRL#, READY[2], мкА, U_{IH} = 1,8 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{IH1}	–	–10,0	
Входной ток низкого и высокого уровня по выводам XTAL1, XTAL3, мкА, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В	U_{IL} = 0 В	I_{IL2}	–20,0	–	
	U_{IH} = 5,5 В	I_{IH2}	–	20,0	
Выходной ток низкого уровня по выводу ALE[2], мкА, U_{OL} = 0,8 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{OL1}	10,0	–	
Выходной ток высокого уровня по выводу ALE[2], мкА, U_{OH} = 1,8 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{OH1}	–	120,0	
Выходной ток низкого уровня по выводам P6.0 – P6.4[2], мкА, U_{OL} = 0,45 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{OL2}	–100,0	–	–60 ± 3 25 ± 10 85 ± 3
Выходной ток высокого уровня по выводам P6.0 – P6.4 [2], мкА, U_{OH} = 2,25 В, $U_{\#VCC1} = U_{\cap VCC3}$ = 4,5 В, $U_{\#VCC2}$ = 2,7 В		I_{OH2}	–	–10,0	
Ток потребления по источнику питания $U_{\#VCC1}$ [3], мА, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В		$I_{\#VCC1}$	–	5,0	
Ток потребления по источнику питания $U_{\#VCC2}$ в активном режиме [3], мА, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В, $f_{CIXTAL1}$ = 40 МГц		$I_{\#VCC2}$	–	60,0	
Ток потребления по источнику питания $U_{\#VCC2}$ в режиме холостого хода [3], мА, $U_{\#VCC2}$ = 2,7 В, $f_{CIXTAL1}$ = 40 МГц		I_{CCI}	–	30,0	
Ток потребления по источнику питания $U_{\#VCC2}$ в режиме хранения, мА, $U_{\#VCC1} = U_{\cap VCC3}$ = 5,5 В, $U_{\#VCC2}$ = 2,7 В, $f_{CIXTAL1}$ = 16 МГц		I_{CCS1}	–	1,15	

Окончание таблицы 1.2

1		2	3	4	5
Ток потребления по источнику питания $U_{\#VCC2}$ в режиме хранения, мА, $U_{\#VCC1} = U_{\cap VCC3} = 5{,}5$ В, $U_{\#VCC2} = 2{,}7$ В, $f_{CIXTAL3} = 32$ кГц		I_{CCS2}	–	0,35	–60 ± 3 25 ± 10 85 ± 3
Функциональный контроль, $U_{\#VCC1} = (4{,}5; 5{,}5)$ В, $U_{\#VCC2} = (2{,}35; 2{,}7)$ В, $U_{\cap VCC3} = (4{,}5; 5{,}6)$ В, $f_{CIXTAL1} = (4{,}0; 40{,}0)$ МГц		ФК	–	–	
Время переключения сигнала на выходе CLKOUT, нс, $C_L = 50$ пФ, $U_{\#VCC1} = 4{,}5$ В, $U_{\#VCC2} = 2{,}35$ В, $U_{\cap VCC3} = 4{,}5$ В, $f_{CIXTAL1} = 10{,}0$ МГц	время спада	t_f	–	5,0	
	время нарастания	t_r	–	5,0	

Примечание – Параметры I_{ILL1}, I_{ILH1}, I_{ILL2}, I_{ILH2}, I_{OZL}, I_{OZH}, I_{IH1}, I_{IL2}, I_{IH2}, I_{OL1}, I_{OH2} при температуре минус 60 ºС не измеряются, а гарантируются нормой при температуре (25 ± 10) ºС.

_____

[1] Параметры не действительны для выводов портов, запрограммированных в режим «открытый сток».

[2] Параметры действительны в течение действия активного сигнала RSTIN#.

[3] Параметры измеряются при отключенных от нагрузок выходах и входах, подключенных к уровням U_{IL} или U_{IH}.

Таблица 1.3 – Значения предельных и предельно допустимых электрических режимов эксплуатации микросхем в диапазоне рабочих температур среды от минус 60 до 85 °С

Наименование параметра режима, единица измерения	Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
1	2	3	4	5	6
Напряжение источника питания периферии микросхемы, В	$U_{\#VCC1}$	4,50	5,50	–0,50	6,50
Напряжение источника питания ядра микросхемы, В	$U_{\#VCC2}$	2,35	2,70	–0,50	3,25
Напряжение опорного источника АЦП, В	$U_{\cap VCC3}$	4,50	5,60	–0,50	6,00
Входное напряжение низкого уровня в режиме «ТТЛ-порог», В	U_{IL1}	–0,50	$0{,}2U_{\#VCC1}–0{,}10$	–0,60	–

Окончание таблицы 1.3

1	2	3	4	5	6
Входное напряжение низкого уровня по выводам XTAL1, XTAL3, В	U_{IL2}	–0,50	$0,3U_{\#VCC1}$	–0,60	–
Входное напряжение высокого уровня в режиме «ТТЛ-порог», В	U_{IH1}	$0,2U_{\#VCC1}+0,90$	$U_{\#VCC1}+0,50$	–	$U_{\#VCC1}+0,60$
Входное напряжение высокого уровня по выводам XTAL1, XTAL3, В	U_{IH2}	$0,7U_{\#VCC1}$	$U_{\#VCC1}+0,50$	–	$U_{\#VCC1}+0,60$
Выходной ток низкого уровня [1], мА	I_{OL}	–	5,0	–	10,0
Выходной ток высокого уровня [1], мА	I_{OH}	–5,0	–	–10,0	–
Емкость нагрузки, пФ	C_L	–	50,0	–	100,0
Тактовая частота, МГц	$f_{CIXTAL1}$	4,0	40,0	–	–
Длительность фронтов сигнала для входа XTAL1 [2), 4)], нс	t_{LH1}, t_{HL1}	–	5,0	–	–
Длительность фронтов сигнала для остальных входов [3), 4)], нс	t_{LH2}, t_{HL2}	–	10,0	–	–

[1)] Значение суммарного максимально допустимого тока при одновременном подключении нескольких портов – не более 50 мА.

[2)] Значения параметров приведены для максимальной тактовой частоты по входу XTAL1, равной 40 МГц, и коэффициенте деления тактовой частоты, равном 2.

При коэффициенте, равном 1 (прямая передача тактового сигнала), и предельно допустимой тактовой частоте длительность тактового сигнала – не более 8 нс. При использовании встроенного PLL в режиме умножения частоты и предельно допустимой тактовой частоте длительность тактового сигнала – не более 10 нс.

[3)] Значения параметров приведены для максимальной рабочей частоты и минимальной длительности цикла внешней шины.

[4)] При уменьшении тактовой частоты длительности фронтов и спадов входных сигналов могут быть увеличены пропорционально уменьшению тактовой частоты.

1.3 Особенности микроконтроллера

Микроконтроллер 1887ВЕ3Т сочетает в себе высокую производительность центрального процессорного устройства с функциональными возможностями периферии.

Высокопроизводительное 16-разрядное ЦПУ с 4-ступенчатым конвейером:

- 50 нс для выполнения одного машинного цикла инструкции, причем большинство инструкций выполняется за 1 цикл;
- 250 нс для выполнения умножения (двух 16-разрядных чисел), 500 нс для выполнения деления (32-разрядного числа на 16-разрядное);
- многочисленные внутренние шины данных с высокой пропускной способностью;
- архитектура с поддержкой переключения банков регистров общего назначения (контекста) всего за одну инструкцию;
- адресное пространство размером 16 Мбайт для кода и данных (архитектура фон-Неймана);
- системный стек с аппаратным детектированием переполнения/опустошения.

Высокоэффективная система инструкций:

- используемые типы данных – бит, байт и слово (2-байтовый тип данных);
- гибкие и эффективные режимы адресации байтов, слов и битов: непосредственная, косвенная, непосредственно-косвенная и базовая;

- улучшенные логические операции для управления периферией и работы с пользовательскими флагами;

- аппаратный механизм детектирования исключительных и ошибочных ситуаций;

- поддержка языка высокого уровня для управления операциями семафора и эффективного доступа к данным.

Интерфейс внешней шины:

- мультиплексная или демультиплексная конфигурации шины;
- возможность сегментации и формирования сигнала выбора внешнего устройства;
- возможность переключения между 8- или 16-разрядным режимами шины данных;
- изменяемые временные параметры циклов шины для пяти программируемых адресных окон.

16-уровневая система приоритетов прерываний:

- 112 источников прерываний, каждый со своим независимым вектором;
- 16 уровней приоритета и 4 (8) групп.

8-канальный контроллер периферийных событий PEC:

- прерывание работы ЦПУ на один машинный цикл для передачи данных;
- счетчик числа передач (прерывание ЦПУ после запрограммированного количества передач PEC);
- длинный счетчик числа передач;
- компоновка каналов;
- возможность исключения излишних действий на сохранение и восстановление состояния системных регистров во время операций обслуживания прерываний.

Интегрированные периферийные модули

Основными периферийными модулями микроконтроллера являются:

- параллельные порты;
- многофункциональный таймерный модуль GPT1, GPT2;
- часы реального времени RTC;
- АЦП (ADC);
- два блока захвата/сравнения CAPCOM1, CAPCOM2;
- блок захвата/сравнения и широтно-импульсной модуляции ШИМ (CAPCOM6);
- два асинхронно/синхронных последовательных канала ASC0, ASC1 типа USART;
- два высокоскоростных синхронных последовательных канала SSC0, SSC1 типа SPI;
- блок интерфейса I2C;
- сдвоенный интерфейс протокола обмена данными CAN;
- блок управления выходом Wakeup из режимов покоя Idle и Sleep;
- отладочная система OCDS/CERBERUS и JTAG;
- блок генератора частоты с PLL;
- блок управления ядра CCB, включающий:
    - блок управления сбросом RC;
    - блок контроля пониженного энергопотребления PSC;
    - генератор сигналов разрешения тактирования CEG;
    - сторожевой таймер WDT.

Дополнительные функции

Режимы пониженного энергопотребления:

- отключение ядра при работе внутренней периферии;
- отключение ядра и внутренней периферии;
- программное понижение тактовой частоты ЦПУ.

2 Структура и организация микроконтроллера

Архитектура микроконтроллера 1887ВЕ3Т объединяет в себе преимущества как RISC, так и CISC процессоров, при этом удалось достичь хорошо сбалансированного результата. Микроконтроллер не только имеет мощное процессорное ядро и набор периферийных модулей, но также имеет высокоэффективную систему взаимодействия между ними. На равне с другими шинами, в микроконтроллере используется шина XBUS – внутренняя шина X-периферии, – обеспечивающая стандартный способ интеграции в микроконтроллер специальных блоков для различных систем.

В состав микроконтроллера входят:

- центральное процессорное устройство ЦПУ:
    - центральный процессор;
    - блок управления ядра CCB;
    - отладочная система OCDS/CERBERUS и отладочный интерфейс JTAG;
    - контроллер прерываний;
    - контроллер внешней шины EBC;
    - генератор сигналов разрешения тактирования CEG;
    - блок управления сбросом RC;
    - блок контроля пониженного энергопотребления PSC;
    - сторожевой таймер WDT;
- оперативная память DPRAM, PSRAM, XRAM и память программ Flash;
- блок генератора частоты и PLL;
- блок захвата/сравнения и широтноимпульсной модуляции ШИМ (CAPCOM6);
- блок интерфейса I2C;
- блоки захвата/сравнения CAPCOM1, CAPCOM2;
- синхронные последовательные интерфейсы SSC0, SSC1 типа SPI;
- асинхронно/синхронные последовательные интерфейсы ASC0, ASC1 типа USART;
- многофункциональный таймерный модуль GPT1, GPT2;
- блок часов реального времени RTC;
- сдвоенный интерфейс протокола обмена данными CAN;
- параллельные порты P0, P1, P2, P3, P4, P5, P6, P7, P9;
- шина локальной памяти LM66;
- внутренняя шина X-периферии XBUS;
- периферийная шина данных PDBUS.

Структурная схема микроконтроллера 1887ВЕ3Т представлена на рисунке 2.1.

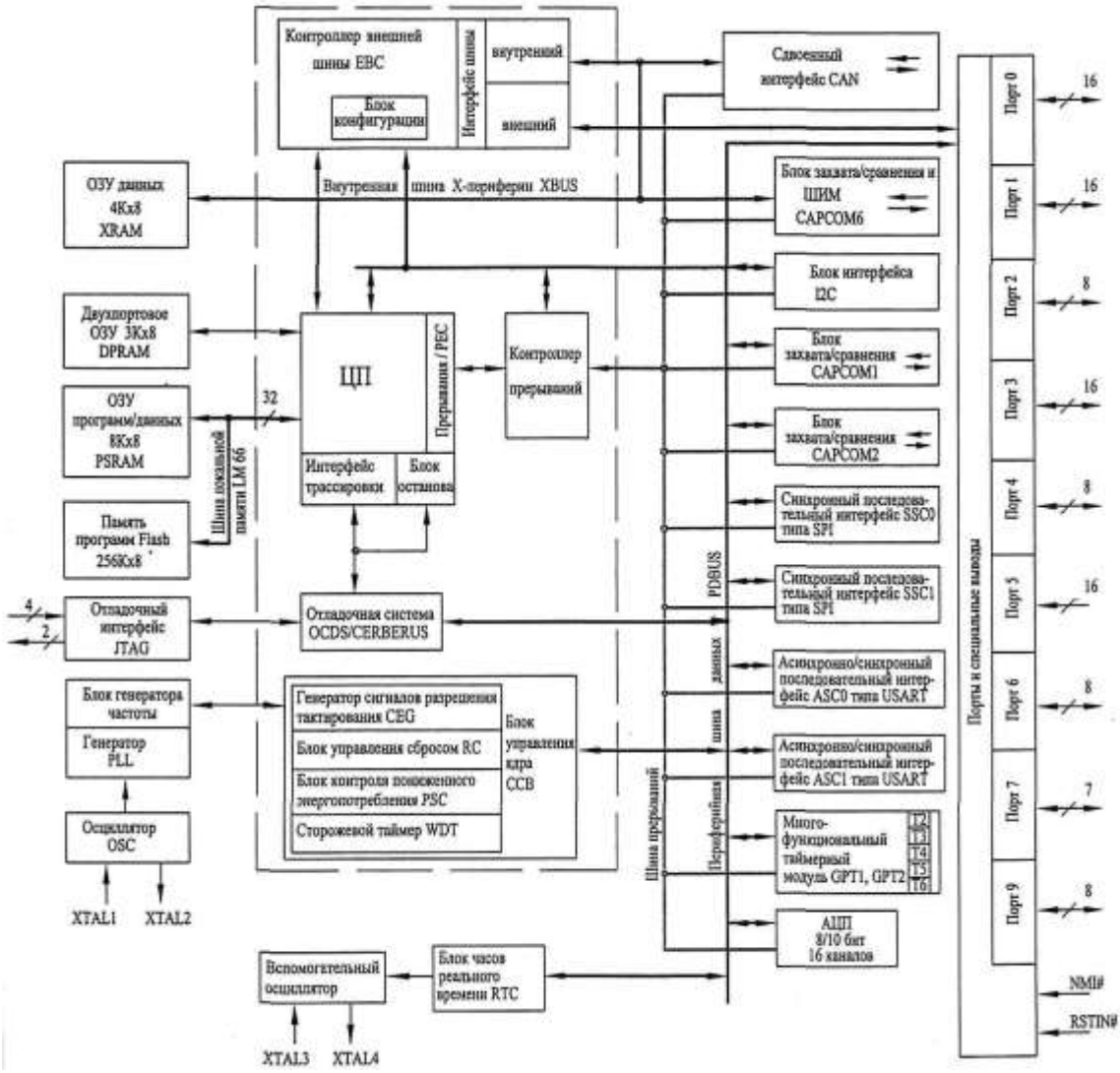


Рисунок 2.1 – Структурная схема ИС 1887ВЕ3Т

2.1 Основная концепция архитектуры центрального процессорного устройства

Ядро центрального процессорного устройства состоит из 4-ступенчатого конвейера команд, 16-разрядного арифметического и логического устройства ALU и регистров специального назначения SFR, также имеется модуль умножения и деления, генератор битовой маски и циклический сдвигатель.

Быстрое выполнение инструкций

Большинство инструкций микроконтроллера выполняется за один машинный цикл, который равен двум тактам частоты ЦПУ. Например, команды сдвига и переноса всегда выполняются за один машинный цикл, независимо от числа сдвигаемых бит.

Команды перехода, умножения и деления, как правило, занимают более одного машинного цикла и, следовательно, требуют оптимизации.

Деление 32-битового числа на 16-битовое занимает 10 тактов частоты ЦПУ, а умножение 16-битового числа на 16-битовое – 5 тактов.

Время выполнения команд значительно сокращается при использовании конвейера команд.

Обработка инструкций на конвейере позволяет существенно сократить время выполнения программ. Использование этой технологии позволяет ядру ЦПУ параллельно обрабатывать разные этапы выполнения четырех последовательно идущих команд. Конвейерная обработка состоит из четырех этапов.

- Выборка. Команда считывается из внутренней или внешней памяти по адресу, определяемому значениями указателя инструкций IP и указателя сегмента кода CSP.
- Декодирование. Выбранная команда декодируется, и считываются необходимые операнды.
- Выполнение. Декодированная команда выполняется.
- Запись результата. Если требуется, производится запись результатов выполнения команды.

Задержки в работе конвейера могут возникать при одновременном доступе к ресурсам по внешней шине на различных этапах конвейера.

Декодер команд

Декодирование команд первично производится на выходах PLA и основано на исходном коде команды. На каждой стадии выполнения команды микрокод в явном виде не используется, для каждой стадии конвейера контрольные сигналы поступают из управляющих регистров, значения которых определяются на стадии декодирования команды. На скорость работы конвейера в первую очередь влияют циклы ожидания при доступе к внешней памяти, при этом сигналы от управляющих регистров остаются без изменения. Многотактные команды выполняются путем вставки команд и при помощи внутренних машинных циклов, которые изменяют необходимые сигналы управления.

8- и 16-разрядное арифметическое и логическое устройство АЛУ

Все стандартные арифметические и логические операции производятся в 16-разрядном АЛУ. При операциях с байтами шестой и седьмой биты результата операции влияют на корректную установку флагов состояний. Большая точность вычислений обеспечивается сигналом «CARRY-IN», поступающим в АЛУ от ранее вычисленной части операции. Внутренние блоки АЛУ, выполняющие операции, оптимизированы для работы с 8-разрядными и 16-разрядными числами. После заполнения конвейера, одна инструкция будет выполняться в течение одного машинного цикла, за исключением инструкций умножения и деления. Передовой алгоритм Бута обеспечивает перемножение четырех битов и деление двух битов за один машинный такт. В этих операциях используются два спаренных 16-разрядных регистра – MDL и MDH, и таким образом эти операции нуждаются в четырех тактах для умножения двух 16-разрядных чисел и 9 тактах – для деления 32-разрядного числа на 16-разрядное число, плюс

дополнительно каждой операции необходим один такт для настройки и регулировки операндов и результатов.

Длинные команды умножения и деления могут быть прерваны во время выполнения, что позволяет добиваться более быстрого ответа на прерывания. Эти команды также позволяют преобразовывать байты данных при знаковом расширении для слова данных. Структура внутренней шины также позволяет передавать байты или слова из периферии или на периферию с учетом требований периферийного оборудования.

Набор флагов автоматически обновляется в PSW после каждой арифметической, логической операции, операции с битами и операции перемещения данных. Наличие флагов позволяет совершать операции условного перехода по специальному условию. Поддержка как арифметики со знаком, так и беззнаковой арифметики обеспечивается с помощью определяемых пользователем условий перехода. Эти флаги также автоматически сохраняются при входе процессора в прерывание или обслуживание ловушки. Все адреса переходов также рассчитываются в АЛУ.

16-разрядный генератор сдвига обеспечивает необходимое количество сдвигов за один такт, также поддерживаются операции сдвига и циклического сдвига.

Расширенная обработка битов и управление периферией

Для обработки битов предназначено большое число команд. Использование этих команд обеспечивает эффективное управление и тестирование периферийного оборудования. В отличие от аналогичных команд других микроконтроллеров, эти команды обеспечивают постоянный доступ к двум операндам в адресуемом побитно пространстве памяти, без необходимости перемещения данных в промежуточные регистры.

Некоторые логические команды доступны как для совершения преобразований слов и байтов, так и поддерживают возможность преобразования битов. Это позволяет пользователям сравнивать и модифицировать биты регистров управления периферийными модулями с помощью одной команды. В систему команд для исключения длинных потоков команд для побитного изменения значений добавлены команды одновременного изменения значения нескольких битов. Эти операции выполняются за один машинный такт.

Широкие возможности для выполнения условных переходов, вызовов подпрограмм и циклических обработок

При выполнении команд перехода после совершения перехода необходим один дополнительный машинный такт, что обусловлено большим числом операций переходов в микроконтроллерных программах. Оптимизация осуществлена путем предварительного расчета адреса во время декодирования команды. Для уменьшения загрузки при обработке циклов были предложены три решения.

Первое решение обеспечивает выполнение перехода за один цикл после первой проверки условия цикла. Таким образом, теряется только один машинный цикл при цикловой обработке. При выходе из цикла не требуется дополнительных машинных циклов. Для цикловых обработок не требуется специальных инструкций и циклы определяются автоматически во время исполнения команд условных переходов.

Второе решение для команд переходов позволяет определять конец таблицы и избегать использования двух команд сравнения, встроенных в цикл. Для этого в конце таблицы помещается наименьшее отрицательное значение и используется условие перехода, если ни это значение, ни сравниваемая величина не были найдены. Цикл прерывается, если одно из двух условий было выполнено. Состояние выхода затем может быть протестировано.

Обеспечивается более гибкое решение, чем имеющаяся в других микроконтроллерах команда «декремент и переход, если равно нулю» («decrement and skip on zero»). С помощью использования команд сравнения и инкремента или декремента, пользователь может сравнивать любые значения. Это позволяет счетчику таблицы охватывать любые

пределы. Это является особым преимуществом системы команд микроконтроллера при табличном поиске.

Сохранение текущего состояния системы автоматически совершается во внутреннем системном стеке, без дополнительного использования команд сохранения состояния до входа и после выхода из прерываний и обслуживания ловушек. Команды работы с подпрограммами записывают значение IP в системный стек при входе в подпрограмму, при этом на выполнение этих команд требуется больше времени, чем для команды безусловного перехода, так как необходимо сохранить состояние системы.

В этих командах также обеспечена поддержка косвенных переходов и вызовов подпрограмм. За счет этого обеспечен множественный режим переходов CASE в ассемблерных макросах и языках высокого уровня.

Сжатый и оптимизированный форматы команд

Для оптимизации производительности конвейера набор команд был разработан таким образом, чтобы включать в себя концепцию RISC. Эта концепция в первую очередь позволяет быстро декодировать команды и операнды при уменьшении загрузки конвейера. Эта концепция, однако, не исключает возможность использования сложных команд, которые необходимы для разработчиков программ. При разработке набора команд преследовались следующие цели:

- Обеспечение емких команд, которые необходимы для выполнения часто используемых команд и потоков повторяющихся команд. Избежать передачи данных во временные регистры и из временных регистров, типа аккумуляторов. Совершение параллельных задач, таких как сохранение режима работы процессора до начала входа в прерывание.
- Избежать сложных схем декодирования путем расположения операндов в согласованных областях для каждой команды, также исключение редко используемых режимов сложной адресации.
- Обеспечение более частого использования команд длиной в слово. Все команды, использующие другой формат, занимают два слова. Это позволяет расположить все команды в границах слов, что упрощает группировку команд. Это также предоставляет возможность использования большего набора команд относительных переходов.

Высокая производительность микроконтроллера 1887ВЕ3Т может эффективно использоваться программистами с помощью высокофункционального набора команд, который включает следующие классы:

- арифметические команды;
- логические команды;
- команды обработки битов;
- команды сравнения и команды, контролирующие метки;
- команды сдвига и циклического сдвига;
- команды приоритетности;
- команды перемещения данных;
- команды системного стека;
- команды перехода и управления подпрограммами;
- команды возврата;
- команды управления системой;
- различные команды.

Возможными операндами могут быть биты, байты и слова. Специальные команды поддерживают преобразование (расширение) байтов в слова. Для операндов предназначены различные варианты режимов: прямой, косвенной и непосредственной адресации.

Система прерываний с программируемым приоритетом

Для возможности обработки большого количества прерываний были включены следующие блоки:

- Контроллер периферийных событий PEC. Используется для разгрузки центрального процессора от ответов на запросы на прерывания. Это исключает дополнительные операции при входе и выходе из прерывания или ловушки, путем однотактного перемещения байта или слова, вызванного запросом на прерывание, между двумя адресами в нулевом сегменте памяти с возможностью увеличения указателя либо адреса источника, либо адреса назначения. Только один такт центрального процессора тратится для обслуживания PEC.

- Многоприоритетный контроллер прерываний. Позволяет расположить все прерывания по приоритетам. Имеется возможность для группировки прерываний, что позволяет предотвращать прерывание друг другом одинаковых по приоритету задач. Для каждого из возможных источников прерываний существует отдельный регистр управления, который включает в себя флаг запроса на прерывание, флаг разрешения прерывания и поле приоритетов прерываний. В случае начала обслуживания прерывания центральным процессором, его можно прервать только посредством запроса более высокого приоритета. Для стандартной обработки прерываний каждый из возможных источников прерываний имеет вектор прерывания.

- Переключаемые банки регистров. Эта особенность позволяет пользователям определять до 16 регистров общего назначения, расположенных в любом месте внутреннего ОЗУ. Специальная однотактная команда позволяет переключать банки регистров с одной задачи на другую.

- Прерываемые многотактные команды. Укороченное время начала прерывания доступно вследствие возможности прерывания многотактных команд (умножения и деления). Время ответа на прерывание находится в пределах от 250 до 500 нс (в случае выполнения внутренней программы). Входы внешних прерываний проверяются каждые 50 нс, что позволяет распознавать очень короткие внешние сигналы.

- Аппаратные ловушки. Микроконтроллер 1887ВЕ3Т обеспечивает великолепный механизм распознавания и обработки некорректных или ошибочных состояний, которые возникают в течение работы (так называемые «аппаратные ловушки»). Аппаратные ловушки вызывают немедленную немаскируемую реакцию системы, которая похожа на стандартное обслуживание прерываний (переход к определенной точке таблицы прерываний). Возможность использования аппаратной ловушки дополнительно указывается в специальном бите регистра флагов ловушек TFR. Аппаратные ловушки прерывают исполнение любой программы, за исключением других ловушек с более высоким приоритетом. В свою очередь обслуживание аппаратных ловушек не может быть обычным путем прервано стандартными прерываниями или PEC прерываниями.

- Программные прерывания. Поддерживаются с помощью команды TRAP, в которой указывается номер прерывания или ловушки.

2.2 Системные ресурсы ядра

Микроконтроллер 1887ВЕ3Т обеспечен большим количеством мощных системных средств, расположенных вокруг ЦПУ.

Область памяти

Пространство памяти микроконтроллера устроено по принципу архитектуры Фон-Неймана. Это означает, что память программного кода, память данных, регистров и портов ввода-вывода организована в одном и том же линейном адресном пространстве, которое может достигать 16 Мбайт. Полное пространство памяти может быть доступно для данных размером байт либо слово. Отдельная часть внутренней памяти может также иметь прямую битовую адресацию.

Внутреннее 16-битное 3 Кбайтное ОЗУ DPRAM обеспечивает быстрый доступ к регистрам общего назначения GPR, пользовательским данным (переменным) и системному стеку. Внутреннее ОЗУ также может использоваться для программного кода. Особенная схема декодирования обеспечивает гибкие пользовательские банки регистров

во внутренней памяти, в то время как остальная часть памяти оптимизируется для пользовательских данных.

Центральный процессор использует банки регистров, состоящие из 16 GPR длиной в байт или слово, которые физически расположены во внутренней DPRAM. Регистр контекстного указателя CPR определяет базовый адрес активного банка регистров, доступного для центрального процессора в настоящее время. Количество банков регистров ограничено доступным объемом DPRAM. Для простой передачи параметров банки регистров могут перекрываться друг с другом.

Системный стек обеспечивает временное хранение данных. Он также расположен в области DPRAM микроконтроллера, и доступ центрального процессора к нему определяется через регистр указателя стека SPR. Содержимое двух независимых регистров общего назначения STKOV и STKUN автоматически сравнивается со значением регистра указателя стека во время каждого доступа к стеку, для определения верхней и нижней границ стека.

Аппаратное определение объема выбранной памяти, расположенной во внутренней памяти, позволяет пользователям производить доступ с помощью прямой или косвенной адресации и получать необходимые данные без использования временных регистров или специальных команд.

Для регистров специальных функций зарезервировано 1024 байта адресного пространства. Стандартная область для регистров специальных функций SFR занимает 512 байт, в то время как область расширенных регистров специальных функций ESFR занимает другие 512 байт. (E)SFR-регистры, используемые для функций управления и контроля различных модулей микроконтроллера, имеют размер в одно слово. Неиспользуемые (E)SFR адреса зарезервированы для будущих моделей с расширенными функциональными возможностями.

Дополнительная локальная память предназначена для хранения кодов и данных. Эта область памяти соединяется с центральным процессором через 32-разрядную шину локальной памяти.

Интерфейс внешней шины

Для создания систем, в которых требуется больший объем памяти, чем расположено внутри микроконтроллера, имеется возможность подключения до 16 Мбайт внешней ОЗУ и/или ПЗУ через интерфейс внешней шины. Встроенный контроллер внешней шины (EBC) позволяет получить доступ к внешней памяти и/или к внешним периферийным устройствам с возможностью широкой настройки.

Может быть запрограммирован как режим работы без внешней памяти (Single Chip) так и один из четырех режимов работы с внешней памятью.

Режимы работы с внешней памятью:

- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, демультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, мультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, демультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, мультиплексная шина.

Режим с демультиплексной шиной порт P1 используется для вывода адресов, и порт P0 используется для ввода-вывода данных. Режим с мультиплексной шиной использует порт P0 для обоих адресов и для ввода-вывода данных.

Для интерфейса внешней шины важными представляются временные характеристики: время обращения к памяти, циклы ожидания, длина ALE и задержка чтения/записи CS# и WR#. Эти характеристики спроектированы легко программируемыми, что позволяет пользователям адаптировать различные типы памяти и/или периферии в широком диапазоне. Кроме того, доступ к разным областям памяти может осуществляться при помощи различных характеристик шины: могут быть сгенерированы до пяти сигналов CS# для сохранения внешней связывающей логики.

Доступ к очень медленной памяти или периферии поддерживается с помощью специальной функции «Ready».

Для приложений, которые нуждаются менее чем в 16 Мбайтах адресного пространства, это адресное пространство может быть ограничено до 1 Мбайта, 256 Кбайт или 64 Кбайт. В этом случае порт P4 выводит четыре или два бита адреса (сегмента) или ни одного. В случае использования 16 Мбайт памяти выводится 8-битовый адрес.

Встроенная в микроконтроллер шина XBus+ является внутренним отображением внешней шины и позволяет организовывать доступ к интегрированным, специально предназначенным для приложений, модулям и встроенной периферии как к внешним компонентам. При этом интерфейс для связи периферии с центральным процессором строго определен.

Включенные в микроконтроллер XRAM и CAN модуль являются представителями X-периферии.

2.3 Периферийные модули микроконтроллера

В микроконтроллере 1887ВЕ3Т произведено четкое разделение периферийных модулей от ядра. Эта структура позволяет производить максимальное количество параллельных операций. Каждый функциональный блок обрабатывает данные независимо, и при этом передача данных и управление осуществляются через общую шину. Для каждого периферийного блока генерируются индивидуальные тактовые сигналы.

Интерфейсы периферийных модулей

В основном, представленная в микроконтроллере периферия имеет два типа интерфейсов. Это интерфейс для связи с ЦПУ и интерфейс для внешнего оборудования. Передача данных между ЦПУ и периферией происходит посредством SFR и прерываний.

Каждый периферийный модуль содержит набор регистров SFR, которые управляют работой блоков и временно хранят результаты работы. Эти SFR расположены либо в стандартной области SFR ($00`FE00_H$...$00`FFFF_H$) или в области дополнительных регистров области ESFR ($00`F000_H$...$00`FlFF_H$). Каждый периферийный модуль имеет набор флагов состояний. Для управления устройствами, подключенными к XBUS, имеется набор регистров области XSFR ($00`E800_H$...$00`EFFE_H$).

Запросы на прерывания генерируются периферийными модулями в ответ на различные события (завершение операции, ошибка и т. д.), возникающие во время их работы.

Для связи с внешним оборудованием используются специальные выводы параллельного интерфейса. Они задействуются в случае использования функций ввода или вывода с внешней периферии. Их также называют альтернативными функциями ввода-вывода для выводов порта, в противоположность функции ввода-вывода основного назначения.

Синхронизация периферии с ЦПУ

Работа ЦПУ и периферии основана на тактовом сигнале ЦПУ FCPU. Внутренний генератор получает сигнал от кристалла или внешнего тактового генератора. Тактовый сигнал периферии f_{PDBUS} не зависит от тактового сигнала ЦПУ FCPU. Во время режима покоя Idle тактовый сигнал ЦПУ перестает вырабатываться, однако периферийное оборудование при этом продолжает свою работу. Периферийные SFR могут быть доступны один раз за такт. Если программа производит запись в SFR, и если одновременно с этим производится изменение значения SFR периферией, то операция программной записи имеет более высокий приоритет.

Примечание – Для исключения разночтения (если не будет оговорено отдельно) фронтом сигнала будет называться переход сигнала с уровня логического 0 на уровень логической 1. Спадом сигнала будет называться переход с уровня логической 1 на уровень логического 0.

Параллельные порты

Микроконтроллер 1887ВЕ3Т имеет 103 канала ввода-вывода, организованных в девять портов ввода-вывода. Все разряды портов являются битадресуемыми и могут индивидуально программироваться на ввод или вывод соответствующими регистрами направления DPx. Установка канала порта на ввод переключает его в третье состояние. После сброса все порты установлены в режим ввода.

Такие функции как контроль драйвера вывода, выбор входных характеристик, температурная компенсация и выбор режима вывода (с открытым стоком или режим push/pull) не поддерживаются. Тем не менее, все эти функции легко могут быть реализованы логикой схемы, поскольку они управляют выводами напрямую и не затрагивают модуль порта.

Сигналы разрешения работы выходных драйверов переключаются асинхронно для быстрого перевода сигналов в неактивные уровни при системном сбросе.

Все каналы ввода-вывода имеют дополнительные функции.

Порт P0 и порт P1 могут использоваться как линии передачи адреса и данных при обращении к внешней памяти, в то же время выводы порта P4 могут использоваться для передачи дополнительного адреса сегмента в приложениях, которым требуется более 64 Кбайт памяти.

Порт P6 может использоваться для передачи дополнительных сигналов арбитража шины и сигналов выбора периферийных устройств.

Все выводы портов, не работающие в альтернативных режимах, могут использоваться как линии общего ввода-вывода.

Блоки таймеров общего назначения

Блоки таймеров общего назначения GPT1 и GPT2 – это многофункциональные структуры таймеров-счетчиков, которые могут использоваться для формирования временных интервалов, измерения длительности внешних импульсов, вывода импульсов и т. п.

В блоке GPT1 содержится три таймера T2, T3, T4 с возможностью каскадирования и захвата/перезагрузки и максимальной тактовой частотой работы $f_{PER}/4$. Блок GPT2 состоит из двух таймеров T5, T6 с максимальной тактовой частотой работы $f_{PER}/2$ и специального регистра для захвата/перезагрузки CAPREL. Каждый таймер может работать независимо от остальных в различных режимах. Максимальное разрешение таймеров блока GPT1 составляет 500 нс, а таймеров блока GPT2 – 250 нс.

Предусмотрена возможность изменения направления счета (инкрементирование и декрементирование) каждого таймера как программно, так и аппаратно, сигналом на соответствующем входе таймера TxEUD.

Таймеры обоих блоков могут быть сконфигурированы индивидуально в один из трех основных режимов работы – таймера, счетчика или таймера, управляемого внешним сигналом.

Таймеры T3 и T6 имеют по выходному триггеру T3OTL и T6OTL, который изменяет свое состояние при каждом переполнении таймера. Изменение состояния этих триггеров может выводиться через соответствующий канал порта P3 или использоваться для каскадирования с другими таймерами, чтобы получить 32-разрядный таймер или 33-разрядный счетчик.

Функции захвата и перезагрузки значений таймеров T3 и T6 управляются внешними сигналами или состоянием соответствующего триггера. Для обоих вариантов возможен выбор между фронтом, спадом или любым переходом.

Часы реального времени

В состав микроконтроллера входит блок часов реального времени RTC, который является независимым 48-разрядным таймером и используется:

- для отсчета текущей даты и времени;
- для периодического формирования запроса на прерывание;

- для измерения длительных интервалов времени.

Тактовый сигнал для часов RTC формируется непосредственно из входного тактового сигнала, поэтому счет продолжается даже в режиме максимального энергосбережения Power Down, в котором отключаются ЦПУ и вся внутренняя периферия.

Блок аналого-цифрового преобразователя

Измерение аналоговых сигналов в микроконтроллере 1887ВЕ3Т осуществляется в блоке 10-разрядного аналого-цифрового преобразователя АЦП с 16-ю мультиплексируемыми входами и цепью выборки и преобразования.

Блок АЦП использует метод последовательного приближения для преобразования напряжения в цифровой код. Время выборки (для зарядки входных внутренних конденсаторов) и время преобразования могут программироваться, давая возможность подстраиваться под характеристики подключенных внешних устройств.

Для защиты результата преобразования от перезаписи и детектирования перезаписи предусмотрены следующие механизмы:

- если результат последнего преобразования не был прочитан до завершения следующего преобразования, то будет сформирован запрос на прерывание;
- следующее преобразование может быть приостановлено до тех пор, пока результат текущего преобразования не будет прочитан.

Блок АЦП дает возможность преобразования в четырех различных режимах:

- режим однократного преобразования для одного выбранного канала;
- режим повторяющегося преобразования для одного выбранного канала;
- режим однократного преобразования для выбранной группы каналов;
- режим повторяющегося преобразования для выбранной группы каналов.

В режимах преобразования сигналов для группы каналов возможна вставка преобразования одного любого канала вне очереди.

Блоки захвата/сравнения

Два блока захвата/сравнения CAPCOM1 и CAPCOM2 обеспечивают управление временными последовательностями с помощью 32 каналов (по 16 каналов на блок). Блоки CAPCOM1 и CAPCOM2 обычно используются для высокоскоростного формирования сигналов сложной формы, широтно-импульсной модуляции и регистрации моментов времени при возникновении определенных условий.

Каждый блок CAPCOM состоит из двух 16-разрядных таймеров: T0 и T1 – CAPCOM1, T7 и T8 – CAPCOM2, причем каждый таймер имеет свой регистр перезагрузки TxREL. Таймеры блоков ведут счет только на увеличение.

Входным сигналом для таймеров может быть тактовый сигнал ЦПУ, частота которого уменьшена в задаваемое число раз, или сигнал переполнения таймера T6 блока GPT2. Это обеспечивает широкий диапазон возможных разрешений по времени и длительностей временных интервалов, формируемых каналами. Кроме того, таймеры T0 и T7 могут работать в режиме счетчиков внешних тактовых импульсов.

Каждый из блоков CAPCOM включает в себя шестнадцать регистров захвата/сравнения: в CAPCOM1 это регистры CC0, …, CC15, в CAPCOM2 – регистры CC16, …, CC31. Каждый регистр захвата/сравнения может быть запрограммирован для работы с одним из двух таймеров блока, которому он принадлежит. Помимо этого, с каждым регистром связан соответствующий канал порта, который используется как вход (в режиме захвата) или как выход (в режиме сравнения).

Каналом захвата/сравнения является объединение при совместной работе регистра захвата/сравнения и ассоциированного с ним канала ввода-вывода. Канал захвата/сравнения, настроенный в режим захвата, будет сохранять текущее содержимое ассоциированного с ним таймера в свой регистр захвата/сравнения по приходу внешнего импульса (по фронту, по спаду или по любому переходу). Кроме того, будет сформирован запрос на прерывание. Если канал захвата/сравнения настроен на режим сравнения, то при

совпадении значения регистра канала со значением ассоциированного с ним таймера будет выдаваться сигнал через соответствующий канал порта и/или выставляться запрос на прерывание. Каждый канал захвата/сравнения имеет отдельный вектор в таблице прерываний.

Блок широтно-импульсной модуляции

Блок CAPCOM6 состоит из блоков таймеров T12 и T13, каждый со своими регистрами захвата/сравнения: T12 с тремя каналами захвата/сравнения и T13 с одним каналом сравнения. Каналы таймера T12 могут независимо генерировать ШИМ сигналы или считывать значения триггеров захвата и работать в двух режимах – «пилы» и «симметричном».

Специальные режимы работы позволяют также осуществлять управление бесщеточными ДПТ с датчиками Холла или контролем обратной ЭДС.

Выходные сигналы блока CAPCOM6 могут программно инвертироваться.

Асинхронные/синхронные последовательные каналы и высокоскоростные синхронные последовательные каналы

Последовательная связь с другими микроконтроллерами, процессорами или внешними периферийными устройствами обеспечивается двумя последовательными интерфейсами – асинхронными/синхронными последовательными каналами ASC0, ASC1 и синхронными последовательными каналами SSC0, SSC1. Каналы ASC0 и ASC1 идентичны, поэтому далее рассматривается канал ASC0. Каналы SSC0 и SSC1 также идентичны, поэтому далее рассматривается канал SSC0.

Интерфейс ASC0 поддерживает полнодуплексный асинхронный режим работы на скорости приема/передачи до 1,25 Мбод и полудуплексный синхронный режим на скорости приема/передачи до 2,5 Мбод, с учетом того, что f_{PER} = 20 МГц.

Интерфейс SSC0 работает в полнодуплексном и полудуплексном синхронном режиме на скорости приема/передачи до 10 Мбод в master-режиме и до 5 Мбод в slave-режиме, с учетом того, что f_{PER} = 20 МГц.

Каждый модуль последовательного интерфейса имеет собственный генератор тактового сигнала, который задает необходимую скорость обмена данными, в том числе стандартную скорость.

Для передачи, приема и детектирования ошибок отведено по три прерывания для канала SSC0 и канала ASC0.

Модуль ASC0 использует для передачи и приема в асинхронном режиме 8- и 9-разрядные пакеты (frame), которые начинаются со стартового бита и заканчиваются одним или двумя стоповыми битами. Бит четности для ASC0 автоматически вычисляется при передаче и проверяется во время приема. Для обмена в мультипроцессорной системе используется механизм различия адресных байтов от байтов данных. В синхронном режиме канал ASC0 передает или принимает 8-разрядные пакеты синхронно с импульсами, вырабатываемыми внутренним тактовым генератором.

Модуль SSC0 передает или принимает данные пакетами от двух до шестнадцати разрядов синхронно с тактовыми импульсами, которые выводятся внутренним тактовым генератором в режиме ведущего (master). В режиме ведомого (slave) используются тактовые импульсы внешнего устройства. Обмен информацией начинается с самого младшего разряда данных или самого старшего, в отличие от канала ASC0, который передает и принимает данные, начиная всегда с самого младшего разряда данных.

Для обеспечения надежной связи используется механизм детектирования ошибок. Детектирование ошибок позволяет выявлять пакеты данных с пропущенными стоповыми битами. Ошибка четности для ASC0, а также ошибка скорости для SSC0 автоматически детектируются во время работы, после чего формируется соответствующий запрос на прерывание. Ошибки переполнения для обоих модулей детектируются в случае, если последние принятые данные не были прочитаны из буфера до записи следующих пришедших данных.

Блок I2C

Блок I2C обеспечивает полную поддержку двухпроводного последовательного синхронного интерфейса I2C/SMBus. Результат такой совместимости – легкое соединение со многими запоминающими устройствами и устройствами ввода-вывода, включая EEPROM, SRAM, счетчики, АЦП, ЦАП, периферийные устройства.

Блок I2C поддерживает стандартный (Standard), скоростной F/S и высокоскоростной Hs режимы, может быть запрограммирован на работу как в режиме ведущего (master) так и ведомого (slave).

Модуль CAN интерфейса

Микроконтроллер 1887ВЕ3Т содержит в своем составе два независимых (с возможностью взаимодействия через шлюзы и FIFO структуры) CAN узла с поддержкой CAN интерфейса, каждый со своей парой выводов для подключения к одной или разным CAN шинам. Два CAN узла вместе составляют в целом модуль CAN интерфейса.

Интегрированные CAN узлы передают и принимают сообщения в соответствии со спецификацией версии 2.0B, т. е. обеспечивается возможность передачи и приема сообщений одновременно со стандартным 11-разрядным и расширенным 29-разрядным идентификаторами. В обоих режимах используются маски для фильтрации принимаемых сообщений. Каждая из областей сообщений (message object) имеет собственный идентификатор и может обрабатывать данные размером до 8 байт включительно, причем независимо от остальных областей сообщений. Несмотря на то, что CAN узлы независимы, они имеют общую память сообщений, состоящую из 32 областей сообщений. Любая область сообщения может быть выделена для работы с любым из двух CAN узлов.

Передаваемые сообщения имеют идентификатор, который в CAN сети является уникальным. При передаче узел сети получает сообщение и проверяет идентификатор. Если сообщение имеет отношение к данному узлу (не только простое совпадение, но и поразрядная проверка с использованием маски), то оно обрабатывается, в противном случае – игнорируется. CAN контроллер может обрабатывать одновременно несколько сообщений с различными идентификаторами. Таким образом, можно легко организовать несколько «виртуальных» каналов обмена информацией с различными устройствами.

Каждый CAN узел обеспечивает работу в сети в тяжелых условиях (по стандарту ISO11898) в следующих случаях:

- любой из трех проводов шины оборван;
- любой провод закорочен на питание;
- любой провод закорочен на общий провод.

При обрыве двух проводов часть функций основной системы может быть реализована в каждой из подсистем, созданных обрывом.

Принятая в CAN сети схема передачи сообщений обеспечивает широкие возможности при создании и модернизации систем, а также при подключении новых устройств. Новые устройства могут добавляться к сети без изменения уже существующих программных средств, если их подключение не приводит к превышению нагрузочной способности и максимальной длины шины. При этом новые сетевые устройства способны обмениваться информацией между собой, не нарушая работоспособность старой системы, если в протоколе обмена были использованы новые идентификаторы.

В -сети существует возможность одновременной передачи сообщений сразу нескольким устройствам. Эта особенность позволяет передавать по ней широковещательные сообщения и сообщения для синхронизации.

CAN использует 6-ступенчатый механизм исправления ошибок:

- постоянный контроль передаваемой и принимаемой информации;
- использование согласующего бита;
- контроль служебной информации;
- контроль подтверждения приема;

- циклический контроль по избыточности CRC;

- автоматический повторный запрос.

CAN узлы имеют программируемую скорость обмена данными до 1 Мбит/с. Для этого используются каналы портов P4, P7 и P9. Причем каждый CAN узел может принимать сообщения с любой из 7 отведенных под эти функции линий портов. Для передачи сообщений могут быть задействованы до 8 линий: по 4 линии для каждого из CAN узлов.

В микроконтроллере 1887ВЕ3Т реализована возможность передачи логического «И» сообщений CAN узлов.

Блок управления выходом Wakeup из режимов Idle или Sleep

Циклический выход из режима холостого хода Idle или режима ожидания Sleep объединяет значительное уменьшение потребления энергии в режиме Idle/Sleep и высокий уровень готовности системы к возвращению в рабочий режим. Внешние сигналы и события могут сканироваться на более низкой скорости при периодической активности ЦПУ и выбранных периферийных устройств, которые возвращаются к экономичному режиму энергопотребления после кратковременного нормального режима работы. Все это значительно уменьшает среднюю потребляемую мощность.

Режим Idle/Sleep также может быть прерван сигналами внешних прерываний.

Система отладки OCDS и JTAG

Блок OCDS осуществляет поддержку отладчику эмулировать возможности и помогает в отладке прикладных программ. Основные параметры:

- эмуляция в реальном масштабе времени;

- расширенная возможность запуска отладки от аппаратных источников отладочных событий: указателя команд, данных или адреса при чтении или записи, внешних выводов, и т. д.;

- поддержка программной остановки отладки (break);

- простой режим монитора или отладка с помощью команд, вводимых через JTAG интерфейс.

Блок OCDS является блоком, который управляется отладчиком через группу регистров, доступных посредством JTAG интерфейса через блок JTAG. Блок OCDS также принимает информацию (такие как IP, данные, состояние) от ядра, чтобы контролировать деятельность и запуск отладочных событий и взаимодействует с ядром через интерфейс остановки программы (break), чтобы приостановить выполнение программы и через интерфейс ввода позволить выполнение процесса отладки.

Блок управления ядра CCB

Блок CCB – основной управляющий блок, выполняющий все основные задачи управления и специфические задачи. Блок CCB объединяет в себе несколько блоков:

- **Управление сбросом RC**

Блок RC управляет функцией сброса и осуществляет три типа сброса:

- аппаратный сброс: система немедленно (асинхронно по отношению к тактовой частоте ЦПУ) переходит в состояние сброса;

- программный сброс: синхронно по отношению к тактовой частоте ЦПУ;

- сброс сторожевого таймера синхронно по отношению к тактовой частоте ЦПУ.

- **Контроль пониженного энергопотребления PSC**

Режимы холостого хода Idle и пониженного энергопотребления Power Down выполняются блоком управления режимом энергопотребления PSC.

- **Генератор сигналов разрешения тактирования CEG**

Основным и единственным сигналом тактирования подсистемы является сигнал Master Clock (subs_clk_i), на основе которого генератор CEG формирует сигналы разрешения тактирования, которые используются различными блоками подсистемы.

На основе сигнала Master Clock формируются три тактовых сигнала:

- тактовый сигнал ЦПУ;

- инверсный тактовый сигнал ЦПУ;
- тактовый сигнал периферии.

Тактовый сигнал периферии совпадает по частоте с тактовым сигналом ЦПУ, но имеет фазовый сдвиг на 180° по отношению к нему. Это предусмотрено для того, чтобы иметь возможность запуска механизма контроллера внешней шины по двум (переднему и заднему) фронтам тактового сигнала. Протокол шины локальной памяти LM-66 также основан на этом механизме тактирования. Все блоки подсистемы используют два (прямой и инверсный) тактовых сигнала ЦПУ, которые доступны только внутри подсистемы.

Для пользователя доступны сигналы разрешения, на основе которых формируются тактовые частоты всех блоков микроконтроллера.

Тактовая частота периферии ограничена, не может превышать 50 МГц и может равняться тактовой частоте ЦПУ или 1/2 тактовой частоты ЦПУ.

Для формирования всех необходимых сигналов, тактовый сигнал Master Clock, подаваемый на генератор сигналов разрешения тактирования, должен в два раза превышать тактовый сигнал ЦПУ по частоте.

Сторожевой таймер WDT

Сторожевой таймер WDT представляет собой специальный защитный механизм, который предотвращает неправильное функционирование микроконтроллера в течение длительного периода времени.

Сторожевой таймер всегда запущен после сброса и может быть запрещен только до выполнения инструкции конца инициализации EINIT. Программа должна быть составлена таким образом, чтобы обрабатывать сторожевой таймер до переполнения. В отсутствие обработки (при программном или аппаратном сбое) сторожевой таймер не будет перезагружаться специальной инструкцией SRVWDT и по истечении заданного времени переполнится, в результате чего произойдет аппаратный сброс микроконтроллера. При этом выходной сигнал сброса RSTOUT# будет также переведен на уровень логического 0, что приведет к сбросу подключенных к микроконтроллеру внешних устройств.

Сторожевой таймер представляет собой два каскадированных 8-разрядных таймера. Входным сигналом для сторожевого таймера является тактовый сигнал ЦПУ, частота которого поделена на 2, 4, 128 или 256. При каждой перезагрузке сторожевого таймера с помощью специальной инструкции SRVWDT происходит заполнение старшего 8-разрядного таймера заданным значением и обнуление младшего.

Генератор тактовых сигналов

Действиями аппаратных средств микроконтроллера и периферийных устройств управляют тактовые сигналы, формируемые блоком генератора тактовых сигналов CGU, состоящим из трех блоков: осциллятора, блока фазовой автоподстройки частоты PLL и схемой распределения тактового сигнала.

Блоком генератора основных тактовых сигналов формируется четыре различных тактовых сигнала:

- FCPU – тактовый сигнал ядра микроконтроллера;
- FPER – тактовый сигнал периферийных модулей, подключенных к шине PDBUS;
- FXPER – тактовый сигнал периферийных модулей, подключенных к шине XBUS;
- FOUT – выходной тактовый сигнал, служащий для тактирования внешних устройств.

Кроме этого дополнительной схемой формируется медленный тактовый сигнал FRTC, поступающий на модуль часов реального времени RTC и позволяющий модулю работать независимо от описанных выше тактовых сигналов.

3 Блок центрального процессорного устройства

Основным предназначением блока центрального процессорного устройства ЦПУ является выбор и декодирование команд, снабжение операндами АЛУ, выполнение операций над этими операндами в АЛУ и сохранение ранее вычисленных результатов. Так как ЦПУ является основным компонентом микроконтроллера, на него также оказывают влияние определенные действия периферийной подсистемы, см. рисунок 3.1.

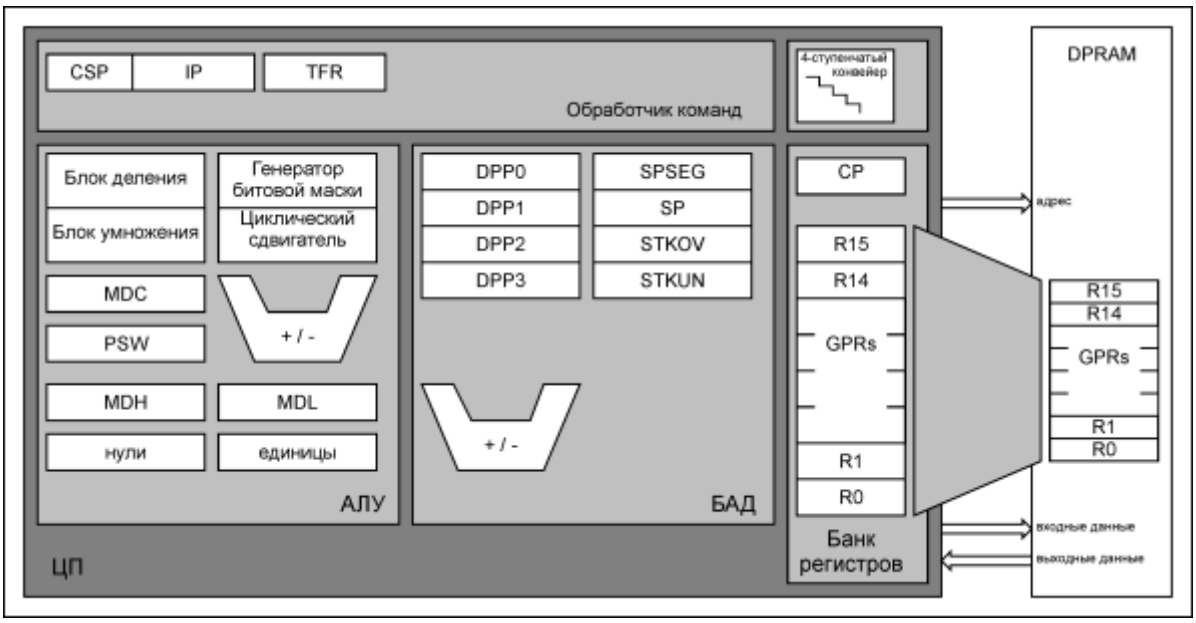


Рисунок 3.1 – Архитектура ЦПУ

Блок центрального процессора включает в свой состав четыре основных блока. Все эти блоки оптимизированы для достижения максимальной производительности и гибкости системы.

Обработчик команд

Обработчик команд – блок, включающий в себя два подблока:

- Блок выборки команд IFU выполняет:
  - выборку команд (высокоскоростную);
  - управление FIFO (первым прибыл – первым обслужен);
  - высокоэффективную обработку ветвлений, вызовов и циклов с использованием алгоритмов прогнозирования команд, что значительно увеличивает скорость вычислений.
- Обработчик вставки/исключения команд выполняет:
  - обработку запросов прерываний;
  - обработку аппаратных сбоев и отказов.

Конвейер команд IPIP

Конвейер команд IPIP – 4-ступенчатый конвейер.

Блок адреса и данных БАД

Блок адреса и данных БАД – 16-разрядный арифметический блок для формирования адреса.

Арифметическое и логическое устройство АЛУ

Арифметическое и логическое устройство АЛУ включает в себя несколько подблоков:

- 8- и 16-разрядный арифметический блок;
- 16-разрядный циклический сдвигатель;

- блок умножения;
- блок деления;
- 8- и 16-разрядный блок логики;
- блок оперирования битами.

3.1 Формат описания регистров

В составе микроконтроллера имеются регистры, которые описаны в соответствующих разделах настоящего руководства. На рисунках 3.2, 3.3 и 3.4 показаны примеры представления 32-, 16- и 8-разрядного регистров и указания к их интерпретации.

На рисунках 3.2, 3.3 и 3.4 приняты следующие обозначения:

REG_NAME	– название регистра;
XSFR / SFR / ESFR	– область памяти, в которой расположен регистр;
A16H	– 16-разрядный адрес регистра;
A8H	– 8-разрядный адрес регистра;
****H / **H	– значение [по умолчанию] регистра после сброса (RESET): 0 / 1 – значение бита определено после сброса; U / X – значение бита не определено после сброса;
бит X (X = A/B/C/D/E)	– краткое название бита;
битовое поле X (X = A/B/C/F)	– краткое название битового поля;
з	– бит доступен для программной записи;
ч	– бит доступен для программного чтения;
ап	– бит доступен для аппаратной записи;
0 / –	– бит (битовое поле) не используется или зарезервирован(о). При чтении всегда возвращается «0». Запись «1» в этот бит (битовое поле) запрещена!

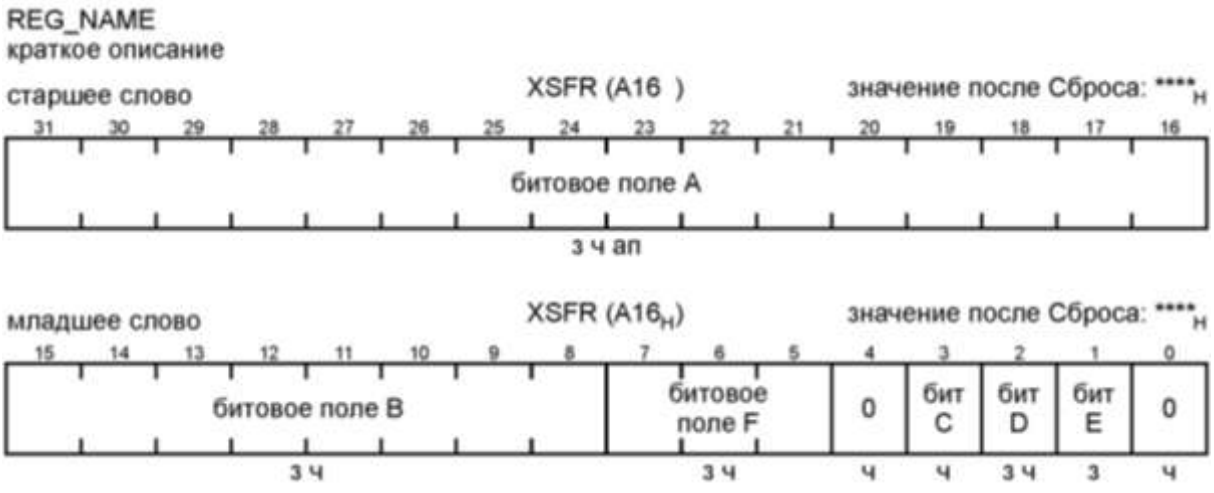


Рисунок 3.2 – Пример представления формата 32-разрядного регистра

Рисунок 3.3 – Пример представления формата 16-разрядного регистра

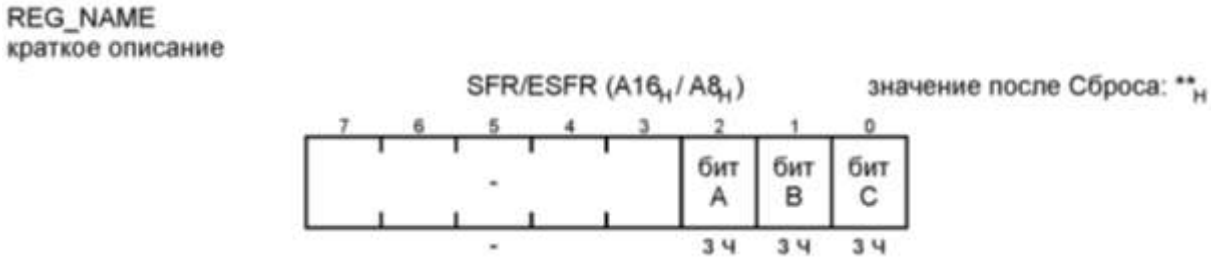


Рисунок 3.4 – Пример представления формата 8-разрядного регистра

Полное побитное описание каждого регистра сведено в таблицу, показанную на рисунке 3.5.

Поле	Биты	Тип	Описание
бит X	m	Тип	Полное название бита Описание, функции, варианты принимаемых значений
битовое поле X	m - n	Тип	Полное название битового поля Описание, функции, варианты принимаемых значений

REG_NAME – название регистра;
бит X – краткое название бита;
битовое поле X – краткое название битового поля;
m – номер бита;
m – n – диапазон битового поля:
m – старший (конечный) бит диапазона;
n – младший (начальный) бит диапазона;
тип – тип бита:
чтение – бит доступен для программного чтения;
запись – бит доступен для программной записи;
аппаратное влияние – бит доступен для аппаратной записи.

Рисунок 3.5 – Описание функционального назначения полей регистра REG_NAME

3.2 Регистры ЦПУ специального назначения SFR

Ядро ЦПУ нуждается в наборе регистров SFR для хранения информации о состоянии системы и необходимых АЛУ констант, для адресации регистров, для управления системой и конфигурацией шины, а также для операций умножения и деления в АЛУ, сегментации памяти кода, разбития памяти данных на страницы и доступа к регистрам основного назначения и системному стеку.

Механизм доступа к этим SFR в ядре ЦПУ идентичен механизму доступа для других SFR. Так как все SFR могут управляться посредством любых команд, действительных для адресного пространства SFR, то отпадает необходимость в создании набора специальных системных команд.

Следует заметить, что к некоторым SFR ядра ЦПУ доступ ограничен, что необходимо для обеспечения выполнения операций процессором. Ни при каком условии нельзя осуществить доступ к указателю команд IP и указателю сегмента кода CSP. Их значения могут быть только косвенно изменены посредством команд перехода.

Регистр слова состояния процессора PSW, указатель стека SP и регистр управления умножением/делением MDC могут быть изменены не только с помощью прямых команд, но и посредством выполнения специальных команд ЦПУ.

Все SFR доступны пословно или побайтно (некоторые – побитно). Чтение байта регистра стандартная операция. В то же время, любые операции программной записи одного байта в SFR очищают значение второго байта этого SFR. Не используемые (зарезервированные) биты SFR не могут быть изменены и всегда выдают значение «0» при попытке чтения.

3.3 Выборка команд и контроль выполнения

В среднем, за каждый машинный цикл (состоит из двух тактов рабочей частоты) микроконтроллер может выбирать одну 32-разрядную или две 16-разрядные команды посредством шины локальной памяти (LM-шина) из внутренней локальной памяти. Такой механизм поддерживает непрерывную обработку команд.

Режимы адресации при переходах

Конечный адрес и сегмент перехода или вызов команды может быть определен одним из нескольких способов. Регистр указателя команд IP может быть обновлен относительным, абсолютным или косвенным способом. Регистр указателя сегмента CSP может быть обновлен только абсолютным значением. Специальные режимы реализуются для адресации векторов переходов прерываний, расположенных в нижней части нулевого сегмента. Способы адресации при переходах представлены в таблице 3.1.

Таблица 3.1 – Способы адресации при переходах

Мнемокод	Адрес перехода	Сегмент перехода	Допустимый диапазон адресов
caddr	(IP) = caddr	–	caddr = 0000_H … $FFFE_H$
rel	(IP) = (IP) + 2 * rel (IP) = (IP) + 2 * (rel# + 1)	–	rel = 00_H…$7F_H$ rel = 80_H…FF_H
[Rw]	(IP) = (Rw)	–	Rww = 0…15
seg	–	(CSP) = seg	seg = 0…255
#trap7	(IP) = 0000_H + 4 * trap7	(CSP) = 0000_H	trap7 = 00_H…$7F_H$

Примечание –

caddr: Определяет абсолютный 16-битовый адрес кода в пределах текущего сегмента. Переходы не могут быть применены к нечетным адресам. Таким образом, самый младший бит в caddr всегда должен быть равен «0». В противном случае возникнет состояние аппаратной ловушки.

rel: Определяет знаковое 8-битовое начало слова адреса относительно текущего содержимого указателя команд IP, который указывает на команду, ледующую за командой перехода. В зависимости от расположения начала диапазона адресов, возможны как переходы «вверх» (rel = 00_H…$7F_H$), так и переходы «вниз» (rel = 80_H…FF_H). Команда перехода может выполняться повторно, если rel = –1 (FF_H) для 16-разрядных команд перехода или если rel = –2 (FE_H) для 32-разрядных команд перехода.

Окончание таблицы 3.1

[Rw]:	Косвенное определение адреса перехода, который определяется содержимым GPR (здесь значение Rw). В отличие от косвенных адресов данных, косвенно определенные адреса кода не вычисляются через дополнительные регистры указателей (такие как DPP). Переходы не могут быть применены к нечетным адресам. Таким образом, самый младший бит в caddr всегда должен быть равен «0», в противном случае возникнет состояние аппаратной ловушки.
seg:	Определяет абсолютный номер сегмента. Микросхема 1887ВЕ3Т поддерживает 256 различных сегментов, и в связи с этим достаточно 8 младших бит в seg для обновления регистра CSP.
#trap7:	Определяет уникальный номер прерывания перехода для подпрограммы обработки прерываний при помощи таблицы векторов прерываний, см. приложение А. Номер прерывания от 00_H до $7F_H$ может быть определен для доступа к любой 32-разрядной позиции кода в пределах диапазона адресов $00`0000_H$ – $00`01FC_H$ в нулевом сегменте (т.е. таблице векторов прерываний).

Конвейерная обработка команд

Обработка команд производится на 4-ступенчатом конвейере. Каждая ступень имеет свой набор выполняемых операций.

Конвейер микроконтроллера состоит из следующих этапов:

- выборка;
- декодирование;
- выполнение;
- запись результата.

На этапе выборки производится чтение команд из внутреннего ПЗУ, ОЗУ или внешней памяти в соответствии с адресом, формируемым из указателя команд IP и указателя сегмента CSP. Далее производится декодирование команд и, если это необходимо, вычисление адреса операнда и выборка его из памяти. Для всех команд, которые неявно производят доступ к системному стеку, значение указателя вершины стека либо уменьшается, либо увеличивается. Команды перехода производят запись адреса перехода в регистры IP и CSP (для команд условного перехода только в том случае, если указанное условие перехода является истинным). После декодирования, команда передается на этап выполнения, на котором осуществляется исполнение требуемой операции с предварительно выбранными операндами. Флаги слова состояния ЦПУ (регистр PSW) обновляются в соответствии с результатом операции. Также на этом этапе производится запись в области регистров SFR или ESFR и выполнение записи с инкрементом/декрементом в регистры общего назначения, значение которых используется для косвенной адресации. Результат выполнения, если требуется, записывается во внутреннюю или внешнюю память на последнем этапе конвейера.

Особенностью конвейера являются так называемые инжектируемые команды (injected instruction), которые автоматически формируются самим ЦПУ для правильного функционирования конвейера. Инжектированные команды необходимы для обработки запросов на прерывание, выполнения переходов и завершения команд, которые не могут быть выполнены за один машинный цикл. Эти инжектируемые команды автоматически вставляются на этапе декодирования и проходят оставшиеся этапы конвейера, как и остальные команды. Управление инжектируемыми командами осуществляется только аппаратно и недоступно для программы. Поэтому необходимо принимать в рассмотрение время, необходимое для их выполнения. Это особенно важно для определения времени реакции на внешний запрос.

Выполнение команд на конвейере

Каждая команда проходит все этапы конвейера. Так как обработка одной команды на каждом этапе конвейера занимает минимум один машинный цикл, то для полного выполнения команд требуется минимум четыре машинных цикла. В микроконтроллере 1887ВЕ3Т конвейер позволяет параллельно обрабатывать до четырех команд. Благодаря этому достигается выполнение большинства команд в среднем за один машинный цикл, но только после заполнения конвейера. Другими словами, после сброса первая команда выполняется минимум за четыре машинных цикла, а остальные, когда конвейер уже заполнится, будут выполняться минимум за один машинный цикл, см. рисунок 3.6.

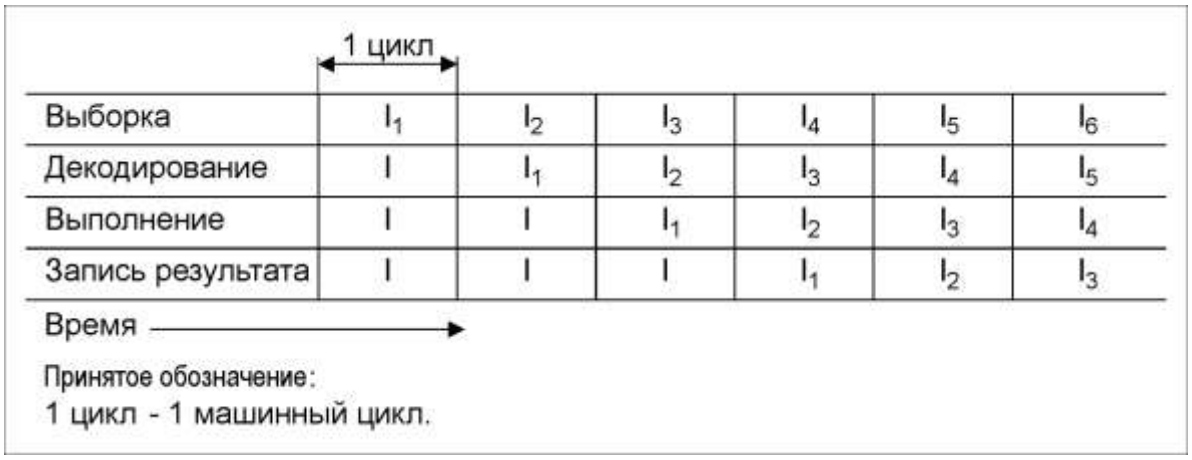


Рисунок 3.6 – Последовательное выполнение команд на конвейере

Обработка команд перехода

Конвейерная обработка позволяет весьма существенно ускорить выполнение программ. Однако в случае, если необходимо выполнить команду перехода, то следующая выбранная команда уже не годится для декодирования, т. к. является недействительной. Таким образом, необходим один дополнительный машинный цикл для выборки команды по адресу перехода. Чтобы недействительная команда не выполнялась, она автоматически заменяется системной инжектированной командой, см. рисунок 3.7.

Для условных переходов, если условие является ложным, переход осуществляться не будет, и в этом случае не требуется дополнительных машинных циклов. Поэтому следующая выбранная команда не будет заменяться инжектированной, а будет помещена на этап декодирования.

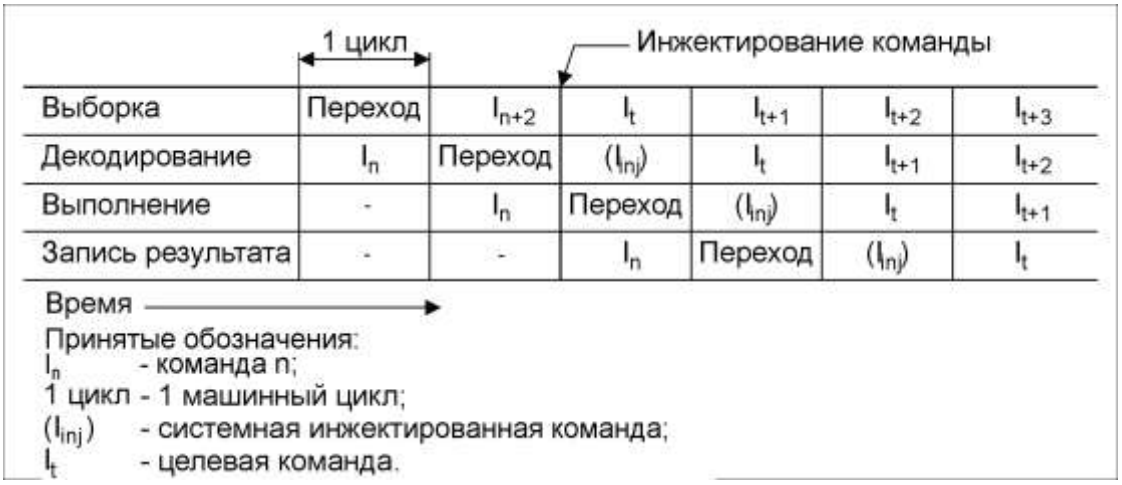


Рисунок 3.7 – Выполнение команд перехода

Кэш для инструкций перехода

Микроконтроллер 1887ВЕ3Т содержит специальный кэш для оптимизации выполнения повторяющихся в цикле команд переходов.

В большинстве случаев повторяющиеся в цикле команды перехода требуют для выполнения всего один машинный такт. Такая производительность достигается за счет использования следующего механизма. Каждый раз, когда команда перехода декодируется на конвейере в первый раз и условие перехода (бит в памяти или флаг АЛУ) является истинным, производится обычная выборка команды по адресу перехода. При этом для перехода требуется один дополнительный машинный цикл. Параллельно с переходом для команд JMPA, JMPR, JB, JBC, JNB и JNBS происходит сохранение в кэш выбранной по адресу перехода команды. Сохранение в кэш позволяет при последующем или повторяющемся в цикле переходе не тратить дополнительный машинный цикл на выборку команды по адресу перехода, а инжектировать ее на этап декодирования из кэш, см. рисунок 3.8.

Кэш команд очищается после выполнения межсегментных команд перехода JMPS, CALLS, RETS, TRAP, RETI либо во время прерывания после обработки поступившего запроса.

Рисунок 3.8 – Выполнение команд перехода с сохранением в кэш

Команда ATOMIC и EXT-команды (расширенные)

Команда ATOMIC и EXT-команды (EXTR, EXTP, EXTS, EXTPR, EXTSR) запрещают стандартные прерывания А-класса, ловушки и прерывания PEC до завершения следующей последовательности команд. Количество команд в последовательности может варьироваться от одного до четырех. Номер команды кодируется 2-битовой константой #irang2 и может принимать значения от 0 до 3. EXT-команды, кроме того, изменяют механизм адресации во время обработки последовательности.

Команда ATOMIC и EXT-команды активируются незамедлительно и потому не требуются дополнительные команды NOP. Все команды для выполнения требуют нескольких циклов или состояния ожидания. Команда ATOMIC и EXT-команды могут использоваться с любыми типами команд.

Примечания

1 Если во время выполнения команды ATOMIC или EXT-команд возникает прерывание В-класса, выполнение последовательности команд приостанавливается до конца выполнения подпрограммы обработки прерываний. Оставшиеся команды приостановленной последовательности команд, после возвращения из подпрограммы обработки прерываний, будут выполняться в нормальном режиме.

2 При использовании команды ATOMIC и EXT-команд требуется особая осторожность. Для контроля длины последовательности команд, (т. е. использования ATOMIC и EXT-команд в последовательности команд) есть только один счетчик, который перезагружает счетчик команд.

Адресация посредством указателей сегмента и команд

Микроконтроллер 1887ВЕ3Т имеет 16 Мбайт адресуемого пространства. Это пространство состоит из 256 сегментов по 64 Кбайта каждый. 24-битовый указатель адреса кода используется для доступа к ячейкам памяти. Этот указатель состоит из двух частей: 8-битового указателя сегмента CSP и 16-битового указателя команды IP (смещения). Объединение CSP и IP дает 24-битовый физический адрес, см. рисунок 3.9.

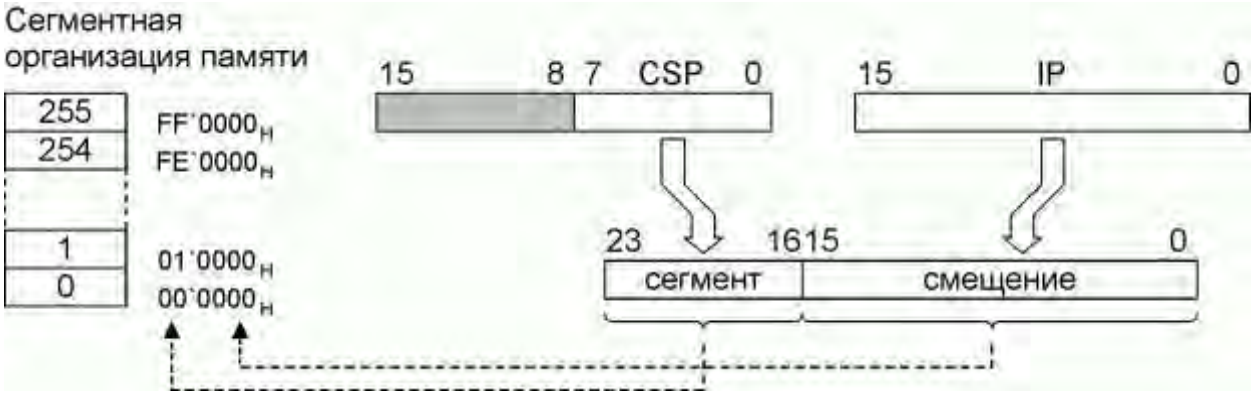


Рисунок 3.9 – Адресация посредством CSP и IP

Указатель команд IP

Регистр IP определяет внутренний 16-битный сегментный адрес текущей команды, см. рисунок 3.10, таблицу 3.2, при этом сегмент данных выбирается при помощи регистра CSP. Регистр IP размещен вне адресного пространства микросхемы 1887ВЕ3Т и поэтому не доступен для пользователя. Однако IP можно изменить косвенным путем с помощью стека, посредством команды возврата.

Значение регистра IP изменяется при выполнении центральным процессором команд перехода, а также инкрементируется после вызова.

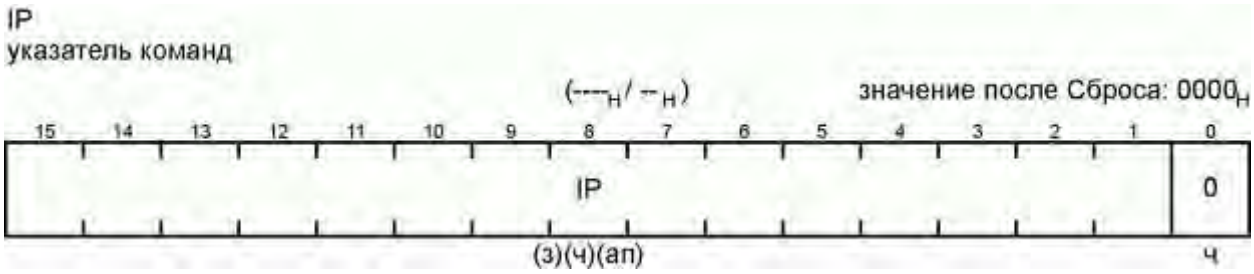


Рисунок 3.10 – Формат регистра IP

Таблица 3.2 – Функциональное назначение полей регистра IP

Поле	Биты	Тип	Описание
IP	15–1	(Запись) (Чтение) (Аппаратное влияние)	Внутрисегментное смещение, по которому необходимо произвести выборку инструкции. IP относится к сегменту, указанному в битовом поле SEGNR регистра CSP
0	0	Чтение	Указатель IP всегда выровнен пословно

Указатель сегмента кода CSP

CSP – побитно неадресуемый регистр, отвечающий за выбор сегмента кода для доступа к командам, см. рисунок 3.11, таблицу 3.3. Младшие 8 бит регистра CSP выбирают один из 256 сегментов.

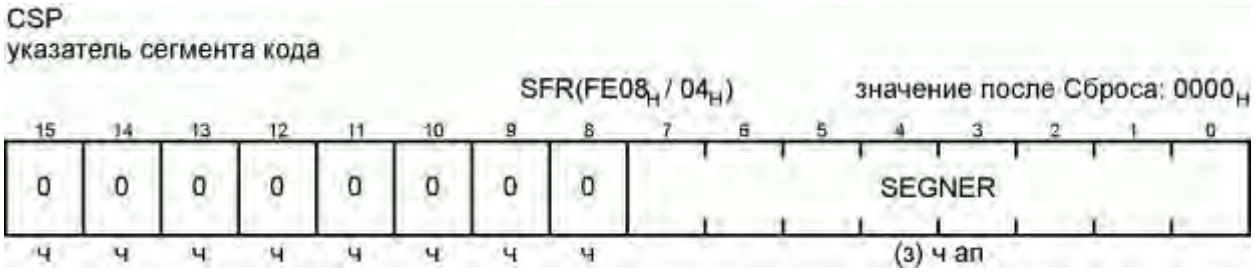


Рисунок 3.11 – Формат регистра CSP

Таблица 3.3 – Функциональное назначение полей регистра CSP

Поле	Биты	Тип	Описание
SEGNER	7-0	(Запись) Чтение Аппаратное влияние	Поле содержит значение сегмента кода, откуда происходит выборка команды
0	15-8	Чтение	Зарезервировано. Не использовать

Адрес ячейки кода создается путем прямого расширения 16-битного содержимого IP-регистра содержимым регистра CSP, как показано на рисунке 3.9.

Есть два режима доступа к коду:

- режим сегментированной памяти;
- режим несегментированной памяти.

Режим выбирается битом SGTDIS регистра SYSCON. После сброса, по умолчанию выбирается режим сегментированной памяти.

Режим сегментированной памяти

Регистр CSP доступен пользователю только для чтения. Содержимое CSP изменяется непосредственно командами JMPS и CALLS или косвенно через стек командами RETS и RETI. В случае прерывания или выполнения команды TRAP, в регистр CSP автоматически загружается адрес сегмента, в котором расположен вектор.

Режим несегментированной памяти

Регистр CSP хранит значение нулевого сегмента и не изменяется непосредственно командами JMPS и CALLS или косвенно через стек командами RETS и RETI. В связи с этим, содержимое CSP регистра не имеет значения, потому что все возможные обращения к коду автоматически ограничиваются нулевым сегментом.

Примечание – При запрещенной сегментации регистр IP может использоваться как 16-битный адрес.

Регистр конфигурации/управления системы SYSCON

Для конфигурации микроконтроллера 1887ВЕ3Т и управления им используется побитно адресуемый регистр SYSCON, см. рисунок 3.12, таблицу 3.4. Состояние регистра после сброса зависит от состояния конфигурационных входов во время сброса.

SYSCON
регистр управления системы

значение после Сброса:
00000XX0 X0000000$_B$

SFR (FF12$_H$/ 89$_H$)

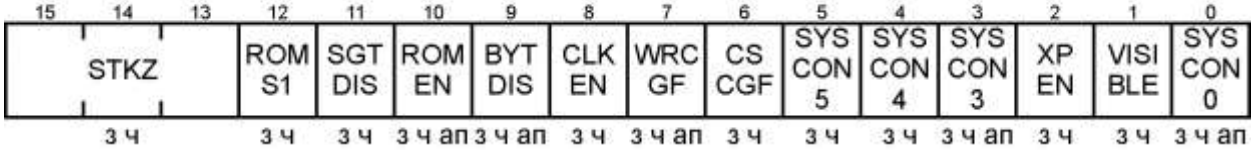


Рисунок 3.12 – Формат регистра SYSCON

Таблица 3.4 – Функциональное назначение полей регистра SYSCON

Поле	Биты	Тип	Описание
1	2	3	4
SYSCON0	0	Запись Чтение Аппаратное влияние	Бит системной конфигурации
VISIBLE	1	Запись Чтение	Управление режимом видимости шины XBUS 0 Обращение по шине XBUS производится только внутри кристалла 1 Обращение по шине XBUS дублируется на внешней шине
XPEN	2	Запись Чтение	Бит разрешения XBUS периферии 0 Доступ к XBUS периферии и обращение к ней запрещены 1 Доступ к XBUS периферии и обращение к ней разрешены
SYSCON3–SYSCON5	3-5	Запись Чтение Аппаратное влияние	Зарезервировано
CSCFG	6	Запись Чтение	Бит управления сигналами выбора кристалла CSx# 0 Режим защелки CSx# сигнала. Сигнал CSx# переключается в низкий уровень с задержкой 1TCL после положительного фронта ALE 1 Режим раннего сигнала CSx#. Сигнал CSx# переключается в низкий уровень по положительному фронту ALE
WRCFG		Запись Чтение Аппаратное влияние	Управление конфигурацией сигнала записи (устанавливается в инверсное значение, считанное с вывода P0H.0 при сбросе) 0 Сигналы WR# и BHE# работают в соответствии со своими функциями 1 WR# работает как WRL#, BHE# работает как WRH#

Окончание таблицы 3.4

1	2	3	4
CLKEN	8	Запись Чтение Аппаратное влияние	Разрешение вывода системного тактового сигнала CLKOUT
BYTDIS	9	Запись Чтение Аппаратное влияние	Бит управления запретом и разрешением сигнала BHE# (устанавливается в зависимости от ширины) 0 Сигнал BHE# разрешен 1 Сигнал BHE# запрещен, вывод P3.12 может использоваться для программного ввода-вывода
ROMEN	10	Запись Чтение Аппаратное влияние	Бит доступа к внутреннему ПЗУ (устанавливается аппаратно в зависимости от напряжения на входе EA#) 0 Обращение к внутреннему ПЗУ запрещено, работа с внешней памятью 1 Обращение к внутреннему ПЗУ разрешено
SGTDIS	11	Запись Чтение	Бит управления запрещением и разрешением сегментации памяти 0 Сегментация разрешена (CSP и IP сохраняются-восстанавливаются при входе-выходе из прерывания) 1 Сегментация запрещена (сохраняются только значения IP)
ROMS1	12	Запись Чтение	Бит локализации внутренней памяти 0 Внутренняя память программ размещается в нулевом сегмент ($00`0000_H$ – $00`7FFF_H$) 1 Внутренняя память программ размещается в первом сегменте ($01`0000_H$ – $01`7FFF_H$)
STKZ	15-13	Запись Чтение	Биты выбора размера системного стека в DPRAM 000 256 слов ($00'FBFE_H$…$00'FA00_H$) 001 128 слов ($00'FBFE_H$…$00'FB00_H$) 010 64 слова ($00'FBFE_H$…$00'FB80_H$) 011 32 слова ($00'FBFE_H$…$00'FBC0_H$) 100 512 слов ($00'FBFE_H$…$00'F800_H$) 101 Зарезервировано 110 Зарезервировано 111 1536 слов ($00'FDFE_H$ …$00'F200_H$)
Примечание – Регистр SYSCON не может быть изменен после выполнения команды EINIT.			

3.4 Использование регистров общего назначения GPR

Микроконтроллер 1887ВЕ3Т использует несколько банков регистров, каждый состоит из 16 регистров R0, …, R15, называемых регистрами общего назначения GPR, к которым можно обратиться за один цикл ЦПУ. GPR являются рабочими регистрами блока АЛУ, а также могут использоваться как указатели адреса в режимах косвенной адресации.

Несколько банков регистров общего назначения расположены в DPRAM. Один банк использует блок из 16 последовательных слов. Регистр контекстного указателя CP определяет базовый адрес используемого банка, см. рисунок 3.13.

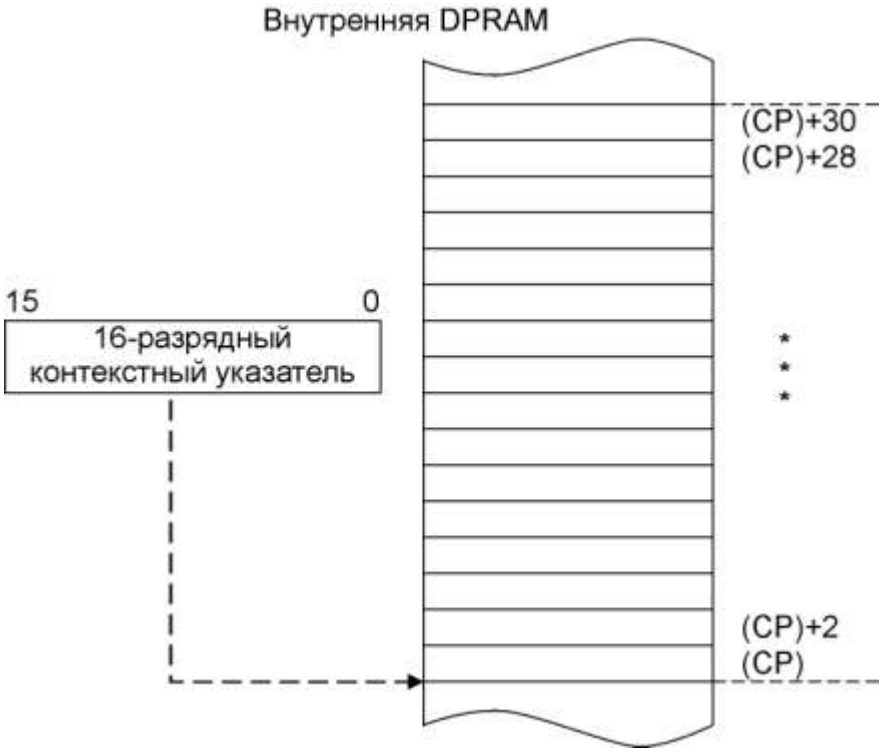


Рисунок 3.13 – Выбор банка регистров посредством регистра CP

Микроконтроллер может переключать банки регистров одной командой для задач, требующих короткого времени выполнения. После переключения, новая задача выполняется уже в рамках ее отдельного контекста.

Для доступа к GPR используются три режима адресации:

- 4-битный;
- 8-битный;
- 24-битный.

4-битная адресация

Короткая 4-битная адресация GPR (обозначение: Rw или Rb) позволяет использовать относительные адреса памяти, привязанные к содержимому регистра CP, т. е. к базовому адресу текущего банка регистров. В зависимости от использования данного режима для относительного доступа к слову Rw или байту Rb, короткий 4-битный адрес перед сложением с содержимым регистра CP может быть умножен на два для доступа к словам или может быть оставлен без изменений.

GPR, используемые в качестве указателя косвенного адреса, всегда доступны пословно. Для некоторых команд только первые четыре регистра GPR могут использоваться в качестве указателя косвенного адреса. Эти GPR доступны с помощью короткой 2-битной адресации. Вычисление физических адресов идентично короткой 4-битной GPR-адресации.

8-битная адресация

Короткая 8-битная адресация регистров (обозначение: reg или bitoff) интерпретирует младшие 4 значащих бита в диапазоне от $F0_H$ до FF_H, как короткие 4-битные адреса регистров GPR, в то время как четыре старших значащих бита игнорируются. Вычисление представленного физического адреса GPR идентично вычислению короткого 4-битного адреса GPR. Для доступа к одиночным битам в GPR, слово адреса GPR вычисляется, как

описано выше, но позиция бита в слове определяется с помощью независимого дополнительного 4-битного значения, см. рисунок 3.14.

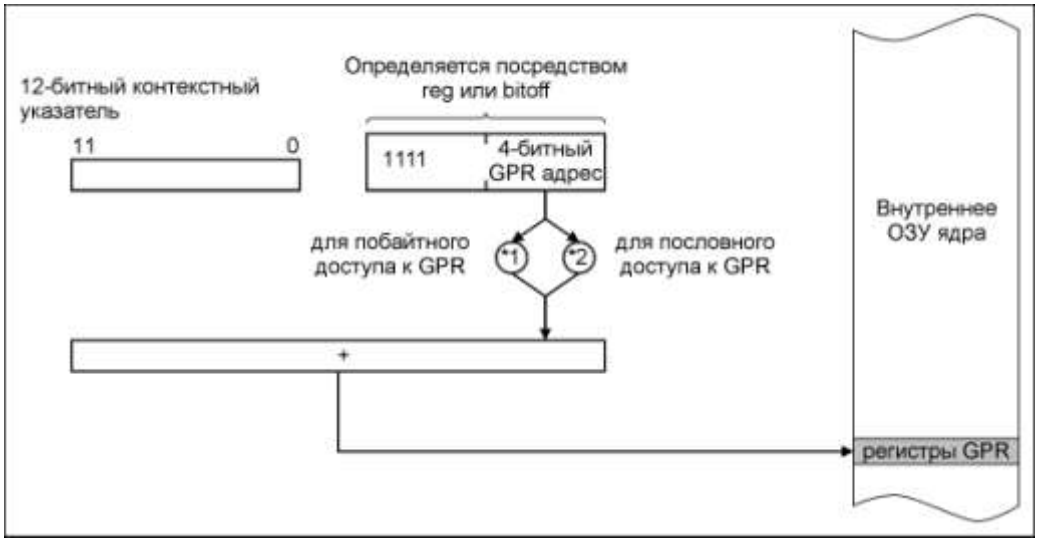


Рисунок 3.14 – Неявное использование регистра CP логикой режима короткой адресации

24-битный режим адресации

24-разрядные адреса памяти в пределах диапазона CP от плюс 0 до плюс 30 могут использоваться для непосредственного доступа к регистрам общего назначения. Возможны оба доступа – пословно и побайтно. 24-битный адрес формируется согласно правилам режима формирования длинного и косвенного адреса, см. таблицу 3.5.

Таблица 3.5 – Режимы адресации для доступа к GPR-словам

Имя	Физический адрес	8-бит адрес	4-бит адрес	Описание	Значение после сброса
R0	(CP)+0	$F0_H$	0_H	Регистр R0 (слово)	$UUUU_H$
R1	(CP)+2	$F1_H$	1_H	Регистр R1 (слово)	$UUUU_H$
R2	(CP)+4	$F2_H$	2_H	Регистр R2 (слово)	$UUUU_H$
R3	(CP)+6	$F3_H$	3_H	Регистр R3 (слово)	$UUUU_H$
R4	(CP)+8	$F4_H$	4_H	Регистр R4 (слово)	$UUUU_H$
R5	(CP)+10	$F5_H$	5_H	Регистр R5 (слово)	$UUUU_H$
R6	(CP)+12	$F6_H$	6_H	Регистр R6 (слово)	$UUUU_H$
R7	(CP)+14	$F7_H$	7_H	Регистр R7 (слово)	$UUUU_H$
R8	(CP)+16	$F8_H$	8_H	Регистр R8 (слово)	$UUUU_H$
R9	(CP)+18	$F9_H$	9_H	Регистр R9 (слово)	$UUUU_H$
R10	(CP)+20	FA_H	A_H	Регистр R10 (слово)	$UUUU_H$
R11	(CP)+22	FB_H	B_H	Регистр R11 (слово)	$UUUU_H$
R12	(CP)+24	FC_H	C_H	Регистр R12 (слово)	$UUUU_H$
R13	(CP)+26	FD_H	D_H	Регистр R13 (слово)	$UUUU_H$
R14	(CP)+28	FE_H	E_H	Регистр R14 (слово)	$UUUU_H$
R15	(CP)+30	FF_H	F_H	Регистр R15 (слово)	$UUUU_H$

Примечание – Первые 8 регистров R0, … , R7 из GPR также доступны побайтно. При этом запись в адресуемый байт регистра не оказывает влияние на содержимое второго байта того же регистра.

Каждая половина регистра-байта (8 бит) имеет свое уникальное имя, см. таблицу 3.6.

Таблица 3.6 – Режимы адресации для доступа к GPR-байтам

Имя	Физический адрес	8-битный адрес	4-битный адрес	Описание	Значение после сброса
RL0	(CP)+0	$F0_H$	0_H	Регистр RL0 (байт)	UU_H
RH0	(CP)+1	$F1_H$	1_H	Регистр RL1 (байт)	UU_H
RL1	(CP)+2	$F2_H$	2_H	Регистр RL2 (байт)	UU_H
RH1	(CP)+3	$F3_H$	3_H	Регистр RL3 (байт)	UU_H
RL2	(CP)+4	$F4_H$	4_H	Регистр RL4 (байт)	UU_H
RH2	(CP)+5	$F5_H$	5_H	Регистр RL5 (байт)	UU_H
RL3	(CP)+6	$F6_H$	6_H	Регистр RL6 (байт)	UU_H
RH3	(CP)+7	$F7_H$	7_H	Регистр RL7 (байт)	UU_H
RL4	(CP)+8	$F8_H$	8_H	Регистр RL8 (байт)	UU_H
RH4	(CP)+9	$F9_H$	9_H	Регистр RL9 (байт)	UU_H
RL5	(CP)+10	FA_H	A_H	Регистр RL10 (байт)	UU_H
RH5	(CP)+11	FB_H	B_H	Регистр RL11 (байт)	UU_H
RL6	(CP)+12	FC_H	C_H	Регистр RL12 (байт)	UU_H
RH6	(CP)+13	FD_H	D_H	Регистр RL13 (байт)	UU_H
RL7	(CP)+14	FE_H	E_H	Регистр RL14 (байт)	UU_H
RH7	(CP)+15	FF_H	F_H	Регистр RL15 (байт)	UU_H

Контекстный указатель

Подпрограмма обработки прерываний или планировщик задач системы, как правило, сохраняет содержимое всех используемых регистров в стек и затем восстанавливает его перед возвращением к выполнению прерванной программы. Увеличение числа регистров, используемых программой, ведет к увеличению времени сохранения и восстановления.

Содержимое банка регистров переключается изменением основного адреса банка GPR. Основным адресом является содержимое регистра контекстного указателя CP, см. рисунок 3.15, таблицу 3.7.

Контекстный указатель CP – побитно не адресуемый регистр, с возможностью изменения содержимого посредством любой команды модификации SFR.

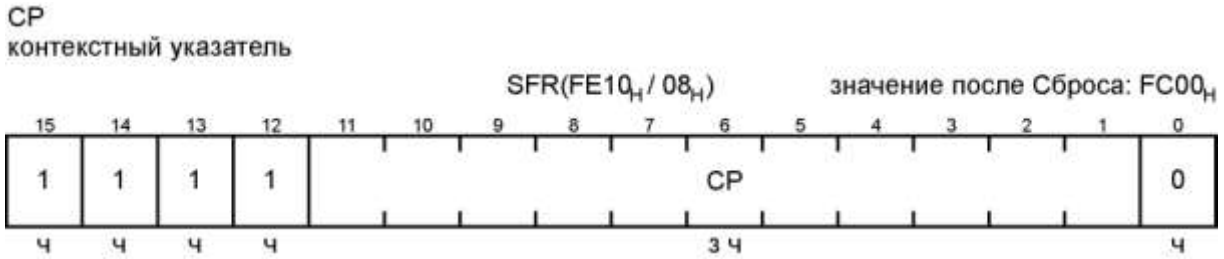


Рисунок 3.15 – Формат регистра CP

Таблица 3.7 – Функциональное назначение полей регистра CP

Поле	Биты	Тип	Описание
1	15–12	Чтение	CP всегда указывает на DPRAM
CP		Запись Чтение	Изменяемое битовое поле указателя Определяет адрес слова/байта используемого банка регистров. Примечание – Если при записи нового значения содержимое битов CP.9, CP.10, CP.11 записываемого значения равно «0», то в биты CP.10 и CP.11 автоматически записываются «1»
0	0	Чтение	Указатель CP всегда выровнен пословно

Необходимо следить за тем, чтобы физический адрес GPR, определяемый регистром CP, а также короткий адрес GPR находились в области ОЗУ. В случае несоблюдения этого, может возникнуть непредсказуемая ситуация. Нельзя записывать в регистр CP значение адреса, лежащее за переделами начала DPRAM. Из-за применения конвейера команд, новое значение регистра CP не может быть использовано для вычисления адреса GPR в течение времени, пока происходит выполнение двух команд, следующих за командой, выполнившей обновление регистра CP.

Команда SCXT CP, #N_B (N_B – номер нового банка регистров) переносит текущее значение контекстного указателя CP в системный стек и загружает регистр CP значением N_B, которое выбирает новый банк регистров. После этого подпрограмма может использовать регистры указанного банка, которые резервируются именно для этой подпрограммы, что означает: содержимое этих регистров может использоваться при дальнейших вызовах этой подпрограммы. Перед возвращением из подпрограммы (командой RETI) сохраненное в стеке значение контекстного указателя заносится в регистр CP и, таким образом, снова активируется используемый ранее банк регистров.

3.5 Адресация данных

Микроконтроллер осуществляет несколько режимов адресации для пословного, побайтного и побитного обращения (короткого, длинного, косвенного) к данным. Разные режимы адресации используют различные форматы и области применения.

Возможно выполнение следующих задач:

- формирование адреса с использованием режимов короткой, длинной и косвенной адресации;
- разбиение на страницы данных или механизм замещения;
- реализация системного стека.

Режимы короткой адресации

Во всех режимах короткой адресации используется неявное смещение адреса для определения 24-битного физического адреса. Режимы короткой адресации позволяют иметь доступ к SFR, GPR и бит-адресуемой области памяти, см. таблицу 3.8.

Для побайтного обращения: физический адрес = начальный адрес + короткий адрес.

Для пословного обращения: физический адрес = начальный адрес + 2* короткий адрес.

Таблица 3.8 – Режимы короткой адресации

Мнемокод	Физический адрес	Диапазон короткого адреса	Границы области применения
Rw	(CP)+2*Rw или локальный	Rw= 0...15	Область GPR (слово)
Rb	(CP)+ 1*Rb или локальный	Rb= 0...15	Область GPR (байт)
reg	$00'FE00_H$+ 2*reg $00'F000_H$+ 2*reg (CP)+ 2*(reg^$0F_H$) (CP)+ 1*(reg^$0F_H$)	reg= 00_H...EF_H reg= 00_H...EF_H reg= $F0_H$...FF_H reg= $F0_H$...FF_H	Область SFR (слово, младший байт) Область ESFR (слово, младший байт) Область GPR (слово, байт)
bitoff	$00'FD00_H$+ 2*bitoff $00'FF00_H$+ 2*(bitoff^$7F_H$) $00'F100_H$+ 2*(bitoff^$7F_H$) (CP) + 2*(bitoff^$0F_H$)	bitoff= 00_H...$7F_H$ bitoff= 80_H...EF_H bitoff= 80_H...EF_H bitoff= $F0_H$...FF_H	Бит в смещении слова из областей DPRAM, SFR, ESFR, GPR
bitaddr	Смещение как в bitoff. Непосредственная позиция бита (bitpos)	bitoff = 00_H...FF_H bitpos = 0...15	Любой бит

Rw, Rb: Задают прямой доступ к любому регистру общего назначения GPR в текущем активном контексте (глобальный или локальный блок регистров). Как Rw, так и Rb требуют 4 бита в формате команды. Базовый (начальный) адрес глобального банка регистров определяется через содержимое регистра контекстного указателя CP. Rw определяет 4-разрядный адрес слова регистра общего назначения. Rb определяет 4-разрядный адрес байта регистра общего назначения в пределах локального банка регистров или относительно CP.

reg: Определяет прямой доступ к любому регистру специального назначения SFR или регистру общего назначения GPR в текущем активном контексте. Для значения reg требуется 8 битов в формате команды. Короткие адреса reg в области от 00_H до EF_H всегда определяют (E)SFR. В этом случае базовый адрес будет $00`FE00_H$ для стандартной области SFR или $00`F000_H$ для расширенной области ESFR. Для осуществления доступов reg к области ESFR требуется предварительное выполнение EXT-команды для переключения базового адреса. В зависимости от кода операции или целое слово (для операций со словами) или младший байт (для операций с байтами) регистра специального назначения могут быть адресованы с помощью reg. Отметим что, используя режим адресации reg, нельзя обратиться к старшему байту SFR. Короткие адреса reg в области от $F0_H$ до FF_H всегда определяют GPR. В этом случае только младшие 4 бита reg имеют значение для формирования физического адреса и, следовательно, ситуация идентична формированию адреса с использованием режимов адресации Rb и Rw.

bitoff: Определяет прямой доступ к любому слову в бит-адресуемой области памяти. Для значения bitoff требуется 8 битов в формате команды. Заданная область bitoff выбирает различные базовые адреса для создания физических адресов. Короткие адреса bitoff в диапазоне от 00_H до $7F_H$ используют в качестве базового адреса значение $00`FD00_H$ для определения 128 верхних слов DPRAM в диапазоне от $00`FD00_H$ до $00`FDFE_H$. Короткая адресация bitoff в диапазоне от $80_H$ к $EF_H$ использует базовый адрес $00`FF00_H$, чтобы определить внутреннее положение слова SFR в диапазоне $00`FF00_H$ до $00`FFDE_H$ или базового адреса 00`F100н, чтобы определить внутреннее положение слова ESFR в диапазоне от $00`F100_H$ до $00`F1DE_H$. Для осуществления доступов bitoff к области дополнительных регистров специального назначения ESFR требуется предварительное выполнение EXT-команды для переключения базового адреса. Для короткой адресации bitoff от $F0_H$ до FF_H только младшие 4 бита используются для генерации адреса слова, выбранного GPR.

bitaddr: Адрес любого бита определяется адресом слова в пределах бит-адресуемой области памяти (аналогично bitoff) и позицией бита (bitpos) в пределах данного слова. Следовательно, для bitaddr требуется 12 битов в формате команды.

Режимы длинной и косвенной адресации

Режимы длинной и косвенной адресации используют один из четырех регистров DPP для формирования 24-битного адреса. С помощью этих режимов может быть доступно любое слово и/или байт данных в пределах внутреннего адресного пространства. Любой длинный или косвенный 16-битный адрес состоит из двух частей различного назначения. Биты 13, …, 0 адреса определяют 14-битное смещение страницы данных, а биты 15 и 14 являются указателем страницы данных DPP (одной из четырех), которая используется для формирования 24-битного адреса, см. рисунок 3.16.

Микроконтроллер имеет механизм замещения для схемы адресации с помощью регистров DPP (команды EXTP(R) и EXTS(R)).

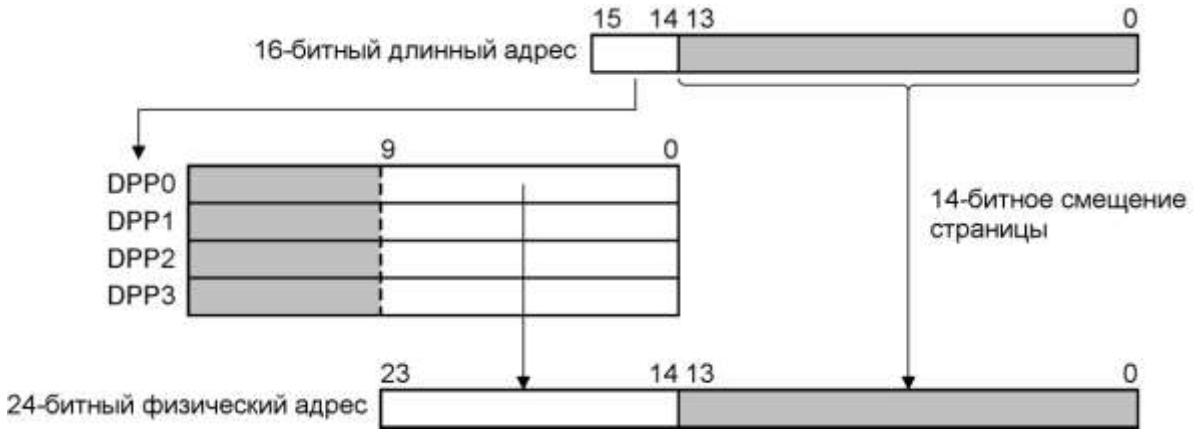


Рисунок 3.16 – Интерпретация 16-битного длинного адреса

Примечание – Обращение к нечетному байту слова запрещено и приводит к аппаратной ловушке.

Адресация с использованием указателя страницы данных DPP

Четыре неадресуемых побитно регистра DPP выбирают одну из четырех страниц данных. Младшие 10 битов каждого регистра DPP выбирают одну из 1024 допускаемых 16 Кбайт страниц данных. Старшие 6 битов зарезервированы.

Регистры DPP используются неявно при доступе к данным (расположенных в любой области памяти) посредством режимов прямой или косвенной 16-битной адресации (за исключением случаев, когда используется механизм замещения EXT-командами и модулем PEC при передачах данных).

Перемещение по страницам данных осуществляется конкатенацией младших 14 битов прямого или косвенного длинного 16-битного адреса с содержимым регистра DPP, выбираемого двумя старшими битами 16-битного адреса. Содержимое выбранного регистра DPP указывает на одну из 1024 страниц данных. Адрес страницы данных и младшие 14 битов длинного адреса вместе составляют 24-битный физический адрес, см. рисунок 3.17.

.

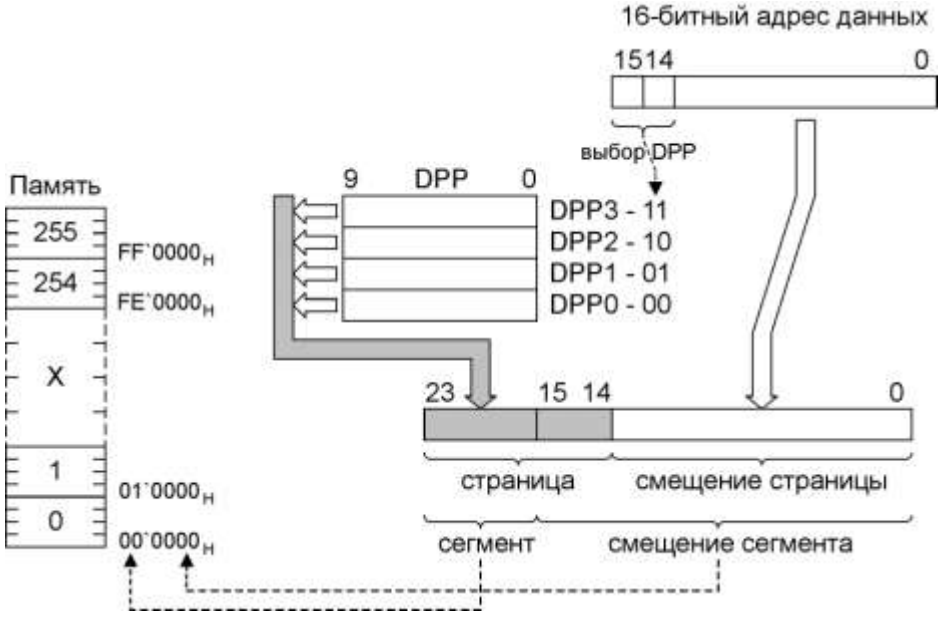


Рисунок 3.17 – Адресация посредством регистров DPP

После сброса содержимое регистров DPP указывает на 3-ю, 2-ю, 1-ю и 0-ю страницы данных нулевого сегмента, представленных на рисунках 3.18 – 3.21, функциональное назначение полей регистров DPP0, DPP1, DPP2, DPP3 представлено в таблице 3.9.

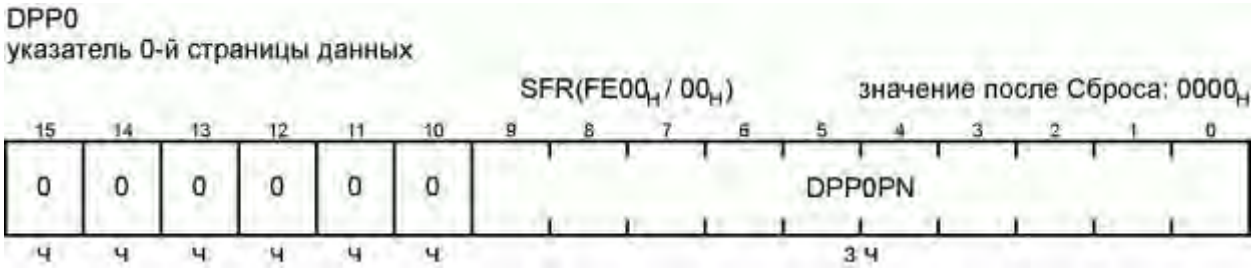


Рисунок 3.18 – Формат регистра DPP0

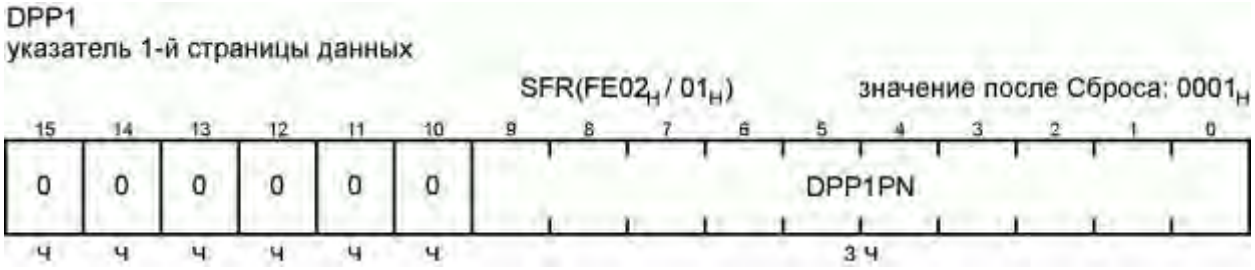


Рисунок 3.19 – Формат регистра DPP1

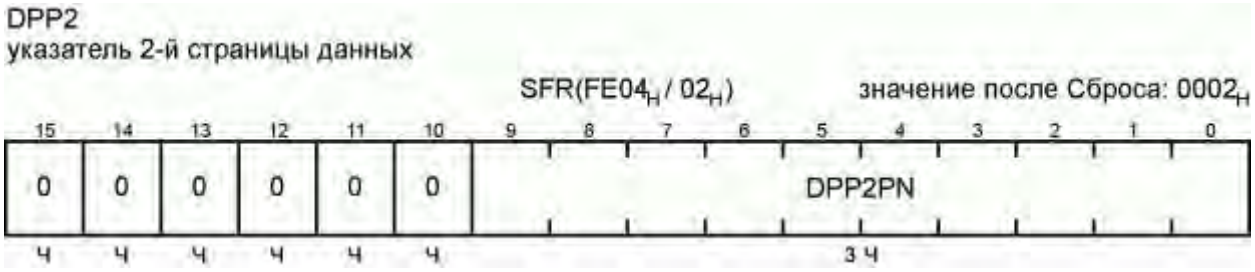


Рисунок 3.20 – Формат регистра DPP2

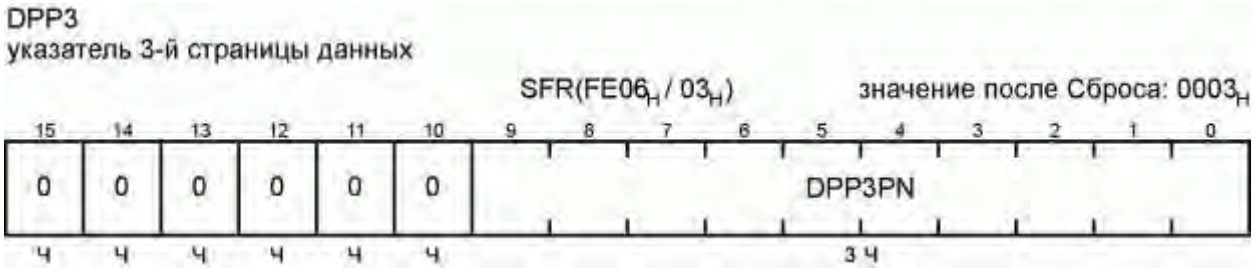


Рисунок 3.21 – Формат регистра DPP3

Таблица 3.9 – Функциональное назначение полей регистров DPP0, DPP1, DPP2, DPP3

Поле	Биты	Тип	Описание
DPPxPN, x = 0, …, 3	9-0	Запись Чтение	Номер страницы данных в пределах выбранного регистра DPP
0	15-10	Чтение	Зарезервировано

Примечание – В режиме несегментированной памяти, все регистры DPP используются для вычисления 24-битного физического адреса.

Содержимое любого регистра DPP может быть обновлено посредством любой команды, которая может изменять содержимое SFR.

Из-за применения конвейера команд, новое значение регистра DPP не может быть использовано для вычисления адреса сразу после команды, выполнившей обновление регистра DPP.

Механизм замещения

Для временного отключения стандартной схемы формирования адреса, в микроконтроллере предусмотрен механизм замещения с использование команд EXTP(R) и EXTS(R). Команда EXTP(R) переносит содержимое регистра DPP, в то время как команда EXTS(R) осуществляет конкатенацию полного длинного 16-битного адреса и определенного базового адреса сегмента. Страница или сегмент, к которым был применен такой механизм, могут быть определены непосредственно как константы #pag, #seg или косвенно через GPR (слово Rw), смотри рисунок 3.22.

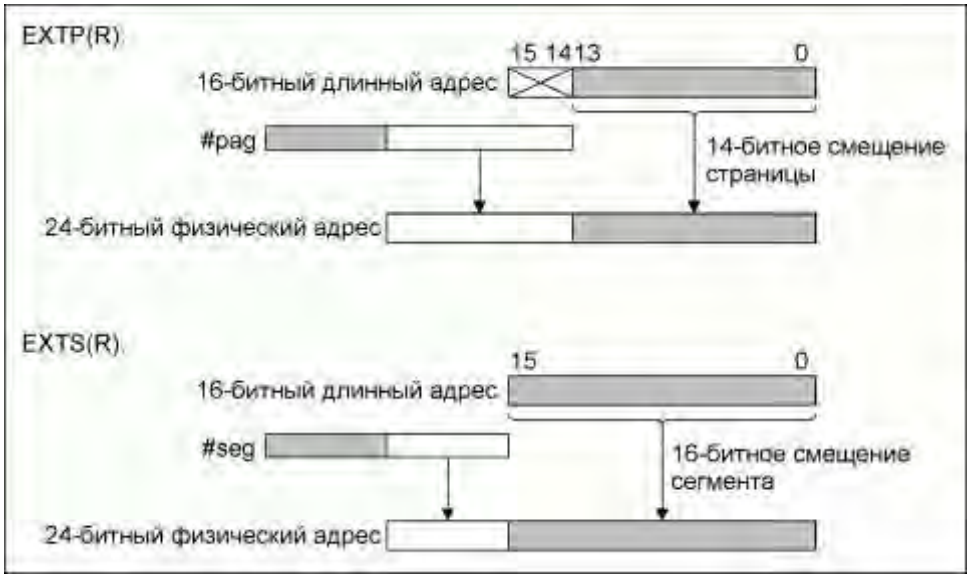


Рисунок 3.22 – Механизм замещения регистра DPP

Режим длинной адресации

Режим длинной адресации использует 16-разрядное постоянное значение, кодируемое в формат команды, которое определяет смещение страницы данных и выбирает регистр DPP, см. таблицу 3.10.

Режим длинной адресации обозначается как mem.

Таблица 3.10 – Режимы длинной адресации

Мнемоника	Физический адрес	Область доступа
mem	(DPP0)|| mem^3FFF$_H$ (DPP1)|| mem^3FFF$_H$ (DPP2)|| mem^3FFF$_H$ (DPP3)|| mem^3FFF$_H$	Любое слово или байт
mem	pag|| mem^3FFF$_H$	
mem	seg|| mem	

Также длинная адресация может быть использована одновременно с механизмом замещения DPP (EXTP(R) и EXTS(R)).

Режимы косвенной адресации

Эти режимы могут быть определены как комбинации режимов короткой и длинной адресаций. Это означает, что длинный 16-битный адрес формируется косвенно содержимым регистра общего назначения (GPR – слово), который в свою очередь выбирается коротким 4-битовым адресом (4 бита достаточно для 16 регистров Rw). В одних режимах косвенной адресации к содержимому GPR добавляется постоянно значение перед вычислением длинного 16-битного адреса, в других – значение указателя косвенного адреса (содержимое GPR) может инкрементироваться или декрементироваться на 2 или 1 (в зависимости от разрядности – байт или слово).

В каждом случае для формирования 24-битного адреса используется один из четырех регистров DPP. Косвенно можно обратиться к любому байту или слову адресного пространства.

Косвенная адресация может быть использована одновременно с механизмом замещения DPP (EXTP(R) и EXTS(R)).

Некоторые команды в качестве косвенных указателей адреса используют только младшие четыре 16-битных регистра (R0, R1, R2, R4), которые выбираются посредством 2-битных адресов.

Физические адреса формируются из косвенных указателей адреса по следующему алгоритму:

1) Вычисляется физический адрес GPR (регистр Rw, который используется для косвенной адресации) с использованием короткого 2-битного адреса:

$$GPR_{АДРЕС} = (СР) + 2* \text{короткий адрес}.$$

2) При необходимости, содержимое косвенного указателя адреса Rw может быть предварительно декрементировано (на единицу, в случае байтовых операций и на 2, в случае операций со словами) до формирования 16-битного адреса:

$$GPR_{АДРЕС} = (GPR_{АДРЕС}) – (2, \text{если байт, или } 1, \text{если слово}).$$

3) Вычисление длинного 16-битного адреса прибавлением постоянного значения (Rw + const16, если выбрано) к содержимому косвенного указателя адреса:

$$\text{Длинный адрес} = (\text{указательGPR}) + \text{постоянное значение}.$$

4) Вычисление 24-битного физического адреса использует конкатенацию длинного адреса и содержимого соответствующего регистра DPP:

$$\text{Физический адрес} = (DPP) + \text{смещение страницы}.$$

5) При необходимости происходит последующее инкрементирование или декрементирование содержимого косвенного указателя адреса на 1 для операций с байтами и на 2 – со словами:

$$GPR_{УКАЗАТЕЛЬ} = (GPR_{УКАЗАТЕЛЬ}) \pm (2, \text{если байт, или } 1, \text{если слово}).$$

Поддерживаемые режимы косвенной адресации сведены в таблицу 3.11.

Таблица 3.11 – Режимы косвенной адресации

Мнемоника	Описание
[Rw]	Для косвенной адресации большинство команд используют любой из GPR (от R0 до R15). Некоторые команды для этих целей используют только четыре регистра – R0, R1, R2, R3
[Rw+]	Косвенный указатель адреса с автоматическим последующим (после обращения) инкрементированием на 1, если операции с байтом, или 2, если операции со словом
[–Rw]	Косвенный указатель адреса с автоматическим предекрементированием (до обращения) на 1, если операции с байтом, или 2, если операции со словом
[Rw+#data16]	16-битная константа, добавляемая к значению косвенного указателя адреса при вычислении длинного адреса

Системный стек

Системный стек предназначен для сохранения векторов возврата, указателей сегментов и состояний процессора для различных процедур и подпрограмм обслуживания прерываний.

Внутренний стек также может использоваться для временного хранения данных или их пересылки между задачами или подпрограммами. Переносом данных из регистров в стек и из стека в регистры управляют соответствующие команды. В то же время организации взаимодействия банков регистров вполне достаточно для межпрограммного и межзадачного переноса данных.

Примечание – Системный стек допускает только работу со словами данных. Байты должны быть дополнены до слов вторыми байтами. Регистр указателя стека SP может быть загружен только четным значением адреса. Указатель стека работает с областью памяти DPRAM.

Регистр указателя стека SP

Не адресуемый побитно регистр SP используется для указания адреса вершины внутреннего системного стека TOS. Значение регистра SP декрементируется, как только в стек посылаются данные, и инкрементируется после возврата данных из стека. Таким образом, системный стек растет от больших к меньшим значениям адресов памяти.

Наименьшему значащему биту в регистре SP всегда присваивается «0» , а битам с 15 по 12 всегда – «1», поэтому регистр SP может принимать значения от $F000_H$ до $FFFE_H$. Это позволяет получить доступ к физическому стеку внутренней области DPRAM. Виртуальный стек (обычно больший, чем физический) может быть реализован программным способом. Этот механизм поддерживается регистрами STKOV и STKUN.

Значение регистра SP может быть изменено любой командой, поддерживающей возможность изменения содержимого регистров области SFR. Функциональное назначение полей регистра SP представлено в таблице 3.12, формат регистра SP смотри на рисунке 3.23.

Примечание – Из-за применения конвейера команд, команды POP и RETURN не должны следовать сразу за командой, изменяющей значение регистра SP.

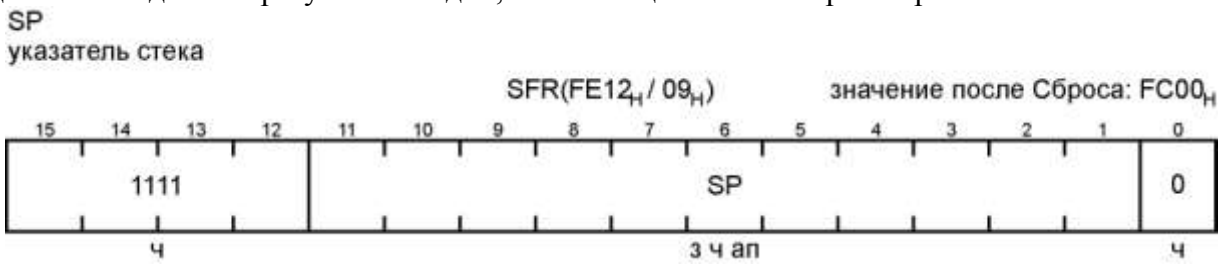


Рисунок 3.23 – Формат регистра SP

Таблица 3.12 – Функциональное назначение полей регистра SP

Поле	Биты	Тип	Описание
1111	15–12	Чтение	По умолчанию всегда равно 1111_B
SP	11–1	Запись Чтение Аппаратное влияние	Указатель вершины системного стека
0	0	Чтение	По умолчанию всегда равно 0_B

Переполнение и опустошение стека

Направлением перемещения данных из стека/в стек управляют два регистра STKOV (указатель переполнения стека) и STKUN (указатель опустошения стека). Специальные системны ловушки (переполнения и опустошения стека) срабатывают при достижении регистром SP границ стека, определенных регистрами STKOV и STKUN.

Зачастую пользователь помещает команду программного сброса SRST в подпрограмму обработки ловушек стека. Однако этот способ приводит к тому, что внутренний стек предназначен только для выполнения конкретной программы и при выходе за его границы возникает ошибка.

Также возможно использовать ловушки переполнения и опустошения стека для кэширования частей большего внешнего стека.

Указатель переполнения стека STKOV

Значение этого побитно не адресуемого регистра сравнивается со значением SP после каждой операции, наполняющей данными системный стек (команды PUSH и CALL или прерывания), и после каждого вычитания из значения SP. Если содержимое регистра SP менее содержимого регистра STKOV, то имеет место аппаратная ловушка переполнения стека.

Так как наименьший значащий бит в регистре STKOV всегда равен нулю, а битам с 15 по 12 всегда аппаратно присваивается «1», то регистр STKOV может содержать значения от $F000_H$ до $FFFE_H$. Функциональное назначение полей регистра STKOV представлено в таблице 3.13, формат регистра STKOV – на рисунке 3.24.

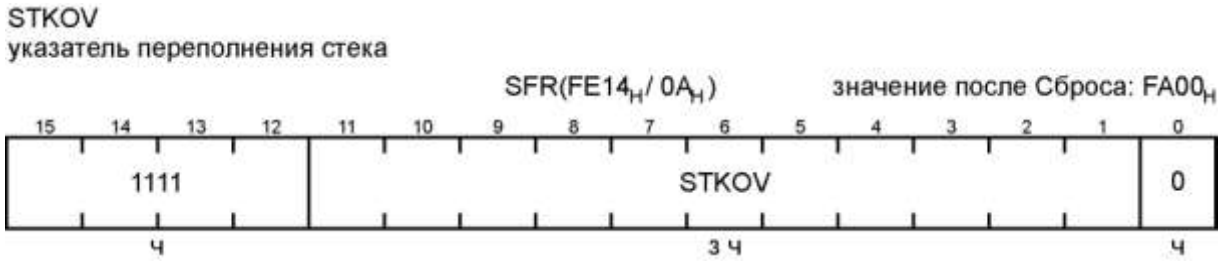


Рисунок 3.24 – Формат регистра STKOV

Таблица 3.13 – Функциональное назначение полей регистра STKOV

Поле	Биты	Тип	Описание
1111	15–12	Чтение	По умолчанию всегда равно 1111_B
STKOV	11–1	Запись Чтение	Битовое поле, определяющее смещение адреса сегмента нижней границы системного стека
0	0	Чтение	По умолчанию всегда равно 0_B

Значение регистра STKOV может быть изменено любой командой, поддерживающей возможность изменения содержимого регистров области SFR.

Примечание – Записываемое в указатель стека значение не проверяется.

Указание на фатальную ошибку обрабатывает переполнение стека как системную ошибку с помощью специальной подпрограммы обслуживания ловушки. В этом случае данные на дне стека могут быть перезаписаны посредством сохраненной в стеке информации о статусе во время обслуживания ловушки переполнения стека.

Автоматическое смещение системного стека позволяет использовать системный стек как стековый кэш для создания большого внешнего пользовательского стека. В этом случае STKOV-регистр должен быть установлен на значение, представляющее сумму адреса желаемой верхней границы стека TOS и смещения, являющегося максимальным размером стека. Наихудшим случаем, который может случиться, является обнаружение состояния переполнения стека в момент входа в обслуживание прерывания ISR или во время выполнения команды ATOMIC или последовательности EXT-команд. В этом случае требуются дополнительные слова стека для сохранения значений IP, PSW, CSP как при обслуживании прерывания, так и при обслуживании аппаратной ловушки.

Указатель опустошения стека STKUN

Значение этого не адресуемого побитно регистра сравнивается с регистром SP после каждой операции, запрашивающей данные из системного стека при помощи команд POP и RET, и после каждого увеличения значения регистра SP. Если содержимое регистра SP больше, чем содержимое регистра STKUN, то будет иметь место аппаратная ловушка.

Так как наименьший значащий бит в регистре STKUN всегда равен нулю, а битам с 15 по 12 всегда аппаратно присваивается «1», то регистр STKUN может содержать значения от $F000_H$ до $FFFE_H$. Функциональное назначение полей регистра STKUN смотри в таблице 3.14, формат регистра STKUN – на рисунке 3.25.

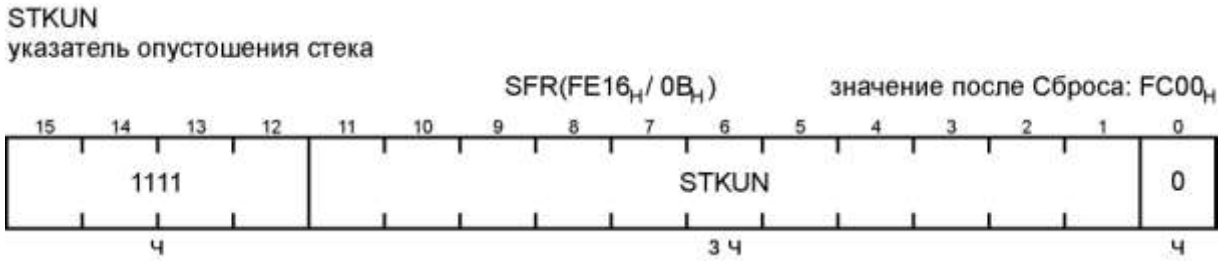


Рисунок 3.25 – Формат регистра STKUN

Таблица 3.14 – Функциональное назначение полей регистра STKUN

Поле	Биты	Тип	Описание
1111	15–12	Чтение	По умолчанию всегда равно 1111_B
STKUN	11–1	Запись Чтение	Битовое поле, определяющее смещение адреса сегмента верхней границы системного стека
0	0	Чтение	По умолчанию всегда равно 0_B

Значение регистра STKUN может быть изменено любой командой, поддерживающей возможность изменения содержимого регистров области SFR.

Примечание – Записываемое в указатель стека значение не проверяется.

Указание на фатальную ошибку обрабатывает опустошение стека как системную ошибку с помощью подпрограммы обслуживания ловушки.

Автоматическое изменение системного стека позволяет использовать системный стек как стековый кэш для получения большего значения объема пользовательского стека. В этом случае, в регистр STKUN устанавливается значение, которое представляет собой верхний адрес основания стека.

Управления областью и границами стека

Управление границами стека реализовано с помощью регистров STKOV и STKUN, определяющих случаи выхода указателя стека SP за пределы определенной области стека. Переполнение и опустошение стека имеет место при использовании команд ADD или SUB, либо при использовании команд PUSH или POP (явных и неявных, т. е. CALL или RET команды).

Механизм ловушек не срабатывает, т. е. ловушки стека не создаются заново в случае:

- значение указателя стека изменяется прямо с помощью команд MOV;
- изменяются пределы области стека STKOV, STKUN, и поэтому SP находится вне новых границ.

Линейный стек

В микроконтроллере 1887ВЕ3Т реализована функция линейного стека $STKSZ = 111_B$, в которой как системный стек может использоваться полностью вся область DPRAM. Это позволяет организовать более объемный стек без необходимости реализации обработки перемещения данных для циркулярного стека. В то же время этот

метод оставляет меньше адресного пространства ОЗУ для переменных и команд. Обращение к области DPRAM, выделенной под стек, осуществляется с помощью указателей STKUN и STKOV. Ловушки опустошения и переполнения стека в этом случае поддерживают только указание на фатальную ошибку.

При использовании линейного стека, для доступа к физическому стеку используются все изменяемые биты регистра SP. Несмотря на то, что указатель стека может покрывать область от $00`F000_H$ до $00`FFFE_H$, системный (физический) стек должен быть расположен в пределах области DPRAM и только в диапазоне от $00`F600_H$ до $00`FDFE_H$. Ограничения размеров стека в указанных пределах остаются на усмотрение пользователя.

Примечание – Стековые обращения к области памяти, лежащей ниже DPRAM (область ESFR и зарезервированная область) и в диапазоне от $00`FE00_H$ до $00`FFFE_H$ (область SFR), могут привести к непредсказуемым результатам.

Циркулярный (виртуальный) стек

Этот способ позволяет заносить в стек значения до тех пор, пока не будет достигнута граница переполнения. По достижении этого предела, часть заносимых в стек данных должна быть сохранена во внешней памяти для формирования пространства для последующих сохранений. Это называется автоматическим смещением. При выполнении некоторого числа команд возвращений или чтения данных из стека достигается верхняя граница. Сохраненные ранее во внешней памяти данные должны быть возвращены. Это называется автоматическим изменением. Поскольку команды вызова подпрограмм не бесконечны и команды вызова и возврата чередуются, автоматические смещения и изменения происходят очень редко. Если использование стека не разрешено для программного обеспечения, то виртуальный стек не может применяться.

Основной механизм – это аппаратная трансформация адресов виртуального стека, управляемых регистрами SP, STKOV и STKUN для определения области физического стека в пределах DPRAM. Эта область виртуального стека покрывает все возможные адреса, на которые может указывать регистр SP (от $00`F000_H$ до $00`FFFE_H$). Регистры STKOV и STKUN охватывают такой же диапазон в 4 Кбайта. Размер области физического стека в пределах DPRAM, который используется для стандартных операций со стеком, определяется битовым полем STKSZ регистра SYSCON, см. таблицу 3.15.

Таблица 3.15 – Трансформация адресов циркулярного стека регистра SYSCON

STKSZ	Размер стека, количество слов	Адрес DPRAM (по словам) физического стека	Значимые биты указателя стека SP
000_B	256	$00`FBFE_H$ – $00`FA00_H$ по умолчанию после сброса	SP.8 – SP.1
001_B	128	$00`FBFE_H$ – $00`FB00_H$	SP.7 – SP.1
010_B	64	$00`FBFE_H$ – $00`FB80_H$	SP.6 – SP.1
011_B	32	$00`FBFE_H$ – $00`FBC0_H$	SP.5 – SP.1
100_B	512	$00`FBFE_H$ – $00`F800_H$ Не использовать, если DPRAM 1 Кбайт!	SP.9 – SP.1
101_B	–	Зарезервировано	–
110_B	–	Зарезервировано	–
111_B	Вся область DPRAM	К виртуальному стеку не относится	SP.11 – SP.1

Адрес виртуального стека трансформируется в адрес физического стека конкатенацией значащих битов регистра SP с дополнительными старшими битами адреса верхней границы физического стека $00`FBFE_H$. Эта трансформация осуществляется аппаратно.

Значения регистров после сброса STKOV = $FA00_H$, STKUN = $FC00_H$, SP = $FC00_H$, STKSZ = 000_B, таким образом распределяют область стека, что создает возможность использования системного стека без каких-либо изменений, не выходя за границы 256 слов. Формирование физического адреса стека представлено на рисунке 3.26.

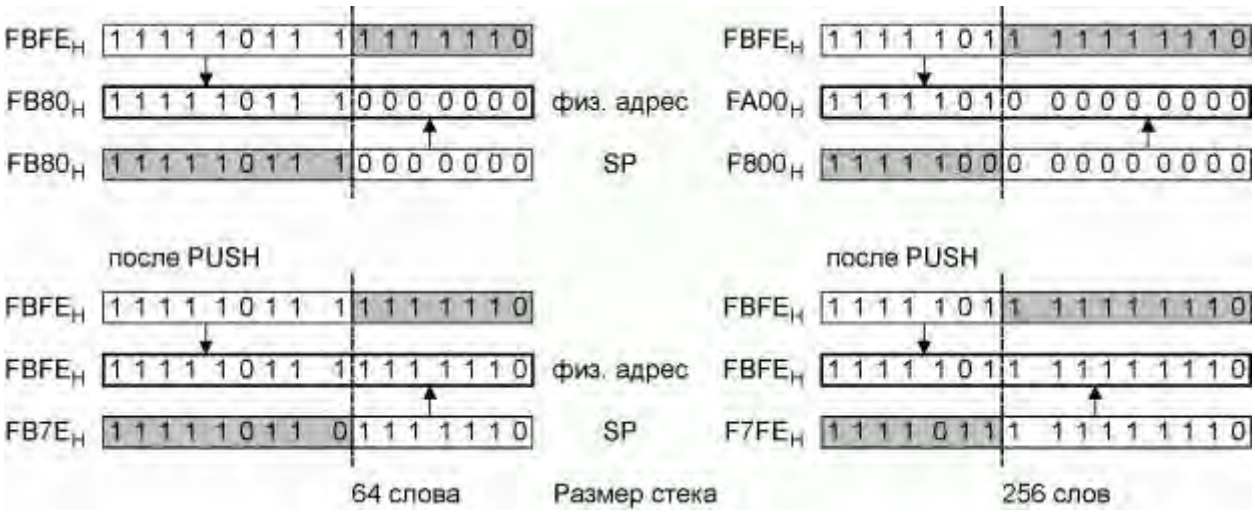


Рисунок 3.26 – Формирование физического адреса стека

Представленный ниже пример демонстрирует механизм циркулярного стека, который также осуществляет разбиение адресного пространства: сначала регистр R1 копируется в нижнюю часть физического стека, согласно выбранному максимальному размеру стека. Следующая команда занесет в верхнюю часть физического стека значение регистра R2, несмотря на то, что значение SP декрементируется на 2, аналогично предыдущей операции записи в стек.

```
MOV SP, #0F802h     ; Установка SP перед последним входом в стек (256 слов)
...                 ; (SP)=F802H: Адрес физического стека = FA02H
PUSH R1             ; (SP)=F800H: Адрес физического стека = FA00H
PUSH R2             ; (SP)=F7FEH: Адрес физического стека = FBFEH
```

Результатом трансформации адреса является то, что адрес стека перемещается по кругу от конца определенной области к началу. При автоматическом смещении и изменении стека, механизм виртуального стека требует переноса лишь той части стека, где находятся используемые данные (т. е. верхняя часть установленной области стека) вместо всей области стека. Данные, размещенные в нижней части внутреннего стека, не нуждаются в переносе при смещениях и изменениях стека, поскольку указатель стека обходит все адреса по кругу.

Примечание – Механизм циркулярного стека применим к стеку размером от 32 до 512 слов, STKSZ принимает значения от 000_B до 100_B. Механизм не работает при STKSZ = 111_B, когда для стека используется DPRAM.

При достижении границ стека, срабатывает ловушка переполнения/опустошения. При обслуживании ловушки, предопределенная область внутреннего стека переносится во внешний стек или из внешнего стека. Количество передаваемых данных определяется средним объемом области стека, который определяется программой и частотой появления вызовов, ловушек, прерываний и возвратов из прерываний. Во многих случаях это составляет от 1/4 до 1/10 размеров внутреннего стека. После завершения перемещения указатели границ обновляются для отражения распределенного по новому стека. Отсюда следует, что пользователь может записывать код вне зависимости от границ стека. На

программу пользователя может повлиять только время выполнения программы обслуживания ловушки.

Для использования циркулярного стека необходимо выполнить следующие действия:

- определить размер области физического стека в пределах DPRAM – битовое поле STKSZ регистра SYSCON;

- определить два указателя верхней и нижней границ внешнего стека. Эти значения затем проверяются на наличие переполнений и опустошений стека при пересылке данных;

- установить STKOV в значение, соответствующее границе определенной области внутреннего стека, плюс еще шесть слов (для резервирования пространства для сохранения двух входов прерываний).

После этого внутренний стек может быть изменен до достижения границы. После входа в подпрограмму обработки ловушки, вершина стека копируется во внешнюю память. После чего внутренний указатель изменяется для отражения нового адресного пространства. После выхода из подпрограммы, указатель внутреннего стека автоматически переходит на вершину стека и продолжает увеличиваться до достижения границы переполнения.

По достижении указателя опустошения, пока стек опустошается, дно стека загружается из внешней памяти и соответствующим образом обновляются внутренние указатели.

3.6 Обработка данных

Все стандартные арифметические, сдвиговые и логические операции производятся в 16-разрядном арифметико-логическом устройстве АЛУ. В дополнение к стандартному АЛУ микроконтроллер 1887ВЕ3Т включает в себя блок операций с битами (генератор битовых масок) и блок умножения и деления. Большинство внутренних блоков, выполняющих операции, оптимизированы для работы с 8-разрядными и 16-разрядными числами. После заполнения конвейера, большинство инструкций будут выполняться в течение одного машинного цикла.

Набор флагов автоматически обновляется в регистре PSW после каждой операции АЛУ. Наличие флагов позволяет совершать операции условного перехода по специальному условию. Поддержка как арифметики со знаком, так и беззнаковой арифметики обеспечивается с помощью определяемых пользователем условий перехода. Эти флаги также автоматически сохраняются при входе процессора в прерывание или обслуживание ловушки.

Типы данных

Микроконтроллер 1887ВЕ3Т поддерживает операции с битами, битовыми строками, символами и целочисленными типами данных. Большинство команд работает с определенными типами данных, см. таблицу 3.17.

Форматы данных поддерживают все типы данных ANSI C. Кроме того, некоторые Си-компиляторы поддерживают новые типы, которые позволяют эффективнее использовать команды.

Микроконтроллер 1887ВЕ3Т непосредственно поддерживает форматы данных ЦПУ, которые приведены в таблице 3.16.

Таблица 3.16 – Форматы данных центрального процессорного устройства

Формат данных ЦПУ	Размер	Диапазон
BIT	1 бит	0 или 1
BYTE	8 бит	от 0 до 255U[1)] или от –128 до 127
WORD	16 бит	от 0 до 65535U[1)] или от –32768 до 32767

[1)] U – беззнаковое число.

Таблица 3.17 – Типы данных ANSI C

Тип данных ANSI C	Размер	Диапазон	Формат данных ЦПУ
bit	1 бит	0 или 1	BIT
sfrbit	1 бит	0 или 1	BIT
esfrbit	1 бит	0 или 1	BIT
signed char	8 бит	от –127 до 128	BYTE
unsigned char	8 бит	от 0 до 255	BYTE
sfr	8 бит	от 0 до 65535	WORD
esfr	8 бит	от 0 до 65535	WORD
signed short	16 бит	от –32768 до 32767	WORD
unsigned short	16 бит	от 0 до 65535	WORD
bitword	16 бит	от 0 до 65535	WORD или BIT
signed int	16 бит	от –32768 до 32767	WORD
unsigned int	16 бит	от 0 до 65535	WORD
signed long	32 бит	от –2147483648 до 2147483647	Не поддерживает
unsigned long	32 бит	от 0 до 4294967295	Не поддерживает
float	32 бит	от $\pm1{,}176\times10^{-38}$ до $\pm3{,}402\times10^{38}$	Не поддерживает
double	64 бит	от $\pm2{,}225\times10^{-308}$ до $\pm1{,}797\times10^{308}$	Не поддерживает
long double	64 бит	от $\pm2{,}225\times10^{-308}$ до $\pm1{,}797\times10^{308}$	Не поддерживает
near pointer	16 бит	16 или 14 бит в зависимости от модели памяти	WORD
far pointer	32 бит	14 бит (16К) на любой странице	Не поддерживает
huge pointer	32 бит	24 бита (16М)	Не поддерживает
shuge pointer	32 бит	24 бита (16М), 16-битные арифметические операции	Не поддерживает

Константы

В дополнение к основным способам адресации микроконтроллер также поддерживает набор команд с использованием непосредственно констант длиной в байт или слово. Для оптимального использования кода программ, эти константы представлены в формате команды (находятся в поле команды) в виде 3, 4, 8 или 16 битов. Короткие константы всегда расширяются, в то время как длинные константы усекаются в случае необходимости, чтобы соответствовать формату данных, требуемому для определенных действий, см. таблицу 3.18.

Таблица 3.18 – Формат констант

Мнемоника	Операция со словом	Операция с байтом
#data3	0000_H + data3	00_H + data3
#data4	0000_H + data4	00_H + data4
#data8	0000_H + data8	Data8
#data16	data16	Data16 ^ FF_H
#mask	0000_H + mask	Mask
Примечание – Непосредственные константы всегда обозначают в начале знаком #.		

16-битный блок сложения/вычитания, циклического сдвига и 16-битный логический модуль

Все стандартные арифметические и логические операции производятся в 16-разрядном АЛУ. Для операций с байтами шестой и седьмой бит результата операции влияет на корректную установку флагов состояний. Большая точность вычислений обеспечивается путем установки флага CARRY-IN в АЛУ, предыдущей вычисленной частью операции.

16-разрядный генератор сдвига обеспечивает необходимое количество сдвигов за один такт. Также поддерживаются операции сдвига и циклического сдвига.

Блок операций с битами

Для обработки битов предназначено большое число команд. Использование этих команд обеспечивает эффективное управление и тестирование периферийного оборудования. В отличие от аналогичных команд других микроконтроллеров, эти команды обеспечивают постоянный доступ к двум операндам в побитно адресуемом пространстве памяти, без необходимости перемещения данных в промежуточные регистры.

Некоторые логические команды доступны как для совершения преобразований слов и байтов, так и поддерживают возможность преобразования битов. Это позволяет пользователям сравнивать и модифицировать биты регистров управления периферийными модулями с помощью одной команды. В систему команд для исключения длинных потоков команд для побитного изменения значений добавлены команды одновременного изменения значения нескольких битов. Эти операции выполняются за один машинный такт.

У метода есть некоторые особенности:

- биты могут быть изменены в пределах внутреннего адресного пространства, то есть в DPRAM и в SFR регистрах;
- команды не работают с битами, если они расположены во внешнем адресном пространстве;
- старшие 256 байт области SFR регистров, область ESFR регистров и DPRAM являются побитно адресуемыми, то есть, те биты, регистры которых расположены в пределах этой области, могут управляться непосредственно с помощью команд работы с битами;
- обращение к другим SFR регистрам должно быть побайтно или пословно.

Примечание – Все GPR, независимо от расположения банка регистров, побитно адресуемы, с помощью контекстного указателя CP. GPR, которые расположены вне области побитно адресуемой памяти, также имеют возможность побитной адресации.

Механизм чтение-изменение-запись может нежелательным образом сказаться на битах, показывающих аппаратное состояние системы, при обращении к этим битам. В этом случае микроконтроллер может аппаратно изменить необходимый бит во время совершения операции чтение-изменение-запись. При этом во время обратной записи возможна перезапись нового значения бита, сгенерированного микроконтроллером. Для решения этой проблемы, либо обеспечивается внутренняя защита, либо используется специальное программирование.

Защищенные биты не изменяют своего значения во время выполнения последовательности чтение-изменение-запись. Аппаратная логика защиты гарантирует, что операция обратной записи будет иметь эффект только со значащими битами.

Примечание – В случае попытки аппаратного доступа одновременно с программным изменением значения бита, более высокий приоритет имеет программный доступ к этому биту и определяет его конечное значение.

Блок умножения и деления

В состав ЦПУ входит отдельный блок умножения и деления. Умножение выполняется за пять машинных циклов, в то время как деление выполняется за десять

машинных циклов. Процесс умножения или деления может быть прерван по прерыванию более высокого уровня.

Старший регистр умножения/деления MDH

Этот регистр является частью 32-битного регистра умножения/деления, используемого ЦПУ при совершении операций умножения и деления. После умножения, этот не адресуемый побитно регистр содержит старшие 16 битов 32-битного результата. Для длинного деления, в MDH должны быть загружены старшие 16 битов 32-битного делимого до начала операции деления. После любого деления в регистре MDH содержится 16-битный остаток. Функциональное назначение полей регистра MDH представлено в таблице 3.19, формат регистра MDH – на рисунке 3.27 .

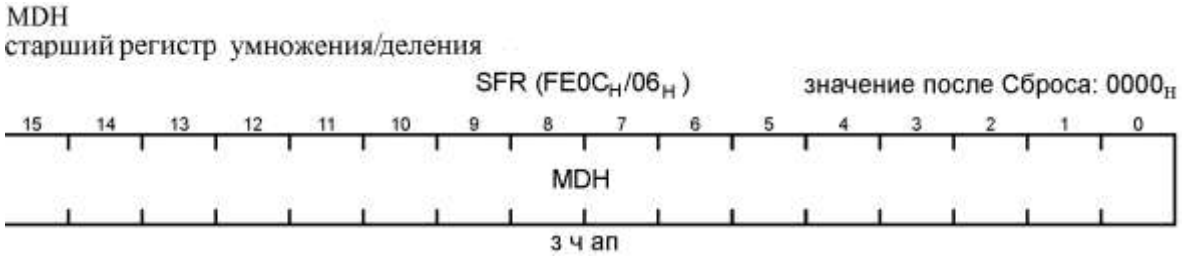


Рисунок 3.27 – Формат регистра MDH

Таблица 3.19 – Функциональное назначение поля регистра MDH

Поле	Биты	Тип	Описание
MDH	15–0	Чтение Запись Аппаратное влияние	Старшая часть MD: старшие 16 бит 32-битного регистра умножения и деления MD

Младший регистр умножения/деления MDL

Этот регистр является частью 32-битного регистра умножения/деления, используемого ЦПУ для совершения умножения или деления. После умножения, этот не адресуемый побитно регистр содержит младшие 16 битов 32-битного результата. Прежде чем начать операцию длинного деления, в MDL-регистр необходимо загрузить младшие 16 битов 32-битного делимого. После любого деления, регистр MDL содержит 16-битное частное. Функциональное назначение полей регистра MDL представлено в таблице 3.20, формат регистра MDL – на рисунке 3.28.

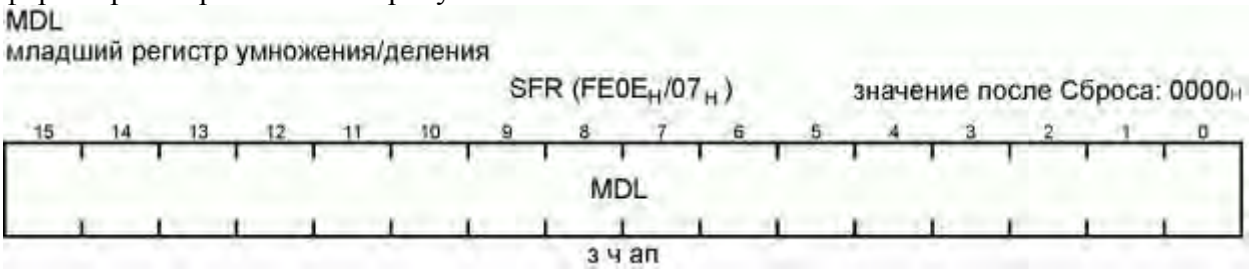


Рисунок 3.28 – Формат регистра MDL

Таблица 3.20 – Функциональное назначение полей регистра MDL

Поле	Биты	Тип	Описание
MDL	15–0	Чтение Запись Аппаратное влияние	Младшая часть MD: младшие 16 бит 32-битного регистра умножения и деления MD.

Как только программно изменяется значение регистра, флаг MDRIU регистра MDC устанавливается в «1». В случае программного чтения регистра MDL, флаг MDRIU сбрасывается в «0».

В случае прерывания операции умножения и деления, в том случае, когда в подпрограмме прерывания совершается умножение и деление, регистры MDH, MDL и MDC должны быть сохранены во избежание ошибочных результатов.

Контрольный регистр умножения/деления MDC

Этот адресуемый побитно 16-битный регистр используется ЦПУ во время совершения умножения или деления. Он используется для сохранения требуемой информации для управления операциями умножения и деления. Значение регистра MDC изменяется аппаратно, в течение каждого такта команды умножения и деления. Функциональное назначение полей регистра MDC представлено в таблице 3.21, формат регистра MDC – на рисунке 3.29.

Рисунок 3.29 – Формат регистра MDC

Таблица 3.21 – Функциональное назначение полей регистра MDC

Поле	Биты	Тип	Описание
MDRIU	4	Чтение Запись Аппаратное влияние	Флаг использования регистра умножения/деления: 0 Устанавливается при чтении регистра MDL. 1 Устанавливается при записи в регистр MDL и MDH, или когда выполняется операция умножения или деления.
!!	7, 6, 5, 3, 2, 1, 0	Чтение Запись Аппаратное влияние	Внутреннее состояние. Модуль умножения/деления использует эти биты для внутренних управляющих операций. Никогда не следует изменять значений этих битов без сохранения и восстановления значения регистра MDC!

Если в момент выполнения операции умножения или деления было совершено прерывание до окончания вычисления и модуль умножения и деления требуется в подпрограмме прерывания, то необходимо сохранить значения регистров MDC, MDH и MDL, чтобы была возможность возобновленовить прерванную операцию, а затем очистить регистр MDC, чтобы подготовить его к новому вычислению. После завершения нового умножения или деления необходимо восстановить состояние регистров.

Флаг MDRIU является той частью регистра MDC, которая представляет интерес для пользователя. Остальная часть регистра MDC зарезервирована для использования микроконтроллером, и пользователям не следует изменять ее значение в других случаях, отличных от описанных выше, иначе нельзя гарантировать корректного продолжения прерванных операций умножения или деления.

Знаковое или беззнаковое умножение или деление выбирается с помощью соответствующих команд умножения и деления. Результат сохраняется в регистре.

Флаг переполнения V устанавливается, если результат умножения или деления не может быть представлен в словесном типе данных (содержит больше 16 разрядов). Этот

флаг используется для определения необходимости копирования обоих слов результата из регистра MD. Сначала необходимо прочитать регистр MDH, чтобы гарантировать правильное отражение флага MDRIU. После чтения регистра MDL этот флаг сбрасывается.

Для выполнения умножения двух беззнаковых 16-битных чисел необходимо совершить следующую последовательность команд:

```
SAVE:
JNB MDRIU, START  ; Проверка использования MD
SCXT MDC, #0010H  ; Сохранение и очистка регистра управления, снятие
                  ; флага MDRIU (требуется только для прерываемых
                  ; команд)
BSET SAVED        ; Указание сохранения
PUSH MDH          ; Сохранение предыдущего значения MD в системном
                  ; стеке
PUSH MDL
START:
MULU R1, R2       ; Беззнаковое умножение 16*16 устанавливается флаг MDRIU
JMPR cc_NV,COPYL  ; Тестирование переполнения результата
MOV R3, MDH       ; Сохранение MDH в R3
COPYL:
MOV R4, MDL       ; Сохранение MDL в R4 и очистка MDRIU
RESTORE:
JNB SAVED, DONE   ; Проверка сохранения старого значения MD в стеке
POP MDL           ; Восстановление регистров
POP MDH
POP MDC
BCLR SAVED        ; Умножение завершено
                  ; Программа продолжается
DONE: ...
```

В этом примере необходимость в сохранении и восстановлении значений в стеке имеется в том случае, если эта последовательность действий является частью подпрограммы прерываний, которая прервала выполнение команд умножения или деления. По этой причине и регистр MDC так же необходимо сохранять. Старое значение регистра MDC должно быть прочитано из стека раньше, чем будет выполнена команда RETI.

Для совершения деления пользователь должен сначала записать делимое в регистр MD. В случае деления 16/16, необходимо произвести запись только в MDL. Результат сохраняется в регистре MD. При этом в MDL содержится целый результат деления, а в MDH – остаток от деления.

Следующая последовательность команд выполняет деление 32/16 бит:

```
MOV MDH, R1       ; Запись делимого в регистр MD
                  ; При этом устанавливается флаг MDRIU
MOV MDL, R2       ; Запись младшей половины делимого
DIV R3            ; Деление 32-битного MD на 16-битный R3
JMPR cc_V, ERROR  ; Проверка переполнения
MOV R3, MDH       ; Сохранения остатка
MOV R4, MDL       ; Сохранения целого результата в R4, сброс MDRIU
```

В том случае, когда команда умножения или деления прерывается во время исполнения, адрес прерванной команды сохраняется в стеке, и в регистре PSW

устанавливается флаг MULIP для подпрограммы прерываний. В момент выхода из подпрограммы прерываний по команде RETI, перед чтением из стека старого значения PSW, проверяется бит MULIP. Если этот бит установлен, команда умножения или деления читается из адреса, сохраненного в стеке, и завершается после выполнения команды RETI.

Примечание – Флаг MULIP – часть окружения прерываемой задачи. В том случае, когда после прерывания нет возврата в прерванную задачу (т. е. пользователь переключает на другую задачу), необходимо изменить флаг MULIP на значение, соответствующее новой задаче.

Регистр слова состояния процессора PSW

В этом побитно адресуемом регистре отражается текущее состояние микроконтроллера. Две группы битов отражают текущее состояние АЛУ и текущее состояние прерываний ЦПУ. Независимый бит USR0 регистра PSW предназначен для использования в качестве флага общего назначения. Функциональное назначение полей регистра PSW представлено в таблице 3.22, формат регистра PSW смотри на рисунке 3.30.

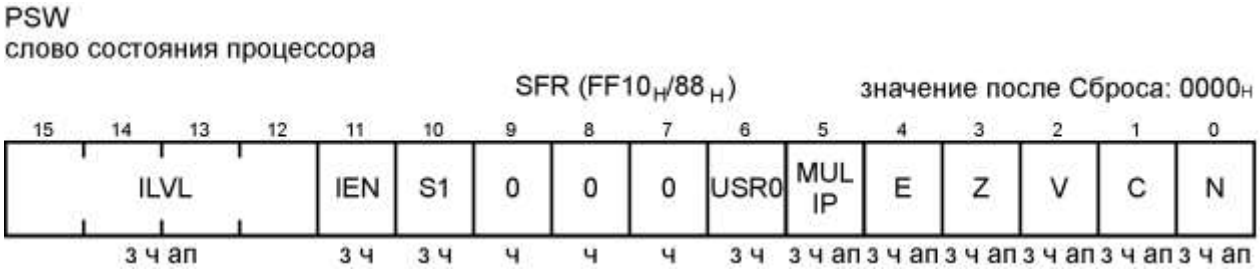


Рисунок 3.30 – Формат регистра PSW

Таблица 3.22 – Функциональное назначение полей регистра PSW

Поле	Биты	Тип	Описание
1	2	3	4
ILVL	15-12	Чтение Запись Аппаратное влияние	Уровень приоритетов ЦПУ: 0_H Самый низкий приоритет. … F_H Самый высокий приоритет.
IEN	11	Чтение Запись	Бит глобального разрешения прерываний (в том числе и PEC): 0 Запрет прерываний. 1 Разрешает прерывания.
S1	10	Чтение Запись	Зарезервировано для системы.
USR0	6	Чтение Запись Аппаратное влияние	Пользовательский флаг основного назначения.

Окончание таблицы 3.22

1	2	3	4
MULIP	5	Чтение	Флаг выполнения операции умножения и деления: 0 Отсутствие выполнения операции умножения и деления. 1 Умножение и деление были прерваны.
E	4	Чтение Запись Аппаратное влияние	Флаг конца таблицы: 0 Адрес исходного операнда команды не 8000_H и не 80_H. 1 Адрес исходного операнда команды 8000_H или 80_H.
Z	3	Чтение Запись Аппаратное влияние	Флаг нулевого результата: устанавливается, когда имеет место нулевой результат операции в АЛУ.
V	2	Чтение Запись Аппаратное влияние	Флаг переполнения результата вычислений: устанавливается, когда в результате операции в АЛУ имеет место переполнение.
C	1	Чтение Запись Аппаратное влияние	Флаг переноса: устанавливается, когда результат операции в АЛУ создает бит переноса.
N	0	Чтение Запись Аппаратное влияние	Флаг отрицательного результата: устанавливается, когда результат операции в АЛУ отрицателен.

Флаги состояния АЛУ: N, C, V, Z, E, MULIP

Флаги состояния АЛУ регистра PSW показывают состояние после выполнения последней операции в АЛУ. Эти флаги устанавливаются после большинства команд и зависят от специальных правил, зависящих от операции в АЛУ, или от операции перемещения данных. После выполнения команды, которая явно изменяет состояние PSW-регистра, флаги состояния не могут быть распознаны, как описано, так как явное чтение состояния PSW-регистра сменит значения флагов состояния, потому что они генерируются ЦПУ. Явное чтение PSW-регистра выдаст результат в виде значения, которое показывает состояние PSW-регистра после выполнения непосредственно предшествовавшей операции.

Примечание – После RESET все биты в регистре, отвечающие за состояние АЛУ, очищаются.

N-флаг: Для большинства операций с АЛУ, N-флаг устанавливается в единицу в том случае, если старший значащий бит в результате операции содержит «1», в противоположном случае флаг устанавливается в «0». В случае целочисленных (integer) операций, N-флаг может быть интерпретирован как бит знака в результате операции (отрицательный $N=1_B$, положительный $N=0_B$). Отрицательные числа всегда представляются в виде дополнительных чисел. Возможный диапазон чисел со знаком: в случае данных из одного слова – от минус 8000_H до плюс $7FFF_H$, или для однобайтных данных – от минус 80_H до плюс $7F_H$. При использовании логических битовых операций с одним операндом, N-флаг показывает предыдущее состояние изменяемого бита. Для логических битовых операций с двумя операндами N-флаг показывает результат операции исключающего «ИЛИ» с двумя изменяемыми битами.

C-флаг: После операции сложения он показывает наличие переноса из старшего значащего бита в вычисляемом байте или слове. После вычитания или сравнения C-флаг

показывает заимствованный бит, который представляет собой отрицательную логическую величину. Это означает, что С-флаг устанавливается в «1», если нет переноса из старшего значащего бита вычисляемого слова или байта данных во время операции вычитания, которая выполняется путем двух дополнительных сложений, и С-флаг устанавливается в «0», когда дополнительное сложение приводит к переносу. С-флаг всегда устанавливается в «0» для логических операций, операций умножения и деления потому, что эти операции не могут вызвать переноса ни при каких условиях. Для операций сдвига битов и циклического сдвига битов, в С-флаг подставляется значение бита, который был изменен в слове или байте данных последним. Если итог операции был «0», то С-флаг принимает нулевое значение. Для логических операций с битами с одним операндом С-флаг всегда принимает «0» значение. Для логических битовых операций с двумя операндами С-флаг принимает результат операции логического «И» с двумя соответствующими битами.

V-флаг: Для сложения, вычитания, операции с дополнительным кодом. V-флаг всегда принимает значение 1, если результат переходит граничные значения для знаковых чисел (для слов от минус 8000_H до плюс $7FFF_H$ или для байтов от минус 80_H до плюс $7F_H$). Заметим, что результат целочисленного сложения, целочисленного вычитания или операции с использованием дополнительного кода не является корректным, если V-флаг показывает арифметическое переполнение. Для умножения и деления V-флаг устанавливается в «1», если результат не может быть представлен в словесном типе данных. Заметим, что деление на ноль всегда приводит к переполнению. В противоположность результату деления, результат умножения правильный, несмотря на значение V-флага. Так как логические операции в АЛУ не могут нести неправильный результат, для этих операций V-флаг устанавливается в «0». V-флаг также используется как «приклеивающийся бит» для операций правого циклического сдвига и правого сдвига. Только с использованием С-флага ошибка сдвига, вызываемая командой правого сдвига, может быть оценена до величины результата половины LSB. Вместе с V-флагом, С-флаг позволяет оценивать ошибку сдвига с лучшим разложением. Для логических битовых операций с только одним операндом V-флаг всегда установлен в «0». Для логических битовых операций с двумя операндами V-флаг выставляет результат операции логического «ИЛИ» с двумя битами.

Оценка ошибки округления при сдвиге вправо представлена в таблице 3.23.

Таблица 3.23 – Оценка ошибки округления при сдвиге вправо

С-флаг	V-флаг	Величина ошибки вращения
0	0	Нет ошибки вращения
0	1	0 < ошибка вращения < 1/2 LSB
1	0	ошибка вращения = 1/2 LSB
1	1	ошибка вращения > 1/2 LSB

Z-флаг: Z-флаг установлен в «1», если результатом операции в АЛУ является нулевая величина. Z-флаг всегда устанавливается в «1» для сложения и вычитания с переносом в том случае, когда одновременно Z-флаг уже содержит «1» и результат текущей операции в АЛУ тоже равен нулю. Этот механизм обеспечивает поддержку вычислений с большой точностью. Для логических битовых операций только с одним операндом в Z-флаг помещается логическое отрицание изменяемого бита. Для логических битовых операций с двумя операндами в Z-флаг помещается результат логической операции NOR над двумя отмеченными битами.

E-флаг: E-флаг может быть изменен командой, совершающей операцию в АЛУ или совершающей перемещение данных. E-флаг устанавливает «0» для тех операндов, которые не могут быть использованы при табличном поиске. Во всех других случаях E-флаг устанавливается в зависимости от значения исходного операнда, иными словами, по достижении конца поиска в таблице. Если значение исходного операнда команды

равно минимальному отрицательному числу (8000_H для слов данных или 80_H для байтов данных), E-флаг устанавливается в «1».

MULIP-флаг: MULIP-флаг аппаратно устанавливается в «1» при входе в подпрограмму обслуживания прерывания, в случае прерывания операции умножения или деления в АЛУ. В зависимости от состояния этого бита, микроконтроллер решает, продолжать или не продолжать умножение или деление после завершения обслуживания прерывания. Значение бита MULIP перезаписывается из стека при выполнении команды возврата из прерывания RETI. Обычно это означает, что MULIP-флаг снова после этого принимает нулевое значение.

Примечание – Когда подпрограмма обслуживания прерывания не осуществляет возврата в прерванную команду умножения или деления (т. е. в случае использования таблицы задач, переключающейся между независимыми задачами), MULIP-флаг должен оставаться установленным, и таким образом должна осуществляться перезапись для новой задачи.

Флаги состояния прерывания ЦПУ: IEN, ILVL

Бит разрешения прерываний глобально разрешает IEN = 1_B или запрещает IEN = 0_B прерывания. Четырехбитное поле уровней прерываний ILVL отмечает уровень приоритета текущего состояния ЦПУ. Уровень прерывания ЦПУ изменяется аппаратно при входе в подпрограмму обслуживания прерываний, также для запрещения прерываний ILVL может быть программно изменен. В случае присвоения ЦПУ максимально возможного 15-го уровня, текущая деятельность ЦПУ не может быть прервана, за исключением аппаратных ловушек и внешних немаскируемых прерываний. Подробности рассматриваются в разделе 10 «Система прерываний и ловушек».

После RESET все прерывания запрещены, и ЦПУ присвоен самый низкий уровень приоритета ILVL = 0_B.

3.7 Конвейерная обработка команд

Обработка команд производится на 4-ступенчатом конвейере. Каждая ступень имеет свой набор выполняемых операций.

Конвейер микроконтроллера состоит из следующих этапов:

- выборка;
- декодирование;
- выполнение;
- запись результата.

На этапе выборки производится чтение команд из внутреннего ПЗУ, ОЗУ или внешней памяти в соответствии с адресом, формируемым из указателя команд IP и указателя сегмента кода CSP. Далее производится декодирование команд и, если это необходимо, вычисление адреса операнда и выборка его из памяти. Для всех команд, которые неявно производят доступ к системному стеку, значение указателя вершины стека либо уменьшается, либо увеличивается. Команды перехода производят запись адреса перехода в регистры IP и CSP (для команд условного перехода только в том случае, если указанное условие перехода является истинным). После декодирования, команда передается на этап выполнения, на котором осуществляется исполнение требуемой операции с предварительно выбранными операндами. Флаги слова состояния ЦПУ (регистр PSW) обновляются в соответствии с результатом операции. Также на этом этапе производится запись в области регистров SFR или ESFR и выполнение записи с инкрементом/декрементом в регистры общего назначения, значение которых используется для косвенной адресации. Результат выполнения, если требуется, записывается во внутреннюю или внешнюю память на последнем этапе конвейера.

Особенностью конвейера являются, так называемые, инжектируемые команды (injected instruction), которые автоматически формируются самим ЦПУ для правильного функционирования конвейера. Инжектированные команды необходимы для обработки

запросов на прерывание, выполнения переходов и завершения команд, которые не могут быть выполнены за один машинный цикл. Эти инжектируемые команды автоматически вставляются на этапе декодирования и проходят оставшиеся этапы конвейера, как и остальные команды. Управление инжектируемыми командами осуществляется только аппаратно и недоступно для программы. Поэтому необходимо принимать в рассмотрение время, необходимое для их выполнения. Это особенно важно для определения времени реакции на внешний запрос.

Влияние конвейера на работу ЦПУ

Для уменьшения времени выполнения команд центральным процессорным устройством используется 4-ступенчатый конвейер. Увеличение быстродействия потребовало применения дополнительных аппаратных средств, исключающих возникновение большого числа конфликтных ситуаций. При этом все же существуют программные конструкции, требующие особого внимания, описание которых приведено ниже.

Изменение указателя контекста CP

Микроконтроллер позволяет организовать несколько банков регистров общего назначения GPR. Адрес начала активного банка GPR содержится в указателе контекста CP. Для переключения между банками GPR значение регистра CP необходимо изменить. После любой команды, изменяющей значение CP, необходимо выполнить, по крайней мере, одну команду, не использующую для доступа к регистрам нового банка GPR короткие виды адресации (8-, 4-, 2-разрядную и битовую адресации). В подпрограммах обработки запросов на прерывание для этого удобно использовать команду SCXT, которая перед обновлением сохраняет на системном стеке содержимое регистра SFR/ESFR, а в данном случае – указателя контекста CP.

Примечание – В связи с тем, что адресация регистров общего назначения является базовой, причем значение базы (указатель контекста CP) может меняться, необходимо следить за тем, чтобы не происходило обращения за пределы IRAM.

Пример:

```
                    ; Сохранение и задание нового значения базового адреса контекста
scxt CP, #0FC00h
                    ; Должна быть любая инструкция, не использующая GPR
…
                    ; Запись в GPR нового значения
mov r0, #data
```

Изменение указателей страниц данных DPP0, …, DPP3

Указатели DPP0, …, DPP3 служат для хранения номеров страниц, которые используются для полной 16-разрядной адресации данных (непосредственно-косвенной или косвенной). Номер используемого регистра DPPx определяется по значению двух старших разрядов 16-разрядного адреса, указанного непосредственно или косвенно в команде. После любой команды изменения DPPx необходимо выполнить, по крайней мере, одну команду, не использующую для доступа к данным этот DPPx.

Пример:

```
                    ; Выбор 4-й страницы данных через DPP0.
mov DPP0, #4
                    ; Любая инструкция, не использующая DPP0.
…
                    ; Номер используемого регистра DPPx определяется по значению
                    ; двух старших разрядов адреса данных. Значение 00B соответствует
                    ; указателю страницы DPP0, содержимое R1 пересылается по адресу
                    ; 01'0000H (4-я страница)
mov 0000h, r1
```

Изменение указателя стека SP

Команды RET, RETI, RETS, RETP и POP будут выполняться некорректно, если предыдущая команда изменила значение SP. Для правильной работы необходимо, чтобы после изменения содержимого SP выполнилась, по крайней мере, одна команда, не использующая системный стек.

Пример:

```
                    ; Загрузка нового значения указателя вершины стека.
mov SP, #0FA40h
                    ; Любая инструкция, не использующая стек.
…
pop r0              ; Загрузка R0 значением, снятым с вершины стека.
```

При записи данных на системный стек командами PUSH, CALLI, CALLA, CALLS и SCXT после изменения SP также возникают конфликты. Однако они решаются внутренней логикой.

Доступ к внешней шине

В конвейере параллельно выполняются 4 команды, и на каждом этапе возможно обращение к внешней памяти. Если эти запросы поступают к контроллеру внешней шины EBC одновременно с нескольких этапов конвейера, то они обрабатываются в следующей последовательности:

- запись данных;
- выборка команды;
- чтение данных.

Запись данных имеет наивысший приоритет, а чтение – самый низший.

Управление прерываниями

Явное или неявное изменение значения слова состояния ЦПУ (регистр PSW) происходит только на этапе выполнения соответствующей команды. Поэтому после любой команды, изменяющей PSW, необходимо выполнить, по крайней мере, одну команду, выполнение которой не зависит от флагов АЛУ, уровня приоритета текущей задачи ILVL, бита IEN, отвечающего за разрешение обработки маскируемых запросов на прерывания и пр. Если необходимо разрешить или запретить обработку маскируемых запросов на прерывания, то после команды, изменяющей значение PSW, необходимо выполнить минимум одну команду, которая (или которые) не критична к обработкам запросов.

Пример:

```
                    ; Запрещение обработки всех маскируемых запросов.
bclr IEN
                    ; Инструкция в группе, которая может быть прервана.
In-1
                    ; Первая инструкция непрерываемой последовательности.
In
…
                    ; K – первая инструкция непрерываемой последовательности.
In+k-1
                    ; Разрешение обработки всех запросов.
bset IEN
                    ; K-ая инструкция непрерываемой последовательности.
In+k
                    ; Первая инструкция в группе, которая может быть прервана.
In+k+1
```

Инициализация портов

Настройка портов микроконтроллера на ввод или вывод осуществляется путем обнуления или установки в единичное значение соответствующего бита в регистре DPx. Изменение отдельных битов использует внутреннюю последовательность «чтение-изменение-запись» (Read-Modify-Write), которая обращается целиком ко всему регистру. Поэтому команда, выполняющая настройку канала (каналов), и команда, использующая обращение к этому порту, должны быть отделены друг от друга командой, которая не использует этот порт. В противном случае возможно чтение неверного значения канала.

Пример:

```
                ; НЕПРАВИЛЬНО:
                ; Установка канала P3.13 на вывод.
bset DP3.13
                ; Установка P3.5 в значение 1b. ВНИМАНИЕ! В результате
                ; выполнения последовательности «чтение-изменение-запись»
                ; прочитается значение с канала P3.13, и это значение записывается в
                ; выходной триггер P3.13.
bset P3.5
                ; ПРАВИЛЬНО:
                ; Установка канала P3.13 на вывод.
bset DP3.13
                ; Любая инструкция, которая не обращается к порту P3.
…
                ; Установка P3.5 в значение 1b. «Чтение-изменение-запись» также
                ; прочитается значение P3.13, но не с входа, а из выходного триггера.
                ; Прочитанное значение запишется обратно, не изменив состояния на
                ; выходе P3.13.
bset P3.5
```

Запись значения канала всегда производится в выходной триггер, для установки значения канала не требуется дополнительная команда, направление этого канала изменяли предыдущей командой, и нет обращения к этому порту в следующей команде.

Оба примера, приведенные ниже, являются корректными.

Пример:

```
                ; Вариант 1
                ; Установка канала P3.13 на вывод.
bset DP3.13
                ; Установка P3.13 в значение 1b.
bset P3.13
                ; Вариант 2
                ; Установка канала P3.13 на вывод.
bset DP3.13
                ; Любая инструкция, которая не обращается к порту P3.
…
                ; Установка P3.13 в значение 1b.
bset P3.13
…
```

Следует обратить внимание, что последовательность команд, которая сначала устанавливает направление на вывод, а затем выходное значение, формально является правильной, и выходное значение будет соответствовать заданному значению. Однако рекомендуется сначала установить значение канала бит Px.y, и только затем установить направление на вывод бит DPx.y. Это необходимо для того, чтобы исключить на выходе нежелательный импульс.

После программного изменения направления, но перед использованием значения канала (чтение или запись), необходимо вставить одну или несколько команд, не использующих этот порт.

Пример:

```
                ; Установка канала P3.13 на ввод.
bclr DP3.13
                ; Любая инструкция, не использующая порт ввода-вывода.
…
                ; Переход, если бит P3.13 установлен.
jb P3.13, label
```

В приведенном примере, при отсутствии дополнительной команды, чтение будет произведено не с входа канала, а с выходного триггера, т. к. на момент чтения из-за обработки команды на конвейере канал все еще настроен на вывод (если был настроен на вывод до этого).

Изменение системной конфигурации

Требования к командам, изменяющим значение регистра системной конфигурации SYSCON, аналогичны изложенным выше. После любой команды изменения системной конфигурации следующая команда не должна использовать ресурсы, параметры которых изменяются. Например, команда, устанавливающая режим сегментированной или несегментированной выборки кода, должна выполняться из нулевого сегмента, чтобы не произошло ложного перехода в другой сегмент, и последующие команды выполнились корректно. Разрешение или запрещение обращения к внутреннему ПЗУ (бит ROMEN) или переадресация ПЗУ в первый сегмент (бит ROMS1) должны исполняться из внутреннего ОЗУ (PSRAM или XRAM) или внешней памяти по адресам, которые не перекрываются с внутренним ПЗУ.

Изменение регистров управления внешней шиной BUSCONx – ADDRSELx

Параметры обращения по внешней шине во время доступа к адресному окну определяются парой регистров BUSCONx и ADDRSELx. Изменения параметров в регистре BUSCONx (кроме временных) и размера и/или начального адреса в регистре ADDRSELx должны выполняться командами, выборка которых осуществляется из другого адресного окна или внутренней памяти. Кроме того, следующая за изменением команда не должна обращаться к адресному окну, параметры которого изменялись.

Допускается изменение временных параметров в регистре BUSCONx с применением команд, выборка которых производится из адресного окна BUSCONx – ADDRSELx – CSx#. Однако при этом необходимо следить, чтобы все временные характеристики обращения соответствовали требуемым значениям для используемой внешней памяти или устройства.

Примечание – Сначала следует установить значение регистра ADDRSELx, а затем соответствующее ему BUSCONx.

3.8 Специализированные регистры CSFR

Регистр постоянного нуля ZEROS

Все биты адресуемого побитно регистра ZEROS аппаратно зафиксированы на нуле. Этот регистр можно только читать.

Регистр ZEROS может быть использован для константы нуля (адресуемой как к регистру), т. е. для использования в операциях с битами или создания масок. Регистр может быть доступен с помощью любой команды, которая может адресоваться к регистру области SFR. Функциональное назначение поля регистра ZEROS представлено в таблице 3.24, формат регистра ZEROS описан на рисунке 3.31.

ZEROS
регистр постоянного нуля

SFR ($FF1C_H/8E_H$) значение после Сброса: 0000_H

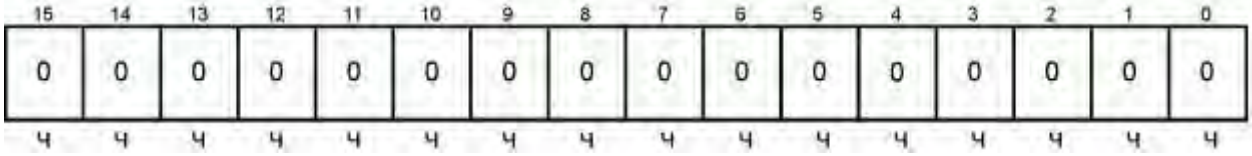


Рисунок 3.31 – Формат регистра ZEROS

Таблица 3.24 – Функциональное назначение поля регистра ZEROS

Поле	Биты	Тип	Описание
0	15-0	Чтение	Все биты постоянно установлены в «0»

Регистр постоянной единицы ONES

Все биты этого адресуемого побитно регистра аппаратно зафиксированы на единице. Этот регистр можно только читать.

Регистр ONES может быть использован для константы единицы (адресуемой как к регистру), т. е. для использования в операциях с битами или создания масок. Регистр может быть доступен с помощью любой команды, которая может быть адресована к регистру области SFR. Функциональное назначение полей регистра ONES представлено в таблице 3.25, формат регистра ONES описан на рисунке 3.32.

ONES
регистр постоянной единицы

SFR ($FF1E_H/8F_H$) значение после Сброса: 0000_H

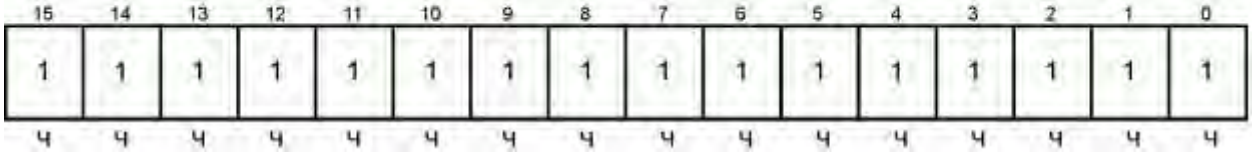


Рисунок 3.32 – Формат регистра ONES

Таблица 3.25 – Функциональное назначение поля регистра ONES

Поле	Биты	Тип	Описание
1	15-0	Чтение	Все биты постоянно установлены в «1»

Регистр идентификации ЦПУ CPUID

16-битный регистр идентификации ЦПУ CPUID содержит номер и версию микроконтроллера 1887ВЕ3Т. Функциональное назначение полей регистра CPUID представлено в таблице 3.26, формат данного регистра – на рисунке 3.33.

CPUID
регистр идентификации ЦПУ

ESFR ($F00C_H/06_H$) значение после Сброса: 0420_H

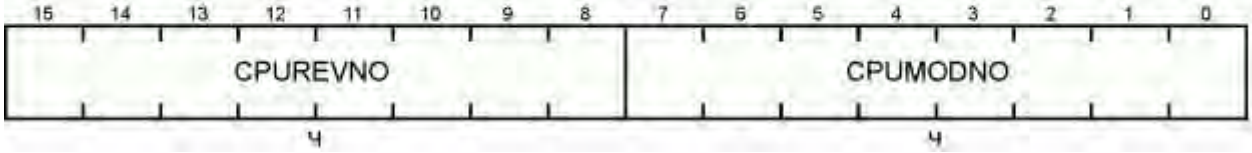


Рисунок 3.33 – Формат регистра CPUID

Таблица 3.26 – Функциональное назначение полей регистра CPUID

Поле	Биты	Тип	Описание
CPUREVNO	15–8	Чтение	Номер чипа, по умолчанию 04_H
CPUMODNO	7–0	Чтение	Версия чипа, по умолчанию 20_H

3.9 Сводное описание регистров ЦПУ

Существует два типа регистров ЦПУ: регистры общего назначения – GPR и регистры специальных функций (центрального процессора) – CSFR. Регистры GPR используются контроллером при арифметических и логических операциях, а также для указания адреса при косвенной адресации. Регистры CSFR используются для управления.

Регистры общего назначения

Режимы адресации для доступа к GPR (слово) регистров общего назначения представлены в таблице 3.27. Все регистры общего назначения адресуемы побитно.

Таблица 3.27 – Режимы адресации для доступа к GPR (слово)

Имя	Физический адрес	8-битный адрес	4-битный адрес	Описание	Значение после сброса
R0	(CP)+0	$F0_H$	0_H	Регистр R0 (слово)	$UUUU_H$
R1	(CP)+2	$F1_H$	1_H	Регистр R1 (слово)	$UUUU_H$
R2	(CP)+4	$F2_H$	2_H	Регистр R2 (слово)	$UUUU_H$
R3	(CP)+6	$F3_H$	3_H	Регистр R3 (слово)	$UUUU_H$
R4	(CP)+8	$F4_H$	4_H	Регистр R4 (слово)	$UUUU_H$
R5	(CP)+10	$F5_H$	5_H	Регистр R5 (слово)	$UUUU_H$
R6	(CP)+12	$F6_H$	6_H	Регистр R6 (слово)	$UUUU_H$
R7	(CP)+14	$F7_H$	7_H	Регистр R7 (слово)	$UUUU_H$
R8	(CP)+16	$F8_H$	8_H	Регистр R8 (слово)	$UUUU_H$
R9	(CP)+18	$F9_H$	9_H	Регистр R9 (слово)	$UUUU_H$
R10	(CP)+20	FA_H	A_H	Регистр R10 (слово)	$UUUU_H$
R11	(CP)+22	FB_H	B_H	Регистр R11 (слово)	$UUUU_H$
R12	(CP)+24	FC_H	C_H	Регистр R12 (слово)	$UUUU_H$
R13	(CP)+26	FD_H	D_H	Регистр R13 (слово)	$UUUU_H$
R14	(CP)+28	FE_H	E_H	Регистр R14 (слово)	$UUUU_H$
R15	(CP)+30	FF_H	F_H	Регистр R15 (слово)	$UUUU_H$

Первые 8 регистров из GPR R0, … , R7 также доступны побайтно. При этом запись в адресуемый байт регистра не оказывает влияние на содержимое второго байта того же регистра.

Каждая половина регистра-байта (8 бит) имеет свое уникальное имя. Режимы адресации для доступа к GPR (байт) представлены в таблице 3.28.

Таблица 3.28 – Режимы адресации для доступа к GPR (байт)

Имя	Физический адрес	8-битный адрес	4-битный адрес	Описание	Значение после сброса
RL0	(CP)+0	$F0_H$	0_H	Регистр RL0 (байт)	UU_H
RH0	(CP)+1	$F1_H$	1_H	Регистр RL1 (байт)	UU_H
RL1	(CP)+2	$F2_H$	2_H	Регистр RL2 (байт)	UU_H
RH1	(CP)+3	$F3_H$	3_H	Регистр RL3 (байт)	UU_H
RL2	(CP)+4	$F4_H$	4_H	Регистр RL4 (байт)	UU_H
RH2	(CP)+5	$F5_H$	5_H	Регистр RL5 (байт)	UU_H
RL3	(CP)+6	$F6_H$	6_H	Регистр RL6 (байт)	UU_H
RH3	(CP)+7	$F7_H$	7_H	Регистр RL7 (байт)	UU_H
RL4	(CP)+8	$F8_H$	8_H	Регистр RL8 (байт)	UU_H
RH4	(CP)+9	$F9_H$	9_H	Регистр RL9 (байт)	UU_H
RL5	(CP)+10	FA_H	A_H	Регистр RL10 (байт)	UU_H
RH5	(CP)+11	FB_H	B_H	Регистр RL11 (байт)	UU_H
RL6	(CP)+12	FC_H	C_H	Регистр RL12 (байт)	UU_H
RH6	(CP)+13	FD_H	D_H	Регистр RL13 (байт)	UU_H
RL7	(CP)+14	FE_H	E_H	Регистр RL14 (байт)	UU_H
RH7	(CP)+15	FF_H	F_H	Регистр RL15 (байт)	UU_H

Регистры CSFR специального назначения, упорядоченные по имени

В таблице 3.29 перечислены регистры CSFR в алфавитном порядке. Бит-адресуемые CSFR регистры отмечены символом «b» в столбце «Имя». CSFR в пределах расширенного пространства CSFR (ECSFR) отмечены символом «E» в столбце «8-битный адрес».

Таблица 3.29 – Регистры специального назначения CSFR, упорядоченные по имени

Имя	Физический адрес	8-битный адрес	Описание	Значение после сброса
CP	$FE10_H$	08_H	Контекстный указатель	$FC00_H$
CPUID	$F00C_H$	$E-06_H$	Идентификатор ЦПУ	0420_H
CSP	$FE08_H$	04_H	Указатель сегмента кода (старшие 8 бит, не доступно для прямой записи)	0000_H
DPP0	$FE00_H$	00_H	Указатель 0-й страницы данных (10 бит)	0000_H
DPP1	$FE02_H$	01_H	Указатель 1-й страницы данных (10 бит)	0001_H
DPP2	$FE04_H$	02_H	Указатель 2-й страницы данных (10 бит)	0002_H
DPP3	$FE06_H$	03_H	Указатель 3-й страницы данных (10 бит)	0003_H
MDC b	$FF0E_H$	87_H	Контрольный регистр умножения/деления	0000_H
MDH	$FE0C_H$	06_H	Старший регистр умножения/деления	0000_H
MDL	$FE0E_H$	07_H	Младший регистр умножения/деления	0000_H
ONES b	$FF1E_H$	$8F_H$	Регистр постоянной единицы (только чтение)	$FFFF_H$
PSW b	$FF10_H$	88_H	Слово состояния процессора	0000_H
SP	$FE12_H$	09_H	Указатель стека	$FC00_H$
STKOV	$FE16_H$	$0A_H$	Указатель переполнения стека	$FA00_H$
STKUN	$FE16_H$	$0B_H$	Указатель опустошения стека	$FC00_H$
SYSCON	$FF12_H$	89_H	Регистр управления системой	*
TFR b	$FFAC_H$	$D6_H$	Регистр флагов ловушек	0000_H
ZEROS b	$FF1C_H$	$8E_H$	Регистр постоянного нуля, только чтение	0000_H

_____

* Значение после сброса 00000XX0 $X0000000_B$.

Регистры CSFR специального назначения, упорядоченные по адресу

В таблице 3.30 перечислены все CSFR регистры по их физическим адресам. Бит-адресуемые CSFR отмечены символом «b» в столбце «Имя». CSFR в пределах расширенного SFR-пространства (ESFR) отмечены символом «E» в столбце «8-битный адрес».

Таблица 3.30 – Регистры CSFR специального назначения, упорядоченные по адресу

Имя	Физический адрес	8-битный адрес	Описание	Значение после сброса
CPUID	$F00C_H$	$E-06_H$	Идентификатор ЦПУ	0420_H
DPP0	$FE00_H$	00_H	Указатель 0-й страницы данных (10 бит)	0000_H
DPP1	$FE02_H$	01_H	Указатель 1-й страницы данных (10 бит)	0001_H
DPP2	$FE04_H$	02_H	Указатель 2-й страницы данных (10 бит)	0002_H
DPP3	$FE06_H$	03_H	Указатель 3-й страницы данных (10 бит)	0003_H
CSP	$FE08_H$	04_H	Указатель сегмента кода	0000_H
MDH	$FE0C_H$	06_H	Старший регистр умножения/деления	0000_H
MDL	$FE0E_H$	07_H	Младший регистр умножения/деления	0000_H
CP	$FE10_H$	08_H	Указатель контекста	$FC00_H$
SP	$FE12_H$	09_H	Указатель стека	$FC00_H$
STKOV	$FE14_H$	$0A_H$	Указатель переполнения стека	$FA00_H$
STKUN	$FE16_H$	$0B_H$	Указатель опустошения стека	$FC00_H$
MDC b	$FF0E_H$	87_H	Контрольный регистр умножения/деления	0000_H
PSW b	$FF10_H$	88_H	Слово состояния процессора	0000_H
SYSCON	$FF12_H$	89_H	Регистр управления системой	*
ZEROS b	$FF1C_H$	$8E_H$	Регистр постоянного значения нуля (только чтение)	0000_H
ONES b	$FF1E_H$	$8F_H$	Регистр постоянного значения единицы, только чтение	$FFFF_H$
TFR b	$FFAC_H$	$D6_H$	Регистр флагов ловушек	0000_H

_____

* Значение после сброса 00000XX0 $X0000000_B$.

Регистры XSFR прерывания микроконтроллера и контроллера периферийных событий PEC

В таблице 3.31 перечислены все регистры XSFR, которые осуществляют прерывания микроконтроллера и контроллера периферийных событий PEC.

Таблица 3.31 – Регистры XSFR прерывания микроконтроллера и контроллера периферийных событий PEC

Имя регистра	Описание регистра	Модуль блока
PECSN0	Регистр указатель 0 адреса сегмента PEC	Указатель PEC
PECSN1	Регистр указатель 1 адреса сегмента PEC	Указатель PEC
PECSN2	Регистр указатель 2 адреса сегмента PEC	Указатель PEC
PECSN3	Регистр указатель 3 адреса сегмента PEC	Указатель PEC
PECSN4	Регистр указатель 4 адреса сегмента PEC	Указатель PEC
PECSN5	Регистр указатель 5 адреса сегмента PEC	Указатель PEC
PECSN6	Регистр указатель 6 адреса сегмента PEC	Указатель PEC
PECSN7	Регистр указатель 7 адреса сегмента PEC	Указатель PEC
PECC0	Регистр управления PEC канала 0	Управление PEC
PECC1	Регистр управления PEC канала 1	Управление PEC
PECC2	Регистр управления PEC канала 2	Управление PEC
PECC3	Регистр управления PEC канала 3	Управление PEC
PECC4	Регистр управления PEC канала 4	Управление PEC
PECC5	Регистр управления PEC канала 5	Управление PEC
PECC6	Регистр управления PEC канала 6	Управление PEC
PECC7	Регистр управления PEC канала 7	Управление PEC
IRQ0IC	Регистр управления прерываний 0	Арбитражное управление
…	…	…
IRQ111IC	Регистр управления прерываний 111	Арбитражное управление

В данном руководстве полное описание регистров областей SFR и ESFR представлено в приложении Б в таблице Б.1, упорядоченных по физическим адресам, в приложении В в таблице В.1 – упорядоченных по именам регистров, в приложении Г в таблицах Г.1 – Г.20 – упорядоченных в зависимости от отношения к блоку микроконтроллера.

4 Организация памяти

Пространство памяти ИС 1887ВЕ3Т организовано по принципу архитектуры Фон-Неймана. Это означает, что программный код и данные содержатся в одном и том же линейном адресном пространстве. Все физически разделенные области памяти, включая внутреннюю память программ Flash (интегрированную), DPRAM, внутренние регистры специального назначения SFR, расширенные регистры специального назначения ESFR и внешнюю память расположены в одном общем адресном пространстве.

Микросхема 1887ВЕ3Т обеспечивает общее адресное пространство памяти 16 Мбайт. Это адресное пространство размещается в 256 сегментов по 64 Кбайт каждый, и каждый сегмент, кроме того, подразделяется на четыре страницы данных по 16 Кбайт каждая, см. рисунок 4.1.

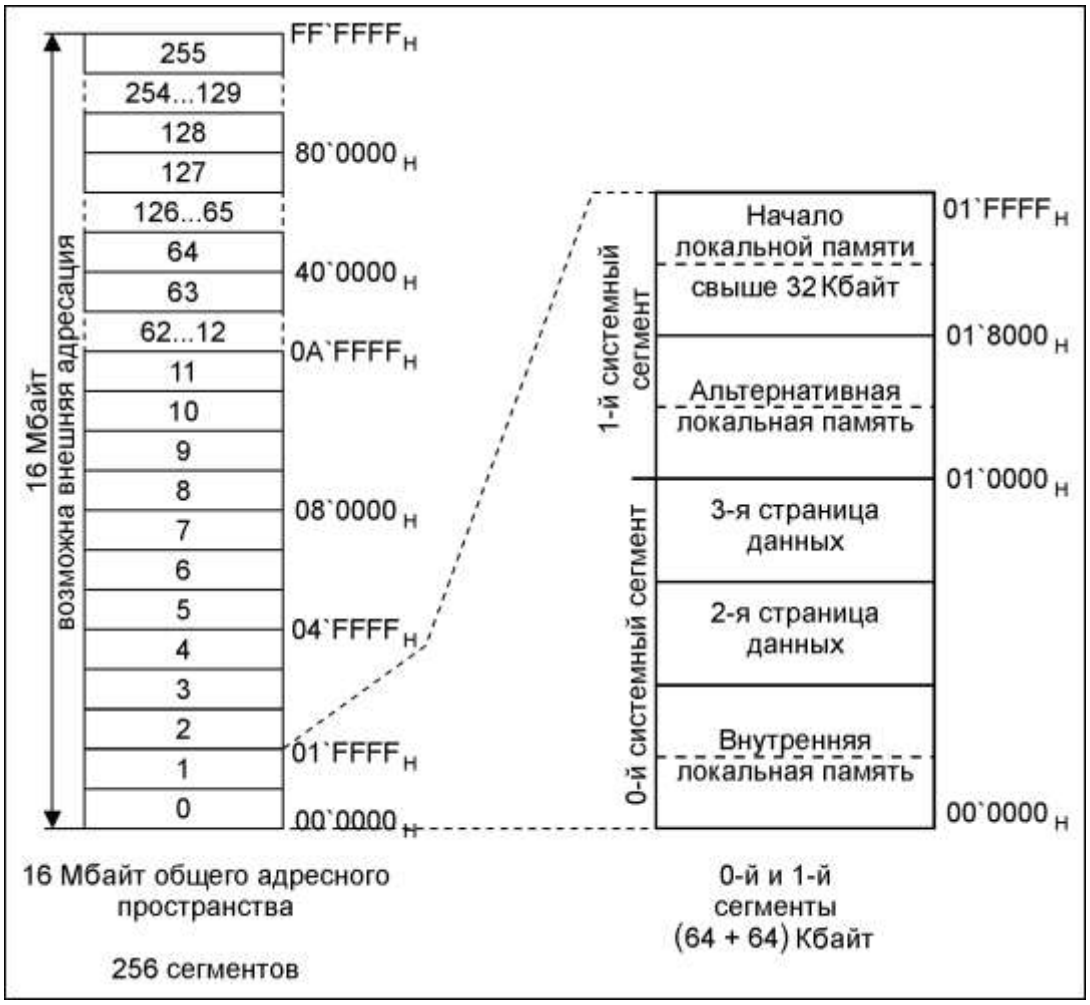


Рисунок 4.1 – Область памяти и адресное пространство

Большая часть внутренней памяти отражена в нулевом (системном) сегменте. Верхние 4 Кбайт нулевого сегмента, адреса $00'F000_H – 00'FFFF_H$, отведены под SFR, ESFR и DPRAM. Младшие 32 Кбайт нулевого сегмента, адреса $00'0000_H – 00'7FFF_H$, отведены под первую часть Flash-памяти, вторая часть Flash-памяти занимает адреса $01'8000_H – 04'FFFF_H$. Область памяти в 2 Кбайт, отведенная под XSFR, занимает адреса $00'E800_H – 00'EFFE_H$. Оперативная память программ/данных PSRAM занимает адреса $05'0000_H – 05'1FFF_H$ и составляет 8 Кбайт. Оперативная память данных XRAM объемом 4 Кбайт занимает адреса $00'D800_H – 00'E7FF_H$.

Коды команд и данные могут сохраняться в любой части памяти, за исключением области SFR, которая может использоваться только для данных. Распределение адресов памяти приведено в таблице 4.1.

Таблица 4.1 – Распределение адресов памяти

Адресная область	Начальный адрес	Конечный адрес	Объем памяти
Память программ Flash	$00'0000_H$ $01'8000_H$	$00'7FFF_H$ $04'FFFF_H$	32 Кбайт 224 Кбайт
PSRAM	$05'0000_H$	$05'1FFF_H$	8 Кбайт
SFR	$00'FE00_H$	$00'FFFF_H$	512 байт
DPRAM	$00'F200_H$	$00'FDFF_H$	3 Кбайт
ESFR	$00'F000_H$	$00'F1FF_H$	512 байт
XRAM	$00'D800_H$	$00'E7FF_H$	4 Кбайт
XSFR	$00'E800_H$	$00'EFFF_H$	2 Кбайт

4.1 Организация данных в памяти

Байты хранятся в четных и нечетных адресах. Слова сохраняются в восходящем порядке. Младший байт располагается в четном адресе и в следующем нечетном адресе – старший байт. Двойные слова располагаются в восходящем порядке как два последовательных слова. Одиночные биты всегда располагаются на отведенных им позициях для битов в адресах слов (не присоединенных). Память и регистры хранят данные и команды в прямом порядке передачи байтов (самые младшие байты в младших адресах). Последовательность байтов отражена на рисунке 4.2. Нулевая позиция бита имеет наименьшее значение в байте с четным адресом, и позиция 15-го бита имеет наибольшее значение в байте со следующим нечетным адресом. Битовая адресация поддерживается для части регистров SFR, части DPRAM и для регистров основного назначения GPR.

Примечание – Блоки байтов для слов или двойных слов всегда должны храниться в одинаковой физической области памяти (внутренней, внешней) и организованной области памяти (страница, сегмент).

Рисунок 4.2 – Хранение слов, байтов и битов памяти, организованных по байтовому принципу

4.2 Внутренняя область локальной памяти LM

Микроконтроллер 1887ВЕ3Т может резервировать адресное пространство в широких пределах (в зависимости от версии) для локальной памяти LM микросхемы. Внутренняя область LM может быть SRAM, Flash.

Внутренняя область LM может быть разрешена, запрещена или перемещена в нулевой или первый сегмент при программной проверке.

Обращение к внутренней области LM глобально разрешено или запрещено через бит ROMEN в регистре SYSCON. Этот бит устанавливается в течение сброса в соответствии с уровнем сигнала на внешнем выводе EA# или может быть изменен программно. Если в этот бит записана «1», то внутренняя область LM занимает нижние 32 Кбайта в нулевом сегменте или в первом сегменте. Размещение управляется изменением значения бита ROMS1 регистра SYSCON.

Примечание – Размер области внутренней LM не зависит от реальной физической LM. Микроконтроллеры с объемом LM менее чем 32 Кбайта или вообще без внутренней LM будут иметь все ту же 32-Кбайтную область, отведенную под LM, в случае разрешения работы с внутренней LM. Микроконтроллеры с большим объемом памяти размещают в этом адресном пространстве только ту часть памяти, которая попадает в эти пределы.

Микроконтроллеры с LM свыше 32 Кбайт оставшуюся часть памяти размещают в середине первого сегмента, начиная с адреса $01`8000_H$.

Внутренняя область LM может использоваться как для данных, так и для программного кода.

Привязка кода всегда осуществляется к четному адресу байта. Наибольший возможный адрес кода во внутренней области LM является либо $XX`XXFE_H$ для команд из одного слова, либо $XX`XXFC_H$ для двухсловных команд. По этим адресам необходимо размещать команды безусловного перехода, потому что последовательный переход через границу между внутренней LM и внешней памятью не поддерживается и может привести к ошибочным результатам.

Любой доступ для чтения слов или байтов может осуществляться с помощью косвенного или прямого 16-разрядного режима адресации. Для операндов во внутренней LM нет режима короткой адресации. Любой доступ к двойным словам производится через четные байты адреса. Наибольший возможный адрес данных для слова в LM – $XXXX`XXFE_H$, для максимального двойного слова – $XXXX`XXFC_H$.

Внутренняя область LM не обеспечивает хранение однобитных данных и поэтому не является побитно адресуемой.

Примечание – X в размещениях выше зависит от доступной внутренней области LM.

4.3 Области памяти DPRAM, SFR, ESFR, XSFR, XRAM и PSRAM

В микросхеме 1887ВЕ3Т память разграничена на внутреннюю память данных DPRAM и внутреннюю память области периферии. Области DPRAM и SFR расположены на 3-й странице данных и обеспечивают быстрый доступ с помощью указателя страницы данных DPP, см. рисунок 4.3.

Примечание – Доступы к коду не возможны из области SFR.

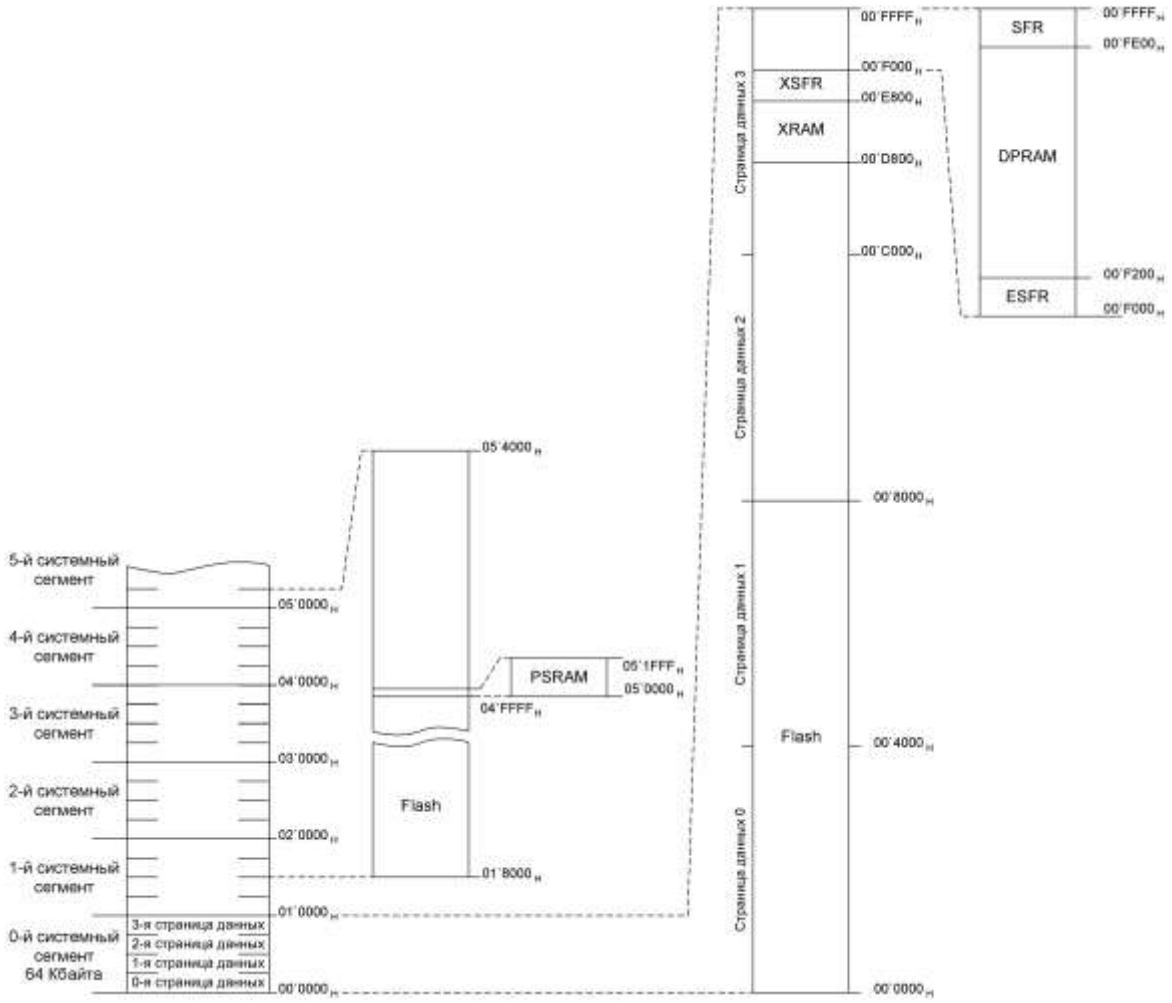


Рисунок 4.3 – Карта памяти микроконтроллера 1887ВЕ3Т

Память данных DPRAM

DPRAM является оперативным запоминающим устройством, которое доступно, главным образом, для хранения данных и служит:

- для банков данных регистров общего назначения GPR;
- для хранения переменных и других данных;
- для системных и пользовательских стеков;
- для указателей адресов источников и назначения для контроллера периферийных событий.

Под DPRAM резервируется 3 Кбайт области памяти ($00'F200_H$ – $00'FDFF_H$). Старшие 256 байтов DPRAM ($00'FD00_H$ – $00'FDFF_H$) и текущий банк регистров области GPR предусматривают хранение единичных битов и являются побитно адресуемыми, см. рисунок 4.3. Любые слова и данные в DPRAM могут быть доступны с помощью косвенного или длинного 16-битного режима адресации, если выбранный регистр DPP указывает на страницу данных 3. При обращении к коду всегда производится доступ к четным адресам байтов. Наибольший адрес, по которому можно обратиться к DPRAM – $00`FDFE_H$.

Для команд, состоящих из одного слова, наибольший возможный адрес обращения в DPRAM – $00'FDFE_H$, а для двухсловных команд – $00'FDFC_H$ (это побитно адресуемый участок, который не должен использоваться для кода). Соответствующее размещение должно содержать команду перехода (безусловный переход), потому что прямой переход из DPRAM в область регистров SFR не поддерживается и может привести к ошибочным результатам.

Области SFR и ESFR

Функции ЦПУ, шинный интерфейс, порты ввода-вывода и расположенной на кристалле ИС 1887ВЕ3Т периферии управляются регистрами специального назначения SFR. Регистры специального назначения расположены в двух 512-байтных областях. Первый блок регистра области SFR занимает 512 байт выше DPRAM ($00'FFFF_H$ – $00'FE00_H$). Второй блок регистра области ESFR занимает 512 байт ниже DPRAM ($00'F1FF_H$ – $00'F000_H$).

Адресация регистров специального назначения осуществляется с помощью косвенного или длинного 16-битного режимов адресации. Слово SFR и соответствующие ему младшие байты могут быть адресованы с использованием 8-битного сдвига вместе с неявным базовым адресом. Однако это не работает для соответствующих старших байтов.

Примечания

1 Обращение к старшему байту SFR с использованием режима 8-битного сдвига невозможно. Запись любого байта SFR является причиной для стирания неадресованного дополнительного байта.

2 К регистрам общего назначения можно обратиться с использованием режима 8-битного сдвига, но в GPR не отображены в участках памяти SFR и ESFR. Использование взамен соответствующего длинного адреса осуществляет доступ к GPR с использованием внутренней периферийной шины.

Верхняя половина каждого блока регистра является побитно адресуемой, таким образом соответствующие биты статуса могут быть изменены непосредственно или при проверке использования побитовой адресации.

Когда регистры выбраны в области ESFR, используется 8-битная адресация или прямая битовая адресация, команда расширения регистра EXTR требуется для доступа к регистрам в области ESFR, прежде чем переключать режим короткой адресации из области стандартных SFR в область ESFR. Это не требуется в случае 16-битной и косвенной адресации. Регистры общего назначения R15, …, R0 дублируются, то есть они доступны в пределах обоих блоков регистра через краткую 2-, 4- или 8-битную адресацию без переключения.

Пример 1:

```
EXTR #4                     ; Переключение на ESFR для следующих четырех
                            ; команд
MOV ODP2, #data16           ; ODP2 (область ESFR) использует 8-битную
                            ; адресацию reg
BFLDL DP6, #mask, #data8    ; DP6 (область ESFR) побитовая адресация для битовых
                            ; полей
BSET DP6.7                  ; DP6 (область ESFR) побитовая адресация для
                            ; отдельных битов
MOV T8REL, R1               ; T8REL использует 16-битный адрес, R1 дублируется
                            ; … и также доступен посредством режима
                            ; ESFR (EXTR не требуются для этого доступа)
;---- ;---------------      ; Окончание действий команды EXTR #4
MOV T8REL, R1               ; T8REL использует 16-битный адрес, R1 дублируется
                            ; … и переключения не требуется
```

Регистры хранения области ESFR в основном требуются для инициализации и выбора режима при минимизации переключения банков SFR. Регистры, которые требуются для частого доступа, размещаются в стандартной области SFR, насколько это возможно.

Примечание – Средства разработки, снабженные мониторным доступом к области регистров ESFR, будут автоматически вставлять команды EXTR, переключая адреса банка SFR или выдавать предупреждение в случае пропущенных или лишних команд EXTR.

Область регистров SFR 2К × 8 XSFR

Область регистров XSFR предназначена для управления устройствами, подключенными к шине XBUS.

ОЗУ данных 4К × 8 XRAM

ОЗУ данных XRAM предназначено для хранения данных. Оно может быть использовано для организации системного стека. XRAM связано с процессором 16-разрядной шиной данных.

ОЗУ программ/данных 8К × 8 PSRAM

ОЗУ программ/данных PSRAM предназначено для хранения программы и может быть использовано для хранения данных.

Память программ Flash 256К × 8 (PM)

PM предназначена для хранения команд программы и постоянных данных. В качестве PM в микроконтроллере используется энергонезависимое запоминающее устройство типа Flash Memory, допускающее многократную перезапись кодов. Контроллер содержит 256 Кбайт встроенной Flash-памяти, начинающейся с адреса $00`0000_H$ для программного кода и данных. Память не требует дополнительного источника напряжения для программирования. Для стабилизации встроенных генераторов напряжения требуется около 100 мкс. 32-х битный доступ к программному коду позволяет извлечь сразу всю четырехбайтную команду или две двухбайтные, что обеспечивает максимальную производительность ЦПУ.

Flash-массив можно условно разделить на 2048 секторов по 128 байт в секторе или на 32 блока по 8 Кбайт. Одной командной последовательностью можно стереть всю Flash-память, блок или сектор, а также записать одно слово. Если память находится в состоянии программирования или стирания, устанавливается бит BUSY регистра состояния Flash-памяти FSR ($08`F000_H$). В этом случае попытка чтения из какой-либо ячейки памяти не удастся. При ошибочной командной последовательности устанавливается бит SQER, который сбрасывается после чтения FSR.

Начальная запись в чистую Flash-память возможна как с помощью программы во внешней памяти, так и с помощью JTAG порта и команд OCDS.

Указатели адресов назначения и источников PEC

16 (24/32) слов, использующие 8 (12/16) каналов PEC в DPRAM от $00'FCE0_H$ ($00'FCD0_H$/$00'FCC0_H$) до $00'FCFE_H$ (только ниже секции с побитовой адресацией), представлены как указатели адресов назначения и источников для переноса данных по каналам PEC. Каждый канал использует пару адресов, которые хранятся в двух последовательных словах указателя источника SRCPx для младшего и указателя адреса назначения DSTPx для старшего адреса.

Всякий раз, как только совершается передача данных, пара указателей SRCPx и DSTPx, выбранных по номеру канала PEC, получают, независимо от текущего значения DPP-регистра, доступ к этим адресам. Если канал PEC не используется, то область, отводящаяся под указатели точек, доступна и может быть использована для хранения байтов или слов данных.

Более детальное описание об использовании SRCPx и DSTPx для передачи данных по каналам PEC приведено в разделе 10 «Система прерываний и ловушек».

4.4 Пространство внешней памяти

Центральное процессорное устройство ИС 1887ВЕ3Т может использовать 16 Мбайт адресного пространства. Всего лишь часть этого адресного пространства занимают внутренняя LM, DPRAM и область ввода-вывода регистров специального назначения. Все адреса, не используемые для этих видов памяти микросхемы или для регистров, могут использоваться для внешней памяти. Обращение к пространству внешней памяти осуществляется через контроллер внешней шины EBC.

С помощью контроллера внешней шины EBC осуществляется связь между ЦПУ микросхемы 1887ВЕ3Т, внешней XBUS и интерфейсом. Внешний шинный интерфейс осуществляет доступ к внешним периферическим устройствам и дополнительной энергозависимой или энергонезависимой памяти. Внешний шинный интерфейс может в дальнейшем ограничивать количество адресуемой внешней памяти. XBUS и интерфейс учитывает внутреннюю системную шину, допускающую интеграцию определяемых заказчиком периферических устройств в микросхеме, энергозависимой или энергонезависимой памяти. Возможность использования внешнего или внутреннего шинного интерфейса зависит от функциональных возможностей интегрированного контроллера внешней шины EBC.

Доступ к внешним данным

К внешнему слову и данным можно обратиться только через косвенный или длинный 16-битный режим адресации, используя один из четырех DPP регистров. Нет никакого укороченного способа для внешних операндов. Любой доступ к данным осуществляется по четному адресу (даже адрес слова).

Примечание – Внешняя память – не для хранения единичных битов и поэтому не адресуется побитно.

4.5 Пересечение границ памяти

Адресное пространство XC167 неявно разделено на блоки одинакового размера, но различной степени детализации, и логические участки памяти. Пересечение границ между блоками (коды или данные) или участков требуют особого внимания, чтобы гарантировать, что контроллер выполняет желательные операции.

Области памяти разделяют адресное пространство, определяя различные виды памяти (если все обеспечено). Эти участки памяти – DPRAM, внутренняя область LM и внешняя память.

Доступ к размещению последующих данных, которые принадлежат к различным участкам памяти, не полностью поддерживается, и может привести к ошибочным результатам. Не является проблемой, если границы памяти выравниваются пословно. Однако, выполняя код, различные участки памяти (области внутренней памяти и внешняя память) должны быть явно коммутированы через команды перехода. Последовательное пограничное пересечение не поддержано и приводит к ошибочным результатам.

Сегментами являются смежные блоки по 64 Кбайт каждый. На них ссылаются через указатель сегмента кода CSP для выборок кода и через явное число сегмента для выборки данных, отменяющих стандартную схему DPP.

В течение выборки кода, сегменты не изменяются автоматически, а точнее должны быть явно коммутированы. Команды JMPS, CALLS и RETS сделают это.

В больших логических программах самое высокое размещение кода в сегменте содержит машинную команду безусловного перехода к соответствующему следующему сегменту, препятствующую тому, чтобы устройство предвыборки попыталось оставить текущий сегмент.

Страницами данных являются смежные блоки по 16 Кбайт каждый. На них ссылаются через указатели страниц данных DPP3, …, DPP0 и через явный номер страницы данных для отмены выборки данных схемой стандарта DPP. Каждый регистр DPP может выбирать одну из 1024 возможных страниц данных. Регистр DPP, который используется для текущего обращения, выбирается через два старших бита 16-разрядного адреса данных. Поэтому последующие 16-разрядные адреса данных, которые пересекают 16-килобайтные поверхности страницы данных, будут использовать указатели разных страниц данных, в то время как физические размещения не должны быть более поздними в пределах памяти.

4.6 Системный стек

Системный стек может быть определен в пределах DPRAM. Размер системного стека контролируется набором битов STKSZ регистра SYSCON, см. таблицу 4.2.

Для всех операций с системным стеком DPRAM доступна через регистр указателя стека SP. Заполнение стека производится по направлению от максимального адреса к наименьшему адресу.

Чтение и запись данных производится только словами, байтовое обращение не поддерживается. После добавления данных на стек, указатель стека SP уменьшается на два, а стек при этом растет в сторону младших адресов.

Таблица 4.2 – Размеры системного стека

<STKSZ>	DPRAM в Кбайт	Размер стека в словах	Адрес DPRAM (слово)
000_B	1, 2, 3	256	$00'FBFE_H$–$00'FA00_H$ (после сброса)
001_B	1, 2, 3	128	$00'FBFE_H$–$00'FB00_H$
010_B	1, 2, 3	64	$00'FBFE_H$–$00'FB80_H$
011_B	1, 2, 3	32	$00'FBFE_H$–$00FBC0_H$
100_B	1	–	Не использовать эту комбинацию!
	2, 3	512	$00'FBFE_H$–$00'F800_H$
101_B	–	–	Зарезервировано, не использовать!
110_B	–	–	Зарезервировано, не использовать!
111_B нециркулярный стек	1	512	$00'FDFE_H$–$00'FA00_H$
	2	1024	$00'FDFE_H$–$00'F600_H$
	3	1536	$00'FDFE_H$–$00'F200_H$

Контроль за нижним и верхним выбранным пределом стека осуществляется с помощью регистров переполнения стека STKOV и опустошения стека STKUN. Эти два контрольных регистра стека могут использоваться не только для защиты от уничтожения данных, но также позволяют использовать циркулирующий стек с аппаратной поддержкой заполнения и очистки стека, за исключением опции STKSZ = 111_B.

4.7 Регистры общего назначения

Регистры общего назначения GPR могут быть объединены в блок из 16 последовательных слов в DPRAM. Содержимое регистра контекстного указателя CP определяет базовый адрес банка регистров, активного в настоящий момент. Банк регистров может содержать до 16 GPR (слова – R0, ..., R15) и/или до 16 GPR (байты – RL0, RH0, ..., RL7, RH7). 16 байт GPR расположены в первых восьми словах GPR.

В противоположность системному стеку нумерация внутри банка регистров возрастает от меньших к большим адресам и занимает в максимальном случае 32 байта. При использовании регистра CP в качестве базового адреса (независимого от содержимого текущего регистра DPP), GPR доступны с помощью 2-, 4- или 8-битного режима адресации. Необходимо отметить, что каждый бит в текущем активном банке доступен индивидуально.

Микроконтроллер поддерживает быстрое переключение между банками регистров. Большое количество банков регистров может одновременно физически существовать в DPRAM. При этом является активным только выбранный в CP банк регистров. Выбор другого активного банка регистров возможен путем простого изменения значения в CP.

Карта адресов регистров области GPR представлена в таблице 4.3.

Таблица 4.3 – Карта адресов регистров области GPR

DPRAM адрес	Байтовые регистры		Регистры слов
<CP> + $1E_H$	–		R15
<CP> + $1C_H$	–		R14
<CP> + $1A_H$	–		R13
<CP> + 18_H	–		R12
<CP> + 16_H	–		R11
<CP> + 14_H	–		R10
<CP> + 12_H	–		R9
<CP> + 10_H	–		R8
<CP> + $0E_H$	RH7	RL7	R7
<CP> + $0C_H$	RH6	RL6	R6
<CP> + $0A_H$	RH5	RL5	R5
<CP> + 08_H	RH4	RL4	R4
<CP> + 06_H	RH3	RL3	R3
<CP> + 04_H	RH2	RL2	R2
<CP> + 02_H	RH1	RL1	R1
<CP> + 00_H	RH0	RL0	R0

Специальная команда переключения контекста SCXT обеспечивает переключение банков регистров и автоматическое сохранение предыдущего содержимого CP.

Количество используемых банков регистров ограничено только размером доступного объема RAM.

Таблицы регистров областей SFR и ESFR, упорядоченные по физическим адресам, представлены в приложении Б.

Таблицы регистров областей SFR и ESFR, упорядоченные по именам регистров, представлены в приложении В.

Таблицы регистров областей SFR и ESFR, упорядоченные в зависимости от отношения к тому или иному блоку микроконтроллера, представлены в приложении Г.

5 Встроенная Flash-память

Контроллер содержит 256 Кбайт встроенной Flash-памяти, начинающейся с адреса $00`0000_H$ для программного кода и данных. Память не требует дополнительного источника напряжения для программирования. Для стабилизации встроенных генераторов напряжения требуется около 100 мкс. 32-битный доступ к программному коду позволяет извлечь сразу всю четырехбайтную команду или две двухбайтные, что обеспечивает максимальную производительность ЦПУ.

Операции стирания и программирования запускаются посредством командных последовательностей, что исключает непредвиденное изменение содержимого Flash-памяти, см. таблицу 5.1.

Flash-массив можно условно разделить на 2048 секторов по 128 байт в секторе или на 32 блока по 8 Кбайт. Одной командной последовательностью можно стереть всю Flash, блок или сектор, а также записать одно слово. Если память находится в состоянии программирования или стирания, устанавливается бит BUSY регистра состояния Flash -памяти FSR ($08`F000_H$). В этом случае попытка чтения из какой-либо ячейки памяти не удастся. При ошибочной командной последовательности устанавливается бит SQER, который сбрасывается после чтения FSR.

Физическое пространство адресов Flash-памяти от $00`0000_H$ до $03`FFFF_H$. Оно отражается на адреса от $00`0000_H$ до $00`7FFF_H$ и от $01`8000_H$ до $04`FFFF_H$ системной памяти.

Начальная запись в чистую Flash-память возможна как с помощью программы во внешней памяти, так и с помощью JTAG порта и команд OCDS.

Примечание – Для уверенного выполнения инструкций, следующих за командной последовательностью, после командной последовательности необходимо использовать не менее четырех команд NOP.

Командные последовательности

Для защиты от непредвиденного изменения данных (например, во время включения/ выключения питания) команды записи/стирания подаются посредством командных последовательностей, см. таблицу 5.1. Командная последовательность состоит из нескольких записей определенных данных по определенным виртуальным адресам.

Таблица 5.1 – Командные последовательности для управления Flash-памятью

Команда	Первый цикл записи		Второй цикл записи		Третий цикл записи	
	Адрес	Данные	Адрес	Данные	Адрес	Данные
Записать слово	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	$08`0555_H$	$A0_H$
Стереть сектор	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	$08`0555_H$	80_H
Стереть блок	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	$08`0555_H$	80_H
Стереть все	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	$08`0555_H$	80_H
Отмена команды	$08`0AAA_H$	$F0_H$	–	–	–	–
Команда	Четвертый цикл записи		Пятый цикл записи		Шестой цикл записи	
	Адрес	Данные	Адрес	Данные	Адрес	Данные
Записать слово	<адрес>	<слово>	–	–	–	–
Стереть сектор	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	<адрес>	30_H
Стереть блок	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	<адрес>	50_H
Стереть все	$08`0555_H$	AA_H	$08`0AAA_H$	55_H	$08`0555_H$	10_H
Отмена команды	–	–	–	–	–	–

Командная последовательность «Записать слово», см. рисунок 5.1, записывает два байта в ячейку памяти. Адрес ячейки и записываемое значение указываются в четвертом цикле командной последовательности.

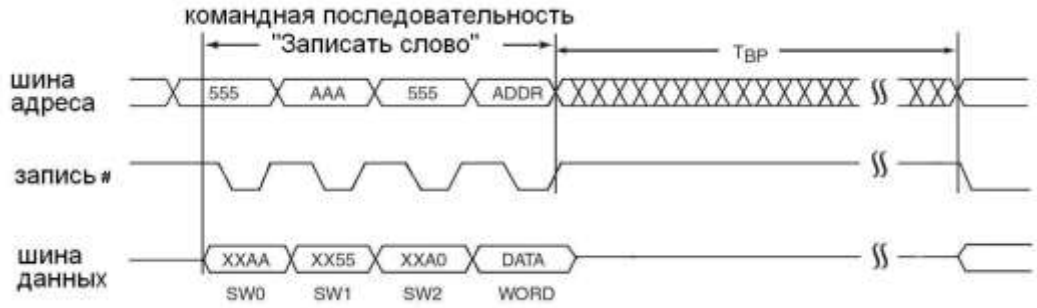


Рисунок 5.1 – Командная последовательность «Записать слово»

Командная последовательность «Стереть сектор», см. рисунок 5.2, стирает один 128-байтный сектор, адрес которого указан в шестом цикле командной последовательности.

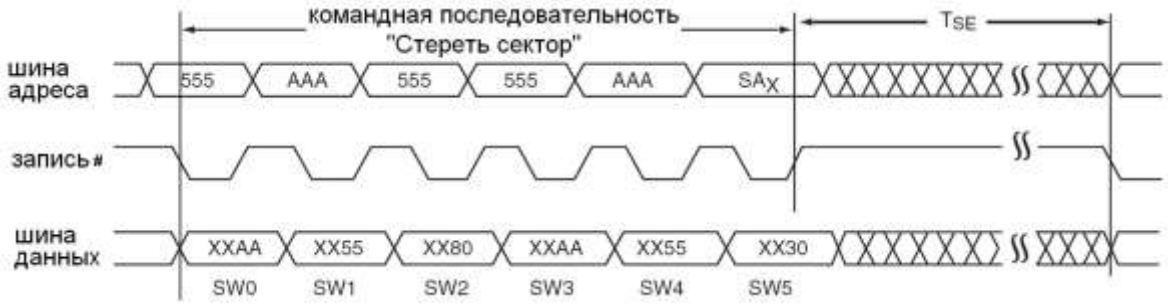


Рисунок 5.2 – Командная последовательность «Стереть сектор»

Командная последовательность «Стереть блок», см. рисунок 5.3, стирает один 8-килобайтный блок, адрес которого указан в шестом цикле командной последовательности.

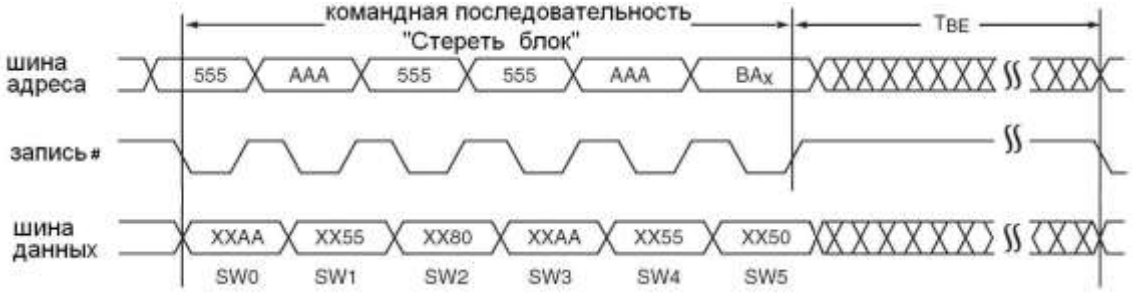


Рисунок 5.3 – Командная последовательность «Стереть блок»

Командная последовательность «Стереть все», см. рисунок 5.4, полностью очищает всю Flash-память.

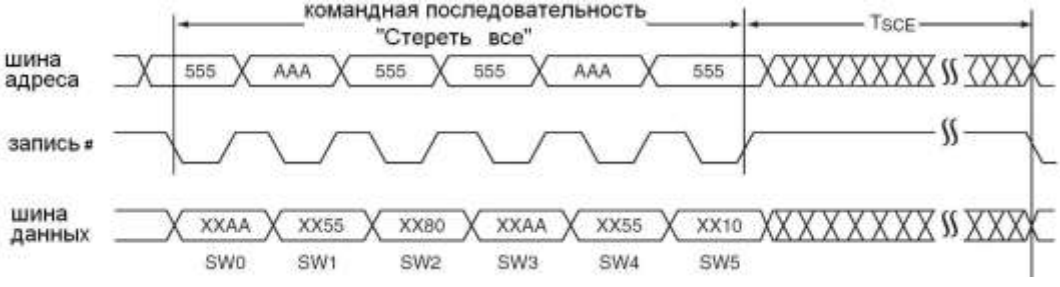


Рисунок 5.4 – Командная последовательность «Стереть все».

Примечание – Из незаписанных ячеек памяти считывается $FFFF_H$.

Командная последовательность «Отмена команды», см. рисунок 5.5, сбрасывает незавершенную командную последовательность. Память переходит в режим чтения.

Рисунок 5.5 – Командная последовательность «Отмена команды»

Если при введении командной последовательности произошла ошибка в адресе или данных, устанавливается бит SQER в регистре FSR, и память также переходит в режим чтения.

Длительность циклов программирования и стирания представлена в таблице 5.2.

Таблица 5.2 – Длительность циклов программирования и стирания

Параметр	Условное обозначение	Значение	Рисунок
Длительность записи слова	T_{BP}	63,00 мкс	5.1
Длительность стирания сектора	T_{SE}	1,33 мс	5.2
Длительность стирания блока	T_{BE}	1,33 мс	5.3
Длительность стирания всей памяти	T_{SCE}	21,3 мс	5.4

Регистр состояния Flash-памяти FSR

Формат регистра состояния Flash-памяти FSR представлен на рисунке 5.6, функциональное назначение полей регистра FSR – в таблице 5.3.

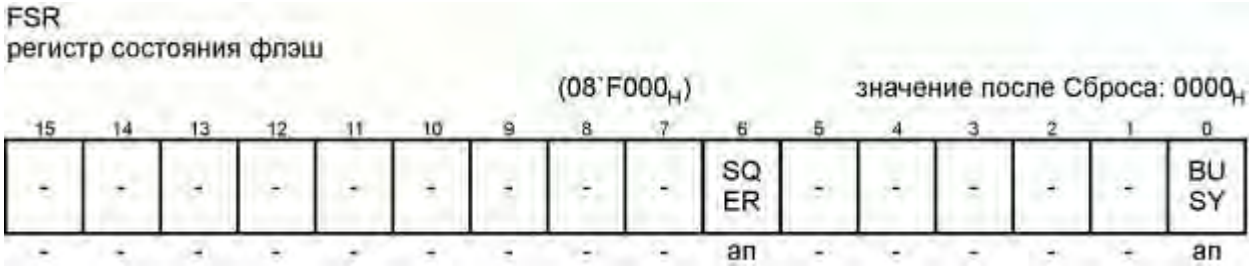


Рисунок 5.6 – Формат регистра FSR

.

Таблица 5.3 – Функциональное назначение полей регистра FSR

Поле	Биты	Тип	Описание
BUSY	0	Аппаратное влияние	Флаг операции. Устанавливается во время операций стирания/программирования. Сбрасывается по окончании операций
SQER	6	Аппаратное влияние	Флаг ошибки. Устанавливается при обнаружении ошибки в командной последовательности. Сбрасывается при чтении содержимого регистра
–	15-7, 5-1	–	Зарезервировано. Не использовать
Примечание – Биты SQER и BUSY устанавливаются и сбрасываются только аппаратно.			

6 Блок умножения-накопления MAC

Блок умножения-накопления MAC является специализированным сопроцессором, добавленным к ядру ЦПУ для улучшения выполнения алгоритмов обработки сигналов. Блок MAC включает в себя:

- блок умножения-накопления;
- блок генерации адреса, способствующий подаче в блок MAC двух операндов за цикл;
- блок повтора для осуществления ряда команд умножения-накопления.

Архитектура блока MAC показана на рисунке 6.1.

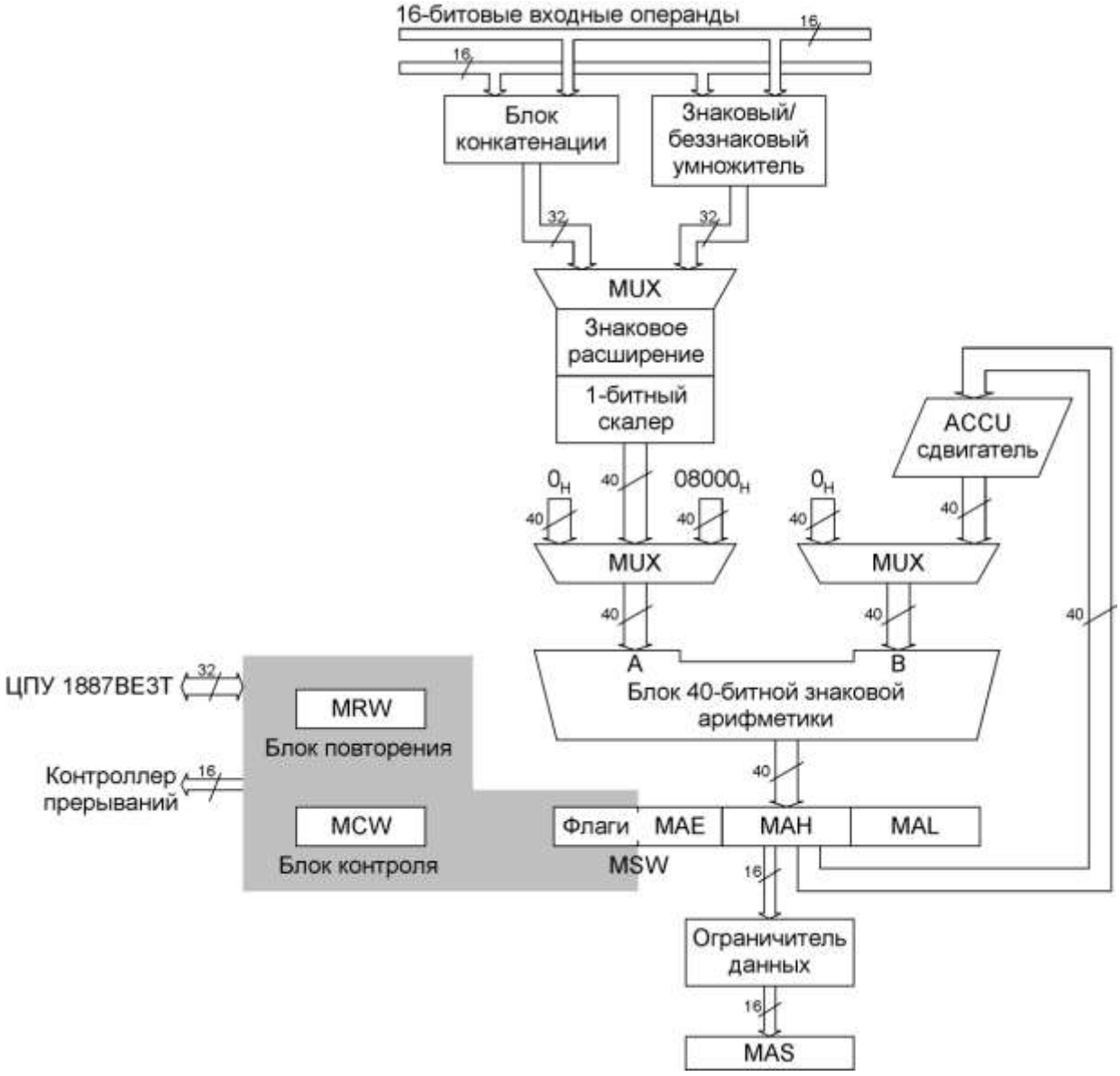


Рисунок 6.1 – Архитектура блока MAC

Блок MAC имеет некоторые особенности.

Расширенные возможности адресации

- Новые режимы адресации, включая режим двойной косвенной адресации с последующим изменением указателя адреса.
- Параллельная пересылка данных, позволяющая пересылать данные одного операнда параллельно с выполнением команд умножения-накопления.

- Новые команды пересылки данных CoSTORE (для быстрого доступа к SFR регистрам блока MAC) и CoMOV (для быстрой пересылки из одной области памяти в другую).

Блок умножения-накопления

- Выполнение всех операций блока MAC за один командный цикл ЦПУ.
- Параллельный умножитель 16x16 со знаком/ без знака.
- Блок 40-битной знаковой арифметики с режимом автоматического ограничения.
- 40-битный аккумулятор со знаком.
- 8-битный сдвигатель вправо/влево.
- Скалер (1-битный сдвигатель влево).
- Ограничитель данных.
- Весь набор команд, включая команды: умножения, умножения-накопления, 32-битные арифметические команды и команды сравнения.
- Статусный регистр и регистры управления:
    - MSW – статусный регистр;
    - MCW – регистр управления;
    - MRW – регистр повторения.

Программный контроль

- Блок повтора, позволяющий некоторым командам блока MAC повторяться до 8192 раз. Повторяемые команды могут быть прерваны.
- Прерывание блока MAC (относится к аппаратным ловушкам класса B) при установке флагов блока MAC.

6.1 Операции блока MAC

Конвейеризация команд

Обработка команд блока MAC производится на 4-ступенчатом конвейере. Каждая ступень имеет свой набор выполняемых операций.

Конвейер микроконтроллера состоит из следующих этапов: выборка, декодирование, выполнение, запись результата. Более подробное описание конвейера находится в описании блока ЦПУ.

Генерация адреса

При выполнении команд блока MAC используются некоторые стандартные режимы адресации: с помощью прямого адреса GPR или непосредственной константы #data4. Новые режимы добавились для обеспечения блока MAC двумя новыми операндами за командный цикл. Это делает возможным косвенную адресацию с последующим изменением указателя адреса.

Для двойной косвенной адресации требуется два указателя адреса. Любой GPR может быть использован в качестве одного указателя адреса, в качестве другого указателя могут использоваться один или два специальных регистра SFR: IDX0 или IDX1. Две пары регистров смещения указателя адреса QR0/QR1 или QX0/QX1 связаны с каждым указателем адреса (GPR или IDXi). Указатели адреса GPR разрешают доступ ко всей области памяти, а доступ с помощью IDXi ограничен внутренней памятью DPRAM, за исключением команды CoMOV. Возможные варианты комбинаций указателей адреса с последующим изменением указателя для каждого из двух новых режимов адресации показаны в таблице 6.1. Символы [$Rw_n\otimes$] и [$IDX_i\otimes$] относятся к данным режимам адресации.

Таблица 6.1 – Указатели адреса с последующим изменением. Комбинации для [IDX_i⊗] и [Rw_n⊗]

Символ	Мнемоника	Операция указателя адреса
[IDX_i⊗]	[IDX_i]	(IDX_i) ← (IDX_i) (нет операции)
	[IDX_i+]	(IDX_i) ← (IDX_i) + 2 (i = 0, 1)
	[IDX_i–]	(IDX_i) ← (IDX_i) – 2 (i = 0, 1)
	[IDX_i + QX_j]	(IDX_i) ← (IDX_i) + (QX_j) (i, j = 0, 1)
	[IDX_i – QX_j]	(IDX_i) ← (IDX_i) – (QX_j) (i, j = 0, 1)
[Rw_n⊗]	[Rw_n]	(Rw_n) ← (Rw_n) (нет операции)
	[Rw_n+]	(Rw_n) ← (Rw_n) + 2 (n = 0 – 15)
	[Rw_n–]	(Rw_n) ← (Rw_n) – 2 (n = 0 – 15)
	[Rw_n + QR_j]	(Rw_n) ← (Rw_n) + (QR_j) (n = 0 – 15, j = 0, 1)
	[Rw_n – QR_j]	(Rw_n) ← (Rw_n) – (QR_j) (n = 0 – 15, j = 0, 1)

Параллельная пересылка данных

Команды класса CoMACM являются определенным набором команд, которые используют алгоритм, называемый параллельной пересылкой данных. Только команды CoMACM используют двойную косвенную адресацию. Параллельная пересылка данных позволяет параллельно с выполнением операции блока MAC переслать данные, адрес которых определяется IDXi, в другую область памяти. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi. Адресация в режиме параллельной пересылки данных показана в таблице 6.2.

Таблица 6.2 – Адресация при параллельной пересылке данных

Команда	Адрес для параллельной пересылки данных
CoMACM [IDX_i+], [Rw_n⊗]	<IDX_i – 2>
CoMACM [IDX_i–], [Rw_n⊗]	<IDX_i + 2>
CoMACM [IDX_i + QX_j], [Rw_n⊗]	<IDX_i – QX_j>
CoMACM [IDX_i – QX_j], [Rw_n⊗]	<IDX_i + QX_j>

Параллельная пересылка данных сдвигает таблицу операндов параллельно с проведением вычислений с этими операндами. Пример параллельной пересылки данных при использовании команды CoMACM показан на рисунке 6.2.

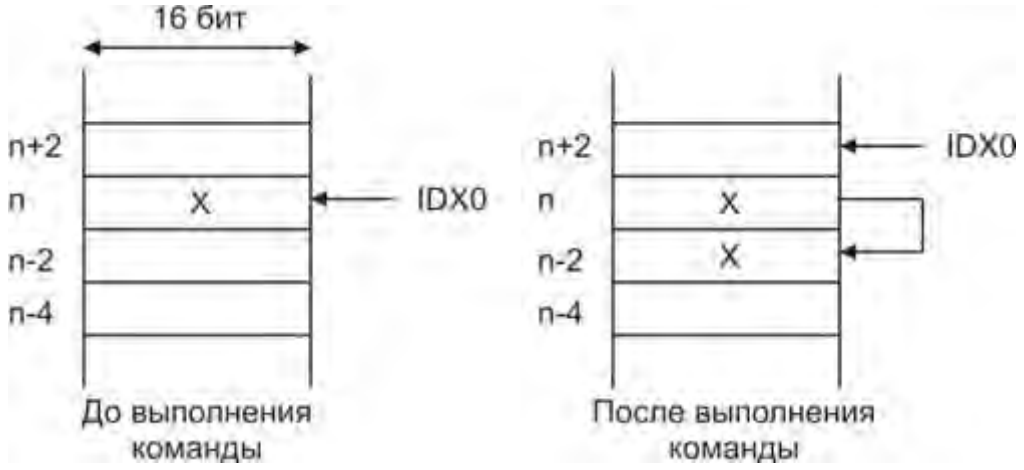


Рисунок 6.2 – Пример параллельной пересылки данных

Режим адресации CoReg

Аккумулятор и регистры управления MAL, MAH, MSW, MCW, MRW блока MAC могут быть адресованы стандартным набором команд, как любой другой SFR. Дополнительно они могут быть адресованы командой CoSTORE. Команда CoSTORE использует специальный 5-битный режим адресации, называемый CoReg, который делает возможной непосредственную запись в регистры блока MAC после операции. Адреса регистров блока MAC в режиме адресации CoReg показаны в таблице 6.3.

Таблица 6.3 – 5-битный режим адресации CoReg

Регистр	Описание	5-битный адрес
MSW	Статусный регистр блока MAC	00000_B
MAH	Старший байт аккумулятора блока MAC	00001_B
MAS	«Ограниченное» значение старшего байта аккумулятора блока MAC	00010_B
MAL	Младший байт аккумулятора блока MAC	00100_B
MCW	Регистр управления блока MAC	00101_B
MRW	Регистр повторения блока MAC	00110_B

Регистр MAS является виртуальным. Если MAS определяется как операнд источника для команды CoSTORE, то чтение регистра MAH проходит через ограничитель данных. Регистр MAS не может быть адресован через стандартный адрес SFR/ESFR.

Представление чисел и округление

Блок MAC поддерживает представление двоичных чисел в дополнительном коде. В этом формате знак числа определяется значащим битом MSB байта двоичного числа. Он равен нулю для положительных чисел и единице для отрицательных. Числа без знака используются только в командах умножения/умножения-накопления; в них устанавливается: у какого операнда есть знак, а у какого – нет.

Блок MAC осуществляет «округление в дополнительном коде», когда единица добавляется к биту справа от точки округления (бит 15 регистра MAL) перед отбрасыванием младшего байта числа (обнулением регистра MAL).

6.2 Компоненты блока MAC

Основные компоненты блока MAC показаны на рисунке 6.1, в данном подразделе они описаны детально.

Параллельный умножитель 16 × 16 со знаком/без знака

Устройство выполняет параллельное умножение двух 16-битных дробных и целых чисел. Умножитель имеет два 16-битных входных порта для двух операндов и 32-битный выходной порт для результата умножения. Произведение всегда представляется в формате целого или дробного числа со знаком.

Блок конкатенации

Блок конкатенации позволяет MAC выполнять 32-битные операции за один командный цикл ЦПУ. В нем происходит конкатенация двух 16-битных операндов в один 32-битный операнд перед тем, как будет выполнена 32-битная операция в 40-битном арифметическом блоке. Второй необходимый операнд всегда является текущим значением аккумулятора.

Блок расширения знака и скалер

Перед загрузкой числа в 40-битное знаковое арифметическое устройство, происходит знаковое расширение результата умножения (или результата конкатенации) до 40-битного числа. При знаковом расширении происходит повторение значащего бита MSB восемь раз. При выполнении команд с двумя числами без знака (например, CoMULu, CoMACu) происходит дополнение нулями результата, независимо от значащего бита MSB байта.

Однобитный скалер может сдвигать результат со знаковым расширением на один бит влево. В зависимости от типа команды скалер может управляться как при помощи бита MP (бит 10 в регистре управления блока MAC MCW), так и самой командой. При выполнении команд умножения (если бит MP установлен) результат умножения автоматически сдвигается на 1 бит влево, чтобы компенсировать дополнительный знаковый бит, возникший при умножении двух чисел со знаком в дополнительном коде. Скалер также применяется при выполнении таких команд, как CoADD2, CoSUB2 и т. д., в которых 32-битный операнд удваивается перед загрузкой в арифметический блок.

Блок 40-битной знаковой арифметики

Блок 40-битной знаковой арифметики допускает промежуточные переполнения во время выполнения операций умножения/накопления. Блок содержит два 40-битных входных порта, порт A и порт B. Порт A принимает такие данные, как 0000000000_H, 0000008000_H (округление) или получившийся результат со знаковым расширением, результат блока конкатенации или умножителя после сдвига.

На входной порт B поступают данные через обратную связь с выхода аккумулятора через 8-битный сдвигатель влево/вправо. Входной порт B также может принимать 0000000000_H, чтобы сделать возможной прямую передачу из порта A в аккумулятор.

Если в процессе накопления произошло 40-битное переполнение аккумулятора, то будет установлен флаг SV в статусном регистре MSW блока MAC.

Результат сложения/вычитания может быть округлен или автоматически ограничен до 32-битной величины после каждого накопления.

Округление выполняется путем добавления к результату числа 0000008000_H и обнуления младшего байта аккумулятора MAL. Автоматическое ограничение разрешается установкой бита ограничения MS в регистре управления MCW блока MAC.

Если аккумулятор работает в режиме ограничения и произошло 32-битное переполнение, то в аккумулятор (в зависимости от направления переполнения) записывается наибольшее положительное число или наименьшее отрицательное число, которое может быть представлено в 32-битном формате в дополнительном коде. Таким образом, после ограничения в аккумулятор запишутся значения $007FFFFFFF_H$ (положительное) или $FF80000000_H$ (отрицательное). При автоматическом ограничении выставляется флаг ограничения SL в статусном регистре MSW.

Примечания

1 Если выполняется и автоматическое ограничение, и округление, то после ограничения регистр MAL округляется, в результате после ограничения и округления в аккумулятор запишется число $007FFF0000_H$.

2 Если аккумулятор содержит число, которое не может быть представлено 32-битным числом в дополнительном коде (то есть бит MS был предварительно обнулен), то ограничение может выполниться только после записи единицы в бит MS и осуществления одной команды блока MAC. Если эта команда вызвала 40-битное переполнение (или опустошение), то в аккумулятор после ограничения запишется величина $007FFFFFFF_H$ или $FF80000000_H$.

40-битный регистр аккумулятора со знаком

Большинство команд MAC используют 40-битный регистр аккумулятора в качестве операнда источника и/или операнда приемника. Аккумулятор включает в себя три SFR регистра:

- MAL – младший байт аккумулятора блока MAC;
- MAH – старший байт аккумулятора блока MAC;
- MAE – расширение аккумулятора.

Регистры MAL и MAH являются 16-битными, MAE состоит только из 8 бит, доступ к которым осуществляется как к младшему байту статусного регистра блока MAC (MSW). Регистр MAE является значащим байтом аккумулятора.

При записи в MAH по стандартному SFR адресу величина в аккумуляторе автоматически переводится в формат 40-битного числа в дополнительном коде с расширением знака. В регистр MAL загружается нулевое значение. Если число положительное, то в MAE автоматически загружается нулевое значение (в регистре MAH значащий бит был равен нулю), в случае отрицательного числа в MAE записываются единицы (в регистре MAH значащий бит был равен единице). Следует заметить, что числа в 32-битном формате в дополнительном коде не изменяются и регистр MAE не содержит значащих битов. Так происходит пока старшие 9 бит 40-битного результата со знаком идентичны.

Во время операций накопления может возникнуть переполнение, и результат может не соответствовать 32-битному формату. После этого аккумулятор превышает 32-битную границу и изменяет содержимое регистра MAE. В результате этого в старших 8 битах аккумулятора есть значащие биты (знак отсутствует). Для обозначения этого расширения флаг E, содержащийся в значащем байте статусного регистра MSW, становится равным единице.

Ограничитель данных

Арифметика ограничения также предусматривает избирательное ограничение при переполнении в случае чтения аккумулятора посредством команды CoSTORE<приемник>, MAS. Если содержимое аккумулятора не может быть представлено в 32-битном формате без переполнения, то разрешается работа ограничителя и происходит ограничение значения регистра MAS. В противном случае значение регистра MAS равно значению регистра MAH, как показано в таблице 6.4.

Таблица 6.4 – Выходные значения ограничителя данных

Бит E	Бит N	Выход ограничителя MAS
0	X	Равно значению регистра MAH
1	0	$7FFF_H$
1	1	8000_H

Примечание – Значение регистра MAS можно прочитать только посредством команды CoSTORE<приемник>, MAS. При чтении содержимое аккумулятора и статусного регистра не изменяется.

Сдвигатель аккумулятора

В качестве сдвигателя аккумулятора используется параллельный сдвигатель с 40-битным входом и 40-битным выходом. Операндом источника сдвигателя является аккумулятор. Возможны следующие операции сдвига:

- нет сдвига (значение не изменяется);
- арифметический сдвиг влево до 8 бит;
- арифметический сдвиг вправо до 8 бит.

Следует отметить, что при сдвиге влево изменяются флаги E, SV, SL и C статусного регистра MSW блока MAC. Поэтому, если разрешено автоматическое ограничение (бит MS равен единице), поведение будет схожим с поведением 40-битного арифметического блока.

Примечание – Определенные предосторожности требуются в случаях сдвига влево при разрешенном автоматическом ограничении (бит MS равен единице). Если флаг MS устанавливается прямо перед командой сдвига, то не может быть гарантировано правильное ограничение, исходя из того, что значащий бит может быть сдвинут без сохранения до того, как выполнилось ограничение. Чтобы избежать такой ситуации, необходимо разрешать автоматическое ограничение раньше, чтобы команда сдвига проводилась уже после ограничения.

Блок повторения

Блок MAC содержит блок повтора, который может повторять некоторые команды блока MAC до 2^{13} (8192) раз. Значение в счетчик повтора может устанавливаться как с помощью непосредственной константы (до 31), так и с помощью содержимого счетчика повтора (биты с 12 по 0) регистра повтора MRW блока MAC. Если значение счетчика повтора равно «N», то команда выполнится «N + 1» раз. При каждом повторении команды счетчик повтора сравнивается с нулем. При равенстве счетчика нулю команда перестает выполняться, в противном случае счетчик декрементируется и команда повторяется. Во время выполнения серии повторений устанавливается флаг повтора MR (бит 15 регистра повторения MRW блока MAC), пока не произойдет выполнение последней повторяемой команды.

Синтаксис повторяемой команды показан на следующем примере:

```
Repeat # 24 times
CoMAC [IDX0+], [R0+]                ; повторение 24 раза
```

В данном примере число повторений команды задано непосредственной 5-битной константой. В счетчик повтора в регистре MRW автоматически загружается данная величина минус один.

```
MOV MRW, #00FF_H                    ; запись в регистр MRW
NOP                                 ; команда задержки
Repeat MRW times
CoMACM [IDX1–], [R2+]               ; повторение 256 раз
```

Данная команда повторяется в соответствии со значением счетчика повтора в регистре MRW. Следует отметить, что вследствие конвейерной обработки между записью в регистр MRW и следующей повторяемой командой должна быть вставлена хотя бы одна команда.

Серия повторений может быть прервана. Если прерывание произошло во время серии повторений, повторения прекращаются, вступает в работу программа обработки прерываний. Серия прерываний возобновляется после завершения работы программы обработки прерываний. Во время прерывания флаг повтора MR остается установленным, показывая, что выполнение повторяемой команды было прервано и что счетчик повтора содержит число повторений (минус одно), которые осталось завершить. Если блок прерываний используется в программе обработки прерываний, пользователь должен сохранить значение регистра MRW и восстановить его перед завершением программы обработки прерываний.

Примечание – Необходимо использовать регистр MRW с осторожностью. Кроме случая записи регистра MRW после прерывания, бит MR не должен устанавливаться пользователем. В противном случае не может быть гарантировано корректное выполнение команды.

6.3 Прерывание блока MAC

Блок MAC может генерировать прерывание в соответствии со значениями статусных флагов C (перенос), SV (переполнение), E (расширение), SL (ограничение) регистра MSW.

При установке бита MIE регистра MCW разрешается прерывание блока MAC. При общем разрешении прерывания флаги C, SV, E или SL могут генерировать прерывание, если были установлены соответствующие им маски флагов в регистре MCW: CM, VM, EM или LM. Флаг MIR устанавливается при первом условии для прерывания. Этот флаг может быть обнулен в течение процесса прерывания. Если флаг MIR установлен, то при возникновении следующего условия для прерывания, новое прерывание не будет сгенерировано.

Прерывание блока MAC относится к аппаратным ловушкам класса B (номер ловушки A_H, приоритет ловушки I). Соответствующий прерыванию флаг ловушки MACTRP находится в регистре TFR (бит 6). Следует отметить, что если произошло прерывание блока MAC, то пользователь должен обнулить флаг MACTRP.

Примечание – Соответствующий флаг ловушки должен быть обнулен программой обработки ловушек. В противном случае новый флаг ловушки выставится только после завершения обработки. Флаг ловушки может быть установлен как программно, так и аппаратно.

6.4 Регистры блока MAC

Все регистры блока MAC являются регистрами SFR/ESFR. Доступ к регистрам может быть осуществлен с помощью стандартных команд и команд блока MAC, называемых CoSTORE. Ниже приведен список регистров MAC и соответствующие им адреса.2

Регистры адреса блока MAC

Для режима двойной косвенной адресации требуются дополнительные SFR/ESFR регистры: два указателя адреса IDX0/IDX1 и четыре регистра смещения QX0/QX1 и QR0/QR1.

Регистры указателя адреса IDX0 и IDX1 являются регистрами SFR. Формат регистра IDX0 представлен на рисунке 6.3, формат регистра IDX1 – на рисунке 6.4, функциональное назначение полей регистров IDX0, IDX1 – в таблице 6.5. Форматы регистров QX0, QX1 представлены на рисунках 6.5, 6.6, форматы регистров QR0, QR1– на рисунках 6.7, 6.8. Функциональное назначение полей регистров QX0, QX1, QR0, QR1 представлено в таблице 6.6.

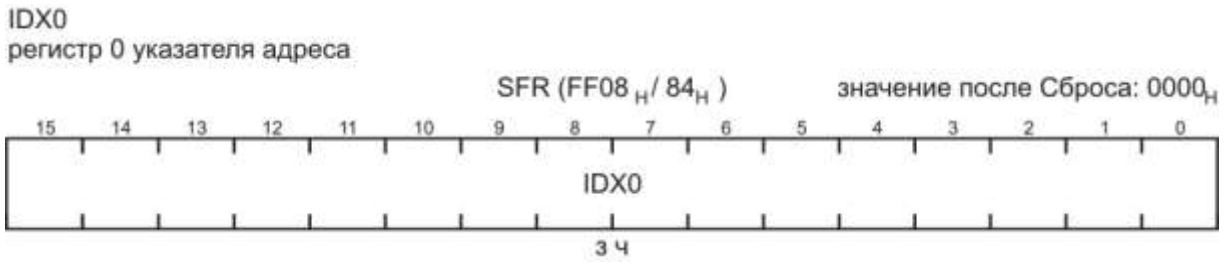


Рисунок 6.3 – Формат регистра IDX0

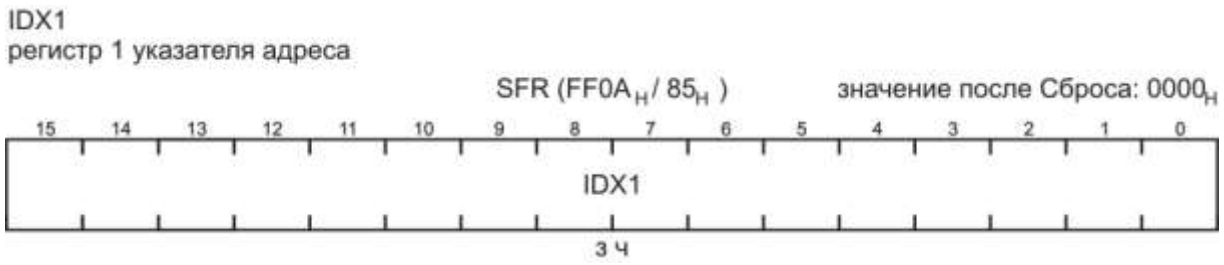


Рисунок 6.4 – Формат регистра IDX1

Таблица 6.5 – Функциональное назначение полей регистров IDX0, IDX1

Поле	Биты	Тип	Описание
IDXi, i = 0, 1	15-1	Запись Чтение	Изменяемая часть указателя адреса
N	0	Чтение	Так как IDXi может содержать только четные числа, то нулевой бит всегда равен нулю

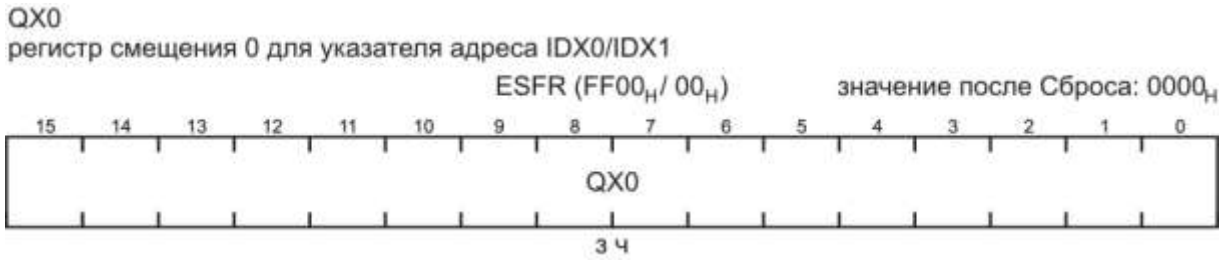


Рисунок 6.5 – Формат регистра QX0

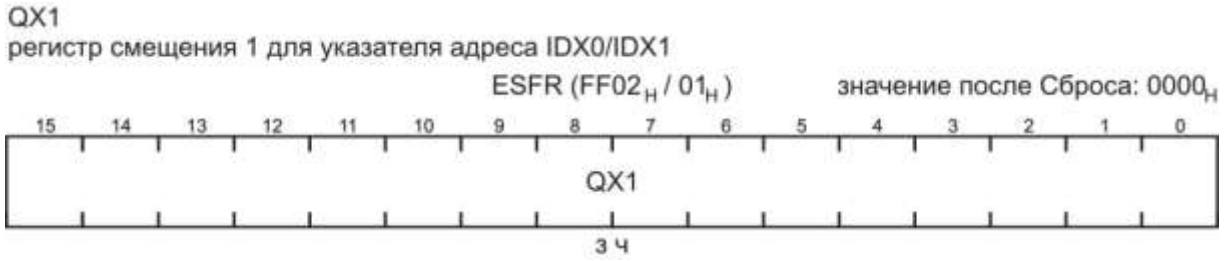


Рисунок 6.6 – Формат регистра QX1

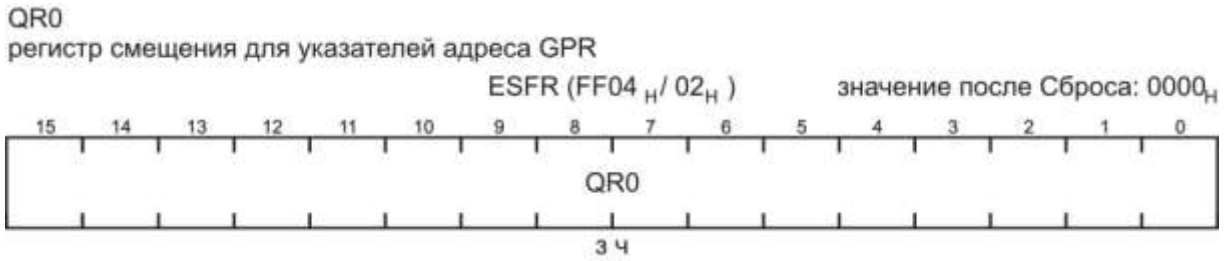


Рисунок 6.7 – Формат регистра QR0

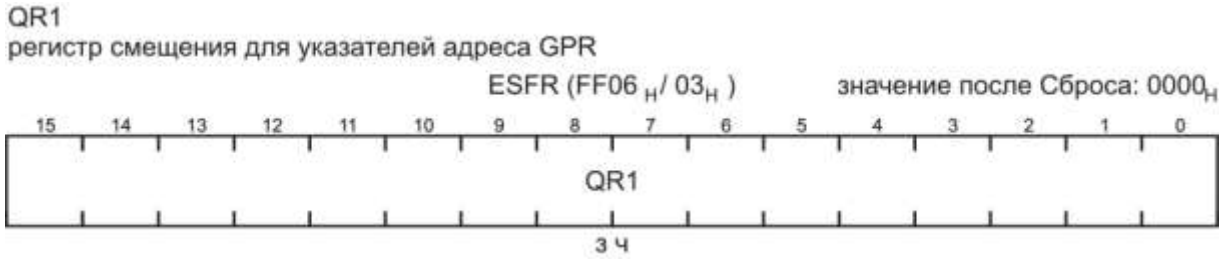


Рисунок 6.8 – Формат регистра QR1

Таблица 6.6 – Функциональное назначение полей регистров QX0, QX1, QR0, QR1

Поле	Биты	Тип	Описание
QRi/QXi, i = 0, 1	15-1	Запись Чтение	Изменяемая часть регистров смещения Определяет 16-битное смещение адреса для указателей адреса IDXi (QXi) или для указателей адреса GPR (QRi)
N	0	Чтение	Так как IDXi может содержать только четные числа, то нулевой бит всегда равен нулю

Регистры аккумулятора

40-битный аккумулятор состоит из следующих регистров: MAH, MAL, см. рисунки 6.9, 6.10 и таблицы 6.7, 6.8, и младшего байта регистра MSW. Регистры MAH и MAL не являются битадресуемыми SFR.

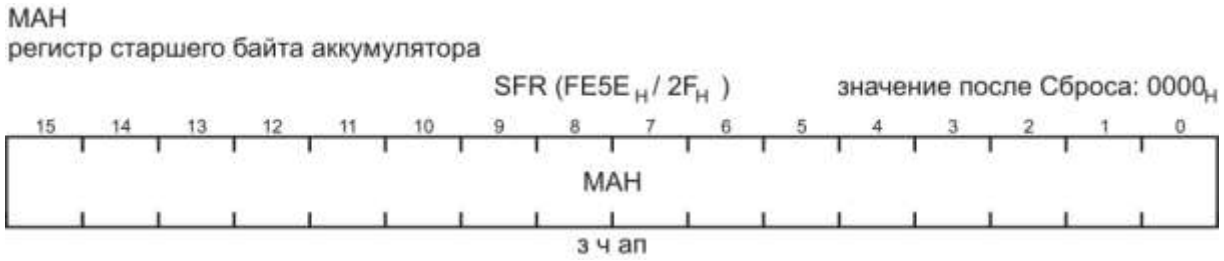


Рисунок 6.9 – Формат регистра MAH

Таблица 6.7 – Функциональное назначение полей регистра MAH

Поле	Биты	Тип	Описание
MAH	15-0	Запись Чтение Аппаратное влияние	Старший байт аккумулятора MAC Содержит биты с 31 по 16 40-битного аккумулятора блока MAC

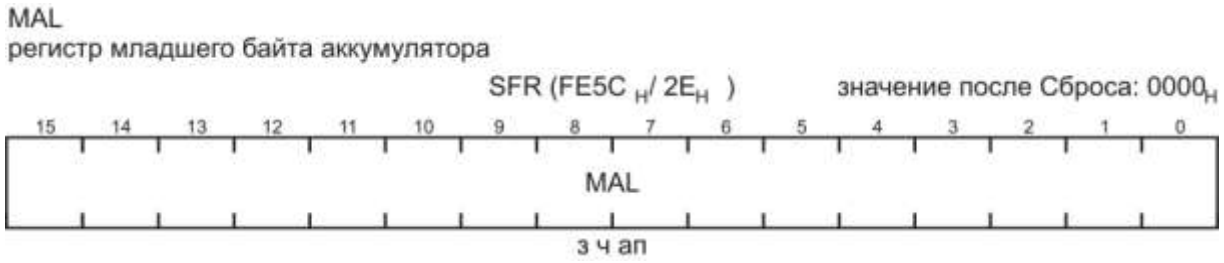


Рисунок 6.10 – Формат регистра MAL

Таблица 6.8 – Функциональное назначение полей регистра MAL

Поле	Биты	Тип	Описание
MAL	15-0	Запись Чтение Аппаратное влияние	Младший байт аккумулятора MAC Содержит биты с 15 по 0 40-битного аккумулятора блока MAC

Примечание – Регистр MAL автоматически обнуляется, если происходит запись в регистр MAH с помощью стандартных команд ЦПУ.

Статусный регистр MSW

Битадресуемый регистр MSW (область SFR), см. рисунок 6.11, таблицу 6.9, отображает текущее состояние блока MAC. Данный регистр состоит из 8-битного расширения аккумулятора MAE и 7 флагов.

Рисунок 6.11 – Формат регистра MSW

Таблица 6.9 – Функциональное назначение полей регистра MSW

Поле	Биты	Тип	Описание
MAE	7-0	Запись Чтение Аппаратное влияние	Расширение аккумулятора. Восемь значащих битов 40-битного аккумулятора блока MAC.
N	8	Запись Чтение Аппаратное влияние	Флаг отрицательного результата: 0 Отрицательный результат блока MAC. 1 Положительный результат блока MAC.
Z	9	Запись Чтение Аппаратное влияние	Флаг нуля: 0 Результат не равен нулю. 1 Результат равен нулю.
C	10	Запись Чтение Аппаратное влияние	Флаг переноса: 0 Не произошел перенос/заем. 1 Произошел перенос/заем.
SV	11	Запись Чтение Аппаратное влияние	Флаг переполнения: 0 Не произошло 40-битного переполнения. 1 Произошло 40-битное переполнение.
E	12	Запись Чтение Аппаратное влияние	Флаг расширения: 0 MAE не содержит значащих битов. 1 MAE содержит значащие биты.
SL	13	Запись Чтение Аппаратное влияние	Флаг ограничения: 0 Не произошло автоматического 32-битного ограничения. 1 Произошло автоматическое 32-битное ограничение.
MIR	15	Запись Чтение Аппаратное влияние	Запрос на прерывание блока MAC: 0 Нет запроса на прерывание блока MAC. 1 Запрос на прерывание блока MAC.

Примечание – Статусные флаги блока MAC изменяются (при необходимости) при выполнении команды. Они не изменяются при выполнении стандартных команд ЦПУ.

Регистр управления блока MAC

Битадресуемый регистр MCW области SFR, см. рисунок 6.12, таблицу 6.10, управляет работой блока MAC и определяет функциональность прерывания блока MAC.

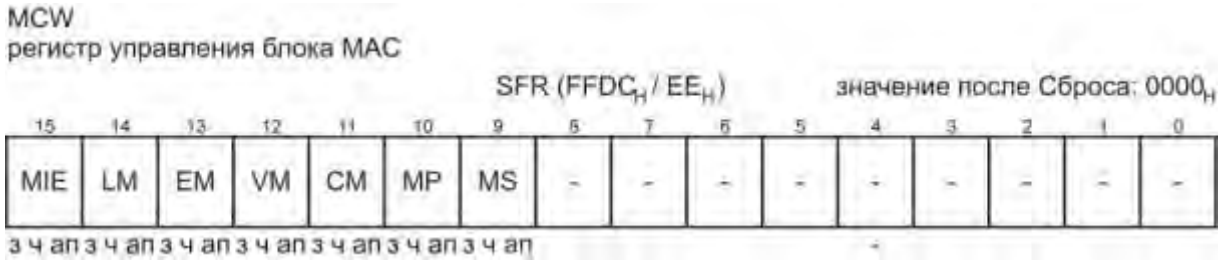


Рисунок 6.12 – Формат регистра MCW

Таблица 6.10 – Функциональное назначение полей регистра MCW

Поле	Биты	Тип	Описание
MS	9	Запись Чтение	Контроль ограничения: 0 Запрещение автоматического ограничения. 1 Разрешение автоматического ограничения.
MP	10	Запись Чтение	Управление однобитным делителем: 0 Запрещение сдвига результата умножения. 1 Разрешение сдвига результата умножения.
CM	11	Запись Чтение Аппаратное влияние	Маска переноса: 0 Флаг C не может генерировать прерывание. 1 Флаг C может генерировать прерывание.
VM	12	Запись Чтение	Маска переполнения: 0 Флаг SL не может генерировать прерывание. 1 Флаг SL может генерировать прерывание.
EM	13	Запись Чтение	Маска расширения: 0 Флаг E не может генерировать прерывание. 1 Флаг E может генерировать прерывание.
LM	14	Запись Чтение	Маска ограничения: 0 Флаг SL не может генерировать прерывание. 1 Флаг SL может генерировать прерывание.
MIE	15	Запись Чтение	Разрешение на прерывание блока MAC: 0 Общее запрещение на прерывание блока MAC. 1 Общее разрешение на прерывание блока MAC.

Регистр MRW повторения блока MAC

Регистр MRW области SFR содержит число повторений, которые должна произвести команда. Формат регистра MRW представлен на рисунке 6.13, функциональное назначение полей регистра MRW – в таблице 6.11.

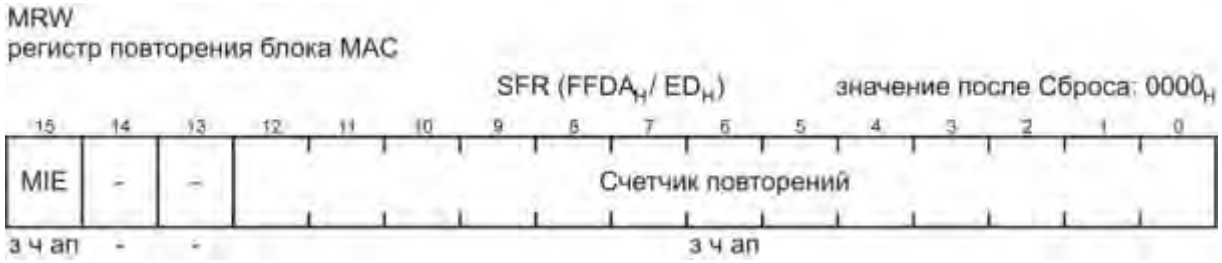


Рисунок 6.13 – Формат регистра MRW

Таблица 6.11 – Функциональное назначение полей регистра MRW

Поле	Биты	Тип	Описание
Счетчик повторений	12-0	Запись Чтение Аппаратное влияние	13-битное целое число без знака. Число повторений команды минус 1.
MR	15	Запись Чтение Аппаратное влияние	Флаг повторения. Устанавливается, когда выполняется повторяемая команда.

6.5 Система команд MAC

В данном подразделе представлено подробное описание команд блока MAC микроконтроллера 1887ВЕ3Т. Система команд включает в себя следующие группы:

- 32-битные арифметические команды;
- команды сдвига;
- команды сравнения;
- команды пересылки данных.

Все команды MAC являются 32-битными, для кодирования каждой команды используется 4 байта.

Описание команд

Операнды

opX – непосредственные данные;
(opX) – содержимое opX;
(opXn) – содержимое бита n операнда opX;
((opX)) – значение, взятое по адресу, равному содержимому opX (т. е. содержимое opX является указателем требуемого значения).

Действие

(opX) ← (opY) – (opY) копируется в (opX);

(opX) + (opY) – (opX) прибавляется к (opY);

(opX) - (opY) – (opY) вычитается из (opX);

(opX) * (opY) – (opX) умножается на (opY);

(opX) ⇔ (opY) – (opX) сравнивается с (opY);

<< – логический сдвиг влево;

>> – логический сдвиг вправо;

>>a – арифметический сдвиг вправо;

(opX) || (opY) – объединение (opX) (MSW) и (opY) (LSW).

Режимы адресации

Rw_n или Rw_m – 2-байтовые регистры общего назначения GPR, «n» и «m» – величина от 0 до 15;
[…] – косвенная адресация в памяти слова;
Mxx – регистры блока MAC: MSW, MAH, MAL, MAS, MRW, MCW;
ACC – аккумулятор блока MAC, состоящий из младшего байта регистра MSW, регистров MAH и MAL;
#datax – непосредственные данные (число значащих битов представлены величиной «x»).

Флаги состояния

\- – Выполнение команды не влияет на флаг.

* – Стандартная установка значения флага.

Регистры, используемые для двойной косвенной адресации

Любой GPR – первый указатель адреса;

QR0/QR1 – регистры смещения для первого указателя адреса GPR;

IDX0/IDX1 – второй указатель адреса;

QX0/QX1 – регистры смещения для второго указателя адреса.

Символы $[Rw_n\otimes]$ и $[IDX_i\otimes]$ обозначают различные комбинации указателя адреса при его изменении, возможные комбинации показаны в таблице 6.12.

Синтаксис повторяемых команд

Повторяемые команды CoXXX при повторении выглядят следующим образом:

repeat #data5 times CoXXX…
или
repeat MRW times CoXXX…

При записи значения в регистр MRW команда повторится (MRW[12:0] + 1) раз, следовательно, максимальное количество повторений команды будет $2^{13} = 8192$ раза.

Целочисленная величина #data5 определяет число повторов команды. Таким образом, величина #data5 должна быть меньше 32, и максимальное количество повторений команды CoXXX – 31 раз.

Величина сдвига

Сдвиговый регистр позволяет произвести сдвиг вправо/влево от 0 до 8 бит. При величине сдвига, большей 8, производится сдвиг на 8 бит.

Код команды

X – 4-битный код режима адресации IDX;

qqq – 3-битный код смещения GPR;

rrrr:r – 5-битное поле повтора;

ssss: – 4-битная непосредственная величина сдвига.

Подробное описание команд блока MAC находится в приложении Д.

Команды блока MAC

Команды и режимы адресации блока MAC представлены в таблице 6.12.

Таблица 6.12 – Команды и режимы адресации

Мнемокод	Режимы адресации	Повтор
1	2	3
CoMul CoMulu CoMulus CoMulsu CoMul- CoMulu- CoMulus- CoMulsu- CoMul + rnd CoMulu + rnd CoMulus + rnd CoMulsu + rnd	Rw_n, Rw_m Rw_n, [$Rw_m\otimes$] [$IDX_i\otimes$], [$Rw_m\otimes$]	Нет
CoMAC CoMACu CoMACus CoMACsu CoMAC- CoMACu- CoMACus- CoMACsu- CoMAC + rnd CoMACu + rnd CoMACus + rnd CoMACsu + rnd CoMACR CoMACRu CoMACRus CoMACRsu CoMACR + rnd CoMACRu + rnd CoMACRus + rnd CoMACRsu + rnd	Rw_n, Rw_m Rw_n, [$Rw_m\otimes$] [$IDX_i\otimes$], [$Rw_m\otimes$]	Нет Да Да

Окончание таблицы 6.12

1	2	3
CoNOP	[Rw_n ⊗] [IDX_i ⊗] [IDX_i ⊗], [Rw_m ⊗]	Да
CoNEG CoNEG + rnd CoRND	–	Нет
CoSTORE	Rw_n,CoReg [Rw_n ⊗],CoReg	Нет Да
CoMov	[IDX_i ⊗], [Rw_m ⊗]	Да
CoMACM CoMACMu CoMACMus CoMACMsu CoMACM- CoMACMu- CoMACMus- CoMACMsu- CoMACM + rnd CoMACMu + rnd CoMACMus + rnd CoMACMsu + rnd	[IDX_i ⊗], [Rw_m ⊗]	Да
CoMACMRu CoMACMRus CoMACMRsu CoMACMR + rnd CoMACMRu + rnd CoMACMRus + rnd CoMACMRsu + rnd	[IDX_i ⊗], [Rw_m ⊗]	Да
CoADD CoADD2 CoSUB CoSUB2 CoSUBR CoSUB2R CoMAX CoMIN	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i ⊗], [Rw_m ⊗]	Нет Да Да
CoLOAD CoLOAD- CoLOAD2 CoLOAD2- CoCMP	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i], [Rw_m⊗]	Нет Нет Нет
CoSHL CoSHR CoASHR CoASHR + rnd	#data4 Rw_m [Rw_m⊗]	Нет Да Да
CoABS	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i⊗], [Rw_m⊗]	Нет

7 Отладочная система микроконтроллера

Устройство обеспечения отладки на кристалле (далее «отладочная система» или блок OCDS) позволяет выполнять эмуляцию аппаратного окружения для большинства потребителей при минимальной стоимости затрат. Отладочная система управляется непосредственно внешним устройством через выводы отладочного интерфейса. Структурная схема отладочной системы микроконтроллера приведена на рисунке 7.1.

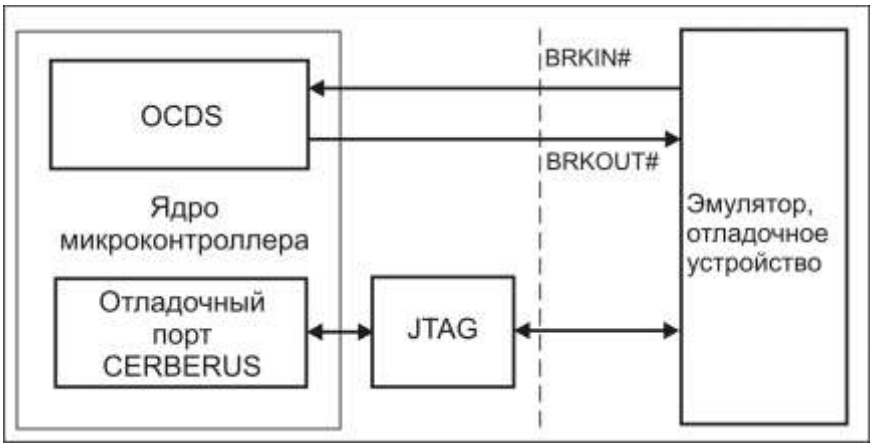


Рисунок 7.1 – Отладочная система микроконтроллера

Отладочная система микроконтроллера состоит из трех блоков:

- блок OCDS генерации контрольных точек останова программы (далее «breakpoint»);
- отладочный порт CERBERUS;
- блок JTAG.

Характеристики блока OCDS:

- аппаратные, программные breakpoint, а также выводы для внешних breakpoint;
- до четырех контрольных точек останова программы breakpoint;
- маскируемое сравнение аппаратных breakpoint;
- пошаговый режим для монитора (monitor) или останова CPU Halt;
- видимость программного счетчика в режиме Halt;
- совместимость с отладочным стандартом Nexus класса 1 и выше.

Назначение блока OCDS в том, что он позволяет вести отладку программного обеспечения, запускаемого пользователем. Это обеспечивается при помощи внешнего отладочного устройства, которое управляет блоком OCDS через независимый отладочный порт JTAG.

Характеристики блока CERBERUS:

- настраиваемый последовательный канал для доступа к полному 24-битному пользовательскому адресному пространству;
- эффективный высокопроизводительный протокол;
- управление внешним хостом всеми транзакциями;
- использование JTAG интерфейса для управления и передачи данных;
- настраиваемая функциональность чтения-записи памяти (режим RW);
- полная поддержка коммуникации между монитором и внешним отладчиком;
- дополнительная защита от ошибок;
- механизм безопасности, позволяющий только санкционированный доступ;
- начальная трассировка для чтения/записи, включаемая блоком OCDS;
- быстрая трассировка посредством передачи по внешней шине;
- регистр анализа для ситуаций блокирования на внутренней шине;

- возможность управления несколькими блоками CERBERUS через один JTAG интерфейс;

- предусмотрена возможность использования API (Application Programming Interface), разработанная фирмой Infineon для ускорения вызова и поддержки нескольких отладочных приложений и для разрешения многозадачного совместного использования блока JTAG.

Основное предназначение блока CERBERUS – это подключение JTAG интерфейса в качестве независимого порта для блока OCDS. Внешние аппаратные средства отладки могут обратиться к регистрам OCDS и произвольным адресам памяти. Архитектура системы хорошо адаптирована для поддержки нескольких отладочных приложений и для разрешения многозадачного совместного использования единственного JTAG интерфейса. До четырех блоков CERBERUS могут быть подключены к блоку JTAG и могут управляться стандартными отладчиками в одном сеансе отладки. Блок JTAG и API представляют собой проверенный и прямой интерфейс для стандартных отладочных устройств и производят арбитраж доступа к JTAG интерфейсу прозрачным способом.

7.1 Блок OCDS

Основная концепция работы блока состоит в обработке отладочных событий и определении действий, когда отладочное событие произошло, то есть запуска процесса отладки. Структурная схема блока OCDS приведена на рисунке 7.2.

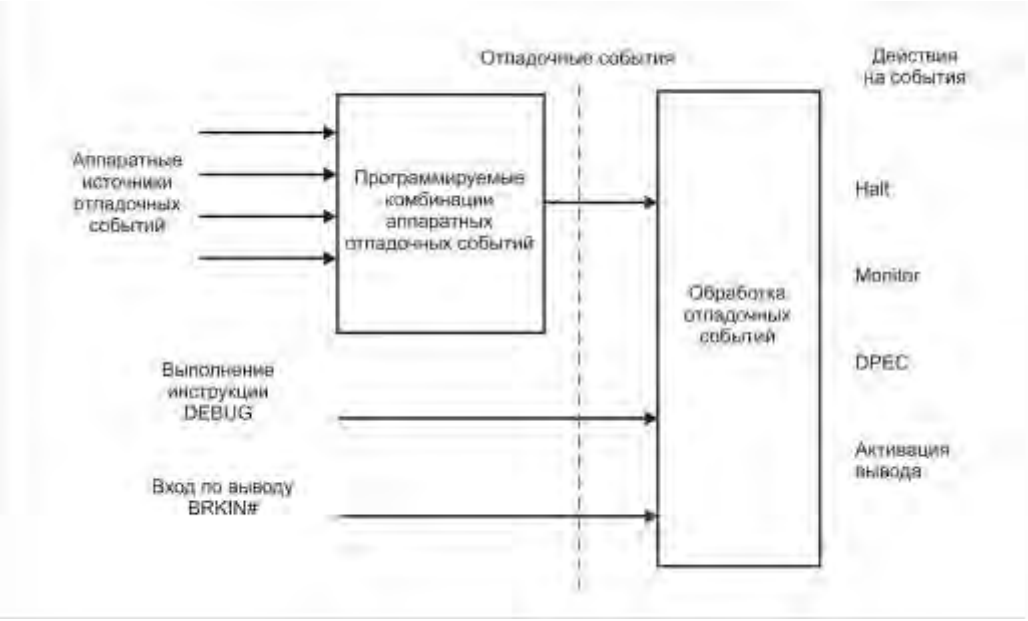


Рисунок 7.2 – Структурная схема блока OCDS

Следующие события являются отладочными:
- комбинация аппаратных источников событий, запускающих отладку;
- выполнение отладочной команды;
- активизация входа остановки отладки (break) по выводу BRKIN#.

Действия на отладочные события:
- останов CPU Halt;
- вызов монитора monitor;
- передача данных DPEC, выполняемая блоком CERBERUS;
- активизация внешнего выхода по выводу BRKOUT#.

В таблице 7.1 приведено описание регистров блока OCDS.

Таблица 7.1 – Регистры блока OCDS

Название регистра	Назначение
DBGSR	Регистр состояния отладки
DEXEVT	Определяет действие, если выбрано внешнее событие по выводу BRKIN#
DSWEVT	Определяет действие, если выполняется отладочная команда
DTREVT	Комбинации критериев для выбора аппаратных отладочных событий
DCMPDP	Регистр программирования данных для регистров сравнения DCMPx
DCMPSP	Регистр выбора и программирования для регистров сравнения DCMPx
DCMP0	Регистр 0 аппаратного события сравнения на равенство
DCMP1	Регистр 1 аппаратного события сравнения на равенство
DCMP2	Регистр 2 аппаратного события сравнения на равенство
DCMPG	Регистр сравнения диапазона аппаратного события (старший)
DCMPL	Регистр сравнения диапазона аппаратного события (младший)
DIP	Регистр указателя команд
DIPX	Регистр расширения указателя команд
DTIDR	Регистр ID задачи

Разрешение и запрещение работы блока OCDS

По умолчанию, работа блока OCDS запрещена в порядке защиты системы в течение нормальной работы. Генерация событий может быть только тогда, когда OCDS разрешен. Модуль OCDS имеет сигнал разрешения, который соединен с внутренним сигналом сброса JTAG. Это означает, что OCDS разрешен, когда блок JTAG не в состоянии сброса. Это происходит всегда, когда внешнее отладочное устройство использует CERBERUS.

Блок OCDS можно дополнительно разрешить программным способом. Чтобы избежать неумышленного разрешения некорректной пользовательской программы, необходимо иметь следующие условия истинными:

- OCDS запрещен;
- DTREVT.MUX_E=10B;
- DTREVT.SELECT_E=00B;
- DCMP0 сравнение соответствует (независимо от SELECT_E);
- текущая запись DBGSR.DEBUG_ENABLED=1B, таким образом, монитор должен выполнить следующее:
  - записать F0FCH в DCMP0 (адрес DBGSR);
  - записать 2200H в DTREVT;
  - записать 0001H в DBGSR.

Если блок OCDS разрешен программным обеспечением, он может быть запрещен только сбросом RESET.

Сброс с переходом в режим останов Halt

Микроконтроллер может быть введен напрямую в режим останов Halt после сброса. Это управляется битом CCONF.RST_HLT в блоке JTAG. Для сброса с переходом в режим останов необходимо выполнить три шага:

- Установить бит CCONF.RST_HLT до выполнения сброса.
- Установить бит DBGSR.DEBUG_STATE для перевода в режим останов после выполнения сброса RESET.
- Очистить бит CCONF.RST_HLT.

Для вывода микроконтроллера из режима останов, необходимо установить бит CCONF.RST_HLT (записать 1).

Источники отладочных событий

В таблице 7.2 приведены источники запуска аппаратных отладочных событий.

Таблица 7.2 – Аппаратные источники событий запуска отладки

Источник запуска	Размер, бит	Назначение
TASKID	16	TASKID в DTIDR регистре
IP	24	Указатель команд
R_ADR	24	Адрес данных при чтении
W_ADR	24	Адрес данных при записи
DA	16	Значение данных (при чтении или записи)

TASKID – это содержание регистра DTIDR. TASKID используется в расширенных системах при операциях реального времени для запоминания ID текущей задачи. Все источники запуска сравниваются и комбинируются в аппаратном устройстве генерации запуска отладочного события. Аппаратное устройство генерации запуска отладочного события программируется записью в управляющий регистр отладочных событий. Оно состоит из двух частей. Верхняя часть служит для одного диапазона сравнения, а нижняя часть служит для трех сравнений на равенство. На структурной схеме, приведенной на рисунке 7.3, часть блока, отвечающего за сравнения на равенство, может быть сконфигурирована для двух маскируемых сравнений на равенство.

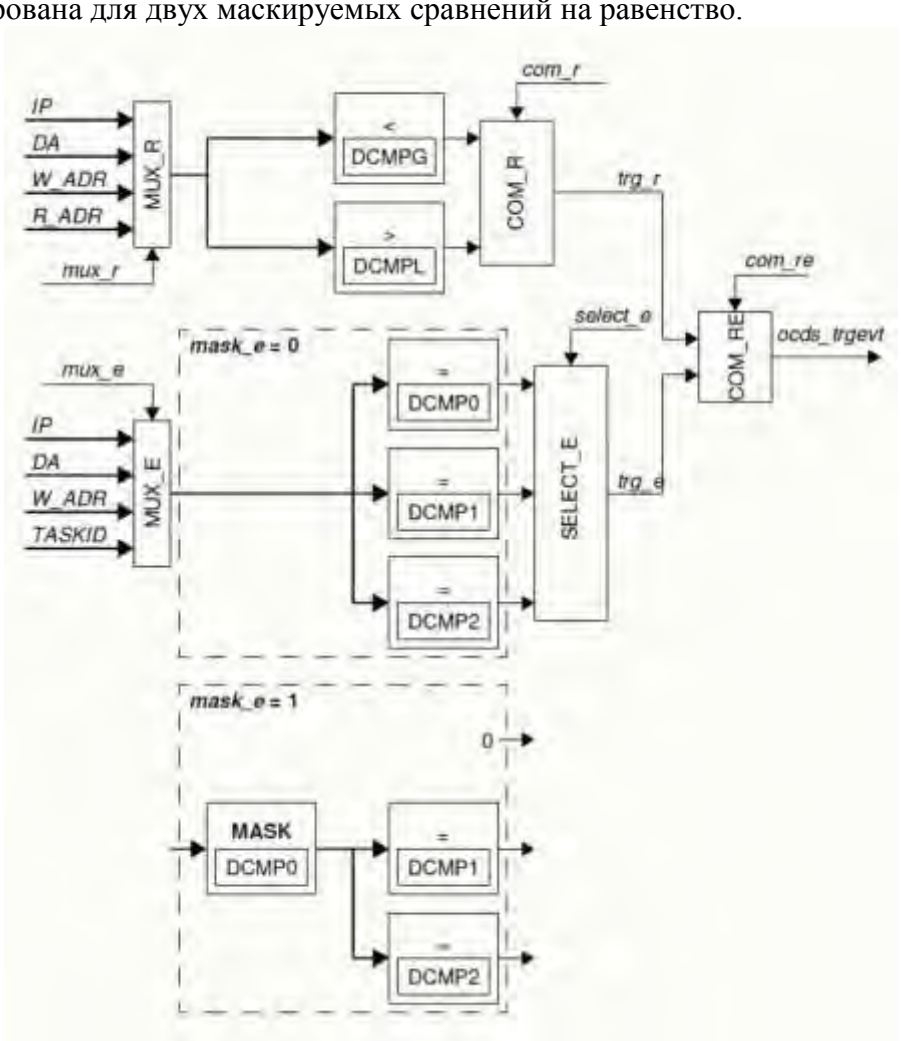


Рисунок 7.3 – Структурная схема аппаратного устройства генерации сигнала запуска отладочного события

Выполнение отладочных команд

Выполнение отладочных команд является механизмом непосредственного запуска процесса отладки, то есть является запуском отладочного события. Это может быть использовано, например, отладочным устройством (отладчиком) для трассировки кода в оперативной памяти RAM в процессе выполнения breakpoint. Специальная команда DEBUG (код $D140_H$) определяет, что выполняются команда «Пользовательский режим», и ее выполнение зависит от того, разрешен ли OCDS. Если OCDS разрешен, команда DEGUG, при наличии отладочного события, запускает процесс отладки. Реакция на отладочное событие определяется содержанием регистра управления отладки DSWEVT. Если блок OCDS не разрешен, команда DEBUG выполняется как NOP.

Вывод отладочного прерывания BRKIN#

Внешний вывод отладочного прерывания предусмотрен как возможность для отладочного устройства (отладчика) выполнять асинхронное прерывание микроконтроллера. Предпринятое действие, когда установлен сигнал BRKIN#, определено в регистре управления отладкой DEXEVT. Этот вход срабатывает по отрицательному фронту сигнала при удержании низкого уровня в течение не менее двух периодов системного тактового сигнала.

Приоритеты событий

Есть возможность более чем одного события в одном цикле. В таком случае обработка событий идет в соответствии с приоритетами. Обработка событий выстраивается в последовательность, в которой сначала обрабатывается событие с высшим приоритетом, а потом с более низким. События и приоритеты приведены в таблице 7.3.

Таблица 7.3 – Приоритеты отладочных событий

Событие	Управляющий регистр отладочного события	Приоритет
Вывод входного сигнала BRKIN#	DEXEVT	1 (высший)
Выполнение команды DEBUG	DSWEVT	2
Комбинация аппаратных источников	DTREVT	3 (низший)

Действия на отладочные события

Когда блок OCDS разрешен и появляется отладочное событие, выполняется одно из действий, приведенное в таблице 7.4.

Таблица 7.4 – Действия на отладочное событие

Действие на отладочное событие	Возможности пользователя	Возможность прерывания	Остановка break до действия
Активирование внешнего вывода	–	–	–
Включение передачи данных DPEC	Занятие цикла памяти для DPEC	–	Только для адресов программы запуска
Вызов монитора	Стек. Адресное пространство пользователя	Да (после входа)	
Останов Halt	–	–	

Включение передачи данных DPEC

Включение блока CERBERUS для выполнения ожидаемой передачи – это одна из акций, которая может быть определена для случая, когда имеет место отладочное событие. Это может быть использовано в критичных подпрограммах, в которых система не может быть прервана для передачи адреса в регистр RWDATA и трассировки через отладочный порт CERBERUS.

Вызов монитора

Вызов монитора специальной аппаратной отладочной ловушкой (trap номер 8, адрес вектора 20_H) является одним из возможных действий при возникновении отладочного события. Этот trap имеет наивысший приоритет, однако подпрограмма монитора может ограничить собственный уровень приоритета при переустановке бита отладочного флага DEBTRAP в регистре trap-флагов TFR и записать поле ILVL в регистре PSW.

Этот короткий вход в прерываемый монитор позволяет гибко отлаживать программное окружение, удовлетворяющее многим требованиям эффективного выполнения отладки в системах реального времени. Например, безопасность критического кода может быть обеспечена, пока отладочное устройство активно. Монитор завершается командой RETI. Бит флага отладки DEBTRAP должен быть очищен при выходе из программы обработки системных прерываний TRAP, иначе она будет запущена снова. Модель отладки приведена на рисунке 7.4.

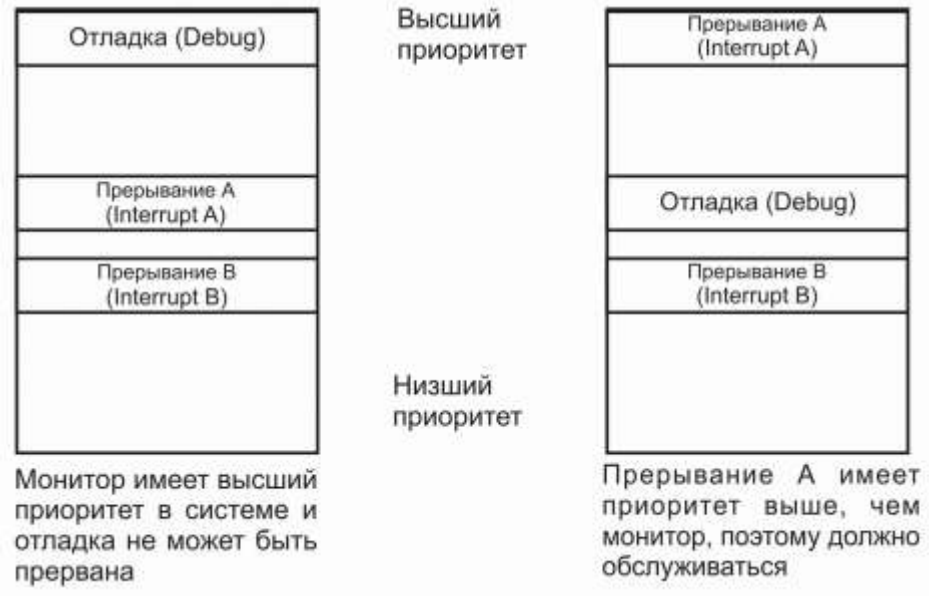


Рисунок 7.4 – Простая и расширенная модели отладки

Структура непрерываемой подпрограммы монитора:
- запуск процесса (непрерываемого);
- выполнить DBGSR=0000_H;
- очистить бит DEBTRAP в регистре TFR;
- возврат в пользовательскую программу командой RETI.

Структура прерываемой подпрограммы монитора:
- установить битовое поле DBGSR.DEBUG_STATE=00_B (пользовательский режим);
- очистить бит DEBTRAP в регистре TFR;
- уменьшить уровень прерывания ILVL в регистре PSW;
- запуск процесса;
- установить DBGSR=0000_H.

Возврат в пользовательскую программу командой RETI

Уменьшение приоритета прерывания монитора может вызвать переполнение стека. Если задача, вызывающая отладочное событие, имеет приоритет выше, чем монитор, то монитор снова и снова будет помещаться в стек. Должно быть предусмотрено, чтобы сам монитор не вызывал отладочное событие, иначе он будет стартовать снова и снова, и стек будет переполнен.

Режим останов Halt

Система приостанавливает режимом Halt выполнение потока команд и не отвечает на прерывания. Управление переходит к внешним средствам отладки, чтобы опросить, полностью прочитать и обновить через отладочный порт CERBERUS. Центральный процессор возобновляет пользовательский режим, когда внешнее аппаратное отладочное устройство сбросит битовое поле DEBUG_STATE в регистре DBGSR в пользовательский режим. Также нужно сбросить биты OCDS_P_SUSPEND и EVENT_SOURCE в регистре DBGSR.

Активация внешнего вывода

Изменение уровня сигнала на внешнем выводе может быть определено как отладочное событие. Это должно быть использовано в критичных подпрограммах, в которых работа системы не может быть прервана для сообщения внешнему миру о возникновении специфического события. Эта возможность может быть полезной для синхронизации внутренних и внешних аппаратных средств. В большинстве случаев активный уровень вывода низкий после выполнения условий срабатывания.

Пошаговая отладка

Пошаговая отладка может быть в режиме Halt или с отладочным монитором.

Пошаговая отладка в режиме Halt

Для этого поведения условия срабатывания должны быть всегда истинными (например: включение IP диапазона с DCMPL=000000_H, DCMPG=000001_H и битовыми полями COM_R=11_B, COM_RE=0_B, BREAK_AFTER_MAKE=1_B в регистре DTREVT). После каждого рестарта микроконтроллер будет остановлен после выполнения очередной команды.

Пошаговая отладка с отладочным монитором

Преимуществом этого типа пошаговой отладки является то, что система может обслуживать высокоприоритетные запросы прерывания. Основной подход к выполнению пошаговой отладки в режиме Halt состоит в двух различных функциях.

Действием на отладочное событие является вызов монитора.

Коды подпрограммы обслуживания прерывания и отладочный монитор могут не быть частью адресного пространства IP, где запускается отладка.

Рекомендована корректировка адресного пространства IP запуска отладочного события для текущей (C-) функции пользовательской программы. Это приводит к надшаговому выполнению отладки, если подфункция вызывается внутри функции. Если требуется выполнение каких-то действий во время шагов (отладки), то может быть введено добавочное адресное пространство IP для входа в подфункцию, и когда вход в подфункцию произведен, диапазон адресов IP для отладки изменяется для обеспечения выполнения подфункции. В таблице 7.5 приведены спецификации регистров блока OCDS.

Таблица 7.5 – Спецификации регистров блока OCDS

Название регистра	Адрес ESFR	Тип	Назначение
1	2	3	4
DBGSR	$F0FC_H$	Аппаратное влияние	Регистр состояния отладки
DEXEVT	$F0F2_H$	–	Определяет действие, если выбрано внешнее событие по выводу BRKIN#

Окончание таблицы 7.5

1	2	3	4
DSWEVT	$F0F4_H$	–	Определяет действие, если выполняется отладочная команда
DTREVT	$F0F0_H$	–	Комбинации критериев для выбора аппаратных отладочных событий
DCMPDP	$F0EE_H$	–	Регистр программирования данных для регистров сравнения DCMPx
DCMPSP	$F0EC_H$	–	Регистр выбора и программирования для регистров сравнения DCMPx
DCMP0	–	–	Регистр 0 аппаратного события сравнения на равенство
DCMP1	–	–	Регистр 1 аппаратного события сравнения на равенство
DCMP2	–	–	Регистр 2 аппаратного события сравнения на равенство
DCMPG	–	–	Регистр сравнения диапазона аппаратного события (старший)
DCMPL	–	–	Регистр сравнения диапазона аппаратного события (младший)
DIP	$F0F8_H$	Аппаратное влияние	Регистр указателя команд
DIPX	$F0FA_H$	Аппаратное влияние	Регистр расширения указателя команд
DTIDR	$F0D8_H$	Аппаратное влияние	Регистр ID задачи
Примечание – Регистры DCMP0, DCMP1, DCMP2, DCMPG, DCMPL доступны с помощью регистров DCMPSP и DCMPSD.			

Регистры управления отладочными событиями DEXEVT, DSWEVT, DTREVT

Каждый возможный источник отладочного события имеет соответствующий регистр, который определяет действие, предпринятое в случае появления отладочного события. Регистры управления отладочными событиями имеют одинаковую структуру для всех определяемых текущих источников. Формат регистров DEXEVT и DSWEVT представлен на рисунке 7.5, функциональное назначение полей регистров DEXEVT, DSWEVT – в таблице 7.6.

Рисунок 7.5 – Формат регистров DEXEVT и DSWEVT

Таблица 7.6 – Функциональное назначение полей регистров DEXEVT, DSWEVT

Поле	Биты	Тип	Описание
EVENT_ SOURCE	2-0	Запись Чтение	Определение действия на отладочное событие: 000 Нет действия. 001 Режим программной отладки. 010 Остановочный отладочный режим Halt. 011 Зарезервировано. 100 Зарезервировано. 101 Выполнение DPEC. 110 Зарезервировано. 111 Установка бита источника в DBGSR.
PERIPHERALS_ STOP	3	Запись Чтение	0 Периферийные устройства не используются этим событием. 1 Периферийные устройства приостанавливают операции в случае события.
0	4	–	Зарезервирован.
ACTIVATE_PIN	5	Запись Чтение	0 Внешний вывод всегда неактивен. 1 Внешний вывод активен в течение отладочного события.
0	15-6	–	Зарезервированы.

Поле EVENT_SOURCE определяет, что выполняется, когда происходит соответствующее отладочное событие. Спецификатор события может иметь одно из указанных значений. Для режимов программной отладки и Halt два младшие бита EVENT_SOURCE устанавливают поле DEBUG_STATE в регистре DBGSR.

Бит PERIPHERALS_STOP управляет режимом операций периферийных устройств, в случае возникновения отладочного события. Если бит PERIPHERALS_STOP установлен, то бит OCDS_P_SUSPEND в регистре DBGSR тоже будет установлен, но это может быть в режиме программной отладки или Halt. Это принуждает останавливаться соответствующие периферийные устройства. Формат регистра DTREVT представлен на рисунке 7.6, функциональное назначение полей регистра DTREVT – в таблице 7.7.

Рисунок 7.6 – Формат регистра DTREVT

Таблица 7.7 – Функциональное назначение полей регистра DTREVT

Поле	Биты	Тип	Описание
1	2	3	4
EVENT_ ACTION	2-0	Чтение Запись	Идентично полю EVENT_SOURCE в DEXEVT
PERIPHERALS_ STOP	3	Чтение Запись	Идентично полю PERIPHERALS_STOP в DEXEVT
BREAK_AFTER_ MAKE	4	Чтение Запись	0 Остановка до выполнения (только IP) 1 Остановка после выполнения (только IP)

Окончание таблицы 7.7

1	2	3	4
ACTIVATE_PIN	5	Чтение Запись	Идентично полю ACTIVATE_PIN в DEXEVT
MUX_R	7-6	Чтение Запись	Область сравнения входного мультиплексора: 00 Указатель команд IP 01 Значение данных DA 10 Адрес записи W_ADR 11 Адрес чтения R_ADR
MUX_E	9-8	Чтение Запись	Управление входным мультиплексором сравнения на равенство: 00 Указатель команд IP 01 Значение данных DA 10 Адрес записи W_ADR 11 ID задачи TASKID
COM_R	11-10	Чтение Запись	Выбор диапазона сравнения, см. таблицу 7.8
MASK_E	12	Чтение Запись	0 Немаскированное сравнение на равенство 1 Маскированное сравнение на равенство
SELECT_E	14-13	Чтение Запись	Выбор сравнения на равенство, см. таблицу 7.9
COM_RE	15	Чтение Запись	Комбинация сравнения по равенству и диапазону: 0 Сигнал ocds_trgevt формируется логической функцией (trg_r OR trg_e) 1 ocds_trgevt формируется логической функцией (trg_r AND trg_e) *

_____

* Вариант COM_RE = 0 предназначается для случая mux_r = mux_e и для того, чтобы иметь только четыре источника событий. В случае различных сравниваемых источников событий эта функция приводит к сложному поведению, потому что такие аппаратные источники отладочных событий как, например, IP и W_ADR, появляются на различных этапах работы конвейера.

Поле COM_R позволяет включать сравнение диапазона в формирование сигнала ocds_trgevt. Для сравнений в диапазоне регистр DCMPG использует верхнюю границу, а регистр DCMPL нижнюю границу диапазона. При сравнении за пределами диапазона все наоборот. В таблице 7.8 показаны значения поля COM_R регистра DTREVT.

Таблица 7.8 – Поле COM_R регистра DTREVT

Значение	Сигнал trg_r
00_B	0 (не разрешено)
01_B	В диапазоне: 1, если DCMPG > (на входе) > DCMPL, иначе 0
10_B	Зарезервировано
11_B	Вне диапазона: 1, если DCMPL < на входе или на входе < DCMPG

Бит MASK_E устанавливает различие между маскированным и немаскированным входом для сравнения на равенство. В маскированном случае регистр DCMP0 управляет соответствующими битами для сравнения. Все биты входного сигнала, для которых соответствующие биты DCMP0 равны «0», также устанавливаются в «0» до компарации.

Сравниваемые значения в DCMP1 и DCMP2 должны быть равны «0», когда маска DCMP0 равна «0» иначе при сравнении не будет соответствия.

Поле SELECT_E разрешает включать сравнение на равенство в генерацию сигнала ocds_trgevt и выбирает режимы как показано в таблице 7.9. Нужно заметить для маскируемого сравнения, что поле SELECT_E должно быть установлено в 10_B или 11_B.

Таблица 7.9 – Формирование сигнала trg_e

Значение	Сигнал mask_e	Сигнал trg_e
00_B	0	0 (не разрешено)
01_B		1, если DCMP0 равен, иначе 0
10_B		1, если DCMP0 или DCMP1 равны, иначе 0
11_B		1, если DCMP0 или DCMP1 или DCMP2 равны, иначе 0
00_B	1	0 (не разрешено)
01_B		0 (всегда)
10_B		1, если DCMP1 равен, иначе 0
11_B		1, если DCMP1 или DCMP2 равны, иначе 0

Регистр состояния отладки DBGSR

Регистр состояния отладки DBGSR, см. рисунок 7.7, таблицу 7.10, содержит некоторую информацию о текущем состоянии OCDS, включающую:

- бит индикации разрешения поддержки отладки;
- источник последнего отладочного события;
- состояние отладочной системы.

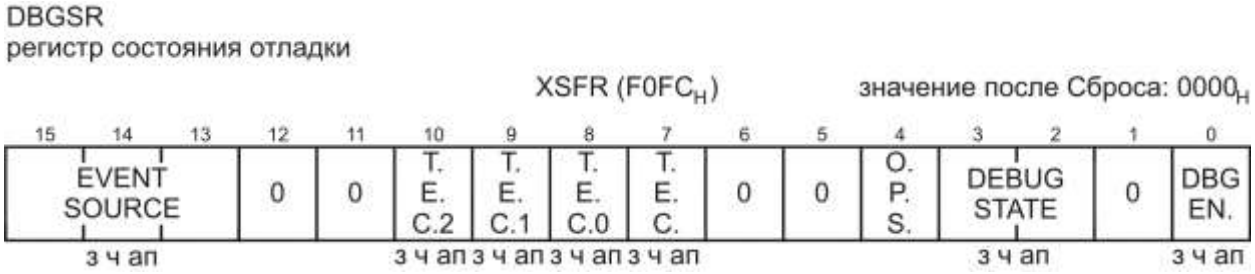


Рисунок 7.7 – Формат регистра DBGSR

Таблица 7.10 – Функциональное назначение полей регистра DBGSR

Поле	Биты	Тип	Описание
1	2	3	4
DEBUG_ ENABLED	0	Запись Чтение Аппаратное влияние	0 OCDS запрещено 1 OCDS разрешено
–	1	0	Зарезервирован
DEBUG_STATE	3-2	Запись Чтение Аппаратное влияние	Текущее состояние отладки: 00 Пользовательский режим 01 Программный отладочный режим 10 Остановочный отладочный режим (Halt) 11 Зарезервировано
OCDS_P_ SUSPEND	4	Запись Чтение Аппаратное влияние	0 Нет эффекта 1 Соответствующая периферия приостанавливает свои операции
–	6-5	0	Зарезервировано

Окончание таблицы 7.10

1	2	3	4
TRGEVT_E_ CMP	7	Запись Чтение Аппаратное влияние	0 Диапазон сравнения не соответствует 1 Сравнение соответствует для текущего события
TRGEVT_E_ CMP0	8	Запись Чтение Аппаратное влияние	0 Сравнение 0 на равенство не соответствует 1 Сравнение соответствует для текущего события
TRGEVT_E_ CMP1	9	Запись Чтение Аппаратное влияние	0 Сравнение 1 на равенство не соответствует 1 Сравнение соответствует для текущего события
TRGEVT_E_ CMP2	10	Запись Чтение Аппаратное влияние	0 Сравнение 2 на равенство не соответствует 1 Сравнение соответствует для текущего события
–	12-11	0	Зарезервированы
EVENT_ SOURCE	15-13	Запись Чтение Аппаратное влияние	Источник сообщения о последнем отладочном событии: XX1 Внешний вывод приостановки DEXEVT X1X Выполняется команда DEBUG (DSEWT) 1XX Комбинация аппаратных источников событий DTREVT

Бит OCDS_P_SUSPEND управляет сигналом, останавливающим периферию. Если он установлен в «1», то соответствующая периферия приостанавливается. Этот бит устанавливается в ходе отладочного события в соответствии с битом PERIPHERALS_STOP в активном регистре управления отладочным событием. Этот бит должен быть переустановлен отладчиком. Если отладочный монитор прерывается пользовательской задачей с более высоким приоритетом, биты DEBUG_STATE и OCDS_P_SUSPEND, а также сигнал остановки периферии не изменяется.

Биты EVENT_SOURCE устанавливаются независимо от поля EVENT_ACTION в соответствующем регистре управления отладочным событием, за исключением значения 000_B. Эти биты должны быть переустановлены отладчиком. Для сравнения на равенство биты TRGEVT_E_CMPx устанавливаются только тогда, когда соответствующее сравнение разрешено (поле SELECT_E в регистре DTREVT). Эти биты должны быть сброшены отладчиком.

Регистр ID задачи DTIDR

Формат регистра DTIDR представлен на рисунке 7.8, функциональное назначение поля регистра DTIDR – в таблице 7.11.

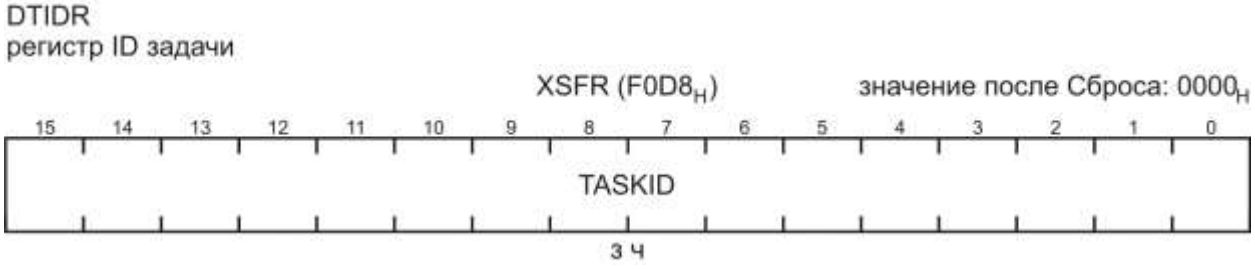


Рисунок 7.8 – Формат регистра DTIDR

Таблица 7.11 – Функциональное назначение поля регистра DTIDR

Поле	Биты	Тип	Описание
TASKID	15-0	Запись Чтение	Входные данные на аппаратное устройство генерации событий. Предназначено для использования в передовых системах реального времени для запоминания ID активной задачи

Регистры указания команд DIP и DIPX

Регистры DIP, см. рисунок 7.9 и таблицу 7.12, и DIPX, см. рисунок 7.10 и таблицу 7.13, предусмотрены для того, чтобы сделать видимым регистр указатель команд IP, когда CPU находится в режиме Halt.

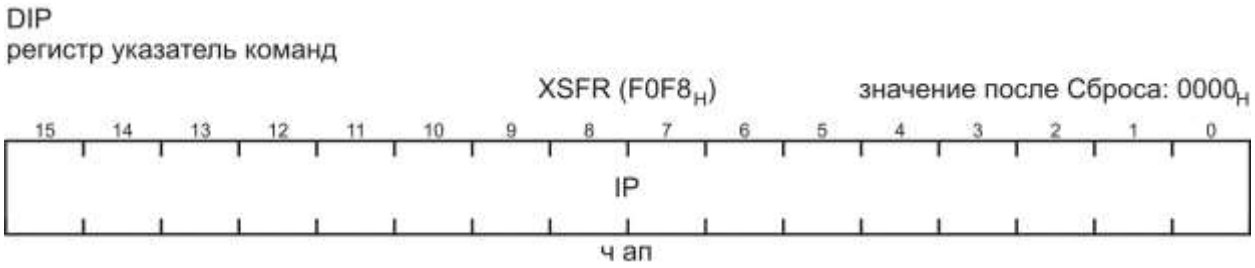


Рисунок 7.9 – Формат регистра DIP

Таблица 7.12 – Функциональное назначение поля регистра DIP

Поле	Биты	Тип	Описание
IP	15-0	Чтение Аппаратное влияние	Биты 15-0 текущего указателя команд в режиме Halt. Значение правильно только в режиме Halt

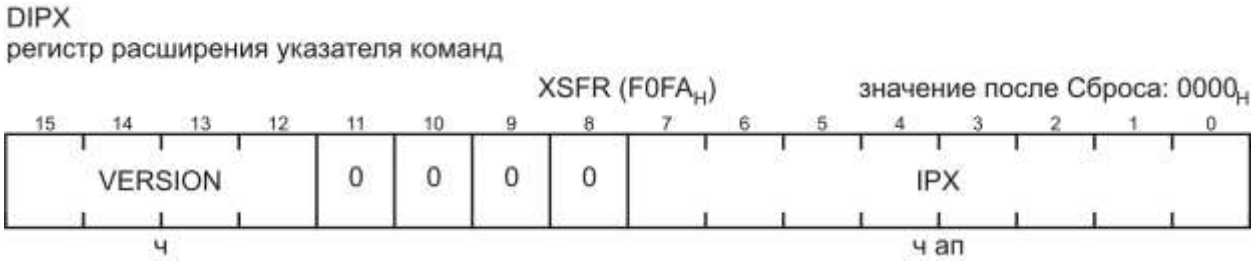


Рисунок 7.10 – Формат регистра DIPX

Таблица 7.13 – Функциональное назначение полей регистра DIPX

Поле	Биты	Тип	Описание
IPX	7-0	Чтение Аппаратное влияние	Биты 23-16 расширенной части текущего указателя команд (CSP) в режиме Halt
–		0	Зарезервировано
VERSION	15-12	Чтение	Аппаратно закодировано как 3

Регистры аппаратного устройства генерации сигнала запуска отладочного события

Регистры DCMP0, DCMP1, DCMP2, DCMPG, DCMPL, см. рисунок 7.11 и таблицу 7.14, используются аппаратным устройством генерации сигнала запуска отладочного события как ссылочные значения для сравнений. Они могут быть запрограммированы с помощью двух SFR регистров, а именно DCMPSP и DCMPDP. Поле SELECT_DCMP регистра DCMPSP, см. рисунок 7.12 и таблицу 7.15, выбирает сравниваемый регистр и записывает его старший байт. Младшие 16 бит могут быть доступны для записи с помощью регистра DCMPDP, см. рисунок 7.13 и таблицу 7.15.

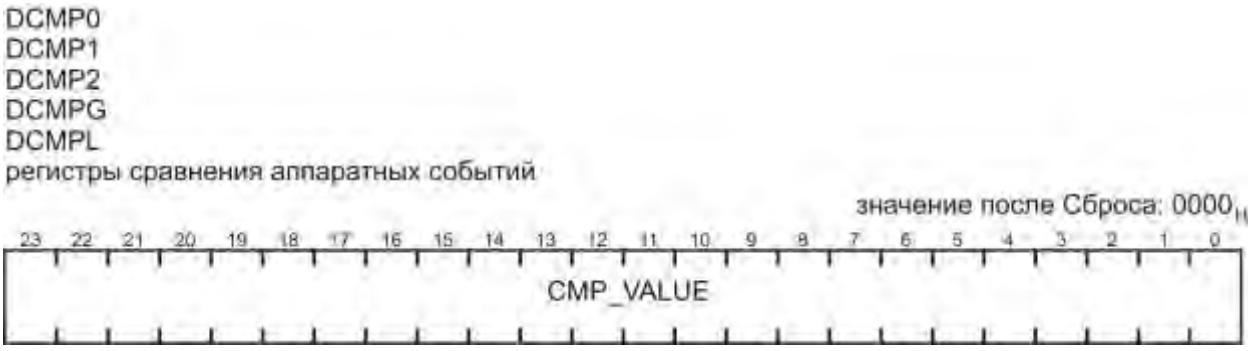


Рисунок 7.11 – Формат регистров сравнения аппаратных событий DCMP0, DCMP1, DCMP2, DCMPG и DCMPL

Таблица 7.14 – Функциональное назначение поля регистров сравнения DCMP0, DCMP1, DCMP2, DCMPG и DCMPL

Поле	Биты	Тип	Описание
CMP_VALUE	23-0	Запись Чтение	Сравниваемые значения для аппаратного устройства генерации сигнала отладочного события могут быть записаны только с помощью регистров DCMPSP и DCMPDP

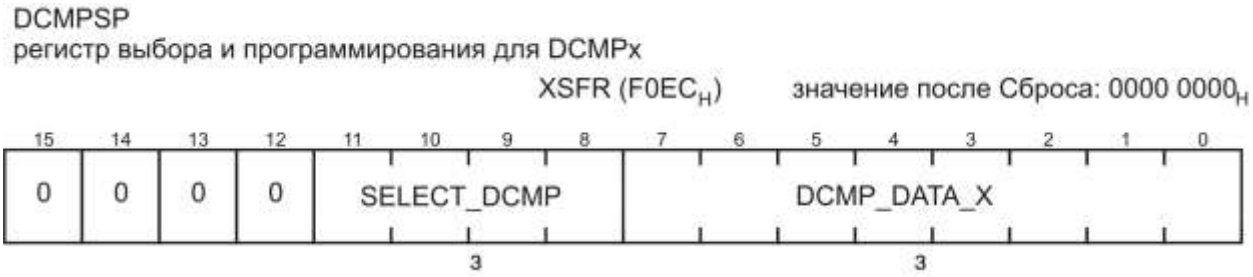


Рисунок 7.12 – Формат регистра DCMPSP

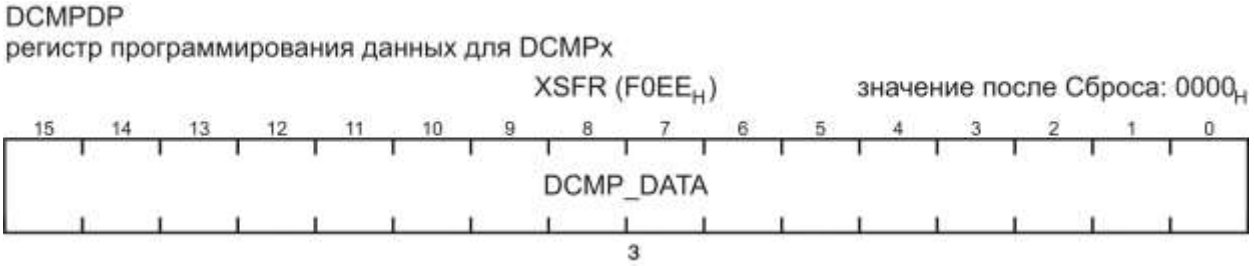


Рисунок 7.13 – Формат регистра DCMPDP

Таблица 7.15 – Функциональное назначение полей регистров DCMPSP и DCMPDP

Поле	Биты	Тип	Описание
Регистр DCMPSP			
DCMP_DATA_X	7-0	Запись	Устанавливает биты 23-16 выбранного (SELECT_DCMP) DCMP регистра
SELECT_DCMP	11-8	Запись	Выбор регистров сравнения: 0000 Выбор DCMP0 0001 Выбор DCMP1 0010 Выбор DCMP2 0011 Зарезервировано 0100 Выбор DCMPL 0101 Выбор DCMPG 1011–1111 Зарезервировано
–	15-12	0	Зарезервировано
Регистр DCMPDP			
DCMP_DATA	15-0	Запись	Устанавливает биты 15-0 выбранного SELECT_DCMP регистра DCMP

Общие положения при обращении к регистрам OCDS

Функциями OCDS в основном управляют записями в регистр состояния отладки DBGSR. Для корректного исполнения любого шага отладки необходимо соответствующее поле, установленное должным образом и в нужное время. Так как DBGSR имеет доступ по шине PDBUS, время, необходимое для нового значения, зависит от скорости шины. Это очень важно, так как скорость шины ниже скорости ядра, то есть скорости выполнения команд. Другое важное свойство ядра это то, что оно является конвейерной машиной с различными операциями чтения или записи, выполняемые на разных уровнях конвейера. Основная потенциальная проблема, которую надо учесть, это то, что значение регистра DBGSR (как и любого регистра SFR) не может быть эффективным сразу после его модификации. Задержка, в терминах команд ядра, выполняемых со старым значением DBGSR, имеет фиксированную часть (в большинстве случаев – одна команда) и переменную часть значения, зависящую от скорости PDBUS.

Имеется два самых критических момента для возможных конфликтов:

- задание значений и разрешение OCDS. Для правильной операции регистр DBGSR должен быть задан после захвата новых запрограммированных значений регистра DTREVT;

- выход из монитора. Все обновления регистра DBGSR должны стать эффективными до возвращения в пользовательскую программу. Иначе существует возможность, что точка останова breakpoint в программе будет достигнута прежде, чем регистр DBGSR захватит новые значения. Это может вызвать множество проблем, таких как вызов монитора после выполнения breakpoint или непосредственное перешагивание через breakpoint, вместо выполнения остановки break.

Общие приемы для избежания проблем программного обеспечения с OCDS

Принципиальное решение избежать проблемы при обращении к регистрам OCDS состоит в том, чтобы удостовериться после команд, записывающих новые значения в регистры, в выполнении команд с новыми значениями, когда эти значения действительно эффективны.

Использование некритических команд

После записи в управляющий регистр OCDS (DBGSR) далее следуют команды, выполнение которых не зависит от новых параметров настройки:

In : Запись в DBGSR;

In+1 : Некритическая команда, DBGSR все еще содержит прежнее значение;

…

In+d : Любая команда, DBGSR уже содержит новое значение.

Самый простой способ гарантировать некоторое время для установки состоит во вставке команд NOP (нет операции) перед очередной критической командой:

```
extr    #1                      ; область адресов ESFR
mov     DTREVT, #02200H  ; SELECT_E=01B, MUX_E=10B
@repeat (10)
nop                             ; фиктивная петля в 10 NOP
@endr
extr    #1                      ; новое значение DTREVT уже эффективно
mov     DBGSR, #00001H      ; разрешение OCDS!
```

Трудность здесь состоит в том, чтобы оценить достаточное ли время для законченной и эффективной записи, еще более изменяемое скоростью программирования шины PDBUS. Пример, приведенный выше, справедлив для соотношения f(PD)/f(CPU)=1/8, для меньшей пропорции необходимо больше команд NOP.

Операции чтения после записи

Немедленно после записи в OCDS может следовать операция чтения (пустышка) с того же адреса:

```
extr    #2                      ; область адресов ESFR
mov     DBGSR, #00005H      ; разрешение OCDS, программный отладочный режим
                                ; выполнение одной команды после RETI
mov     R7, DBGSR               ; новое значение в DBGSR эффективно
reti                            ; выход из монитора
```

Таким образом, гарантируется новое значение регистра DBGSR при продолжении уже следующей команды. Более того, в этом случае нет зависимости от скорости PDBUS, потому что центральное процессорное устройство обеспечивает выполнение операции записи, которая будет закончена, прежде чем начнется чтение по тому же адресу. Таким образом, фактически это самый легкий способ гарантировать правильность работы OCDS.

Поведение при сбросе

Если OCDS запрещен (обычно, когда блок JTAG находится в состоянии сброса), все его регистры сбрасываются при каждом сбросе CPU (RESET), в другом случае не сбрасываются. Это поведение позволяет определять сброс в случае, когда нет подключенного отладчика, или когда отладчик управляет блоком OCDS косвенно, с помощью монитора. В другом случае, когда отладчик управляет блоком OCDS напрямую, регистры OCDS не затрагиваются пользовательской программой или сбросом системного окружения. Это позволяет хорошо отлаживать очень недружелюбные системы.

7.2 Блок JTAG

Блок JTAG является мостом между выводами JTAG и блоком CERBERUS. Блок JTAG не является частью субсистемы. Порт JTAG является специальным интерфейсом, стандартизованным для периферийного сканирования. Дополнительно он может быть использован для внутреннего тестирования микросхемы, а также для программирования внутренней Flash-памяти. Поскольку эти приложения не используются во время нормальных операций, порт JTAG является интерфейсом для специальных пользовательских режимов. Функциональность основана на стандарте IEEE 1149 JTAG Standard. Блок имеет 8-битный регистр команд JTAG. На рисунке 7.14 изображено межсоединение блоков JTAG, CERBERUS, порта JTAG.

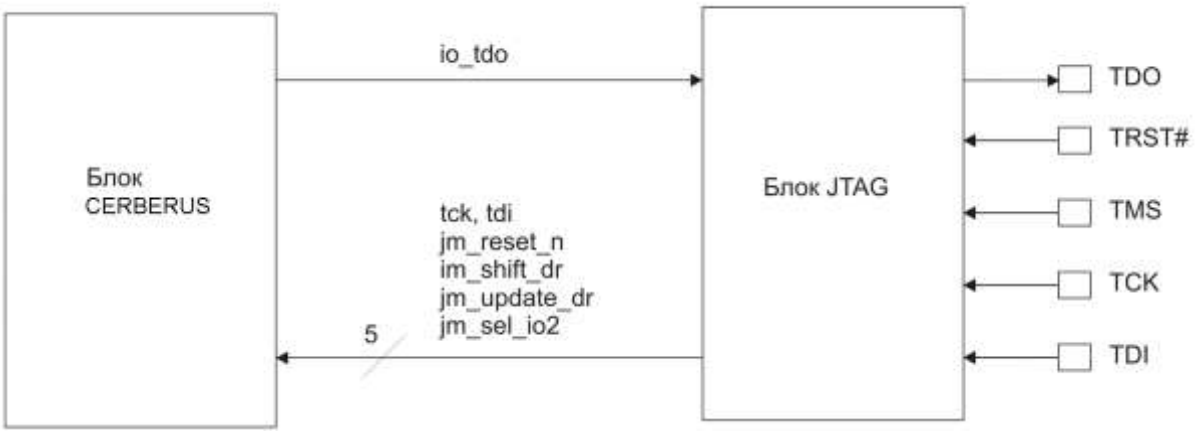


Рисунок 7.14 – Соединение блоков JTAG, CERBERUS, порта JTAG

На рисунке 7.15 показана структурная схема блока JTAG.

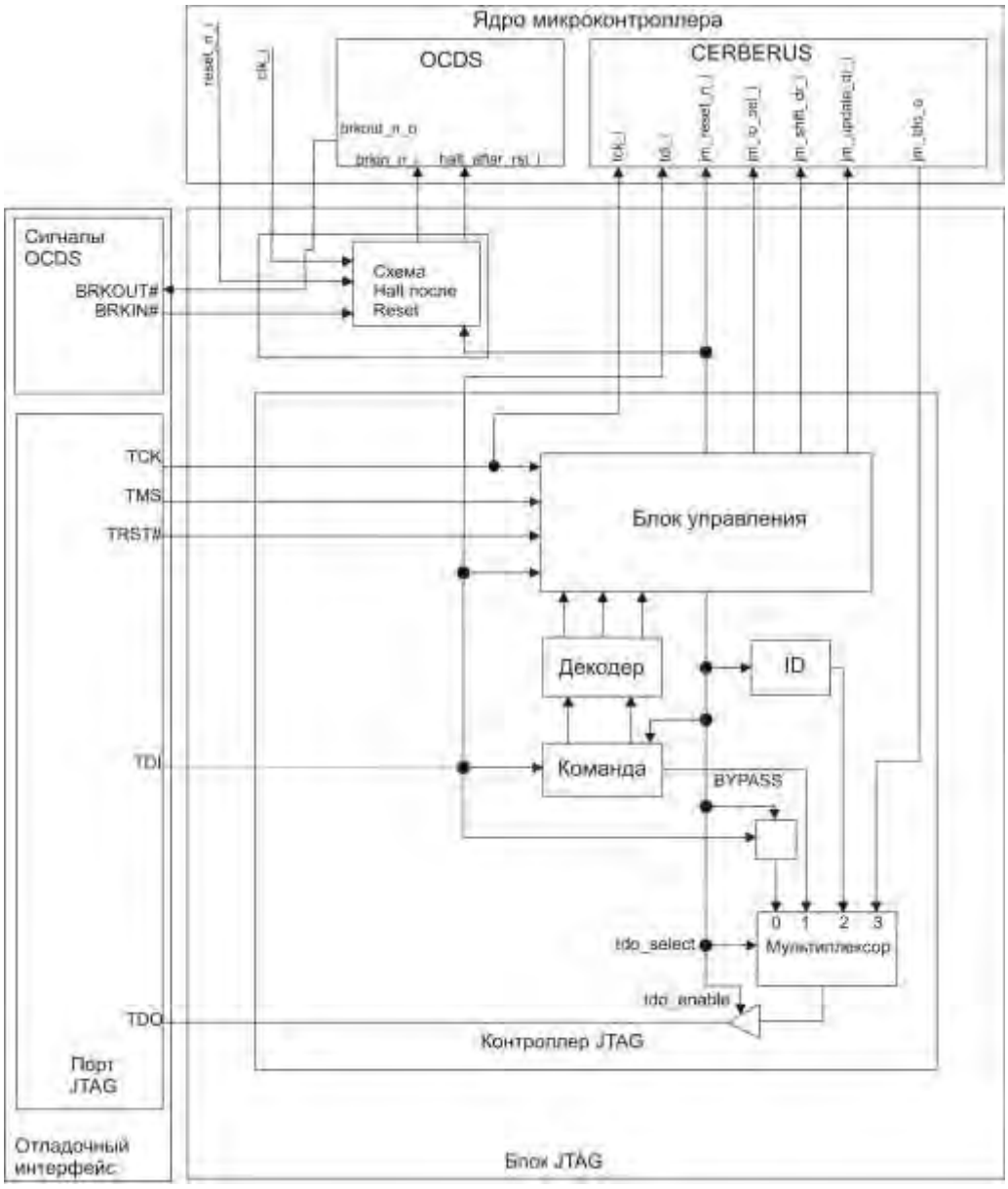


Рисунок 7.15 – Структурная схема блока JTAG

Схема состояний контроллера JTAG

Схема состояний контроллера JTAG (JTAG Controller State Machine IEEE.1149) является сердцем блока JTAG. Это так же относится к контроллеру порта тестового доступа TAP. Все переключения состояний происходят по положительному фронту, управляемые выводом TMS. После сброса TRST# схема состояний переходит в состояние сброса тестовой логики (reset testlogic state). При низком уровне сигнала на выводе TMS и положительном фронте сигнала на выводе TCK схема переводится в тестовое состояние холостого хода (run test/idle). Все последующие переключения происходят по тому же принципу.

Схема состояний контроллера JTAG имеет два параллельных управляющих пути. Один путь предназначен для регистра команд JTAG, находящегося в блоке JTAG (INSTRUCTION или IR). Другой путь предназначен для сканирования регистра данных DR. Регистр команд IR выбирает последовательность сканирования для последующих просмотров данных. Схема сканирования JTAG позволяет регистрам просмотра произвольной длины быть захваченными и обновленными. На рисунке 7.16 приведена схема состояний контроллера JTAG.

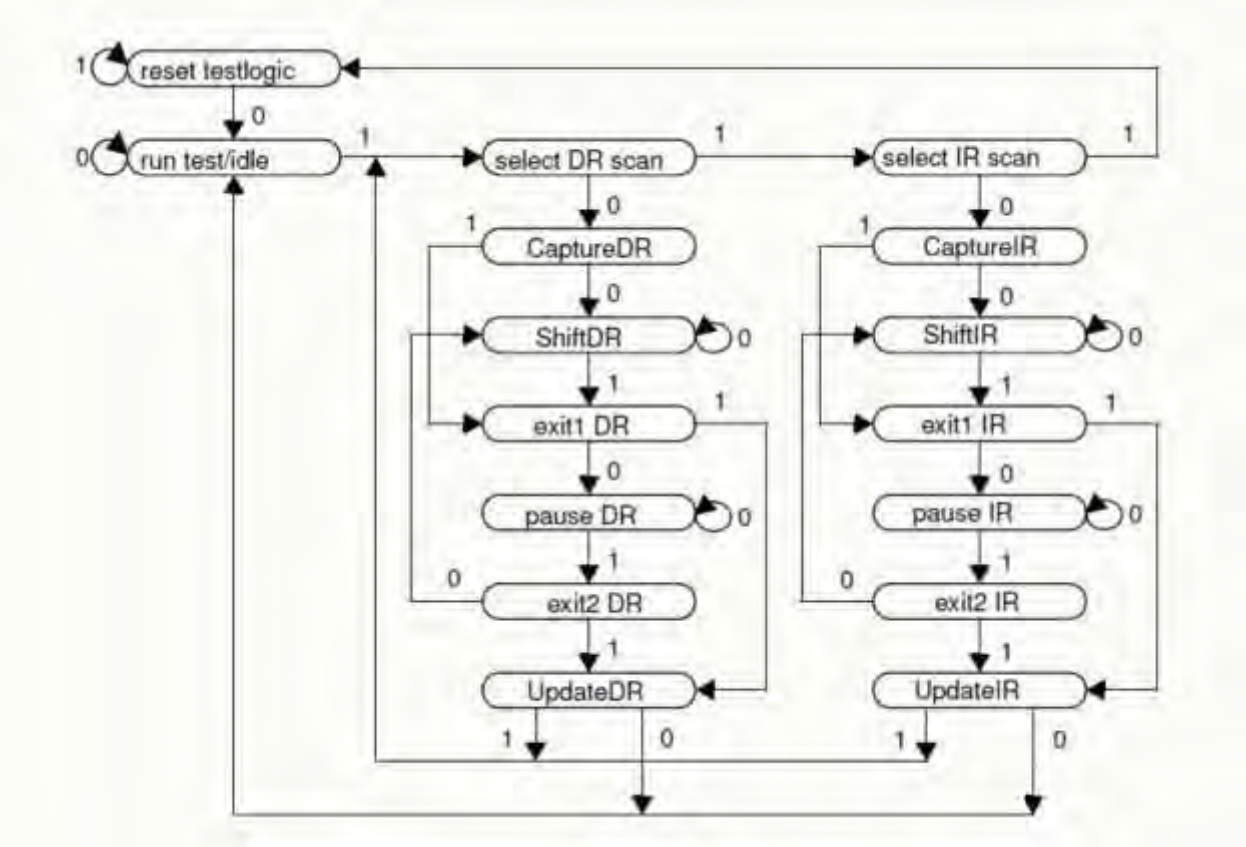


Рисунок 7.16 – Схема состояний контроллера JTAG

Все сигналы, приведенные на рисунке 7.15, ведут себя как описано стандартом IEEE.1149: выходные, входные, внутренние сигналы, их переключение относительно фронта сигнала TCK.

Сигналы osds_brk_n_i и ocds_brkout_n_o функционально такие же, как BRKIN# и BRKOUT#. Сигнал ocds_halt_after_rst_i активным состоянием запускает режим «Останов» Halt микроконтроллера. Ввести этот режим возможно, когда микроконтроллер находится в состоянии сброса RESET. Поэтому этот режим называется «Останов после сброса». В этом случае отладчик может управлять микроконтроллером до начала выполнения любой программы. Для активации этого режима необходимо:

- активировать сигнал BRKIN#, когда еще активен RESET#;
- выполнить процедуру сброса для ввода в режим Halt;

- выполнить запросы доступа и отладочные процедуры через блок CERBERUS;
- дезактивировать сигнал BRKIN# для старта микроконтроллера.

Сигналы tck_i, tdi_i функционально такие же, как сигналы TCK и TDI в порту JTAG. Сигнал im_reset_n_i активен в течение состояния «reset testlogic». Схема, приведенная на рисунке 7.15, приходит в это состояние при включении с активным входом TRST# или из любого текущего состояния не более чем через пять циклов сигнала TCK, при удержании высокого уровня TMS. Следующие сигналы должны быть активированы только после загрузки регистром команд IR кода команды SAMPLE/RELOAD во время состояний «IR scan». Сигнал jm_io_sel_i должен быть активен в течение всех состояний «DR scan». В то же время сигнал jm_tdo_o может быть выбранным и разрешенным для управления выходом TDO контроллера JTAG. Сигналы jm_shift_dr_i и jm_update_dr_i, показанные на рисунке 7.15, активируются в течение состояний «ShiftDR» и «UpdateDR», соответственно.

Команды модуля JTAG

Все возможные команды модуля JTAG, передаваемые в регистр команд в течение состояний «IR scan», приведены в таблице 7.16.

Таблица 7.16 – Команды модуля JTAG в течение состояний «IR scan»

Код операции	Диапазон	Тип	Команда
00000000_B-00001111_B	00_H-$0F_H$, 16 команд	Зарезервировано	–
00010000_B	10_H	Конфигурация схемы	CCONF_SET
00010001_B-10111111_B	11_H-BF_H, 174 команды	Зарезервировано	–
11000000_B	$C0_H$	Режим чтения-записи JTAG	JTAG_IO_SELECT_PATH
11000001_B	$C1_H$		JTAG_IO_INSTRUCTION1
11000010_B-11111110_B	$C2_H$-FE_H, 60 команд	Зарезервировано	–
11111111_B	FF_H	IEEE1149	BYPASS

Регистры блока JTAG

Блок JTAG содержит стандартные регистры INSTRUCTION (IR) и BYPASS, а также два специальных регистра CCONF и IOPATH.

В таблице 7.17 приведено краткое описание регистров блока JTAG.

Таблица 7.17 – Описание регистров блока JTAG

Название регистра	Размер, бит	RESET TRST#	Описание
BYPASS	1	X	Стандартный регистр обхода (bypass) JTAG. Если выбрана команда BYPASS, то сигнал на TDO равен сигналу на TDI, задержанному на один цикл сигнала TCK
CCONF	16	0000_H	Регистр конфигурации кристалла
ID	32	–	Дополнительный регистр ID стандарта JTAG. Содержание ID выдается наружу, когда поле INSTRUCTION содержит команду IDCODE
INSTRUCTION (IR)	8	04_H	Регистр команд стандарта JTAG В отличие от остальных регистров, он устанавливается во время состояния «IR scan»
IOPATH	2	00_B	Выбор блока CERBERUS

Регистр BYPASS

Это обязательный JTAG регистр. Если происходит выбор, то сигнал на выходе TDO равен сигналу на входе TDI, задержанный на один цикл сигнала TCK.

Регистр ID

Регистр ID не является частью блока JTAG. Его применение имеет специальное назначение. Он позволяет обслуживать версию и часть регистров, которые имеют доступ через CPU как регистры SFR или через порт JTAG по команде IDCODE. В соответствии со стандартом JTAG команда IDCODE должна иметь структуру, показанную на рисунке 7.17 и поля, описанные в таблице 7.18.

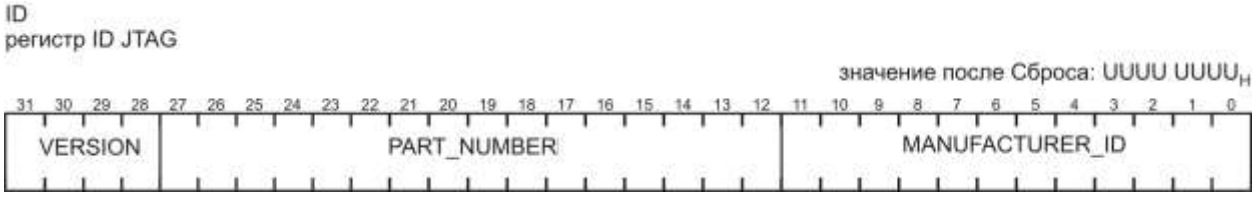


Рисунок 7.17 – Структура команды IDCODE

Таблица 7.18 – Поля команды IDCODE

Название поля	Биты	Тип	Описание
VERSION	31-28	Чтение	Версия кристалла
PART_NUMBER	27-12	Чтение	Частичный номер
MANUFACTURER_ID	11-0	Чтение	ID производителя в соответствии с JEDEC стандартом идентификационных кодов производителей JEP-106-G

Регистр IOPATH

Регистр IOPATH является модификацией регистра сканирования JTAG для обеспечения защиты от ошибок. Для IOPATH сигнал TDO является входным сигналом, как показано на рисунке 7.18, а не выходным. Это позволяет определять передачу бита ошибки. Поведение TDI/TDO такое же, как выполнение команды BYPASS, за исключением того, что первый выходной бит «1», не «0», как при BYPASS. Это различие

важно в случае ошибочного бита, когда команда JTAG вдвигается в порт. В наиболее вероятном случае ошибочная команда не будет выполнена, блок JTAG установит режим BYPASS, который нельзя иначе отличить от команды JTAG_IO_SELECT_PATH.

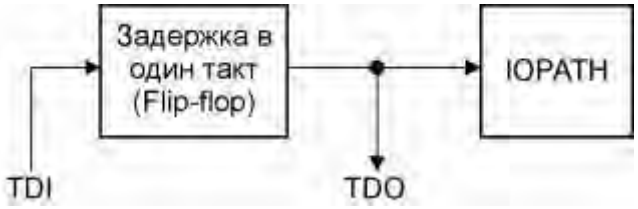


Рисунок 7.18 – Регистр IOPATH

Регистр IOPATH используется для выбора блока CERBERUS. Если команда JTAG находится в диапазоне адресов и не равна C0H, то соответствующий сигнал выбора активен. Регистр IOPATH имеет ширину 2 бита и установлен командой JTAG_IO_SELECT_PATH как регистр регулярного сканирования. Для выбора CERBERUS он должен быть установлен как 10_B (это рекомендуется устанавливать по умолчанию для аппаратных средств).

Регистр CCONF

Регистр CCONF предусмотрен для конфигурирования специальных состояний микроконтроллера. Это можно рассматривать как альтернативный механизм перезагрузки конфигурации. Все конфигурационные биты имеют соответствующие биты защиты. Это позволяет различным инструментам, имеющим интерфейс JTAG, иметь прямой доступ к специализированным битам. В конкретном случае регистр CCONF позволяет конфигурировать контроллер таким образом, что после системного сброса он не пытается считывать команды из памяти, а переходит в режим ожидания. Кроме того, регистр CCONF имеет бит защиты, который, будучи установленным, не позволяет изменять текущую конфигурацию. Регистр CCONF, см. рисунок 7.19 и таблицу 7.19, выбирается подачей команды CCONF_SET и ведет себя так же, как и регистр IOPATH.

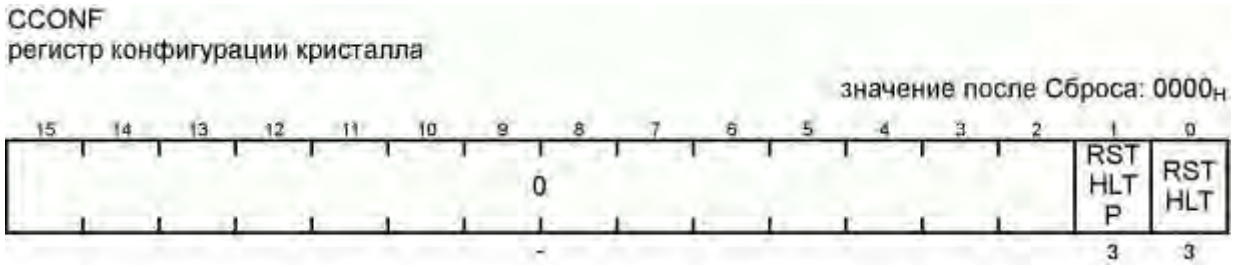


Рисунок 7.19 – Формат регистра CCONF

Таблица 7.19 – Функциональное назначение полей регистра CCONF

Поле	Биты	Тип	Описание
RST_HLT	0	Запись	Halt после сброса: 0 Нет действия. 1 Режим Halt после сброса RESET.
RST_HLT_P	1	Запись	0 Бит защиты: RST_HLT не меняется. 1 RST_HLT можно изменить.
–	15-2	0	Зарезервированы.

Инициализация блока CERBERUS

Используемые в микроконтроллере коды команд, приведенные в таблице 7.20, заносятся в регистр INSTRUCTION, когда JTAG модуль находится в состоянии «IR scan».

Таблица 7.20 – Команды модуля JTAG

Код команды	Команда	Описание
10_H	CCONF_SET	Выбирает регистр CCONF
$C0_H$	JTAG_IO_SELECT_PATH	Выбирает регистр IOPATH
$C1_H$	JTAG_IO_INSTRUCTION1	Выбирает модуль CERBERUS
FF_H	BYPASS	Выбирает регистр BYPASS

Шаги для инициализации:

1 Сброс JTAG. На вывод TRST# кратковременно подается низкий уровень сигнала.

2 Загрузка регистра CCONF. «IR scan»: загрузка команды CCONF_SET (10_H). DR scan: загрузка 0003_H в регистр CCONF для останова после сброса, иначе 0000_H.

На вход TDI надо послать 16 бит младшим битом вперед.

3 Загрузка регистра IOPATH. «IR scan»: загрузка команды JTAG_IO_SELECT_PATH ($C0_H$). DR scan: загрузка 10_B в регистр IOPATH.

Из-за задержки в один такт на вход TDI надо послать 3 бита младшим битом вперед.

4 Выбор модуля CERBERUS. «IR scan»: загрузка команды JTAG_IO_INSTRUCTION1 ($C1_H$). После выполнения этих операций блок CERBERUS готов к работе.

7.3 Блок CERBERUS

Блок CERBERUS является универсальным средством для подключения интерфейса JTAG. Ядро блока содержит сдвигатель ядра JTAG. Сдвигатель управляется сигналами JTAG, поэтому является асинхронным к другим частям блока CERBERUS. Структурная схема блока CERBERUS приведена на рисунке 7.20.

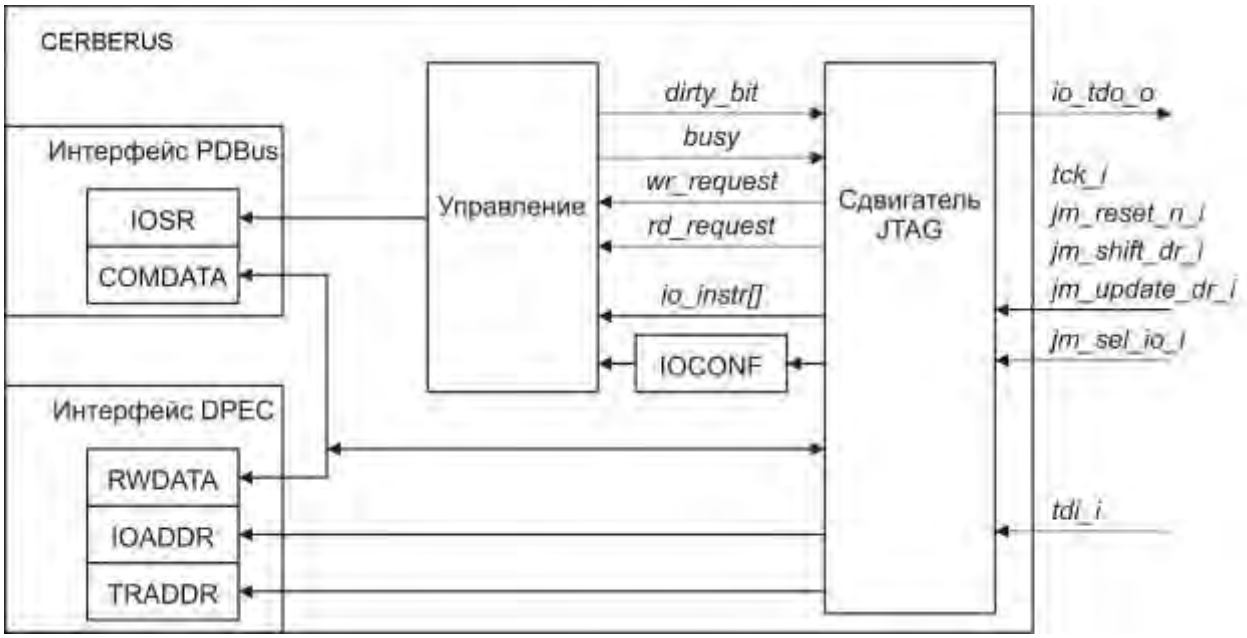


Рисунок 7.20 – Структурная схема блока CERBERUS

Определение режимов блока CERBERUS

Блок CERBERUS может использоваться в двух различных целях. Первый состоит в чтении и записи ячейки памяти (режим чтения-записи RW Mode), второй состоит в обмене данными с программой monitor, запущенной CPU (режим коммуникации Communication Mode).

Защита от ошибок error correction

Стандарт JTAG не включает в себя какую-либо защиту для последовательной передачи по выводам TDI, TDO и управления (вывод TMS). Однако есть способы включить защиту от ошибки, не уходя слишком далеко от возможностей структуры JTAG. Защита от ошибки для входных данных по выводу TDI может быть достигнута задержкой выходных данных на один цикл TCK на выводе TDO. Выходные данные могут быть перемещены дважды (многократно) и затем сравнены для максимальной защиты от ошибок. Блок CERBERUS считается занятым (busy), если запрошенные операции чтения или записи не завершены.

Все данные и адреса вводятся и выводятся, начиная с младшего бита. Внешнее отладочное устройство является главным (master) при всех приемопередачах, инициализируя передачи для обоих направлений.

Синтаксис последовательного потока битов для выводов TDI, TDO

Когда выбран блок CERBERUS, он контролируется через вывод TDI потоком битов с помощью последовательности JTAG состояний CaptureDR, ShiftDR и UpdateDR. Первые вводимые четыре бита – есть команда чтения-записи. Следующие биты (биты занятости) игнорируются, пока не появится стартовый бит на выводе TDO. Биты занятости busy bits имеют место для всех команд чтения-записи за исключением IO_CONFIG, когда предыдущая операция еще не окончилась.

Если команда по типу записывающая, то следующий TDI бит, после параллельного стартового TDO бита, используется, как первый бит данных. Далее следует передача данных до конца. На рисунке 7.21 приведен синтаксис битового потока для выводов TDI и TDO в состоянии ShiftDR.

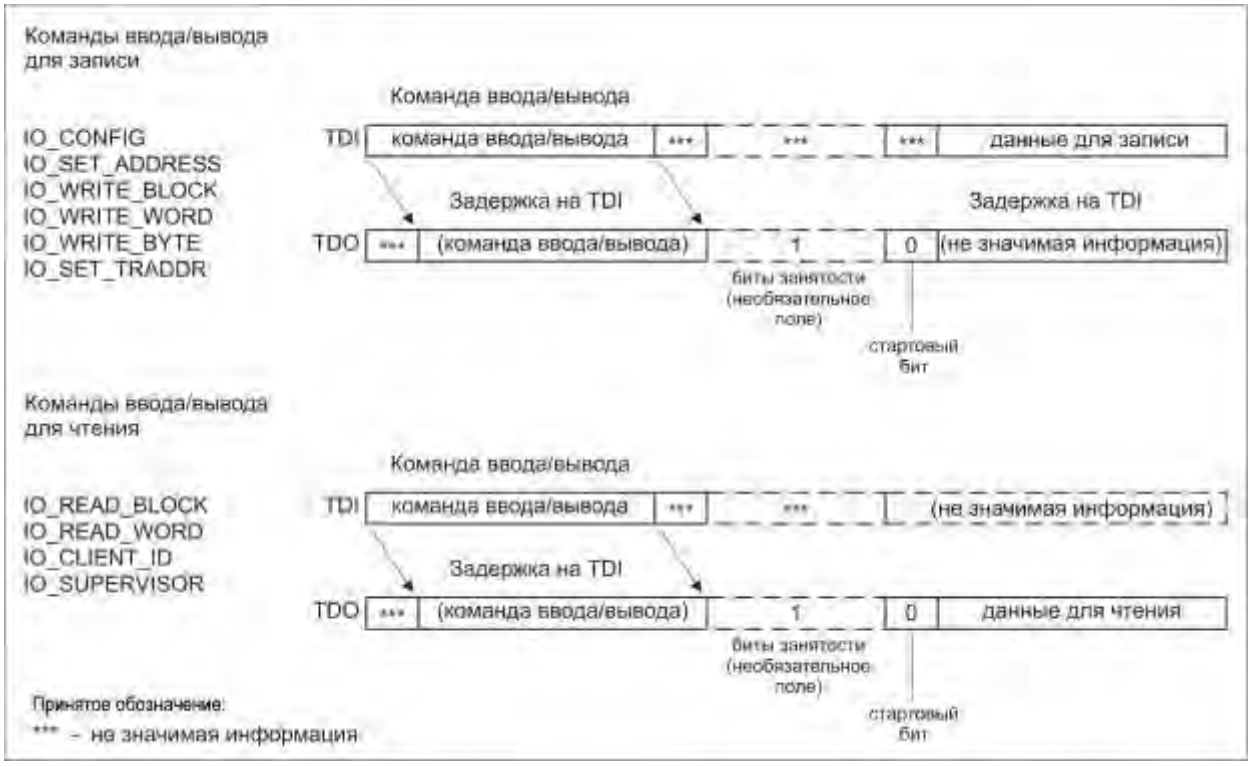


Рисунок 7.21 – Синтаксис последовательной передачи для выводов TDI и TDO в состоянии ShiftDR

Если команда по типу читающая, то все TDI биты после команды игнорируются. После стартового бита на TDO читаемые биты начинают выдаваться. Если команда не определена или невыполняема, выдается неопределенное число битов занятости.

Команды чтения-записи блока CERBERUS

В таблице 7.21 приведены команды чтения-записи блока CERBERUS. В отличие от команд блока JTAG они не передаются в регистр команд JTAG во время «IR scan», а первые четыре бита во время «DR scan» передаются в сдвиговый регистр блока CERBERUS.

Таблица 7.21 – Команды чтения-записи блока CERBERUS

Команда	Код	Тип	Описание
IO_CONFIG	0_H	Запись	Устанавливает регистр конфигурации IOCONF.
IO_SET_ADDRESS	1_H	Запись	Устанавливает регистр адреса IOADDR.
IO_WRITE_BLOCK	2_H	Запись	Старт записи блока данных, начиная с адреса в IOADDR.
IO_READ_BLOCK	3_H	Чтение	Старт чтения блока данных, начиная с адреса в IOADDR.
IO_WRITE_WORD	4_H	Запись	Режим чтения-записи: запись слова. Режим коммуникации: передача слова.
IO_READ_WORD	5_H	Чтение	Режим чтения-записи: чтение слова. Режим коммуникации: запрос слова.
Зарезервировано	7_H-6_H		
IO_WRITE_BYTE	8_H	Запись	Режим чтения-записи: запись байта. Режим коммуникации: зарезервировано.
Зарезервировано	9_H	–	–
IO_SET_TRADDR	A_H	Запись	Установка регистра TRADDR.
IO_SUPERVISOR	B_H	Чтение	Подтверждение сброса и анализ состояния захвата шины.
Зарезервировано	E_H-C_H	–	–
IO_CLIENT_ID	F_H	Чтение	Чтение ID клиента.

Команда IO_CONFIG используется для прерывания операций записи в режиме чтения-записи RW и для конфигурирования блока CERBERUS с помощью регистра IOCONF. Команда IO_CONFIG никогда не вырабатывает биты занятости. Необходимо отметить, что в случае активизации команды IO_CONFIG прерывается последняя операция записи режима чтения-записи (программный сброс).

Команда IO_SET_ADDRESS устанавливает адрес IOADDR для следующего доступа к режиму чтения-записи.

Команда IO_READ_WORD используется для чтения данных в режиме чтения-записи или для приема данных в режиме коммуникации. Команда IO_READ_BLOCK используется только в режиме чтения-записи. Единственное отличие от команды IO_READ_WORD состоит в том, что происходит инкрементация адреса слова. Команды чтения могут быть прерваны, когда внешнее устройство устанавливает состояние UpdateDR. Для команды IO_READ_WORD в режиме коммуникации должно произойти, по крайней мере, четыре цикла сдвига после вывода стартового бита для подтверждения чтения. Это предупреждает потерю прочитанных слов данных.

Команда IO_WRITE_WORD используется для записи данных в режиме чтения-записи или для передачи данных в режиме коммуникации. Команда IO_WRITE_BLOCK используется только в режиме чтения-записи. Единственное отличие от команды IO_WRITE_WORD состоит в том, что происходит инкрементация адреса слова. Для всех

команд записи (так же и для IO_WRITE_BYTE) должно произойти, по крайней мере, четыре цикла после вывода стартового бита для записи, что фактически необходимо в состоянии UpdateDR. Это позволяет успешно проверять последнюю запись (стартовый бит), не начиная новую запись.

Команда IO_WRITE_BYTE является особым случаем команды IO_WRITE_WORD для записи байтов. Для IO_WRITE_BYTE необходимо вводить полное 16-битное слово, младший байт которого всегда записывается (для четных и нечетных адресов).

Команда IO_SET_TRADDR устанавливает регистр TRADDR, который используется для трассировки (просмотра) адресов по внешней шине.

Команда IO_SUPERVISOR используется для вывода режимов чтения-записи и коммуникации из ошибочного состояния (error state). Эта инструкция выводит регистр IOINFO после стартового бита.

Команда IO_CLIENT_ID выдает клиентский специфицированный ID код из регистра CLIENT_ID.

Поведение сдвигового регистра

На рисунке 7.22 показана взаимосвязь выводов TDI, TDO и содержания сдвигового регистра блока CERBERUS после ввода клиентской команды. Мультиплексор MUX1 управляется активной командой, а мультиплексор MUX2 управляется состоянием клиента (занято или операция завершена). В случае команды (чтения-записи) типа записи, после появления на TDO стартового бита задержанные данные вдвигаются в сдвиговый регистр и параллельно попадают на вывод TDO. В случае команды (чтения-записи) типа чтения захваченные данные выдвигаются через MUX1 и MUX2. Сдвиговый регистр формирует циклический буфер, который может быть использован для двойного сдвига с целью защиты от ошибок (error protection).

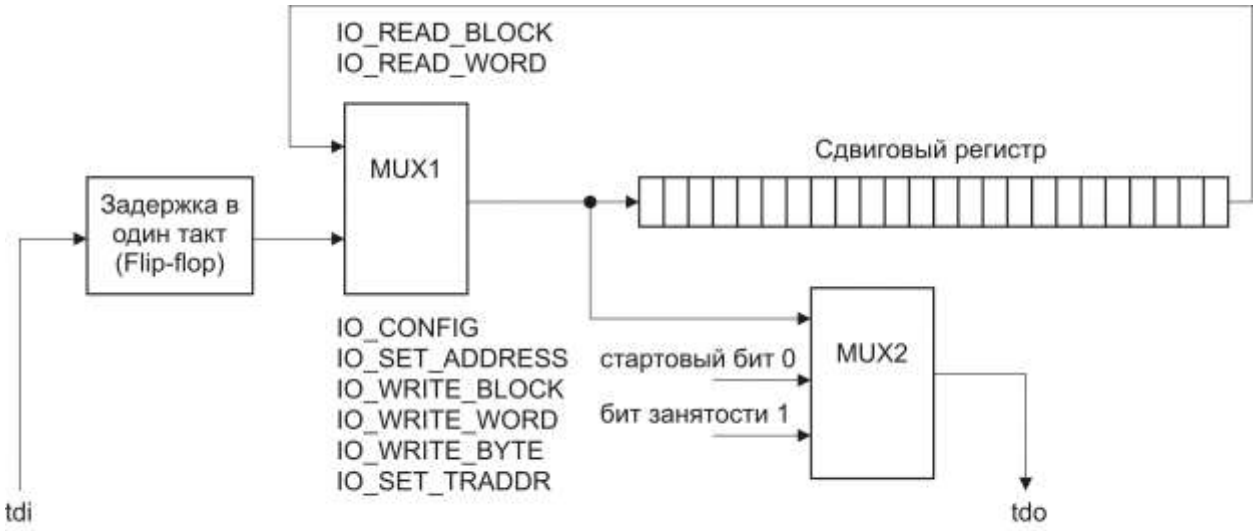


Рисунок 7.22 – Сдвиговый регистр в состоянии ShiftDR

Примеры передачи данных

Ниже описано поведение интерфейса ввода-вывода порта JTAG для команды IO_CONFIG. В этом примере последовательность битов на выводах TDI и TDO показана только в состоянии ShiftDR.

IO_CONFIG
IOCONF 01_H RWDATA 0000_H IOADDR 000000_H

TDI:	**0**	**0**	**0**	**0**	0	0	1	0	0	0	0	0	0	0
TDO:	0	**0**	**0**	**0**	**0**	0	0	1	0	0	0	0	0	0

В следующем примере показаны два случая поведения интерфейса чтения-записи порта JTAG для команды IO_SET_ADDRESS. В первом случае нет битов занятости (busy bit), и первый адресный бит сдвинут параллельно стартовому биту. Во втором случае есть четыре бита занятости, и внешний интерфейс начинает вводить в адрес один цикл после стартового бита. Результат обоих случаев одинаков.

1. IO_SET_ADDRESS
IOCONF 01_H RWDATA 0000_H IOADDR 000033_H
TDI: **1 0 0 0** 1 1 **1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**
TDO:0 **1 0 0 0** 0 1 **1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0**

2. IO_SET_ADDRESS
IOCONF 01_H RWDATA 0000_H IOADDR 000033_H
TDI: **1 0 0 0** 1 1 1 1 1 1 **1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0**
TDO:0 **1 0 0 0** 1 1 1 1 0 1 **1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0**

В примере, приведенном ниже, показано поведение интерфейса ввода-вывода порта JTAG для команды IO_WRITE_WORD. Имеется один бит занятости, и первый бит данных сдвигается параллельно стартовому биту. Необходимо отметить, что поведение на выводах TDI и TDO такое же, как для JTAG команды BYPASS. Чтобы избежать этого при всех обстоятельствах, внешний интерфейс должен вводить единичный бит после команды ввода-вывода до тех пор, пока не появится стартовый бит на TDO.

IO_WRITE_WORD
IOCONF 01_H RWDATA 1234_H IOADDR 000033_H
TDI: **0 0 1 0** 1 0 0 **0 1 0 1 1 0 0 0 1 0 0 1 0 0 0** 0
TDO:0 **0 0 1 0** 1 0 0 **0 1 0 1 1 0 0 0 1 0 0 1 0 0 0**

В следующем примере показано поведение интерфейса ввода-вывода JTAG для команды IO_READ_WORD. Здесь имеется три бита занятости с последующим стартовым битом. Следующие биты на TDO являются битами данных. Во втором случае читаемые данные выдвигаются (выдаются) дважды, включая старший неиспользуемый байт, для защиты от ошибок (error protection).

1. IO_RAD_WORD
IOCONF 01_H RWDATA 1234_H IOADDR 000033_H
TDI: **1 0 1 0** 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
TDO:0 **1 0 1 0** 1 1 1 0 **0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0**

2. IO_READ_WORD
IOCONF 01_H RWDATA 1234_H IOADDR 000033_H
TDI: **1010**11111111111111111111111111111111111111111111111111111111
TDO: 0**1010**1110**0010110001001000**xxxxxxxx**0010110001001000**xxxxxxxx

Регистры блока CERBERUS

В таблице 7.22 приведены все регистры блока CERBERUS.

Таблица 7.22 – Регистры блока CERBERUS

Регистр	Ширина	Адрес	Описание
1	2	3	4
CLIENT_ID	16	–	ID код клиента

Окончание таблицы 7.22

1	2	3	4
IOADDR	24	–	Адрес для доступа к следующему режиму чтения-записи
IOCONF	8	–	Регистр конфигурации
IOINFO	16	–	Регистр анализа состояния микроконтроллера
TRADDR	4	–	Адрес в режиме трассировки внешней шины
COMDATA	16	$F068_H$	Регистр данных режима коммуникации
RWDATA	16	$F06A_H$	Регистр данных режима чтения-записи
IOSR	16	$F06C_H$	Регистр состояния

Регистр CLIENT_ID

Регистр CLIENT_ID, см. рисунок 7.23 и таблицу 7.23, позволяет внешнему отладочному устройству проверять аппаратные средства в автоконфигурационном режиме.

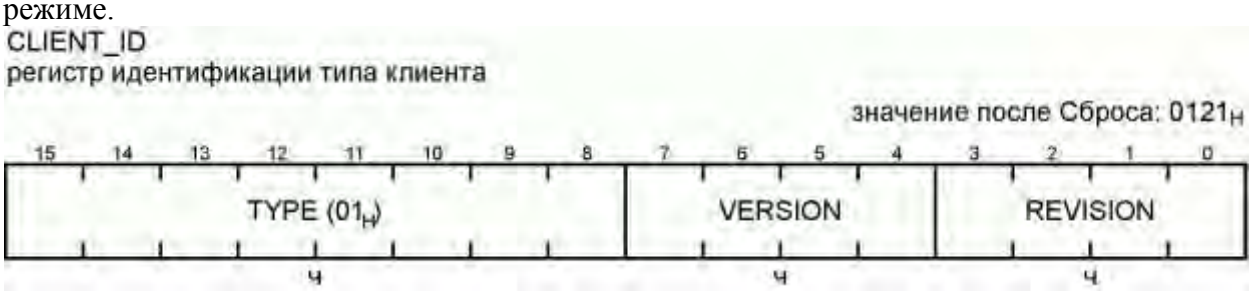


Рисунок 7.23 – Формат регистра CLIENT_ID

Таблица 7.23 – Функциональное назначение полей регистра CLIENT_ID

Поле	Биты	Тип	Описание
REVISION	3-0	Чтение	Версия кристалла
VERSION	7-4	Чтение	Версия блока CERBERUS
TYPE	15-8	Чтение	Клиентский тип

Регистр IOADDR

Регистр IOADDR захватывает 24-битный адрес для следующего обращения к блоку CERBERUS. Регистр IOADDR обновляется в состоянии UpdateDR содержимым сдвигового регистра, когда команда IO_SET_ADDRESS активна, или увеличивается на два (16-битное слово), если были выполнены команды IO_READ_BLOCK или IO_WRITE_BLOCK.

Регистр IOCONF

Регистр IOCONF используется для конфигурации блока CERBERUS. Регистр IOCONF, см. рисунок 7.24 и таблицу 7.24, записывается со стороны внешнего интерфейса и не доступен со стороны CPU.

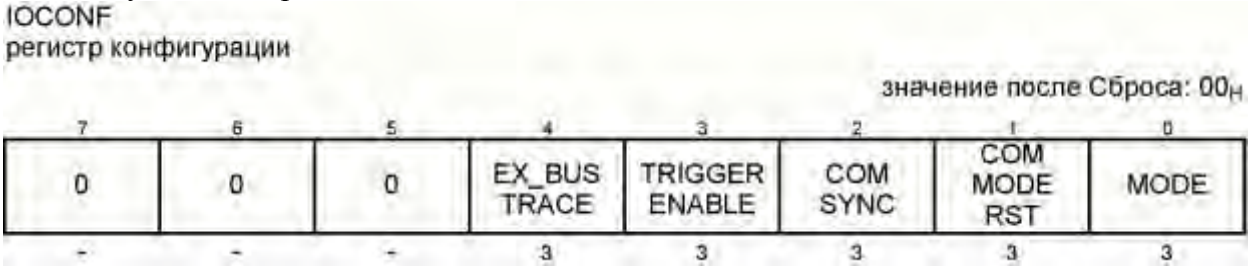


Рисунок 7.24 – Формат регистра IOCONF

Таблица 7.24 – Функциональное назначение полей регистра IOCONF

Поле	Биты	Тип	Описание
MODE	0	Запись	Если 0, режим коммуникации, иначе режим чтения-записи
COM_MODE_ RST	1	Запись	Если 1, CRCSYNC и CWSYNC сбрасываются в регистре IOSR
COM_SYNC	2	Запись	Устанавливается бит COM_SYNC в регистре IOSR
TRIGGER_ ENABLE	3	Запись	Если 1, следующая передача должна быть вызвана действием события DPEC, предусмотренным блоком OCDS (только режим чтения-записи)
EX_BUS_ TRACE	4	Запись	Разрешение трассировки адресов внешней шины
–	7-5	0	Зарезервированы

Бит MODE определяет режимы CERBERUS: чтения-записи или коммуникации. Бит COM_MODE_RST обеспечивает сброс битов CRSYNC и CWSYNC в регистре IOSR для прекращения запросов в коммуникационном режиме. Этот сброс не статический, он выполняется только один раз, когда обновляется регистр IOCONF. Бит COM_SYNC устанавливает соответствующий бит в регистре IOSR. Бит TRIGGER_ENABLE разрешает включение приемопередачи в режиме чтения-записи. Бит EX_BUS_TRACE разрешает включение приемопередачи по адресам внешней шины.

Регистр IOINFO

Регистр IOINFO, см. рисунок 7.25, таблицу 7.25, предусмотрен, чтобы анализировать ситуации, блокирующие передачу, или определять другие ошибочные состояния микроконтроллера. Это не физический регистр, но он предоставляет определенную информацию о состоянии микроконтроллера. После выполнения команды IO_SUPERVISOR эта информация выводится. Нужно отметить, что захваченные сигналы обычно статические только в течение этих блокировок и ошибочных ситуаций. Это означает, что регистр IOINFO не надо использовать в течение нормальной операции, и если используется в ошибочной ситуации (нет стартового бита для операций чтения-записи), он должен быть считан несколько раз для гарантии статичности.

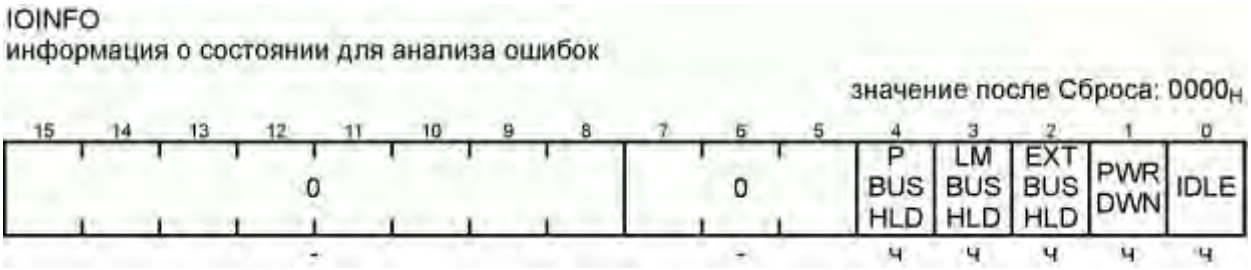


Рисунок 7.25 – Формат регистра IOINFO

Таблица 7.25 – Функциональное назначение полей регистра IOINFO

Поле	Биты	Тип	Описание
IDLE	0	Чтение	Микроконтроллер в состоянии Idle
POWER_DOWN	1	Чтение	Микроконтроллер в состоянии Power Down
EXTBUS_HOLD	2	Чтение	Внешняя шина занята
LMBUS_HOLD	3	Чтение	Шина локальной памяти (LM66 bus) занята
PBUS_HOLD	4	Чтение	Периферийная шина (PDBus) занята
–	7-5	0	Зарезервировано
–	15-8	0	Зарезервировано

Регистр TRADDR

Четырехбитный регистр TRADDR используется для трассировки (прослеживания) по адресам внешней шины. Это определяет четыре старшие бита адреса по внешней шине. Это устанавливается командой IO_SET_TRADDR со стороны внешнего интерфейса.

Регистры RWDATA и COMDATA

Регистр RWDATA является регистром данных для обоих типов передач: чтения и записи в режиме чтения-записи (RW Mode). Регистр COMDATA является эквивалентом для режима коммуникации (Communication Mode).

Регистр IOSR

Регистр IOSR, см. рисунок 7.26, таблицу 7.26, используется в режиме коммуникации для запрещения работы блока CERBERUS со стороны CPU из соображений безопасности и выполнения монитором трассировки. Регистр IOSR доступен только со стороны CPU.

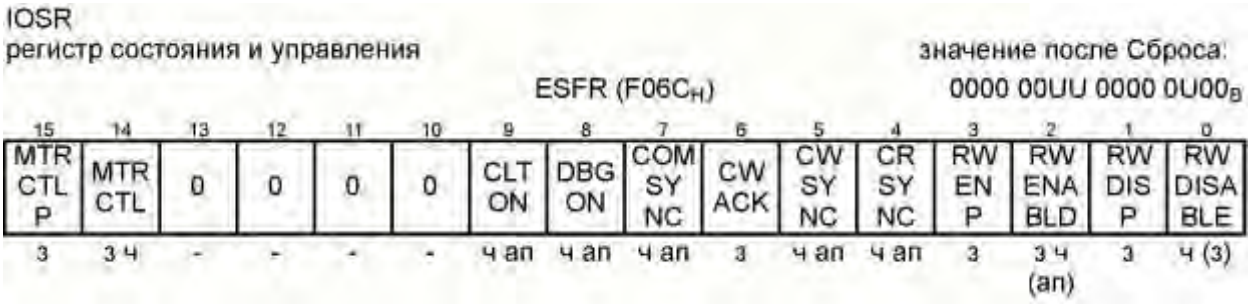


Рисунок 7.26 – Формат регистра IOSR

Таблица 7.26 – Функциональное назначение полей регистра IOSR

Поле	Биты	Тип	Описание
1	2	3	4
RW_DISABLE	0	Запись Чтение	Защита в режиме чтения-записи (RW Mode): 0 Режим чтения-записи разрешен. 1 Режим чтения-записи запрещен. Запись возможна только в режиме коммуникации.
RW_DIS_P	1	Запись	Бит защиты: 0 RW_DISABLE нельзя изменить. 1 RW_DISABLE можно изменить.
RW_ENABLED	2	Запись Чтение Аппаратное влияние	Используется для безопасности пользовательской программы. Сброс только через сброс JTAG, а не через сброс микроконтроллера.
RW_EN_P	3	Запись	Бит защиты: 0 RW_ENABLE нельзя изменить. 1 RW_ENABLE можно изменить.
CRSYNC	4	Чтение Аппаратное влияние	Чтение бита синхронизации для режима коммуникации: 0 Нет ожидания запроса приема. 1 Внешний отладчик запрашивает значение (COMDATA).
CWSYNC	5	Чтение Аппаратное влияние	Запись бита синхронизации для режима коммуникации: 0 Нет ожидания запроса передачи. 1 Внешний отладчик предлагает значение (COMDATA).

Окончание таблицы 7.26

1	2	3	4
CW_ASK	6	Запись	Подтверждение запроса записи в режиме коммуникации: 0 Нет действия. 1 Подтверждение, что посланное значение было прочитано монитором из COMDATA.
COM_SYNC	7	Чтение Аппаратное влияние	Высокий уровень синхронизации для режима коммуникации: 0 COM_SYNC в IOCONF равен 0. 1 COM_SYNC в IOCONF равен 1.
DBG_ON	8	Чтение Аппаратное влияние	0 Нет внешнего отладчика. 1 Есть внешний отладчик.
CLNT_ON	9	Чтение Аппаратное влияние	0 Клиент не выбран. 1 Клиент выбран.
–	11-13	–	Зарезервировано.
MTR_CTRL	14	Запись Чтение	Монитор, управляющий трассировкой: 0 Запрещен. 1 Разрешен.
MTR_CTRL_P	15	Запись	Бит защиты: 0 MTR_CTRL неизменяем. 1 MTR_CTRL может быть изменен.

Бит RW_DISABLE используется для предотвращения входа в режим чтения-записи. Он может быть установлен только центральным процессором в коммуникационном режиме. Если блок CERBERUS уже вошел в режим чтения-записи, все попытки CPU установить этот бит будут проигнорированы. Использование RW_DISABLE будет описано далее.

Бит RW_ENABLED не имеет никакого влияния на функционирование блока CERBERUS. Он предусмотрен для пользовательской программы для запоминания, разрешен ли уже режим чтения-записи или нет, так как он не затрагивается при сбросе микроконтроллера. Применение RW_ENABLED описано далее.

Биты CRSYNC, CWSYNC, CW_ACK и COM_SYNC используются в режиме коммуникации.

Бит DBG_ON показывает, присутствует ли внешний отладчик. Он непосредственно управляется внутренним сигналом сброса JTAG. Применение бита описано далее под заголовком «Безопасность системы» в подразделе 7.4 «Режимы работы».

Бит CLNT_ON показывает, выбран ли CERBERUS в текущее время внешним отладчиком. Он напрямую управляется сигналом выбора блока CERBERUS, который устанавливается регистром IOPATH в блоке JTAG.

Бит MTR_CTRL может быть использован монитором для управления трассировки памяти. Необходимо отметить, что это можно использовать, только если внешний отладчик управляет блоком CERBERUS через интерфейс JTAG.

7.4 Режимы работы

Режим чтения-записи RW Mode

Режим чтения-записи используется внешним интерфейсом (host) для чтения или записи ячеек памяти. В режиме чтения-записи используются команды IO_READ_WORD, IO_WRITE_WORD, IO_READ_BLOCK, IO_WRITE_BLOCK и IO_WRITE_BYTE. Значение адреса находится в IOADDR и устанавливается командой IO_SET_ADDRESS. Режиму чтения-записи необходим интерфейс DPEC для активного запроса чтения или записи данных.

Вход в режим чтения-записи

Вход в режим чтения-записи происходит, когда бит RW_ENABLED в регистре IOSR равен «0» и внешний интерфейс записывает «1» в бит MODE регистра IOCONF.

Поддерживаемые типы данных

Типом данных по умолчанию является 16-битное слово, и оно используется для передачи отдельных слов и блоков. Если внешний интерфейс хочет прочитать отдельный байт, он должен прочитать командой IO_READ_WORD соответствующее слово и извлечь необходимый байт самостоятельно.

Запись байтов поддерживается командой IO_WRITE_BYTE. Также для этой команды, внешний интерфейс должен сдвинуть все 16-битное слово, однако только выбранный байт будет записан. Его позиция определяется младшим битом адреса в регистре IOADDR.

Интерфейс DPEC

Интерфейс DPEC фактически выполняет чтение или запись ячеек памяти. Он конфигурируется регистром IOCONF и выполняет требуемое действие через сдвигатель JTAG. Данные передаются из или в регистр RWDATA. Передачи данных DPEC интерфейса всегда имеют наивысший приоритет CPU, однако не могут прервать последовательности ATOMIC/EXTx.

Режим коммуникации

Режим коммуникации является режимом блока CERBERUS для обеспечения связи (коммуникации) внешнего интерфейса (отладчика) и программы (монитора), запущенной CPU. Дополнительно, внешний интерфейс в этом режиме является мастером для всех приемов/передач. Внешний интерфейс запрашивает монитор для записи или чтения значения в или из COMDATA. Отличие коммуникационного режима от режима чтения-записи в том, что запрос чтения или записи не выполняется блоком CERBERUS, однако он устанавливает требуемые биты в доступных регистрах CPU для сообщения монитору, что внешний интерфейс желает передать IO_WRITE_WORD или принять IO_READ_WORD данные. Монитор опрашивает IOSR (регистр состояния ввода-вывода). Регистр IOADDR не используется.

Внешний интерфейс и монитор обмениваются информацией напрямую с регистром COMDATA. Для синхронизации доступов внешнего интерфейса и монитора в регистре состояния блока CERBERUS IOSR имеются четыре ассоциированных бита: CRSYNC, CWSYNC, CW_ACK и COM_SYNC.

Биты CRSYNC, CWSYNC и COM_SYNC устанавливаются и очищаются аппаратно, однако могут читаться монитором (CPU). Блок JTAG не влияет на стартовый бит на выводе TDO. Бит CW_ACK устанавливается монитором и подтверждает, что переданное значение было прочитано из COMDATA.

Режим коммуникации гарантирует, что все трансакции чтения и записи обсуживаются по всем правилам и в правильной последовательности, даже в случае перехода блока CERBERUS в режим ввода-вывода в это время.

Для двунаправленной коммуникации внешний интерфейс просто переключает между командами IO_READ_WORD и IO_WRITE_WORD.

Вход в режим коммуникации

Режим коммуникации является режимом по умолчанию после сброса RESET. Если блок CERBERUS находится в режиме чтения-записи, вход в режим коммуникации осуществляется записью внешним интерфейсом «0» в бит MODE регистра IOCONF.

Команды режима коммуникации

Режим коммуникации использует только команды IO_READ_WORD и IO_WRITE_WORD. Команда IO_SET_ADDRESS устанавливает регистр IOADDR только в режиме чтения-записи (не эффективна для режима коммуникации).

Пересылка данных монитором внешнему интерфейсу (прием Receive)

Бит CRSYNC сообщает монитору CPU, что внешний интерфейс желает принять новое значение регистра COMDATA. Он устанавливается в коммуникационном режиме для команды IO_READ_WORD. Бит CRSYNC автоматически очищается, когда монитор CPU делает запись в регистр COMDATA, независимо от режима (режим коммуникации или режим чтения-записи). Внешний интерфейс может запросить данные (CRSYNC не сбрасывается во время UpdateDR), обработать в режиме чтения-записи и затем перенести запрошенные данные в следующий цикл приема. Состояния бита CRSYNC показаны в таблице 7.27.

Таблица 7.27 – Бит CRSYNC

CRSYNC	Описание
1	Внешний интерфейс запрашивает у монитора запись данных в COMDATA
0	Нет запросов ожидания чтения

Пересылка данных внешним интерфейсом монитору (передача Send)

Бит CWSYNC сообщает монитору, что внешний интерфейс записал новое значение в регистр COMDATA. Он устанавливается в режиме коммуникации командой IO_WRITE_WORD. Бит CWSYNC очищается, когда монитор CPU устанавливает бит подтверждения CW_ACK в регистре IOSR независимо от режима (режим коммуникации или режим чтения-записи). Это позволяет посылать данные в режиме коммуникации, переключиться в режим чтения-записи и выполнять некоторые операции без необходимости ожидать прочтения монитором регистра COMDATA. В следующий вход в режим коммуникации биты занятости (busy bits) выходят, когда COMDATA еще не прочитан монитором.

Необходимо отметить, что в случае передачи командой IO_WRITE_WORD и последующего приема командой IO_READ_WORD, оба бита SWSYNC и CRSYNC устанавливаются и должны быть последовательно обслужены монитором. Предыдущий запрос приема блокирует передачу. Это означает, что требуемые данные должны быть переданы внешним интерфейсом, прежде чем выйдет новая команда передачи. Состояния бита CWSYNC показаны в таблице 7.28.

Таблица 7.28 – Бит CWSYNC

CWSYNC	Описание
1	Внешний интерфейс запрашивает у монитора чтение данных в COMDATA
0	COMDATA не действительно или монитор (CPU) читает COMDATA

Прерванные запросы

Если монитор CPU не обслуживает запрос чтения или записи регистра COMDATA, биты CWSYNC или CRSYNC могут быть сброшены битом COM_MODE_RST в регистре IOCONF.

Высокий уровень синхронизации

Чтобы улучшить надежность канала коммуникации, очень полезно видеть различие между командами отладчика и регулярным обменом данными. Например, отладчик (внешний интерфейс) прерывает свой запрос в тот момент, когда монитор отвечает, то высокий уровень синхронизации между внешним интерфейсом и монитором может быть потерян. Чтобы предотвратить это, предусмотрен бит COM_SYNC для синхронизации канала связи (коммуникации) между отладчиком и монитором на более высоком уровне. Это устанавливается в регистре IOCONF и может читаться отладчиком в регистре IOSR. Отладчик и монитор могут использовать этот бит просто для перезагрузки канала связи или для более продвинутого применения, могут использовать этот бит для пометки данных от отладчика к монитору как команды.

Включенные передачи (Triggered Transfer – DPEC)

Включенные передачи являются особенностью OCDS блока CERBERUS. Они могут использоваться для чтения или записи определенных адресов памяти, когда схема вызова становится активной.

Включенные передачи выполняются, когда CERBERUS находится в режиме чтения-записи, бит TRIGGER_ENABLE в регистре IOCONF равен 1, сдвиговый регистр JTAG запрашивается транзакцией (запросом с получением результата), и имеет место DPEC событие блока OCDS. Включенные передачи ведут себя как нормальные передачи, за исключением того, что передачи включаются после запроса передачи сдвигателем JTAG.

Трассировка памяти

Основным приложением для вызванных передач является трассировка определенных областей памяти. Это может быть сделано в тот момент, когда блок OCDS активирует по событию действие DPEC, если эта область памяти записана пользовательской программой. Блок CERBERUS конфигурируется для чтения ячеек памяти во время вызова. Максимальная скорость передач, которая может быть достигнута, определяется формулой

$$\text{NINSTR} = \frac{30}{\text{TINSTR} \times f_{\text{JTAG}}}, \qquad (7.1)$$

где NINSTR – число циклов команд, которые необходимы между двумя обращениями CPU к определенным областям памяти (memory location);

TINSTR – время цикла команды CPU, нс;

f_{JTAG} – тактовая частота интерфейса JTAG (TCK), МГц.

Например, если TINSTR = 100 нс и f_{JTAG} = 10 МГц, то доступы к памяти в каждый 30-й цикл команд могут быть полностью трассированы (ячейки памяти найдены, чтение проведено). Коэффициент 30 является суммой 16 бит данных, 10 бит для схемы состояний JTAG, инструкции ввода-вывода и стартового бита, а также 4 бит синхронизации между началом передач и выводом данных. Если частота включения выше, некоторые доступы могут быть потеряны. Для регистрации внешним отладчиком этих пропущенных событий устанавливается бит чтения признака dirty_bit. Этот бит дополняет прочитанные данные, когда они выведены. Описание бита dirty_bit приведено в таблице 7.29.

Таблица 7.29 – Бит чтения признака dirty_bit

Значение	Описание
1	По крайней мере, одно событие пропущенной вызванной передачи между последним вызванным чтением и текущим
0	Нет случая пропуска

Трассировка по адресам внешней шины

Это специальный рабочий режим интерфейса DPEC для ускоренной трассировки. В этом режиме данные не записываются в RWDATA и выдаются (выдвигаются) через JTAG порт, однако напрямую записываются по адресам внешней шины. Данные захватываются отладчиком из внешней шины («trace box»). Этот вид трассировки (просмотра) может быть разрешен только в режиме коммуникации и может применяться параллельно ему.

Для вызова передачи следует установить биты: MODE=0_B, TRIGGER_ENABLE=1_B, EX_BUS_TRACE=1_B (все в регистре IOCONF). Адрес по внешней шине имеет формат, приведенный на рисунке 7.27.

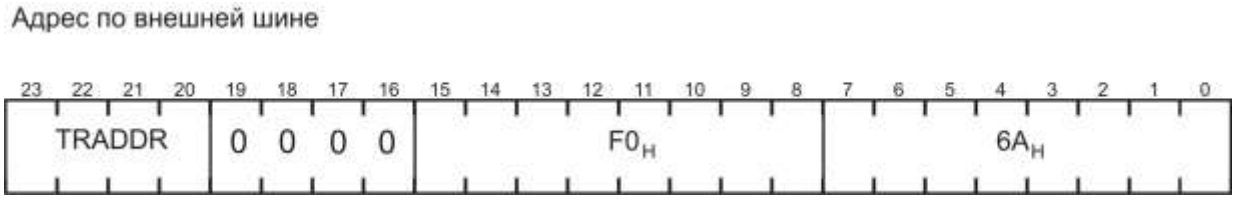


Рисунок 7.27 – Адрес по внешней шине

Регистр TRADDR устанавливает старшие значащие биты, остальные аппаратно установлены в $0F06A_H$.

Управляемая монитором трассировка

Управляемая монитором трассировка предусмотрена для трассировки в конечном продукте, когда неудобно делать доступ через интерфейс JTAG. Очень важно, что монитор использует эту функциональность только тогда, когда нет внешнего отладчика, связанного с блоком CERBERUS через JTAG. Иначе произойдут ошибки, потому что эта функциональность разделяет ресурсы регистра COMDATA, см. рисунок 7.28, таблицу 7.30, и регистра RWDATA, см. рисунок 7.29, таблицу 7.31, с нормальным режимом, использующим внешний отладчик. Контролируемая монитором трассировка не является риском для безопасности. Даже если она неумышленно разрешена пользовательской программой, передача происходит только тогда, когда OCDS ее вызывает. Разрешение OCDS является очень хорошей защитой.

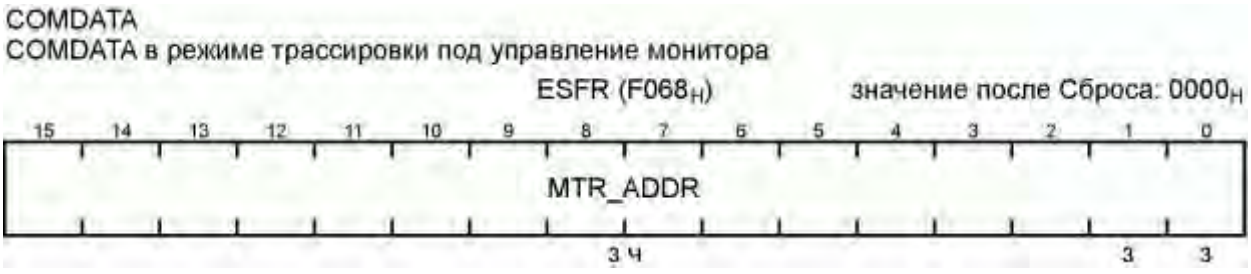


Рисунок 7.28 – Формат регистра COMDATA

Таблица 7.30 – Функциональное назначение поля регистра COMDATA

Поле	Биты	Тип	Описание
MT_ADDR	15-0	Запись	Установленные биты (15-0) выбранного адресного регистра MTR_SELECT_ADDR

RWDATA

RWDATA в режиме трассировки под управление монитора

ESFR ($F06A_H$) значение после Сброса: 0000_H

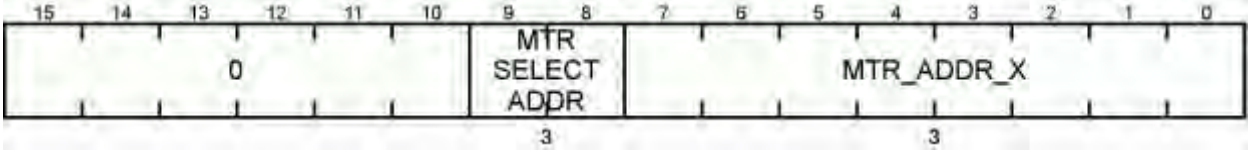


Рисунок 7.29 – Формат регистра RWDATA

Таблица 7.31 – Биты регистра RWDATA

Поле	Биты	Тип	Описание
MTR_ADDR_X	7-0	Запись	Установление битов (23-16) выбранного (MT_SELECT_ADDR) адресного регистра:
MTR_SELECT_ADDR	9-8	Запись	00 Нет выбора. 01 Выбор адреса источника MTR. 10 Выбор адреса назначения MTR. 11 Зарезервировано.
–	15-10	–	Зарезервировано.

Управляемая монитором трассировка является эквивалентом вызванных передач, однако управляется монитором, запущенным CPU. Это может быть использовано для перемещения произвольного интервала ячеек памяти блоком OCDS действием DPEC на событие. Перемещение выполняется, когда CERBERUS не выбран (CLNT_ON=0_B, MTR_CTRL=1_B) и есть вызванная передача.

Адрес источника и адрес назначения программируются с помощью MTR_SELECT_ADDR, MTR_ADDR_X и MTR_ADDR в регистрах RWDATA и COMDATA. Записью в RWDATA выбирается адрес (источника и назначения с помощью MTR_SELECT_ADDR) и записывается старший байт адреса. Младшие 16 бит могут быть запрограммированы с помощью COMDATA.

Следующий пример С-кода показывает разрешение трассировки монитором:

```
                                    // Адрес начала трассировки: 0x543210
Unsigned trace_source_ptr_ext = 0x54;
Unsigned trace_source_ptr = 0x3210;
                                    // Адрес конца трассировки: 0xABCDEF
Unsigned trace_target_ptr_ext = 0xAB;
Unsigned trace_target_ptr = 0xCDEF;
                                    // Установка адреса начала трассировки
RWDATA = 0x0100 | trace_source_ptr_ext;
COMDATA = trace_source_ptr;
                                    // Установка адреса конца  трассировки
RWDATA = 0x0200 | trace_target_ptr_ext;
COMDATA = trace_target_ptr;
                                    // Старт трассировки под управлением монитора
                                    // с MTRL_CTRL
IOSR = 0xC000
                                    // Программирование OCDS для формирования
  …                                 // триггеров DPEC:
```

Обработка ошибок

Блок CERBERUS входит в ошибочное состояние (error state) при внутреннем сбросе (кроме сброса JTAG). Его можно обойти командой IO_SUPERVISOR. Пока есть

ошибочное состояние, каждая команда, за исключением IO_SUPERVISOR, отвечает неопределенным количеством битов занятости (busy bits).

Другим ошибочным состоянием является блокировка внутренней шины для DPEC передач. Если имеет место такое состояние, можно использовать команду IO_SUPERVISOR для чтения регистра IOINFO, который предоставляет информацию для анализа.

Безопасность системы

После сброса блок CERBERUS находится в режиме коммуникации и ему необходимо, по крайней мере, 30 циклов тактового сигнала TCK, чтобы перейти в режим чтения-записи (10 циклов для подтверждения сброса командой IO_SUPERVISOR и 20 циклов для установки регистра IOCONF). Если пользовательская программа, запущенная CPU, устанавливает RST_HLT сразу после сброса, то нет возможности с внешней стороны ввести блок CERBERUS в режим чтения-записи через интерфейс JTAG.

Чтобы иметь защищенную систему в области, к которой могут получить доступ зарегистрированные пользователи, может применяться следующее ниже решение (все биты находятся в регистре IOSR).

Первая команда пользовательской программы после сброса запрещает режим чтения-записи битом RST_HLT=1_B, если RW_ENABLED=0_B.

Пользовательская программа проверяет бит DBG_ON для определения подключения внешнего отладчика, если его нет, то программа продолжает свою работу.

Внешний отладчик передает ключевые коды (n × 16) бит в режиме коммуникации.

Пользовательская программа стартует прием и сравнение этих кодов (чисел) некоторое время t_D после сброса. Это время должно быть достаточно долгим (около 100 мс), чтобы позволить даже медленным (5 кГц) драйверам JTAG выполнять запрошенную передачу. Дополнительно, рекомендуется опрашивать CRSYNC в разумных интервалах, чтобы позволить «горячее подключение» внешнего отладчика.

Если все коды корректны, пользовательская программа сбрасывает RST_HLT и устанавливает RW_ENABLRD.

Теперь пользовательская программа знает, что блок CERBERUS однажды разрешен и не предотвращает разрешение после последующих сбросов.

Сохранение мощности

Блок CERBERUS находится в режиме сохранения мощности, когда он не выбран со стороны порта JTAG. Только регистр IOSR всегда работает и доступен. Если разрешен режим управляемой монитором трассировки, требуемые ресурсы функционируют.

Сброс со стороны порта JTAG

Если активизируется внутренний сброс JTAG, то все запросы режима чтения-записи и режима коммуникации прекращаются и также сбрасываются биты CRSYNC и CWSYNC. Поведение регистров при сбросе описано в таблице 7.32.

Сброс со стороны микроконтроллера

В этом случае все команды ввода-вывода, за исключением IO_CONFIG, отвечают неопределенным числом бит занятости. Это ошибочное состояние (error state). Внешний интерфейс должен подтвердить это состояние командой IO_SUPERVISOR. Это делается для уведомления внешнего интерфейса, что кое-что неожидаемое возможно случилось и необходимо проверить канал связи с монитором.

Сброс JTAG всегда требует последующий сброс CPU для гарантии того, что сдвигатель JTAG и управляющая часть блока CERBERUS находятся в определенных состояниях при всех условиях.

Таблица 7.32 – Поведение регистров при сбросе

Регистр	Сброс JTAG	Сброс микроконтроллера
CLIENT_ID	Аппаратно закодировано	Аппаратно закодировано
COMDATA	Не изменяется	0000_H
IOADDR	000000_H	Не изменяется
IOCONF	00_H	Не изменяется
IOINFO	Спецификации кристалла	Спецификации кристалла
IOSR	UUUU UUUU UUUU $U0UU_B$ UUUU UU00 UUUU $U0UU_B$ [1)]	0000 0000 0000 $0U00_B$ 0000 00UU 0000 $0U00_B$ [1)]
RWDATA	Не изменяется	0000_H
TRADDR	0_H	Не изменяется

[1)] С точки зрения программного обеспечения биты 9 и 8 имеют такое поведение, потому что источником является сброс JTAG области регулирования (U – бит не изменяется). Только их синхронизация в триггерах типа «flip-flop» связана со сбросом микроконтроллера.

8 Система команд микроконтроллера

Таблицы 8.1 и 8.2 дают краткую информацию о командах и кодах команд микроконтроллера 1887ВЕ3Т.

Таблица 8.1 – Перекрестная таблица команд и кодов команд

	0x	1x	2x	3x	4x	5x	6x	7x
x0	ADD	ADDC	SUB	SUBC	CMP	XOR	AND	OR
x1	ADDB	ADDCB	SUBB	SUBCB	CMPB	XORB	ANDB	ORB
x2	ADD	ADDC	SUB	SUBC	CMP	XOR	AND	OR
x3	ADDB	ADDCB	SUBB	SUBCB	CMPB	XORB	ANDB	ORB
x4	ADD	ADDC	SUB	SUBC	–	XOR	AND	OR
x5	ADDB	ADDCB	SUBB	SUBCB	–	XORB	ANDB	ORB
x6	ADD	ADDC	SUB	SUBC	CMP	XOR	AND	OR
x7	ADDB	ADDCB	SUBB	SUBCB	CMPB	XORB	ANDB	ORB
x8	ADD	ADDC	SUB	SUBC	CMP	XOR	AND	OR
x9	ADDB	ADDCB	SUBB	SUBCB	CMPB	XORB	ANDB	ORB
xA	BFLDL	BFLDH	BCMP	BMOVN	BMOV	BOR	BAND	BXOR
xB	MUL	MULU	PRIOR	–	DIV	DIVU	DIVL	DIVLU
xC	ROL	ROL	ROR	ROR	SHL	SHL	SHR	SHR
xD	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR
xE	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR
xF	BSET	BSET	BSET	BSET	BSET	BSET	BSET	BSET

Таблица 8.2 – Перекрестная таблица команд и кодов команд

	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
x0	CMPI1	CMPI2	CMPD1	CMPD2	MOVBZ	MOVBS	MOV	MOV
x1	NEG	CPL	NEGB	CPLB	-	AT/EXTR	MOVB	MOVB
x2	CMPI1	CMPI2	CMPD1	CMPD2	MOVBZ	MOVBS	PCALL	MOV
x3	–	–	–	–	–	–	–	MOVB
x4	MOV	MOV	MOVB	MOVB	MOV	MOV	MOVB	MOVB
x5	–	–	DISWDT	EINIT	MOVBZ	MOVBS	–	–
x6	CMPI1	CMPI2	CMPD1	CMPD2	SCXT	SCXT	MOV	MOV
x7	IDLE	PWRDN	SRVWDT	SRST	–	EXTP/S/R	MOVB	MOVB
x8	MOV	MOV	MOV	MOV	MOV	MOV	MOV	–
x9	MOVB	MOVB	MOVB	MOVB	MOVB	MOVB	MOVB	–
xA	JB	JNB	JBC	JNBS	CALLA	CALLS	JMPA	JMPS
xB	–	TRAP	CALLI	CALLR	RET	RETS	RETP	RETI
xC	–	JMPI	ASHR	ASHR	NOP	EXTP/S/R	PUSH	POP
xD	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR	JMPR
xE	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR	BCLR
xF	BSET	BSET	BSET	BSET	BSET	BSET	BSET	BSET

8.1 Команды микроконтроллера

В настоящем подразделе представлено подробное описание команд микроконтроллера в таблицах 8.3 – 8.15.

Условные обозначения в указанных таблицах:

Rw	–	2-байтовый регистр общего назначения GPR: R0, …, R15.
Rb	–	Байтовый регистр общего назначения GPR: RL0, RH0, …, RL7, RH7.
reg	–	Регистры специального назначения SFR или GPR (если команда работает с типом данных BYTE и один из операндов – SFR, то доступ при адресации «reg» возможен только к младшему байту).
mem	–	Прямая адресация в памяти слова или байта.
[...]	–	Косвенная адресация в памяти слова или байта. Любой 2-байтовый GPR может использоваться как косвенный указатель адреса, кроме арифметических, логических команд и команд сравнения, для которых разрешено использовать только регистры R0, …, R3.
bitaddr	–	Указатель бита в битадресуемом пространстве памяти.
bitoff	–	Указатель слова в битадресуемом пространстве памяти.
#datax	–	Непосредственная константа (число значащих младших разрядов, которые используются в команде, представлены соответствующим добавлением «x»).
#mask8	–	Непосредственная 8-битовая маска, используемая для изменения нескольких разрядов.

Операции умножения и деления

MDL, MDH регистры являются регистрами источников и/или приемников для команд умножения и деления.

Операции перехода:

caddr	–	Прямой 16-битовый внутрисегментный адрес перехода, изменяет значение указателя IP.
seg	–	Прямой 8-битовый номер сегмента для перехода, изменяет значение указателя сегмента.
rel	–	Знаковое 8-битовое смещение по отношению к указателю инструкции IP.
#trap7	–	Прямой 7-битовый номер вектора прерывания.

Расширенные операции

Команды EXTxx изменяют стандартную схему адресации регистров (reg) для SFR/ESFR и страничную адресацию, использующую регистры DPPx.

#pag10	–	Непосредственный 10-битовый номер страницы памяти.
#seg8	–	Непосредственный 8-битовый номер сегмента памяти.

Коды условий перехода (cc):

cc_UC	– безусловный переход;
cc_Z	– если результат равен нулю;
cc_NZ	– если результат не равен нулю;
cc_V	– если произошло переполнение во время выполнения команды;
cc_NV	– если не произошло переполнения во время выполнения команды;
cc_N	– если результат отрицательный;
cc_NN	– если результат не отрицательный;
cc_C	– если произошел перенос во время выполнения инструкции;
cc_NC	– если не произошел перенос во время выполнения инструкции;
cc_EQ	– если сравниваемые операнды эквивалентны;
cc_NE	– если сравниваемые операнды не эквивалентны;

cc_ULT – меньше (без знака);
cc_ULE – меньше или равно (без знака);
cc_UGE – больше или равно (без знака);
cc_UGT – больше (без знака);
cc_SLE – меньше или равно (со знаком);
cc_SGE – больше или равно (со знаком);
cc_SGT – больше (со знаком);
cc_NET – если сравниваемые операнды не эквивалентны и один из операндов не равен наименьшему отрицательному числу.

Таблица 8.3 – Команды и режимы адресации

Мнемокод	Режимы адресации			Размер, байт
1	2			3
ADD[B] ADDC[B] AND[B] OR[B] SUB[B] SUBC[B] XOR[B]	Rwn Rwn Rwn Rwn Reg Reg mem	Rwm [Rwi] [Rwi+] #data3 #data16 mem reg	* * * * 	2 2 2 2 4 4 4
ASHR ROL / ROR SHL / SHR	Rwn Rwn	Rwm #data4		2 2
BAND BCMP BMOV BMOVN BOR / BXOR	bitaddrZ.z	bitaddrQ.q		4
BCLR BSET	bitaddrQ.q			2
BFLDH BFLDL	bitoffQ.q	#mask8#data8		4
MOV[B]	Rwn Rwn Rwn Rwn [Rwm] [–Rwm] [Rwn] [Rwn + 1] [Rwn] reg Rwn [Rwm + #d16] [Rwn] Mem reg mem	Rwm #data4 [Rwm] [Rwm+] Rwn Rwn [Rwm] [Rwm] [Rwm +] #data16 [Rwm+#d16] Rwn mem [Rwn] mem reg	* * * * * * ** * * 	2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4

Продолжение таблицы 8.3

1	2		3
MOVBS MOVBZ	Rwn reg mem	Rbm mem reg	2 4 4
EXTS EXTSR	Rwm #seg	#irang2 #irang2	2 4
NOP RET RETI RETS	–		2
CPL[B] NEG[B]	Rwn	*	2
DIV DIVL DIVLU DIVU	Rwn		2
MUL MULU	Rwn	Rwm	2
CMPD1 / 2 CMPI1 / 2	Rwn Rwn Rwn	#data4 #data16 mem	2
CMP[B]	Rwn Rwn Rwn Rwn reg reg	Rwm [Rwi] [Rwi+] #data3 #data16 mem	* * * * ** 	2 2 2 2 4 4
CALLA JMPA	cc	caddr	4
CALLI JMPI	Cc	[Rwn]	2
CALLS JMPS	Seg	caddr	4
CALLR	Rel		2
JMPR	cc	rel	2
JB JBC JNB JNBS	bitaddrQ.q	rel	4
PCALL	reg	caddr	4
POP PUSH RETP	reg		2
SCXT	Reg reg	#data16 mem	4 4
PRIOR	Rwn	Rwn	2
TRAP	#trap7		2
ATOMIC EXTR	#irang2		2

Окончание таблицы 8.3

1	2	3
EXTP	Rwm #irag2	2
EXTPR	#pag #irag2	4
SRST / IDLE PWRDN SRVWDT DISWDT EINIT	–	4

_____

* Команды для работы с байтами ([B]) используют Rb вместо Rw (не для [Rwn] !).

** Команды для работы с байтами ([B]) используют #data8 вместо #data16.

Таблица 8.4 – Арифметические команды

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
ADD	Rw, Rw	2	Сложение GPR с GPR
ADD	Rw, [Rw]	2	Сложение ячейки памяти (с косвенной адресацией) и GPR
ADD	Rw, [Rw+]	2	Сложение ячейки памяти (с косвенной адресацией) и GPR с последующим увеличением указателя ячейки памяти на 2
ADD	Rw, #data3	2	Сложение непосредственного операнда с GPR
ADD	reg, #data16	4	Сложение непосредственного операнда с регистром
ADD	reg, mem	4	Сложение ячейки памяти с регистром
ADD	mem, reg	4	Сложение регистра с ячейкой памяти
ADDB	Rb, Rb	2	Побайтное сложение GPR с GPR
ADDB	Rb, [Rw]	2	Побайтное сложение ячейки памяти (с косвенной адресацией) и GPR
ADDB	Rb, [Rw+]	2	Побайтное сложение ячейки памяти (с косвенной адресацией) и GPR с последующим увеличением указателя ячейки памяти на 1
ADDB	Rb, #data3	2	Побайтное сложение непосредственного операнда с GPR
ADDB	reg, #data8	4	Побайтное сложение непосредственного операнда с регистром
ADDB	reg, mem	4	Побайтное сложение ячейки памяти с регистром
ADDB	mem, reg	4	Побайтное сложение регистра с ячейкой памяти
ADDC	Rw, Rw	2	Сложение GPR с GPR и перенос
ADDC	Rw, [Rw]	2	Сложение ячейки памяти (с косвенной адресацией) и GPR и перенос
ADDC	Rw, [Rw+]	2	Сложение ячейки памяти (с косвенной адресацией) и GPR с переносом и последующим увеличением указателя ячейки памяти на 2
ADDC	Rw, #data3	2	Сложение непосредственного операнда с GPR и перенос
ADDC	reg, #data16	4	Сложение непосредственного операнда с регистром и перенос

Продолжение таблицы 8.4

1	2	3	4
ADDC	reg, mem	4	Сложение ячейки памяти с регистром и перенос
ADDC	mem, reg	4	Сложение регистра с ячейкой памяти и перенос
ADDCB	Rb, Rb	2	Побайтное GPR и GPR и перенос
ADDCB	Rb, [Rw]	2	Побайтное сложение ячейки памяти (с косвенной адресацией) и GPR и перенос
ADDCB	Rb, [Rw+]	2	Побайтное сложение ячейки памяти (с косвенной адресацией) и GPR с переносом и последующим увеличением указателя ячейки памяти на 1
ADDCB	Rb, #data3	2	Побайтное сложение непосредственного операнда с GPR и перенос
ADDCB	reg, #data8	4	Побайтное сложение непосредственного операнда с регистром и перенос
ADDCB	reg, mem	4	Побайтное сложение ячейки памяти с регистром и перенос
ADDCB	mem, reg	4	Побайтное сложение регистра с ячейкой памяти и перенос
SUB	Rw, Rw	2	Вычитание GPR из GPR
SUB	Rw, [Rw]	2	Вычитание ячейки памяти (с косвенной адресацией) из GPR
SUB	Rw, [Rw+]	2	Вычитание ячейки памяти (с косвенной адресацией) из GPR с последующим увеличением указателя памяти на 2
SUB	Rw, #data3	2	Вычитание непосредственного операнда из GPR
SUB	reg, #data16	4	Вычитание непосредственного операнда из регистра
SUB	reg, mem	4	Вычитание ячейки памяти из регистра
SUB	mem, reg	4	Вычитание регистра из ячейки памяти
SUBB	Rb, Rb	2	Побайтовое вычитание GPR из GPR
SUBB	Rb, [Rw]	2	Побайтовое вычитание ячейки памяти (с косвенной адресацией) из GPR
SUBB	Rb, [Rw+]	2	Побайтовое вычитание ячейки памяти (с косвенной адресацией) из GPR с последующим увеличением указателя памяти на 1
SUBB	Rb, #data3	2	Побайтовое вычитание непосредственного операнда из GPR
SUBB	reg, #data8	4	Побайтовое вычитание непосредственного операнда из регистра
SUBB	reg, mem	4	Побайтовое вычитание ячейки памяти из регистра
SUBB	mem, reg	4	Побайтовое вычитание регистра из ячейки памяти
SUBC	Rw, Rw	2	Вычитание GPR из GPR и перенос
SUBC	Rw, [Rw]	2	Вычитание ячейки памяти (с косвенной адресацией) из GPR и перенос
SUBC	Rw, [Rw+]	2	Вычитание ячейки памяти (с косвенной адресацией) из GPR с последующим увеличением указателя памяти на 2 и перенос
SUBC	Rw, #data3	2	Вычитание непосредственного операнда из GPR и перенос
SUBC	reg, #data16	4	Вычитание непосредственного операнда из регистра и перенос

Окончание таблицы 8.4

1	2	3	4
SUBC	reg, mem	4	Вычитание ячейки памяти из регистра и перенос
SUBC	mem, reg	4	Вычитание регистра из ячейки памяти и перенос
SUBCB	Rb, Rb	2	Побайтовое вычитание GPR из GPR и перенос
SUBCB	Rb, [Rw]	2	Побайтовое вычитание ячейки памяти (с косвенной адресацией) из GPR и перенос
SUBCB	Rb, [Rw+]	2	Побайтовое вычитание ячейки памяти (с косвенной адресацией) из GPR с последующим увеличением указателя памяти на 1 и перенос
SUBCB	Rb, #data3	2	Побайтовое вычитание непосредственного операнда из GPR и перенос
SUBCB	reg, #data8	4	Побайтовое вычитание непосредственного операнда из регистра и перенос
SUBCB	reg, mem	4	Побайтовое вычитание ячейки памяти из регистра и перенос
SUBCB	mem, reg	4	Побайтовое вычитание регистра из ячейки памяти и перенос
MUL	Rw, Rw	2	Умножение со знаком GPR (слово) и GPR (слово)
MULU	Rw, Rw	2	Умножение без знака GPR (слово) и GPR (слово)
DIV	Rw	2	Деление со знаком регистра MDL (слово) на GPR (слово)
DIVL	Rw	2	Деление (длинное) со знаком регистра MD (2 слова) на GPR (слово)
DIVLU	Rw	2	Деление (длинное) без знака регистра MD (2 слова) на GPR (слово)
DIVU	Rw	2	Деление без знака регистра MDL (слово) на GPR (слово)
CPL	Rw	2	Арифметическое умножение GPR на GPR
CPLB	Rb	2	Побайтовое арифметическое умножение GPR на GPR
NEG	Rw	2	Изменение знака GPR
NEGB	Rb	2	Побайтовое изменение знака GPR

Таблица 8.5 – Логические команды

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
AND	Rw, Rw	2	Побитовое логическое «И» GPR с GPR
AND	Rw, [Rw]	2	Побитовое логическое «И» ячейки памяти (с косвенной адресацией) и GPR
AND	Rw, [Rw+]	2	Побитовое логическое «И» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 2
AND	Rw, #data3	2	Побитовое логическое «И» непосредственного операнда с GPR
AND	reg, #data16	4	Побитовое логическое «И» непосредственного операнда с регистром
AND	reg, mem	4	Побитовое логическое «И» ячейки памяти с регистром

Продолжение таблицы 8.5

1	2	3	4
AND	mem, reg	4	Побитовое логическое «И» регистра с ячейкой памяти
ANDB	Rb, Rb	2	Побайтовое побитовое логическое «И» GPR с GPR
ANDB	Rb, [Rw]	2	Побайтовое побитовое логическое «И» ячейки памяти (с косвенной адресацией) и GPR
ANDB	Rb, [Rw+]	2	Побайтовое побитовое логическое «И» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 1
ANDB	Rb, #data3	2	Побайтовое побитовое логическое «И» непосредственного операнда с GPR
ANDB	reg, #data8	4	Побайтовое побитовое логическое «И» непосредственного операнда с регистром
ANDB	reg, mem	4	Побайтовое побитовое логическое «И» ячейки памяти с регистром
ANDB	mem, reg	4	Побайтовое побитовое логическое «И» регистра с ячейкой памяти
OR	Rw, Rw	2	Побитовое логическое «ИЛИ» GPR с GPR
OR	Rw, [Rw]	2	Побитовое логическое «ИЛИ» ячейки памяти (с косвенной адресацией) и GPR
OR	Rw, [Rw+]	2	Побитовое логическое «ИЛИ» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 2
OR	Rw, #data3	2	Побитовое логическое «ИЛИ» непосредственного операнда с GPR
OR	reg, #data16	4	Побитовое логическое «ИЛИ» непосредственного операнда с регистром
OR	reg, mem	4	Побитовое логическое «ИЛИ» ячейки памяти с регистром
OR	mem, reg	4	Побитовое логическое «ИЛИ» регистра с ячейкой памяти
ORB	Rb, Rb	2	Побайтовое побитовое логическое «ИЛИ» GPR с GPR
ORB	Rb, [Rw]	2	Побайтовое побитовое логическое «ИЛИ» ячейки памяти (с косвенной адресацией) и GPR
ORB	Rb, [Rw+]	2	Побайтовое побитовое логическое «ИЛИ» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 1
ORB	Rb, #data3	2	Побайтовое побитовое логическое «ИЛИ» непосредственного операнда с GPR
ORB	reg, #data8	4	Побайтовое побитовое логическое «ИЛИ» непосредственного операнда с регистром
ORB	reg, mem	4	Побайтовое побитовое логическое «ИЛИ» ячейки памяти с регистром
ORB	mem, reg	4	Побайтовое побитовое логическое «ИЛИ» регистра с ячейкой памяти
XOR	Rw, Rw	2	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» GPR с GPR

Окончание таблицы 8.5

1	2	3	4
XOR	Rw, [Rw]	2	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти (с косвенной адресацией) и GPR
XOR	Rw, [Rw+]	2	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 2
XOR	Rw, #data3	2	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» непосредственного операнда с GPR
XOR	reg, #data16	4	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» непосредственного операнда с регистром
XOR	reg, mem	4	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти с регистром
XOR	mem, reg	4	Побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» регистра с ячейкой памяти
XORB	Rb, Rb	2	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» GPR с GPR
XORB	Rb, [Rw]	2	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти (с косвенной адресацией) и GPR
XORB	Rb, [Rw+]	2	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти (с косвенной адресацией) с GPR с последующим увеличением указателя памяти на 1
XORB	Rb, #data3	2	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» непосредственного операнда с GPR
XORB	reg, #data8	4	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» непосредственного операнда с регистром
XORB	reg, mem	4	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» ячейки памяти с регистром
XORB	mem, reg	4	Побайтовое побитовое логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» регистра с ячейкой памяти

Таблица 8.6 – Логические команды с битами

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
BCLR	bitaddr	2	Сброс бита
BSET	bitaddr	2	Установка бита
BMOV	bitaddr, bitaddr	4	Копирование бита в бит
BMOVN	bitaddr, bitaddr	4	Копирование инвертированного бита в бит
BAND	bitaddr, bitaddr	4	Логическое «И» бита с битом
BOR	bitaddr, bitaddr	4	Логическое «ИЛИ» бита с битом
BXOR	bitaddr, bitaddr	4	Логическое «ИСКЛЮЧАЮЩЕЕ ИЛИ» бита с битом

Окончание таблицы 8.6

1	2	3	4
BCMP	bitaddr, bitaddr	4	Сравнение бита с битом
BFLDH	bitoff, #mask8, #data8	4	Побитовая модификация маскированных битов старшего байта битадресуемого слова в памяти с непосредственным операндом
BFLDL	bitoff, #mask8, #data8	4	Побитовая модификация маскированных битов младшего байта битадресуемого слова в памяти с непосредственным операндом
CMP	Rw, Rw	2	Сравнение GPR с GPR
CMP	Rw, [Rw]	2	Сравнение ячейки памяти (с косвенной адресацией) и GPR
CMP	Rw, [Rw+]	2	Сравнение ячейки памяти (с косвенной адресацией) и GPR с последующим увеличением указателя ячейки памяти на 2
CMP	Rw, #data3	2	Сравнение непосредственного операнда с GPR
CMP	reg, #data16	4	Сравнение непосредственного операнда с регистром
CMP	reg, mem	4	Сравнение ячейки памяти с регистром
CMPB	Rb, Rb	2	Побайтное сравнение GPR с GPR
CMPB	Rb, [Rw]	2	Побайтное сравнение ячейки памяти (с косвенной адресацией) с GPR
CMPB	Rb, [Rw+]	2	Побайтное сравнение ячейки памяти (с косвенной адресацией) и GPR с последующим увеличением указателя ячейки памяти на 1
CMPB	Rb, #data3	2	Побайтное сравнение непосредственного операнда с GPR
CMPB	reg, #data8	4	Побайтное сравнение непосредственного операнда с регистром
CMPB	reg, mem	4	Побайтное сравнение регистра с непосредственным операндом

Таблица 8.7 – Команды сравнения и управления циклом

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
CMPD1	Rw, #data4	2	Сравнение непосредственного операнда с GPR с последующим уменьшением регистра на 1
CMPD1	Rw, #data16	4	Сравнение непосредственного операнда с GPR с последующим уменьшением регистра на 1
CMPD1	Rw, mem	4	Сравнение ячейки памяти с GPR с последующим уменьшением регистра на 1
CMPD2	Rw, #data4	2	Сравнение непосредственного операнда с GPR с последующим уменьшением регистра на 2
CMPD2	Rw, #data16	4	Сравнение непосредственного операнда с GPR с последующим уменьшением регистра на 2
CMPD2	Rw, mem	4	Сравнение ячейки памяти с GPR с последующим уменьшением регистра на 2
CMPI1	Rw, #data4	2	Сравнение непосредственного операнда с GPR с последующим увеличением регистра на 1

Окончание таблицы 8.7

1	2	3	4
CMPI1	Rw, #data16	4	Сравнение непосредственного операнда с GPR с последующим увеличением регистра на 1
CMPI1	Rw, mem	4	Сравнение ячейки памяти с GPR с последующим увеличением регистра на 1
CMPI2	Rw, #data4	2	Сравнение непосредственного операнда с GPR с последующим увеличением регистра на 2
CMPI2	Rw, #data16	4	Сравнение непосредственного операнда с GPR с последующим увеличением регистра на 2
CMPI2	Rw, mem	4	Сравнение ячейки памяти с GPR с последующим увеличением регистра на 2

Таблица 8.8 – Команда приоритета

Мнемокод команды	Операнды	Размер, байт	Описание команды
PRIOR	Rw, Rw	2	Определение количества циклов сдвига для нормализации GPR и сохранение результата в GPR

Таблица 8.9 – Команды сдвига и циклического сдвига

Мнемокод команды	Операнды	Размер, байт	Описание команды
SHL	Rw, Rw	2	Логический сдвиг влево GPR на количество разрядов, указанное в GPR
SHL	Rw, #data4	2	Логический сдвиг влево GPR на количество разрядов, указанное в непосредственном операнде
SHR	Rw, Rw	2	Логический сдвиг вправо GPR на количество разрядов, указанное в GPR
SHR	Rw, #data4	2	Логический сдвиг вправо GPR на количество разрядов, указанное в непосредственном операнде
ROL	Rw, Rw	2	Циклический сдвиг влево GPR на количество разрядов, указанное в GPR
ROL	Rw, #data4	2	Циклический сдвиг влево GPR на количество разрядов, указанное в непосредственном операнде
ROR	Rw, Rw	2	Циклический сдвиг вправо GPR на количество разрядов, указанное в GPR
ROR	Rw, #data4	2	Циклический сдвиг вправо GPR на количество разрядов, указанное в непосредственном операнде
ASHR	Rw, Rw	2	Арифметический (со знаковым битом) сдвиг вправо содержимого GPR на количество разрядов, указанное в GPR
ASHR	Rw, #data4	2	Арифметический (со знаковым битом) сдвиг вправо GPR на количество разрядов, указанное в непосредственном операнде

Таблица 8.10 – Команды пересылки данных

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
MOV	Rw, Rw	2	Копирование GPR в GPR
MOV	Rw, #data4	2	Копирование непосредственного операнда в GPR
MOV	reg, #data16	4	Копирование непосредственного операнда в регистр
MOV	Rw, [Rw]	2	Копирование ячейки памяти (с косвенной адресацией) в GPR
MOV	Rw, [Rw+]	2	Копирование ячейки памяти (с косвенной адресацией) в GPR с последующим увеличением указателя памяти на 2
MOV	[Rw], Rw	2	Копирование GPR в ячейку памяти
MOV	[–Rw], Rw	2	Уменьшение указателя памяти на 2 и копирование GPR в ячейку памяти
MOV	[Rw], [Rw]	2	Копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией)
MOV	[Rw +], [Rw]	2	Копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией) с последующим увеличением указателя памяти на 2
MOV	[Rw], [Rw+]	2	Копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией) с последующим увеличением указателя памяти (копируемой ячейки) на 2
MOV	Rw, [Rw + #data16]	4	Копирование ячейки памяти (с косвенной адресацией с суммированием с константой) в GPR
MOV	[Rw + #data16], Rw	4	Копирование GPR в ячейку памяти (с косвенной адресацией с суммированием с константой)
MOV	[Rw], mem	4	Копирование ячейки памяти в ячейку памяти (с косвенной адресацией)
MOV	mem, [Rw]	4	Копирование ячейки памяти (с косвенной адресацией) в ячейку памяти
MOV	reg, mem	4	Копирование ячейки памяти в регистр
MOV	mem, reg	4	Копирование регистра в ячейку памяти
MOVB	Rb, Rb	2	Побайтовое копирование GPR в GPR
MOVB	Rb, #data4	2	Побайтовое копирование непосредственного операнда в GPR
MOVB	reg, #data8	4	Побайтовое копирование непосредственного операнда в регистр
MOVB	Rb, [Rw]	2	Побайтовое копирование ячейки памяти (с косвенной адресацией) в GPR
MOVB	Rb, [Rw+]	2	Побайтовое копирование ячейки памяти (с косвенной адресацией) в GPR с последующим увеличением указателя памяти на 1
MOVB	[Rw], Rb	2	Побайтовое копирование GPR в ячейку памяти

Окончание таблицы 8.10

1	2	3	4
MOVB	[–Rw], Rb	2	Уменьшение указателя памяти на 1 и побайтовое копирование GPR в ячейку памяти
MOVB	[Rw], [Rw]	2	Побайтовое копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией)
MOVB	[Rw+], [Rw]	2	Побайтовое копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией) с последующим увеличением указателя памяти на 1
MOVB	[Rw], [Rw+]	2	Побайтовое копирование ячейки памяти (с косвенной адресацией) в ячейку памяти (с косвенной адресацией) с последующим увеличением указателя памяти (копируемой ячейки) на 1
MOVB	Rb, [Rw + #data16]	4	Побайтовое копирование ячейки памяти (с косвенной адресацией суммированием с константой) в GPR
MOVB	[Rw + #data16], Rb	4	Побайтовое копирование GPR в ячейку памяти (с косвенной адресацией с суммированием с константой)
MOVB	[Rw], mem	4	Побайтовое копирование ячейки памяти в ячейку памяти (с косвенной адресацией)
MOVB	mem, [Rw]	4	Побайтовое копирование ячейки памяти (с косвенной адресацией) в ячейку памяти
MOVB	reg, mem	4	Побайтовое копирование ячейки памяти в регистр
MOVB	mem, reg	4	Побайтовое копирование регистра в ячейку памяти
MOVBS	Rw, Rb	2	Копирование GPR со знаковым расширением в GPR
MOVBS	reg, mem	4	Копирование ячейки памяти со знаковым расширением в регистр
MOVBS	mem, reg	4	Копирование в регистр со знаковым расширением в ячейку памяти
MOVBZ	Rw, Rb	2	Побайтовое копирование GPR с нулевым расширением в GPR
MOVBZ	reg, mem	4	Побайтовое копирование ячейки памяти с нулевым расширением в регистр
MOVBZ	mem, reg	4	Побайтовое копирование регистра с нулевым расширением в ячейку памяти

Таблица 8.11 – Команды перехода и вызова

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
JMPA	cc, caddr	4	Абсолютный переход по условию
JMPI	cc, [Rw]	2	Косвенный переход по условию
JMPR	cc, rel	2	Относительный переход по условию

Окончание таблицы 8.11

1	2	3	4
JMPS1	seg, caddr	4	Абсолютный переход на сегмент программы
JB	bitaddr, rel	4	Относительный переход, если бит установлен
JBC	bitaddr, rel	4	Относительный переход и сброс бита, если бит установлен
JNB	bitaddr, rel	4	Относительный переход, если бит сброшен
JNBS	bitaddr, rel	4	Относительный переход и установка бита, если бит сброшен
CALLA	cc, caddr	4	Абсолютный вызов подпрограммы по условию
CALLI	cc, [Rw]	2	Косвенный вызов подпрограммы по условию
CALLR	rel	2	Относительный вызов подпрограммы
CALLS	seg, caddr	4	Абсолютный вызов подпрограммы в любом сегменте программы
PCALL	reg, caddr	4	Загрузка регистра в системный стек и абсолютный вызов подпрограммы
TRAP	#trap7	2	Вызов подпрограммы прерывания через непосредственный номер прерывания

Таблица 8.12 – Команды системного стека

Мнемокод команды	Операнды	Размер, байт	Описание команды
POP	reg	2	Извлечение значения из стека в регистр
PUSH	reg	2	Размещение в стеке значения регистра
SCXT	reg, #data16	4	Размещение в стеке значения регистра и загрузка в регистр непосредственного операнда
SCXT	reg, mem	4	Размещение в стеке значения регистра и загрузка в регистр ячейки памяти

Таблица 8.13 – Команды возврата

Мнемокод команды	Операнды	Размер, байт	Описание команды
RET	–	2	Возврат из внутрисегментной подпрограммы
RETS	–	2	Возврат из межсегментной подпрограммы
RETP	reg	2	Возврат из внутрисегментной подпрограммы и извлечение значения из стека в регистр
RETI	–	2	Возврат из подпрограммы обработки прерывания

Таблица 8.14 – Команды управления системой

Мнемокод команды	Операнды	Размер, байт	Описание команды
1	2	3	4
SRST	–	4	Программный сброс
IDLE	–	4	Режим Idle пониженного энергопотребления микроконтроллера, при котором работает его внутренняя периферия
PWRDN	–	4	Полная остановка микроконтроллера, в том числе внутренней периферии (полагается, что на выводе NMI# поддерживается низкий уровень)

Окончание таблицы 8.14

1	2	3	4
SRVWDT	–	4	Обращение к сторожевому таймеру WDT
DISWDT	–	4	Запрещение работы сторожевого таймера WDT
EINIT	–	4	Сигнализирование на выводе RSTOUT о завершении инициализации
ATOMIC	#irang2	2	Временное запрещение обработки запросов на прерывание
EXTR	#irang2	2	Переадресация регистров
EXTP	Rw, #irang2	2	Страничная переадресация
EXTP	#page10, #irang2	4	Страничная переадресация
EXTPR	Rw, #irang2	2	Страничная переадресация и переадресация регистров
EXTPR	#page10, #irang2	4	Страничная переадресация и переадресация регистров
EXTS	Rw, #irang2	2	Сегментная переадресация
EXTS	#seg8, #irang2	4	Сегментная переадресация
EXTSR	Rw, #irang2	2	Сегментная переадресация и переадресация регистров
EXTSR	#seg8, #irang2	4	Сегментная переадресация и переадресация регистров

Таблица 8.15 – Описание вспомогательной команды

Мнемокод команды	Операнды	Размер, байт	Описание команды
NOP	–		Нет операции

8.2 Коды команд

Список кодов команд

Коды команд показаны в таблицах 8.16 – 8.19.

1 Команды кодируются посредством добавочных битов в поле операнда команды.

$x0_H – x7_H$: Rw, #data 3 или Rb, #data3

$x8_H – xB_H$: Rw, [Rw] или Rw, [Rw]

$xC_H – xF_H$: Rw, [Rw+] или Rw, [Rw+]

Для этих команд для косвенной адресации могут использоваться только регистры R0, R1, R2, R3.

2 Команды кодируются посредством добавочных битов в поле операнда команды.

$00xx.xxxx_B$: EXTS или ATOMIC

$01xx.xxxx_B$: EXTP

$10xx.xxxx_B$: EXTSR или EXTR

$11xx.xxxx_B$: EXTPR

Команды JMPR

Код состояния, тестируемый для команд JMPR, определяется кодом операции.

Для нескольких вариантов кодов состояния существует два альтернативных варианта мнемонического кода.

Команды BCLR и BSET

Позиция бита, который должен быть установлен или сброшен, определяется кодом операции. Операнд bitoff.n (n = от 0 до 15) указывает на конкретный бит в пределах бит-адресуемого слова.

Неопределенные команды

Если ЦПУ декодирует один из неопределенных кодов операций («----»), возникает аппаратное прерывание.

Таблица 8.16 – Коды команд 00_H – $3F_H$

Код (hex)	Байт	Мнемокод	Операнды	Код (hex)	Байт	Мнемокод	Операнды
00	2	ADD	Rw, Rw	20	2	SUB	Rw, Rw
01	2	ADDB	Rb, Rb	21	2	SUBB	Rb, Rb
02	4	ADD	reg, mem	22	4	SUB	reg, mem
03	4	ADDB	reg, mem	23	4	SUBB	reg, mem
04	4	ADD	mem, reg	24	4	SUB	mem, reg
05	4	ADDB	mem, reg	25	4	SUBB	mem, reg
06	4	ADD	reg, #data16	26	4	SUB	reg, #data16
07	4	ADDB	reg, #data8	27	4	SUBB	reg, #data8
08	2	ADD	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	28	2	SUB	Rw, [Rw+] или Rw, [Rw] или Rw, #data3
09	2	ADDB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	29	2	SUBB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3
0A	4	BFLDL	bitoff, #mask8, #data8	2A	4	BCMP	bitaddr, bitaddr
0B	2	MUL	Rw, Rw	2B	2	PRIOR	Rw, Rw
0C	2	ROL	Rw, Rw	2C	2	ROR	Rw, Rw
0D	2	JMPR	cc_UC, rel	2D	2	JMPR	cc_EQ, rel или cc_Z, rel
0E	2	BCLR	bitoff.0	2E	2	BCLR	bitoff.2
0F	2	BSET	bitoff.0	2F	2	BSET	bitoff.2
10	2	ADDC	Rw, Rw	30	2	SUBC	Rw, Rw
11	2	ADDCB	Rb, Rb	31	2	SUBCB	Rb, Rb
12	4	ADDC	reg, mem	32	4	SUBC	reg, mem
13	4	ADDCB	reg, mem	33	4	SUBCB	reg, mem
14	4	ADDC	mem, reg	34	4	SUBC	Mem, reg
15	4	ADDCB	mem, reg	35	4	SUBCB	Mem, reg
16	4	ADDC	reg, #data16	36	4	SUBC	reg, #data16
17	4	ADDCB	reg, #data8	37	4	SUBCB	reg, #data8
18	2	ADDC	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	38	2	SUBC	Rw, [Rw+] или Rw, [Rw] или Rw, #data3
19	2	ADDCB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	39	2	SUBCB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3
1A	4	BFLDH	bitoff, #mask8, #data8	3A	4	BMOVN	bitaddr, bitaddr
1B	2	MULU	Rw, Rw	3B	–	–	–
1C	2	ROL	Rw, #data4	3C	2	ROR	Rw, #data4
1D	2	JMPR	cc_NET, rel	3D	2	JMPR	cc_NE, rel или cc_NZ
1E	2	BCLR	bitoff.1	3E	2	BCLR	bitoff.3
1F	2	BSET	bitoff.1	3F	2	BSET	bitoff.3

Таблица 8.17 – Коды команд 40_H – $7F_H$

Код (hex)	Байт	Мнемокод	Операнды	Код (hex)	Байт	Мнемокод	Операнды
40	2	CMP	Rw, Rw	60	2	AND	Rw, Rw
41	2	CMPB	Rb, Rb	61	2	ANDB	Rb, Rb
42	4	CMP	reg, mem	62	4	AND	reg, mem
43	4	CMPB	reg, mem	63	4	ANDB	reg, mem
44	–	–	–	64	4	AND	mem, reg
45	–	–	–	65	4	ANDB	mem, reg
46	4	CMP	reg, #data16	66	4	AND	reg, #data16
47	4	CMPB	reg, #data8	67	4	ANDB	reg, #data8
48	2	CMP	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	68	2	AND	Rw, [Rw+] или Rw, [Rw] или Rw, #data3
49	2	CMPB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	69	2	ANDB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3
4A	4	BMOV	bitaddr, bitaddr	6A	4	BAND	bitaddr, bitaddr
4B	2	DIV	Rw	6B	2	DIVL	Rw
4C	2	SHL	Rw, Rw	6C	2	SHR	Rw, Rw
4D	2	JMPR	cc_V, rel	6D	2	JMPR	cc_N, rel
4E	2	BCLR	Bitoff.4	6E	2	BCLR	bitoff.6
4F	2	BSET	Bitoff.4	6F	2	BSET	bitoff.6
50	2	XOR	Rw, Rw	70	2	OR	Rw, Rw
51	2	XORB	Rb, Rb	71	2	ORB	Rb, Rb
52	4	XOR	reg, mem	72	4	OR	reg, mem
53	4	XORB	reg, mem	73	4	ORB	reg, mem
54	4	XOR	mem, reg	74	4	OR	mem, reg
55	4	XORB	mem, reg	75	4	ORB	mem, reg
56	4	XOR	reg, #data16	76	4	OR	reg, #data16
57	4	XORB	reg, #data8	77	4	ORB	reg, #data8
58	2	XOR	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	78	2	OR	Rw, [Rw+] или Rw, [Rw] или Rw, #data3
59	2	XORB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	79	2	ORB	Rb, [Rw +] или Rb, [Rw] или Rb, #data3
5A	4	BOR	bitaddr, bitaddr	7A	4	BXOR	bitaddr, bitaddr
5B	2	DIV	Rw	7B	2	DIVLU	
5C	2	SHL	Rw, #data4	7C	2	SHR	Rw, #data4
5D	2	JMPR	cc_NV, rel	7D	2	JMPR	cc_NN, rel
5E	2	BCLR	Bitoff.5	7E	2	BCLR	bitoff.7
5F	2	BSET	Bitoff.5	7F	2	BSET	bitoff.7

Таблица 8.18 – Коды команд 80_H – BF_H

Код (hex)	Байт	Мнемокод	Операнды	Код (hex)	Байт	Мнемокод	Операнды
80	2	CMPI1	Rw, #data4	A0	2	CMPD1	Rw, #data4
81	2	NEG	Rw	A1	2	NEGB	Rb
82	4	CMPI1	Rw, mem	A2	4	CMPD1	Rw, mem
83	4	CoXXX	xx	A3	4	CoXXX	xx
84	–	MOV	[Rw], mem	A4	4	MOVB	[Rw], mem
85	–	–	–	A5	4	DISWDT	
86	4	CMPI1	Rw, #data16	A6	4	CMPID	Rw, #data16
87	4	IDLE		A7	4	SRVWDT	
88	2	MOV	[–Rw], Rw	A8	2	MOV	Rw, [Rw]
89	2	MOVB	[–Rw], Rb	A9	2	MOVB	Rb, [Rw]
8A	4	JB	bitaddr, rel	AA	4	JBC	bitaddr, rel
8B	2	–	–	AB	2	CALLI	cc, [Rw]
8C	2	–	–	AC	2	ASHR	Rw, Rw
8D	2	JMPR	cc_C, rel или cc_ULT, rel	AD	2	JMPR	cc_SGT, rel
8E	2	BCLR	Bitoff.8	AE	2	BCLR	bitoff.10
8F	2	BSET	Bitoff.8	AF	2	BSET	bitoff.10
90	2	CMPI2	Rw, #data4	B0	2	CMPD2	Rw, #data4
91	2	CPL	Rw	B1	2	CPLB	Rb
92	4	CMPI2	Rw, mem	B2	4	CMPD2	Rw, mem
93	4	–	–	B3	4	–	–
94	4	MOV	mem, [Rw]	B4	4	MOVB	mem, [Rw]
95	4	–	–	B5	4	EINIT	–
96	4	CMPI2	Rw, #data16	B6	4	CMPD2	Rw, #data16
97	4	PWRDN		B7	4	SRST	
98	2	MOV	Rw, [Rw+]	B8	2	MOV	[Rw], Rw
99	2	MOVB	Rb, [Rw+]	B9	2	MOVB	[Rw], Rb
9A	4	JNB	bitaddr, rel	BA	4	JNBS	bitaddr, rel
9B	2	TRAP	#trap7	BB	2	CALLR	rel
9C	2	JMPI	cc, [Rw]	BC	2	ASHR	Rw, #data4
9D	2	JMPR	cc_NC, rel или cc_UGE, rel	BD	2	JMPR	cc_SLE, rel
9E	2	BCLR	Bitoff.9	BE	2	BCLR	bitoff.11
9F	2	BSET	Bitoff.9	BF	2	BSET	bitoff.11

Таблица 8.19 – Коды команд $C0_H$ – FF_H

Код (hex)	Байт	Мнемокод	Операнды	Код (hex)	Байт	Мнемокод	Операнды
C0	2	MOVBZ	Rw, Rb	E0	2	MOV	Rw, #data4
C1	2	–	–	E1	2	MOVB	Rb, #data4
C2	4	MOVBZ	reg, mem	E2	4	PCALL	reg, caddr
C3	4	–	–	E3	4	–	–
C4	–	MOV	[Rw + #data16], Rw	E4	4	MOVB	[Rw + #data16], Rb
C5	–	MOVBZ	mem, reg	E5	4	–	–
C6	4	SCXT	reg, #data16	E6	4	MOV	reg, #data16
C7	4	–		E7	4	MOVB	reg, #data8
C8	2	MOV	[Rw], [Rw]	E8	2	MOV	[Rw], [Rw+]
C9	2	MOVB	[Rw], [Rw]	E9	2	MOVB	[Rw], [Rw+]
CA	4	CALLA	cc, addr	EA	4	JMPA	cc, caddr
CB	2	RET	–	EB	2	RETP	reg
CC	2	NOP	–	EC	2	PUSH	reg
CD	2	JMPR	cc_SLT, rel	ED	2	JMPR	cc_UGT, rel
CE	2	BCLR	Bitoff.12	EE	2	BCLR	bitoff.14
CF	2	BSET	Bitoff.12	EF	2	BSET	bitoff.14
D0	2	MOVBS	Rw, Rb	F0	2	MOV	Rw, Rw
D1	2	ATOMIC / EXTR	#irang2	F1	2	MOVB	Rb, Rb
D2	4	MOVBS	reg, mem	F2	4	MOV	reg, mem
D3	4	–	–	F3	4	MOVB	reg, mem
D4	4	MOV	Rw, [Rw + #data16]	F4	4	MOVB	Rb, [Rw + #data16]
D5	4	MOVBS	mem, reg	F5	4	–	–
D6	4	SCXT	reg, mem	F6	4	MOV	mem, reg
D7	4	EXTP(R), EXTS(R)	#pag10, #irang2 #seg8, #irang2	F7	4	MOVB	mem, reg
D8	2	MOV	[Rw+], [Rw]	F8	2	–	–
D9	2	MOVB	[Rw+], [Rw]	F9	2	–	–
DA	4	CALLS	seg, caddr	FA	4	JMPS	seg, caddr
DB	2	RETS		FB	2	RETI	
DC	2	EXTP(R), EXTS(R)	Rw, #irang2	FC	2	POP	reg
DD	2	JMPR	cc_SGE, rel	FD	2	JMPR	cc_ULE, rel
DE	2	BCLR	Bitoff.13	FE	2	BCLR	bitoff.15
DF	2	BSET	Bitoff.13	FF	2	BSET	bitoff.15

8.3 Описание команд

Детальное описание команд микроконтроллера 1887ВЕ3Т находится в приложении Е, где команды расположены по списку в алфавитном порядке.

Описание команды включает в себя следующие элементы.

Имя команды

Уникальный мнемонический код команды.

Синтаксис

Определяет мнемонический код команды и необходимые операнды, взаимодействие которых указывается в действии. В инструкциях без операторов, а так же в инструкциях с одним, двумя или тремя операторами, операторы должны быть отделены друг от друга фигурной скобкой:

MNEMONIC {op1{,op2{,op3}}}

Синтаксис операндов команды зависит от выбранного режима адресации. Все доступные режимы адресации представлены в конце описания каждой команды.

Действие

Представляет логическое описание взаимодействия операторов команды в символьном виде или в виде конструкции языка программирования высокого уровня.

Для представления операторов перемещения, арифметических и логических операторов применяются следующие символы:

(opX) ← (opY) – (opY) копируется в (opX);
(opX) + (opY) – (opX) прибавляется к (opY);
(opX) - (opY) – (opY) вычитается из (opX);
(opX) * (opY) – (opX) умножается на (opY);
(opX) / (opY) – (opX) делится на (opY);
(opX) ^ (opY) – операция побитового логического «И» над операндами;
(opX) v (opY) – операция побитового логического «ИЛИ» над перандами;

(opX) ⊕ (opY) – операция побитового «ИСКЛЮЧАЮЩЕГО ИЛИ» над операндами;

(opX) ⇔ (opY) – (opX) сравнивается с (opY);
(opX) mod (opY) – вычисляется остаток от деления (opX) на (opY);
¬ (opX) – побитовая инверсия (opX).

Круглые скобки указывают на метод адресации операндов:

opX – непосредственные данные;
(opX) – содержимое opX;
(opX[n]) – содержимое бита n операнда opX;
((opX)) – значение, взятое по адресу, равному содержимому opX (т.е. содержимое opX является указателем требуемого значения).

В описании команд используются следующие сокращения:

CP – контекстный указатель;
CSP – указатель сегмента кода;
IP – указатель команд;
MD – регистр умножения/деления (32-разрядный, состоит из MDH и MDL);
MDL, MDH – младший и старший регистры умножения/деления (по 16-разрядов);
PSW – слово состояния процессора;
SP – указатель системного стека;
SYSCON – регистр конфигурации системы;

C – флаг переноса в регистре PSW;
V – флаг переполнения в регистре PSW;
SGTDIS – бит запрета сегментации в регистре SYSCON;
count – временная переменная, используемая в качестве счетчика;
tmp – временная переменная для промежуточных вычислений.

Тип данных

Определяет тип данных в зависимости от формата команды.

Возможны варианты:

BIT – Бит
BYTE – Байт (8 бит)
WORD – Слово (16 бит)

Менять тип данных могут только те команды, которые увеличивают байт данных до слова данных. Следует помнить, что тип данных BYTE не предусматривает доступа к указателям косвенных адресов или системного стека. Это возможно только для WORD.

Описание

Описание действий, выполняемых командой.

Код состояния

Код состояния показывает, что команда выполняется, если выполнено необходимое условие и пропускается, если условие не выполнено. В таблице 8.20 представлены возможные коды состояний (условий перехода), которые могут использоваться командами вызова подпрограмм и переходов.

Таблица 8.20 – Коды условий перехода (cc)

Мнемокод	Условие	Описание	Номер
cc_UC	1 = 1	Безусловный переход	0_H
cc_Z	Z = 1	Если результат равен нулю	2_H
cc_NZ	Z = 0	Если результат не равен нулю	3_H
cc_V	V = 1	Если произошло переполнение во время выполнения команды	4_H
cc_NV	V = 0	Если не произошло переполнения во время выполнения команды	5_H
cc_N	N = 1	Если результат отрицательный	6_H
cc_NN	N = 0	Если результат неотрицательный	7_H
cc_C	C = 1	Если произошел перенос во время выполнения инструкции	8_H
cc_NC	C = 0	Если не произошел перенос во время выполнения инструкции	9_H
cc_EQ	Z = 1	Если сравниваемые операнды эквивалентны	2_H
cc_NE	Z = 0	Если сравниваемые операнды не эквивалентны	3_H
cc ULT	C = 1	Меньше (без знака)	8_H
cc_ULE	(Z v C) = 1	Меньше или равно (без знака)	F_H
cc_UGE	C = 0	Больше или равно (без знака)	9_H
cc_UGT	(Z v C) = 0	Больше (без знака)	E_H
cc_SLT	(N ⊕ V) = 1	Меньше (со знаком)	C_H
cc_SLE	(Z v (N ⊕ V)) = 1	Меньше или равно (со знаком)	B_H
cc_SGE	(N ⊕ V) = 0	Больше или равно (со знаком)	D_H
cc_SGT	(Z v (N ⊕ V)) = 0	Больше (со знаком)	A_H
cc_NET	(Z v E) = 0	Если сравниваемые операнды не эквивалентны и один из операндов не равен наименьшему отрицательному числу	1_H

Флаги состояния

В данной части описания команды отражается состояние флагов N, C, V, Z и E регистра PSW после выполнения команды, за исключением случаев, когда регистр PSW сам является операндом-приемником для выполняемой команды.

Результирующее состояние флагов представлено символами:

«*» – Флаг устанавливается по следующим стандартным правилам:

- N = 1 – Значащий бит результата установлен.
- N = 0 – Значащий бит результата не установлен.
- C = 1 – Произошел перенос во время операции.
- C = 0 – Во время операции не было переноса.
- V = 1 – Во время операции произошло арифметическое переполнение.
- V = 0 – Во время операции не произошло арифметического переполнения.
- Z = 1 – Результат равен нулю.
- Z = 0 – Результат не равен нулю.
- E = 1 – Операнд-источник является наименьшим отрицательным числом (8000_H для данных типа WORD и 80_H для данных типа BYTE).
- E = 0 – Операнд-источник не является наименьшим отрицательным числом.

«S» – Флаг устанавливается по правилам, которые отличаются от стандартных.

«-» – Выполнение инструкции не влияет на флаг.

«0» – Значение флага всегда обнуляется при выполнении инструкции.

«NOR» – Флаг содержит инверсию результата выполнения операции логического «ИЛИ» над содержимым двух битовых операндов инструкции.

«AND» – Флаг содержит результат выполнения операции логического «И» над содержимым двух битовых операндов инструкции.

«OR» – Флаг содержит результат выполнения операции логического «ИЛИ» над содержимым двух битовых операндов инструкции.

«XOR» – Флаг содержит результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым двух битовых операндов инструкции.

«B» – Флаг содержит значение битового операнда до выполнения инструкции.

«B#» – Флаг содержит инверсию значения битового операнда до выполнения инструкции.

Примечание – Если регистр PSW был определен как операнд-приемник для выполнения команды, флаги состояния не могут интерпретироваться как описано выше, поскольку состояние регистра PSW изменяется в зависимости от формата данных следующим образом:

- для операций типа WORD регистр PSW перезаписывается результатом типа WORD;
- для операций типа BYTE не адресуемый байт очищается, а адресуемый (в который производиться запись) – перезаписывается;
- при операциях типа BIT (или операций с битовыми полями) перезаписываются только определенные биты;
- если флаги состояния не были выбраны как биты для перезаписи, то они остаются без изменений и, следовательно, находятся в состоянии, которое явилось результатом выполнения предыдущей команды.

В любом случае, если регистр PSW был определен как операнд-приемник для выполнения команды, флаги состояния не могут являться достоверным источником информации о результате завершения выполнения команды.

Режимы адресации

Режим адресации определяется кодом команды. В то же время имеются некоторые арифметические и логические команды, для которых режим адресации определяется не кодом команды, а отдельными битами в поле операнда.

Выбор режима адресации основывается на трех составляющих.

Мнемонический код

Отражает допустимые операнды для соответствующей команды.

Формат

Определяет формат команды в том виде, в каком она представляется на ассемблере. На рисунке 8.1 показано соотношение между форматом команды и соответствующей ей внутренней организацией (N = полубайт = 4 бита).

Для описания формата команд используются следующие символы:

$00_H – FF_H$	– коды команд;
0, 1	– константы;
:….	– каждый из четырех символов после «:» представляет собой один бит;
:..ii	– 2-битовый адрес GPR (Rwi);
SS	– номер кода сегмента;
:..##	– 2-битовая непосредственная константа (#irang2);
:..###	– 3-битовая непосредственная константа (#data2);
…#:#	– 5-битовая непосредственная константа (#data5);
c	– 4-битовый номер условия перехода cc;
n	– 4-битовый адрес GPR Rwn или Rbn;
m	– 4-битовый адрес GPR Rwm или Rbm;
q	– 4-битовый номер разряда-источника 2-байтового операнда QQ;
z	– 4-битовый номер разряда-приемника 2-байтового операнда ZZ;
#	– 4-битовая непосредственная константа (#data4);
t:ttt0	– 7-битовый непосредственный номер вектора прерывания (#trap7);
QQ	– 8-битовый адрес разряда-источника (bitoff);
rr	– 8-битовое смещение для относительных переходов (rel);
ZZ	– 8-битовый адрес бита-приемника (bitoff);
##	– 8-битовая непосредственная константа (#data8);
## xx	– 8-битовая непосредственная константа представлена как #data16, байт xx – не значимый;
@@	– 8-битовая непосредственная константа (#mask8);
MMMM	– 16-битовый адрес mem или caddr; младший байт, старший байт;
####	– 16-битовая непосредственная константа #data16; младший байт, старший байт.

Число байт

Все команды микроконтроллера 1887ВЕ3Т двух и четырех байтные.

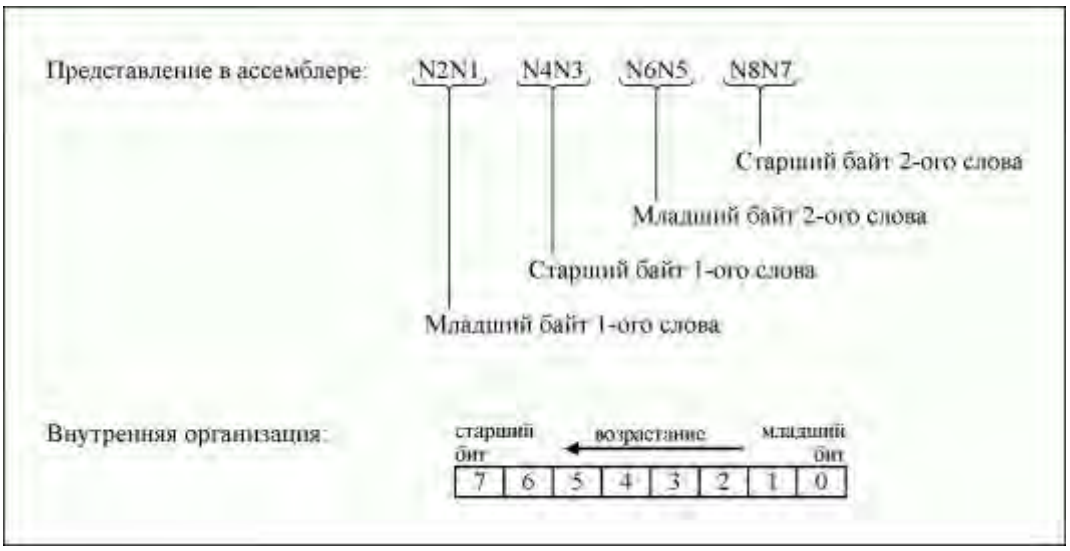


Рисунок 8.1 – Формат команды микроконтроллера 1887ВЕ3Т

9 Параллельные порты

Этот раздел подробно описывает работу параллельных портов 1887ВЕ3Т. Для передачи или приема параллельных данных и одиночных внешних сигналов управления в микроконтроллере имеется 103 линии параллельного ввода-вывода. Архитектура портов содержит три 16-разрядных порта ввода-вывода: порт P0, порт P1, порт P3; один 16-разрядный входной порт: порт P5; четыре 8-разрядных порта ввода-вывода: порт P2, порт P4, порт P6, порт P9; один 7-разрядный порт ввода-вывода: порт P7.

Эти линии портов могут быть использованы для команд ввода-вывода основного назначения под программным управлением, или могут быть использованы для интегрированной периферии микроконтроллера, или для контроллера внешней шины.

Все линии портов являются адресуемыми побитно, и все линии ввода-вывода индивидуально программируются на вход или выход с помощью регистров управления (за исключением порта P5). Порты ввода-вывода являются двунаправленными портами, которые можно переключать в состояние высокого сопротивления при настройке на вход. Выходные драйверы шести портов ввода-вывода: порт P2, порт P3, порт P4, порт P6, порт P7, порт P9, – могут быть сконфигурированы для работы в режиме двух комплементарных выходных транзисторов (двухтактный каскад «push/pull») или для работы в режиме с открытым стоком с помощью контрольных регистров. Логический уровень на входе измеряется во входном триггере один раз за такт, при этом не имеет значения конфигурация порта (на вход или на выход).

Если произвести действие записи в порт, настроенный на вход, то данное значение будет записано в выходной триггер порта, однако команда чтения порта перезаписывает значение на входе в триггер порта. Команда чтение-изменение-запись читает значение вывода микросхемы, модифицирует его и записывает назад в выходной триггер.

Запись в вывод, сконфигурированный на выход DPx.y = 1_B, приводит к тому, что выходной триггер и вывод микросхемы соединены, поскольку выходной буфер включен. Чтение этого вывода приводит к возврату значения выходного триггера. Действие чтение-изменение-запись читает значение выходного триггера, изменяет его и записывает назад в выходной триггер, поэтому также изменяя уровень на выводе микросхемы.

Большинство функций, использующих ввод-вывод данных, а также функции управления, необходимые для работы микроконтроллера, реализованы в виде альтернативных функций параллельных портов. Однако для некоторых сигналов предназначены независимые выводы. К ним можно отнести входы осциллятора, входы специальных сигналов управления и входы питания микросхемы.

9.1 Режим с открытым стоком

В микросхеме 1887ВЕ3Т часть портов имеют возможность работы в режиме с открытым стоком. В режиме push/pull выходной драйвер имеет транзисторы как на высокой, так и на нижней стороне, таким образом, линия замыкается либо на высокий, либо на низкий уровень напряжения. В режиме с открытым стоком верхний транзистор всегда выключен, и поэтому выходной драйвер может устанавливать только низкий уровень на выходе. При записи единицы в триггер порта нижний транзистор выключается, и выход переходит в состояние высокого сопротивления. Высокий уровень в этом случае должен быть обеспечен внешним устройством. На рисунке 9.1 показана реализация этой функции. Использование этой возможности позволит объединять по несколько портов вместе для обеспечения конфигурации wired-AND, т. е. без дополнительного программного кода можно осуществлять операции и/или для разрешения или запрещения выходного сигнала. На рисунке 9.1 сигнал $\overline{Q}$ – сигнал, приходящий с выхода триггера порта.

Эта функция управляется с помощью специальных регистров управления режимом с открытым стоком ODPx. Регистры ODPx позволяют переопределять отдельные выводы

в этот режим. Если в ODPx.y установлена единица, то в этом случае выбран режим с открытым стоком. Заметим, что все эти регистры расположены в ESFR области.

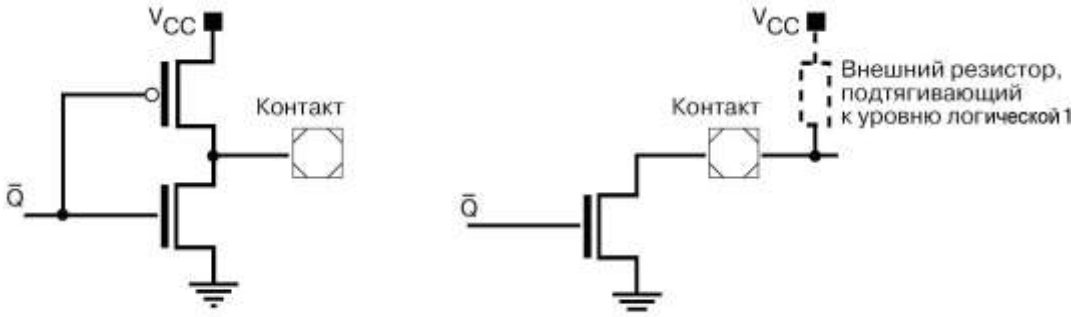


Рисунок 9.1 – Выходные буферы в режиме push/pull и в режиме с открытым стоком, соответственно

9.2 Альтернативные функции портов

Для обеспечения максимальной гибкости при использовании ИС 1887ВЕ3Т, линии портов можно запрограммировать на функции альтернативного ввода или вывода.

При использовании альтернативной функции вывода порта для вывода данных, порт должен быть направлен на выход DPx.y = 1_B, за исключением некоторых сигналов, постоянно использующихся после сброса и настраиваемых автоматически. В ином случае выводы остаются в состоянии высокого сопротивления и не используются в качестве альтернативных выходных функций. В триггер порта необходимо записать единицу, потому что он объединяется с данными альтернативного вывода по функции AND.

Если используется альтернативная функция ввода, то направление вывода порта должно быть запрограммировано на вход DPx.y = 0_B. После сброса направление порта автоматически устанавливается на вход. Если никаких внешних устройств не подключено к выводу порта, то направление порта не влияет на значение в выходном триггере порта. В этом случае, вход порта отражает состояние выходного триггера порта. Таким образом, альтернативная входная функция читает значение, сохраненное в выходном триггере.

Для большинства линий портов, во время использования альтернативной функции ввода или вывода, за установки необходимого направления отвечает пользовательская программа. Это достигается путем установки значения в бите управления направлением вывода порта DPx.y перед началом использования альтернативной функции. Однако имеются линии порта, в которых автоматически устанавливается направление. Например, в режиме мультиплексной внешней шины порта P0, направление переключается несколько раз за один цикл чтения шины. Очевидно, что это невозможно сделать посредством команд. В этих случаях, направление порта меняется автоматически аппаратно. Программное управление функциями порта обеспечивается двумя специальными регистрами ALTSELnPx, n равно 0 или 1.

Все линии портов, не пользующиеся альтернативными функциями, могут быть использованы для линий ввода-вывода основного назначения. При использовании выводов портов для вывода основного назначения, для включения выходных драйверов, необходимо записать инициализирующее значение в регистр порта, во избежание нежелательных передач на выход порта. Это относится как к отдельным выводам, так и к группе выводов.

```
SINGLE BIT
    BSET P4.7                    ; установленный уровень выхода – высокий
    BSET DP4.7                   ; переключение на выходной драйвер
BIT_GROUP:
```

```
BFLDH P4, #24H, #24H        ; установленный уровень выхода – высокий
BFLDH DP4, #24H, #24H       ; переключение на выходной драйвер
```

Примечание – При использовании нескольких пар BSET, для управления большим количеством выводов порта необходимо, чтобы эти пары были независимыми с помощью команд, не ссылающихся на порт.

9.3 Порт P0

Два 8-битных порта P0L, см. рисунок 9.2, таблицу 9.1, и P0H, см. рисунок 9.3, таблицу 9.1, представляют собой две части порта P0. В обе половины порта P0 может производиться запись без изменения значения другой половины. В том случае, когда этот порт используется для функции ввода-вывода основного назначения, направление каждой линии может быть сконфигурировано с помощью регистров направления DP0L, см. рисунок 9.4, таблицу 9.2, и DP0H, см. рисунок 9.5, таблицу 9.2.

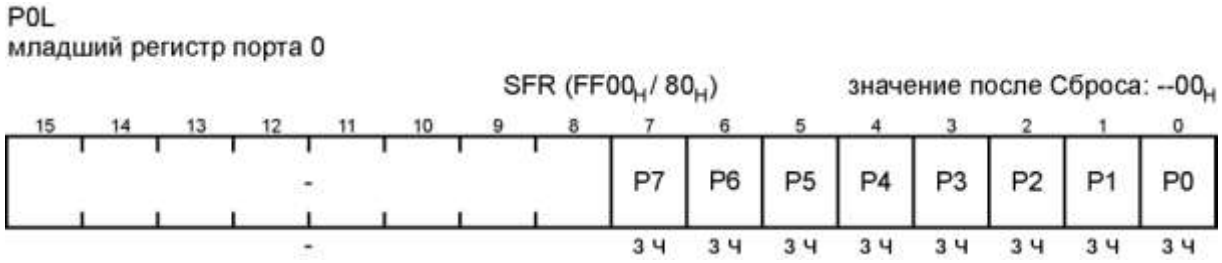


Рисунок 9.2 – Формат регистра P0L

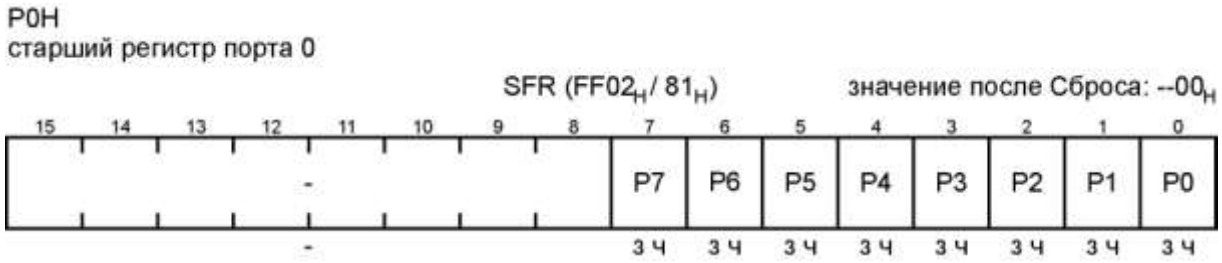


Рисунок 9.3 – Формат регистра P0H

Таблица 9.1 – Функциональное назначение полей регистров P0L и P0H

Поле	Биты	Тип	Описание
P0x.y	7-0	Запись Чтение	Регистр данных P0H/P0L бита y порта P0

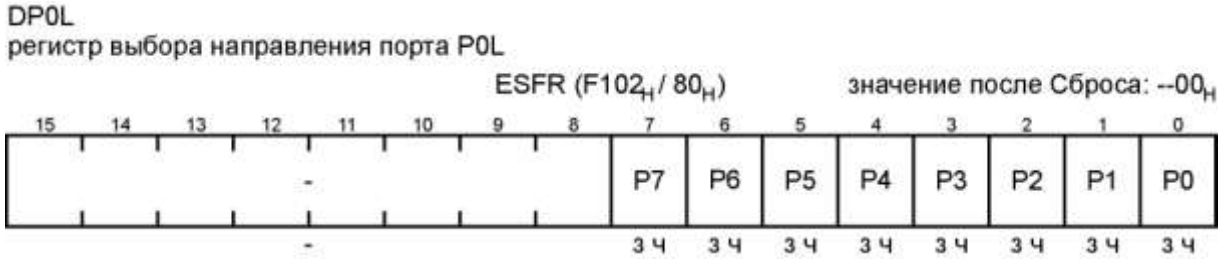


Рисунок 9.4 – Формат регистра DP0L

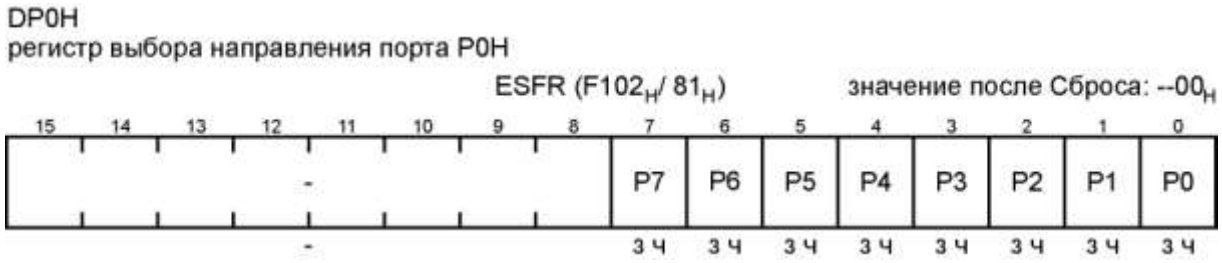


Рисунок 9.5 – Формат регистра DP0H

Таблица 9.2 – Функциональное назначение полей регистров DP0L и DP0H

Поле	Биты	Тип	Описание
DP0x.y	7-0	Чтение Запись	Регистр выбора направления DP0H/DP0L бита y порта P0: 0 Линия порта P0x.y настроена на вход. 1 Линия порта P0x.y настроена на выход.

Альтернативные функции порта P0

При включенной внешней шине порт P0 используется в качестве шины данных или в качестве шины адреса и данных. Заметим, что внешняя 8-битная демультиплексная шина используется только P0L, в то время как P0H остается свободным для ввода-вывода (при условии, если не включен другой режим шины). Во время внешнего доступа с помощью мультиплексной шиной, в порт P0 сначала выводится 16-битный внутрисегментный адрес. Затем порт переключается в режим высокого сопротивления для чтения входных команд или данных. В 8-битном режиме шины данных, необходимо два цикла для доступа к слову, сначала для доступа к младшему байту и потом для доступа к старшему байту слова. Альтернативные функции порта P0 представлены на рисунке 9.6.

Порт P0 также используется для выбора конфигурации системы при старте, см. приложение И. Во время сброса, порт P0 настроен на вход, и каждая линия выставлена на высокий уровень через внутренний pullup. Каждая линия может быть индивидуально подключена к нулевому потенциалу, с помощью внешнего pulldown устройства. Выставляя необходимые линии на низкий уровень напряжения, можно изменять установочные значения.

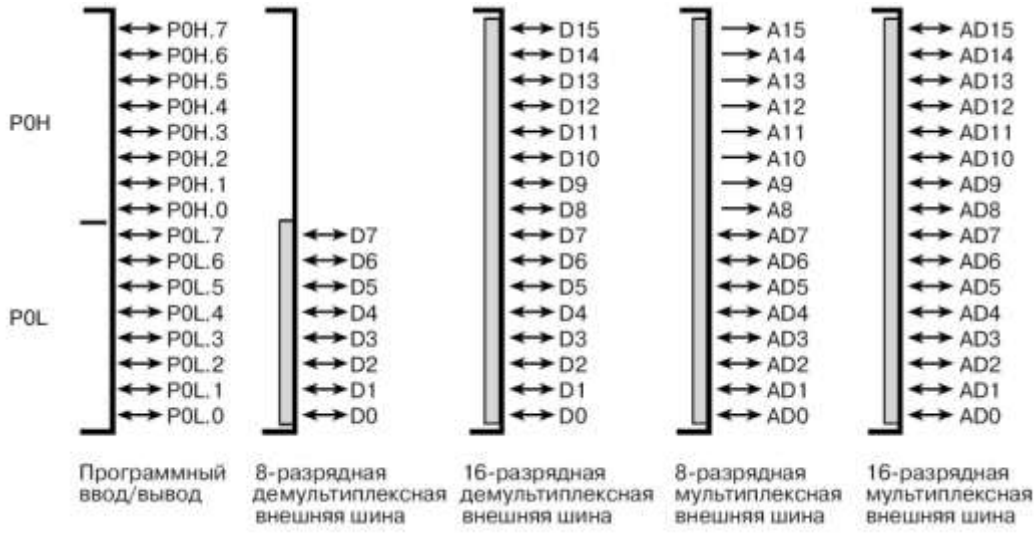
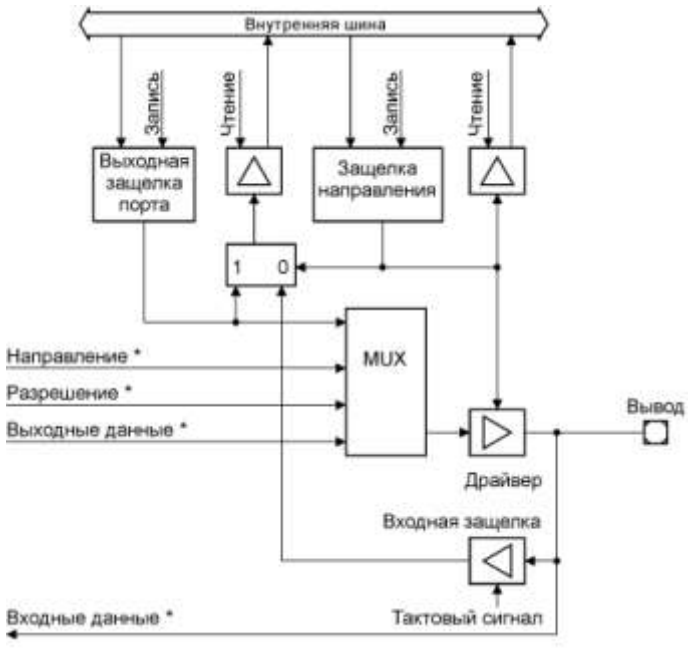


Рисунок 9.6 – Ввод и вывод порта P0 и его альтернативные функции

Внутреннее pullup устройство разработано подобно внешним pulldown резисторам, которые могут использоваться для подачи корректного низкого уровня напряжения. Внешние pulldown резисторы могут оставаться подсоединенными к порту P0 во время нормальной работы микроконтроллера, однако необходимо обращать внимание на то, чтобы их значения не мешали нормальному функционированию порта P0.

После окончания сброса, выбранная конфигурация будет записана в регистр BUSCON.

Когда включен режим внешней шины, направление вывода порта и загрузка данных в выходной триггер порта управляются с помощью блока контроллера шины. Вход порта и выходной триггер не подконтрольны внутренней шине и переключаются на линию, названную альтернативным выводом данных с помощью мультиплексора. Альтернативные данные могут быть либо 16-битным внутрисегментным адресом или 8/16-битными данными. Входные данные порта P0 читаются на линии «Альтернативный вход данных». В то время, когда включен режим внешней шины, не должна производиться запись в выходной триггер порта пользовательской программой, иначе могут иметь место непредсказуемые результаты. Когда внешняя шина отключена, вступает в силу последнее записанное пользователем содержимое регистра направления. На рисунке 9.7 показана структура выводов порта P0.



* Альтернативная функция.

Рисунок 9.7 – Структурная схема выводов порта P0

Режим Adapt

Установка низкого уровня напряжения на входе P0L.1 во время сброса обеспечивает работу в режиме Adapt. В этом режиме контроллер переходит в пассивное состояние. При этом выводы контроллера находятся в состоянии высокого сопротивления. Вывод RSTOUT также находится в состоянии высокого сопротивления. Прекращает работу внутренний осциллятор.

В этом режиме можно смоделировать виртуальное исключение контроллера из системы. Поэтому внешний эмулятор может управлять работой системы, даже в тех случаях, когда микроконтроллер остается на месте. Выход из режима электрической изоляции выполняется после сброса, во время которого необходимо обеспечить P0L.1 = 1.

По умолчанию режим Adapt отключен.

9.4 Порт P1

Два 8-битных порта P1L, см. рисунок 9.8, таблицу 9.3, и P1H, см. рисунок 9.9, таблицу 9.3, представляют собой две части порта P1. В обе половины порта P1 может быть произведена запись без изменения значения другой половины. Если этот порт используется для ввода-вывода основного назначения, направление каждой линии может быть сконфигурировано с помощью регистров направления DP1L, см. рисунок 9.10, таблицу 9.4, и DP1H, см. рисунок 9.11, таблицу 9.4.

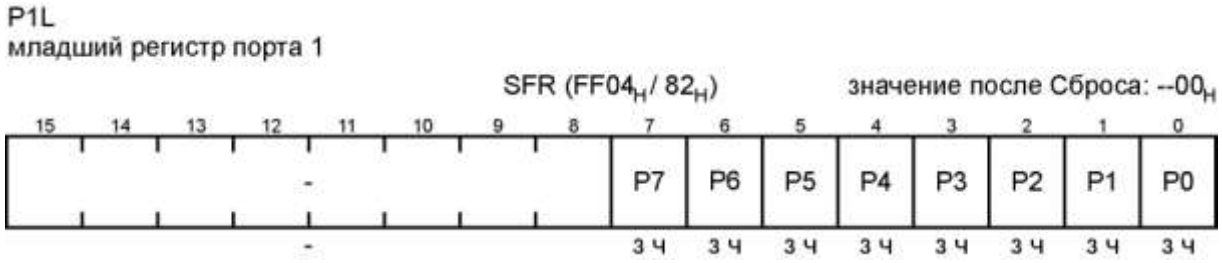


Рисунок 9.8 – Формат регистра P1L

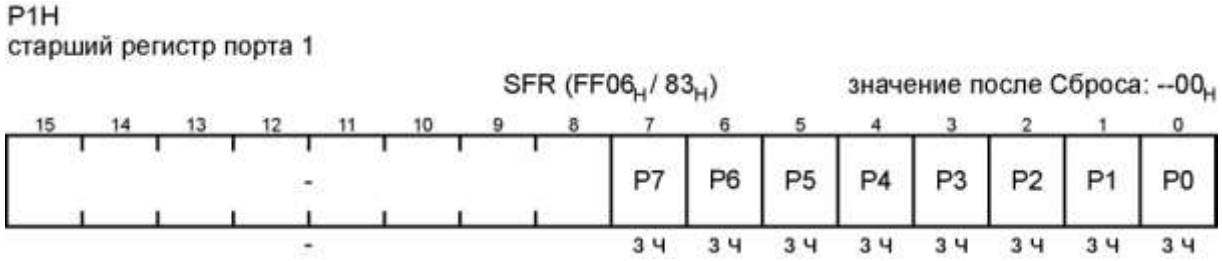


Рисунок 9.9 – Формат регистра P1H

Таблица 9.3 – Функциональное назначение полей регистров P1L и P1H

Поле	Биты	Тип	Описание
P1X.y	7-0	Запись Чтение	Регистр данных P1H и P1L бита y порта P1

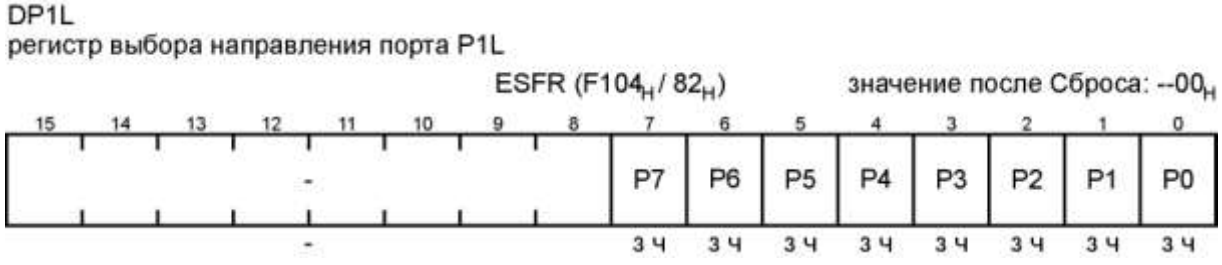


Рисунок 9.10 – Формат регистра DP1L

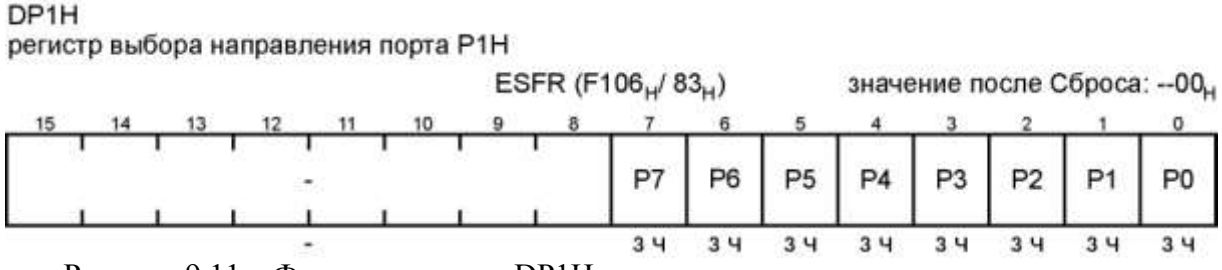


Рисунок 9.11 – Формат регистра DP1H

Таблица 9.4 – Функциональное назначение полей регистров DP1L и DP1H

Поле	Биты	Тип	Описание
DP1X.y	7-0	Чтение Запись	Регистр выбора направления DP1H и DP1L бита y порта P1: 0 Линия порта P1X.y настроена на вход. 1 Линия порта P1X.y настроена на выход.

Альтернативные функции для CAPCOM2 и SSC1 выбираются через регистры альтернативных функций ALTSEL0P1L, см. рисунок 9.12, таблицу 9.5, и ALTSEL0P1H, см. рисунок 9.13, таблицу 9.6.

Рисунок 9.12 – Формат регистра ALTSEL0P1L

Таблица 9.5 – Функциональное назначение полей регистра ALTSEL0P1L

Поле	Биты	Тип	Описание
ALTSEL0P1L.y	7-0	Чтение Запись	Регистр управления альтернативными функциями P1L бита y порта P1: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Рисунок 9.13 – Формат регистра ALTSEL0P1H

Таблица 9.6 – Функциональное назначение полей регистра ALTSEL0P1H

Поле	Биты	Тип	Описание
ALTSEL0P1H.y	7-0	Чтение Запись	Регистр управления альтернативными функциями P1H бита y порта P1: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P1

При включенном режиме демультиплексной внешней шины, порт P1 используется в качестве шины адреса. Заметим, что в режиме демультиплексной шины используется порт P1, в качестве 16-битного порта. В ином случае все 16 линий порта используются для ввода-вывода основного назначения.

Выводы порта P1 и их альтернативные функции показаны на рисунке 9.14, структурная схема вывода порта P1 представлена на рисунке 9.15.

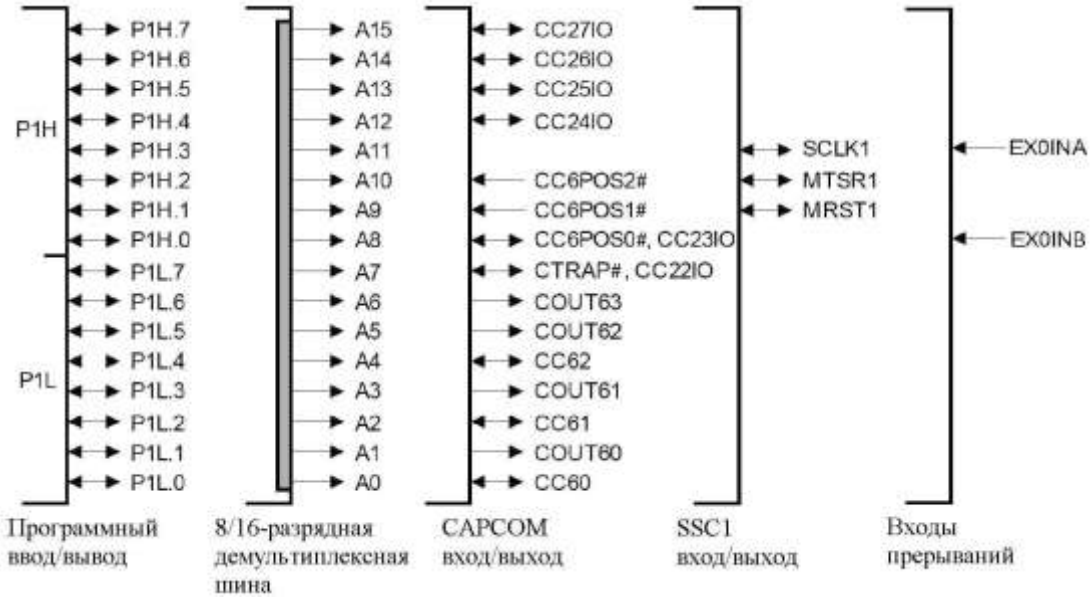
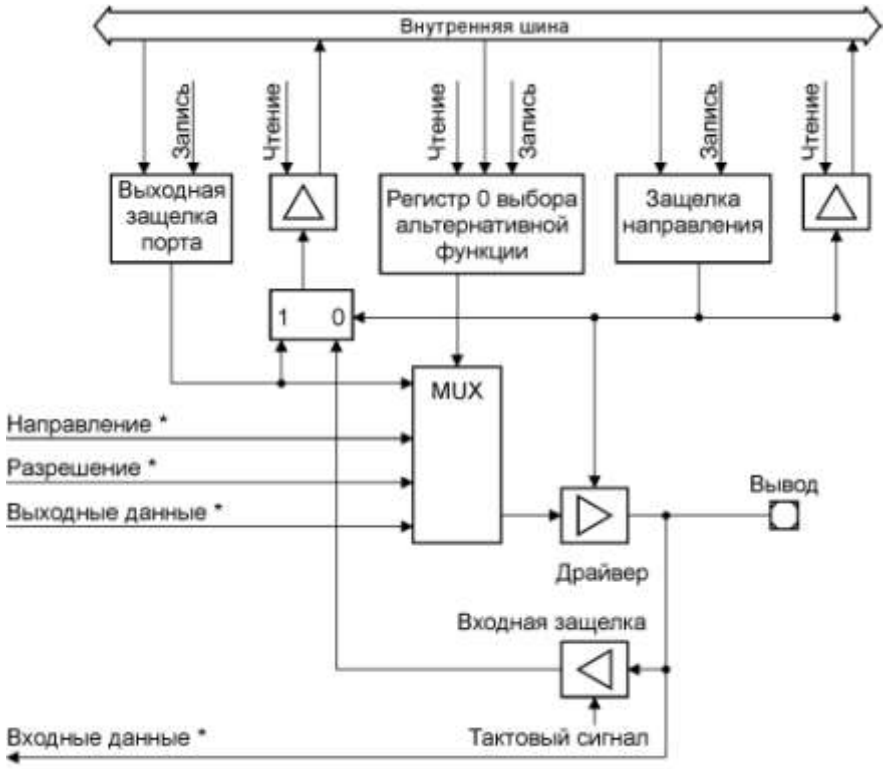


Рисунок 9.14 – Выводы порта P1 и их альтернативные функции



* Альтернативная функция.

Рисунок 9.15 – Структурная схема вывода порта P1

Во время внешнего доступа в режиме демультиплексной шины, в качестве альтернативной функции порт P1 выводит 16-битный адрес внутри сегмента. Во время внешнего доступа в режиме мультиплексной шины, порт P1 не используется и доступен для ввода и вывода основного назначения.

В таблице 9.7 показаны возможные функции выводов порта P1.

Таблица 9.7 – Функции выводов порта P1

Вывод порта P1	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
1	2	3	4	5
P1L.0	Программный вход	P1L.0	EBC не активен и ALTSEL0P1L.P0 = 0	DP1L.P0 = 0
	Программный выход			DP1L.P0 = 1
	Выходная линия адреса A0	EBC	EBC активен	
	Вход модуля CAPCOM6 CC60 (захват)	CAPCOM6	–	DP1L.P0 = 0
	Выход модуля CAPCOM6 CC60 (сравнение)		ALTSEL0P1L.P0 = 1	DP1L.P0 = 1
P1L.1	Программный вход	P1L.1	EBC не активен и ALTSEL0P1L.P1 = 0	DP1L.P1 = 0
	Программный выход			DP1L.P1 = 1
	Выходная линия адреса A1	EBC	EBC активен	
	Выход модуля CAPCOM6 COUT60	CAPCOM6	ALTSEL0P1L.P1 = 1	DP1L.P1 = 1
P1L.2	Программный вход	P1L.2	EBC не активен и ALTSEL0P1L.P2 = 0	DP1L.P2 = 0
	Программный выход			DP1L.P2 = 1
	Выходная линия адреса A2	EBC	EBC активен	
	Вход модуля CAPCOM6 CC61(захват)	CAPCOM6	–	DP1L.P2 = 0
	Выход модуля CAPCOM6 CC61 (сравнение)		ALTSEL0P1L.P2 = 1	DP1L.P2 = 1
P1L.3	Программный вход	P1L.3	EBC не активен и ALTSEL0P1L.P3 = 0	DP1L.P3 = 0
	Программный выход			DP1L.P3 = 1
	Выходная линия адреса A3	EBC	EBC активен	
	Выход модуля CAPCOM6 COUT61	CAPCOM6	ALTSEL0P1L.P3 = 1	DP1L.P3 = 1

Продолжение таблицы 9.7

1	2	3	4	5
P1L.4	Программный вход	P1L.4	EBC не активен и ALTSEL0P1L.P4 = 0	DP1L.P4 = 0
	Программный выход			DP1L.P4 = 1
	Выходная линия адреса A4	EBC	EBC активен	
	Вход модуля CAPCOM6 CC62 (захват)	CAPCOM6	–	DP1L.P4 = 0
	Выход модуля CAPCOM6 CC62 (сравнение)		ALTSEL0P1L.P3 = 1	DP1L.P4 = 1
P1L.5	Программный вход	P1L.5	EBC не активен и ALTSEL0P1L.P5 = 0	DP1L.P5 = 0
	Программный выход			DP1L.P5 = 1
	Выходная линия адреса A5	EBC	EBC активен	
	Выход модуля CAPCOM6 COUT62	CAPCOM6	ALTSEL0P1L.P5 = 1	DP1L.P5 = 1
P1L.6	Программный вход	P1L.6	EBC не активен и ALTSEL0P1L.P6 = 0	DP1L.P6 = 0
	Программный выход			DP1L.P6 = 1
	Выходная линия адреса A6	EBC	EBC активен	
	Выход модуля CAPCOM6 COUT63	CAPCOM6	ALTSEL0P1L.P6 = 1	DP1L.P6 = 1
P1L.7	Программный вход	P1L.7	EBC не активен и ALTSEL0P1L.P7 = 0	DP1L.P7 = 0
	Программный выход			DP1L.P7 = 1
	Выходная линия адреса A7	EBC	EBC активен	
	CC22I вход захвата	CAPCOM2	–	DP1L.P7 = 0
	CC22O выход сравнения		EBC не активен и ALTSEL0P1L.P7 = 1	DP1L.P7 = 1
	Вход прерывания CTRAP#	CAPCOM6	–	DP1L.P7 = 0
P1H.0	Программный вход	P1H.0	EBC не активен и ALTSEL0P1H.P0 = 0	DP1H.P0 = 0
	Программный выход			DP1H.P0 = 1
	Выходная линия адреса A8	EBC	EBC активен	
	CC23I вход захвата	CAPCOM2	–	DP1H.P0 = 0
	CC23O выход сравнения		EBC не активен и ALTSEL0P1H.P0 = 1	DP1H.P0 = 1
	Вход позиции CC6POS0#	CAPCOM6	–	DP1H.P0 = 0
	Вход альтернативного сигнала B «прерывание 0» EX0INB		–	DP1H.P0 = 0

Окончание таблицы 9.7

1	2	3	4	5
P1H.1	Программный вход	P1H.1	EBC не активен и ALTSEL0P1H.P1 = 0	DP1H.P1 = 0
	Программный выход			DP1H.P1 = 1
	Выходная линия адреса A9	EBC	EBC активен	
	SSC1 принимает по входу MRST1 в режиме master	SSC1	–	DP1H.P1 = 0
	SSC1 передача через выход MRST1 в режиме slave		EBC не активен и ALTSEL0P1H.P1 = 1	DP1H.P1 = 1
	Вход позиции CC6POS1#	CAPCOM6	–	DP1H.P1 = 0
P1H.2	Программный вход	P1H.2	EBC не активен и ALTSEL0P1H.P2 = 0	DP1H.P2 = 0
	Программный выход			DP1H.P2 = 1
	Выходная линия адреса A10	EBC	EBC активен	
	SSC1 прием по входу MTSR1 в режиме slave	SSC1	–	DP1H.P2 = 0
	SSC1 передача через выход MTSR1 в режиме master		EBC не активен и ALTSEL0P1H.P2 = 1	DP1H.P2 = 1
	Вход позиции CC6POS2#	CAPCOM6	–	DP1H.P2 = 0
P1H.3	Программный вход	P1H.3	EBC не активен и ALTSEL0P1H.P3 = 0	DP1H.P3 = 0
	Программный выход			DP1H.P3 = 1
	Выходная линия адреса A11	EBC	EBC активен	
	SSC1 вход синхронизации SCLK1	SSC1	–	DP1H.P3 = 0
	SSC1 выход синхронизации SCLK1		EBC не активен и ALTSEL0P1H.P3 = 1	DP1H.P3 = 1
	Вход альтернативного сигнала A «прерывание 0» EX0INA		–	DP1H.P3 = 0
P1H.x, x = 7, …, 4	Программный вход	P1H.x	EBC не активен и ALTSEL0P1H.Px = 0	DP1H.Px = 0
	Программный выход			DP1H.Px = 1
	Выходная линия адреса A15, …, A12	EBC	EBC активен	
	CC27I, …, CC24I вход захвата	CAPCOM2	–	DP1H.Px = 0
	CC27O, …, CC24O выход сравнения		EBC не активен и ALTSEL0P1H.Px = 1	DP1H.Px = 1
Примечание – EBC – контроллер внешней шины.				

9.5 Порт P2

Если 8-битный порт P2 используется для ввода и вывода основного назначения, то направление каждой линии может быть сконфигурировано с помощью регистра направления DP2, см. рисунок 9.17, таблицу 9.9. Каждая линия порта P2 может быть переключена в режим push/pull или в режим с открытым стоком, с помощью регистра управления открытым стоком ODP2, см. рисунок 9.18, таблицу 9.10. Формат регистра ALTSEL0P2 представлен на рисунке 9.19, функциональное назначение полей регистра ALTSEL0P2 – в таблице 9.11.

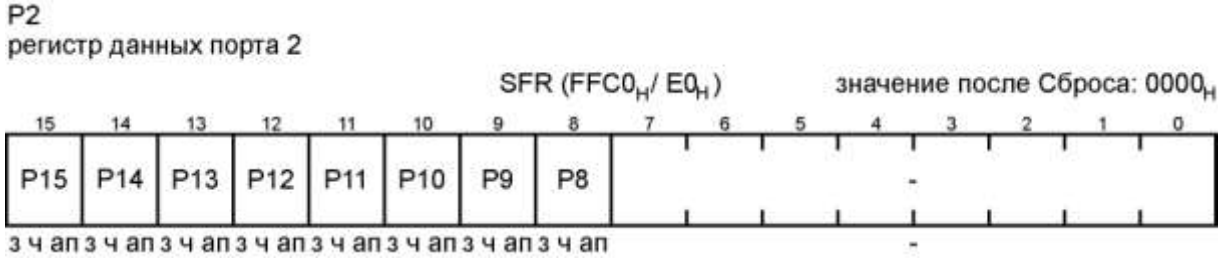


Рисунок 9.16 – Формат регистра P2

Таблица 9.8 – Функциональное назначение полей регистра P2

Поле	Биты	Тип	Описание
P2.y	15-8	Чтение Запись Аппаратное влияние	Регистр данных P2 бита y порта P2

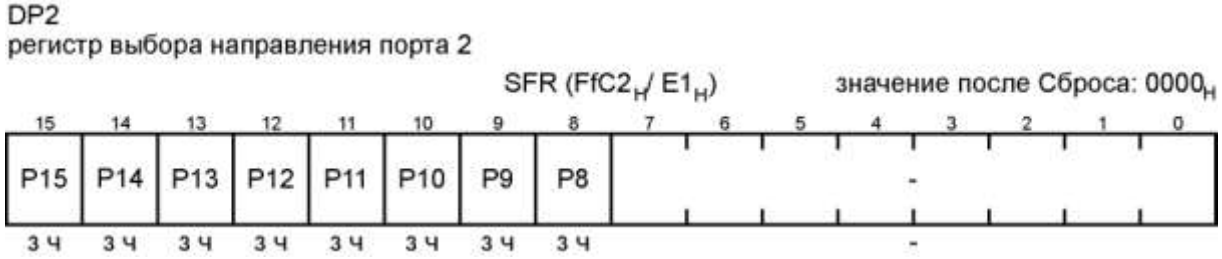


Рисунок 9.17 – Формат регистра DP2

Таблица 9.9 – Функциональное назначение полей регистра DP2

Поле	Биты	Тип	Описание
DP2.y	15-8	Чтение Запись	Регистр выбора направления DP2 бита y порта P2: 0 Линия порта P2.y настроена на вход. 1 Линия порта P2.y настроена на выход.

Рисунок 9.18 – Формат регистра ODP2

Таблица 9.10 – Функциональное назначение полей регистра ODP2

Поле	Биты	Тип	Описание
ODP2.y	15-8	Чтение Запись	Регистр управления режимом с открытым стоком ODP2 бита y порта P2: 0 Выходная линия порта P2.y функционирует в режиме push/pull. 1 Выходная линия порта P2.y функционирует в режиме работы с открытым стоком.

Рисунок 9.19 – Формат регистра ALTSEL0P2

Таблица 9.11 – Функциональное назначение полей регистра ALTSEL0P2

Поле	Биты	Тип	Описание
ALTSEL0P2.y	15-8	Чтение Запись	Регистр управления альтернативными функциями P2 бита y порта P2: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P2

Все линии P2.15, …, P2.8 порта P2 обслуживаются как входы захвата или выходы сравнения CC15IO, …, CC8IO блока CAPCOM1. Альтернативные функции порта P2 представлены на рисунке 9.20.

Когда линии порта P2 используются в качестве входов захвата, состояние входного триггера, показывающего состояние вывода порта P2, прямо подключено к блоку CAPCOM при помощи линии альтернативный вход данных. Если используется сигнал внешнего триггера захвата, направление вывода порта должно быть выставлено на вход. Если порт направлен на выход, то будет читаться состояние выходного триггера порта, поскольку вывод порта представляет состояние выходного триггера. Это может быть использовано для переключения триггера захватываемого события, с помощью программы, устанавливающей значение в выходном триггере порта. Заметим, что при настройке на вывод, не должно быть подключено никаких внешних устройств, которые могут подать сигнал на вход, иначе может произойти конфликт.

Структурная схема вывода порта P2 представлена на рисунке 9.21.

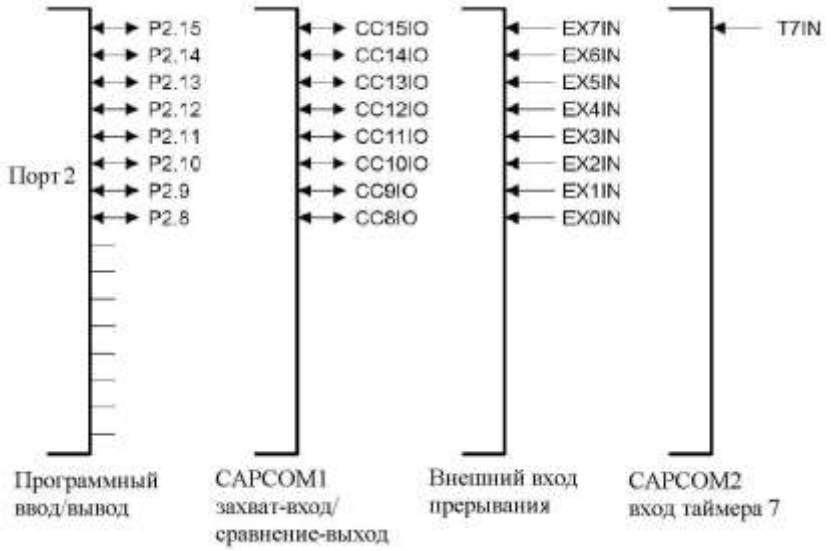


Рисунок 9.20 – Выводы порта P2 и их альтернативные функции

Когда линия порта P2 используется в качестве выходов сравнения, то сравниваемое событие прямо влияет на состояние выходного триггера порта. Как можно увидеть из структуры порта P2, см. рисунок 9.21, пользовательская программа имеет всегда свободный доступ к выводам порта в том случае, когда они используются в качестве выходов сравнения.

Все линии порта P2 также могут обслуживать входы внешних прерываний EX7IN, …, EX0IN.

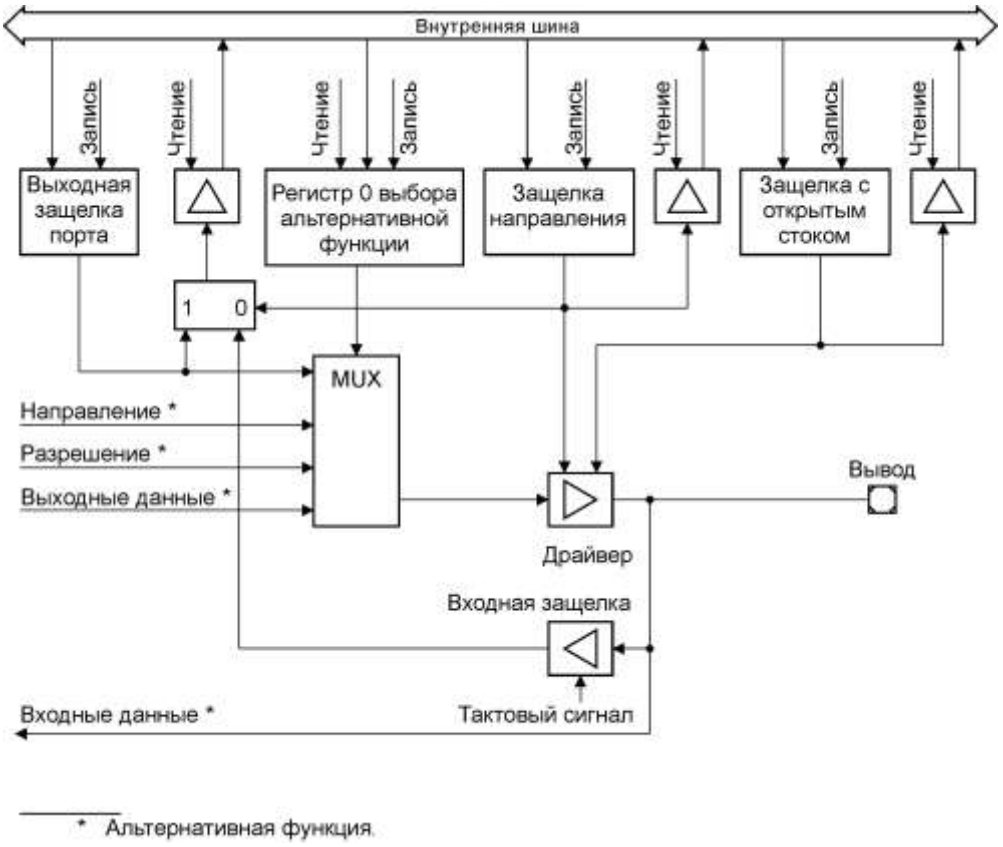


Рисунок 9.21 – Структурная схема вывода порта P2

В таблице 9.12 показаны возможные функции выводов порта P2.

Таблица 9.12 – Функции выводов порта P2

Вывод порта P2	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
P2.x, x = 15, …, 8	Программный вход	P2.x	ALTSEL0P2.Px = 0	DP2.Px = 0
	Программный выход			DP2.Px = 1
	CC15I, …, CC8I вход захвата	CAPCOM1	–	DP2.Px = 0
	CC15O, …, CC8O выход сравнения	–	ALTSEL0P2.Px = 1	DP2.Px = 1
	Вход внешнего прерывания EX7IN, …, EX0IN	–	–	DP2.Px = 0
	Вход таймера 7* T7IN	CAPCOM2*	–	DP2.P15 = 0*

_____
* Для вывода P2.15.

9.6 Порт P3

Если этот 16-битный порт использовать для ввода и вывода основного назначения, то направление каждой линии может быть изменено с помощью регистра направления DP3, см. рисунок 9.23, таблицу 9.14. Большинство линий порта может быть переключено в режим push/pull или режим с открытым стоком с помощью регистра управления открытым стоком ODP3, см. рисунок 9.23, таблицу 9.15: выводы P3.15, P3.14 и P3.12 не поддерживают режим с открытым стоком!

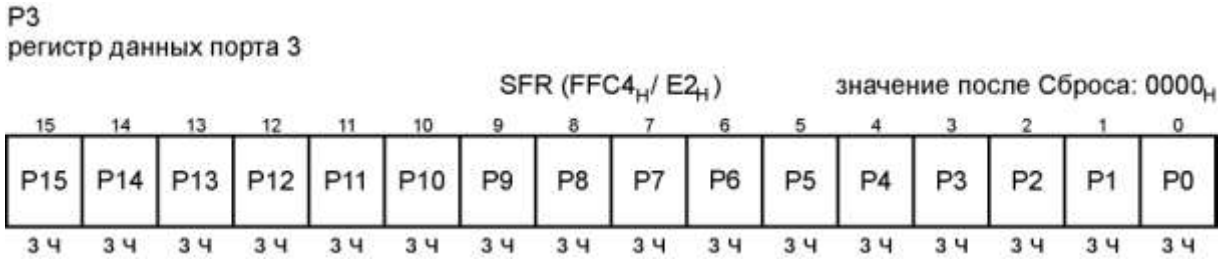


Рисунок 9.22 – Формат регистра P3

Таблица 9.13 – Функциональное назначение полей регистра P3

Поле	Биты	Тип	Описание
P3.y	15-0	Чтение Запись	Регистр данных P3 бита y порта P3

DP3
регистр выбора направления порта 3

SFR ($FFC6_H$/ $E3_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.23 – Формат регистра DP3

Таблица 9.14 – Функциональное назначение полей регистра DP3

Поле	Биты	Тип	Описание
DP3.y	15-0	Чтение Запись	Регистр выбора направления DP3 бита y порта P3: 0 Линия порта P3.y настроена на вход. 1 Линия порта P3.y настроена на выход.

ODP3
регистр управления режимом с открытым стоком порта 3

ESFR ($F1C6_H$/ $E3_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P13	-	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
-	-	З Ч	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.24 – Формат регистра ODP3

Таблица 9.15 – Функциональное назначение полей регистра ODP3

Поле	Биты	Тип	Описание
ODP3.y	13, 11-0	Чтение Запись	Регистр управления режимом с открытым стоком ODP3 бита y порта P3: 0 Выходная линия порта P3.y функционирует в режиме push/pull. 1 Выходная линия порта P3.y функционирует в режиме работы с открытым стоком.

Формат регистра ALTSEL0P3 представлен на рисунке 9.25, функциональное назначение полей регистра ALTSEL0P3 – в таблице 9.16.

Формат регистра ALTSEL1P3 представлен на рисунке 9.26, функциональное назначение полей регистра ALTSEL1P3 – в таблице 9.17.

ALTSEL0P3
регистр 0 управления альтернативными функциями порта 3

ESFR ($F1F0_H$/ $F8_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	-	P13	P12	P11	P10	P9	P8	-	-	-	-	P3	-	P1	P0
З Ч	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	-	-	-	-	З Ч	-	З Ч	З Ч

Рисунок 9.25 – Формат регистра ALTSEL0P3

Таблица 9.16 – Функциональное назначение полей регистра ALTSEL0P3

Поле	Биты	Тип	Описание
ALTSEL0P3.y	15, 13-8, 3, 1, 0	Чтение Запись	Регистр управления альтернативными функциями P3 бита y порта P3: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Рисунок 9.26 – Формат регистра ALTSEL1P3

Таблица 9.17 – Функциональное назначение полей регистра ALTSEL1P3

Поле	Биты	Тип	Описание
ALTSEL1P3.y	15, 12, 5, 1	Чтение Запись	Регистр управления альтернативными функциями P3 бита y порта P3: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P3

Выводы порта P3 предназначены для различных функций, включая линии управления внешним таймером, два последовательных интерфейса и линии управления BHE#/WRH# и CLKOUT/FOUT.

Ввод и вывод порта P3 и его альтернативные функции показаны на рисунке 9.27.

В таблице 9.18 подробно представлено использование порта P3. Когда внутренняя периферия, связанная с портом P3, настроена на использование функции альтернативного ввода, имеет место чтение входного триггера, который содержит значение на выводе микросхемы. Альтернативные функции ввода для порта включают: T0IN, T2IN, T3IN, T4IN, CAPIN, T3EUD.

Когда внутренняя периферия, связанная с портом P3, настроена на использование порта P3 для вывода, линия альтернативного вывода данных и линия выходного триггера порта соединяются между собой по AND. При использовании этих альтернативных функций пользователь обязан устанавливать направление линии порта на выход DP3.y = 1. Выводы порта P3, которые используют функции альтернативного вывода: TxD1, T6OUT, T3OUT, MRST0, MTSR0, TxD0, RxD0 и SCLK0. Структурная схема вывода порта P3 показана на рисунке 9.28.

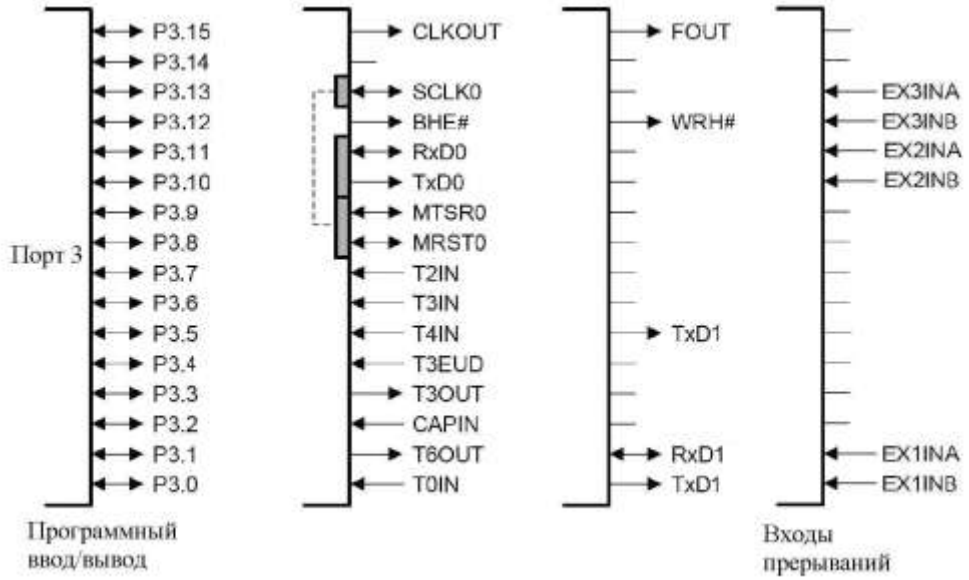
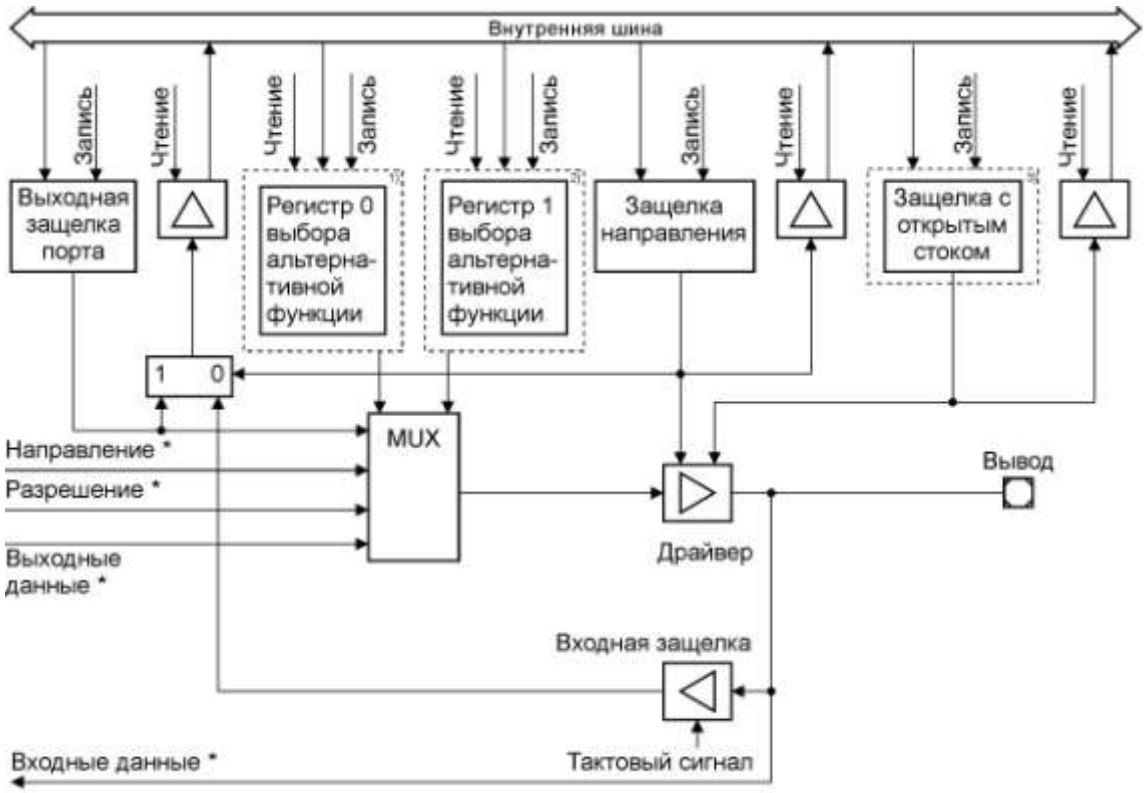


Рисунок 9.27 – Ввод и вывод порта P3 и его альтернативные функции



* Альтернативная функция.

[1] Используется для разрядов порта: P3.15, P3.13 – P3.8, P3.3, P3.1, P3.0.

[2] Используется для разрядов порта: P3.15, P3.12, P3.5, P3.1.

[3] Используется для разрядов порта: P3.13, P3.11 – P3.0.

Рисунок 9.28 – Структурная схема вывода порта P3

Разрешение функции CLKOUT/FOUT автоматически включает выходной драйвер для вывода P3.15. Не требуется установки единицы в бите DP3.15.

Таблица 9.18 – Функции выводов порта P3

Вывод порта P3	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
1	2	3	4	5
P3.0	Программный вход	P3.0	ALTSEL0P3.P0 = 0	DP3.P0 = 0
	Программный выход			DP3.P0 = 1
	ASC1 выход передатчика TxD1	ASC1	ALTSEL0P3.P0 = 1	DP3.P0 = 1
	CAPCOM1 вход счета для таймера 0 T0IN	CAPCOM1	–	DP3.P0 = 0
	Вход альтернативного сигнала B «прерывание 1» EX1INB	–	–	DP3.P0 = 0
P3.1	Программный вход	P3.1	ALTSEL0P3.P1 = 0 и ALTSEL1P3.P1 = 0	DP3.P1 = 0
	Программный выход			DP3.P1 = 1
	Выход таймера 6 через триггер T6OUT	GPT	ALTSEL0P3.P1 = 0 и ALTSEL1P3.P1 = 1	DP3.P1 = 1
	ASC1 вход приемника RxD1, использовать как вход	ASC1	–	DP3.P1 = 0
	ASC1 вход приемника RxD1, использовать как выход		ALTSEL0P3.P1 = 1	DP3.P1 = 1
	Вход альтернативного сигнала A «прерывание 1» EX1INA	–	–	DP3.P1 = 0
P3.2	Программный вход	P3.2	–	DP3.P2 = 0
	Программный выход			DP3.P2 = 1
	GPT12 вход захвата CAPIN	GPT		DP3.P2 = 0
P3.3	Программный вход	P3.3	ALTSEL0P3.P3 = 0	DP3.P3 = 0
	Программный выход			DP3.P3 = 1
	Выход таймера 3 через триггер T3OUT	GPT	ALTSEL0P3.P3 = 1	DP3.P3 = 1

Продолжение таблицы 9.18

1	2	3	4	5
P3.4	Программный вход	P3.4	–	DP3.P4 = 0
	Программный выход			DP3.P4 = 1
	Внешнее управление направлением счета таймера 3 T3EUD	GPT		DP3.P4 = 0
P3.5	Программный вход	P3.5	ALTSEL0P3.P5 = 0	DP3.P5 = 0
	Программный выход			DP3.P5 = 1
	Вход счета для таймера 4 T4IN	GPT	–	DP3.P5 = 0
	ASC1 выход передатчика TxD1	ASC1	ALTSEL1P3.P5 = 1	DP3.P5 = 1
P3.6	Программный вход	P3.6	–	DP3.P6 = 0
	Программный выход			DP3.P6 = 1
	Вход счета таймера 3 T3IN	GPT	–	DP3.P6 = 0
P3.7	Программный вход	P3.7	ALTSEL0P3.P7 = 0	DP3.P7 = 0
	Программный выход			DP3.P7 = 1
	Вход счета таймера 2 T2IN	GPT	–	DP3.P7 = 0
P3.8	Программный вход	P3.8	ALTSEL0P3.P9 = 0	DP3.P8 = 0
	Программный выход			DP3.P8 = 1
	SSC0 вход приемника в режиме master MRST0	SSC0		DP3.P8 = 0
	SSC0 выход передатчика в режиме slave MRST0		ALTSEL0P3.P9 = 1	DP3.P8 = 1
P3.9	Программный вход	P3.9	ALTSEL0P3.P9 = 0	DP3.P9 = 0
	Программный выход			DP3.P9 = 1
	SSC0 вход приемника в режиме slave MTSR0	SSC0	–	DP3.P9 = 0
	SSC0 выход передатчика в режиме master MTSR0		ALTSEL0P3.P9 = 1	DP3.P9 = 1

Продолжение таблицы 9.18

1	2	3	4	5
P3.10	Программный вход	P3.10	ALTSEL0P3.P10 = 0	DP3.P10 = 0
	Программный выход			DP3.P10 = 1
	ASC0 выход передатчика TxD0	ASC0	ALTSEL0P3.P10 = 1	DP3.P10 = 1
	Вход альтернативного сигнала B «прерывание 2» EX2INB	–	–	DP3.P10 = 0
P3.11	Программный вход	P3.11	ALTSEL0P3.P11 = 0	DP3.P11 = 0
	Программный выход			DP3.P11 = 1
	ASC0 вход приемника RxD0, использовать как вход	ASC0	–	DP3.P11 = 0
	ASC0 вход приемника RxD0, использовать как выход		ALTSEL0P3.P11 = 1	DP3.P11 = 1
	Вход альтернативного сигнала A «прерывание 2» EX2INA	–	–	DP3.P11 = 0
P3.12	Программный вход	P3.12	BHE# и WRH# запрещены	DP3.P12 = 0
	Программный выход			DP3.P12 = 1
	Выход «выбор старшего байта» BHE#	EBC	Разрешены после сброса	
	Выход «запись старшего байта» WRH#	–		
	Вход альтернативного сигнала B «прерывание 3» EX3INB	–	–	DP3.P12 = 0
P3.13	Программный вход	P3.13	ALTSEL0P3.P13 = 0	DP3.P13 = 0
	Программный выход			DP3.P13 = 1
	SSC0 вход синхронизации в режиме slave SCLK0	SSC0	–	DP3.P13 = 0
	SSC0 выход синхронизации в режиме master SCLK0		ALTSEL0P3.P13 = 1	DP3.P13 = 1
	Вход альтернативного сигнала A «прерывание 3» EX3INA	–	–	DP1H.P13 = 0
P3.14	Программный вход	P3.14	–	DP3.P14 = 0
	Программный выход			DP3.P14 = 1

Окончание таблицы 9.18

1	2	3	4	5
P3.15	Программный вход	P3.15	CLKOUT и FOUT запрещены	DP3.P15 = 0
	Программный выход			DP3.P15 = 1
	Выход системного тактового сигнала CLKOUT	–	ALTSEL0P3.P15 = 1	
	Выход программируемой частоты FOUT	–	ALTSEL1P3.P15 = 1	
Примечание – Для использования выходной альтернативной функции необходимо установить нужные биты в регистрах альтернативных функций. В регистр порта записывать ничего не нужно.				

9.7 Порт P4

Если 8-битный порт P4 использовать для ввода и вывода основного назначения или использовать альтернативные функции, то направление каждой линии может быть изменено с помощью регистра направления DP4, см. рисунок 9.30, таблицу 9.20. Только если порт P4 используется как внешняя шина, то его функциями управляет EBC.

P4
регистр данных порта 4

SFR ($FFC8_H$/ $E4_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.29 – Формат регистра P4

Таблица 9.19 – Функциональное назначение полей регистра P4

Поле	Биты	Тип	Описание
P4.y	7-0	Чтение Запись	Регистр данных P4 бита y порта P4

DP4
регистр выбора направления порта 4

SFR ($FFCA_H$/ $E5_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.30 – Формат регистра DP4

Таблица 9.20 – Функциональное назначение полей регистра DP4

Поле	Биты	Тип	Описание
DP4.y	7-0	Чтение Запись	Регистр выбора направления DP4 бита y порта P4: 0 Линия порта P4.y настроена на вход. 1 Линия порта P4.y настроена на выход.

Рисунок 9.31 – Формат регистра ODP4

Таблица 9.21 – Функциональное назначение полей регистра ODP4

Поле	Биты	Тип	Описание
ODP4.y	7-0	Чтение Запись	Регистр управления режимом с открытым стоком ODP4 бита y порта P4: 0 Выходная линия порта P4.y функционирует в режиме push/pull. 1 Выходная линия порта P4.y функционирует в режиме работы с открытым стоком.

Рисунок 9.32 – Формат регистра ALTSEL0P4

Таблица 9.22 – Функциональное назначение полей регистра ALTSEL0P4

Поле	Биты	Тип	Описание
ALTSEL0P4.y	7, 6, 5	Чтение Запись	Регистр управления альтернативными функциями P4 бита y порта P4: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P4

При использовании режима сегментированной памяти, во время циклов работы внешней шины, вывода порта P4 могут использоваться для вывода адреса сегмента. Количество выводов, использующихся для адреса сегмента, определяется внешним адресным пространством. Количество линий сегментированного адреса выбирается во

время инициализации контроллера. Так же вывода порта P4 используются модулем CAN для связи с внешними устройствами.

Ввод и вывод порта P4 и его альтернативные функции показаны на рисунке 9.33.

Рисунок 9.33 – Ввод и вывод порта P4 и его альтернативные функции

В таблице 9.23 показаны возможные функции выводов порта P4.

Таблица 9.23 – Функции выводов порта P4

Вывод порта P4	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
1	2	3	4	5
P4.x, x = 3, …, 0	Программный вход	P4.x	EBC не активен ALTSEL0P4.Px = 0	DP3.P0 = 0
	Программный выход			DP3.P0 = 1
	Выходные линии адреса A19, …, A16	EBC	EBC активен	
P4.4	Программный вход	P4.4	EBC активен ALTSEL0P4.P4 = 0	DP4.P4 = 0
	Программный выход			DP4.P4 = 1
	Выходная линия адреса A20	EBC	EBC не активен	
	Вход 1 данных CAN1RxD Вход 2 данных CAN2RxD	CAN	–	DP4.P4 = 0
	Вход альтернативного сигнала B «прерывание 5» EX5INB		–	DP4.P4 = 0

Окончание таблицы 9.23

1	2	3	4	5
P4.5	Программный вход	P4.5	EBC не активен ALTSEL0P4.P5 = 0	DP4.P5 = 0
	Программный выход			DP4.P5 = 1
	Выходная линия адреса A21	EBC	EBC активен	
	Вход 1 данных CAN1RxD Вход 2 данных CAN2RxD	CAN	ALTSEL0P4.P5 = 0	DP4.P5 = 0
	Выход данных модуля CAN (CAN1TxD & CAN2TxD)		ALTSEL0P4.P5 = 1	DP4.P5 = 1
	Вход альтернативного сигнала B «прерывание 4» EX4INB		–	DP4.P5 = 0
P4.6	Программный вход	P4.6	EBC не активен ALTSEL0P4.P6 = 0	DP4.P5 = 0
	Программный выход			DP4.P6 = 1
	Выходная линия адреса A22	EBC	EBC активен	
	Выход данных модуля CAN CAN1TxD	CAN	ALTSEL0P4.P6 = 1	DP4.P6 = 1
	Вход альтернативного сигнала A «прерывание 5» EX5INA		–	DP4.P6 = 0
P4.7	Программный вход	P4.7	EBC не активен ALTSEL0P4.P6 = 0	DP4.P7 = 0
	Программный выход			DP4.P7 = 1
	Выходная линия адреса A23	EBC	EBC активен	
	Вход 1 данных CAN1RxD Вход 2 данных CAN2RxD	CAN	–	DP4.P7 = 0
	Выход данных модуля CAN CAN2TxD		ALTSEL0P4.P7 = 1	DP4.P7 = 1
	Вход альтернативного сигнала A «прерывание 4» EX4INA		–	DP4.P7 = 0

Структурная схема вывода порта P4 представлена на рисунке 9.34.

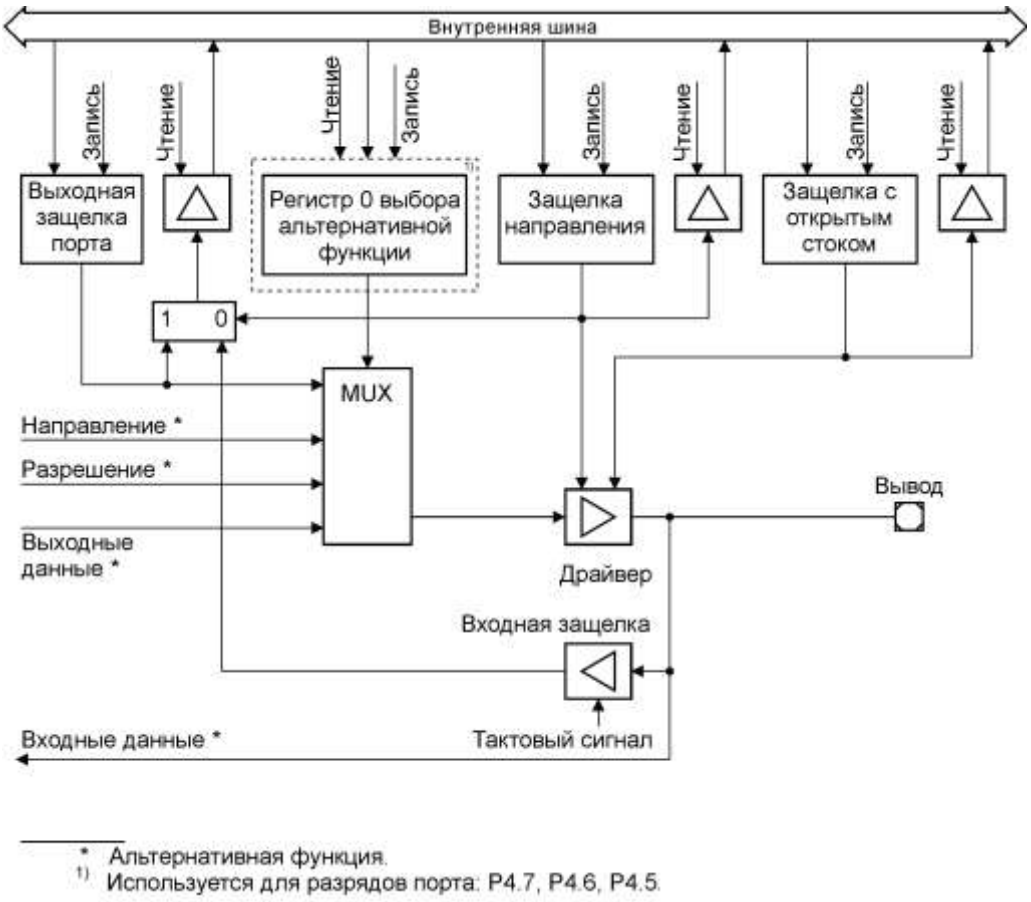


Рисунок 9.34 – Структурная схема вывода порта P4

9.8 Порт P5

16-битный входной порт P5 может только читать данные. Для этого порта нет выходного триггера и регистра направления. Данные, записанные в порт P5, будут потеряны. Формат регистра P5 представлен на рисунке 9.35, функциональное назначение полей регистра P5 – в таблице 9.24.

P5
регистр данных порта 5

SFR ($FFA2_H$/ $D1_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч

Рисунок 9.35 – Формат регистра P5

Таблица 9.24 – Функциональное назначение полей регистра P5

Поле	Биты	Тип	Описание
P5.y	15-0	Чтение	Регистр данных P5 бита y порта P5

Альтернативные функции порта P5

Каждая линия порта P5 подсоединена к входному мультиплексору АЦП. Все линии P5.15, …, P5.0 могут работать с аналоговыми сигналами AN15, …, AN0, для АЦП. Старшие шесть входов порта P5 также могут работать в качестве линий внешнего

управления таймерами модуля GPT1 и GPT2. На рисунке 9.36 показаны назначения выводов порта P5.

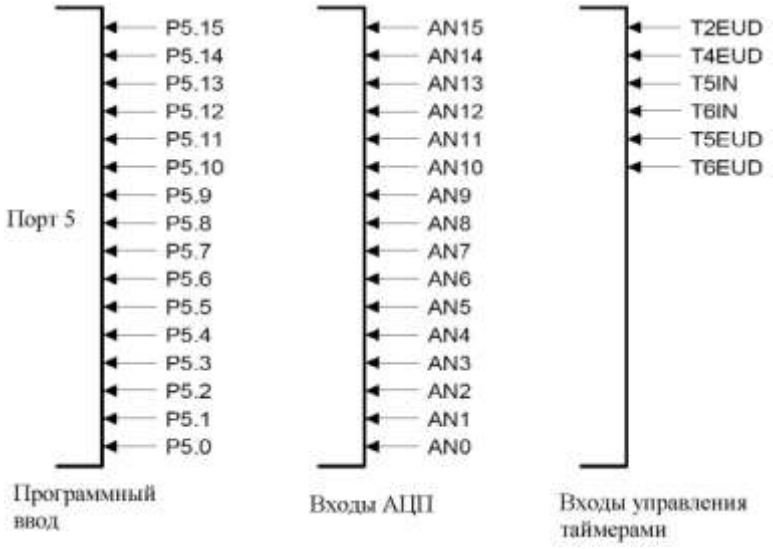


Рисунок 9.36 – Ввод порта P5 и его альтернативные функции

В таблице 9.25 подробно показаны возможные функции выводов порта P5, с описанием их назначения.

Таблица 9.25 – Функции выводов порта P5

Выводы порта P5	Альтернативная функция для АЦП	Альтернативная функция для блока таймеров	
P5.15	Аналоговый вход AN15	T2EUD	Вход таймера 2, внешнее управление направлением счета
P5.14	Аналоговый вход AN14	T4EUD	Вход таймера 4, внешнее управление направлением счета
P5.13	Аналоговый вход AN13	T5IN	Вход счета для таймера 5
P5.12	Аналоговый вход AN12	T6IN	Вход счета для таймера 6
P5.11	Аналоговый вход AN11	T5EUD	Вход таймера 5, внешнее управление направлением счета
P5.10	Аналоговый вход AN10	T6EUD	Вход таймера 6, внешнее управление направлением счета
P5.9	Аналоговый вход AN9	–	
P5.8	Аналоговый вход AN8	–	
P5.7	Аналоговый вход AN7	–	
P5.6	Аналоговый вход AN6	–	
P5.5	Аналоговый вход AN5	–	
P5.4	Аналоговый вход AN4	–	
P5.3	Аналоговый вход AN3	–	
P5.2	Аналоговый вход AN2	–	
P5.1	Аналоговый вход AN1	–	
P5.0	Аналоговый вход AN0	–	

Все выводы порта P5 работают только на вход, поэтому имеют особую структуру, показанную на рисунке 9.37.

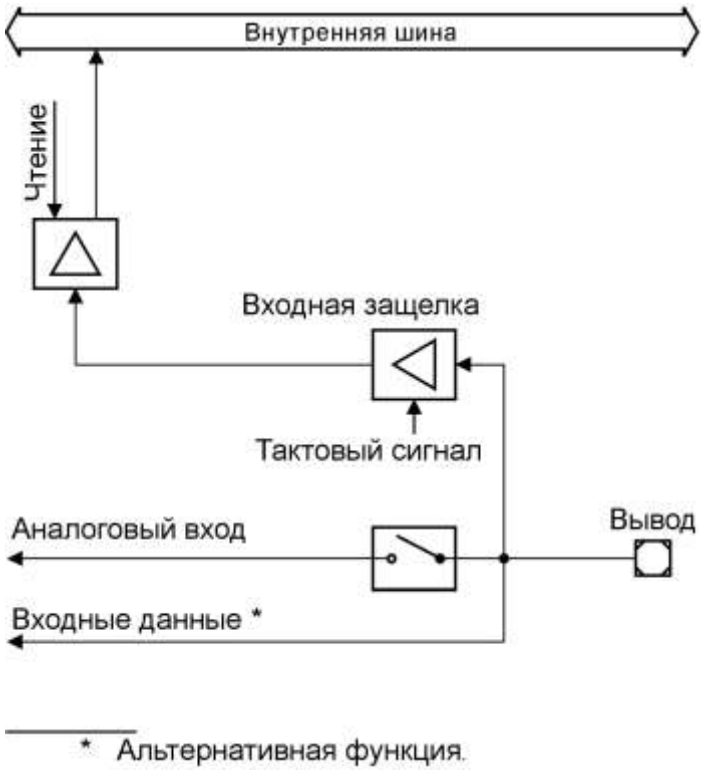


Рисунок 9.37 – Структурная схема вывода порта P5

9.9 Порт P6

Если 8-битный порт P6 используется для ввода и вывода основного назначения, то направление каждой линии может быть сконфигурировано с помощью регистра направления DP6, см. рисунок 9.39, таблицу 9.27. Каждая линия порта может быть переключена либо в режим push/pull или в режим с открытым стоком, с помощью регистра управления открытым стоком ODP6, см. рисунок 9.40, таблицу 9.28.

P6
регистр данных порта 6

SFR ($FFCC_H$ / $E6_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.38 – Формат регистра P6

Таблица 9.26 – Функциональное назначение полей регистра P6

Поле	Биты	Тип	Описание
P6.y	7-0	Чтение Запись	Регистр данных P6 бита y порта P6

DP6
регистр выбора направления порта 6

SFR ($FFCE_H$/ $E7_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.39 – Формат регистра DP6

Таблица 9.27 – Функциональное назначение полей регистра DP6

Поле	Биты	Тип	Описание
DP6.y	7-0	Чтение Запись	Регистр выбора направления DP6 бита y порта P6: 0 Линия порта P6.y настроена на вход. 1 Линия порта P6.y настроена на выход.

ODP6
регистр управления режимом с открытым стоком порта 6

ESFR ($F1CE_H$/ $E7_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	З Ч	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.40 – Формат регистра ODP6

Таблица 9.28 – Функциональное назначение полей регистра ODP6

Поле	Биты	Тип	Описание
ODP6.y	7-0	Чтение Запись	Регистр управления режимом с открытым стоком ODP6 бита y порта P6: 0 Выходная линия порта P6.y функционирует в режиме push/pull. 1 Выходная линия порта P6.y функционирует в режиме работы с открытым стоком.

Формат регистра ALTSEL0P6 представлен на рисунке 9.41, функциональное назначение полей данного регистра – в таблице 9.29.

ALTSEL0P6
регистр управления альтернативными функциями порта 6

ESFR ($F1F6_H$/ FB_H) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	P0
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.41 – Формат регистра ALTSEL0P6

Таблица 9.29 – Функциональное назначение полей регистра ALTSEL0P6

Поле	Биты	Тип	Описание
ALTSEL0P6.y	7-0	Чтение Запись	Регистр управления альтернативными функциями P6 бита y порта P6: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P6

Сигналы выбора кристалла chip select CS4#, …, CS0# могут быть выведены на пять выводов порта P6. Число задействованных сигналов устанавливается во время системного сброса в соответствии с конфигурацией кристалла. Выводы CC7IO, …, CC0IO порта P6 могут использоваться для альтернативных функций модуля CAPCOM1. На рисунке 9.42 показаны все основные функции порта P6.

Линии выбора кристалла, предназначенные для вывода соответствующих сигналов, имеют внутреннее устройство, подтягивающее к уровню логической единицы (weak pullup). Это устройство включается всегда во время сброса.

Подтягивающие цепи позволяют предотвратить ошибочный выбор внешних устройств во время системного сброса и при одновременном подключении на одну внешнюю шину более одного микроконтроллера.

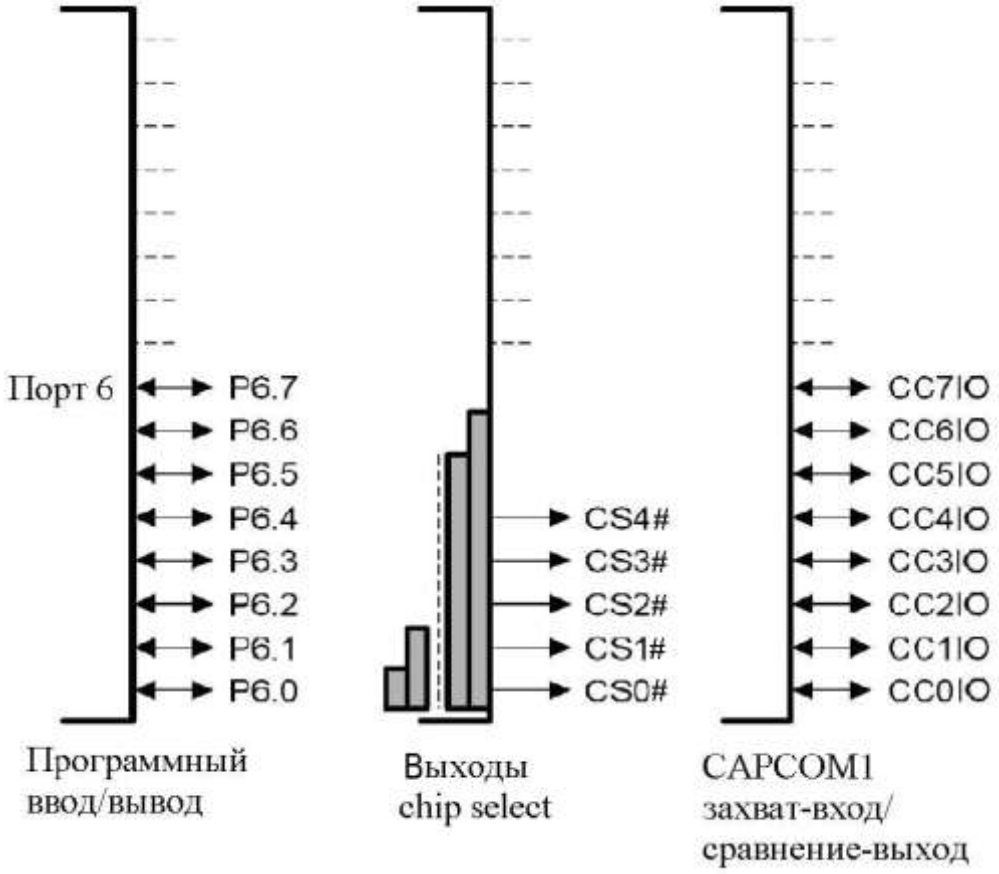


Рисунок 9.42 – Ввод и вывод порта P6 и его альтернативные функции

В таблице 9.30 показаны возможные функции выводов порта P6.

Таблица 9.30 – Функции выводов порта P6

Вывод порта P6	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
P6.x, x = 4, …, 0	Программный вход	P6.x	Выбор кристалла (chip select) не разрешен и ALTSEL0P6.Px = 0	DP6.Px = 0
	Программный выход			DP6.Px = 1
	Выходы chip select CS4#, ..., CS0#	EBC	Chip Select разрешен	–
	Входы захвата CC4I, …, CC0I	CAPCOM1	–	DP6.Px = 0
	Выходы сравнения CC4O, …, CC0O		Выбор кристалла (chip select) не разрешен ALTSEL0P6.Px = 1	DP6.Px = 1
P6.5	Программный вход	P6.5	Выбор кристалла (chip select) не разрешен и ALTSEL0P6.P5 = 0	DP6.P5 = 0
	Программный выход			DP6.P5 = 1
	Вход захвата CC5I	CAPCOM1	–	DP6.P5 = 0
	Выход сравнения CC5O		Выбор кристалла (chip select) и ALTSEL0P6.P5 = 1	DP6.P5 = 1
P6.6	Программный вход	P6.6	Выбор кристалла (chip select) не разрешен и ALTSEL0P6.P6 = 0	DP6.P6 = 0
	Программный выход			DP6.P6 = 1
	Вход захвата CC6I	CAPCOM1	–	DP6.P6 = 0
	Выход сравнения CC6O		Выбор кристалла (chip select) не разрешен и ALTSEL0P6.P6 = 1	DP6.P6 = 1
P6.7	Программный вход	P6.7	ALTSEL0P6.P7 = 0	DP6.P7 = 0
	Программный выход			DP6.P7 = 1
	Вход захвата CC7I	CAPCOM1	–	DP6.P7 = 0
	Выход сравнения CC7O		ALTSEL0P6.P7 = 1	DP6.P7 = 1

Структурная схема вывода порта P6 представлена на рисунке 9.43.

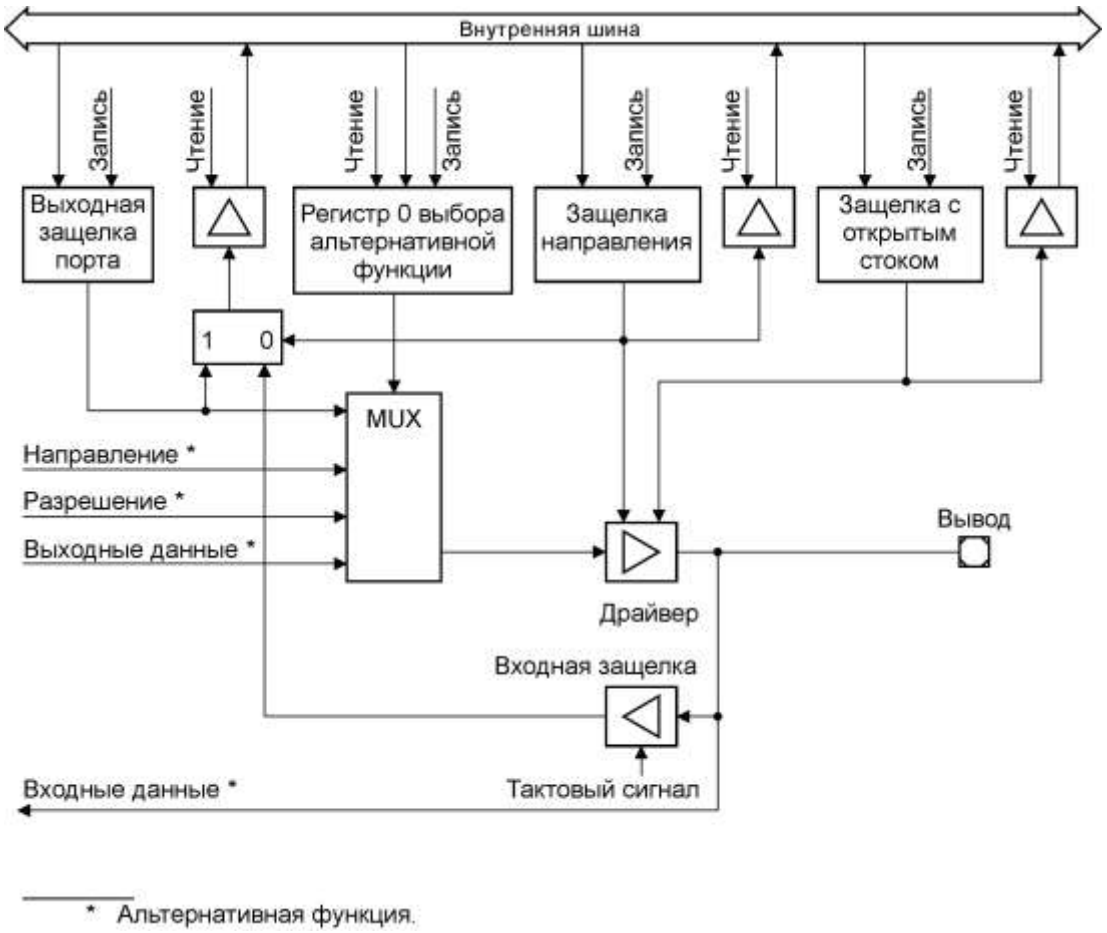


Рисунок 9.43 – Структурная схема вывода порта P6

9.10 Порт P7

При использовании этого 7-битного порта для ввода и вывода основного назначения, направление каждой линии можно настроить с помощью соответствующего регистра направления DP7, см. рисунок 9.45, таблицу 9.32. Каждая линия порта может быть переключена в режим push/pull или в режим с открытым стоком с помощью регистра ODP7, см. рисунок 9.46, таблицу 9.33.

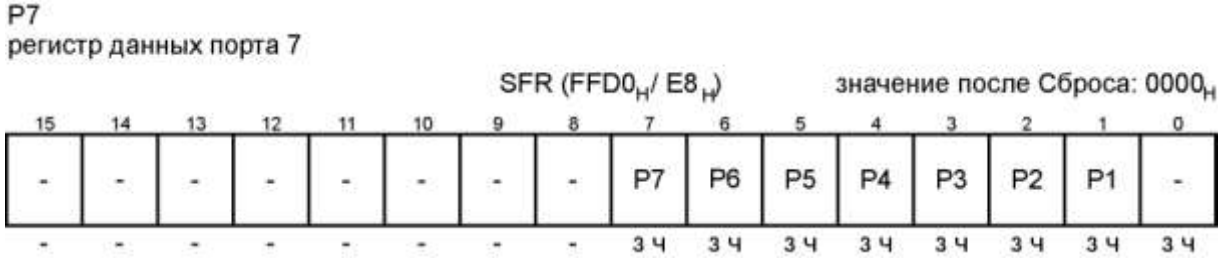


Рисунок 9.44 – Формат регистра P7

Таблица 9.31 – Функциональное назначение полей регистра P7

Поле	Биты	Тип	Описание
P7.y	7-1	Чтение Запись	Регистр данных P7 бита y порта P7

DP7
регистр выбора направления порта 7

SFR ($FFD2_H$/ $E9_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	-
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.45 – Формат регистра DP7

Таблица 9.32 – Функциональное назначение полей регистра DP7

Поле	Биты	Тип	Описание
DP7.y	7-1	Чтение Запись	Регистр выбора направления DP7 бита y порта P7: 0 Линия порта P7.y настроена на вход. 1 Линия порта P7.y настроена на выход.

ODP7
регистр управления режимом с открытым стоком порта 7

ESFR ($F1D2_H$/ $E9_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	P4	P3	P2	P1	-
-	-	З Ч	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.46 – Формат регистра ODP7

Таблица 9.33 – Функциональное назначение полей регистра ODP7

Поле	Биты	Тип	Описание
ODP7.y	7-1	Чтение Запись	Регистр управления режимом с открытым стоком ODP7 бита y порта P7: 0 Выходная линия порта P7.y функционирует в режиме push/pull. 1 Выходная линия порта P7.y функционирует в режиме работы с открытым стоком.

Формат регистра ALTSEL0P7 представлен на рисунке 9.47, функциональное назначение полей данного регистра – в таблице 9.34. Формат регистра ALTSEL1P7 представлен на рисунке 9.48, функциональное назначение полей регистра ALTSEL1P7 – в таблице 9.35.

ALTSEL0P7
регистр 0 управления альтернативными функциями порта 7

ESFR ($F1F8_H$/ FC_H) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P7	P6	P5	-	P3	P2	P1	-
-	-	-	-	-	-	-	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 9.47 – Формат регистра ALTSEL0P7

Таблица 9.34 – Функциональное назначение полей регистра ALTSEL0P7

Поле	Биты	Тип	Описание
ALTSEL0P7.y	7-5, 3-1	Чтение Запись	Регистр управления альтернативными функциями P7 бита y порта P7: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Рисунок 9.48 – Формат регистра ALTSEL1P7

Таблица 9.35 – Функциональное назначение полей регистра ALTSEL1P7

Поле	Биты	Тип	Описание
ALTSEL1P7.y	7-4	Чтение Запись	Регистр управления альтернативными функциями P7 бита y порта P7: 0 Связанный с периферией вывод не выбран как альтернативная функция. 1 Связанный с периферией вывод выбран как дополнительная функция.

Альтернативные функции порта P7

Старшие линии P7.7, …, P7.4 порта P7 могут использоваться в качестве входов захвата и выходов сравнения CC31IO, …, CC28IO блока CAPCOM2. Так же некоторые линии порта используются в качестве альтернативных функций модуля CAN.

На рисунке 9.49 показаны все основные функции порта P7.

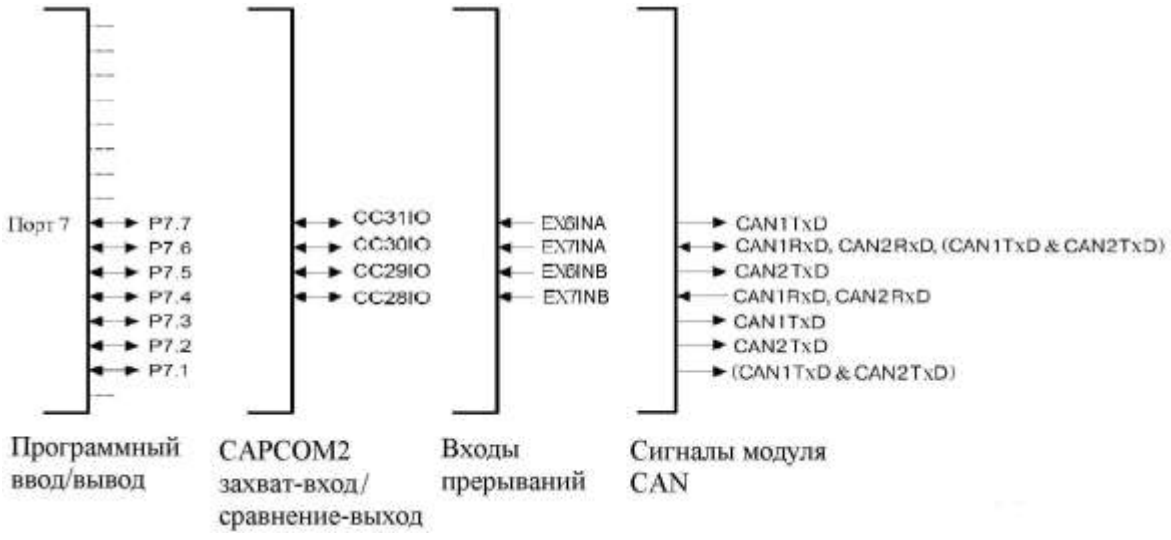


Рисунок 9.49 – Альтернативные функции и входы/выходы порта P7

В таблице 9.36 показаны возможные функции выводов порта P7.

Таблица 9.36 – Функции выводов порта P7

Вывод порта P7	Функции вывода	Связь вывода с регистром или модулем	Альтернативная функция	Управление направлением
1	2	3	4	5
P7.1	Программный вход	P7.1	ALTSEL0P7.P1 = 0	DP7.P1 = 0
	Программный выход			DP7.P1 = 1
	Выход данных модуля CAN (CAN1TxD & CAN2TxD)	CAN	ALTSEL0P7.P1 = 1	DP7.P1 = 1
P7.2	Программный вход	P7.2	ALTSEL0P7.P2 = 0	DP7.P2 = 0
	Программный выход			DP7.P2 = 1
	Выход данных модуля CAN CAN2TxD	CAN	ALTSEL0P7.P2 = 1	DP7.P2 = 1
P7.3	Программный вход	P7.3	ALTSEL0P7.P3 = 0	DP7.P3 = 0
	Программный выход			DP7.P3 = 1
	Выход данных модуля CAN CAN1TxD	CAN	ALTSEL0P7.P3 = 1	DP7.P3 = 1
P7.4	Программный вход	P7.4	ALTSEL1P7.P4 = 0	DP7.P4 = 0
	Программный выход			DP7.P4 = 1
	Вход данных модуля CAN CAN1RxD и CAN2RxD	CAN	–	DP7.P4 = 0
	Вход захвата CC28I	CAPCOM2	–	DP7.P4 = 0
	Выход сравнения CC28O		ALTSEL1P7.P4 = 1	DP7.P4 = 1
	Вход альтернативного сигнала B «прерывание 7» EX7INB	–	–	DP7.P4 = 0
P7.5	Программный вход	P7.5	ALTSEL0P7.P5 = 0 и ALTSEL1P7.P5 = 0	DP7.P5 = 0
	Программный выход			DP7.P5 = 1
	Выход данных модуля CAN CAN2TxD	CAN	ALTSEL0P7.P5 = 1 и ALTSEL1P7.P5 = 0	DP7.P5 = 1
	Вход захвата CC29I	CAPCOM2	–	DP7.P5 = 0

Окончание таблицы 9.36

1	2	3	4	5
P7.5	Выход сравнения CC29O	–	ALTSEL1P7.P5 = 1 и ALTSEL0P7.P5 = 0	DP7.P5 = 1
	Вход альтернативного сигнала B «прерывание 6» EX6INB	–	–	DP7.P5 = 0
P7.6	Программный вход	P7.6	ALTSEL0P7.P6 = 0 и ALTSEL1P7.P6 = 0	DP7.P6 = 0
	Программный выход			DP7.P6 = 1
	Выход данных модуля CAN (CAN1TxD & CAN2TxD)	CAN	ALTSEL0P7.P6 = 1 и ALTSEL1P7.P6 = 0	DP7.P6 = 1
	Вход данных модуля CAN CAN1RxD и CAN2RxD		–	DP7.P6 = 0
	Вход захвата CC30I	CAPCOM2	–	DP7.P6 = 0
	Выход сравнения CC30O		ALTSEL1P7.P6 = 1 и ALTSEL0P7.P6 = 0	DP7.P6 = 1
	Вход альтернативного сигнала A «прерывание 7» EX7INA	–	–	DP7.P6 = 0
P7.7	Программный вход	P7.7	ALTSEL0P7.P7 = 0 и ALTSEL1P7.P7 = 0	DP7.P7 = 0
	Программный выход			DP7.P7 = 1
	Выход данных модуля CAN CAN1TxD	CAN	ALTSEL0P7.P7 = 1 и ALTSEL1P7.P7 = 0	DP7.P7 = 1
	Вход захвата CC31I	CAPCOM2	–	DP7.P7 = 0
	Выход сравнения CC31O	–	ALTSEL1P7.P7 = 1 и ALTSEL0P7.P7 = 0	DP7.P7 = 1
	Вход альтернативного сигнала A «прерывание 6» EX6INA	–	–	DP7.P7 = 0

Структурная схема вывода порта P7 представлена на рисунке 9.50.

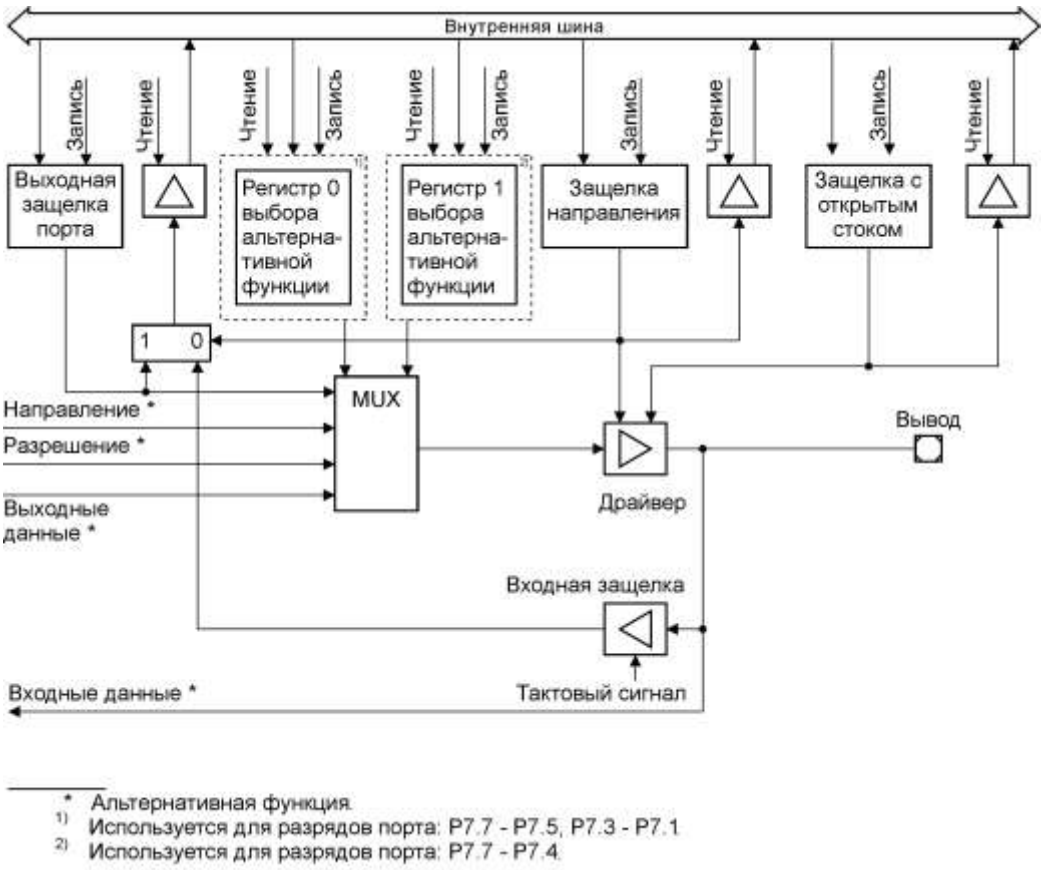


Рисунок 9.50 – Структурная схема вывода порта P7

9.11 Порт P9

Если данный 8-битный порт используется как стандартный порт входа/выхода, то управление для каждого бита конфигурируется с помощью соответствующих битов регистра управления DP9, см. рисунок 9.52, таблицу 9.38. Каждая линия порта может быть переключена в режим с открытым стоком с помощью регистра управления режимом push/pull и режимом с открытым стоком ODP9, см. рисунок 9.53, таблицу 9.39.

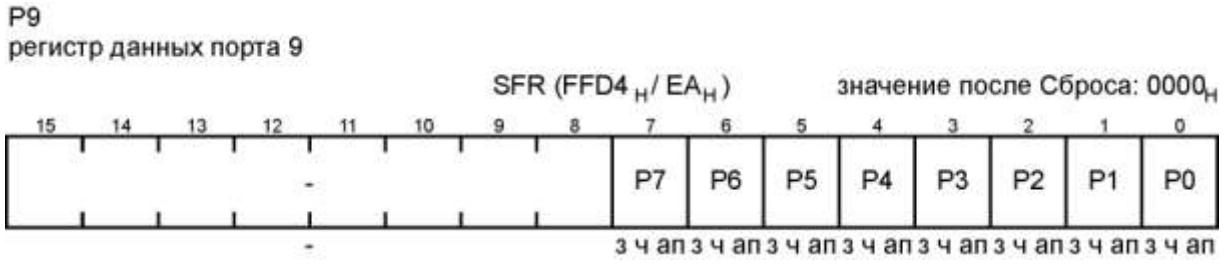


Рисунок 9.51 – Формат регистра P9

Таблица 9.37 – Функциональное назначение полей регистра P9

Поле	Биты	Тип	Описание
P9.y, y = 7, …, 0	7-0	Запись Чтение Аппаратное влияние	Биты 5–0 регистра данных порта P9

DP9
регистр выбора направления порта 9

SFR (FFD6 $_H$ / EB$_H$) значение после Сброса: 0000$_H$

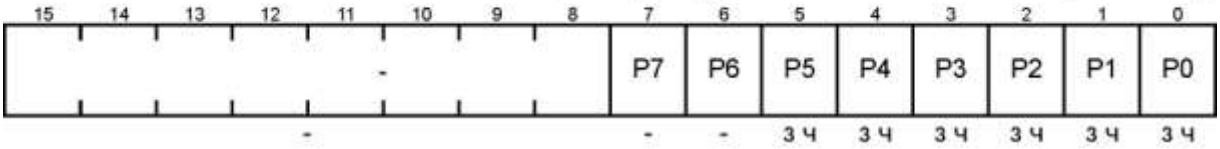


Рисунок 9.52 – Формат регистра DP9

Таблица 9.38 – Функциональное назначение полей регистра DP9

Поле	Биты	Тип	Описание
DP9.y, y = 7, …, 0	7-0	Запись Чтение	Биты 7–0 регистра управления порта P9: 0 Линия порта P9.y является входной. 1 Линия порта P9.y является выходной.

ODP9
регистр управления режимом с открытым стоком порта 9

ESFR (F1D6$_H$ / EB$_H$) значение после Сброса: 0000$_H$

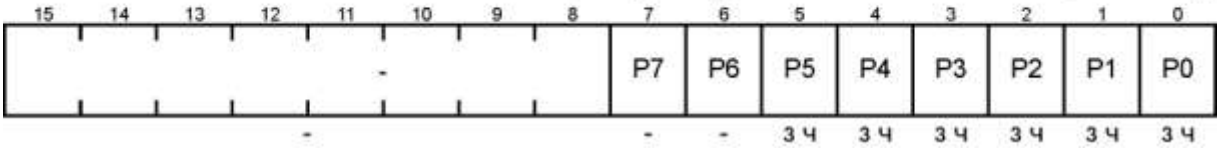


Рисунок 9.53 – Формат регистра ODP9

Таблица 9.39 – Функциональное назначение полей регистра ODP9

Поле	Биты	Тип	Описание
ODP9.y y = 7, …, 0	7-0	Запись Чтение	Биты 7–0 регистра управления порта P9: 0 Выходная линия порта P9.y функционирует в режиме push/pull. 1 Выходная линия порта P9.y функционирует в режиме с открытым стоком.

Альтернативные функции модулей I2C, CAN и CAPCOM2 выбираются с помощью регистров ALTSEL0P9, см. рисунок 9.54, таблицу 9.40, и ALTSEL1P9, см. рисунок 9.55, таблицу 9.41.

ALTSEL0P9
регистр 0 управления альтернативными функциями порта 9

ESFR (F1FC$_H$ / FE$_H$) значение после Сброса: 0000$_H$

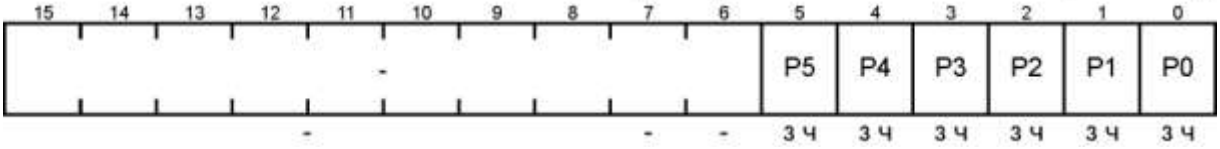


Рисунок 9.54 – Формат регистра ALTSEL0P9

Таблица 9.40 – Функциональное назначение полей регистра ALTSEL0P9

Поле	Биты	Тип	Описание
ALTSEL0 P9.y, y = 5, …, 0	5-0	Запись Чтение	Биты 5–0 регистра управления порта P9: 0 Соответствующий выход периферийного устройства не выбран в качестве альтернативной функции. 1 Соответствующий выход периферийного устройства выбран в качестве альтернативной функции.

Рисунок 9.55 – Формат регистра ALTSEL1P9

Таблица 9.41 – Функциональное назначение полей регистра ALTSEL1P9

Поле	Биты	Тип	Описание
ALTSEL1 P9.y, y = 5, …, 0	5-0	Запись Чтение	Биты 5–0 регистра управления порта P9: 0 Соответствующий выход периферийного устройства не выбран в качестве альтернативной функции. 1 Соответствующий выход периферийного устройства выбран в качестве альтернативной функции.

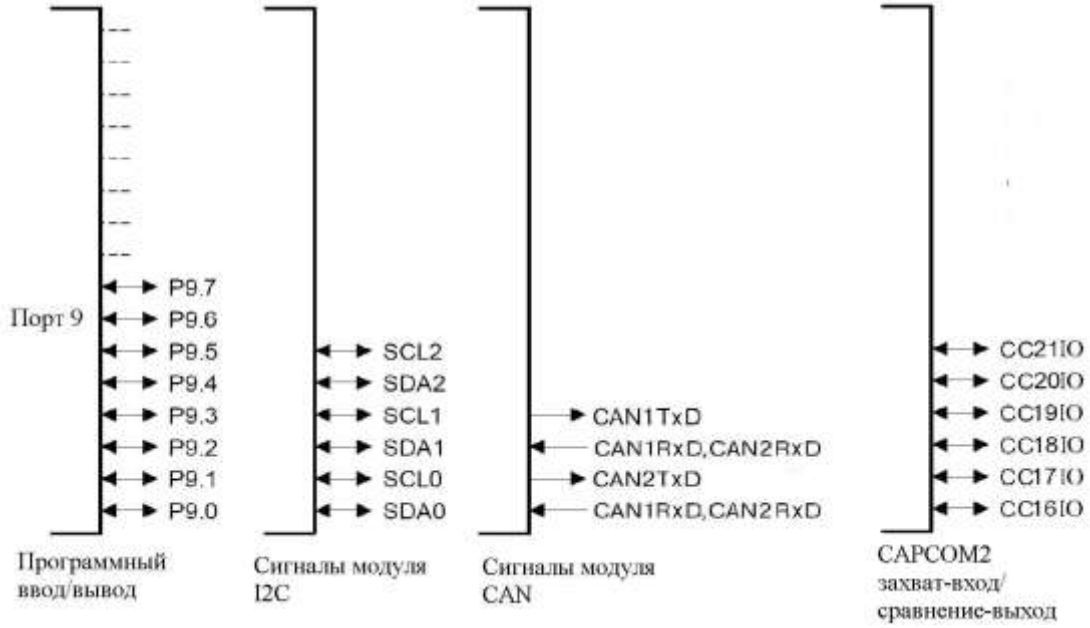


Рисунок 9.56 – Альтернативные функции и входы/выходы порта P9

В таблице 9.42 показаны возможные функции выводов порта P9.

Таблица 9.42 – Функции выводов порта P9

Вывод порта P9	Функция вывода	Соответствующий регистр/модуль	Альтернативная функция	Управление направлением
1	2	3	4	5
P9.0	Программный вход	P9.0	ALTSEL0P9.P0 = 0 и ALTSEL1P9.P0 = 0	DP9.P0 = 0
	Программный выход			DP9.P1 = 1
	CC16I вход захвата	CAPCOM2	–	DP9.P0 = 0
	CC16O выход сравнения		ALTSEL0P9.P0 = 0 и ALTSEL1P9.P0 = 1	DP9.P0 = 1
	CAN1RxD вход 1 данных CAN, CAN2RxD вход 2 данных CAN	CAN	–	DP9.P0 = 0
	Входная линия 0 данных I2C SDA0	I2C	–	DP9.P0 = 0
	Выходная линия 0 данных I2C SDA0		ALTSEL0P9.P0 = 1 и ALTSEL1P9.P0 = X	DP9.P0 = 1
P9.1	Программный вход	P9.1	ALTSEL0P9.P1 = 0 и ALTSEL1P9.P1 = 0	DP9.P1 = 0
	Программный выход			DP9.P1 = 1
	CC17I вход захвата	CAPCOM2	–	DP9.P1 = 0
	CC17O выход сравнения		ALTSEL0P9.P1 = 0 и ALTSEL1P9.P1 = 0	DP9.P1 = 1
	CAN2TxD выход 2 данных CAN	CAN	ALTSEL0P9.P1 = 1 и ALTSEL1P9.P1 = 1	DP9.P1 = 1
	Входная линия 0 тактового сигнала I2C SCL0	I2C	–	DP9.P1 = 0
	Выходная линия 0 тактового сигнала I2C SCL0		ALTSEL0P9.P1 = 1 и ALTSEL1P9.P1 = 0	DP9.P1 = 1

Продолжение таблицы 9.42

1	2	3	4	5
P9.2	Программный вход	P9.2	ALTSEL0P9.P2 = 0 и ALTSEL1P9.P2 = 0	DP9.P2 = 0
	Программный выход			DP9.P2 = 1
	CC18I вход захвата	CAPCOM2	–	DP9.P2 = 0
	CC18O выход сравнения		ALTSEL0P9.P2 = 0 и ALTSEL1P9.P2 = 1	DP9.P2 = 1
	CAN1RxD вход 1 данных CAN, CAN2RxD вход 2 данных CAN	CAN	–	DP9.P2 = 0
	Входная линия 1 данных I2C SDA1	I2C	–	DP9.P2 = 0.
	Выходная линия 1 данных I2C SDA1		ALTSEL0P9.P2 = 1 и ALTSEL1P9.P2 = 0	DP9.P2 = 1
P9.3	Программный вход	P9.3	ALTSEL0P9.P3 = 0 и ALTSEL1P9.P3 = 0	DP9.P3 = 0
	Программный выход			DP9.P3 = 1
	CC19I вход захвата	CAPCOM2	–	
	CC19O выход сравнения		ALTSEL0P9.P3 = 0 и ALTSEL1P9.P3 = 1	DP9.P3 = 1
	CAN1TxD выход 1 данных CAN	CAN	ALTSEL0P9.P3 = 1 и ALTSEL1P9.P3 = 1	DP9.P3 = 1
	Входная линия 1 тактового сигнала I2C SCL1	I2C	–	DP9.P3 = 0
	Выходная линия 1 тактового сигнала I2C SCL1		ALTSEL0P9.P3 = 1 и ALTSEL1P9.P3 = 0	DP9.P3 = 1

Окончание таблицы 9.42

1	2	3	4	5
P9.4	Программный вход	P9.4	ALTSEL0P9.P4 = 0 и ALTSEL1P9.P4 = 1	DP9.P4 = 0
	Программный выход			DP9.P4 = 1
	CC20I вход захвата	CAPCOM2	–	DP9.P4 = 0
	CC20O выход сравнения		ALTSEL0P9.P4 = 0 и ALTSEL1P9.P4 = 1	DP9.P4 = 1
	Входная линия 2 данных I2C SDA2	I2C	–	DP9.P4 = 0
	Выходная линия 2 данных I2C SDA2		ALTSEL0P9.P4 = 1 ALTSEL1P9.P4 = X	DP9.P4 = 1
P9.5	Программный вход	P9.5	ALTSEL0P9.P5 = 0 ALTSEL1P9.P5 = 0	DP9.P5 = 0
	Программный выход			DP9.P5 = 1
	CC21I вход захвата	CAPCOM2	–	DP9.P5 = 0
	CC21O выход сравнения		ALTSEL0P9.P5 = 0 ALTSEL1P9.P5 = 1	DP9.P5 = 1
	Входная линия 2 тактового сигнала I2C SCL2	I2C	–	DP9.P5 = 0
	Выходная линия 2 тактового сигнала I2C SCL2		ALTSEL0P9.P5 = 1 и ALTSEL1P9.P5 = X	DP9.P5 = 1
P9.6	Программный вход	P9.6	ALTSEL0P9.P6 = 0 ALTSEL1P9.P6 = 0	DP9.P6 = 0
	Программный выход		–	DP9.P6 = 1
P9.7	Программный вход	P9.7	ALTSEL0P9.P7 = 0 ALTSEL1P9.P7 = 0	DP9.P7 = 0
	Программный выход		–	DP9.P7 = 1

Структурная схема вывода порта P9 показана на рисунке 9.57.

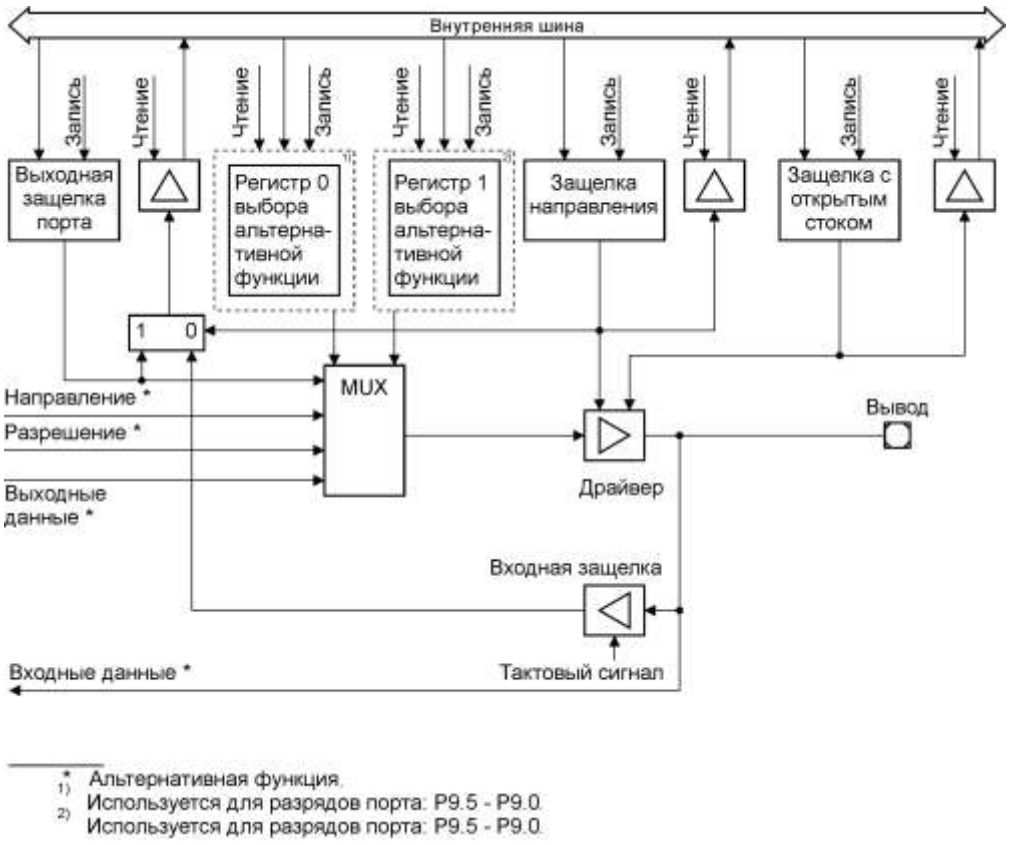


Рисунок 9.57 – Структурная схема вывода порта P9

9.12 Специальные выводы

Большинство входных и выходных сигналов реализованы как альтернативные функции выводов параллельных портов. Однако существуют сигналы, которые используют отдельные выводы: сигналы осциллятора, специальные сигналы управления, выводы питания, представленные в таблице 9.43.

Таблица 9.43 – Специальные выводы

Вывод (выводы)	Функция
ALE	Включение защелки (latch) адреса
RD#	Стробирующий сигнал чтения внешней периферии
WR#/WRL#	Стробирующий сигнал записи для внешней периферии, запись данных/запись младшего байта
READY	Вход готовности
EA#	Включение внешнего доступа
NMI#	Вход немаскируемого прерывания
XTAL1, XTAL2	Вход и выход осциллятора (основного)
XTAL3, XTAL4	Вход и выход осциллятора (вспомогательного)
RSTIN#	Вход сброса
RSTOUT#	Выход сброса
TRST#	Вход сброса для системы отладки OCDS
TMS, TCK, TDI, TDO	Интерфейс JTAG, используется OCDS
BRKIN#, BRKOUT#	Интерфейс останова для отладки
∩VCC3, ∩0V3	Выводы питания АЦП
#VCC2, #0V2	Выводы питания ядра микросхемы
#VCC1, #0V1	Выводы питания периферийной части микросхемы

Вход немаскируемого прерывания NMI# позволяет активировать ловушку с приоритетом высокого уровня с помощью внешнего сигнала. Также этот вывод используется для подтверждения команды PWRDN, переводящей контроллер в режим сохранения мощности. Значение на выводе проверяется каждый такт ЦПУ для обнаружения отрицательного перепада.

Вход осциллятора XTAL1 и выход XTAL2 предназначены для подключения кварцевого резонатора или внешнего генератора к схеме внутреннего тактового генератора. Стандартная схема подключения кварца включает кварцевый резонатор, два конденсатора и резистор, ограничивающий ток через кварцевый резонатор. Основной осциллятор предназначен для генерации основного тактового сигнала контроллера. При работе от внешнего тактового сигнала, например, от внешнего генератора или master-устройства, он подается на вход XTAL1, оставляя XTAL2 неиспользованным.

Вход осциллятора XTAL3 и выход XTAL4 связывают вспомогательный осциллятор с внешним кварцем. Этот дополнительный осциллятор используется для формирования медленного тактового сигнала, который поступает на тактовый вход модуля часов реального времени RTC, и работает независимо от генератора системного тактового сигнала. При работе от внешнего генератора медленного тактового сигнала, сигнал подают на вход XTAL3, оставляя XTAL4 неиспользованным.

Вход TRST# используется для внешнего сброса системы отладки OCDS. Во время нормальных операций этот вывод должен быть активным.

Выводы TMS, TCK, TDI и TDO интерфейса JTAG представляют собой стандартный интерфейс системы отладки OCDS. Вход данных TDI и выход данных TDO синхронизируются выводом TCK. Вывод TMS обеспечивает управление режимом.

Выводы BRKIN# и BRKOUT# используются системой отладки OCDS. При обнаружении сигнала на входе BRKIN# контроллер приостанавливает выполнение операций и переходит в режим отладки. На вывод BRKOUT# поступает сигнал с системы отладки OCDS, который может быть использован для останова другой, связанной с контроллером системы или в качестве монитора, для указания контрольной точки.

Выводы питания АЦП ∩VCC3 и ∩0V3 обеспечивают отдельное электропитание. Раздельное питание для АЦП уменьшает шум от цифровой части контроллера, передаваемый по линиям питания, делая работу АЦП более стабильной и повышая точность преобразования.

Выводы питания #VCC1, #0V1, #VCC2 и #0V2 предназначены для подключения питания цифровой логики контроллера. Разделительные конденсаторы рекомендуется подключать как можно ближе к выводам питания. Выводы #VCC2 и #0V2 запитывают ядро схемы, в то время как выводы #VCC1, #0V1 обеспечивают питание периферийной части микросхемы.

Сигнал ALE осуществляет управления защелками адреса, обеспечивая стабильный адрес в режиме мультиплексной шины. ALE выдается в каждом цикле внешней шины независимо от выбранного режима, т. е. формируется даже для режима демультиплексной шины. Сигнал ALE не активируется во время внутреннего доступа, т. е. при обращении к внутреннему ОЗУ, внутреннему ПЗУ и внутренним специальным функциональным регистрам. Во время сигнала сброса внутренняя схема обеспечивает низкий уровень сигнала ALE. Уровень сигнала на выводе ALE во время сброса выбирает режим работы: если ALE = 0_B и RD# = 0_B – контроллер работает от осциллятора, если ALE = 1_B – контроллер запускается в режиме работы от PLL.

Сигнал RD# предназначен для управления выходными буферами внешней памяти или периферии при работе с внешними устройствами. Во время доступа к X-периферии остается неактивным (высокий уровень). Во время сброса внутренняя схема обеспечивает высокий уровень сигнала на выводе RD#. В конце сброса уровень на выводе RD# защелкивается и используется для конфигурации. Уровень сигнала на выводе RD# во

время сброса выбирает режим работы: если RD# = 0_B и ALE = 0_B – контроллер работает от осциллятора, если RD# = 1_B – контроллер запускается в режиме работы от PLL.

Сигнал управления WR# / WRL# управляет передачей данных во время работы с внешней памятью или периферийными устройствами. Вывод может формировать либо сигнал WR#, общий для записи слов и байтов, либо сигнал WRL# – для записи только младшего байта слова, во время доступа к X-периферии – остается неактивным (высокий уровень). Во время сброса внутренняя схема обеспечивает высокий уровень на выводе WR# / WRL#.

Вход готовности READY# используется для увеличения времени доступа по внешней шине при совместной работе с менее быстродействующими внешними устройствами. Сигнал READY# от внешнего устройства обрабатывается во время доступа к адресному окну в том случае, если это разрешено для текущего шинного цикла. Выборка READY# может быть синхронной или асинхронной. Если для адресного окна установлены циклы задержки, то READY# не обрабатывается до окончания этой задержки. Полярность (активный уровень) сигнала на выводе READY# программируется.

Сигнал на вывод внешнего доступа EA# определяет область памяти, из которой начнется выполнение команды. Если во время сигнала сброса EA# = 0_B, то выборка программы начнется из внешней памяти, если EA# = 1_B, то контроллер будет сконфигурирован в режиме работы с внутренним ПЗУ.

Вход RSTIN# внешнего сигнала сброса обеспечивает аппаратный сброс микроконтроллера при включении питания в случае отказа аппаратных средств или при ручном сбросе.

Выход RSTOUT# выдает сигнал сброса для внешних цепей микроконтроллера. Сигнал RSTOUT# активизируется после поступления сигнала сброса на вход RSTIN#, после переполнения сторожевого таймера или после выполнения инструкции SRST. Если сигнал сброса используется только для внутренних цепей контроллера, то вывод RSTOUT# может быть заблокирован. RSTOUT# остается активным (низкий уровень сигнала) до выполнения команды EINIT, что позволяет произвести инициализацию микроконтроллера до обращения к внешним устройствам.

10 Система прерываний и ловушек

Архитектура микроконтроллера поддерживает несколько способов быстрого и гибкого обслуживания запросов, создаваемых различными источниками, как внутренними, так и внешними. Существует четыре различных механизма обработки прерываний.

Нормальная обработка прерываний

ЦПУ временно приостанавливает выполнение текущей программы и переходит к выполнению подпрограммы обработки прерывания, чтобы обслужить устройство, пославшее запрос на прерывание. Текущее состояние программы (указатель команды IP, слово состояния процессора PSW и в режиме сегментации указатель сегмента кода CSP) сохраняется во внутреннем системном стеке. Схема установления приоритетов с 16 приоритетными уровнями позволяет определить, какой из текущих запросов на прерывания должен быть обслужен.

Программные и аппаратные ловушки

Функции ловушек активируются в ответ на специальные состояния, которые могут возникнуть во время выполнения команд. Ловушка также может быть вызвана внешним воздействием при помощи немаскируемого вывода прерываний NMI#. Некоторые функции аппаратных ловушек созданы для выявления ошибочных состояний и исключений, возникающих во время выполнения команд. Аппаратные ловушки имеют самый высокий приоритет и вызывают немедленную реакцию системы. Выполнение программных ловушек вызывается командой TRAP, которая вырабатывает программное прерывание по собственному вектору. Текущее состояние системы для всех типов ловушек сохраняется в системном стеке.

Работа с прерываниями с помощью контроллера периферийных событий PEC

Обслуживание запросов на прерывания от устройств с помощью интегрированного контроллера периферийных событий PEC обеспечивает более быструю альтернативу нормальному программному выполнению прерываний. Переходя на запрос на прерывание, PEC совершает передачу одного слова или байта между двумя точками памяти через один из 8 программируемых каналов PEC. Во время передачи PEC данных нормальное выполнение программы приостанавливается. Внутренняя информация состояния программы не сохраняется. Для обслуживания PEC используется та же самая схема уровней приоритетов, как и для нормального обслуживания прерываний.

Структура системы прерываний

Структура контроллера обеспечивает 112 каналов прерываний, для которых могут быть определены 16 групп приоритетов с 4/8 подуровнями в каждой группе. Для того чтобы обеспечить возможность модульной и совместимой разработки программ, каждый источник прерываний или запросов PEC обеспечивается независимым контрольным регистром и вектором прерываний. Контрольный регистр содержит флаг запроса на прерывание, бит разрешения прерывания и уровень приоритета прерывания объединенных источников. Каждый запрос на прерывание вызывается соответствующим событием, зависящим от выбранного режима работы оборудования. В некоторых случаях несколько источников прерываний могут быть объединены в узлы для эффективного использования системных ресурсов. Эти узлы могут быть активированы несколькими источниками запросов.

Контроллер содержит векторную систему прерываний. В этой системе для векторов зарезервированы соответствующие адреса в пространстве памяти для обслуживания прерываний, ловушек и сигнала сброса RESET. Как только получен запрос, ЦПУ совершает переход по вектору, связанному с соответствующим источником прерывания. Это позволяет напрямую идентифицировать источники, совершившие запрос.

В случае выставления нескольких прерываний, ЦПУ обрабатывает прерывание с более высоким уровнем приоритета, вызывая соответствующее действие (нормальная обработка прерываний, PEC и т. д.).

Запрос будет принят центральным процессором, если у источника требования будет более высокий приоритет, чем текущий уровень приоритета центрального процессора, и если прерывание разрешено. Если приоритет источника прерывания низкий, то запрос будет обработан с задержкой.

Арбитраж прерываний

Система прерываний контроллера может обрабатывать запросы на прерывания от 112 источников. Запросы могут быть вызваны периферийными устройствами или сигналами по внешним выводам. Прерывание «окончание PEC» обеспечивает расширение функциональных возможностей PEC и связанно с одной из внутренних линий запросов прерываний.

Процесс арбитража прерываний будет запущен при разрешении запроса на прерывание и остается активным в течение всего времени рассмотрения запроса. Если запрос отсутствует, то арбитражная логика переходит в режим сохранения мощности.

Каждой линией запроса на прерывание управляет регистр xxIC (стандартная мнемоника источников прерываний). В случае выставления запроса на прерывание в соответствующем регистре управления прерыванием устанавливается флаг (бит xxIC.xxIR). Запрос на прерывание также может быть вызван программным обеспечением, если программа устанавливает соответствующий бит запроса на прерывание. Эта особенность используется операционными системами.

Если бит запроса установлен, и этот запрос на прерывание разрешен (установлен бит xxIC.xxIE этого регистра), то арбитражный цикл начинается на следующем такте. Однако если арбитражный цикл в настоящее время выполняется, новый запрос на прерывание будет обработан только во время выполнения следующего арбитражного цикла.

Все запросы на прерывания, которые находятся на рассмотрении в начале нового арбитражного цикла, рассматриваются одновременно. В пределах арбитражного цикла арбитраж независим от фактического времени запроса.

Контроллер использует двухэтапную схему установления приоритетов прерываний для арбитража прерываний, как показано на рисунке 10.1.

На первом этапе арбитражного цикла происходит сравнение до 112 уровней приоритета линий запросов на прерывания. Уровень приоритета каждого запроса состоит из уровня приоритета прерывания ILVL и уровня приоритета группы GLVL. Уровень приоритета прерывания программируется для каждой линии запроса записью значения в 4-битовое поле ILVL соответствующего регистра прерывания xxIC. Уровень приоритета группы программируется 2-битовым полем GLVL и расширением приоритета группы (xxGP) регистра xxIC.

Примечание – У всех источников запроса на прерывание, имеющих одинаковый уровень приоритета, запрограммированный в ILVL, должны быть различные уровни группового приоритета, иначе может быть сгенерирован неверный вектор прерывания.

На второй стадии арбитражного цикла уровень приоритетного прерывания сравнивается с приоритетом текущей задачи ЦПУ. Запрос прерывания будет активирован лишь в том случае, если уровень приоритета прерывания будет выше текущего уровня приоритета центрального процессора (биты ILVL регистра PSW), и если установлен флаг глобального разрешения прерываний IEN в регистре PSW. В случае если бит IEN очищен, никакой запрос на прерывание не будет обработан. Если уровень запрашиваемого прерывания ниже или равен уровню приоритета текущей задачи ЦПУ, текущий запрос будет задержан.

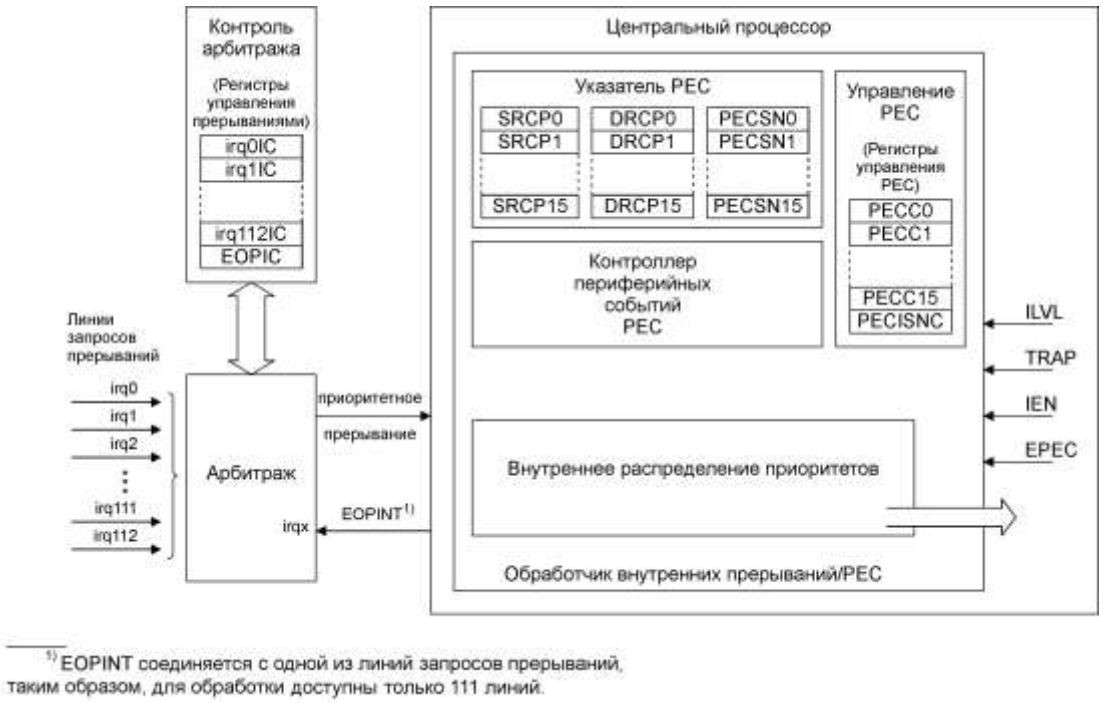


Рисунок 10.1 – Арбитраж прерываний

Уровень приоритета 0000_B является значением по умолчанию центрального процессора. Поэтому запрос с уровнем ILVL 0000_B не будет принят ЦПУ. Однако каждый разрешенный запрос на прерывание, включая все запросы с уровнем 0000_B, вызовет выход ЦПУ из режима простоя Idle, если установлен бит глобального разрешения прерываний PSW.IEN.

Все регистры управления прерываниями организованы идентично. Младшие 8 разрядов регистра управляют прерыванием и содержат информацию о состоянии источника прерывания, требующемся во время определения приоритета (арбитражный цикл). Старшие 8 разрядов регистра управления зарезервированы. Все регистры управления прерываниями являются битадресуемыми, все их биты могут читаться или записываться программно. Поэтому каждый источник может быть запрограммирован или изменен только одной командой. Чтение старшего байта (разряды с 15 по 8) регистра управления прерываниями возвращает нули. При записи в старший байт необходимо всегда записывать ноль. Размещение и назначение битов регистра управления прерываниями, показанное ниже, применимо ко всем регистрам xxIC, см. рисунок 10.2, таблицу 10.1.

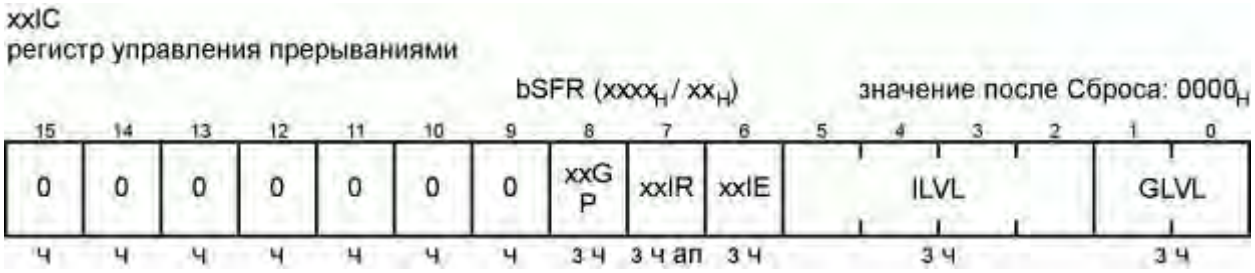


Рисунок 10.2 – Формат регистра управлениями xxIC

Таблица 10.1 – Функциональное назначение полей регистра xxIC

Поле	Биты	Тип	Описание
xxGP[1)]	8	Чтение Запись	Расширение группового приоритета. Определяет значение старшего разряда уровня группы.
xxIR[2)]	7	Чтение Запись Аппаратное влияние	Флаг запроса прерываний: 0 Ожидание запроса. 1 Источник выставил запрос на прерывание.
xxIE	6	Чтение Запись	Бит разрешения прерываний. Индивидуальное разрешение/запрет прерывания определенного источника. 0 Запрос на прерывание запрещен. 1 Запрос на прерывание разрешен.
ILVL	5-2	Чтение Запись	Уровень приоритета прерываний: FH Самый высокий уровень приоритета. … 0H Самый низкий уровень приоритета.
GLVL	1-0	Чтение Запись	Уровень группового приоритета. Определяет внутренний приоритет в случае одновременного выставления запроса на прерывание с одинаковым приоритетом.

_____
[1)] См. описание «Расширение группового приоритета» раздела 10.
[2)] Бит xxIR поддерживает бит-защиту.

Арбитражная схема может поддерживать до 15 ISR различных уровней приоритета (уровень 0 не может использоваться).

Примечание – Для уменьшения потребляемой мощности схема арбитража отключается в том случае, если отсутствуют активные запросы на прерывания.

Расширение группового приоритета

Не следует записывать бит xxGP, если задействовано меньше 64 узлов прерываний и меньше 8 каналов PEC.

Таблица векторов прерываний

В контроллере реализована векторная система прерываний. Эта система резервирует определенные адреса векторов в пространстве памяти для обслуживания прерываний, ловушек и сброса. Всякий раз, когда происходит запрос, ЦПУ переходит по адресу вектора, связанного с источником прерываний. Таким образом, этот вектор непосредственно идентифицирует источник, который выставил запрос на прерывание.

Примечание – Исключением являются аппаратные ловушки класса B, которые все определены через один вектор прерываний. Флаги состояний в регистре флагов ловушек TFR могут быть использованы для определения исключения, которое вызвало ловушку.

Зарезервированные адреса векторов собраны в таблицу переходов, которая расположена в адресном пространстве контроллера. Таблица переходов содержит соответствующие команды перехода, которые передают управление сервисной программе обслуживания прерываний, расположенной в любом месте адресного пространства. Таблица векторов расположена в нулевом сегменте адресного пространства. Начало таблицы переходов расположено в наименьшем адресе кодового нулевого сегмента. Область каждого вектора содержит два слова, за исключением вектора сброса и вектора аппаратных ловушек, занимающих соответственно по 4 и 8 слов.

Функции управления прерываниями регистра PSW

Регистр слово состояния процессора PSW, см. рисунок 10.3, таблицу 10.2, функционально делится на две части. Младший байт PSW представляет арифметическое состояние ЦПУ, старший байт управляет системой прерываний контроллера и механизмом управления интерфейсом внешней шины.

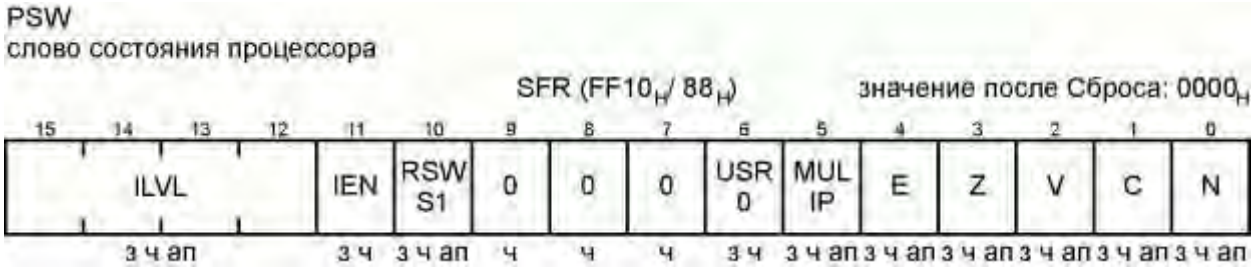


Рисунок 10.3 – Формат регистра PSW

Таблица 10.2 – Функциональное назначение отдельных битов полей регистра слово состояния процессора PSW

Поле	Биты	Тип	Описание
ILVL	15-12	Чтение Запись Аппаратное влияние	Уровень приоритета ЦПУ: FH Самый высокий уровень приоритета. … 0H Самый низкий уровень приоритета.
IEN	11	Чтение Запись	Прерывание/флаг разрешения (глобальный) PEC: 0 Прерывание/запросы PEC запрещены. 1 Прерывание/запросы PEC разрешены.

Поле ILVL обозначает текущий уровень операций центрального процессора. Это поле отражает текущий уровень приоритета исполняемой программы. При входе в подпрограмму прерываний значение этого поля изменяется на значение уровня приоритета обслуживаемого прерывания. Перед этим в стеке сохраняется предыдущее значение PSW. Если уровень приоритета запроса выше текущего уровня приоритета центрального процессора, прерывание будет обслужено. Любое прерывание с таким же уровнем или ниже останется без ответа. Текущий уровень приоритета ЦПУ может быть изменен программно. Это дает возможность управления прерываниями с низким приоритетом.

Передачи PEC не прерывают ЦПУ, а «выхватывают» один такт, поэтому обслуживание PEC не оказывает влияние на поле ILVL в регистре PSW.

Аппаратные ловушки устанавливают приоритет ЦПУ на максимальный уровень (то есть 15-й), поэтому никакие прерывания или запросы PEC не будут обслуживаться, пока выполняется подпрограмма обслуживания ловушек.

Примечание – Команда TRAP не изменяет уровень приоритета ЦПУ, поэтому программно вызванная подпрограмма обслуживания ловушек может быть прервана запросом с более высоким уровнем приоритета.

Флаг разрешения прерываний IEN глобально разрешает или запрещает запросы на прерывания и операции PEC. После очищения бита IEN, никакие новые запросы на прерывания не будут приняты. Запросы, активированные ранее, будут обработаны. Если бит IEN установлен, все источники прерываний глобально разрешены.

Примечания

1 Для разрешения прерываний каждого из источников должен быть установлен бит разрешения прерываний IEN в регистре управления прерываниями, связанный с конкретным источником.

2 Ловушки являются немаскируемыми, поэтому не управляются битом IEN.

Сохранение состояния во время обслуживания прерываний

Прежде чем запрос на прерывания начнет обслуживаться, состояние текущей задачи автоматически будет сохранено в системном стеке. Состояние ЦПУ (PSW) сохраняется до тех пор, пока выполнение прерванной задачи не будет продолжено после возвращения из подпрограммы прерываний. Адрес возврата определяется с помощью IP и в случае использования сегментированного режима памяти – указателя сегмента кода CSP.

В системном стеке сначала сохраняется значение PSW, затем IP (в несегментированном режиме) или затем CSP и IP (в сегментированном режиме), см. рисунок 10.4 . Эта последовательность оптимизирует использование системного стека при отключенной сегментации.

Значение поля уровня приоритета ЦПУ (ILVL в PSW) изменяется на значение приоритета обслуживаемого запроса на прерывания, после чего ЦПУ работает с новым уровнем приоритета.

Рисунок 10.4 – Сохранение состояния в системном стеке

После принятия запроса на прерывание флаг обслуживаемого прерывания устанавливается в 0. Вектор, связанный с источником запроса прерывания загружается в IP (в случае режима сегментированной памяти значение CSP очищается), после чего первая команда подпрограммы прерывания вызывается из адреса вектора, необходимого для перехода в саму подпрограмму прерываний. Значение указателя страницы данных и контекстный указатель при этом не изменяются.

Когда подпрограмма обслуживания прерываний заканчивает свою работу после выполнения команды RETI, информация о состоянии вызывается из системного стека в обратном порядке.

Переключение контекста

Подпрограмма обслуживания прерываний обычно сохраняет значения всех используемых регистров в стеке и восстанавливает эти значения перед возвращением. Чем больше регистров используется в подпрограмме прерываний, тем больше времени тратится при сохранении и восстановлении. Контроллер позволяет переключать полный банк ЦПУ регистров (регистры области GPR) за одну команду, таким образом, подпрограмма обслуживания использует собственный независимый банк GPR.

Команда CP SCXT «New Bank» отсылает содержимое контекстного указателя CP в системный стек и загружает в CP значение базового адреса «New Bank». С этого момента подпрограмма использует собственные регистры GPR. При завершении выполнения подпрограммы прерываний этот банк регистров сохраняется, и содержимое банка будет доступно при следующем прерывании.

Перед возвращением из прерывания (команда RETI), предыдущее значение CP просто записывается назад из системного стека.

Примечания

1 Ресурсы, используемые подпрограммой прерываний, необходимо сохранять при входе в подпрограмму обслуживания и необходимо восстанавливать при возврате в основную программу, в том числе DPP и регистры модуля умножения и деления.

2 Первые две команды, следующие после SCXT, не должны использовать GPR.

Ловушки

Программные ловушки

Команда TRAP используется, чтобы произвести программный вызов подпрограммы обработки прерываний. Номер ловушки, обозначенный в поле операнда команды TRAP, определяет адрес вектора перехода.

При выполнении команды TRAP имеет место эффект, аналогичный запросу на прерывание по этому вектору. Значения PSW, CSP (в случае сегментации памяти) и IP охраняются во внутреннем системном стеке, и после этого происходит переход по адресу вектора. При включенной сегментации в случае выполнения ловушки, для обслуживания подпрограммы прерываний значение CSP устанавливается для нулевого сегмента кода. При выполнении команды TRAP не устанавливаются флаги запроса на прерывания. Вызванная командой TRAP подпрограмма обслуживания прерываний должна быть завершена командой RETI (возврат из прерывания) для обеспечения корректного выполнения.

Примечание – Уровень ЦПУ в регистре PSW не изменяется после использования команды TRAP, поэтому подпрограмма обслуживания выполняется на том же уровне приоритета, что и основная программа. Таким образом, подпрограмма обслуживания, выполняемая по команде TRAP, может быть прервана другой ловушкой или прерыванием с более высоким уровнем приоритета.

Аппаратные ловушки

Аппаратные ловушки активируются в результате ошибок либо особых состояний системы, которые могут происходить во время выполнения программы. Аппаратные ловушки можно также совершать преднамеренно, т. е. вводя некорректный код, и при этом генерируется ловушка неправильного программного кода. Микроконтроллер различает восемь различных функций аппаратных ловушек. При активировании аппаратной ловушки ЦПУ переходит по адресу вектора данной ловушки. В зависимости от состояния обнаруженной ловушки, вызвавшая ее команда может быть либо завершена, либо не завершена, прежде чем будет введена подпрограмма обслуживания ловушки. Аппаратные ловушки – немаскируемы и всегда имеют уровень приоритета выше, чем какое-либо другое состояние ЦПУ. В случае определения в одном и том же командном такте нескольких состояний аппаратных ловушек, будет обслужена аппаратная ловушка с наивысшим уровнем приоритета. При активации аппаратной ловушки запускается команда TRAP, в результате выполнения которой производятся следующие действия: содержимое регистров PSW, CSP (в режиме сегментированной памяти) и IP сохраняются во внутреннем системном стеке; уровень приоритета ЦПУ устанавливается в максимально возможное значение, при этом запрещаются все прерывания; происходит переход по адресу ловушки. В режиме включенной сегментации CSP выставляет нулевой сегмент. Подпрограмма обслуживания ловушки завершает свою работу по команде RETI.

Восемь аппаратных ловушек подразделяются на два класса.

Класс А ловушек содержит:

- внешнее немаскируемое прерывание NMI;
- переполнение стека;
- опустошение стека.

Ловушки класса А делят между собой один уровень приоритета, при этом каждая имеет индивидуальный вектор.

Класс В ловушек содержит:
- неопределенный программный код;
- ошибку защиты;
- неправильный доступ к операнду слов;
- неправильный командный доступ;
- неправильный доступ к внешней шине.

Ловушки класса В имеют один и тот же уровень приоритета и одинаковый адрес вектора. Регистр флагов ловушек TFR позволяет подпрограмме прерываний определить тип активированной ловушки. Каждая ловушка идентифицируется с помощью независимого флага. При активации аппаратной ловушки устанавливается соответствующий ей флаг в регистре TFR, см. рисунок 10.5, таблицу 10.3.

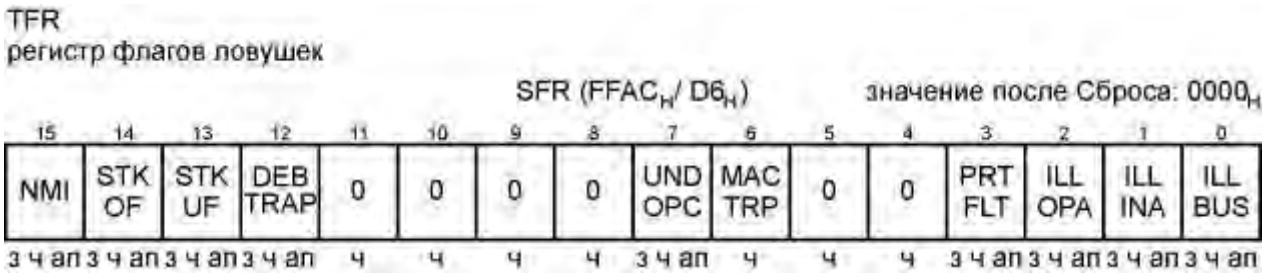


Рисунок 10.5 – Формат регистра флагов ловушек TFR

Таблица 10.3 – Функциональное назначение полей регистра TFR

Поле	Бит	Тип	Описание
1	2	3	4
ILLBUS	0	Чтение Запись Аппаратное влияние	Флаг неправильного доступа к внешней шине: 0 Нет ошибки доступа. 1 Обнаружена попытка внешнего доступа с неопределенной внешней шиной.
ILLINA	1	Чтение Запись Аппаратное влияние	Флаг неправильного командного доступа: 0 Нет ошибки командного доступа. 1 Обнаружена попытка перехода к нечетному адресу.
ILLOPA	2	Чтение Запись Аппаратное влияние	Флаг неправильного доступа к слову операнда: 0 Нет ошибки доступа. 1 Обнаружена попытка доступа (чтения или записи) по нечетному адресу слова операнда.
PRTFLT	3	Чтение Запись Аппаратное влияние	Флаг ошибки защиты: 0 Нет ошибки защиты. 1 Обнаружена защищенная команда в неправильном формате.
MACTRP	6	Чтение Запись Аппаратное влияние	Флаг прерывания MAC: 0 Нет прерывания MAC. 1 Обнаружено прерывание MAC.

Окончание таблицы 10.3

1	2	3	4
UNDOPC	7	Чтение Запись Аппаратное влияние	Флаг неопределенного программного кода: 0 Нет ошибочного программного кода. 1 Обнаружен неверный программный код, текущая декодируемая команда не является командой контроллера.
DEBTRAP	12	Чтение Запись Аппаратное влияние	Флаг отладки
STKUF	13	Чтение Запись Аппаратное влияние	Флаг опустошения стека: 0 Нет опустошения стека. 1 Обнаружено опустошение стека, текущее значение указателя стека превышает содержимое регистра STKUN.
STKOF	14	Чтение Запись Аппаратное влияние	Флаг переполнения стека: 0 Нет переполнения стека. 1 Обнаружено переполнение стека, текущее значение указателя стека меньше содержимого регистра STKOV.
NMI	15	Чтение Запись Аппаратное влияние	Флаг не маскируемого прерывания: 0 Нет не маскируемого прерывания. 1 Обнаружен отрицательный перепад на выводе NMI#.

Примечание – Подпрограмма обслуживания ловушки должна очистить флаг, соответствующий ей, в противном случае будет произведен новый запрос на обслуживание ловушки после выхода из подпрограммы обслуживания. Программная установка флага запроса ловушки приводит к эффекту, аналогичному аппаратной активации ловушки.

Функции сброса (аппаратный, программный, от сторожевого таймера) могут быть отнесены к типу ловушек. Функции сброса имеют высочайший приоритет (приоритет ловушки IV).

Ловушка отладки имеет второй по высоте уровень приоритета (уровень приоритета III), третий по высоте уровень приоритета принадлежит ловушкам класса A (уровень приоритета II) и четвертый по высоте уровень приоритета имеют ловушки класса B (уровень приоритета I).

Таким образом ловушка отладки может прервать ловушку класса A и класса B, а ловушка класса A может прервать ловушку класса B. Ловушка отладки – специальный вид прерывания, использующийся в целях отладки. Эта ловушка позволяет отладчику прерывать ловушки аппаратных средств и аппаратные прерывания.

Детальное описание ловушки отладки можно найти в подразделе 7.1 описания блока OCDS.

Таблица 10.4 – Приоритет ловушек

Ловушка	Флаг ловушки	Вектор ловушки	Номер ловушки	Приоритет ловушки
1	2	3	4	5
Функции RESET: Аппаратный сброс Программный сброс Переполнение сторожевого таймера	 – – –	 RESET RESET RESET	 00_H 00_H 00_H	 IV IV IV
Ловушка отладки	DEBUG	DEBTAP	08_H	III

Окончание таблицы 10.4

1	2	3	4	5
Класс А аппаратных ловушек:				
Немаскируемое прерывание	NMI	NMITRAP	02_H	II.3
Переполнение стека	STKOF	STOTRAP	04_H	II.2
Очистка стека	STKUF	STUTRAP	06_H	II.1
Программный разрыв	SOFTBRK	SDRKTRAP	08_H	II.0
Класс В аппаратных ловушек:				
Неопределенный программный код	UNDOPC	BTRAP	$0A_H$	I
Ошибка защиты	PRTFLT	BTRAP	$0A_H$	I
Неверный доступ к операнду слова	ILLOPA	BTRAP	$0A_H$	I
Неверный командный доступ	ILLINA	BTRAP	$0A_H$	I
Неверный доступ к внешней шине	ILLBUS	BTRAP	$0A_H$	I

Ловушки класса А

Ловушка класса А активируется системными событиями NMI или специальными событиями ЦПУ, такими как программный разрыв, переполнение стека или очистка стека. Ловушки класса А не используются для указания отказов аппаратных средств. Каждой ловушке класса А соответствует собственный вектор в таблице векторов. В случае одновременной активации нескольких ловушек класса А, происходит их распределение по приоритетам. При этом соответствующие флаги ловушек устанавливаются одновременно. После обслуживания ловушки с высшим приоритетом значение IP считывается из стека и начинается обслуживание следующей ловушки, если флаг следующей ловушки не был очищен подпрограммой обслуживания первой ловушки.

Внешняя NMI ловушка

При обнаружении отрицательного фронта на входе NMI# (немаскируемое прерывание), устанавливается флаг NMI, и ЦПУ начинает выполнять подпрограмму обслуживания ловушки NMI.

Примечание – Выводом NMI# проверяется каждый такт ЦПУ для выявления отрицательного перепада.

Ловушка переполнения стека STKOF

Всякий раз, когда значение указателя стека SP становится меньше значения регистра STKOV, устанавливается флаг переполнения стека STKOF в регистре TFR и запускается подпрограмма обработки ловушки переполнения стека. То, какое значение IP будет помещено в системный стек, зависит от операции, вызвавшей декремент SP. Когда декремент стека совершен с помощью команды PUSH, или команды CALL, или прерывания ловушки, помещенное значение IP является адресом следующей команды. Когда SP декрементируется командой вычитания, помещенное значение IP представляет собой адрес первой или второй команды после команды вычитания.

Для устранения переполнения стека, стек должен содержать достаточно места для того, чтобы сохранить состояние системы (PSW, IP и в режиме сегментации – CSP) дважды. В противном случае должен быть произведен системный сброс.

Ловушка опустошения стека STKUF

Всякий раз, когда значение указателя становится больше содержимого регистра STKUF, устанавливается флаг опустошения стека STKUF в регистре TFR, и ЦПУ запускает выполнение подпрограммы обработки ловушки для опустошения стека. То, какое значение IP будет помещено в системный стек, зависит от операции, вызвавшей приращение значения SP. Когда приращение SP осуществляется с помощью выполнения команды POP или команды возврата, помещенное значение IP является адресом следующей команды. Когда SP увеличивается командой сложения, помещенное значение IP представляет адрес первой или второй команды после команды сложения.

Ловушки класса B

Класс B ловушек активируется неисправимым отказом аппаратных средств. В случае отказа аппаратных средств ЦПУ должено немедленно запустить подпрограмму обслуживания отказа. Ловушка класса B может прервать выполнение последовательности команд ATOMIC/EXTEND. После выполнения программы обработки ловушки класса B, прерванная текущая инструкция не может быть восстановлена.

В случае, когда ловушки класса A и класса B активируются одновременно, оба флага ловушек устанавливаются. Если это происходит во время выполнения последовательности EXT команд или команды ATOMIC, ловушка класса B прерывает выполнение последовательности и ловушка класса A будет немедленно обработана. После выполнения программы обработки ловушки IP извлекается из стека и сразу же помещается обратно в стек, после чего запускается подпрограмма обработки ловушки класса B. В этой ситуации не сохраняется значение IP команды, при которой ловушка имела место. Все ловушки класса B имеют одинаковый уровень приоритета. При одновременной активизации нескольких ловушек класса B в регистре TFR устанавливаются соответствующие флаги и запускается подпрограмма обработки ловушки. Ловушки класса B имеют один и тот же вектор, поэтому приоритет определяется программно во время выполнения подпрограммы обслуживания. Во время выполнения подпрограммы обслуживания ловушки класса A ни одна ловушка класса B не будет обслужена до тех пор, пока подпрограмма обслуживания ловушки не завершится командой RETI. В этом случае ловушка класса B сохраняется в TFR, но значение IP команды, при которой ловушка имела место, будет потеряно.

Ловушка неопределенного программного кода UNDOPC

Если при дешифрации текущей команды ЦПУ определяет неверный код команды, флаг UNDOPC в регистре TFR устанавливается, и ЦПУ запускает выполнение программы обработки ловушки неопределенного программного кода. Значение IP, помещенное в системный стек, является адресом команды, вызвавшей активацию ловушки.

Эта ловушка может быть использована для эмуляции неизвестных команд. Подпрограмма обслуживания ловушки может проверить ошибочный код, базируемый на расположенном в стеке IP. Для продолжения работы основной программы, перед выполнением команды RETI необходимо увеличить расположенное в стеке значение IP на размер неопределенной команды.

Ловушка ошибки защиты PRTFLT

Всякий раз при исполнении одной из защищенных команд, в том случае, когда код команды не повторяется дважды во втором слове команды и байт последующего кода не является дополнительным к первому, устанавливается флаг PRTFLT регистра TFR, и ЦПУ начинает исполнять подпрограмму ловушки ошибки защиты. Защищенные команды включают DISWDT, EINIT, IDLE, PWRDN, SRST и SRVWDT. Значение IP посылается в системный стек перед выполнением подпрограммы и затем записывается назад при выходе из подпрограммы.

Ловушка некорректного доступа к слову операнда ILLOPA

Всякий раз, когда при записи или чтении слова ЦПУ обращается к нечетному адресу байта, флаг ILLOPA в регистре TFR устанавливается, и ЦПУ запускает подпрограмму обработки ловушки неправильного доступа к слову операнда. Значение IP, помещенное в стек, является адресом команды после той, которая вызвала активацию ловушки.

Ловушка некорректного командного доступа ILLINA

Каждый раз, когда переход выполняется к нечетному адресу байта, флаг ILLINA в регистре TRF устанавливается, и ЦПУ переходит к выполнению подпрограммы обработки ловушки некорректного командного доступа. Значение IP, помещенное в стек, является неправильным нечетным адресом команды перехода.

Ловушка некорректного доступа к внешней шине ILLBUS

Всякий раз при получении команды чтения или записи данных с внешней шины, когда конфигурация шины не была определена, флаг ILLBUS в регистре EKA устанавливается, и ЦПУ переходит к выполнению подпрограммы обработки ловушки. Значение IP, помещенное в системный стек, является адресом команды после той, которая вызвала активацию ловушки.

Контроллер периферийных событий PEC

Контроллер периферийных событий PEC определяет способ обработки запроса на прерывание. Это может быть нормальное обслуживание прерывания либо быстрая передача данных между двумя точками памяти. PEC управляет 8 каналами быстрой передачи данных.

Во время обработки нормального прерывания ЦПУ останавливает выполнение программы и переходит к подпрограмме обслуживания прерываний. Текущее состояние программы и слово состояния процессора должны быть сохранены в стеке.

Если для обслуживания прерываний выбран канал PEC, то осуществляется пересылка слова или байта данных между двумя ячейками памяти. Во время PEC передачи нормальное выполнение программы ЦПУ останавливается на 1 машинный цикл. Никакая внутренняя информация состояния программы или системы не сохраняется. PEC передача – самый быстрый ответ на прерывание. Во многих случаях PEC передачи достаточно, чтобы обслужить периферийный запрос на прерывание.

Каналы PEC могут выполнять следующие действия:

- пересылка байта или слова;
- непрерывная пересылка данных;
- выработка специфического прерывания для отдельного канала после завершения передачи данных или для всех каналов;
- автоматический инкремент источника или указателя адреса;
- линия связи между двумя каналами PEC.

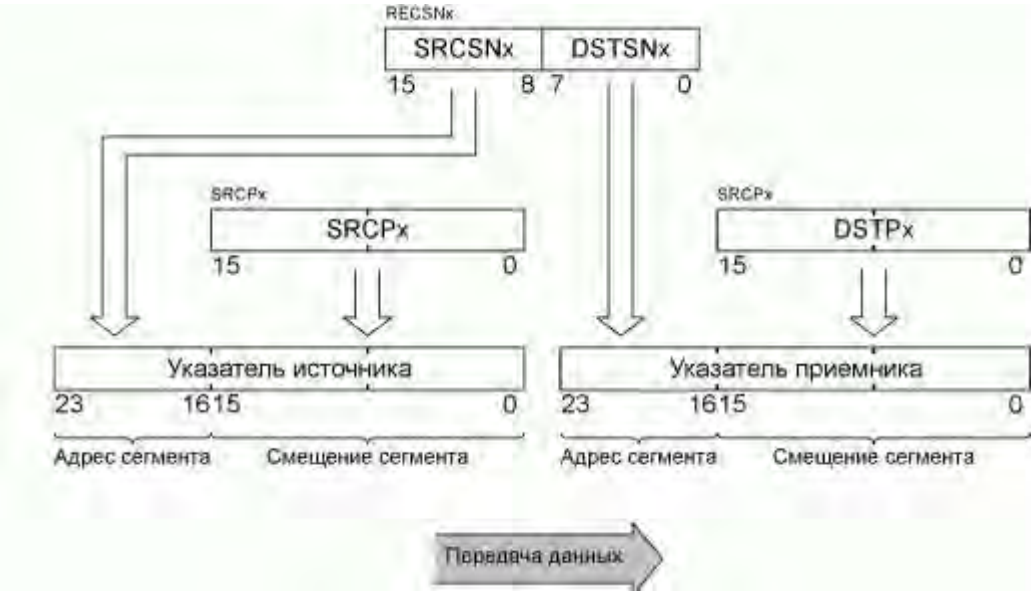
Примечание – PEC передача выполняется, если ее уровень приоритета выше, чем текущий уровень приоритета ЦПУ.

Указатели адреса источника и приемника PEC

Указатели адреса источника и приемника определяют местоположения регистров, между которыми должны быть перемещены данные. Все указатели имеют ширину 24 бита. 24-битовый адрес сохраняется в регистрах SRCPx (младшие 16 битов адреса) и PECSNx (старшие 8 битов адреса).

Только младшие 16 разрядов указателей адреса PEC (смещение внутри сегмента) могут быть изменены механизмом передачи PEC. Старшие 8 битов представляют номер сегмента и не изменяются аппаратно. Поэтому указатели PEC могут быть увеличены в пределах адресуемого пространства одного сегмента. Если указатель адреса смещения примет значение $FFFF_H$ в случае передачи байта BWT = 1_B или $FFFE_H$ в случае передачи слова BWT = 0_B, то следующее наращивание значения приведет к переполнению указателя адреса.

Примечание – Если для определенного канала PEC выбрана передача слова данных BWT = 0_B, то указатели адреса источника и соответствующего ему приемника должны содержать правильный адрес слова. В противном случае активируется ловушка неправильного доступа к слову операнда во время использования канала PEC.



Примечание – x = 7, ..., 0, в зависимости от номера канала PEC.

Рисунок 10.6 – Работа указателя адреса PEC

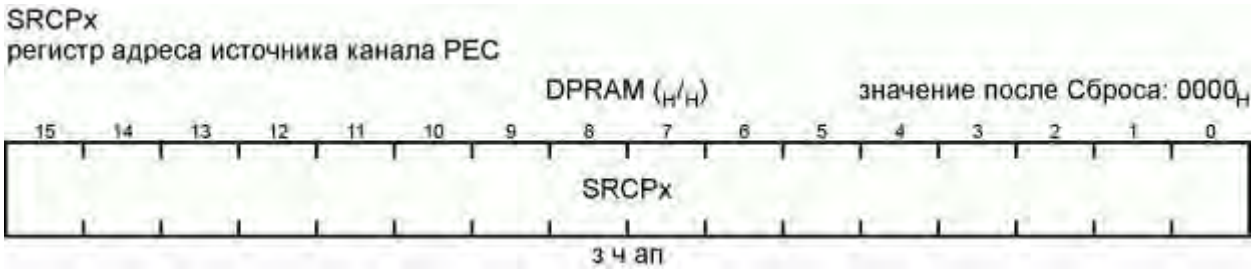


Рисунок 10.7 – Формат регистра SRCPx

Таблица 10.5 – Функциональное назначение полей регистра SRCPx

Поле	Бит	Тип	Описание
SRCPx	15-0	Чтение Запись Аппаратное влияние	Адрес источника канала x

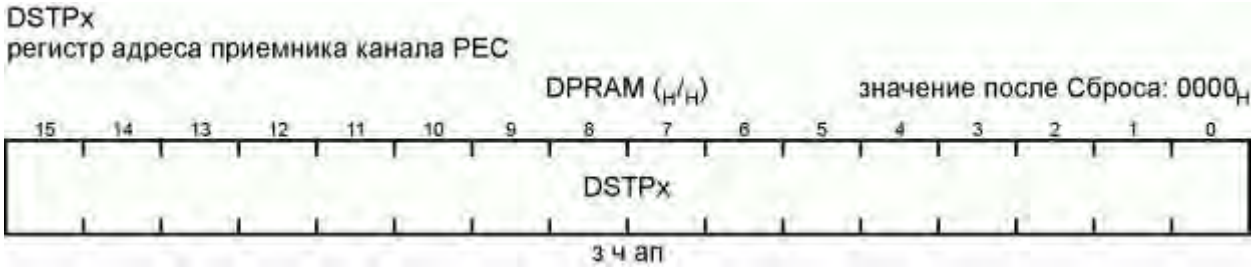


Рисунок 10.8 – Формат регистра DSTPx

Таблица 10.6 – Функциональное назначение полей регистра DSTPx

Поле	Биты	Тип	Описание
DSTPx	15-0	Чтение Запись Аппаратное влияние	Адрес приемника канала x

Таблица 10.7 – Адреса указателей источников и приемников PEC в DPRAM

Указатель	Адрес	Указатель	Адрес
SRCP15	$00`FCCC_H$	DSTP15	$00`FCCE_H$
SRCP14	$00`FCC8_H$	DSTP14	$00`FCCA_H$
SRCP13	$00`FCC4_H$	DSTP13	$00`FCC6_H$
SRCP12	$00`FCC0_H$	DSTP12	$00`FCC2_H$
SRCP11	$00`FCDC_H$	DSTP11	$00`FCDE_H$
SRCP10	$00`FCD8_H$	DSTP10	$00`FCDA_H$
SRCP9	$00`FCD4_H$	DSTP9	$00`FCD6_H$
SRCP8	$00`FCD0_H$	DSTP8	$00`FCD2_H$
SRCP7	$00`FCFC_H$	DSTP7	$00`FCFE_H$
SRCP6	$00`FCF8_H$	DSTP6	$00`FCFA_H$
SRCP5	$00`FCF4_H$	DSTP5	$00`FCF6_H$
SRCP4	$00`FCF0_H$	DSTP4	$00`FCF2_H$
SRCP3	$00`FCEC_H$	DSTP3	$00`FCEE_H$
SRCP2	$00`FCE8_H$	DSTP2	$00`FCEA_H$
SRCP1	$00`FCE4_H$	DSTP1	$00`FCE6_H$
SRCP0	$00`FCE0_H$	DSTP0	$00`FCE2_H$

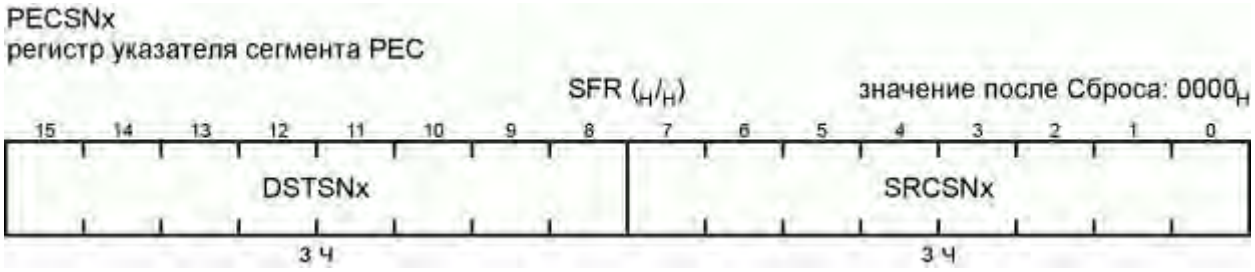


Рисунок 10.9 – Формат регистра PECSNx

Таблица 10.8 – Функциональное назначение полей регистра PECSNx

Поле	Биты	Тип	Описание
DSTSNx	15-8	Чтение Запись	Указатель сегмента адреса приемника канала x
SRCSNx	7-0	Чтение Запись	Указатель сегмента адреса источника канала x

Регистры управления PEC

Каждым каналом PEC управляет соответствующий регистр контроля канала PEC (PECCx), регистр адреса источника SRCPx, регистр адреса приемника DSTPx и регистр указателя сегмента адреса источника и приемника PECSNx, где x – номер канала PEC. Регистры PECCx управляют уровнем приоритета и определяют действие, которое будет выполнено.

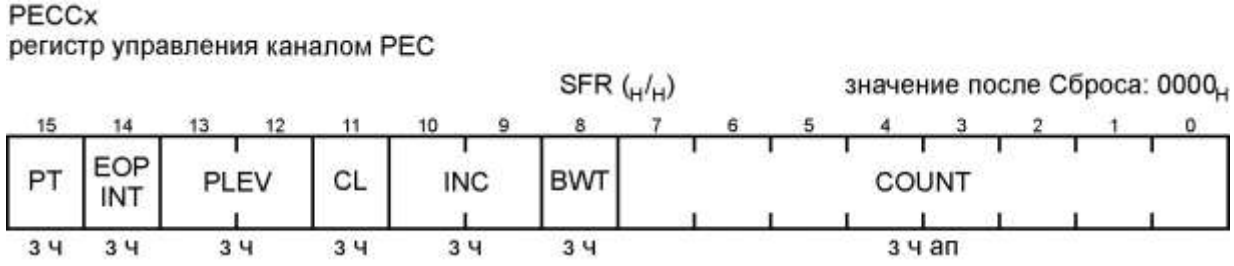


Рисунок 10.10 – Формат регистра PECCx

Таблица 10.9 – Функциональное назначение полей регистра PECCx

Поле	Биты	Тип	Описание
COUNT	7-0	Чтение Запись Аппаратное влияние	Отсчет PEC передач. Отсчитывает PEC передачи и влияет на действия канала.
BWT	8	Чтение Запись	Выбор типа передачи байт/слово: 0 Передается слово. 1 Передается байт.
INC	10-9	Чтение Запись	Поле управления инкрементом указателей адреса. Изменение SRCPx и DSTPx. 0 Указатели не изменяются. 0 0 Инкремент указателя адреса приемника DSTPx на 1. 1 (BWT = 1_B) или 2 (BWT = 0_B). 1 Инкремент указателя адреса источника SRCPx на 1. 0 (BWT = 1_B) или 2 (BWT = 0_B). 1 Зарезервировано. Не использовать. 1
CL	11	Чтение Запись	Управление каналами PEC: 0 Каналы PEC работают независимо. 1 Каналы PEC объединены попарно.
PLEV	13-12	Чтение Запись	Выбор уровня приоритета PEC.
EOPINT	14	Чтение Запись	Выбор окончания прерывания PEC: 0 Прерывание окончания передачи с тем же самым уровнем приоритета, что и передача. 1 Прерывание окончания передачи обслуживается отдельным узлом прерывания с программируемым уровнем приоритета (EOPIC) и прерыванием, совместно использующим регистр управления (PECISNC).
PT	15	Чтение Запись	Режим передачи: 0 Режим короткой передачи. 1 Режим долгой передачи.

Бит выбора типа передачи байт/слово BWT в регистре PECCx указывает, что байт данных или слово данных будут переданы во время обслуживания PEC, и, в соответствии с этим, выбирает размер шага приращения указателя, который будет изменен при передаче.

Поле управления инкрементом указателей адреса INC регистра PECCx определяет значение приращения указателей PEC после передачи. Если указатели не должны изменяться (т. е. INC = 00_B), соответствующий канал будет перемещать данные из того же самого источника в тот же самый приемник при каждом обслуживании канала PEC.

Режим короткой передачи

Если режим короткой передачи разрешен PT = 0_B в регистре управления PECCx, поле счетчика передачи COUNT управляет действиями соответствующего канала.

Содержимое поля COUNT может разрешить определенное число передач, неограниченное число передач или полное отсутствие PEC обслуживания:

- Если значение счетчика передач PEC установлено в «00_H», PEC передачи запрещены, происходит нормальная обработка запросов на прерывание.

- Если значение счетчика передач PEC установлено в «FF_H», разрешено неограниченное число передач соответствующего канала.

- Если требуется обслужить определенное количество PEC запросов, то после окончания PEC передачи значение счетчика COUNT декрементируется, а флаг запроса PEC обслуживания очищается – это показывает, что запрос был обслужен. Когда значение счетчика устанавливается в 00_H, вырабатывается запрос на прерывание по окончанию PEC передач, имеющий тот же уровень приоритета что и передача (если EOPINT = 0_B), или другой уровень приоритета (EOPINT = 1_B). В момент перехода содержимого счетчика COUNT передач из «01_H» в «00_H», после того как передача данных будет завершена, флаг запроса очищается, если EOPINT будет установлен. Если EOPINT = 0_B, то флаг запроса не очищается, и следующий запрос на прерывание будет обслужен с тем же уровнем приоритета. Если COUNT имеет значение «00_H», то соответствующий канал PEC не будет активирован, а вместо этого будет выставлен запрос на нормальную обработку прерывания.

Режим долгой передачи

Если режим долгой передачи разрешен (PT = 1_B в регистре управления PECCx), то действиями PEC канала управляет поле COUNT2 в регистре PECXCx.

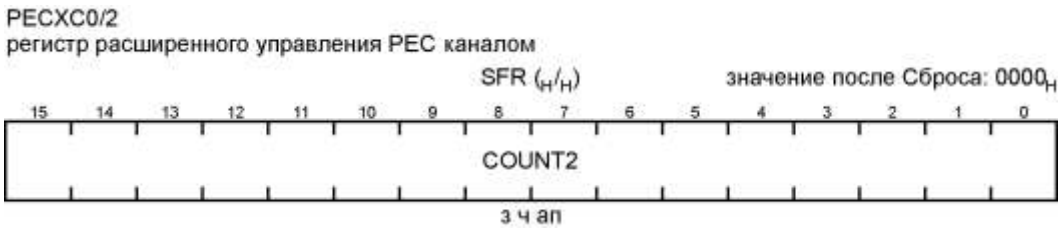


Рисунок 10.11 – Формат регистра PECXC0/2

Таблица 10.10 – Функциональное назначение полей регистра PECXC0/2

Поле	Биты	Тип	Описание
COUNT2	15-0	Чтение Запись Аппаратное влияние	Счетчик расширенного управления PEC каналом

Режим долгой передачи доступен только для выбранных PEC каналов.

Режим независимой работы канала не зависит от режима долгой передачи. Эти два режима могут использоваться совместно. Счетчик передач COUNT в регистре PECCx должен быть установлен в значение 00_H.

Содержимое поля COUNT2 регистра PECXCx определят число передач или отменяет передачи. 16-разрядный счетчик передач разрешает обслуживание до 65535 передач байта или слова (в зависимости от BWT регистра PECCx).

Если значение счетчика передач COUNT2 равно 0000_H, то PEC передачи запрещены, выполняется нормальное обслуживание прерываний.

Если в поле счетчика передач COUNT2 определено конкретное число передач, то после завершения каждой передачи счетчик декрементируется, а флаг запроса очищается, указывая, что запрос обслужен. Когда значение счетчика становится равным 0000h, активизируется запрос прерывания с таким же уровнем приоритета, как и у передачи (EOPINT = 0_B), или с другим уровнем приоритета (EOPINT = 1_B). Когда счетчик COUNT2

переходит из значения 0001_H в 0000_H, после окончания передачи, флаг запроса будет очищен, если EOPINT = 1_B. Если EOPINT = 0_B, то флаг запроса не очищается, а другой запрос на прерывание будет сформирован с тем же самым уровнем приоритета. Если значение счетчика COUNT2 равно 0000_H, то PEC передачи отключены, а вместо этого происходит нормальная обработка прерываний.

Режим связи каналов для передачи цепочки данных

Если режим связи каналов разрешен, то в этом случае два канала объединяются в пару. Передача данных в этом случае разделена на отдельно управляемые поблочные пересылки. Два канала данных, связанные в соединение, позволяют осуществлять поблочную пересылку цепочки данных поочередно друг с другом. По окончании передачи блока данных, которой управляет один канал PEC, связанный с ним канал начнет передачу данных автоматически. Каналы объединяются попарно следующим образом: 0 и 1, 2 и 3, 4 и 5, 6 и 7. Каждым блоком данных управляет один канал пары.

Соединение канала разрешено, если биты CL регистров управления PECCx обоих каналов установлены. Передача данных всегда начинается с четного канала пары. Как только блок данных полностью передан, бит CL регистра управления PECCx ранее активного канала сбрасывается, после чего происходит автоматическое переключение обработки на другой канал пары.

Каждый канал имеет флаг, показывающий ЦПУ окончание передачи PEC. После завершения передачи для возобновления работы канала требуется установить бит CL в регистре управления. Запрос прерывания по окончании PEC передачи индицируется и разрешается в соответствующем регистре управления подузлом прерываний PECISNC или PECXISNC.

Все прерывания завершения передачи управляются регистром EOPIC и имеют один уровень приоритета. Этот регистр прерывания определяет очередность обработки запросов в случае выставления одного или более запросов на прерывания по окончании передачи. Если соответствующие биты прерываний в регистрах управления разрешают обработку запросов, то регистр EOPIC регистрирует приход прерывания.

Если бит CL в регистре управления предыдущего канала обнулен и содержимое счетчика передач (COUNT = 0_B или COUNT2 = 0_B, в зависимости от режима) активного канала так же равно нулю, то передача данных окончена. В этом случае выставляется запрос об окончании PEC передач.

В таблице 10.11 приведены данные о возможных соединениях каналов в пары и очередность запуска каналов внутри каждой пары.

Таблица 10.11 – Объединение PEC каналов в пары и очередность запуска передач

Объединение каналов в пары		Очередность запуска передач
Канал A	Канал B	
0	1	0
2	3	2
4	5	4
6	7	6

Два регистра управления PEC связаны с одним регистром управления прерыванием, поэтому биты приоритета группы обозначены только для четного канала.

Уровень приоритета PEC каналов и арбитраж

Каждому каналу PEC может быть назначен арбитражный уровень приоритета. Формулы и таблица 10.12 позволяют вычислить значение поля PLEV в регистре PECCx для определения уровня приоритета прерываний и групповой уровень приоритета для канала.

PEC канал: x = (x.3, x.2, x.1, x.0). Уровень приоритета прерываний: (1, ~PLEV.1, ~PLEV.0, x.2). Групповой уровень: (x.3, x.1, x.0).

В таблице 10.12 перечислены все возможные комбинации.

Таблица 10.12 – Уровни приоритета и значения соответствующих битов PLEV в регистре управления PECCx

Уровень приоритета		Выбранный PEC канал (x)			
Уровень приоритета ILVL	Уровень группового приоритета xxGP, GLVL	PLEV = 00_B	PLEV = 01_B	PLEV = 10_B	PLEV = 11_B
15	3-0	7-4			
14	3-0	3-0			
13	3-0		7-4		
12	3-0		3-0		
11	3-0			7-4	
10	3-0			3-0	
9	3-0				7-4
8	3-0				3-0

Программирование уровня приоритета прерываний окончания PEC передачи

Программирование уровня приоритета прерываний необходимо для случаев одновременного поступления запросов на прерывание по окончании передач PEC. Запросы прерываний для всех каналов объединяются в один узел прерываний, управление которым осуществляет регистр управления EOPIC (см. таблицу 10.13).

EOPIC
регистр управления прерываниями

bSFR ($F180_H$/ $C0_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	xxGP	EOPIR	EOPIE	ILVL				GLVL	
Ч	Ч	Ч	Ч	Ч	Ч	Ч	З Ч	З Ч ап	З Ч	З Ч				З Ч	

Рисунок 10.12 – Формат регистра EOPIC

Таблица 10.13 – Функциональное назначение полей регистра EOPIC

Поле	Биты	Тип	Описание
xxGP[1)]	8	Чтение Запись	Расширение группового приоритета. Определяет значение старшего бита уровня группового приоритета.
EOPIR[2)]	7	Чтение Запись Аппаратное влияние	Флаг запроса прерывания: 0 Нет запроса прерывания. 1 Источник выставил запрос на прерывание.
EOPIE	6	Чтение Запись	Бит разрешения прерывания: 0 Запрос прерывания разрешен. 1 Запрос прерывания запрещен.
ILVL	5-2	Чтение Запись	Уровень приоритета прерываний: FH Высший уровень приоритета. … 0H Низший уровень приоритета.
GLVL	1-0	Чтение Запись	Уровень группового приоритета: 3H Высший уровень группового приоритета. … 0H Низший уровень группового приоритета.

Окончание таблицы 10.13

[1] Смотри описание уровней группового приоритета. [2] Бит xxIR является защищенным.

Регистр PECISNC и PECXISN содержит флаги узла запроса прерываний по окончанию передачи PEC. Это узел используется в случае использования усовершенствованного приоритета прерываний по завершению передачи PEC, и если бит EOPINT в соответствующем регистре PECCx установлен.

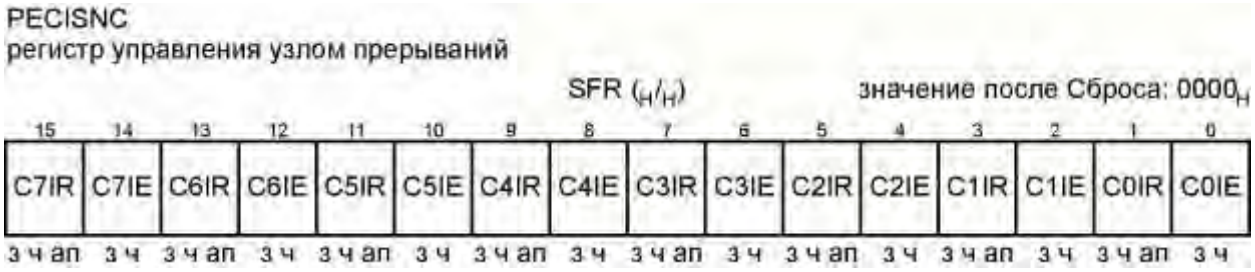


Рисунок 10.13 – Формат регистра PECISNC

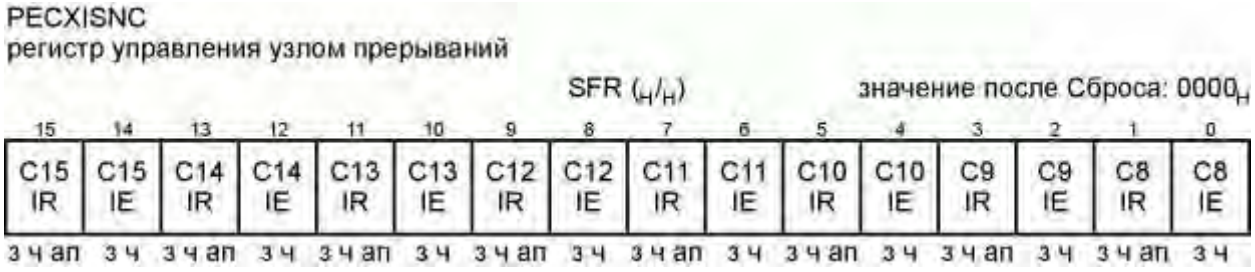


Рисунок 10.14 – Формат регистра PECXISNC

Таблица 10.14 – Функциональное назначение полей регистров PECISNC и PECXISNC

Поле	Биты	Тип	Описание
CxIR	15, 13, 11, 9, 7, 5, 3, 1	Чтение Запись Аппаратное влияние	Флаги запроса узла прерываний PEC канала x [1), 2)]: 0 Нет специальных запросов прерываний для канала x. 1 Установлен флаг запроса прерываний для канала x.
CxIE	14, 12, 10, 8, 6, 4, 2, 0	Чтение Запись	Биты разрешения узловых запросов прерываний для канала x [1), 3)] (индивидуальное разрешение/запрет для определенного источника): 0 Запрос запрещен. 1 Запрос разрешен.

1 x = 15, …, 0.

2 Флаги запросов на прерывания не очищаются аппаратно и должны быть очищены подпрограммой обработки прерываний.

3 Рекомендуется производить очистку флага CxIR перед разрешением соответствующего прерывания. В противном случае ожидающие решения прежние запросы немедленно вызовут запрос на прерывание после установки бита разрешения.

Программный запрос прерываний

Все регистры управления прерываниями разрешают программную запись. Установка флага прерываний в регистре управления прерываниями активирует соответствующий запрос на прерывание, если он разрешен.

Все регистры прерываний, управляемые регистрами прерываний IRQx (смотрите таблицу А.1 векторов прерываний в приложении А), могут быть активированы лишь программно.

Быстрые внешние прерывания

Выборка на выводах быстрых внешних прерываний производится каждый системный такт, таким образом, внешние события могут обнаруживаться в промежуток времени от 1/Fosc. Процесс арбитража и обработки этих прерываний происходит в обычном режиме.

С помощью регистра управления внешними прерываниями EXICON выбирается тип события для внешних прерываний (положительный перепад, отрицательный перепад, оба вида перепадов) отдельно для каждого из восьми быстрых прерываний.

Быстрые внешние прерывания используют вектора прерываний блока CAPCOM каналов CC15 – CC8, поэтому не могут быть использованы функции захвата/сравнения на соответствующих выводах порта 2 (при EXIxES ≠ 00b). Тем не менее, порт 2 может работать в обычном режиме порта.

Выбор источника внешнего прерывания

Входными источниками для каждого из быстрых внешних прерываний (выбирается с помощью регистра EXICON, см. таблицу 10.16) могут стать сигналы с соответствующих выводов порта (стандартный вывод EXnIN или два альтернативных источника). Выбор источников производится с помощью регистров EXISEL0 и EXISEL1, согласно таблице 10.17). Помимо выбора одного из трех возможных источников, два или три из них могут быть логически объединены.

В таблице 10.15 показаны соответствия полей регистра EXISEL (т.е. сигналов прерываний) входам.

Таблица 10.15 – Соединение входных сигналов с выводами внешних прерываний

Поле управления прерыванием	Вывод EXnIN	Альтернативный вывод «A»	Альтернативный вывод «B»	Регистр управления прерыванием
EXI0SS	P2.8	P1H.3	P1H.0	CC8IC
EXI1SS	P2.9	P3.1	P3.0	CC9IC
EXI2SS	P2.10	P3.11	P3.10	CC10IC
EXI3SS	P2.11	P3.13	P3.12	CC11IC
EXI4SS	P2.12	P4.7	P4.5	CC12IC
EXI5SS	P2.13	P4.6	P4.4	CC13IC
EXI6SS	P2.14	P7.7	P7.5	CC14IC
EXI7SS	P2.15	P7.6	P7.4	CC15IC

Таблица 10.16 – Регистр управления быстрыми внешними прерываниями

EXICON

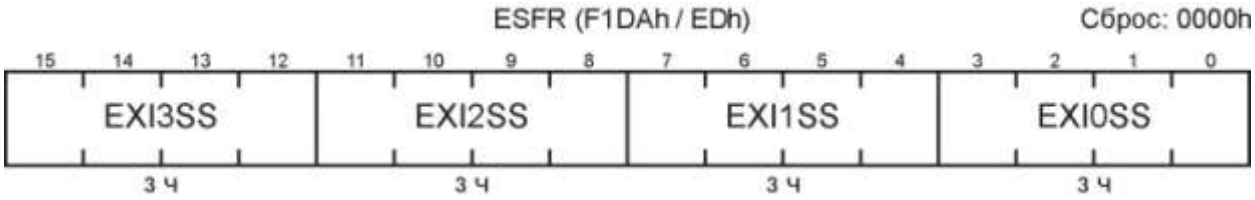
ESFR (F1C0h/ E0h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXI7ES		EXI6ES		EXI5ES		EXI4ES		EXI3ES		EXI2ES		EXI1ES		EX0ES	
3 Ч		3 Ч		3 Ч		3 Ч		3 Ч		3 Ч		3 Ч		3 Ч	

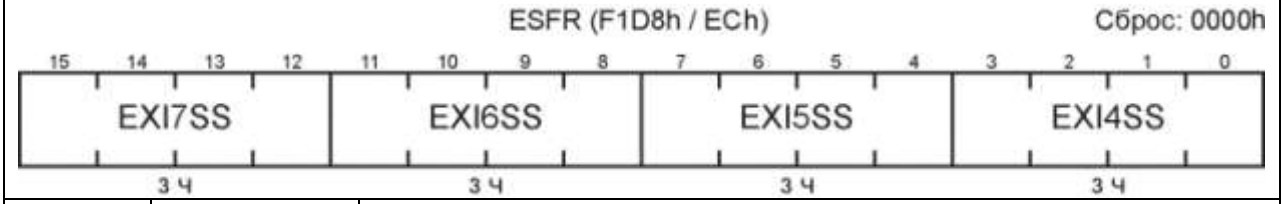
Поле	Бит	Описание	
EXIxES	15–14, 13–12, 11–10, 9–8, 7–6, 5–4, 3–2, 1–0	Поле выбора события для внешнего прерывания с номером x	
		00	Стандартный режим. Внешние прерывания запрещены
		01	Прерывание по положительному фронту
		10	Прерывание по отрицательному фронту
		11	Прерывание по любому фронту

Таблица 10.17 – Регистр выбора источника для каждого внешнего прерывания

EXISEL0



EXISEL1



Поле	Бит	Описание	
EXIxSS	15–12, 11–8, 7–4, 3–0	Поле выбора события для внешнего прерывания x	
		0000	Вход EXxIN
		0001	Альтернативный вход AltA
		0010	Альтернативный вход AltB
		0011	Логическое ИЛИ сигналов со входов EXxIN и AltA
		0100	Логическое И сигналов со входов EXxIN и AltA
		0101	Логическое ИЛИ сигналов со входов AltA и AltB
		0110	Логическое И сигналов со входов AltA и AltB
		0111	Логическое ИЛИ сигналов со входов ExxIN, AltA и AltB
		Другие комбинации зарезервированы	

11 Интерфейс внутренней шины XBUS

Разработка эффективных систем и увеличение производительности при уменьшении числа имеющихся компонентов требуют интеграции специфических периферийных устройств, ориентированных на конкретные встраиваемые системы управления. Такая возможность предоставляется при помощи шины XBUS, используемой в микроконтроллерах второго поколения. XBUS является внутренним представлением интерфейса внешней шины, позволяющим интегрировать новые периферийные устройства без изменения внутренней архитектуры микроконтроллера. Одними из таких периферийных устройств являются модули CAN интерфейса и область внутреннего ОЗУ XRAM .

Для каждого периферийного блока, подключенного к XBUS (X-периферия), имеется отдельное адресное окно, управляемое парой регистров XBCONx/XADRSx (подобно регистрам BUSCONx и ADDRSELx). Поскольку для каждого периферийного блока требуется ограниченное количество регистров, регистры XADRSx выбирают меньшие адресные окна, чем стандартные ADDRSELx. Поскольку регистровая пара управляет объединенной периферией быстрее, чем через внешнее управление, она фиксируется программной маской вместо того, чтобы быть программированной пользователем.

XBUS обеспечивает доступ к X-периферии с разрядностью шины 8 бит или 16 бит. Поскольку внутрикристальное соединение может быть очень эффективным, то для обеспечения этого преимущества поддерживается режим демультиплексной шины. Для интеграции X–периферии предусмотрены узлы прерывания.

Разрешение X-периферии

Доступ к X-периферии также находится под управлением контроллера внешней шины EBC. Во время обращения к внутренней X-периферии, на внешнюю шину выдаются адрес через порт P1 и порт P4 и сигналы управления ALE и BHE#. Сигналы управления чтением RD#, записью WR# / WRL#, BHE# / WRH# и сигналы выборки CSx# переводятся в неактивное состояние (уровень логической 1). Если производится запись по шине XBUS, то на внешнюю шину также выдаются данные через порт P0.

После сброса микроконтроллера RESET запрещены все устройства, подключенные к шине XBUS (X-периферия). X-периферия не может быть использована, если это не разрешено битом глобального разрешения XPEN в регистре SYSCON. Дополнительный к биту XPEN регистр XPERCON определяет, какая периферия разрешена или запрещена. Если периферия запрещена, то ее адреса невидимы. Регистр доступен до выполнения команды EINIT. Выбор XPER регистром XPERCON должен быть выполнен до разрешения X-периферии битом XPEN в регистре SYSCON. Структура регистра XPERCON приведена на рисунке 11.1.

XPERCON
регистр управления X-периферии

XSFR ($F024_H/12_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PER 15	PER 14	PER 13	PER 12	PER 11	PER 10	PER 9	PER 8	PER 7	PER 6	PER 5	PER 4	PER 3	PER 2	PER 1	PER 0
ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ	ЧЗ

Рисунок 11.1 – Формат регистра управления X-периферии XPERCON

Микропроцессор использует биты с 5 по 0. Может быть подключено 6 устройств периферии. Если в бите «1», устройство разрешено, иначе «0» – запрещено. Каждый бит PERx включает сигнал выбора периферийных устройств XCSx.

В микроконтроллере используется четыре внутренних сигнала выбора устройств:

- XCS1 по биту PER0 для интерфейса CAN;
- XCS2 по биту PER1 для первых 2 Кбайт XRAM;
- XCS3 по биту PER2 для вторых 2 Кбайт XRAM;
- XCS4 по биту PER3 для блока ШИМ (CAPCOM6).

Управление доступом по шине XBUS

Размер адресных окон и их расположение определяется регистрами XADRSx. Тип соответствующей шины определяется регистрами XBCONx, смотри рисунок 11.2, таблицу 11.2.

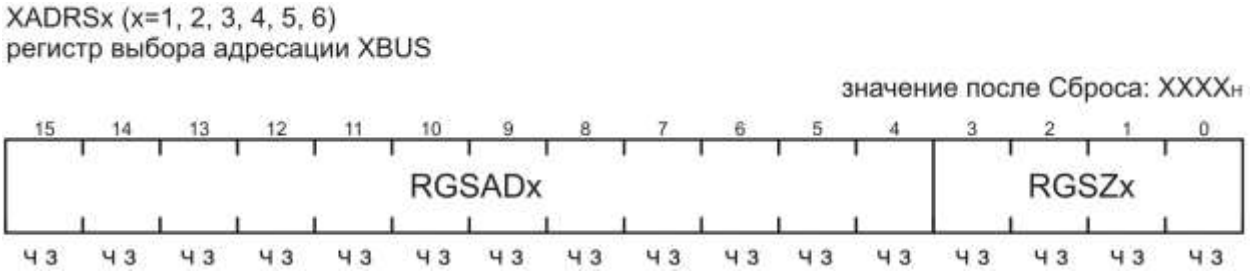


Рисунок 11.2 – Формат регистра XADRSx

В таблице 11.1 представлены регистры XADRSx, адреса которых находятся в области ESFR.

Таблица 11.1 – Адреса регистров XADRSx, x = 1, 2, 3, 4, 5, 6

Название регистра	Адрес в области ESFR
XADRS1	$F014_H$
XADRS2	$F016_H$
XADRS3	$F018_H$
XADRS4	$F01A_H$
XADRS5	$F01C_H$
XADRS6	$F01E_H$

Таблица 11.2 – Функциональное назначение полей регистров XADRSx

Поле	Биты	Тип	Описание
RGSADx	15-4	Чтение Запись	Выбор стартового адреса адресного окна x
RGSZx	3-0	Чтение Запись	Выбор размера адресного окна x. Коды значений для битового поля RGSZx представлены в таблице 11.3.
Примечание – x = 1, 2, 3, 4, 5, 6.			

Таблица 11.3 – Коды значений для битового поля RGSZx

Код в поле RGSZ	Размер адресного окна	Соответствующие (R) биты выбора стартового адреса RGSAD	Стартовый адрес выбранного адресного окна
1	2	3	4
0000	256 байт	RRRR RRRR RRRR	0000 RRRR RRRR RRRR 0000 0000
0001	512 байт	RRRR RRRR RRR0	0000 RRRR RRRR RRR0 0000 0000
0010	1 кбайт	RRRR RRRR RR00	0000 RRRR RRRR RR00 0000 0000
0011	2 кбайта	RRRR RRRR R000	0000 RRRR RRRR R000 0000 0000
0100	4 кбайта	RRRR RRRR 0000	0000 RRRR RRRR 0000 0000 0000

Окончание таблицы 11.3

1	2	3	4
0101	8 кбайт	RRRR RRR0 0000	0000 RRRR RRR0 0000 0000 0000
0110	16 кбайт	RRRR RR00 0000	0000 RRRR RR00 0000 0000 0000
0111	32 кбайта	RRRR R000 0000	0000 RRRR R000 0000 0000 0000
1000	64 кбайта	RRRR 0000 0000	0000 RRRR 0000 0000 0000 0000
1001	128 кбайт	RRR0 0000 0000	0000 RRR0 0000 0000 0000 0000
1010	256 кбайт	RR00 0000 0000	0000 RR00 0000 0000 0000 0000
1011	512 кбайт	R000 0000 0000	0000 R000 0000 0000 0000 0000
11xx	Зарезервировано	–	–
Примечание – x = 1, 2, 3, 4, 5, 6.			

Если используются регистры от XADRS1 до XADRS4, то адреса располагаются в первом мегабайте адресного пространства. Старшие четыре линии адреса A23-A20 удерживаются в нуле. Адресные окна и стартовые адреса для регистров XADRS5 и XADRS6 определяются аналогично внешним устройствам. Регистры XBCONx локализованы в адресном пространстве ESFR.

В таблице 11.4 представлены регистры XBCONx, адреса которых находятся в области ESFR.

Формат регистров XBCONx представлен на рисунке 11.3, функциональное назначение полей данных регистров – в таблице 11.5.

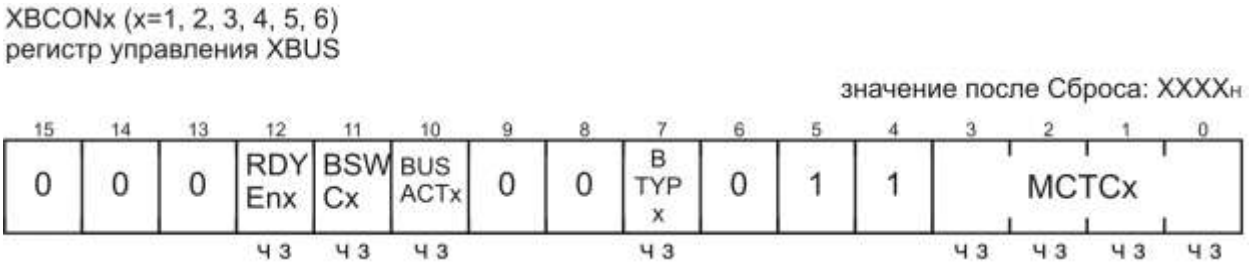


Рисунок 11.3 – Формат регистров XBCONx, x = 1, 2, 3, 4, 5, 6

Таблица 11.4 – Адреса регистров XBCONx, x = 1, 2, 3, 4, 5, 6

Регистр	Адрес
XBCON1	$F114_H$
XBCON2	$F116_H$
XBCON3	$F118_H$
XBCON4	$F11A_H$
XBCON5	$F11C_H$
XBCON6	$F11E_H$

Таблица 11.5 – Функциональное назначение полей регистров XBCONx

Поле	Биты	Тип	Описание
1	2	3	4
MCTCx	3-0	Чтение Запись	Управление циклами памяти.
BTYPx	7	Чтение Запись	Определение типа XBUS: 0 8-битная демультиплексная шина. 1 16-битная демультиплексная шина.

Окончание таблицы 11.5

1	2	3	4
BUSACTx	10	Чтение Запись	Управление активностью XBUS: 0 X-периферия запрещена. 1 X-периферия разрешена. Разрешение XBUS и соответствующего бита XCSx идет в соответствии с адресным окном, выбранным регистром XADRSx.
BSWCx	11	Чтение Запись	Управление переключением BUSCON: 0 Адресное окно переключается немедленно. 1 Вставляется ожидание третьего состояния, если в следующем шинном цикле различные окна управляются этим регистром BUSCON.
RDYEnx	12	Чтение Запись	Разрешение READY: 0 Длина шинного цикла управляется полем MCTC. 1 Длина шинного цикла управляется сигналом READY#.
Примечание – x = 1, 2, 3, 4, 5, 6.			

При каждом обращении к памяти, контроллер внешней шины EBC сравнивает текущий адрес с адресными диапазонами, указанными в ADDRSELx и аппаратно установленными регистрами XADRSx, задающими положение адресных окон X-периферии. Сравнение выполняется в четыре этапа.

Наивысшим приоритетом (приоритет 1) обладают адресные окна DPRAM и X-периферии. Если адрес попадает в диапазон X-периферии, указанный в XADRSx, производится обращение по шине XBUS и регистры ADDRSELx не проверяются. Регистры ADDRSEL2 и ADDRSEL4 (приоритет 2) проверяются раньше регистров ADDRSEL1 и ADDRSEL3 (приоритет 3), соответственно. Проверка регистра ADDRSELx осуществляется, только если адресное окно используется BUSACTx = 1_B. При совпадении адреса с одним из адресных окон, указанных в ADDRSELx, на внешнюю шину выдается текущий адрес (полный 24-разрядный адрес или его младшая часть), и конфигурация шины устанавливается в соответствии с парным регистром BUSCONx. Во время цикла шины формируется соответствующий сигнал выборки CS1#, …, CS4#. Если адрес не попадает в диапазоны, указанные в XADRSx и ADDRSELx, и разрешено использование BUSCON0–CS0# (BUSACT0 = 1_B), то такое обращение обладает самым низшим приоритетом – приоритет 4. При этом текущий адрес (полный или часть) выводится на внешнюю шину, конфигурация которой определяется регистром BUSCON0, и формируется сигнал выборки CS0#.

12 Контроллер внешней шины EBC

Все доступы к внешней памяти выполняются встроенным контроллером внешней шины EBC. Он может быть запрограммирован или в однокристальном режиме, когда не требуется внешняя память, или в одном из четырех различных режимов доступа к внешней памяти:

- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, демультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, мультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, демультиплексная шина;
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, мультиплексная шина.

Режимы шины переключаются динамически, если используются несколько различных адресных окон с различными установками режима.

В режимах демультиплексной шины адреса являются выходными данными порта P1, и данные являются входными/выходными данными порта P0/P0L, соответственно. В режимах мультиплексной шины как адреса, так и данных используют порт P0 для ввода-вывода. Линии адресов (сегментов) высокого порядка используют порт P4. Количество адресных линий активных сегментов выбирается, ограничивая внешнее адресное пространство от 8 Мбайт до 64 Кбайт. Это требуется в том случае, когда интерфейсные линии выделены для порт P4.

До пяти внешних CSx# сигналов (четыре сигнала «выбор окна» плюс сигнал, задаваемый по умолчанию) могут быть созданы для поддержки внешней логики. Внешние модули могут быть прямо подсоединены к общей шине адресов/данных и их отдельным линиям.

Доступ к медленнодействующим запоминающим устройствам или модулям с изменяющимися временами доступа поддерживается через специальную функцию «READY». Активный уровень управляющего входного сигнала разрешает выборку.

Режимами работы внешней шины управляет контроллер внешней шины EBC. С его помощью может быть организовано эффективное использование устройств внешней памяти и периферийных устройств. Быстродействие и интерфейс этих устройств может весьма различаться, поэтому переключение режимов работы внешней шины при обращении к каждому из них выполняется автоматически и не требует программной обработки. Важным достоинством контроллера внешней шины EBC является возможность группировать сходные по интерфейсу внешние устройства в пределах, так называемых, адресных окон, для каждого из которых определяется диапазон адресов и временные параметры внешней шины.

Конфигурация контроллера внешней шины EBC устанавливается при помощи регистров SYSCON, BUSCON0 и четырех пар регистров BUSCONx – ADDRSELx. Содержимое регистров ADDRSELx задает размеры и положение четырех областей памяти – адресных окон. Для каждого из них индивидуально настраивается режим работы внешней шины при помощи соответствующего регистра BUSCONx. Регистром BUSCONx задаются тип шины (мультиплексная/немультиплексная), разрядность шины данных (16 или 8 разрядов), длительность цикла чтения-записи и прочие временные параметры. Как правило, различным адресным окнам соответствуют различные внешние устройства. Режим работы внешней шины при доступе к адресному пространству, не занятому с помощью ADDRSELx, устанавливается регистром BUSCON0 (адресное окно 0).

12.1 Режим работы без внешних устройств

Микропроцессор 1887ВЕ3Т начинает работать в режиме без внешних устройств в том случае, если в момент системного сброса на контакте EA# был зафиксирован сигнал уровня логической 1. При этом в регистр BUSCON0 записывается значение 0000_H. Остальные регистры BUSCON1, …, BUSCON4 всегда после сброса равны 0000_H. Таким образом, контроллер внешней шины EBC отключается.

В этом режиме доступ к внешним устройствам невозможен и используются только внутренние запоминающие устройства микроконтроллера. Поэтому порты P0, P1, P4 и P6 могут использоваться для программного ввода-вывода.

Выполнение программы начинается из внутреннего ПЗУ. Попытка обращения к внешней памяти в этом режиме вызывает аппаратное прерывание ошибочной ситуации ILLBUS. Подробнее описано в разделе 10 «Система прерываний и ловушек».

Обращения по внешней шине можно в дальнейшем разрешить, установив хотя бы один бит BUSACTx в значение 1. После этого контроллер EBC включается, и можно осуществлять чтение и запись в пределах разрешенного адресного окна. Остальные параметры шины также необходимо настроить в соответствующем регистре BUSCONx.

12.2 Режимы работы внешней шины

Интерфейс внешней шины микроконтроллера используется в том случае, если хотя бы в одном из регистров BUSCONx установлен в единичное значение бит BUSACTx. При обращении к адресному окну режим работы внешней шины устанавливается в зависимости от значения битового поля BTYP соответствующего регистра BUSCONx. Возможные значения битового поля BTYP и режимы работы внешней шины приведены в таблице 12.1.

Таблица 12.1 – Тип и разрядность внешней шины

BTYP	Разрядность шины данных	Тип шины
00_B	8	Демультиплексная
01_B	8	Мультиплексная
10_B	16	Демультиплексная
11_B	16	Мультиплексная

Необходимые режимы работы шины для адресных окон устанавливаются в регистрах BUSCON0, …, BUSCON4. Значение BUSCON0 определяется в соответствии с сигналами, считанными из порта P0 во время системного сброса. После сброса содержимое регистра BUSCON0 может быть модифицировано для подстройки временных параметров.

Внутренняя шина адреса всегда использует 24 разряда, т. е. внутреннее адресное пространство микроконтроллера составляет 16 Мбайт. Количество внешних адресных разрядов определяет только внешнее адресное пространство, которое можно адресовать без изменения сигналов выборки. Внешние разряды адреса выводят полный внутренний адрес или его младшую часть. В режиме мультиплексной шины 16 младших разрядов адресной шины A15, …, A0 выдаются через порт P0, в режиме немультиплексной шины – через порт P1. Старшая часть A23, …, A16 выводится через порт P4.

По окончании сброса считывается значение порта P0 для определения системной конфигурации. Считанная из порта P0H информация записывается в регистр RP0H. Количество используемых сигналов выборки и разрядность внешней шины адреса отображаются в битовых полях CSSEL и SALSEL.

Режим несегментированной разрядности внешней шины адреса определяется значением 01_B битового поля SALSEL регистра RP0H. Этот режим ограничивает внешнее адресное пространство до 64 Кбайт на каждый сигнал выборки CSx#. Только младшие 16 разрядов адресной шины обеспечивают доступ к внешним устройствам. В режиме, когда сегментация разрешена, старшая часть адреса A23, …, A16; A19, …, A16 или A17, A16, см. таблицу 12.3, выдается на внешнюю шину через порт P4. Для доступа к отдельным устройствам памяти или периферийным устройствам могут быть использованы сигналы выбора внешних устройств CSx#.

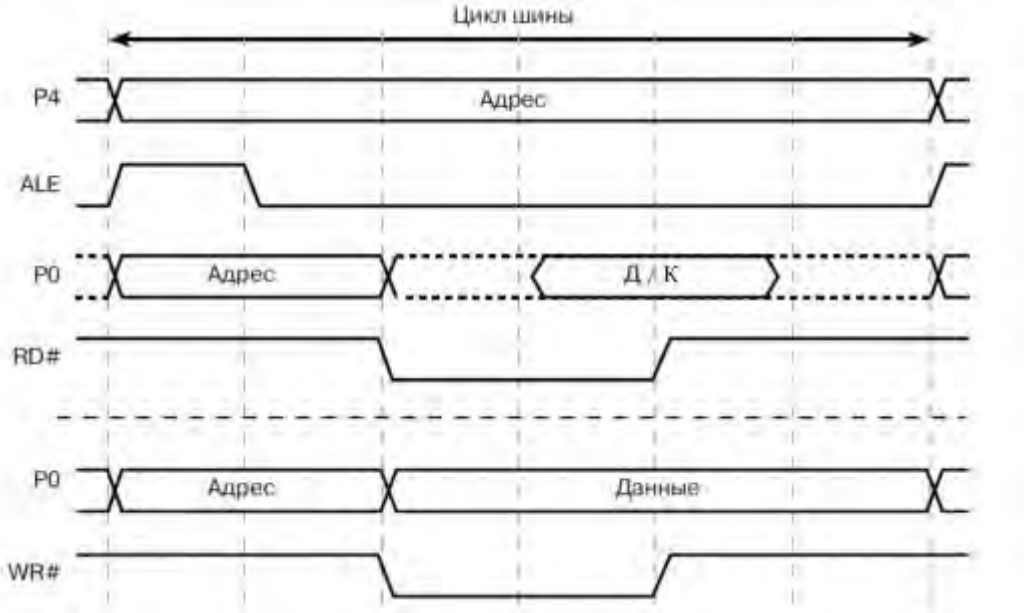
Несегментированная разрядность внешней шины относится только к обращению по внешней шине. Режим несегментированной выборки кода определяет использование регистра CSP для выборки кода (во всем адресном пространстве) и сохранение и

восстановление CSP при входе и выходе из процедур обработки запросов на прерывание. Режим несегментированной выборки кода устанавливается битом SGTDIS регистра SYSCON.

12.3 Мультиплексный режим работы внешней шины

В мультиплексном режиме работы младшие шестнадцать разрядов шины адреса (A15, …, A0) и данные (D15, …, D0 или D7, …, D0) выдаются через порт P0. Адрес формируется в начальной фазе цикла внешней шины и для обеспечения доступа к внешним устройствам должен быть зафиксирован во внешних регистрах сигналом ALE. Разрядность внешних регистров, необходимых для фиксации адреса, зависит от выбранной разрядности шины данных. В режиме 8-разрядной шины данных необходимо зафиксировать во внешнем регистре младшие восемь разрядов адреса A7, …, A0, в то время как старшие разряды A15, …, A8 выдаются через P0H и не мультиплексируются. В 16-разрядном режиме шины данных требуется фиксация всех 16 разрядов адреса, выводимых через порт P0. Старшие разряды адреса (A23, …, A16; A19, …, A16 или A17, A16, см. таблицу 12.3) выдаются на внешнюю шину через порт P4 и не требуют фиксации.

В мультиплексном режиме шина работает следующим образом. Началу цикла соответствует фронт сигнала ALE. Одновременно с этим фронтом в порт P0 поступает внутрисегментная часть адреса, а в порт P4 – старшие разряды адреса, определяющие номер сегмента. По спаду сигнала ALE младшая часть адреса сохраняется во внешнем регистре. Через программируемый период времени, требуемый для сохранения, контроллер внешней шины EBC выводит сигнал управления (RD#, WR#/WRL#, BHE#/WRH#) и выдает данные по шине или принимает их от внешних устройств. Через программируемый промежуток времени данные фиксируются в микроконтроллере фронтом сигнала чтения RD# или во внешнем устройстве фронтом сигнала записи WR#/WRL# или BHE#/WRH#. После окончания чтения внешнему устройству необходимо перевести выводы шины данных в третье состояние. После записи во внешнее устройство данные остаются на внешней шине до начала следующего цикла. Цикл мультиплексной шины приведен на рисунке 12.1.



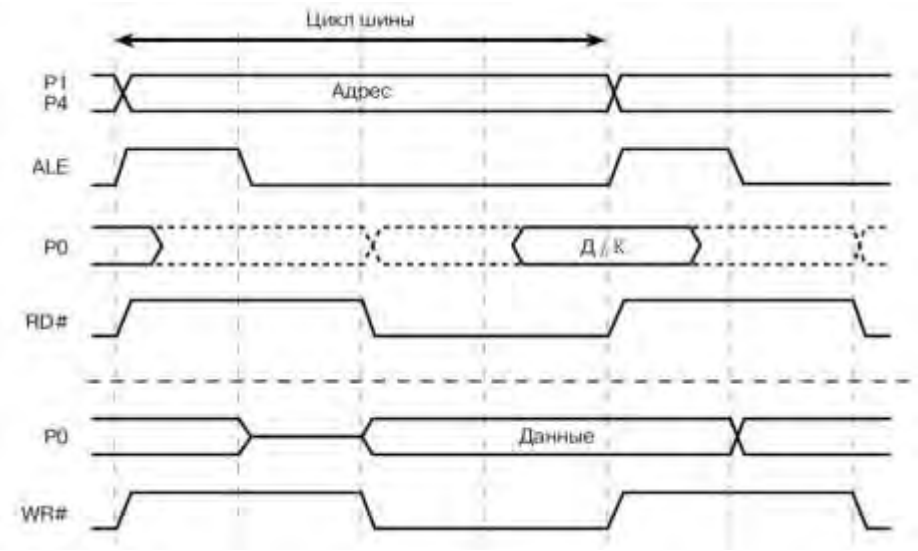
Принятое обозначение: Д/К – Данные/Команда.

Рисунок 12.1 – Цикл мультиплексной шины

12.4 Демультиплексный режим работы внешней шины

В демультиплексном режиме работы младшие 16 разрядов внутрисегментного адреса A15, …, A0 выдаются через порт P1. Данные выдаются через порт P0 (для 16-разрядной шины данных) или только через P0L (для 8-разрядной шины данных). Старшие разряды адреса A23, …, A16; A19, …, A16 или A17, 16, см. таблицу 12.3, выдаются на внешнюю шину через порт P4, см. рисунок 12.2.

Контроллер внешней шины EBC в начале цикла доступа выдает адрес на внешнюю шину. Затем, после программируемого промежутка времени, формирует сигнал управления (RD#, WR#/WRL#, BHE#/WRH#) и выдает данные по шине или принимает их от внешних устройств в зависимости от типа операции (чтение-запись). Через программируемый промежуток времени данные фиксируются в микроконтроллере фронтом сигнала чтения RD# или во внешнем устройстве фронтом сигнала записи WR#/WRL# или BHE#/WRH#. После окончания чтения внешнему устройству необходимо перевести выводы шины данных в третье состояние. После записи во внешнее устройство данные остаются на внешней шине до начала следующего цикла.



Принятое обозначение: Д/К – Данные/Команда.

Рисунок 12.2 – Цикл демультиплексной шины

12.5 Переключение режимов внешней шины

Контроллер EBC позволяет динамически (в процессе работы) переключать режимы внешней шины. Доступ к разным областям внешней памяти может осуществляться с использованием мультиплексной или немультиплексной шин, сигнала готовности READY# или заранее определенными задержками.

Переключение между адресными окнами, границы которых заданы в регистрах ADDRSELx, позволяет изменять режим работы внешней шины и временные параметры (определяются значением соответствующего регистра BUSCONx). Регистром BUSCON0 определяется режим работы внешней шины для адресов, не занятых адресными окнами. Число одновременно возможных режимов работы шины ограничено количеством регистров BUSCONx и равно пяти. При этом способе переключения используются аппаратные средства микропроцессора, и дополнительная программная обработка не требуется.

Перепрограммирование регистров BUSCONx дает возможность управлять режимами работы внешней шины в пределах установленных адресных окон. Значение регистра ADDRSELx задает размер и положение адресного окна, использующего определенный режим шины.

Используя программное управление адресными окнами, можно организовать гораздо больше адресных окон, чем позволяет количество регистров BUSCONx и ADDRSELx, но перезагрузка регистров требует дополнительного времени и некоторого количества памяти для хранения таблиц со значениями.

Необходимо отметить, что если хотя бы в одном из регистров BUSCONx для какого-либо адресного окна установлен немультиплексный режим работы, то внутрисегментная часть адреса A15, …, A0 будет всегда выводиться через порт P1, даже если для доступа к текущему адресному окну используется мультиплексная шина, для которой адрес выводится через порт P0. Это позволяет для разных режимов работы внешней шины использовать одни и те же адресные дешифраторы внешних устройств.

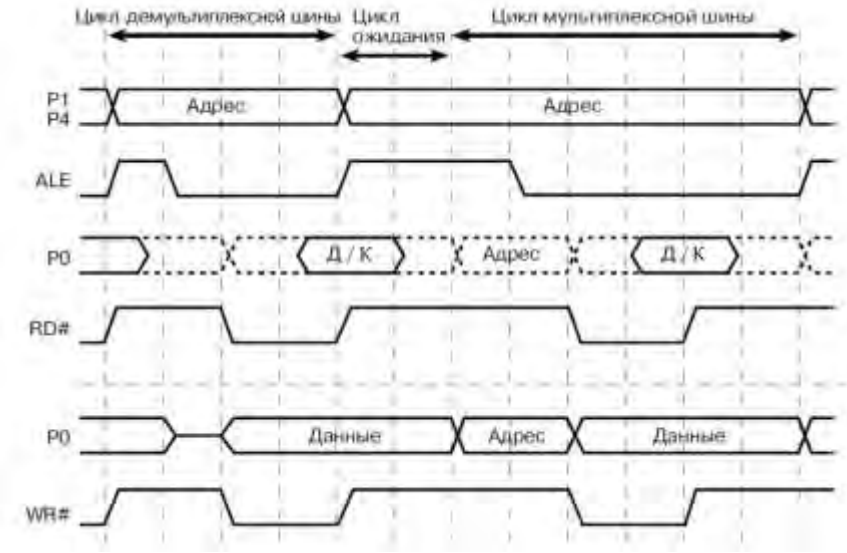
Изменение параметров в регистре BUSCONx (кроме временных) и размера и/или начального адреса в регистре ADDRSELx должны выполняться инструкциями, выборка которых осуществляется из другого адресного окна или внутренней памяти. Допускается изменение временных параметров в регистре BUSCONx с помощью команд, выборка которых производится из того же адресного окна, определяемого BUSCONx–ADDRSELx–CSx#. Однако при этом необходимо следить, чтобы все временные характеристики соответствовали требуемым значениям для используемой внешней памяти или устройства.

Переключение режимов работы внешней шины для различных адресных окон производится автоматически во время работы. После определения полного 24-разрядного адреса инструкции или данных производится выбор разрешенного адресного окна, которому принадлежит указанный адрес. Начальный адрес и размер адресного окна определяется регистром ADDRSELx. Перед обращением к данным или инструкциям осуществляется переключение режима шины в соответствии со значением парного регистра BUSCONx.

12.6 Переключение из демультиплексного в мультиплексный режим

Переключение из демультиплексного в мультиплексный режим представляет собой особый случай. Цикл внешней шины всегда начинается с формирования сигнала ALE и для демультиплексного режима адрес выдается через порты P1 и P4. В мультиплексном режиме младшая часть адреса A15, …, A0 должна выводиться через порт P0. Поэтому при переключении из демультиплексного режима в мультиплексный адресная часть A15, …, A0 будет выдаваться через P0 с задержкой на один машинный цикл. Длительность сигнала ALE также увеличится, см. рисунок 12.3. Задержка сделана для того, чтобы дать внешнему устройству, доступ к которому осуществлялся через демультиплексную шину, дополнительное время для освобождения шины данных. Таким образом, длительность доступа в случае переключения из демультиплексного в мультиплексный режим увеличивается на один машинный цикл.

Адресные окна обычно используются для обращения к различным модулям внешней периферии. Переключение между адресуемыми устройствами может вызвать конфликт на шине в результате того, что для какого-либо внешнего устройства может потребоваться дополнительное время для отключения шинных буферов. В таких случаях в микроконтроллере предусмотрена вставка одного дополнительного машинного цикла, аналогично дополнительному циклу при переключении из демультиплексного в мультиплексный режим, см. рисунок 12.3. Вставка дополнительного машинного цикла контролируется битом BSWCx в регистре BUSCONx для адресного окна, из которого производится переключение в другое адресное окно. Разрешение вставки BSWCx = 1_B не влияет на быстродействие, если производится последовательное обращение к одному и тому же адресному окну.



Принятое обозначение: Д/К – Данные/Команда.

Рисунок 12.3 – Переключение от демультиплексной шины к мультиплексной

12.7 Разрядность шины данных

Возможность работы с 8- и 16-разрядными внешними устройствами обеспечивает контроллер внешней шины EBC. В 16-разрядном режиме для шины данных используются все каналы порта P0, в 8-разрядном – только младшая часть P0L порта P0. Использование 8-разрядной шины позволяет сократить количество внешних регистров, шинных буферов и запоминающих устройств, что в конечном итоге приведет к уменьшению стоимости и габаритов. Однако быстродействие системы также сократится.

Контроллер внешней шины обеспечивает доступ к 16- и 8-разрядным данным независимо от установленной разрядности. В 8-разрядном режиме доступ к словам осуществляется за два цикла шины. Сначала выполняется обращение к младшему байту, затем к старшему. Разделение слов на байты при записи и их слияние при чтении осуществляется контроллером внешней шины EBC без вмешательства ЦПУ и программы.

Запись байта по 16-разрядной шине данных выполняется независимо в младшую и старшую части. Обращение может осуществляться двумя способами. Первый способ: старший байт выбирается сигналом BHE#, тогда как младший байт – адресным сигналом A0. Оба байта могут пересылаться во внешнее устройство независимо друг от друга или вместе в 16-разрядном режиме.

Второй способ используется в режиме записи байта во внешнее 16-разрядное устройство, которое имеет один вход для сигнала выборки CS# и два сигнала для разрешения записи старшего и младшего байтов. Контроллер EBC может самостоятельно сформировать оба сигнала записи. Это позволяет сократить внешнюю логику, объединяющую сигнал WR# с сигналами BHE# и A0.

Бит WRCFG в регистре SYSCON позволяет определить функции сигналов WR# и BHE#. В случае если бит WRCFG установлен в единичное значение, канал WR# будет использоваться для записи младшего байта (сигнал WRL#), а канал BHE# – для записи старшего байта (сигнал WRH#). Во время записи байта (старшего или младшего) во внешнее 16-разрядное устройство, выдаваемый байт дублируется на обе части шины данных.

Чтение байта из внешнего 16-разрядного устройства отличается от записи одиночного байта. Контроллер EBC считывает 16-разрядное слово и после этого выделяет необходимый байт. Данную особенность работы необходимо иметь в виду при работе с устройствами, которые изменяют свое состояние после операции чтения (например,

FIFO). В этом случае доступ к отдельным байтам должен осуществляться с помощью сигналов BHE# и A0.

В таблице 12.2 приведен коэффициент увеличения времени доступа по отношению ко времени обращения по 16-разрядной демультиплексной шине к 16-разрядным данным.

Таблица 12.2 – Коэффициент увеличения К времени доступа к данным

Тип шины	Доступ к 8-/ 16-/ 32-разрядным данным, фактор скорости	Свободные каналы ввода-вывода
8-разрядная мультиплексная	К = 1,5/3/6, очень низкая	P1H, P1L
8-разрядная демультиплексная	К = 1/2/4, низкая	P0H
16-разрядная мультиплексная	К = 1,5/1,5/3, высокая	P1H, P1L
16-разрядная демультиплексная	К = 1/1/2, очень высокая	–

12.8 Использование сигнала BHE#

Сигнал BHE# автоматически разрешается (BYTDIS = 0_B) для контроллера EBC, если во время сброса была выбрана 16-разрядная конфигурация шины данных. В режиме старта с использованием 8-разрядной внешней шины бит BYTDIS равен 1, и вывод сигнала BHE# запрещается. В этом случае канал P3.12 может использоваться как стандартный вывод. После выполнения выхода на старт (RESET) функция BHE# автоматически разрешена (BYTDIS = 0_B), если выбрана 16-битная шина данных. Иначе – запрещена (BYTDIS = 1_B). Эта функция может быть запрещена, если байтовый доступ к 16-разрядной памяти не нужен и если сигнал BHE# не используется. Сигнал BHE# обычно используется для выбора одной из двух 8-разрядных микросхем памяти, которые подключены к микроконтроллеру через 16-разрядную шину данных.

Во время доступа к внешней памяти контроллер EBC выдает младшие шестнадцать разрядов адреса A15, …, A0 через порт P0 или порт P1 (в зависимости от режима работы внешней шины), при этом старшая часть адреса A23, …, A16; A19, …, A16 или A17, 16, см. таблицу 12.3, выдается через порт P4. Количество разрядов старшей части адреса, используемых для организации внешней шины, устанавливается во время сброса микроконтроллера и отображается в битовом поле SALSEL регистра RP0H, см. таблицу 12.3.

Таблица 12.3 – Выбор количества разрядов старшей части адреса

SALSEL	Сегментная часть адреса	Линейное адресное пространство
11_B	2: A17, A16	256 Кбайт
10_B	8: A23, A22, A21, A20, A19, A18, A17, A16	16 Мбайт
01_B	-	64 Кбайт
00_B	4: A19, A18, A17, A16	1 Мбайт

Размер адресуемой внешней памяти может увеличиваться за счет использования нескольких банков, каждый из которых выбирается сигналами CSx# (x = 0, …, 4) и другими сигналами (например, каналами ввода-вывода).

12.9 Сигналы выборки внешних устройств CSx#

Во время доступа к внешним устройствам контроллер EBC выводит через каналы порта P6 сигналы выборки внешних устройств CS0#, …, CS4#, которые могут непосредственно выбирать внешнюю периферию или банки памяти без внешнего адресного дешифратора, см. рисунок 9.42. Количество используемых каналов CSx#

устанавливается во время сброса и отображается в битовом поле CSSEL регистра RP0H, см. таблицу 12.4.

Таблица 12.4 – Количество выбранных каналов CSx#, x = 0, …, 4

CSSEL	Количество сигналов выборки	Используемые каналы порта P6	Свободные каналы порта P6
00_B	3: CS0#, CS1#, CS2#	P6.0, P6.1, P6.2	P6.3, …, P6.7
01_B	2: CS0#, CS1#	P6.0, 6.1	P6.2, …, P6.7
10_B	не используются	-	P6.0, …, P6.7
11_B	5: CS0#, …, CS4#	P6.0, …, P6.4	P6.5, P6.6, P6.7 (без pull-down резисторов)

Каждый выход CSx# переводится в активное состояние (уровень логического 0) во время доступа в пределах адресного пространства, определяемого соответствующей парой регистров BUSCONx и ADDRSELx. Каждый из сигналов CSx# активизируется в начале цикла внешней шины, все другие сигналы выборки в это время переводятся в неактивное состояние – уровень логической 1. Сигналы CSx# остаются неизменными до тех пор, пока не потребуется доступа к другому адресному окну: чтение или запись данных/ инструкций.

Сигналы CSx# не изменяют своих значений во время доступа в пределах внутренней области памяти, т. е. когда нет обращения к внешним устройствам, даже если эта область памяти перекрывает адресное окно внешней памяти. При доступе к внутренней X-периферии интерфейс XBUS переводит все сигналы выборки CSx# на уровень логической 1 (неактивное состояние).

Режим работы каждого из сигналов CSx# устанавливается при помощи битов CSWENx и CSRENx соответствующего регистра BUSCONx, см. таблицу 12.5.

Таблица 12.5 – Режим генерации сигналов CSx#

CSWENx	CSRENx	Режим вывода CSx#
0	0	Формирование для всего цикла шины
0	1	Формирование на время активности RD#
1	0	Формирование на время активности WR#/ WRL#
1	1	Формирование на время активности RD#, WR#/WRL# и BHE#/WRH#

В режиме формирования для всего цикла шины, сигнал CSx# остается активным до начала обращения к другому адресному окну. В этом случае сигнал выборки переходит в активное состояние (уровень логического 0) по спаду сигнала ALE и остается активным до спада сигнала ALE в цикле внешней шины, который обращается к другому адресному окну. Если обращения в соседних циклах шины производятся в одно и то же адресное окно, то сигнал CSx# остается активным без изменения во время спада ALE.

В микроконтроллере можно активировать сигнал выборки вместе с фронтом ALE, выбор момента изменения CSx# при этом определяется значением бита SYSCON.6 (CSCFG). При CSCFG = 0_B сигнал CSx# (защелкнутый сигнал выбора кристалла) становится активным по отрицательному фронту ALE и становится неактивным с началом цикла внешней шины с доступом к различным адресным окнам.

Никакие изменения не происходят с CSx#, если адрес находится в пределах собственного окна или во внутренней памяти (исключая устройства, подключенные к XBUS). При CSCFG = 1_B сигнал CSx# (ранний сигнал выбора кристалла) становится активным вместе с адресом и с сигналом BHE# (если разрешено) и остается активным до конца текущего цикла. В данном случае CSx# не защелкивается и может немедленно переключиться при изменении адреса.

В режимах формирования на время активности чтения RD# или записи WR#/WRL# и BHE#/WRH# сигнал CSx# находится в активном состоянии (уровень логического 0), пока активен соответствующий сигнал управления. Если устанавливается этот режим только для чтения, то CSx# формируется только при чтении, а при записи сигнал выборки активен для всего цикла шины. Режим формирования только для записи аналогичен режиму чтения.

При старте из внешней памяти после сброса (во время сброса напряжение на EA# соответствует уровню логического 0) для сигнала CS0# устанавливается режим формирования для всего цикла шины.

Во время сброса внутренние подтягивающие цепи удерживают сигналы CSx# в состоянии логической 1. После сброса подтягивающие цепи отключаются, и состояние выходов CSx# определяется буферами соответствующих каналов. Каналы, предназначенные для вывода сигналов CSx#, но незадействованные при работе в режиме выборки внешних устройств, переводятся в третье состояние и могут использоваться для программного ввода-вывода.

12.10 Разрядность шины адреса и сигналы выборки

Интерфейс внешней шины микроконтроллера позволяет работать с различными конфигурациями внешней памяти. В зависимости от количества используемых разрядов адреса, внешнее адресное пространство может составлять 64 Кбайта, 256 Кбайт, 1 Мбайт или 16 Мбайт. Сигналы выборки внешних устройств CSx# позволяют подключать банки памяти и периферию к микроконтроллеру без дополнительной внешней логики.

Если на внешнюю шину выводится только младшая часть внутреннего адреса, то старшие разряды выполняют функцию дешифратора адресных окон при соответствующей настройке регистров ADDRSELx. Тогда при адресации (выборка инструкций или чтение-запись данных) старшая часть полного внутреннего адреса определяет адресное окно, для которого формируется соответствующий сигнал выборки CSx#.

Например, устанавливая четыре разряда для сегментной части адреса A19, …, A16 и используя пять сигналов CSx# для выборки внешних устройств, можно организовать пять банков внешней памяти по 1 Мбайт каждый. Начальные адреса окон необходимо разнести друг от друга не менее чем на 1 Мбайт. Общий размер адресуемой внешней памяти при этом составит 5 Мбайт.

12.11 Программируемые временные параметры внешней шины

В микроконтроллере основные временные характеристики могут задаваться программно. Этим достигается возможность работы микроконтроллера с различными типами периферийных устройств и устройств внешней памяти.

Следующие параметры цикла внешней шины программируются:

- Длительность сигнала ALE и время удержания адреса на шине после отрицательного фронта ALE.
- Длительность циклов памяти (с расширением от 1 до 15 TCL) определяет необходимое время доступа.
- Длительность времени высокоимпедансного состояния на шине (с расширением на 1 TCL).
- Задержка сигналов чтения и записи после отрицательного фронта ALE.
- Использование сигнала готовности внешнего устройства READY#.

Программируемые интервалы цикла внешней шины приведены на рисунке 12.4.

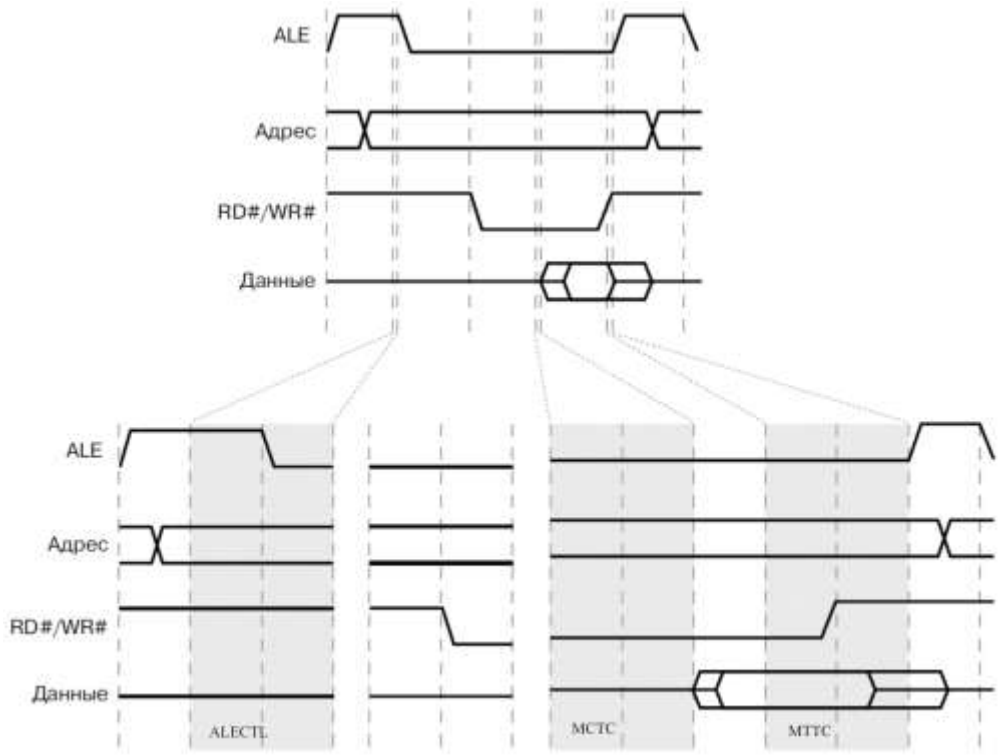


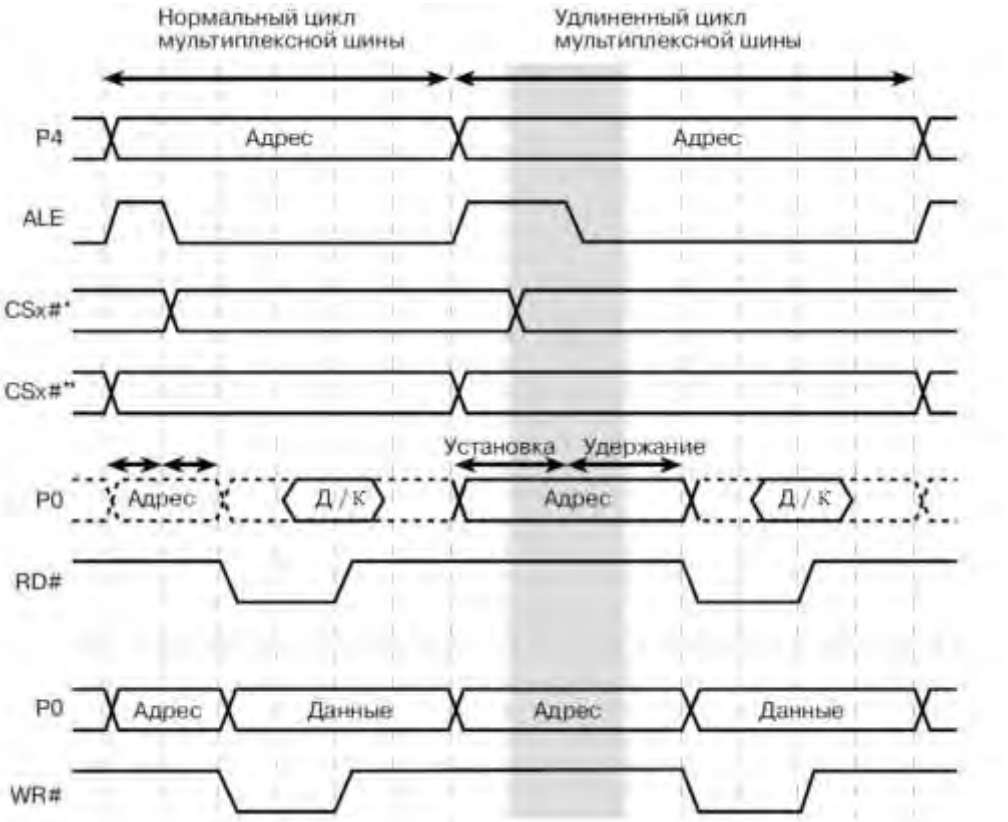
Рисунок 12.4 – Программируемые временные параметры циклов шины

Выполнение программы из внутреннего ПЗУ осуществляется с максимальной скоростью, при этом время доступа не программируется. После сброса микроконтроллера для внешнего шинного интерфейса устанавливается самый медленный цикл шины. Параметры внешней шины могут быть переустановлены в подпрограмме начальной инициализации.

Длительность сигнала ALE и времени удержания адреса

Длительность сигнала ALE и времени удержания адреса после отрицательного фронта сигнала ALE управляется битами ALECTLx в регистрах BUSCONx. Когда бит ALECTL установлен в 1, длительность сигнала ALE увеличивается на половину периода сигнала тактового CPU (1TCL). Время удержания адреса для мультиплексной шины при ALECTLx = 1_B увеличивается на 1TCL после отрицательного фронта сигнала ALE. Передача данных при этом задерживается на один период системного тактового сигнала CLKOUT (2TCL), чтобы она начиналась по тому же фронту тактового сигнала. После сброса (Reset) бит ALECTL0 устанавливается в единичное значение для обеспечения самого медленного цикла шины при обращении к адресному окну 0 (BUSCON0-CS0#). Регистры ALECTL1, …, ALECTL4 после сброса обнуляются.

Нулевое значение бита CSCFG (значение после сброса) программирует контроллер EBC на формирование отрицательных фронтов всех сигналов выборки CSx#, …, CS4# через один TCL после фронта ALE CSx#*, см. рисунок 12.5. При этом захват адреса осуществляется всеми внешними устройствами, подключенными к внешней шине. Формирование отрицательных фронтов сигналов CSx# одновременно с фронтом сигнала ALE определяется единичным значением бита CSCFG, что позволяет внешнему устройству, выбранному сигналом CSx#, захватить адрес по сигналу ALE CSx#**, см. рисунок 12.5.



Принятое обозначение: Д/К – Данные/Команда.

_____

* , ** – смотри текст на предыдущей странице «Длительность сигнала ALE и времени удержания адреса».

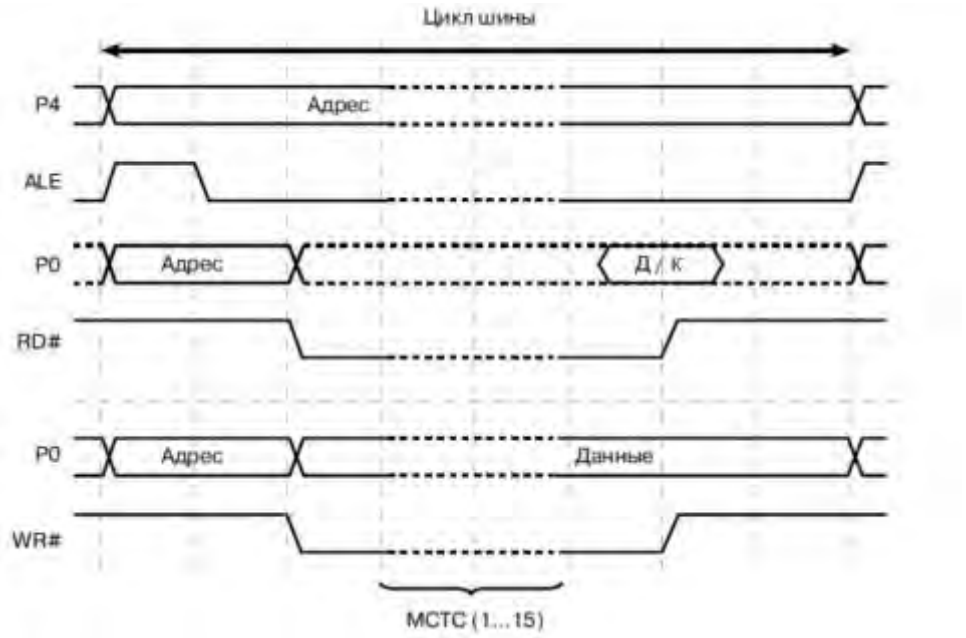
Рисунок 12.5 – Цикл сигнала ALE

Длительность циклов чтения и записи

Контроллер внешней шины позволяет регулировать время пересылки данных к внешним периферийным устройствам во время записи и от внешних устройств во время чтения, см. рисунок 12.6. Увеличение времени циклов может потребоваться для работы с более медленными внешними устройствами или памятью. Во время цикла чтения и записи остаются неизменными все сигналы внешней шины.

Изменение времени пересылки производится за счет введения дополнительных тактов задержки при передаче данных для обращения к тому или иному адресному окну. Число дополнительных тактов устанавливается в битовом поле MCTC соответствующего регистра BUSCONx. Во время дополнительных тактов ЦПУ находится в режиме ожидания, если завершение пересылки данных требуется для выполнения текущей инструкции.

Количество дополнительных тактов задержки программируется от 0 до 15 изменением значения битового поля MCTC в регистре BUSCONx и вычисляется как разность числа 15 и значения поля MCTC. Значение битового поля MCTC, равное 0000_B, соответствует максимальной задержке в 15 тактов. Если MCTC равно 1111_B, то чтение и запись выполняются без дополнительных задержек. Один такт задержки равен одному периоду тактового сигнала CPU (2TCL). После сброса битовые поля MCTC всех регистров BUSCONx устанавливаются в нулевое значение, что соответствует максимальной задержке.



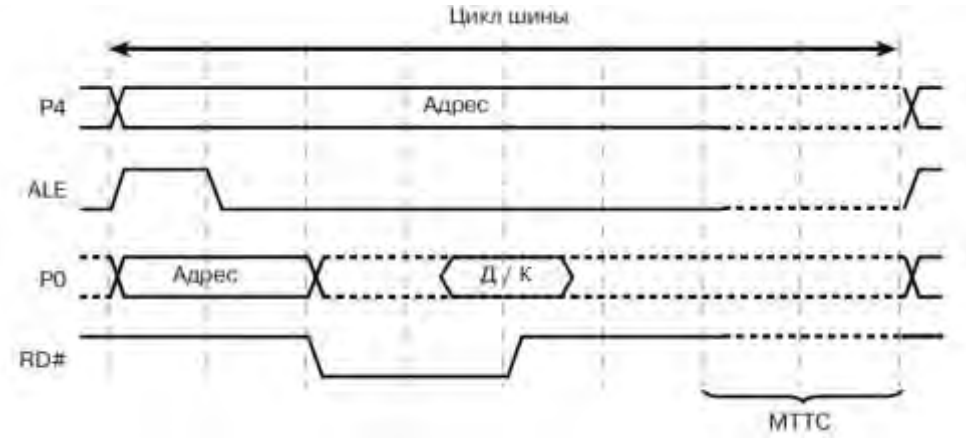
Принятое обозначение: Д/К – Данные/Команда.

Рисунок 12.6 – Длительность циклов чтения и записи

Длительность освобождения шины внешним устройством

Время задержки после окончания импульса чтения также можно программировать, если внешнему устройству необходимо время для перевода выходного буфера в третье состояние до начала следующего цикла. Время задержки устанавливается на один период сигнала тактового сигнала ЦПУ 2TCL = 50 нс и управляется битом MTTCx регистра BUSCONx.

Как показано на рисунке 12.7, циклы чтения и записи будут завершаться с дополнительной задержкой, если MTTCx = 0_B – значение по умолчанию после сброса микроконтроллера.



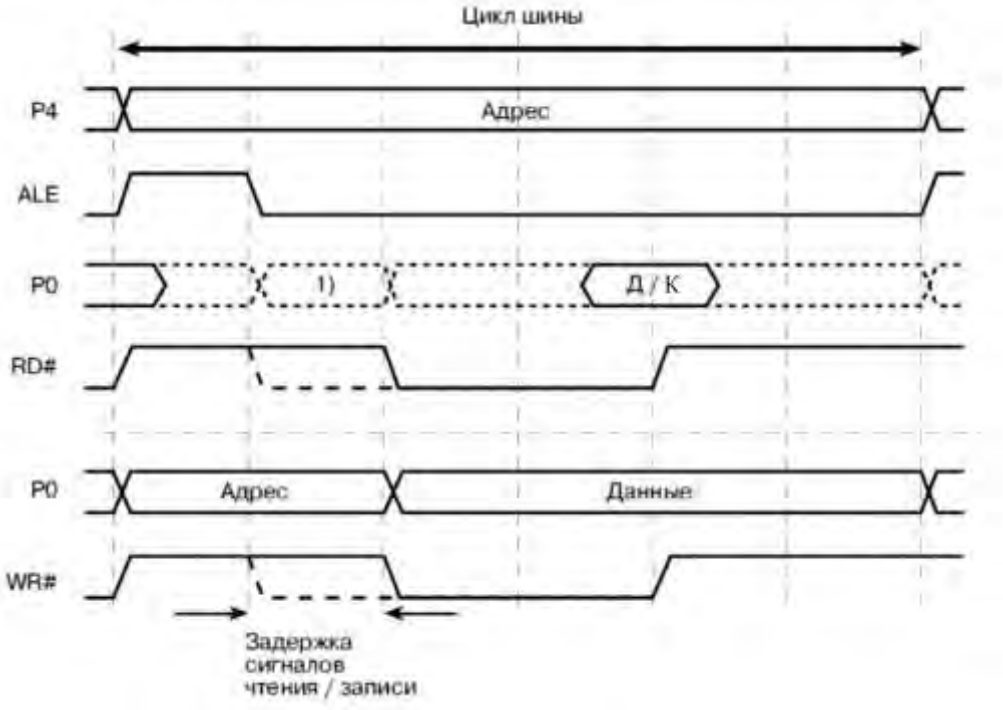
Принятое обозначение: Д/К – Данные/Команда.

Рисунок 12.7 – Время задержки после чтения

Во время этой задержки ЦПУ не переходит в режим ожидания, а продолжает выполнение программы. Однако если последующие циклы чтения снова обращаются к этому же адресному окну, то выполнение программы замедлится. В мультиплексном режиме работы шины, независимо от значения бита MTTCx, добавляется еще задержка на один период сигнала тактового сигнала ЦПУ 2TCL.

Задержка сигналов чтения и записи

Задержка сигналов чтения/записи представлена на рисунке 12.8.



Принятое обозначение: Д/К – Данные/Команда.

[1)] Шинные буферы внешнего устройства должны отключаться от шины до начала активного уровня сигнала чтения RD#.

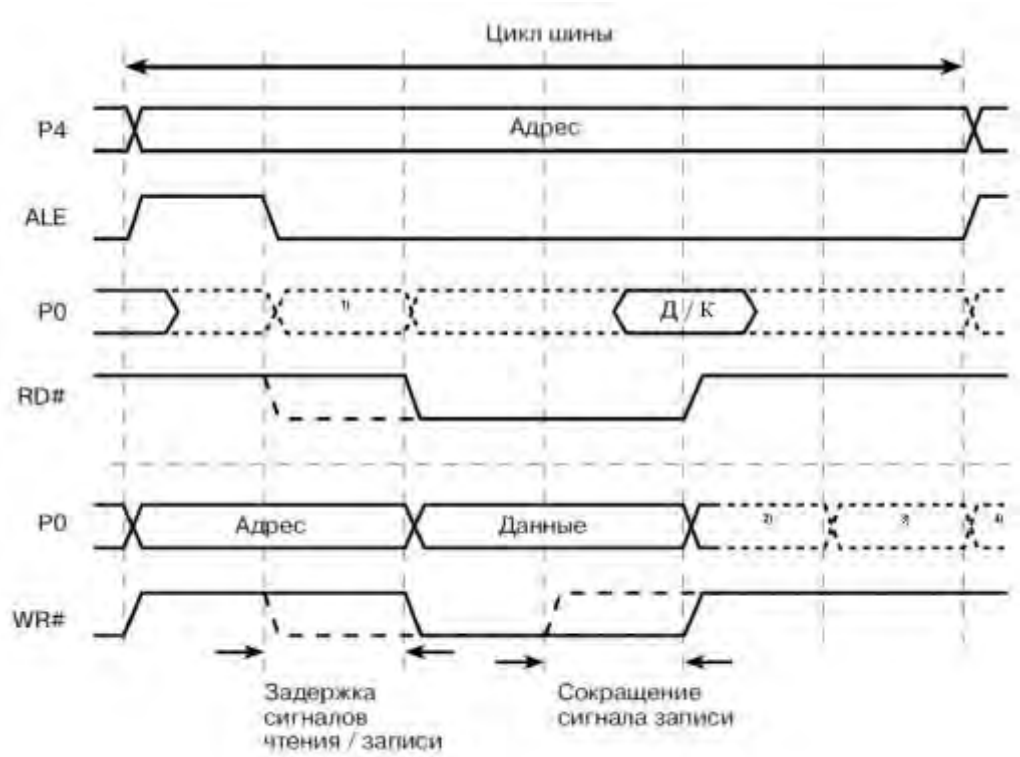
Рисунок 12.8 – Задержка сигналов чтения /записи

Настройки контроллера внешней шины EBC позволяют задать параметры диаграмм обращения чтения/записи с учетом временных характеристик внешней периферии, см. рисунок 12.8. Задержка сигналов записи и чтения устанавливается между задним фронтом сигнала ALE и задним фронтом сигнала чтения RD# или записи WR#/WRL#, BHE#/WRH#. Без задержки BUSCONx.RWDCx = 1_B задний фронт сигналов чтения и записи совпадает с задним фронтом сигнала ALE. Задержка, равная 1TCL, устанавливается при обнулении RWDCx. Данная задержка не увеличивает общее время доступа и не уменьшает быстродействие микроконтроллера. После сброса все биты RWDCx обнуляются. В мультиплексных режимах работы данные с выхода внешнего устройства могут конфликтовать с адресом, выдаваемым микроконтроллером, если сигнал чтения формируется без задержки. Поэтому в мультиплексных режимах работы рекомендуется иметь задержку для сигналов чтения и записи.

Раннее завершение сигнала записи

Продолжительность внешнего доступа записи может быть сокращена на 1TCL. Сигнал WR# активируется (низкий уровень) в стандартном режиме, однако может деактивироваться на 1TCL раньше, чем определено стандартной временной диаграммой, см. рисунок 12.9. В этом случае выходной драйвер деактивируется на 1TCL раньше. Этот режим используется системой для работы с более высокой частотой и применяется для внешних модулей (память, периферия и т. д.), чтобы быстро переключать собственные выходы в соответствии, например, с CSx# сигналами. Конфликта между микроконтроллером и выходами внешней периферии можно избежать выбором раннего сигнала WR# (на 1TCL). Однако надо удостовериться, что ранний сигнал WR# отвечает требованиям внешней периферии.

Деактивация раннего сигнала WR# управляется битом EWENx регистров BUSCON. Сигнал WR# будет укорочен, если EWENx = 1_B. После старта микроконтроллера EWENx = 0_B и сигнал WR# работает в стандартном режиме.



Принятое обозначение: Д/К – Данные/Команда.

[1)] Выходные буферы внешнего устройства должны быть переведены в третье состояние до начала формирования сигнала RD#.

[2)] Выходные буферы микроконтроллера переводятся в третье состояние при длительности записи.

[3)] Выходные буферы микроконтроллера переводятся в третье состояние в демультиплексном режиме без сокращения длительности записи.

[4)] Выходные буферы микроконтроллера переводятся в третье состояние в мультиплексном режиме без сокращения длительности записи.

Рисунок 12.9 – Завершение формирования сигнала записи

Управление шинным циклом сигналом READY#

Если программируемого времени задержки недостаточно или если время доступа периферии не постоянно, то следует использовать входной сигнал готовности READY# от внешнего устройства, информирующий контроллер внешней шины о завершении циклов чтения и записи. В этом случае контроллер EBC сначала ожидает окончания программируемой задержки 0, …, 7 тактов, а затем отслеживает сигнал READY# для определения необходимости завершения цикла внешней шины. Внешнее устройство должно установить сигнал READY# в состояние логического 0 после того, как данные для чтения выданы на шину или данные для записи были сохранены.

Функция сигнала READY# разрешается битом RDYENx в регистрах BUSCON. Когда эта функция выбрана RDYENx = 1_B, только соответствующие младшие три бита MCTC определяют число вставленных циклов ожидания от 0 до 7 TCL. Как показано на рисунке 12.10, асинхронный сигнал READY# требует добавочные циклы ожидания в соответствии с внутренней синхронизацией. Асинхронный сигнал READY# имеет

внутреннюю синхронизацию, и запрограммированные циклы ожидания необходимы для обеспечения параметров шинных циклов. Асинхронный сигнал READY#, активированный каким-то внешним устройством, может быть деактивирован передним фронтом сигнала WR# или RD#. Когда функция READY# разрешена для определенного адресного окна, каждый шинный цикл должен быть завершен при помощи активного уровня сигнала READY#, в противном случае, микроконтроллер останавливает свою работу до прихода следующего аппаратного сброса RSTIN# или сброса после переполнения сторожевого таймера WDT.

Комбинированная функция сигнала READY# с предопределенными циклами ожидания полезна в двух случаях. Компоненты памяти с фиксированным временем доступа и периферия с READY# могут быть сгруппированы в одно адресное окно. Внешняя логика в этом случае должна перевести READY# в активное состояние во время выборки памяти или когда периферия сообщает о своей готовности. После программируемой задержки контроллер внешней шины опрашивает вход READY# для определения окончания цикла шины. При обращении к памяти сигнал READY# уже будет в активном состоянии – цикл шины с ранним READY#, см. рисунок 12.10, а для доступа к периферии переход в активное состояние может быть задержан – цикл шины с поздним READY#, см. рисунок 12.10. Обычно внешняя память является более быстродействующим устройством, чем периферия, поэтому использование сигнала READY# не должно оказывать влияния на скорость выполнения программы.

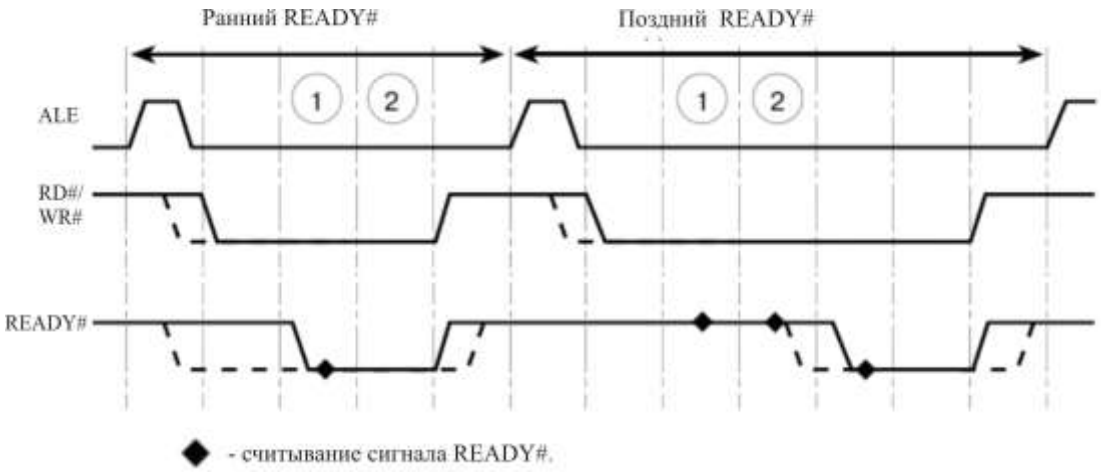


Рисунок 12.10 – Циклы шины с использованием сигнала READY#

Использование периферии с так называемым «нормальным сигналом готовности» может привести к ошибочному завершению цикла шины, если сигнал READY# опрашивается слишком рано. Это происходит в том случае, когда периферия переводит сигнал готовности в активное состояние на время ожидания, а неактивное состояние соответствует циклу обращения к периферии. В этом случае, если периферия переведет сигнал READY# в неактивное состояние после того, как микроконтроллер опросил состояние входа, то шинный цикл ошибочно закончится раньше действительного окончания. Использование программируемой задержки позволяет переместить опрос сигнала готовности на время, необходимое внешней периферии для перевода сигнала READY# в неактивное состояние, например, на два такта, смотри рисунок 12.10.

12.12 Настройка контроллера внешней шины

Контроллер внешней шины EBC управляется набором специальных регистров, расположенных в области SFR.

Регистр SYSCON позволяет задавать режим работы с сегментированной выборкой кода, конфигурацию внутреннего ПЗУ и конфигурацию выводов WR# и BHE#.

Регистрами BUSCON1, …, BUSCON4 устанавливаются параметры внешней шины при обращении к адресным окнам: использование сигнала готовности READY#, длительность сигнала ALE, мультиплексный или демультиплексный режимы работы внешней шины, разрядность шины данных, задержку сигналов чтения/записи и их длительность. Регистр BUSCON0 определяет временные параметры во время доступа к адресному пространству, не занятому адресными окнами (к адресному окну 0).

Регистры ADDRSEL1, …, ADDRSEL4 связаны с соответствующими регистрами BUSCON1, …, BUSCON4 и позволяют задавать четыре адресных окна.

Во время обращения к адресному окну формируется соответствующий сигнал выборки CSx#. При чтении-записи инструкций и данных в пределах адресного окна BUSCON1-ADDRSEL1 формируется сигнал CS1# и т. д. Сигнал выборки CS0# относится к регистру BUSCON0 и формируется во время обращения к адресному пространству, не относящемуся к адресным окнам.

Использование адресных окон дает возможность подключать периферийные устройства и память с различными интерфейсами и оптимизировать параметры внешней шины для каждого устройства.

Когда хотя бы один из регистров BUSCONx имеет установленный в единичное значение бит BUSACTx, работа контроллера внешней шины EBC разрешается и интерфейс внешней шины используется. Порт P1 выдает младшие шестнадцать разрядов адреса A15, …, A0 в том случае, когда хотя бы в одном регистре BUSCONx установлен немультиплексный режим шины.

Функции битов и битовых полей всех регистров BUSCONx одинаковы. Значения регистров BUSCON1, …, BUSCON4 определяют режим работы шины при обращении к адресным окнам, начальный адрес и размер которых задается в соответствующих регистрах ADDRSEL1, …, ADDRSEL4. BUSCON0 определяет временные параметры во время доступа к адресному окну 0, т. е. пространству, незанятому адресными окнами. Значения BUSACT0, ALECTL0 и BTYP регистра BUSCON0 устанавливаются во время сброса через порт P0. Если во время сброса к входу EA# было приложено напряжение уровня логического 0, то BUSACT0 и ALECTL0 устанавливаются в 1, а значение BTYP определяется каналами P0L.7 и P0L.6. Остальные битовые поля и биты обнуляются, устанавливая, таким образом, цикл шины с максимальными задержками. При напряжении на EA#, равном уровню логической 1, значение BUSCON0 обнуляется. При этом обращение по внешней шине запрещается, т. к. остальные регистры BUSCON1, …, BUSCON4 всегда обнуляются после системного сброса. Значения требуемых регистров BUSCON0, …, BUSCON4 и ADDRSEL1, …, ADDRSEL4 можно переустановить.

Следует обратить особое внимание, что сначала устанавливается значение ADDRSELx, а затем – соответствующий ему бит BUSCONx. Это замечание является очень важным, т. к. его несоблюдение приводит к наиболее часто встречающейся ошибке, которая может привести к нестабильной работе.

На рисунках 12.11 – 12.16 и в таблицах 12.6, 12.7 приведено описание регистров управления внешней шиной.

SYSCON
регистр управления системы

значение после Сброса:
00000XX0 X0000000$_B$

SFR (FF12$_H$/ 89$_H$)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STKZ			ROMS1	SGTDIS	ROMEN	BYTDIS	CLKEN	WRCGF	CSCGF	SYSCON5	SYSCON4	SYSCON3	XPEN	VISIBLE	SYSCON0
з ч			з ч	з ч	з ч ап	з ч ап	з ч	з ч ап	з ч	з ч	з ч	з ч ап	з ч	з ч	з ч ап

Рисунок 12.11 – Формат регистра SYSCON

Таблица 12.6 – Функциональное назначение полей регистра SYSCON

Поле	Биты	Тип	Описание
1	2	3	4
SYSCON0	0	Запись Чтение Аппаратное влияние	Бит системной конфигурации.
VISIBLE	1	Запись Чтение	Управление режимом видимости шины XBUS: 0 Обращение по шине XBUS производится только внутри кристалла. 1 Обращение по шине XBUS дублируется на внешней шине.
XPEN	2	Запись Чтение	Бит разрешения XBUS периферии: 0 Доступ к XBUS периферии и обращение к ней запрещены. 1 Доступ к XBUS периферии и обращение к ней разрешены.
SYSCON3–SYSCON5	3-5	Запись Чтение Аппаратное влияние	Зарезервировано
CSCFG	6	Запись Чтение	Бит управления сигналами выбора кристалла CSx#: 0 Режим защелки CSx# сигнала. Сигнал CSx# переключается в низкий уровень с задержкой 1TCL после положительного фронта ALE. 1 Режим раннего сигнала CSx#. Сигнал CSx# переключается в низкий уровень по положительному фронту ALE.
WRCFG	7	Запись Чтение Аппаратное влияние	Управление конфигурацией сигнала записи (устанавливается в инверсное значение, считанное с вывода P0H.0 при сбросе): 0 Сигналы WR# и BHE# работают в соответствии со своими функциями. 1 WR# работает как WRL#, BHE# работает как WRH#.
CLKEN	8	Запись Чтение Аппаратное влияние	Разрешение вывода системного тактового сигнала CLKOUT.
BITDYS	9	Запись Чтение Аппаратное влияние	Бит управления запретом и разрешением сигнала BHE# (устанавливается в зависимости от ширины): 0 Сигнал BHE# разрешен. 1 Сигнал BHE# запрещен, вывод P3.12 может использоваться для программного ввода-вывода.

Окончание таблицы 12.6

1	2	3	4
ROMEN	10	Запись Чтение Аппаратное влияние	Бит доступа к внутреннему ПЗУ (устанавливается аппаратно в зависимости от напряжения на входе EA#): 0 Обращение к внутреннему ПЗУ запрещено, работа с внешней памятью. 1 Обращение к внутреннему ПЗУ разрешено.
SGTDIS	11	Запись Чтение	Бит управления запрещением и разрешением сегментации памяти: 0 Сегментация разрешена (CSP и IP сохраняются-восстанавливаются при входе/ выходе из прерывания). 1 Сегментация запрещена (сохраняются только значения IP).
ROMS1	12	Запись Чтение	Бит локализации внутренней памяти: 0 Внутренняя память программ размещается в сегменте 0 (000000_H - $007FFF_H$). 1 Внутренняя память программ размещается в сегменте 1 (010000_H - $017FFF_H$).
STKSZ	15-13	Запись Чтение	Биты выбора размера системного стека в DPRAM: 000 256 слов ($00'FBFE_H$…$00'FA00_H$). 001 128 слов ($00'FBFE_H$…$00'FB00_H$). 010 64 слова ($00'FBFE_H$…$00'FB80_H$). 011 32 слова ($00'FBFE_H$…$00'FBC0_H$). 100 512 слов ($00'FBFE_H$…$00'F800_H$). 101 Зарезервировано. 110 Зарезервировано. 111 1536 слов ($00'FDFE_H$ …$00'F200_H$).
Примечание – Регистр SYSCON не может быть изменен после выполнения команды EINIT.			

BUSCON0
регистр 0 управления шиной

значение после Сброса:
$00000XX0\ XX000000_B$

SFR ($FF0C_H$/ 86_H)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN0	CSR EN0	-	RDY EN0	BSW C0	BUS ACT0	ALE CTL0	EW EN0	BTYP		MTT C0	RWD C0	MCTC			
з ч	з ч	-	з ч	з ч	з ч ап	з ч ап	з ч	з ч ап		з ч	з ч	з ч			

Рисунок 12.12 – Формат регистра BUSCON0

BUSCON1
регистр 1 управления шиной

SFR ($FF14_H$/ $8A_H$) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN1	CSR EN1	-	RDY EN1	BSW C1	BUS ACT1	ALE CTL1	EW EN1	BTYP		MTT C1	RWD C1	MCTC			
з ч	з ч	-	з ч	з ч	з ч	ап	з ч	ап	з ч	ап	з ч	з ч	з ч		

Рисунок 12.13 – Формат регистра BUSCON1

BUSCON2
регистр 2 управления шиной

SFR ($FF16_H$/ $8B_H$) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN2	CSR EN2	-	RDY EN2	BSW C2	BUS ACT2	ALE CTL2	EW EN2	BTYP		MTT C2	RWD C2	MCTC			
з ч	з ч	-	з ч	з ч	з ч	ап	з ч	ап	з ч	ап	з ч	з ч	з ч		

Рисунок 12.14 – Формат регистра BUSCON2

BUSCON3
регистр 3 управления шиной

SFR ($FF18_H$/ $8C_H$) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN3	CSR EN3	-	RDY EN3	BSW C3	BUS ACT3	ALE CTL3	EW EN3	BTYP		MTT C3	RWD C3	MCTC			
з ч	з ч	-	з ч	з ч	з ч	ап	з ч	ап	з ч	ап	з ч	з ч	з ч		

Рисунок 12.15 – Формат регистра BUSCON3

BUSCON4
регистр 4 управления шиной

SFR ($FF1A_H$/ $8D_H$) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN4	CSR EN4	-	RDY EN4	BSW C4	BUS ACT4	ALE CTL4	EW EN4	BTYP		MTT C4	RWD C4	MCTC			
з ч	з ч	-	з ч	з ч	з ч	ап	з ч	ап	з ч	ап	з ч	з ч	з ч		

Рисунок 12.16 – Формат регистра BUSCON4

Таблица 12.7 – Функциональное назначение полей регистров BUSCON0, BUSCON1, BUSCON2, BUSCON3, BUSCON4

Поле	Биты	Тип	Описание
1	2	3	4
MCTC	3-0	Запись Чтение	Биты управления циклами памяти (количеством дополнительных циклов ожидания, цикл ожидания = 2TCL). Количество дополнительных циклов вычисляется по формуле: число циклов = 15 – MCTC. 0000 15 циклов ожидания. … (15 – MCTC) циклов ожидания. 1111 Без дополнительных циклов ожидания. При RDYENx = 1_B старший бит (3) поля MCTC не используется. При этом число циклов ожидания может выбираться от 0 до 7.
RWDCx	4	Запись Чтение	Бит управления задержкой сигналов RD# и WR#: 0 Задержка сигналов на 1TCL после отрицательного фронта сигнала ALE. 1 Нет задержки сигналов, отрицательные фронты сигналов RD# и WR# соответствуют отрицательному фронту ALE.
MTTCx	5	Запись Чтение	Бит управления третьим состоянием, то есть управления временем, необходимым для освобождения шины внешним устройством при чтении: 0 Формирование одного цикла ожидания (2TCL). 1 Нет дополнительного цикла ожидания.
BTYP	7-6	Запись Чтение	Биты управления внешней конфигурацией: 00 - разрядная демультиплексная шина. 01 - разрядная мультиплексная шина. 10 - 6-разрядная демультиплексная шина.
EWENx	8	Запись Чтение	Бит разрешения раннего сигнала WR#: 0 Нормальный сигнал WR#. Длительность сигнала определяется по формуле (стандартный вариант) ((15 – MCTC) + 2) ×TCL. 1 Ранний сигнал записи WR#. Длительность сигнала определяется по формуле ((15 – MCTC) + 1) ×TCL.
ALECTLx	9	Запись Чтение	Бит управления длительностью ALE#: 0 Нормальный сигнал ALE#. 1 Удлиненный сигнал ALE#. Увеличение длительности на 1TCL. Для мультиплексного режима дополнительное удержание адреса на шине на время одного TCL.
BUSACTx	10	Запись Чтение	Бит управления активностью шины (использования адресного окна): 0 Внешняя шина запрещена, адресное окно используется при внутренней дешифрации адреса. 1 Внешняя шина разрешена в пределах соответствующего адресного окна ADDRSEL.

Окончание таблицы 12.7

1	2	3	4
BSWCx	11	Запись Чтение	Бит управления переключением BUSCON (изменение сигналов внешней шины для следующего адресного окна): 0 Изменение сигналов для следующего окна происходит сразу после окончания предыдущего. 1 Вставляется задержка третьего состояния 2TCL для следующего адресного окна.
RDYENx	12	Запись Чтение	Бит разрешения сигнала READY#: 0 READY# не используется и окончание цикла шины определяется только битовым полем MCTC. 1 Окончание цикла шины определяется битовым полем MCTC и входным сигналом READY#.
CSRENx	14	Запись Чтение	Бит разрешения сигнала CSx# во время цикла чтения: 0 Сигнал CSx# не зависит от сигнала RD#. 1 Сигнал CSx# генерируется во время активности RD#.
CSWENx	15	Запись Чтение	Бит разрешения сигнала записи WR#, WRL#, WRH#: 0 Формирование сигнала CSx# не зависит от сигнала записи. 1 Формирование сигнала CSx# во время активного сигнала записи WR#, WRL#, WRH#, BHE#.

На рисунках 12.17 – 12.20 приведены форматы регистров ADDRSEL1, …, ADDRSEL4, в таблице 12.8 – функциональное назначение полей этих регистров.

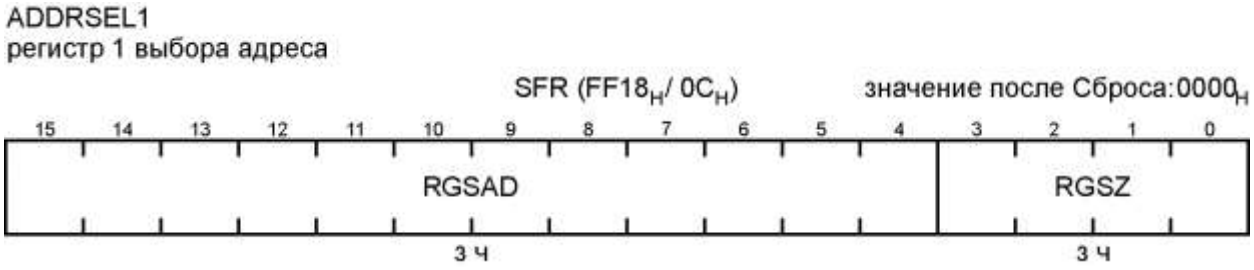


Рисунок 12.17 – Формат регистра ADDRSEL1

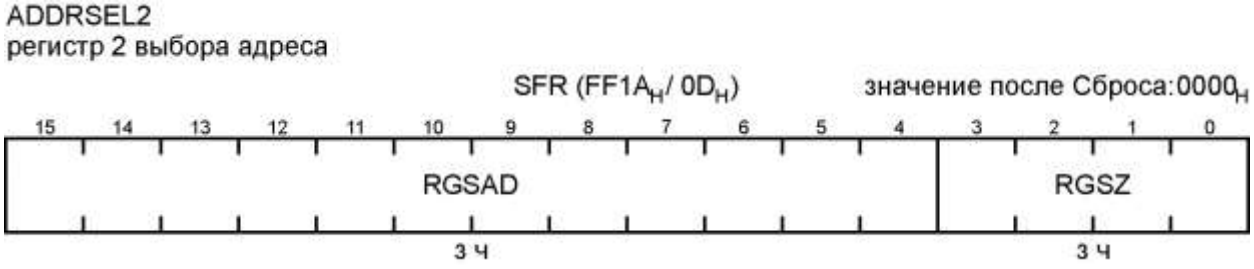


Рисунок 12.18 – Формат регистра ADDRSEL2

ADDRSEL3
регистр 3 выбора адреса

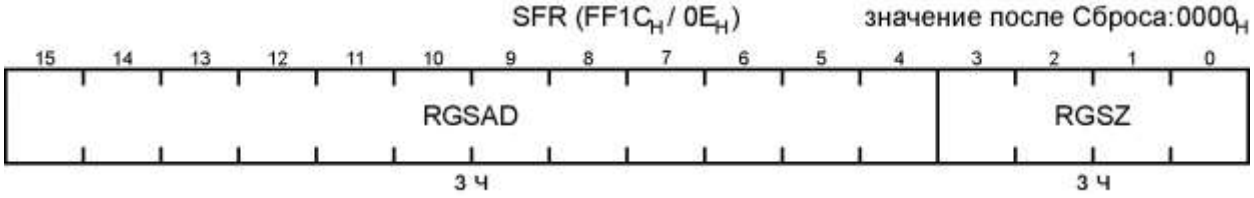


Рисунок 12.19 – Формат регистра ADDRSEL3

ADDRSEL4
регистр 4 выбора адреса

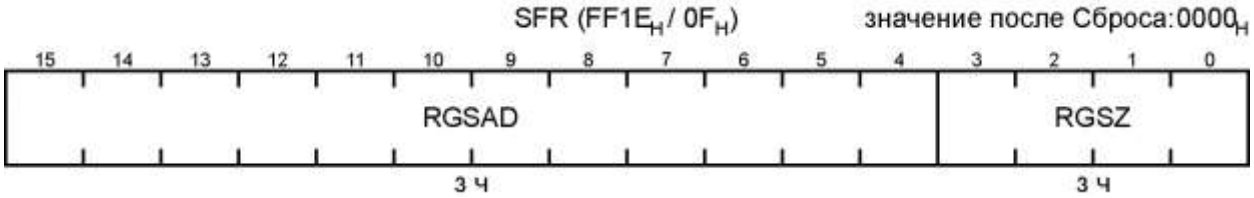


Рисунок 12.20 – Формат регистра ADDRSEL4

Таблица 12.8 – Функциональное назначение полей регистров ADDRSEL1, ADDRSEL2, ADDRSEL3, ADDRSEL4

Поле	Биты	Тип	Описание
RGSZ	3–0	Запись Чтение	Биты управления размером адресного окна, управляемого парой регистров BUSCONx – ADDRSELx.
RGSAD	15–4	Запись Чтение	Биты разрядов A23 – A12 начального адреса окна.

Регистр ADDRSEL0 отсутствует, так как регистр BUSCON0 управляет всеми внешними доступами во всем адресном пространстве, кроме области, занимаемой четырьмя адресными окнами для BUSCON1, …, BUSCON4.

12.13 Синхронизация интерфейса внешней шины

Синхронизация всех операций передач данных интерфейсом внешней шины (EBC) осуществляется с помощью внутренних тактовых сигналов системы. Для синхронизации периферии, подключенной к внешней шине, используется синхросигнал CLKOUT.

Синхронизация внешней шины привязана к фронту нарастания синхросигнала на выходе CLKOUT.

Главный тактовый сигнал микроконтроллера SYS_CLK, а также два тактовых сигнала POS_CLK_EN и NEG_CLK_EN, имеющие частоту в два раза меньше, чем у CLKOUT, формируют сигналы CLK_POS и CLK_NEG, см. рисунок 12.21. Эти сигналы являются тактовыми сигналами для внутренних блоков микроконтроллера. Они же используются для формирования сигнала CLKOUT. Используя бит CLKEN регистра SYSCON, можно разрешить выдачу сигнала CLKOUT на вывод P3.15 порта P3. Параметры сигнала CLKOUT приведены в таблице 12.9 и на рисунках 12.21, 12.22.

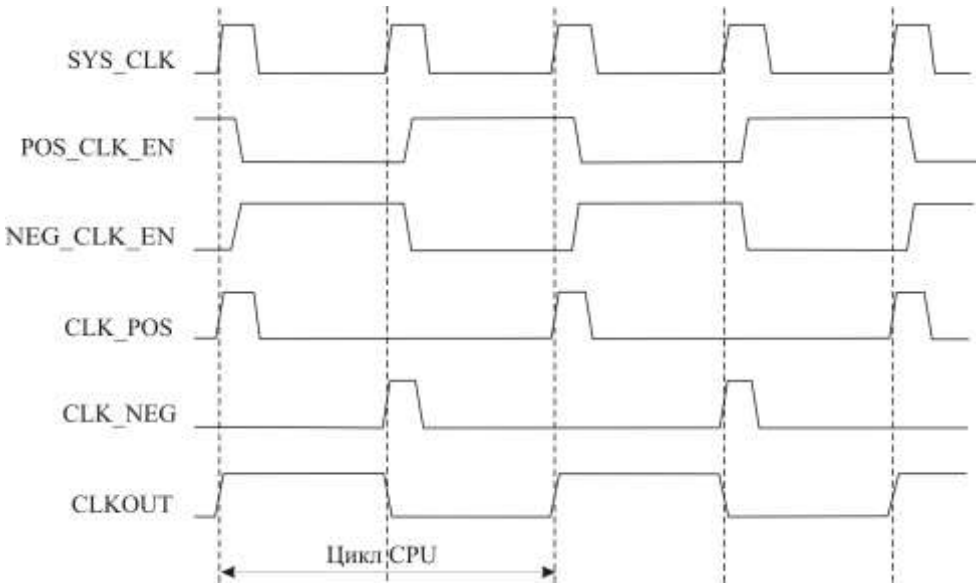


Рисунок 12.21 – Тактовые сигналы микроконтроллера

Таблица 12.9 – Системный тактовый сигнал CLKOUT

Параметр сигнала, единица измерения	Символ	Граничные значения	
		минимальное	максимальное
Период следования сигнала CLKOUT, нс	t_{C5}	50[1)]	500
Длительность сигнала высокого уровня CLKOUT, нс	t_{C6}	8	–
Длительность сигнала низкого уровня CLKOUT, нс	t_{C7}	6	–
Время нарастания сигнала CLKOUT, нс	t_{C8}	–	10
Время спада сигнала CLKOUT, нс	t_{C9}	–	10

1) На длительность периода CLKOUT оказывает влияние джиттер (случайные помехи) системы фазовой автоподстройки PLL. Значения для $f_{CPU} = 20$ МГц. Для более длительных периодов относительное отклонение уменьшается.

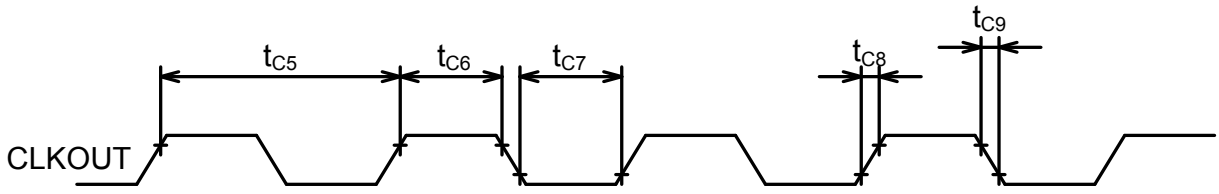


Рисунок 12.22 – Временная диаграмма сигнала CLKOUT

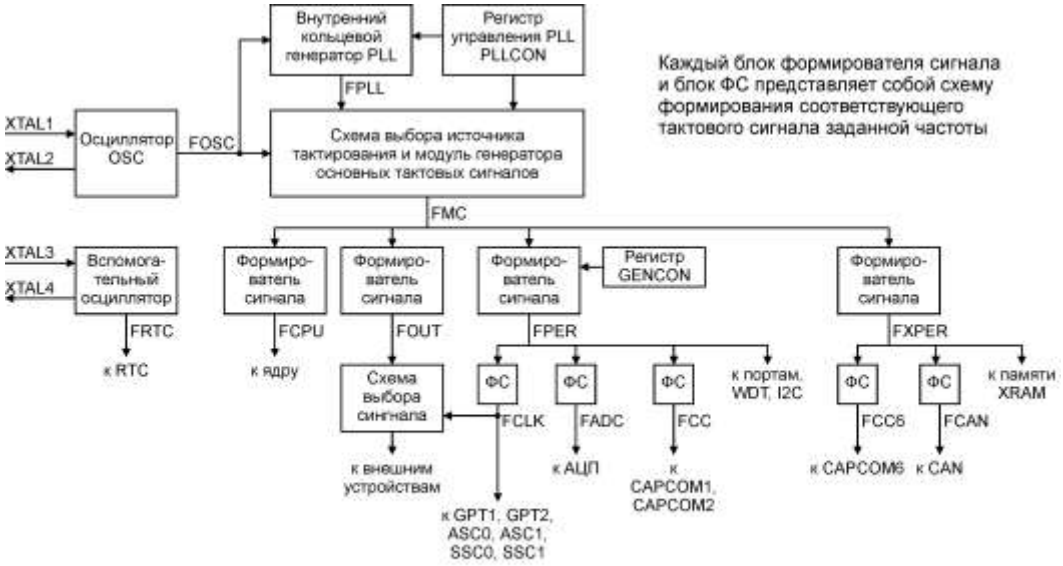
13 Генератор тактовых сигналов

Всеми действиями аппаратных средств микроконтроллера и периферийных устройств управляют тактовые сигналы, генерируемые модулем генератора тактовых сигналов. Модуль генератора тактовых сигналов состоит из трех блоков: осциллятора OSC, блока PLL (phase locked loop) и схемой распределения тактового сигнала MUX, смотри рисунок 13.1.

Модулем генератора основных тактовых сигналов формируется четыре различных тактовых сигнала:

- FCPU – тактовый сигнал ядра контроллера;
- FPER – тактовый сигнал периферийных модулей, подключенных к шине PDBUS;
- FXPER – тактовый сигнал периферийных модулей, подключенных к шине XBUS;
- FOUT – выходной тактовый сигнал, служит для тактирования внешних устройств.

Кроме этого, дополнительной схемой формируется медленный тактовый сигнал FRTC, поступающий на модуль часов реального времени RTC, позволяющий модулю работать независимо от описанных выше тактовых сигналов.



Обозначение сигнала	Назначение сигнала	Обозначение частоты сигнала	Соотношения между сигналами
FOSC	Выходной сигнал осциллятора	f_{OSC}	$f_{OSC} = f_{OXTAL1}$
FPLL	Выходной сигнал генератора PLL	f_{PLL}	см. формулу (13.1)
FMC	Сигнал формирователя машинного цикла	f_{MC}	см. рисунок 13.5
FRTC	Тактовый сигнал для синхронизации блока RTC	f_{RTC}	
FCPU	Тактовый сигнал ядра контроллера	f_{CPU}	$f_{CPU} = f_{MC}/2$
FOUT	Выходной тактовый сигнал контроллера для синхронизации внешних устройств	f_{OUT}	см. формулу (13.2)
FPER	Тактовый сигнал периферийных модулей, подключенных к шине PDBUS	f_{PER}	$f_{PER} = f_{MC}/2$ (CPPER = 0b) $f_{PER} = f_{MC}/4$ (CPPER = 1b)
FCLK	Тактовый сигнал таких периферийных модулей как GPT1, ASC0 и др.	f_{CLK}	$f_{CLK} = f_{PER}$
FADC	Тактовый сигнал АЦП	f_{ADC}	$f_{ADC} = f_{PER}$
FCC	Тактовый сигнал блоков CAPCOM1 и CAPCOM2	f_{CC}	$f_{CC} = f_{PER}$
FXPER	Тактовый сигнал периферийных модулей, подключенных к шине XBUS	f_{XPER}	$f_{XPER} = f_{MC}/2$
FCC6	Тактовый сигнал блока CAPCOM6	f_{CC6}	$f_{CC6} = f_{XPER}$
FCAN	Тактовый сигнал модуля CAN	f_{CAN}	$f_{CAN} = f_{XPER}$

Рисунок 13.1– Функциональная схема модуля генератора основных тактовых сигналов

Осциллятор

Осциллятор может работать либо с внешним кварцем и соответствующей схемой включения, либо может быть запущен посредством сигналов внешнего тактового генератора. При работе от внешнего генератора осциллятор выдает тактовые сигналы для оборудования микроконтроллера, производя при этом деление входной частоты на два.

Блок PLL

Блок PLL умножает частоту тактового сигнала, поступающего на его вход с выхода осциллятора, на программируемый коэффициент, задаваемый в регистре управления. Блок может быть отключен, при этом схема переходит в режим работы с тактированием от осциллятора.

Блок распределения тактового сигнала

Схема распределения тактового сигнала обеспечивает формирование раздельных тактовых импульсов для ядра микроконтроллера и группы периферийных модулей.

Генератор медленного тактового сигнала

Кроме основного генератора тактового сигнала, который используется для работы ядра и периферии, в схеме контроллера имеется генератор медленного тактового сигнала частотой 32,768 кГц. Медленный тактовый сигнал используется для работы модуля реального времени RTC.

13.1 Осцилляторы

Основной осциллятор

Блок осциллятора, входящий в структуру модуля генератора тактовых сигналов, способен работать с внешним кварцем и схемой запуска или от внешнего тактового генератора, см. рисунок 13.2.

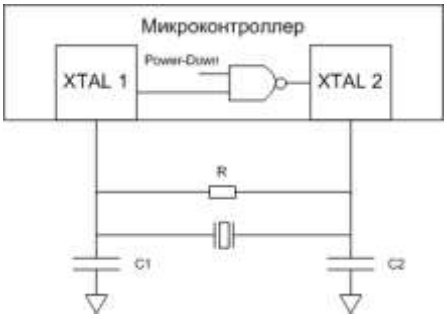


Рисунок 13.2 – Внешняя схема включения осциллятора

Осциллятор оптимизирован на работу для диапазона входных частот от 10 до 20 МГц.

При работе, например от внешнего генератора или master-устройства, тактовый сигнал подается на вход XTAL1. В этом случае частота входного тактового сигнала может находиться в диапазоне от 0 до 40 МГц. Максимальная частота входного сигнала ограничена максимальной тактовой частотой контроллера.

Осциллятор автоматически выключается при переходе контролера в режим Power_Down. Однако модуль часов реального времени продолжает функционировать, т. к. синхронизируется медленным тактовым сигналом, формируемым от отдельного генератора частотой 32,768 кГц.

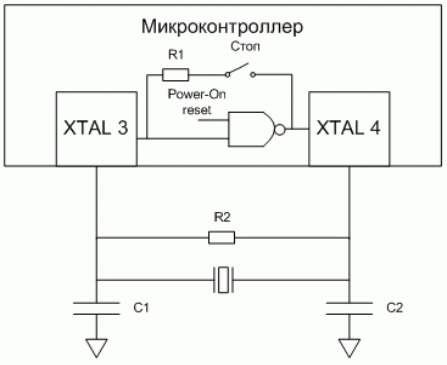
Осциллятор запускается во время системного сброса.

Вспомогательный осциллятор

В составе модуля генератора тактовых сигналов микроконтроллера присутствует вспомогательный осциллятор. Этот дополнительный осциллятор используется для формирования медленного тактового сигнала, который поступает на тактовый вход модуля часов реального времени RTC, и работает независимо от генератора системного

тактового сигнала. Вспомогательный осциллятор оптимизирован для работы на частоте $f_{CIXTAL3}$ = 32,768 кГц. Такая узкая рабочая полоса частот позволила существенно снизить потребляемую мощность вспомогательного осциллятора.

Для подключения внешнего кварца и схемы запуска используются выводы XTAL3 и XTAL4. На рисунке 13.3 представлена схема включения вспомогательного осциллятора.



Примечание – Значения емкостей внешних конденсаторов C1 и C2 обычно выбираются в диапазоне от 10 до 30 пФ.

Рисунок 13.3 – Схема включения вспомогательного осциллятора

Вспомогательный осциллятор содержит внутренний резистор обратной связи, который можно подключать вместо внешнего резистора обратной связи.

Управление работой вспомогательного осциллятора, а также контроль коэффициента деления частоты периферийного тактового сигнала, выполняет регистр GENCON, находящийся в области пространства ESFR, см. рисунок 13.4, таблицу 13.1.

GENCON

регистр управления вспомогательным осциллятором

ESFR ($F1CC_H$/ $E6_H$) значение после Сброса: 00_H

7	6	5	4	3	2	1	0
-					CPPER	OSC32DIS	ROSC32EN
-					3 Ч	3 Ч	3 Ч

Рисунок 13.4 – Формат регистра GENCON

Таблица 13.1 – Функциональное назначение полей регистра GENCON

Поле	Биты	Тип	Описание
ROSC32EN	0	Чтение Запись	Разрешает внутренний резистор в цепи обратной связи: 0 Внутренний резистор отключен. 1 Внутренний резистор подключен.
OSC32DIS	1	Чтение Запись	Запрещает работу вспомогательного осциллятора: 0 Осциллятор включен. 1 Осциллятор выключен.
CPPER	2	Чтение Запись	Коэффициент деления частоты тактового сигнала периферии: 0 $f_{PER} = f_{MC}/2$ 1 $f_{PER} = f_{MC}/4$

Вспомогательный осциллятор может работать от внешнего тактового сигнала частотой 32,768 кГц, который в этом случае подается на вход XTAL3. Если модуль RTC не используется, то вспомогательный осциллятор можно отключить, записав «1» в бит OSC32DIS регистра GENCON. Это позволит уменьшить потребляемую контроллером мощность.

13.2 Тактовый генератор PLL и контроль частоты

В состав модуля тактового генератора системного тактового сигнала входит схема программируемого генератора, формирующего тактовые сигналы с высокой гибкостью.

Операциями контроллера управляет тактовый сигнал f_{MC}. Частота тактового сигнала, поступающего на CPU, определяется как $f_{CPU} = f_{MC}/2$. Частота периферийного тактового сигнала определяется битом CPPER регистра GENCON и может быть: $f_{PER} = f_{MC}/2$ (CPPER = 0_B) или $f_{PER} = f_{MC}/4$ (CPPER = 1_B).

Тактовая частота, поступающая с осциллятора на вход PLL, может быть умножена схемой PLL путем программирования коэффициентов или разделена на программируемый коэффициент делителя частоты прескалером. Коэффициенты позволяют гибко изменять выходной сигнал в широкой полосе частот. На рисунке 13.5 дана диаграмма, на которой представлены возможности по формированию тактового сигнала f_{MC} при различных частотах сигнала f_{OSC}, поступающего с осциллятора.

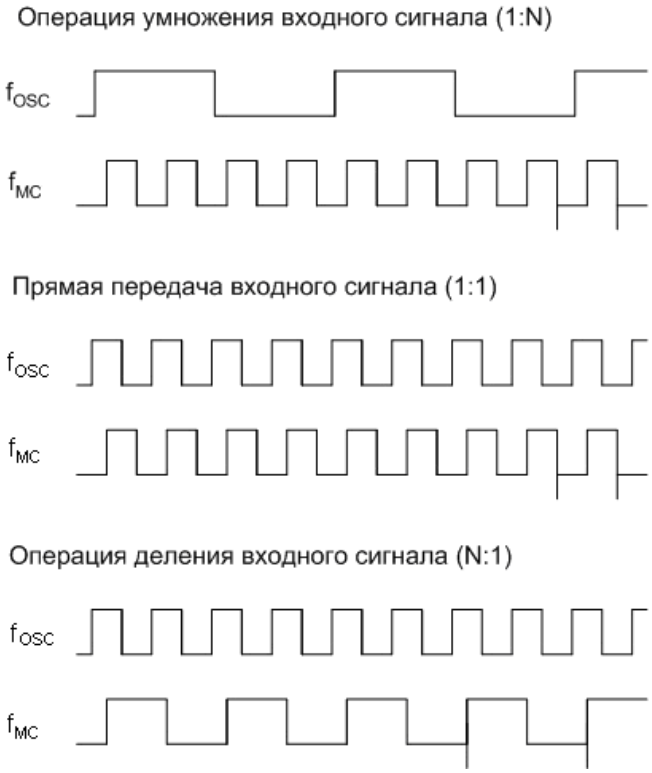


Рисунок 13.5 – Механизм формирования тактового сигнала

В примере, показанном на рисунке 13.5, входной сигнал умножается на коэффициент 1:4, для операции деления коэффициент равен 2:1.

Управление PLL осуществляется с помощью регистра управления PLLCON, см. рисунок 13.6, таблицу 13.2.

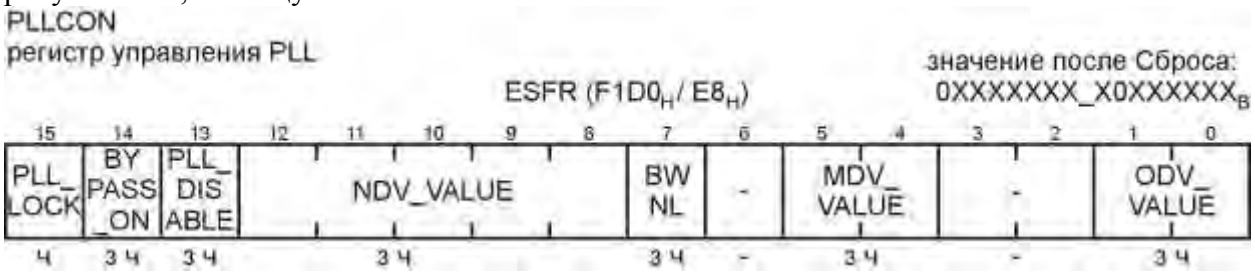


Рисунок 13.6 – Формат регистра PLLCON

Таблица 13.2 – Функциональное назначение полей регистра PLLCON

<table>
<tr><th>Поле</th><th>Биты</th><th>Тип</th><th colspan="3">Описание</th></tr>
<tr><td rowspan="6">ODV_VALUE</td><td rowspan="6">1-0</td><td rowspan="6">Чтение
Запись</td><td colspan="3">Коэффициент деления выходного каскада PLL</td></tr>
<tr><td></td><td>O – значение</td><td>Значение деления</td></tr>
<tr><td>00</td><td>0</td><td>2</td></tr>
<tr><td>01</td><td>1</td><td>4</td></tr>
<tr><td>10</td><td>2</td><td>8</td></tr>
<tr><td>11</td><td>3</td><td>16</td></tr>
<tr><td rowspan="6">MDV_VALUE</td><td rowspan="6">5-4</td><td rowspan="6">Чтение
Запись</td><td colspan="3">Коэффициент деления входного каскада PLL</td></tr>
<tr><td></td><td>M – значение</td><td>Значение деления</td></tr>
<tr><td>00</td><td>0</td><td>1</td></tr>
<tr><td>01</td><td>1</td><td>2</td></tr>
<tr><td>10</td><td>2</td><td>4</td></tr>
<tr><td>11</td><td>3</td><td>8</td></tr>
<tr><td>BWNL</td><td>7</td><td>Чтение
Запись</td><td colspan="3">Бит управления работой микроконтроллера:

0 Схема работает от PLL. При потере частоты схема отключается и включается при нахождении частоты.
1 Схема работает от PLL. При потере частоты схема переключается на работу от осциллятора, при нахождении частоты автоматически возвращается в режим работы от PLL.</td></tr>
<tr><td rowspan="8">NDV_VALUE</td><td rowspan="8">12-8</td><td rowspan="8">Чтение
Запись</td><td colspan="3">Коэффициент умножения частоты PLL:</td></tr>
<tr><td></td><td>N – значение</td><td>Значение деления</td></tr>
<tr><td>000000</td><td>0</td><td>0</td></tr>
<tr><td>000001</td><td>1</td><td>1</td></tr>
<tr><td>000010</td><td>2</td><td>2</td></tr>
<tr><td>…</td><td>…</td><td>…</td></tr>
<tr><td>111110</td><td>30</td><td>30</td></tr>
<tr><td>111111</td><td>31</td><td>31</td></tr>
<tr><td>PLL_DISABLE</td><td>13</td><td>Чтение
Запись</td><td colspan="3">Выключение PLL (перевод его в режим пониженного энергопотребления):

0 Схема работает от PLL, если это разрешено соответствующими битами управления.
1 PLL выключен, схема автоматически переходит в режим работы от осциллятора.</td></tr>
<tr><td>BYPASS_ON</td><td>14</td><td>Чтение
Запись</td><td colspan="3">Выключение режима работы схемы непосредственно от осциллятора:
0 Схема работает от PLL или от осциллятора (если BWNL = 1_B и частота PLL потеряна).
1 Схема работает только от осциллятора.</td></tr>
<tr><td>PLL_LOCK</td><td>15</td><td>Чтение</td><td colspan="3">Статусный бит синхронизации PLL с частотой осциллятора:
0 PLL не синхронизирован с частотой осциллятора.
1 PLL синхронизирован с частотой осциллятора, и схема может работать от PLL.</td></tr>
<tr><td>–</td><td>3-2, 6</td><td>–</td><td colspan="3">Не используется, при чтении возвращает «0».</td></tr>
</table>

Регистр PLLCON устанавливается во время сброса по внешним выводам.

Если сигнал rd = 0_B во время сброса, бит PLL_DISABLE устанавливается, тем самым, выключая PLL.

Выводы P0H.5, P0H.6, P0H.7 порта P0 устанавливают коэффициенты деления/ умножения.

В таблице 13.3 представлены варианты программирования коэффициентов умножения/деления во время сброса в зависимости от сигналов на выводах P0H.5, P0H.6 и P0H.7.

Таблица 13.3 – Программирование коэффициентов умножения/деления во время сброса

Состояние выводов порта P0			Коэффициенты			Частота
P0H.7	P0H.6	P0H.5	ODV	MDV	NDV	на выходе
1	1	1	01	00	01011_B	$f_{OSC} \times 3$
1	1	0	01	00	10001_B	$f_{OSC} \times 4{,}5$
1	0	1	01	01	01111_B	$f_{OSC} \times 2$
1	0	0	01	00	10011_B	$f_{OSC} \times 5$
0	1	1	01	01	00111_B	$f_{OSC} \times 1$
0	1	0	01	01	10011_B	$f_{OSC} \times 2{,}5$
0	0	1	00	10	10011_B	$f_{OSC} \times 2{,}5$
0	0	0	01	10	00111_B	$f_{OSC} / 2$

Операции PLL

Основной тактовый сигнал может формироваться как осциллятором, так и PLL. Управление режимами работы PLL осуществляется с помощью регистра PLLCON. Также этот регистр управляет мультиплексором, обеспечивающим переключение между выходными сигналами PLL и осциллятора.

Инициализация генератора тактовых сигналов происходит во время сигнала сброса путем выставления на выводах P0H.5, P0H.6 и P0H.7 порта P0 необходимых уровней. При этом схема автоматически начинает работать от PLL, а коэффициенты, управляющие выходной частотой, устанавливаются в регистре PLLCON в зависимости от сигналов порта на выводах порта. Установленный бит PLL_LOCK показывает, что PLL синхронизирован с частотой осциллятора, и схема может работать от PLL.

PLL постоянно синхронизируется с осциллятором. В случае определения PLL отсутствия входного тактового сигнала, генерируется запрос на прерывание. Это прерывание сигнализирует о том, что частота PLL вышла из установившегося режима и не является более стабильной. Эта ошибка также может возникнуть, если частота внешнего тактового генератора нестабильна. Если при работе схемы от PLL произошла потеря частоты, то схема возвратится в режим работы от PLL при нахождении частоты лишь спустя 200 мс.

Бит BWNL управляет режимом работы схемы в ситуациях, когда PLL перестает работать по каким-либо причинам. Возможны два варианта работы:

- если BWNL = 0_B, схема работает от PLL, при потере частоты отключается, а при нахождении частоты вновь включается;

- если BWNL = 1_B, схема работает от PLL, при потере частоты схема переключается в режим работы от осциллятора, а при нахождении частоты автоматически переходит в режим работы от PLL.

Установка бита BYPASS_ON принудительно переводит схему в режим работы от осциллятора. Если BYPASS_ON = 0_B, схема работает от PLL в режиме, управляемом битом BWNL.

В схеме предусмотрено отключение PLL (перевод его в режим пониженного потребления). При этом схема автоматически переходит в режим работы от осциллятора. Этот режим включается установкой бита PLL_DISABLE.

Примечание – Частота тактового сигнала на выходе осциллятора в режиме работы от внешнего кварца находится в диапазоне от 10 до 20 МГц, поэтому возможна ситуация, когда при переходе в режим работы от осциллятора схема будет работать медленнее, чем от PLL.

Частота тактового сигнала

Частота тактового сигнала на выходе PLL определяется коэффициентами деления/умножения частоты, поступающей на вход и снимаемой с выхода PLL. Частотой сигнала управляют три коэффициента:

NDV – коэффициент умножения входной частоты;

MDV – коэффициент деления входного каскада PLL;

ODV – коэффициент деления выходной частоты PLL.

Сигнал, поступающий на вход PLL, проходит через три блока, которыми управляют представленные выше коэффициенты.

Коэффициент MDV управляет делителем входной частоты.

Коэффициент NDV управляет умножением сигнала, поступающего с выхода делителя. Максимальное умножение можно получить, используя максимальное значение этого коэффициента и минимальные значения коэффициентов делителей.

Коэффициент ODV используется для управления выходным делителем, который масштабирует частоту выходного сигнала ядра. Выходной делитель может быть использован для генерации основного тактового сигнала без изменения настроек умножителя PLL.

Для расчета частоты сигнала используется формула

$$f_{PLL} = \frac{f_{OSC} \times (NDV[4:0] + 1)}{2^{ODV[1:0]+1} \times 2^{MDV[1:0]}} \qquad (13.1)$$

При расчете выходной частоты необходимо учесть рабочий диапазон частот PLL и максимальную рабочую частоту контроллера. Поэтому коэффициенты PLL следует выбирать таким образом, чтобы результирующая частота удовлетворяла следующему неравенству:

$$3{,}2 \text{ МГц} \le f_{PLL} \le 40 \text{ МГц}$$

Домены тактовых сигналов

Модулем тактового генератора формируется пять типов тактовых сигналов: тактовый сигнал CPU, два тактовых сигнала периферии, тактовый сигнал модуля реального времени и тактовый сигнал для внешних устройств. В таблице 13.4 представлены домены тактовых сигналов и их состояние в различных режимах работы контроллера.

Таблица 13.4 – Домены тактовых сигналов

Домен тактового сигнала	Название тактового сигнала	Активный режим	Режим IDLE	Режим Power_Down	Подключенные модули
1	2	3	4	5	6
XBUS	FXPER	Включен	Включен	Выключен	CAN, CAPCOM6, XRAM
CPU	FCPU	Включен	Выключен	Выключен	CPU, DPRAM
PDBUS	FPER	Включен	Включен	Выключен	ADC, ASC0, ASC1, CAPCOM1, I2C, GPT1, GPT2, CAPCOM2, SSC0, SSC1, порты, RTC[1)], WDT

Окончание таблицы 13.4

1	2	3	4	5	6
RTC	FRTC	Включен	Включен	Включен/ выключен	RTC[1)]

[1)] Модуль RTC относится к двум доменам тактовых сигналов:

- запись в регистры и чтение производится по основному периферийному тактовому сигналу FPER;
- счетчик реального времени и прескалер работают в домене медленного тактового сигнала FRTC.

Генерация прерываний

При нахождении или потере частоты PLL могут вырабатываться запросы на прерывания по потере или нахождению частоты, которыми управляют регистры контроля прерываний:

- PLLLOCK_IC – прерыванием по нахождению частоты;
- PLLUNLOCK_IC – прерыванием по потере частоты.

В таблице 13.5 представлены адреса регистров управления прерываниями PLLLOCK_IC и PLLUNLOCK_IC и соответствующие им векторы прерываний.

Таблица 13.5 – Регистры управления прерываниями и векторы прерываний

Регистр	Адрес	Вектор	Значение после сброса
PLLLOCK_IC	$0F19E_H$	$010C_H$	0000_H
PLLUNLOCK_IC	$0F156_H$	$01AC_H$	0000_H

Прерывания по потере и нахождению частоты можно использовать для повышения эффективности работы. Ниже приведен пример использования прерываний.

Запускаем схему в режиме работы от PLL.

Разрешаем переключение частоты на осциллятор при потере частоты PLL установкой бита PLLCON_BWNL.

Разрешаем прерывание по потере частоты.

При потере частоты PLL схема автоматически перейдет в режим работы от осциллятора и вырабатывается прерывание по вектору $01AC_H$.

В подпрограмме обработки прерываний разрешаем прерывание по нахождению частоты PLL и переводим контроллер в режим Idle.

После нахождения частоты контроллер выйдет из режима Idle и начнет обработку подпрограммы, в которой запрещено прерывание по нахождению частоты. После этого продолжается выполнение основной программы.

Генерация внешнего тактового сигнала

Для тактирования внешних устройств можно использовать тактовый сигнал, выводимый наружу через вывод P3.15. Могут быть выбраны два типа сигнала:

- CLKOUT, в качестве которого используется тактовый сигнал FXPER;
- FOUT – с программируемой частотой.

Программируемый сигнал формируется с помощью перезагружаемого счетчика, таким образом, изменение частоты может происходить с достаточно маленьким шагом. Генератор может работать с производительностью 50 %, посредством включения соответствующей схемы. На рисунке 13.7 представлена схема формирования выходного тактового сигнала.

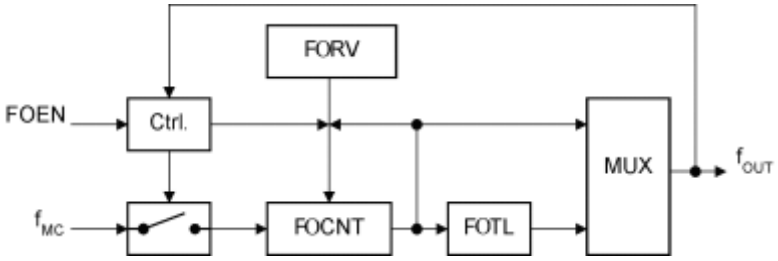


Рисунок 13.7 – Схема формирования выходного тактового сигнала

Контроль над формированием тактового сигнала FOUT осуществляется с помощью регистра управления FOCON, см. рисунок 13.8, таблицу 13.6.

Генератор выходного сигнала FOUT всегда обеспечивает формирование законченного периода тактового сигнала:

- когда FOUT стартует (FOEN → 1), FOCNT загружается значением FORV;
- когда FOUT завершается (FOEN → 0), FOCNT останавливается в момент перехода FOUT в «0», см. рисунок 13.9.

Рисунок 13.8 – Формат регистра FOCON

Таблица 13.6 – Функциональное назначение полей регистра FOCON

Поле	Биты	Тип	Описание
FOCNT	5-0	чтение аппаратное влияние	Счетчик выходной частоты.
FOTL	6	чтение аппаратное влияние	Защелка переключателя выходной частоты. Переключается после каждого обнуления FOCNT.
RORV	13-8	чтение запись	Значение перезагрузки счетчика выходной частоты. Копируется в FOCNT после каждого обнуления FOCNT.
FOSS	14	чтение запись	Выбор типа выходного сигнала: 0 Выходной сигнал с производительностью 50 %. 1 Производительность зависит от FORV.
FOEN	15	чтение запись	Разрешение вывода частоты. Генератор выходной частоты выключается, когда сигнал FOUT переходит в низкий уровень. FOCNT работает, FOUT выводится на выход контролера. Первая перезагрузка выполниться после переключения из «0» в «1»
–	7	–	Зарезервирован – не использовать
Примечание – Запись в битовые поля FOCNT и FOTL регистра FOCON запрещена. Это ограничение не позволит изменить частоту тактового сигнала или остановить тактовый генератор.			

Тактовые сигналы FOUT и FCLK выводятся наружу через вывод P3.15. Для этого необходимо переключить вывод P3.15 на использование альтернативной функции, см. таблицу 13.7.

За вывод сигнала CLKOUT отвечает бит SYSCON.CLKEN. Для вывода CLKOUT необходимо установить бит SYSCON.CLKEN или бит ALTSEL0P3.15. Для вывода FOUT необходимо сбросить указанные биты и установить бит FOEN регистра FOCON и бит ALTSEL1P3.15.

Таблица 13.7 – Приоритет сигналов на выводе P3.15

Приоритет	Функция	Управление
1	CLKOUT	CYSCON.CLKEN = 1_B или ALTSEL0P3.15 = 1_B
2	FOUT	FOCON.FOEN = 1_B, ALTSEL1P3.15 = 1_B
3	Основное назначение	CYSCON.CLKEN = 0_B, FOCON.FOEN = 0_B

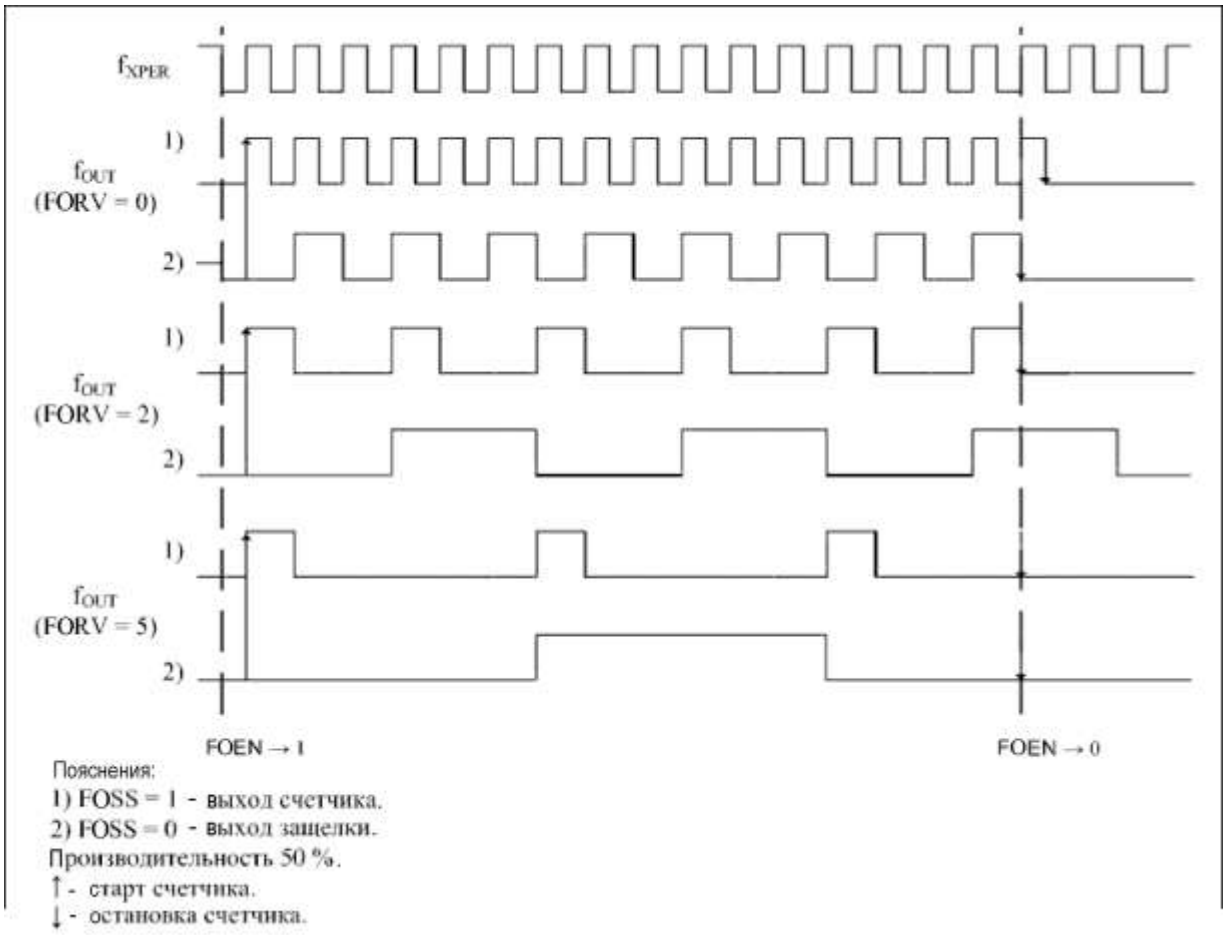


Рисунок 13.9 – Временная диаграмма формирования выходных сигналов

Вычисление выходной частоты

Значение выходной частоты может быть вычислено по формуле

$$f_{OUT} = \frac{f_{MC}}{(FORV + 1) \times 2^{(1 - FOSS)}} \qquad (13.2)$$

В таблице 13.8 представлен список возможных выходных частот сигнала FOUT.

Таблица 13.8 – Список возможных выходных частот сигнала FOUT

f_{MC}, МГц	f_{OUT}, кГц, для FORV = XX, FOSS = $1_B/0_B$					FORV для f_{OUT} = 1 МГц	
	00_H	01_H	02_H	$3E_H$	$3F_H$	FOSS = 0_B	FOSS = 1_B
4	4 000 2 000	2 000 1 000	1 333,33 666,67	63,492 31,746	62,500 31,250	01_H	03_H
10	10 000 5 000	5 000 2 500	3 333,33 1 666,67	158,730 79,365	156,250 78,125	04_H	09_H
12	12 000 6 000	6 000 3 000	4 000,00 2 000,00	190,476 95,238	187,500 93,750	05_H	$0B_H$
16	16 000 8 000	8 000 4 000	5 333,33 2 666,67	253,968 126,984	250,000 125,000	07_H	$0F_H$
20	20 000 10 000	10 000 5 000	6 666,67 333,33	317,460 158,730	312,500 156,250	09_H	13_H
25	25 000 12 500	12 500 6 250	8 333,33 4 166,67	396,825 198,413	390,265 195,313	$0B_H$ (1,04167) $0C_H$ 0,96154	18_H
33	33 000 16 500	16 500 8 250	11 000,00 5 500,00	523,810 361,905	515,625 257,816	$0F_H$ (1,03125) 10_H (0,97059)	20_H
40	40 000 20 000	20 000 10 000	13 333,33 6 666,67	634,920 317,460	625,000 312,500	13_H	27_H

14 Блоки GPT1 и GPT2 таймеров общего назначения

Блоки таймеров основного назначения GPT1 и GPT2 имеют гибкую многофункциональную структуру. Таймеры могут использоваться для измерения времени, подсчета количества событий, измерения длительности импульсов, генерации импульсов, умножения частоты и других функций. Пять 16-битных таймеров объединены в два блока GPT1 и GPT2.

Любой таймер любого блока может работать независимо в различных режимах. Любой таймер блока может быть объединен с другим таймером этого же блока. У каждого блока есть дополнительные функции ввода-вывода и свои прерывания.

Таймеры первого блока T2, T3, T4 имеют максимальное разрешение $f_{CLK}/4$. Вспомогательные таймеры GPT1 могут быть настроены как регистры перезагрузки или захвата для основного таймера.

Таймеры второго блока T5 и T6 имеют максимальную разрешающую способность $f_{CLK}/2$. Дополнительный CAPREL регистр GPT2 поддерживает операции захвата и перезагрузки.

Независимо от режимов работы таймеры T2, …, T6 и регистр CAPREL используются аппаратно, в связи с чем доступны только для записи. Попытки чтения этих регистров будут приводить к получениию непредсказуемых значений и поэтому являются недопустимыми.

14.1 Функциональное описание блока GPT1

С точки зрения программиста блок GPT1 представляет собой набор регистров, см. рисунок 14.1.

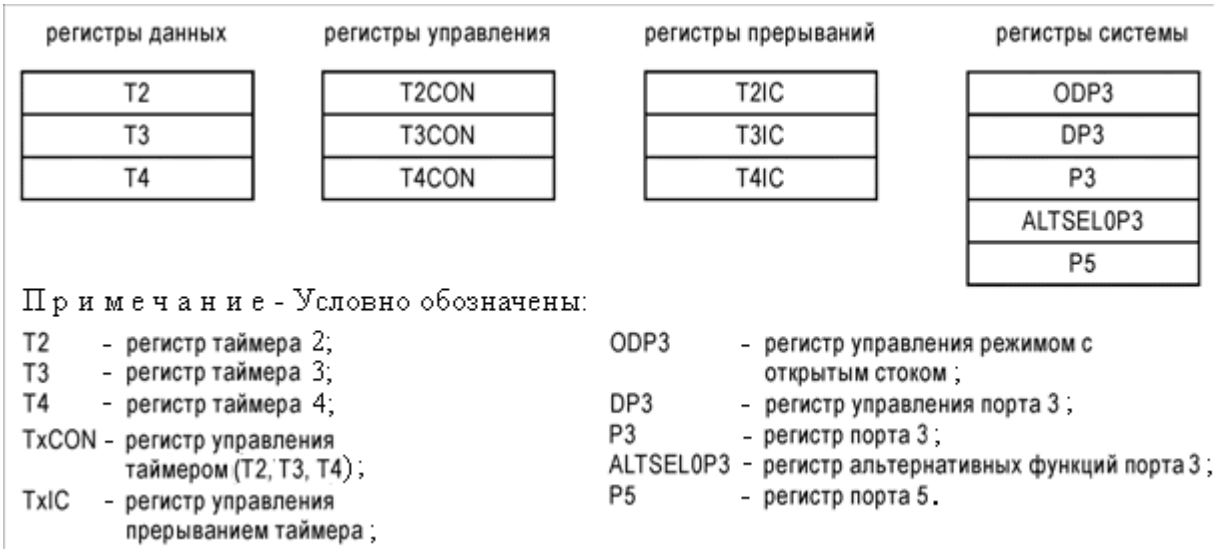


Рисунок 14.1 – Регистры и выводы портов блока GPT1

Все регистры блока GPT1 расположены в областях памяти SFR и ESFR. Все три таймера блока GPT1 могут работать в четырех основных режимах: режим таймера, режим внешнего управления таймером, режим считчика, режим внешнего инкрементирования по двум входам. Все таймеры могут считать как в положительном, так и отрицательном направлении. Каждым таймером управляет отдельный регистр управления таймером TxCON. У каждого таймера есть вход (линия TxIN), который является альтернативной функцией вывода порта P3. Этот вход может использоваться в качестве управления логикой в режиме внешнего управления таймером или в качестве входа отсчета для режима счетчика. Направление отсчета может быть запрограммировано программно, а также может изменяться по ходу работы при помощи внешнего управляющего сигнала

(линия TxEUD). Сигнал переполнения/опустошения таймера T3 может быть выведен через вывод порта в качестве функции альтернативного вывода T3OUT. Вспомогательные таймеры T2 и T4 могут быть связаны с основным таймером T3 через линию T3OTL, или могут использоваться как регистры захвата или перезагрузки для основного таймера.

Текущее содержимое каждого таймера не может быть прочтено, но может быть изменено с помощью ЦП посредством доступа к соответствующим регистрам таймеров T2, T3, T4, расположенным в неадресуемом побитно пространстве SFR-регистров. В случае совершения записи значения в регистр таймера при помощи ЦП перед инкрементированием или декрементированием значения таймера или перед совершением захвата значения таймера, более высокий приоритет имеет команда ЦП для обеспечения правильности результата, см. рисунок 14.2.

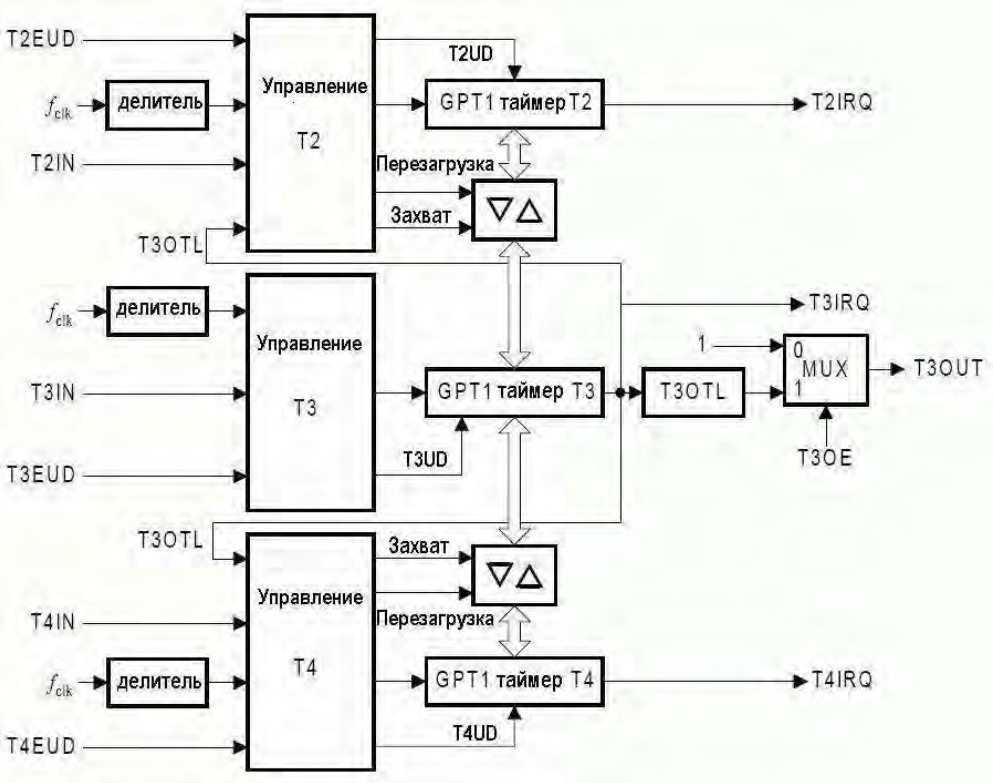


Рисунок 14.2 – Структурная схема блока GPT1

14.1.1 Основной таймер T3 блока GPT1

Основной таймер T3 настраивается и управляется с помощью побитно адресуемого регистра T3CON, см. рисунки 14.3, 14.4 и таблицы 14.1, 14.2. Регистр таймера T3 доступен только для записи.

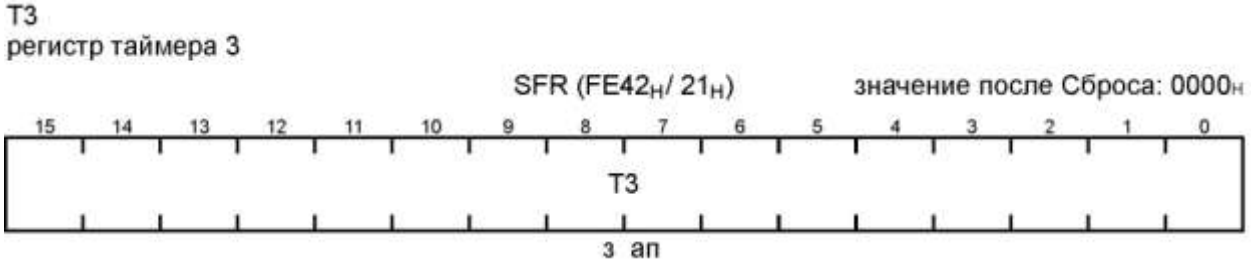


Рисунок 14.3 – Формат регистра T3

Таблица 14.1 – Функциональное назначение полей регистра T3

Поле	Биты	Тип	Описание
T3	15-0	Запись Аппаратное влияние	Таймер 3 Текущее значение таймера T3

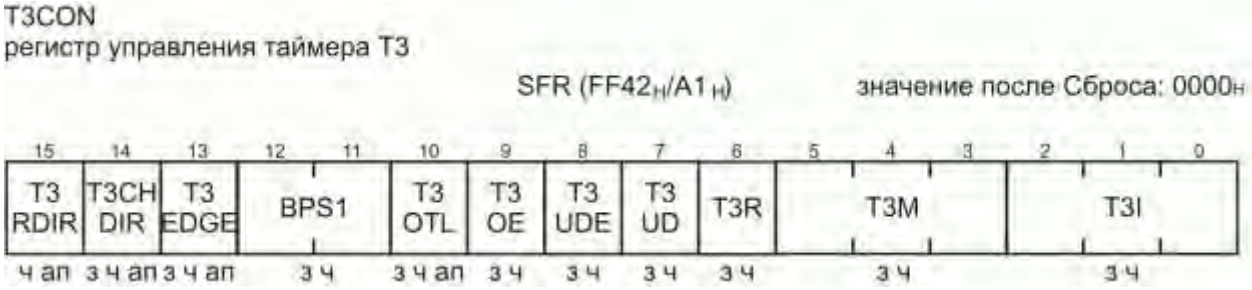


Рисунок 14.4 – Формат регистра T3CON

Таблица 14.2 – Функциональное назначение полей регистра T3CON

Поле	Биты	Тип	Описание
1	2	3	4
T3I	2-0	Чтение Запись	Выбор режима тактирования таймера T3. Режим таймера: комбинации в таблице 14.4. Режим внешнего управления: комбинации в таблице 14.4. Режим счетчика: комбинации в таблице 14.5. Режим внешнего инкрементирования по двум входам: комбинации в таблице 14.6
T3M	5-3	Чтение Запись	Выбор режима таймера T3: 000 Режим таймера. 001 Режим счетчика. 010 Режим внешнего управления низким активным уровнем. 011 Режим внешнего управления высоким активным уровнем. 100 Зарезервировано. 101 Зарезервировано. 110 Режим внешнего инкрементирования (определение периода). 111 Режим внешнего инкрементирования (обнаружение фронта).
T3R	6	Чтение Запись	Бит работы таймера T3: 0 Таймер/счетчик остановлен. 1 Таймер/счетчик работает.
T3UD	7	Чтение Запись	Бит установки направления счета таймера T3: 0 Таймер считает «вверх». 1 Таймер считает «вниз».

Окончание таблицы 14.2

1	2	3	4
T3UDE	8	Чтение Запись	Бит разрешения внешнего управления направлением счета T3: 0 Направление счета таймера контролируется программно. 1 Направление счета таймера контролируется посредством линии T3EUD.
T3OE	9	Чтение Запись	Бит разрешения вывода сигнала о переполнении/ опустошении таймера T3: 0 Переполнение/опустошение таймера T3 не детектируется извне. 1 Переполнение/опустошение таймера T3 детектируется по сигналу на линии T3OUT.
T3OTL	10	Чтение Аппаратное влияние	Бит выходной защелки таймера T3. Этот бит переключается каждый раз при переполнении/ опустошении таймера T3. Бит может быть установлен/сброшен программно.
BPS1	12-11	Чтение Запись	Делитель блока таймеров GPT1: Максимальная входная частота сигнала тактирования[1)] 00 $f_{CLK}/8$ 01 $f_{CLK}/4$ 10 $f_{CLK}/32$ 11 $f_{CLK}/16$
T3EDGE	13	Чтение Запись Аппаратное влияние	Флаг обнаружения фронта сигнала на входе таймера T3. Флаг устанавливается при каждом обнаружении корректного фронта входного сигнала. 0 Фронт сигнала не обнаружен. 1 Фронт сигнала обнаружен. Бит должен очищаться программно.
T3CHDIR	14	Чтение Запись Аппаратное влияние	Флаг изменения направления счета таймера T3. Этот бит устанавливается каждый раз при изменении направления счета. 0 Изменения направления счета не обнаружено. 1 Обнаружено изменение направления счета. Бит должен сбрасываться программно.
T3RDIR	15	Чтение Аппаратное влияние	Бит направления счета таймера T3: 0 Таймер T3 считает «вверх» (инкрементирование). 1 Таймер T3 считает «вниз» (декрементирование).

[1)] Дополнительно частота входного сигнала тактирования может быть изменена, см. T3I для режима таймера, режима счетчика и режима внешнего управления.

Управление работой таймера T3

Таймер T3 можно запускать и останавливать программно. Установка бита T3R запускает таймер, очистка T3R останавливает таймер. В режиме внешнего управления таймер будет работать, только если T3R = 1_B и выбран активный уровень («0» или «1», в зависимости от запрограммированного значения).

Примечание – Когда бит T2RC/T4RC установлен в регистрах управления T2CON/T4CON, бит T3R будет также управлять вспомогательным таймером T2/T4.

Управление направлением счета

Направление счета таймеров блока GPT1 (основной таймер и вспомогательные таймеры) может управляться программно или с помощью внешнего входа. При программном управлении (бит T3UDE = 0_B) направление счета может быть изменено установкой или очисткой бита T3UD. Когда бит T3UDE = 1_B, то источником управления направления счета будет внешний вывод T3EUD. Направление счета можно изменять вне зависимости от того, работает таймер или нет. Для этого используется бит T3UD, см. таблицу 14.3. Когда вывод T3UED/P3.4 используется для управления направлением счета, этот вывод должен быть сконфигурирован как вход, то есть DP3.4 = 0_B.

Таблица 14.3 – Управление направлением счета блока GPT1

Линия T3EUD	Бит T3UDE	Бит T3UD	Направление счета
X	0	0	Инкрементирование
X	1	1	Декрементирование
0	0	0	Инкрементирование
1	0	0	Декрементирование
0	1	1	Декрементирование
1	1	1	Инкрементирование
Примечание – Управление направлением счета идентично для таймера T3 и для вспомогательных таймеров T2 и T4.			

Переполнение/опустошение таймера T3

Сигнал переполнения/опустошения с выхода таймера T3 подключен к схеме выходного триггера.

Переполнение или опустошение таймера T3 изменяет значение бита T3OTL регистра T3CON. Бит T3OTL может быть установлен и сброшен программно. Бит T3OE (разрешение вывода переполнения и опустошения) регистра T3CON разрешает наблюдение через внешнюю линию T3OUT. Если эта линия связана с внешним выводом (сконфигурирована как выход, DP3.3 = 1_B), T3OUT можно использовать для управления внешним устройством.

Таймер T3OTL может использоваться при переполнении и опустошении таймера в качестве входа тактового сигнала счетчика, или как источник переключения для функции перезагрузки вспомогательных таймеров T2 и T4. При использовании этой функции, состояние T3OTL не может быть выведено через T3OUT, поскольку в этом случае реализуется внутреннее соединение.

Режимы работы основного таймера T3 блока GPT1

Таймер T3 в режиме таймера

Режим таймера T3 выбирается путем установки 000_B в битовом поле T3M регистра T3CON. Структурная схема работы таймера T3 в режиме таймера показана на рисунке 14.5.

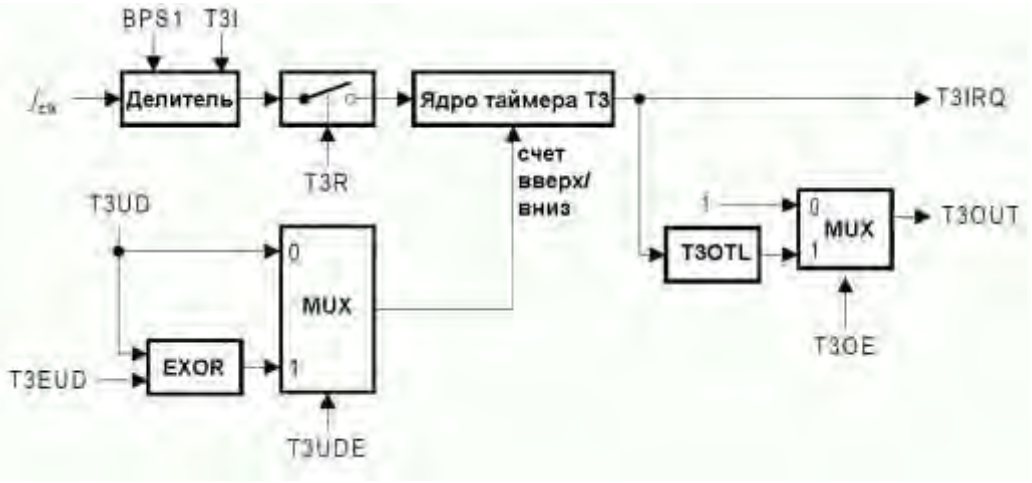


Рисунок 14.5 – Структурная схема работы таймера T3 в режиме таймера

В этом режиме тактовый сигнал f_{CLK}, МГц, приходит на программируемый блок делителя, которым управляют битовые поля T3I и BPS1 регистра T3CON. Входная частота f_{T3} таймера и длительность такта r_{T3}, мс, линейно зависят от частоты ЦП и рассчитываются по формулам

$$f_{T3} = \frac{f_{CLK}}{<BPS1> \times 2^{<T3I>}}, \quad (14.1)$$

$$r_{T3} = \frac{<BPS1> \times 2^{<T3I>}}{f_{CLK}} \quad (14.2)$$

Формулы (14.1) и (14.2) применимы к таймеру T3 в режиме внешнего управления и к вспомогательным таймерам T2 и T4 в режимах таймера и внешнего управления.

Таблица 14.4 – Выбор входных параметров для таймера T3: режим таймера и режим внешнего управления таймером

T3I	Делитель для f_{CLK}, МГц, BPS1 = 00_B	Делитель для f_{CLK}, МГц, BPS1 = 01_B	Делитель для f_{CLK}, МГц, BPS1 = 10_B	Делитель для f_{CLK}, МГц, BPS1 = 11_B
000_B	8	4	32	16
001_B	16	8	64	32
010_B	32	16	128	64
011_B	64	32	256	128
100_B	128	64	512	256
101_B	256	128	1024	512
110_B	512	256	2048	1024
111_B	1024	512	4096	2048

T3 в режиме внешнего управления таймером

Режим внешнего управления таймером T3 выбирается установкой 010_B или 011_B в битовом поле T3M регистра T3CON.

Структурная схема работы таймера T3 в режиме внешнего управления таймером представлена на рисунке 14.6.

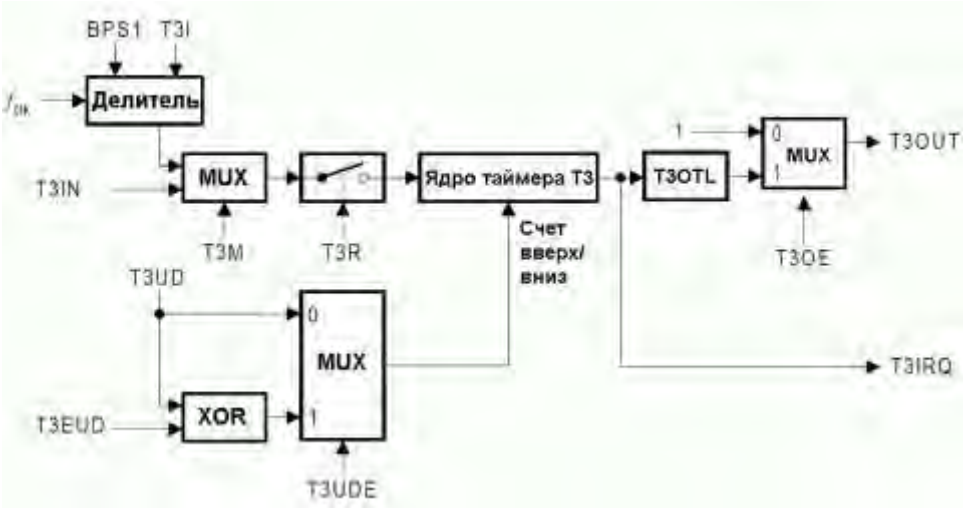


Рисунок 14.6 – Структурная схема работы таймера T3 в режиме внешнего управления таймером

Значения бита T3M.0 (T3CON.3) указывает на активный уровень напряжения на входе внешнего управления. В режиме внешнего управления таймером входная частота устанавливается по правилам, аналогичным режиму таймера. Однако сигнал тактового генератора на входе таймера T3 отключается после подачи сигнала на вход T3IN, являющейся альтернативной функцией вывода P3.6 порта P3. Для работы в этом режиме необходимо сконфигурировать P3.6 как вход DP3.6 = 0_B.

Если T3M = 010_B, то таймер будет работать, пока на T3IN держится «0». Как только на T3IN появиться «1», таймер остановиться.

Если T3M = 011_B, то таймер работает, когда на T3IN держится «1».

Дополнительно таймер можно программно включить или выключить, изменяя значение бита T3R. Таймер будет находиться в рабочем режиме только в том случае, когда выполняются сразу оба вышеперечисленные условия, то есть T3R = 1_B и T3IN – на активном уровне.

Примечание – Подача сигнала внешнего управления на вход T3IN не вызовет запрос на прерывание.

Таймер T3 в режиме счетчика

Режим счетчика таймера T3 выбирается путем установки 001_B в битовом поле T3M регистра T3CON. В этом режиме отсчеты таймера производятся по фронту сигнала на входе T3IN, являющегося альтернативной функцией P3.6. Фронт сигнала, приводящего к увеличению или уменьшению значения в счетчике, может быть как положительный, так и отрицательный, также оба фронта могут производить изменения значения счетчика. Битовое поле T3I управляющего регистра T3CON выбирает фронт, см. рисунок 14.7 и таблицу 14.5.

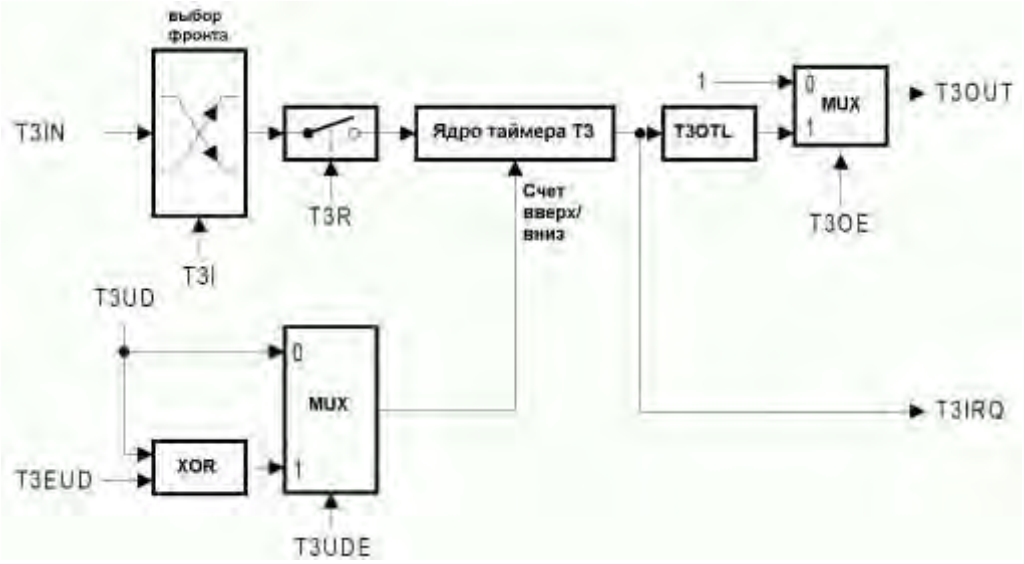


Рисунок 14.7 – Структурная схема работы таймера T3 в режиме счетчика

Таблица 14.5 – Выбор фронта срабатывания таймера T3 в режиме счетчика

T3I	Фронт срабатывания для увеличения/уменьшения значения счетчика
000_B	Счетчик отключен
001_B	Положительный фронт на входе T3IN
010_B	Отрицательный фронт на входе T3IN
011_B	Положительный и отрицательный фронты на входе T3IN
$1xx_B$	Зарезервировано. Не использовать

Для работы счетчика необходимо настроить T3IN (P3.6) как вход, то есть DP3.6 = 0_B. Максимальная входная частота, допустимая в режиме счетчика, составляет $f_{CLK}/8$ (BPS1 = 01_B). Для уверенности в том, что фронт входного сигнала счетчика воспринимается корректно, необходимо, как минимум, четыре такта f_{CLK} (BPS1 = 01_B) сохранять значение на входе постоянным.

Таймер T3 в режиме внешнего инкрементирования

Режим внешнего инкрементирования таймера T3 выбирается путем установки 110_B или 111_B в битовом поле T3M регистра управления T3CON, см. рисунок 14.8. В этом режиме два входа, связанные с таймером T3 (T3IN, T3EUD), используются для соединения с внешним кодером. T3 тактируется каждым фронтом одной или двух внешних линий, который дает 2-кратное или 4-кратное разрешение входов кодера.

Битовое поле T3I в регистре управления T3CON выбирает фронт для запуска, см. таблицу 14.6 и рисунок 14.8.

Таблица 14.6 – Таймер T3 – режим внешнего инкрементирования, выбор входного фронта

T3I	Выбор фронта для положительного или отрицательного счета
000_B	Счетчик не считает
001_B	Любой фронт (положительный или отрицательный) на T3IN
010_B	Любой фронт (положительный или отрицательный) на T2EUD
011_B	Любой фронт (положительный или отрицательный) на любом из входов T3 (T3IN или T3EUD)
$1xx_B$	Зарезервировано. Не использовать

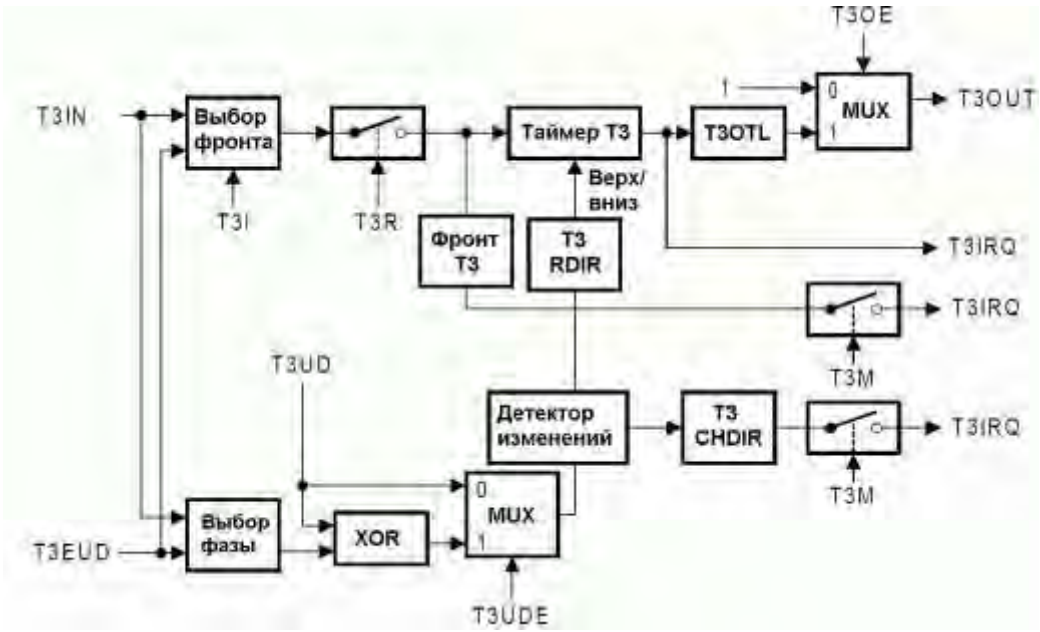


Рисунок 14.8 – Структурная схема работы таймера T3 внешнего инкрементирования

Последовательность импульсов двух входных сигналов оценивается, затем генерируются счетные импульсы для таймера, и выбирается направление счета. В зависимости от выбранного режима определения периода (T3M = 110_B) или определения фронта (T3M = 111_B), формируется запрос на прерывание T3IRQ. В режиме определения периода прерывание генерируется каждый раз при изменении направления счета таймера T3. В режиме определения фронта прерывание генерируется каждый раз при счете таймера T3. Направление счета, изменение направления счета и запросы на счет контролируются с помощью статусных битов T3RDIR, T3CHDIR, T3EDGE в регистре управления T3CON. Содержимое таймера T3 всегда представляет собой текущее значение.

Кодер для инкрементирования может быть непосредственно подключен к микроконтроллеру без логики внешнего интерфейса. В стандартной системе будут использоваться компараторы, чтобы преобразовать дифференциальные выходы кодера такие как A, A# к цифровым сигналам, таким как «A», см. рисунок 14.9.

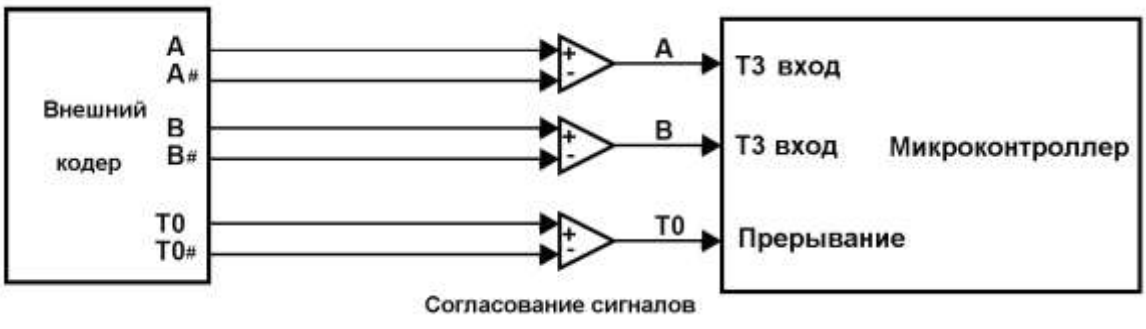


Рисунок 14.9 – Интерфейс кодера и микроконтроллера

Примечание – Третий выход T0 кодера, который указывает начальное положение, может быть подключен к внешнему прерыванию, чтобы вызвать сброс таймера T3.

Для работы в режиме внешнего инкрементирования необходимо соблюдать следующие условия:

- битовое поле T3M должно равняться 110_B или 111_B;

- выводы контроллера, связанные с линиями T3IN и T3EUD, должны быть сконфигурированы как входы, т. е. соответствующие биты управления конфигурацией должны быть равны «0»;

- бит T3UDE должен быть равен «1» для автоматической конфигурации выводов.

Максимально разрешенная частота в режиме внешнего инкрементирования является $f_{CLK}/8$, $BPS1 = 01_B$. Чтобы гарантировать правильное распознание любого изменения входного сигнала, его уровень (низкий или высокий) должен держаться, по крайней мере, четыре f_{CLK}, $BPS1 = 01_B$, прежде чем измениться. В режиме внешнего инкрементирования направление счета автоматически берется из последовательности, в которой изменение входных сигналов соответствует направлению вращения подключенного датчика. В таблице 14.7 объединены возможные варианты.

Таблица 14.7 – Таймер T3 – режим внешнего инкрементирования, направления счета

Уровень другого входа	Вход T3IN		Вход T3EUD	
	Положительный фронт	Отрицательный фронт (спад)	Положительный фронт	Отрицательный фронт (спад)
Высокий уровень	Счет «вниз»	Счет «вверх»	Счет «вверх»	Счет «вниз»
Низкий уровень	Счет «вверх»	Счет «вниз»	Счет «вниз»	Счет «вверх»

На рисунках 14.10 и 14.11 показаны примеры работы таймера T3, которые отображают генерирование счетных сигналов и управление направлением. На примерах также видно, как компенсируется входной джиттер (дрожание), который возникает в случае отсутствия сигнала датчика в моменты переключения таймера.

Рисунок 14.10 – Пример работы таймера T3 в режиме внешнего инкрементирования $T3I = 011_B$

Примечание – Пример на рисунке 14.10 отражает работу таймера в условиях, когда таймер T3 подсчитывает каждое переключение сигнала на обоих входах, то есть $T3I = 011_B$.

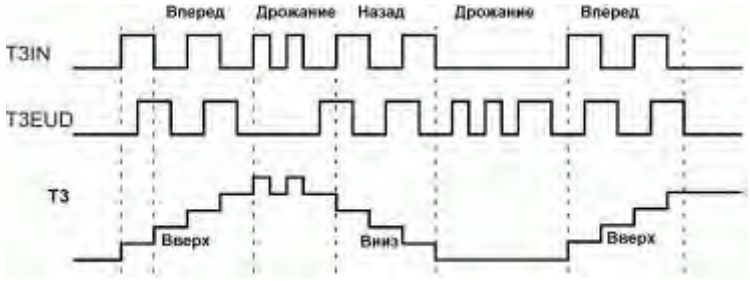


Рисунок 14.11 – Пример работы таймера T3 в режиме внешнего инкрементирования $T3I = 001_B$

Примечание – Пример на рисунке 14.11 отражает работу таймера в условиях, когда таймер T3 подсчитывает каждое переключение сигнала на входе T3IN, то есть $T3I = 001_B$.

Таймер T3, работающий в режиме внешнего инкрементирования автоматически хранит информацию о текущем состоянии датчика. Динамическая информация: скорость, ускорение, замедление – может быть получена при измерении периода входного сигнала.

14.1.2 Вспомогательные таймеры T2 и T4

Функциональное назначение полей регистров T2 и T4 (регистры таймеров T2 и T4 доступны только для записи) представлено в таблице 14.8, формат регистра T2 – на рисунке 14.12, формат регистра T4 – на рисунке 14.13. Функциональное назначение полей регистров T2CON и T4CON представлено в таблице 14.9, формат регистра T2CON – на рисунке 14.14, формат регистра T4CON – на рисунке 14.15.

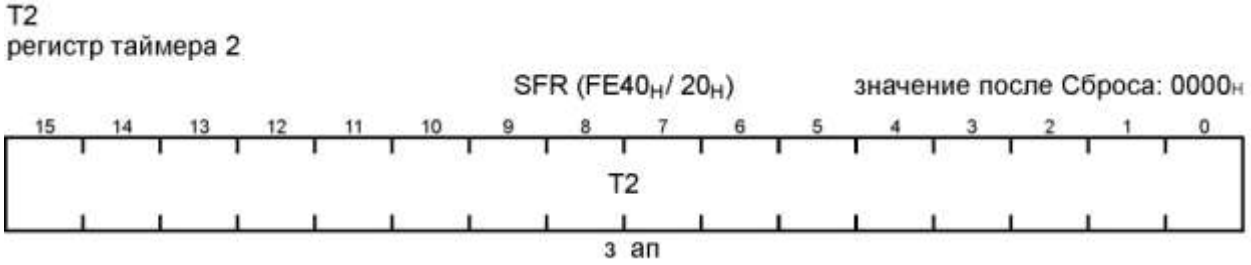


Рисунок 14.12 – Формат регистра T2

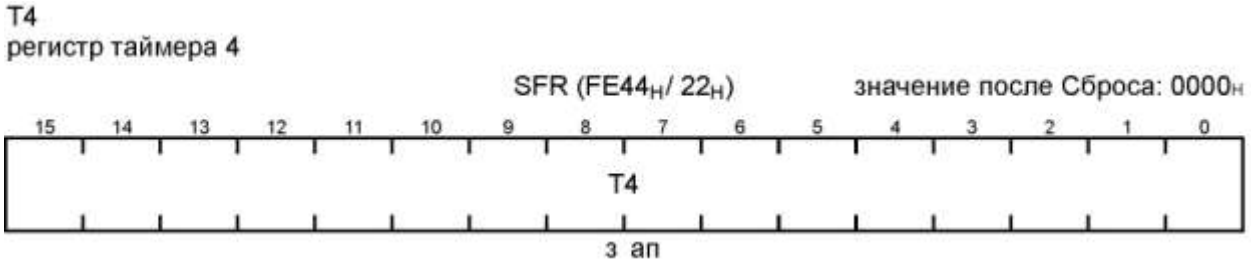


Рисунок 14.13 – Формат регистра T4

Таблица 14.8 – Функциональное назначение полей регистров T2 и T4

Поле	Биты	Тип	Описание
Tx[1)]	15-0	Запись Аппаратное влияние	Таймер Tx Текущее значение таймера Tx
[1)] x = 2, 4.			

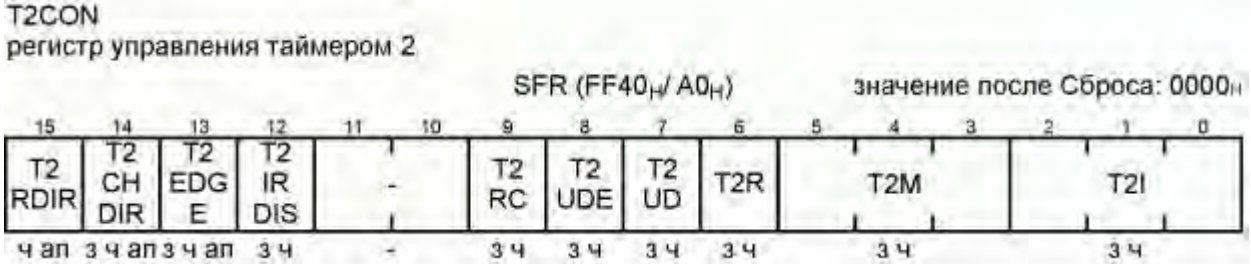


Рисунок 14.14 – Формат регистра T2CON

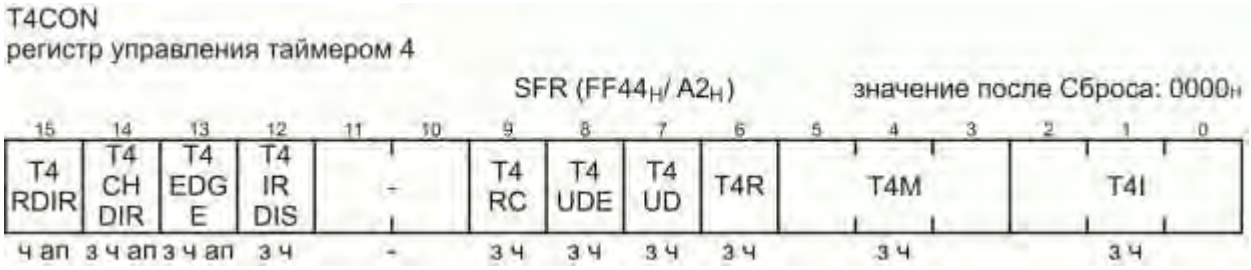


Рисунок 14.15 – Формат регистра T4CON

Таблица 14.9 – Функциональное назначение полей регистров T2CON и T4CON

Поле	Биты	Тип	Описание
1	2	3	4
TxI	2-0	Чтение Запись	Выбор режима тактирования таймера Tx. Режим таймера: комбинации в таблице 14.10. Режим внешнего управления таймером: комбинации в таблице 14.10. Режим счетчика: комбинации в таблице 14.11. Режим внешнего инкрементирования: комбинации в таблице 14.12.
TxM	5-3	Чтение Запись	Выбор режима таймера Tx (основной режим работы): 000 Режим таймера. 001 Режим счетчика. 010 Режим внешнего управления активным низким уровнем. 011 Режим внешнего управления активным высоким уровнем. 100 Режим перезагрузки. 101 Режим захвата. 110 Режим внешнего инкрементирования (режим детектирования периодов). 111 Режим внешнего инкрементирования (режим детектирования фронтов).
TxR	6	Чтение Запись	Бит работы таймера Tx: 0 Таймер/счетчик остановлен. 1 Таймер/счетчик работает.
TxUD	7	Чтение Запись	Бит установки направления счета таймера Tx (когда TxUDE 0_B): 0 Таймер считает «вверх» (инкрементирование). 1 Таймер считает «вниз» (декрементирование).
TxUDE	8	Чтение Запись	Бит разрешения внешнего управления направлением счета таймера Tx: 0 Направление счета таймера контролируется программно. 1 Направление счета таймера контролируется посредством линии TxEUD.

Окончание таблицы 14.9

1	2	3	4
TxRC	9	Чтение Запись	Удаленный контроль таймером Tx: 0 Таймер/счетчик Tx управляется собственным битом работы TxR. 1 Таймер/счетчик Tx управляется битом работы основного таймера T3.
TxIRDIS	12	Чтение Запись	Запрет прерываний таймера Tx: 0 Формирование запросов для прерываний TxCHDIR и TxEDGE в режиме внешнего инкрементирования разрешено. 1 Формирования запросов для прерываний TxCHDIR и TxEDGE в режиме внешнего инкрементирования запрещено.
TxEDGE	13	Чтение Запись Аппаратное влияние	Бит обнаружения фронта входного сигнала таймера Tx. Этот бит устанавливается при каждом обнаружении фронта изменения уровня входного сигнала таймера Tx. 0 Фронт входного сигнала не обнаружен. 1 Обнаружен фронт входного сигнала. Бит должен очищаться программно.
TxCHDIR	14	Чтение Запись Аппаратное влияние	Бит изменения направления счета таймера Tx. Этот бит устанавливается каждый раз при изменении направления счета таймера Tx. 0 Изменение направления счета не обнаружено. 1 Обнаружено изменение направления счета. Бит должен очищаться программно.
TxRDIR	15	Чтение Аппаратное влияние	Индикатор направления счета таймера Tx: 0 Таймер считает «вверх». 1 Таймер считает «вниз».
[1)] x = 2, 4.			

Оба вспомогательных таймера T2 и T4 имеют одинаковый набор функций. Они могут быть сконфигурированы для работы в режиме таймера, в режиме внешнего управления таймером, в режиме счетчика или в режиме внешнего инкрементирования с такими же настройками тактовой частоты и входного сигнала, как для основного таймера T3. В дополнение к этим четырем режимам, вспомогательные таймеры могут быть объединены с основным таймером, или же они могут использоваться как регистры перезагрузки/захвата в совокупности с основным таймером.

Индивидуальные настройки таймеров T2 и T4 определяются их побитно адресуемыми регистрами управления T2CON и T4CON (имеют идентичную организацию).

Управление работой вспомогательных таймеров T2 и T4 осуществляется посредством их битов управления T2R/T4R регистра T2CON/T4CON. В то же время, можно запускать и останавливать T2/T4 с помощью удаленного контроля (установленные T2RC и T4RC), посредством бита управления T3R таймера T3.

Таймеры T2 и T4 в режиме таймера или режиме внешнего управления таймером

Работа таймеров T2 и T4 в режиме таймера или режиме внешнего управления таймером аналогична работе основного таймера T3. Описание, рисунки и таблицы, относящиеся к таймеру T3, применимы и к таймерам T2 и T4. Имеются только два различия:

- для таймеров T2 и T4 отсутствует выходной сигнал TxOUT;
- отсутствует контроль переполнения/опустошения (отсутствует бит TxOTL в регистре TxCON).

Выбор входных параметров таймера: режим таймера и режим внешнего управления таймером, представлен в таблице 14.10.

Таблица 14.10 – Выбор входных параметров таймера: режим таймера и режим внешнего управления таймером

TxI	Делитель для f_{CLK}			
	BPS1 = 00_B	BPS1 = 01_B	BPS1 = 10_B	BPS1 = 11_B
000_B	8	4	32	16
001_B	16	8	64	32
010_B	32	16	128	64
011_B	64	32	256	128
100_B	128	64	512	256
101_B	256	128	1024	512
110_B	512	256	2048	1024
111_B	1024	512	4096	2048

Таймеры T2 и T4 в режиме счетчика

В режиме счетчика каждый из таймеров T2 и T4 может тактироваться сигналом, приходящим на один из двух соответствующих ему входов TxIN или T3OTL.

Инкрементирование или декрементирование таймера Tx может инициироваться появлением положительного и/или отрицательного фронта входного сигнала на одном (выбранном) из двух входов TxIN или T3OTL, соответствующих этому таймеру, см. рисунок 14.16.

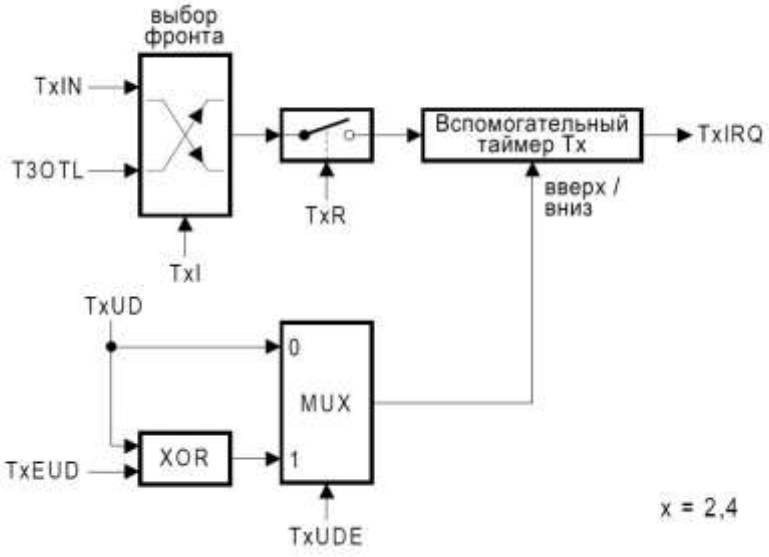


Рисунок 14.16 – Блок-схема вспомогательного таймера в режиме счетчика

Выбор режима осуществляется посредством битового поля TxI регистра TxCON, см. таблицу 14.11.

Таблица 14.11 – Выбор входного фронта вспомогательного таймера в режиме счетчика

TxI	Ожидаемый фронт входного сигнала для инициирования переключения таймера
$X00_B$	Счетчик Tx отключен
001_B	Положительный фронт на входе TxIN
010_B	Отрицательный фронт на входе TxIN
011_B	Положительный и отрицательный фронты на входе TxIN
101_B	Положительный фронт на входе T3OTL
110_B	Отрицательный фронт на входе T3OTL
111_B	Положительный и отрицательный фронты на входе T3OTL

Примечание – Только переполнение/опустошение таймера T3 будет влиять на счетчик таймера T2/T4 (вход T3OTL). Программное изменение значения T3OTL не окажет влияния на счетчик T2/T4.

При работе таймера Tx в режиме счетчика, его вывод, связанный с TxIN, должен быть сконфигурирован как вход. Максимально допустимая входная частота в режиме счетчика равна $f_{CLK}/8$ ($BPS1 = 01_B$). Для корректного детектирования и, соответственно, переключения счетчика таймера Tx, уровень входного сигнала на входе TxIN должен оставаться неизменным, как минимум, четыре такта f_{CLK} ($BPS1 = 01_B$).

Объединение таймеров

Использование T3OTL в качестве источника сигнала тактирования для вспомогательного таймера Tx блока GPT1 в режиме счетчика, объединяет этот таймер с основным таймером T3. В зависимости от того, какой фронт входного сигнала (на линии T3OTL) является инициирующим переключение счетчика вспомогательного таймера Tx, объединенные таймеры могут сформировать 32- или 33-битный таймер/счетчик.

32-битный таймер/счетчик: если для тактирования вспомогательного таймера Tx выбраны оба фронта входного сигнала (линия T3OTL), тогда этот таймер будет переключаться при каждом переполнении/опустошении основного таймера T3. И таким образом будет сформирован 32-разрядный таймер.

33-битный таймер/счетчик: если для тактирования вспомогательного таймера Tx выбран один из фронтов (положительный или отрицательный) входного сигнала (линия T3OTL), тогда этот таймер будет переключаться при каждом втором переполнении/опустошении основного таймера T3. И таким образом будет сформирован 33-разрядный таймер (16-битный основной таймер + T3OTL + 16-битный вспомогательный таймер).

Направления счета основного и вспомогательного таймеров, в случае их объединения, могут не совпадать. При этом таймер T3 может функционировать в режимах таймера, внешнего управления и счетчика.

Примечание – На изменение уровня сигнала на линии «*)», см. рисунок 14.17, оказывает влияние только переполнение/опустошение таймера T3. Программное изменение T3OTL игнорируется.

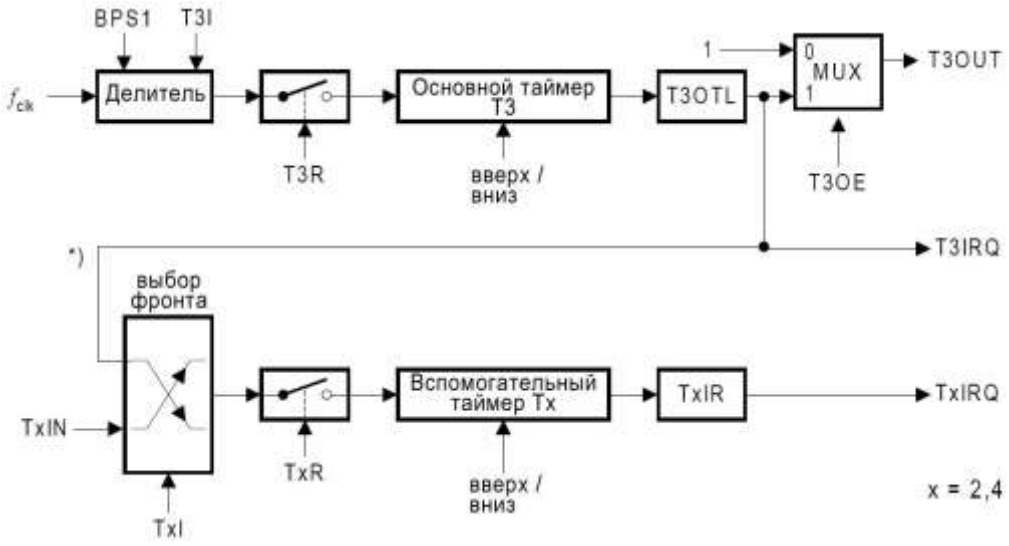


Рисунок 14.17 – Объединение основного таймера T3 и вспомогательного таймера Tx

Вспомогательный таймер в режиме перезагрузки

Режим перезагрузки вспомогательных таймеров T2 и T4 выбирается путем установки в битовом поле TxM регистра TxCON значения 100_B. В режиме перезагрузки в таймер T3 происходит загрузка содержимого регистра вспомогательного таймера, инициируемая одним из двух различных сигналов. Этот сигнал выбирается так же, как сигнал тактирования в режиме счетчика, см. таблицу 14.11.

Примечания

1 В режиме перезагрузки вспомогательный таймер T2 или T4 может быть остановлен программно, вне зависимости от состояния его флага T2R или T4R.

2 На рисунке 14.18 на изменение уровня сигнала на линии «*)» оказывает влияние только переполнение/опустошение таймера T3. Программное изменение T3OTL игнорируется.

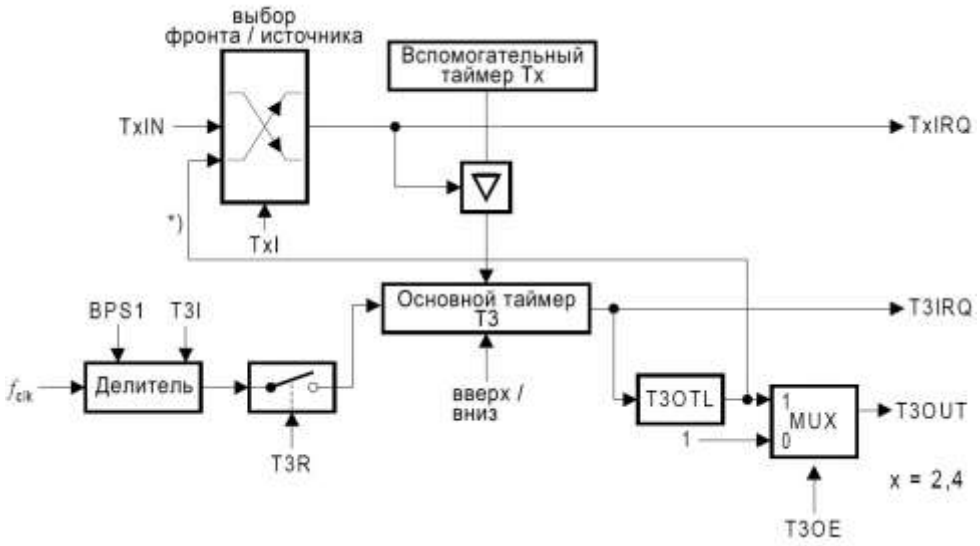


Рисунок 14.18 – Вспомогательный таймер Tx блока GPT1 в режиме перезагрузки

С появлением сигнала тактирования в таймер T3 загружается содержимое регистра соответствующего таймера T2 или T4, T2IRQ или T4IRQ переходят в состояние «1».

Примечание – Только переполнение/опустошение таймера T3 будет влиять на счетчик таймера T2/T4 (вход T3OTL). Программное изменение значения T3OTL не окажет влияния на счетчик T2/T4.

Режим перезагрузки с тактированием по входу T3OTL может использоваться в различных конфигурациях. В зависимости от выбранного типа активного фронта тактового сигнала, возможны следующие функции:

- Стандартный режим перезагрузки. Используются оба фронта сигнала тактирования по входу T3OTL. При переполнении/опустошении основного таймера, в него загружается содержимое вспомогательного таймера.

- Если используется один из фронтов сигнала тактирования по входу T3OTL, то загрузку в основной таймер содержимого вспомогательного таймера будет инициировать каждое второе переполнение/опустошение основного таймера.

- Возможность выбора одного из фронтов сигнала тактирования для каждого из вспомогательных таймеров (для каждого таймера может быть выбран свой фронт) позволяет осуществлять широтно-импульсную модуляцию. Если запрограммировать один из вспомогательных таймеров на тактирование передним (положительным) фронтом входного сигнала, а другой – задним (отрицательным), то основной таймер будет поочередно загружаться значениями вспомогательных таймеров.

На рисунке 14.19 показан пример генерирования ШИМ-сигнала с использованием механизма поочередной загрузки основного таймера. Таймер T2 устанавливает длительность высокого уровня ШИМ-сигнала (перезагрузка по переднему фронту), таймер T4 устанавливает длительность низкого уровня ШИМ-сигнала (перезагрузка по заднему фронту). ШИМ-сигнал может быть выведен на линию T3OUT, если разрешающий бит T3OE установлен. Использование такого механизма позволяет варьировать длительности высокого и низкого уровней ШИМ-сигнала в широком диапазоне.

Примечания

1 Значение T3OTL можно программно изменять для формирования желаемого ШИМ-сигнала. При этом перезагрузка таймера T3 происходить не будет.

2 Соответствующий вывод порта, связанный с линией T3OUT, должен быть сконфигурирован как выход.

3 Следует избегать выбора одного типа фронта сигнала тактирования для обоих вспомогательных таймеров, поскольку в этом случае оба таймера одновременно будут пытаться передать свои значения в основной таймер. В случае возникновения такой ситуации, приоритет имеет таймер T4 и его значение будет загружено в основной таймер.

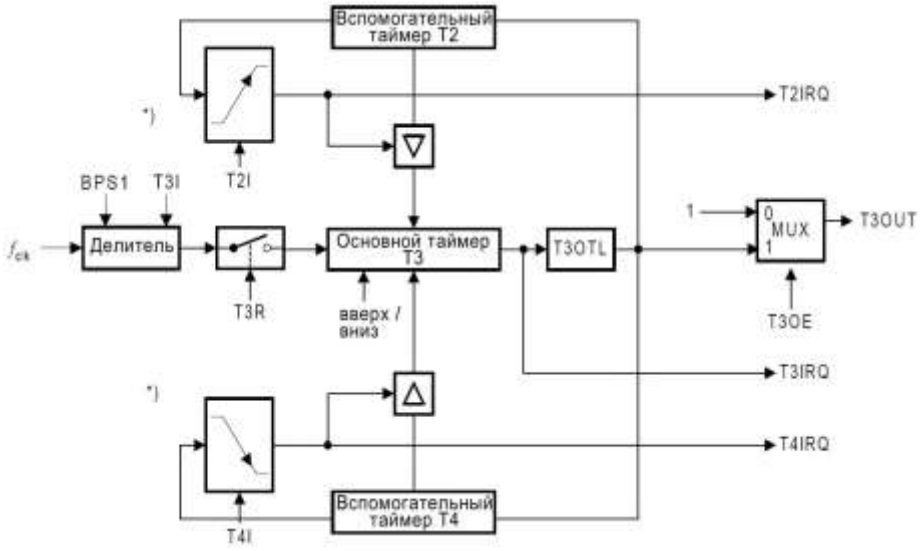


Рисунок 14.19 – Конфигурация таймера блока GPT1 для формирования PWM

Вспомогательный таймер в режиме захвата

Режим захвата для вспомогательных таймеров T2 и T4 выбирается установкой в битовом поле TxM регистра TxCON значения 101_B. В режиме захвата выбранный фронт сигнала тактирования, приходящий на вход TxIN вспомогательного таймера, инициирует загрузку содержимого основного таймера в регистр вспомогательного таймера. Можно использовать как передний или задний фронт сигнала тактирования на входе TxIN, так и оба фронта.

В режиме захвата два младших бита битового поля TxI используются для выбора фронта сигнала тактирования, см. таблицу 14.12, при этом значение старшего бита битового поля TxI не имеет значения, рекомендуется устанавливать TxI.2 = 0_B.

Примечание – Вспомогательный таймер T2 или T4 может быть остановлен программно, независимо от состояния его бита T2R или T4R.

Вспомогательный таймер блока GPT1 в режиме счета показан на рисунке 14.20.

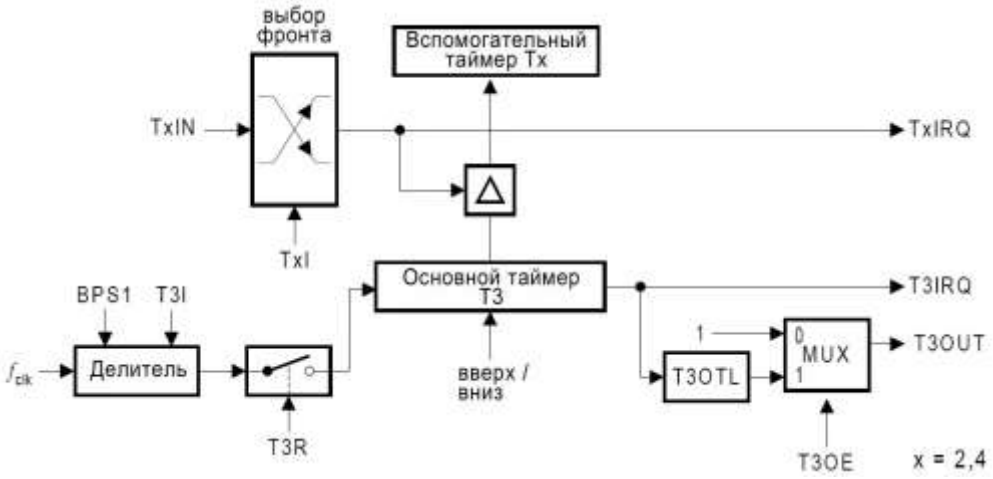


Рисунок 14.20 – Вспомогательный таймер Tx блока GPT1 в режиме счета

С появлением сигнала тактирования (выбранного) на входе TxIN вспомогательного таймера, в его регистр загружается содержимое основного таймера и TxIRQ переходят в состояние «1».

Примечание – Входы, связанные с T2IN и T4IN, должны быть сконфигурированы на вход, а уровень входного сигнала тактирования должен оставаться неизменным, по крайней мере, 4 такта f_{CLK} (BPS = 01_B) для его корректного детектирования.

Вспомогательный таймер в режиме внешнего инкрементирования

Работа таймеров T2 и T4 в режиме внешнего инкрементирования аналогична работе основного таймера T3. Описание, рисунки и таблицы, относящиеся к таймеру T3, применимы и к таймерам T2 и T4. Имеются только два исключения:

- для T2 и T4 отсутствует выходной сигнал TxOUT;
- отсутствует контроль переполнения/опустошения (отсутствует бит TxOTL в регистре TxCON).

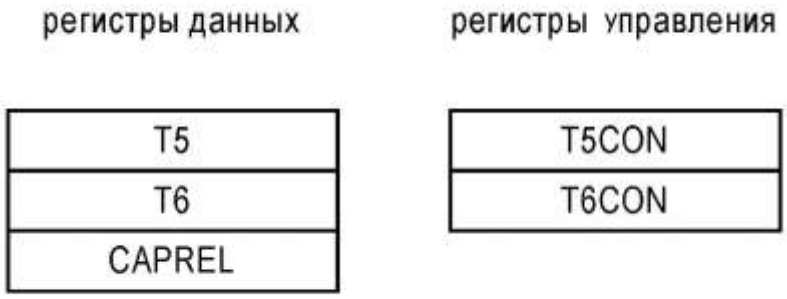
Выбор параметров таймера для режима внешнего инкрементирования представлен в таблице 14.12.

Таблица 14.12 – Выбор параметров таймера для режима внешнего инкрементирования

TxI	Ожидаемый фронт входного сигнала для инициирования переключения таймера
$X00_B$	Счетчик Tx отключен
001_B	Любой фронт на входе TxIN
010_B	Любой фронт на входе TxEUD
011_B	Любой фронт на входе TxIN или TxEUD
$1XX_B$	Зарезервировано. Не использовать

14.2 Функциональное описание таймеров блока GPT2

Все регистры блока GPT2 расположены в области SFR/ESFR, см. рисунок 14.21, структура блока GPT2 представлена на рисунке 14.21.



Примечание – Принятые условные обозначения:
- T5CON – регистр управления таймером 5;
- T6CON – регистр управления таймером 6;
- T5 – регистр таймера 5;
- T6 – регистр таймера 6;
- CAPREL – регистр захвата/перезагрузки.

Рисунок 14.21 – Регистры блока GPT2

Блок GPT2 включает в себя два таймера: вспомогательный таймер T5 и основной таймер T6, а также 16-битный регистр захвата/перезагрузки CAPREL. Каждый таймер блока GPT2 управляется отдельным регистром управления TxCON.

Каждый таймер имеет вход TxIN, который выполняет функцию управления в режиме внешнего управления и является входом сигнала тактирования в режиме счетчика. Направление счета «вверх»/«вниз» может как программироваться, так и динамически изменяться сигналом на внешнем выводе управления. Индикатором переполнения/опустошения основного таймера T6 является бит T6OTL, чье состояние может быть выведено на линии T6OUT и T6OFL. При перезагрузке, в таймер T6 может быть записано содержимое регистра CAPREL.

Имеется возможность объединения таймеров T6 и T5. Объединение таймера T6 с другими таймерами осуществляется через линию T6OUT.

По сигналу тактирования содержимое таймера T5 может записываться в регистр CAPREL. При необходимости, таймер T5 может быть сброшен.

Таймеры T6 и T5 могут считать как «вверх» (инкрементирование), так и «вниз» (декрементирование). ЦПУ может в любой момент времени изменить текущее значение регистра таймера T6 или T5.

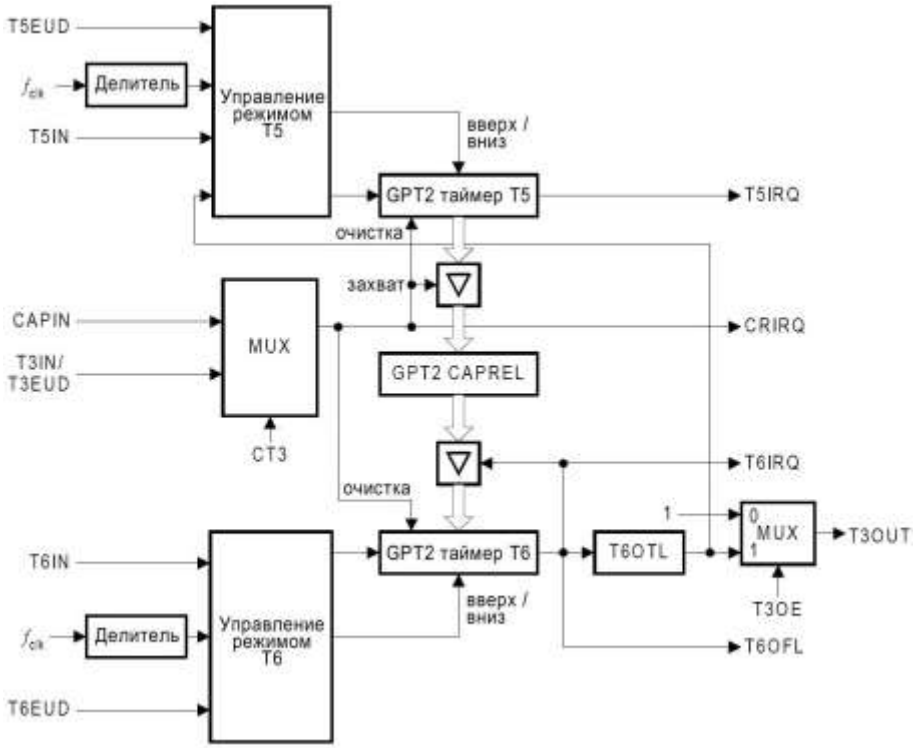


Рисунок 14.22 – Структура блока GPT2

14.2.1 Основной таймер T6

Регистр таймера T6 доступен только для записи. Работа основного таймера T6 контролируется посредством его побитно адресуемого регистра управления T6CON, см. рисунки 14.23, 14.24, таблицы 14.13, 14.14.

T6
регистр таймера 6

SFR ($FE48_H$/ 24_H) значение после Сброса: 0000_H

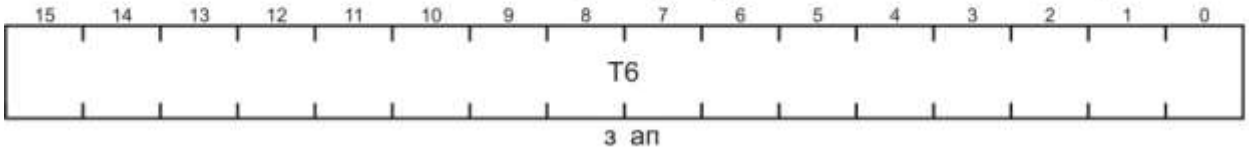


Рисунок 14.23 – Формат регистра T6

Таблица 14.13 – Функциональное назначение полей регистра T6

Поле	Биты	Тип	Описание
T6	15-0	Запись Аппаратное влияние	Таймер T6. Текущее значение таймера T6.

T6CON
регистр управления таймером 6

SFR ($FF48_H$/ $A4_H$) значение после Сброса: 0000_H

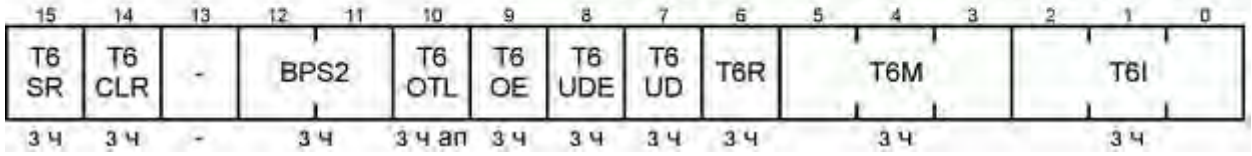


Рисунок 14.24 – Формат регистра T6CON

Таблица 14.14 – Функциональное назначение полей регистра T6CON

Поле	Биты	Тип	Описание
1	2	3	4
T6I	2-0	Чтение Запись	Выбор режима тактирования таймера T6. Режим таймера: комбинации в таблице 14.16. Режим внешнего управления таймером: комбинации в таблице 14.16. Режим счетчика: комбинации в таблице 14.17.
T6M	5-3	Чтение Запись	Выбор режима таймера T6 (основной режим работы): 000 Режим таймера. 001 Режим счетчика. 010 Режим внешнего управления активным низким уровнем. 011 Режим внешнего управления активным высоким уровнем. 1XX Зарезервировано. Не использовать.
T6R	6	Чтение Запись	Бит работы таймера T6: 0 Таймер/счетчик остановлен. 1 Таймер/счетчик работает.
T6UD	7	Чтение Запись	Бит установки направления счета таймера T6 (когда T6UDE = 0_B): 0 Таймер считает «вверх» (инкрементирование). 1 Таймер считает «вниз» (декрементирование).
T6UDE	8	Чтение Запись	Бит разрешения внешнего управления направлением счета таймера T6: 0 Направление счета таймера контролируется программно. 1 Направление счета таймера контролируется посредством линии T6EUD.
T6OE	9	Чтение Запись	Бит разрешения вывода сигнала о переполнении/ опустошении таймера T6: 0 Переполнение/опустошение таймера T6 не детектируется извне. 1 Переполнение/опустошение таймера T6 детектируется по сигналу на линии T6OUT.

Окончание таблицы 14.14

1	2	3	4
T6OTL	10	Чтение Запись Аппаратное влияние	Бит выходной защелки таймера T6. Этот бит переключается каждый раз при переполнении/ опустошении таймера T6. Бит может быть установлен/сброшен программно.
BPS2	12-11	Чтение Запись	Делитель блока таймеров GPT2: Максимальная входная частота сигнала тактирования[1)] 00 $f_{CLK}/4$ 01 $f_{CLK}/2$ 10 $f_{CLK}/16$ 11 $f_{CLK}/8$
T6CLR	14	Чтение Запись	Бит очистки таймера T6: 0 Таймер T6 не очищается при захвате. 1 Таймер T6 очищается после захвата.
T6SR	15	Чтение	Бит разрешения загрузи таймера T6: 0 Загрузка значения из регистра CAPREL запрещена. 1 Загрузка значения из регистра CAPREL разрешена.

[1)] Дополнительно частота входного сигнала тактирования может быть изменена (смотри T6I) для режима таймера, режима счетчика и режима внешнего управления.

Бит управления работой таймера T6

Таймер можно запускать и останавливать программно. Установка бита T6R запускает таймер, очистка T6R останавливает таймер. В режиме внешнего управления таймер будет работать, только если T6R = 1_B и выбран активный уровень «0» или «1», в зависимости от запрограммированного значения.

Примечание – Когда бит T5RC установлен в регистрах управления T5CON, бит T6R будет также управлять вспомогательным таймером T5.

Управление направлением счета

Направление счета основного таймера может контролироваться программно или с помощью внешнего входа управления T6EUD. За выбор источника управления отвечают биты T6UD и T6UDE регистра управления T6CON. При программном управлении бит T6UDE = 0_B направление счета может быть изменено установкой или очисткой бита T6UD. Когда бит T6UDE = 1_B, то источником управления направления счета будет внешний вывод T6EUD. Помимо прочего, бит T6UD может использоваться для реверса действительного направления счета. Если T6UD = 0_B и на линии T6EUD держится низкий уровень входного сигнала, таймер считает «вверх». Если на линии T6EUD держится высокий уровень входного сигнала, таймер считает «вниз». Если T6UD = 1_B, то высокий уровень входного сигнала на линии T6EUD устанавливает направление счета «вверх», а низкий уровень входного сигнала на линии T6EUD устанавливает направление счета «вниз». Направление счета можно изменять вне зависимости от того, работает таймер или нет.

Управление направлением счета таймера T6 согласно таблицы 14.15.

Таблица 14.15 – Управление направлением счета таймера Т6

Линия T6EUD	Бит T6UDE	Бит T6UD	Направление счета
X	0	0	«вверх» (инкрементирование)
X	0	1	«вниз» (декрементирование)
0	1	0	«вверх» (инкрементирование)
1	1	0	«вниз» (декрементирование)
0	1	1	«вниз» (декрементирование)
1	1	1	«вверх» (инкрементирование)
Примечание – Управление направлением счета идентично как для таймера Т6, так и для вспомогательного таймера Т5.			

Мониторинг переполнения/опустошения таймера Т6

Переполнение или опустошение таймера Т6 изменяет значение бита T6OTL регистра T6CON. Бит T6OTL может быть установлен и сброшен программно. Бит T6OE регистра T6CON разрешает вывод сигнала состояния регистра таймера Т6 T6OTL на линию T6OUT. Соответствующий вывод должен быть сконфигурирован как выход.

В дополнение, T6OTL может использоваться совместно (конъюнктивно) с переполнением/опустошением в качестве сигнала тактирования на входе вспомогательного таймера Т5. В этом случае значение T6OTL не должно выводиться на линию T6OUT, поскольку для этого существует внутреннее соединение.

Переполнение или опустошение таймера Т6 может использоваться для тактирования других таймеров. Для этого существует выход T6OFL.

Таймер Т6 в режиме таймера

Режим таймера Т6 выбирается путем установки 000_B в битовом поле T6M регистра T6CON. Структурная схема работы таймера Т6 в режиме таймера показана на рисунке 14.25. Выбор входных параметров для таймера Т6: режим таймера и режим внешнего управления таймером представлен в таблице 14.16.

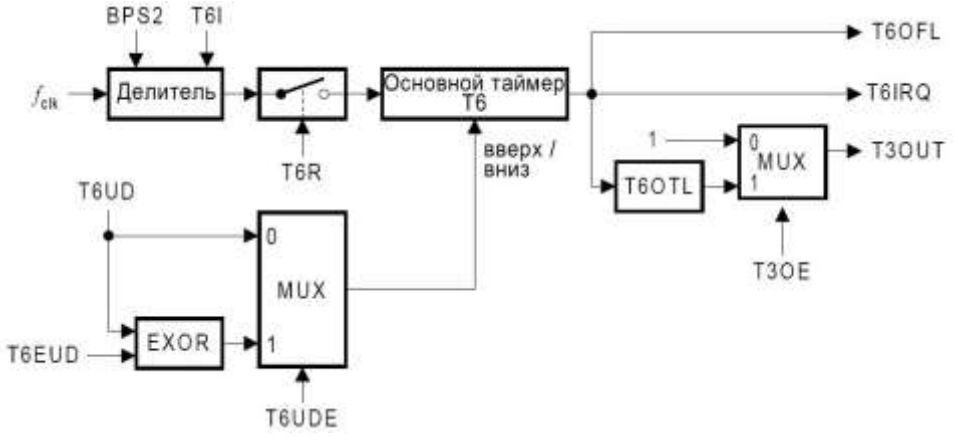


Рисунок 14.25 – Структурная схема работы таймера Т6 в режиме таймера

В этом режиме тактовый сигнал f_{CLK}, МГц, приходит на программируемый блок делителя, которым управляют битовые поле T6I и BPS2 регистра T6CON. Входная частота f_{T6}, МГц, таймера Т6 и длительность такта r_{T6}, мс, линейно зависят от частоты ЦП и рассчитываются по формулам

$$f_{T6} = \frac{f_{CLK}}{<BPS2> \times 2^{<T6I>}}, \qquad (14.3)$$

$$r_{T6} = \frac{<BPS2> \times 2^{<T6I>}}{f_{CLK}} \qquad (14.4)$$

Таблица 14.16 – Выбор входных параметров для таймера T6: режим таймера и режим внешнего управления таймером

T6I	Делитель для f_{CLK}			
	$BPS2 = 00_B$	$BPS2 = 01_B$	$BPS2 = 10_B$	$BPS2 = 11_B$
000_B	4	2	16	8
001_B	8	4	32	16
010_B	16	8	64	32
011_B	32	16	128	64
100_B	64	32	256	128
101_B	128	64	512	256
110_B	256	128	1024	512
111_B	512	256	2048	1024

T6 в режиме внешнего управления таймером

Режим внешнего управления таймером T6 выбирается установкой 010_B или 011_B в битовом поле T6M регистра T6CON, см. рисунок 14.26.

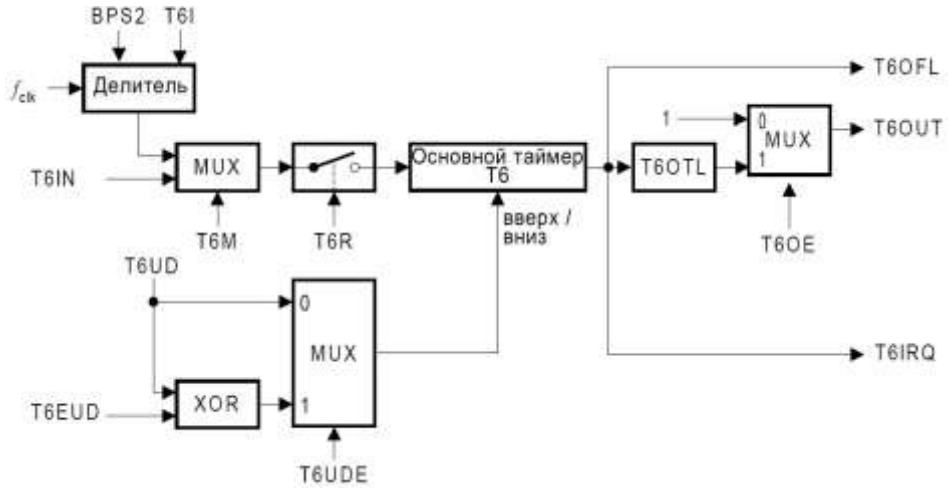


Рисунок 14.26 – Структурная схема работы таймера T6 в режиме внешнего управления таймером

Значения бита T6M.0 регистра T6CON.3 выбирает активный уровень напряжения на входе внешнего управления. В режиме внешнего управления таймером доступны те же опции входного сигнала тактирования, что и для режима таймера. Сигнал тактирования подается на вход T6IN.

Если младший бит поля $T6M = 0_B$, то таймер будет работать, пока на T6IN держится «0». Как только на T6IN появится «1», таймер остановится.

Если младший бит поля $T3M = 1_B$, то таймер работает, пока на T6IN держится «1».

Дополнительно таймер может быть запущен или остановлен программно, посредством бита T6R. Таймер будет работать только в случае одновременного выполнения условий: $T3R = 1_B$ и на T3IN активный уровень. При невыполнении одного из условий, таймер будет остановлен.

Примечание – Сигнал тактирования на входе T3IN не вызывает запрос на прерывание.

Таймер T6 в режиме счетчика

Режим счетчика таймера T6 выбирается путем установки 001_B в битовом поле T6M регистра T6CON. В этом режиме таймер переключается по фронту сигнала на входе T6IN. Этот сигнал является сигналом тактирования таймера. Переключения таймера могут инициироваться передним (положительным) фронтом сигнала тактирования или задним (отрицательным) фронтом, или обоими фронтами, в зависимости от значения битового поля T6I управляющего регистра T6CON, см. таблицу 14.17, рисунок 14.27.

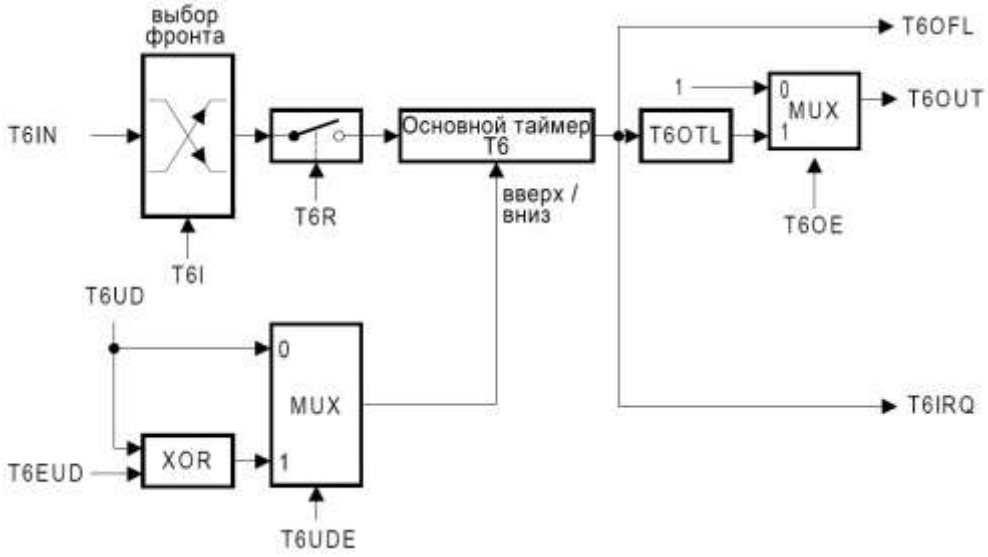


Рисунок 14.27 – Структурная схема работы таймера T6 в режиме счетчика

Таблица 14.17 – Выбор фронта срабатывания таймера T6 в режиме счетчика

T6I	Фронт сигнала тактирования таймера T6
000_B	Счетчик выключен
001_B	Передний положительный фронт на входе T6IN
010_B	Задний отрицательный фронт на входе T6IN
011_B	Любой фронт, передний или задний, на входе T6IN
$1xx_B$	Зарезервировано. Не использовать.

Максимальная входная частота, допустимая в режиме счетчика, составляет $f_{CLK}/4$ (BPS2 = 01_B). Для корректного определения уровень входного сигнала должен оставаться неизменным, как минимум, 2 такта f_{CLK} (BPS2 = 01_B).

14.2.2 Вспомогательный таймер T5

Регистр таймера T5 доступен только для записи. Вспомогательный таймер T5 может быть сконфигурирован для работы в режиме таймера, в режиме внешнего управления таймером или в режиме счетчика с такими же настройками частоты сигнала тактирования как для основного таймера T6. В дополнение к этим трем режимам, вспомогательный таймер может быть объединен с основным таймером, см. рисунки 14.28 – 14.31, таблицы 14.18 – 14.21.

Индивидуальные настройки таймера T5 определяются его побитно адресуемым регистром управления T5CON. Управление работой вспомогательного таймера T5 осуществляется посредством его бита управления T5R регистра T5CON. Также можно запускать и останавливать таймер T5 с помощью функции удаленного контроля (бит T5RC установлен), посредством бита управления T5R.

Примечание – Вспомогательный таймер T5 не имеет бит T5OTL, и вывод сигнала о его переполнении/опустошении не поддерживается.

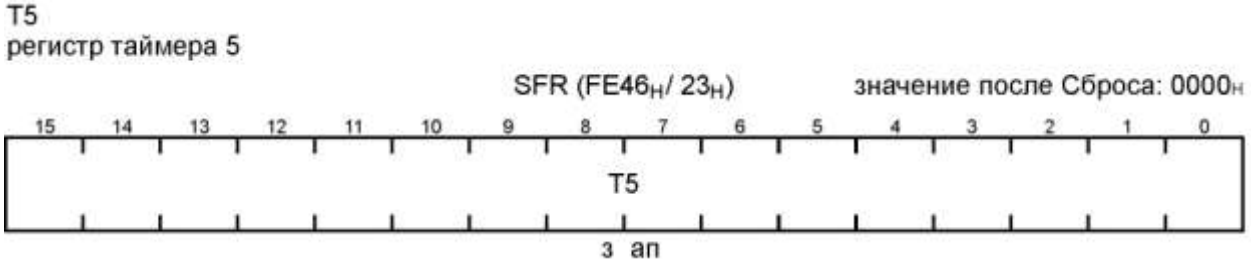


Рисунок 14.28 – Формат регистра T5

Таблица 14.18 – Функциональное назначение полей регистра T5

Поле	Биты	Тип	Описание
T5	15-0	Запись Аппаратное влияние	Таймер 5. Текущее значение таймера T5.

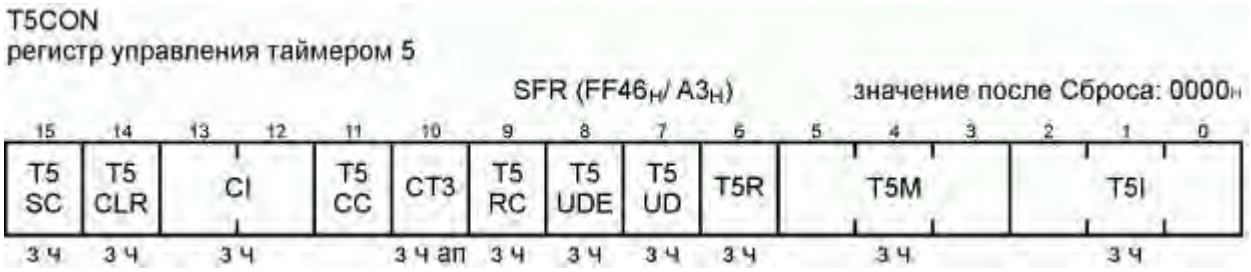


Рисунок 14.29 – Формат регистра T5CON

Таблица 14.19 – Функциональное назначение полей регистра T5CON

Поле	Биты	Тип	Описание
1	2	3	4
T5I	2-0	Чтение Запись	Выбор режима тактирования таймера T5. Режим таймера: комбинации в таблице 14.20. Режим внешнего управления таймером: комбинации в таблице 14.20. Режим счетчика: комбинации в таблице 14.21.
T5M	5-3	Чтение Запись	Выбор режима таймера T5 (основной режим работы): 000 Режим таймера. 001 Режим счетчика. 010 Режим внешнего управления активным низким уровнем. 011 Режим внешнего управления активным высоким уровнем. 1XX Зарезервировано. Не использовать.
T5R	6	Чтение Запись	Бит работы таймера T5: 0 Таймер/счетчик остановлен. 1 Таймер/счетчик работает.

Окончание таблицы 14.19

1	2	3	4
T5UD	7	Чтение Запись	Бит установки направления счета таймера T5 (когда T5UDE = 0_B): 0 Таймер считает «вверх» (инкрементирование). 1 Таймер считает «вниз» (декрементирование).
T5UDE	8	Чтение Запись	Бит разрешения внешнего управления направлением счета таймера T5: 0 Направление счета таймера контролируется программно. 1 Направление счета таймера контролируется посредством линии T5EUD.
T5RC	9	Чтение Запись	Бит удаленного контроля таймером Tx: 0 Таймер/счетчик T5 управляется собственным битом работы T5R. 1 Таймер/счетчик T5 управляется битом работы основного таймера T6.
CT3	10	Чтение Запись	Бит выбора линии таймера T3: 0 Захват содержимого таймера T5 по сигналу на линии CAPIN. 1 Захват содержимого таймера T5 по сигналу на линии T3IN и/или T3EUD.
T5CC	11	Чтение Запись	Бит коррекции при захвате содержимого таймера T5: 0 Содержимое таймера T5 захватывается без изменений. 1 Содержимое таймера T5 декрементируется на 1 перед захватом.
CI	13-12	Чтение Запись	Выбор события на линии таймера T3 для инициации захвата содержимого таймера T5 (в зависимости от бита CT3): 00 Захват запрещен. 01 Передний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3IN. 10 Задний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3EUD. 11 Любой фронт сигнала на входе CAPIN или любой фронт сигнала на входах T3IN или T3EUD.
T5CLR	14	Чтение Запись	Бит очистки таймера T5: 0 Таймер T5 не сбрасывается после захвата его содержимого. 1 Таймер T5 сбрасывается после захвата его содержимого.
T5SC	15	Чтение Запись	Бит разрешения захвата содержимого таймера T5: 0 Захват содержимого таймера T5 в регистр CAPREL запрещен. 1 Захват содержимого таймера T5 в регистр CAPREL разрешен.

Управление направлением счета вспомогательного таймера осуществляется аналогично управлению таймера T6.

Таймер T5 в режиме таймера или режиме внешнего управления таймером

Работа таймера T5 в режиме таймера или режиме внешнего управления таймером аналогична работе основного таймера T6, см. таблицу 14.20. Описание, рисунки и таблицы, относящиеся к таймеру T6, применимы и к таймеру T5. Имеются только три исключения:

- отсутствует вход T5OUT;
- отсутствует вход T5OFL;
- отсутствует контроль переполнения/опустошения (отсутствует бит T5OTL).

Таблица 14.20 – Выбор входных параметров таймера T5: режим таймера и режим внешнего управления таймером

T5I	Делитель для f_{CLK}			
	BPS2 = 00_B	BPS2 = 01_B	BPS2 = 10_B	BPS2 = 11_B
000_B	4	2	16	8
001_B	8	4	32	16
010_B	16	8	64	32
011_B	32	16	128	64
100_B	64	32	256	128
101_B	128	64	512	256
110_B	256	128	1024	512
111_B	512	256	2048	1024

Таймер T5 в режиме счетчика

Для таймера T5 режим счетчика выбирается записью в битовое поле T5M регистра T5CON значения 001_B. В режиме счетчика, см. рисунок 14.30, таблицу 14.21, таймер T5 может тактироваться сигналом, приходящим на его вход T5IN, или сигналом, приходящим на вход T6OTL таймера T6.

Инкрементирование и декрементирование таймера тактируется положительным и/или отрицательным фронтом сигнала приходящего на вход T5IN или с линии T6OTL.

Таблица 14.21 – Выбор фронта входного сигнала тактирования вспомогательного таймера T5 в режиме счетчика

T5I	Фронт входного сигнала тактирования таймера T5
$X00_B$	Счетчик T5 выключен
001_B	Передний положительный фронт на входе T5IN
010_B	Задний отрицательный фронт на входе T5IN
011_B	Любой фронт передний и задний на входе T5IN
101_B	Передний положительный фронт на входе T6OTL
110_B	Задний отрицательный фронт на входе T6OTL
111_B	Любой фронт, передний и задний, на входе T6OTL

Примечание – Только переполнение/опустошение таймера T6 будет влиять на счетчик таймера T5 (линия T6OTL). Программное изменение значения T3OTL не окажет влияния на счетчик T5.

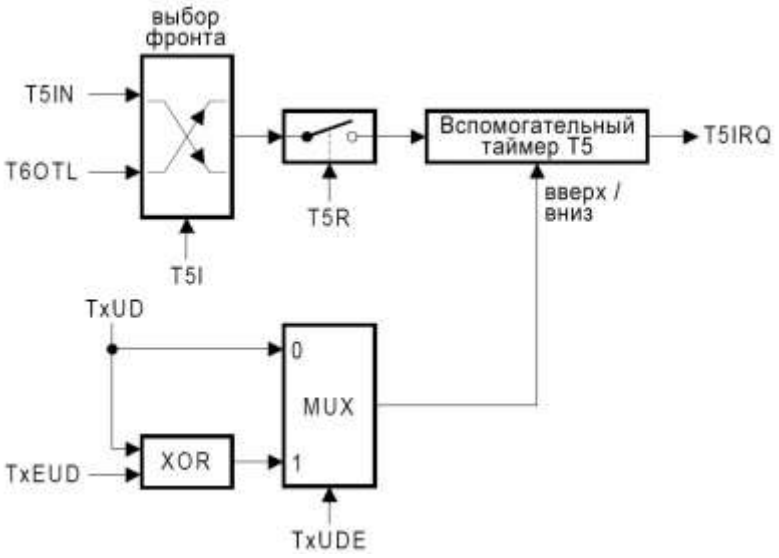


Рисунок 14.30 – Вспомогательный таймер T5 в режиме счетчика

При работе таймера T5 в режиме счетчика его вывод, связанный с T5IN, должен быть сконфигурирован как вход. Максимально допустимая входная частота в режиме счетчика равна $f_{CLK}/4$ ($BPS2 = 01_B$). Для корректного детектирования и, соответственно, переключения счетчика таймера T5, уровень входного сигнала на входе T5IN должен оставаться неизменным, как минимум, 2 такта f_{CLK} ($BPS2 = 01_B$).

Объединение таймеров

Использование T6OTL в качестве источника сигнала тактирования для вспомогательного таймера T5 блока GPT2 в режиме счетчика, объединяет этот таймер с основным таймером T6. В зависимости от того, какой фронт входного сигнала (на линии T6OTL) является инициирующим переключение счетчика вспомогательного таймера T5, объединенные таймеры могут сформировать 32- или 33-битный таймер/счетчик:

- 32-битный таймер/счетчик, если для тактирования вспомогательного таймера T5 выбраны оба фронта входного сигнала (линия T6OTL), тогда этот таймер будет переключаться при каждом переполнении/опустошении основного таймера T6. Таким образом, будет сформирован 32-разрядный таймер;

- 33-битный таймер/счетчик, если для тактирования вспомогательного таймера T5 выбран один из фронтов (положительный или отрицательный) входного сигнала (линия T6OTL), тогда этот таймер будет переключаться при каждом втором переполнении/опустошении основного таймера T6. Таким образом, будет сформирован 33-разрядный таймер (16-битный основной таймер, плюс T6OTL, плюс 16-битный вспомогательный таймер).

Направления счета основного и вспомогательного таймеров, в случае их объединения, могут не совпадать. При этом таймер T6 может функционировать в режимах таймера, внешнего управления и счетчика.

Примечание – На рисунке 14.31 на изменение уровня сигнала на линии «*)» оказывает влияние только переполнение/опустошение таймера T5. Программное изменение T6OTL игнорируется.

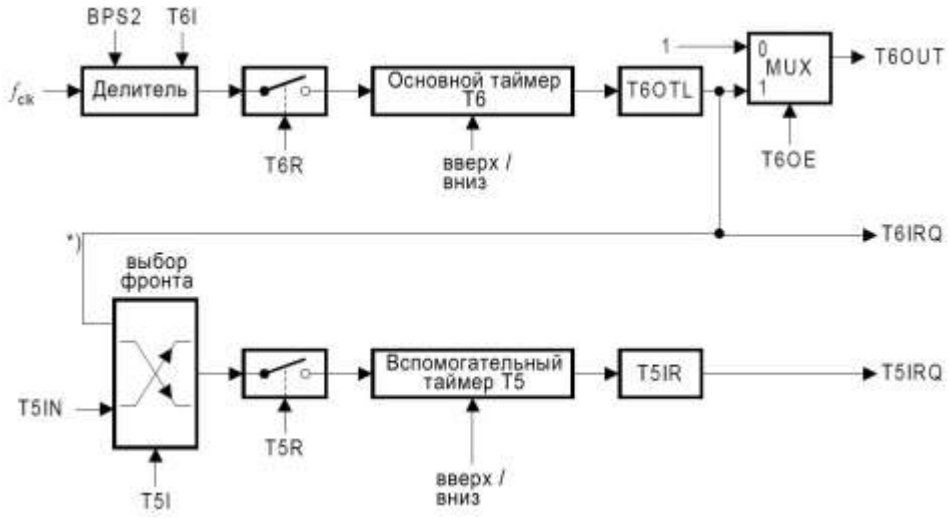


Рисунок 14.31 – Объединение основного таймера T6 и вспомогательного таймера T5

14.3 Регистр захвата/перезагрузки CAPREL

Регистр захвата/перезагрузки CAPREL доступен только для записи

Регистр захвата/перезагрузки CAPREL в режиме захвата

Функциональное назначение полей регистра CAPREL представлено в таблице 14.22, формат регистра CAPREL – на рисунке 14.32, регистр CAPREL блока GPT2 в режиме захвата – на рисунке 14.33.

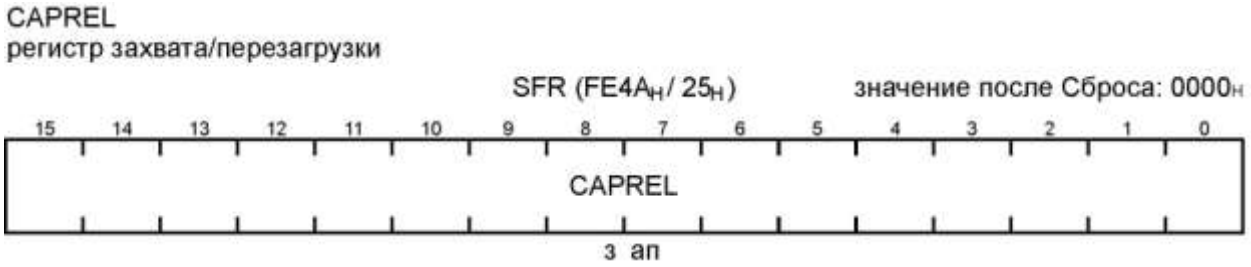


Рисунок 14.32 – Формат регистра CAPREL

Таблица 14.22 – Функциональное назначение полей регистра CAPREL

Поле	Биты	Тип	Описание
CAPREL	15-0	Запись Аппаратное влияние	Значение регистра захвата/перезагрузки

Этот 16-битный регистр может использоваться как регистр захвата для вспомогательного таймера T5. Этот режим выбирается установкой бита T5SC в регистре T5CON. Захват содержимого таймера T5 происходит по сигналу (далее – сигнал захвата), приходящему на выбранную битом CT3 входную линию (CAPIN либо вход T3IN и/или T3EUD). За выбор источника сигнала захвата, по которому происходит захват в регистр CAPREL, отвечает битовое поле CI регистра T5CON.

Максимально допустимая входная частота сигнала захвата равна $f_{CLK}/2$ (BPS2 = 01_B). Для корректного детектирования уровень входного сигнала захвата должен оставаться неизменным, как минимум, такт f_{CLK} (BPS2 = 01_B).

Каждый раз с приходом фронта сигнала тактирования на выбранный вход CAPIN или T3IN/T3EUD, согласно значению бита CT3, регистр CAPREL захватывает (если

T5SC = 1_B) содержимое таймера T5. Эти значения могут использоваться для измерения входного сигнала таймера T3. При захвате содержимого таймера T5 регистром CAPREL на линии прерывания CRIRQ появляется «1». Одновременно с этим таймер T5 может быть сброшен в 0000_H (если бит T5CLR = 1_B). В случае если бит T5SC = 0_B, захвата содержимого таймера не происходит, но все остальные операции могут выполняться.

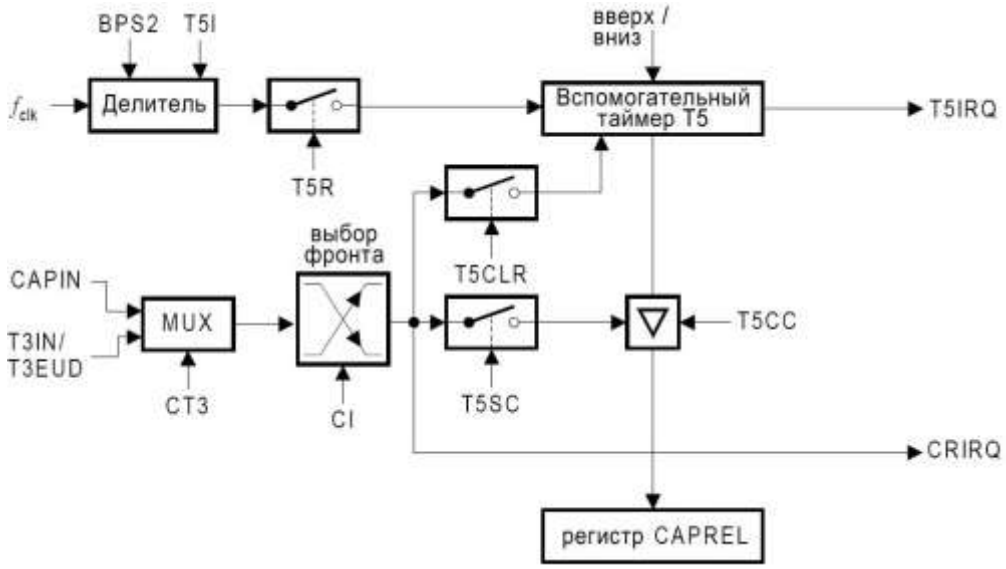


Рисунок 14.33 – Регистр CAPREL блока GPT2 в режиме захвата

Регистр захвата/перезагрузки CAPREL в режиме перезагрузки

Этот 16-битный регистр может использоваться как регистр перезагрузки для основного таймера T6. Этот режим выбирается установкой бита T6SR в регистре T6CON. Событием, вызывающим перезагрузку, является переполнение/опустошение таймера T6, см. рисунок 14.34.

Когда таймер T6 переполняется при счете «вверх» (переключается из $FFFF_H$ в 0000_H) или опустошается при счете «вниз» (переключается из 0000_H в $FFFF_H$), значение, хранящееся в регистре CAPREL, загружается в регистр таймера T6. При этом на линии CRIRQ, соответствующей регистру CAPREL, запрос на прерывание не формируется. В то же время на линии прерываний T6IRQ, соответствующей таймеру T6, выставляется «1», которая сигнализирует о переполнении/опустошении таймера T6.

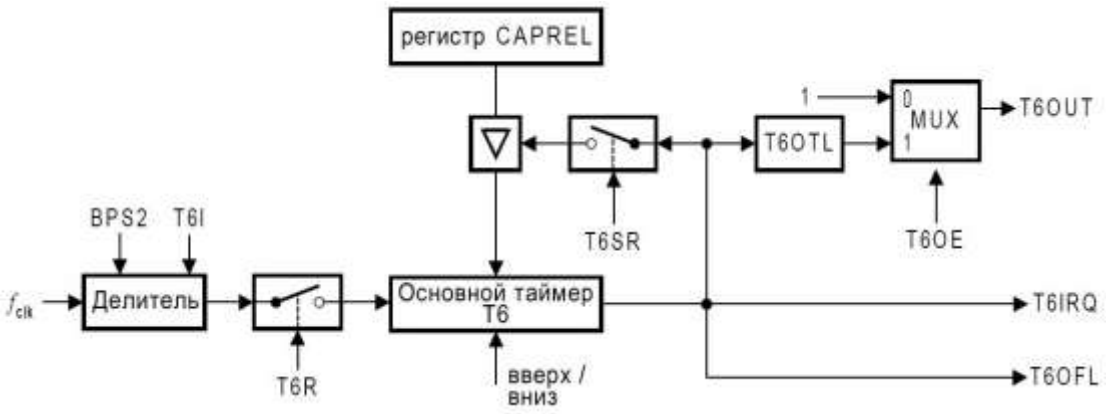


Рисунок 14.34 – Регистр CAPREL блока GPT2 в режиме перезагрузки

Регистр захвата/перезагрузки CAPREL в комбинированном режиме

Поскольку каждый из двух режимов (режим захвата и режим перезагрузки) регистра CAPREL может быть включен независимо от другого (биты T5SC и T6SR), имеется возможность одновременного включения обоих режимов. Это можно использовать для генерирования выходного сигнала, в желаемой зависимости от входного сигнала захвата.

Комбинированный режим может применяться для отслеживания высокочастотных последовательностей фронтов входных сигналов, которые могут приходить на входы апериодически. На рисунке 14.35 представлен регистр CAPREL блока GPT2 в комбинированном режиме.

Для измерений времени между фронтами таймер T5 и регистр CAPREL могут использоваться следующим образом.

Например, таймер T5 работает в режиме таймера и считает «вверх» с частотой $f_{CLK}/32$. Отслеживание фронтов входного сигнала происходит на линии CAPIN. Как только фронт обнаруживается, содержимое таймера T5 захватывается регистром CAPREL, после чего таймер T5 сбрасывается. Таким образом, в регистре CAPREL хранится значение промежутка времени между двумя фронтами сигнала, отсчитанного таймером T5.

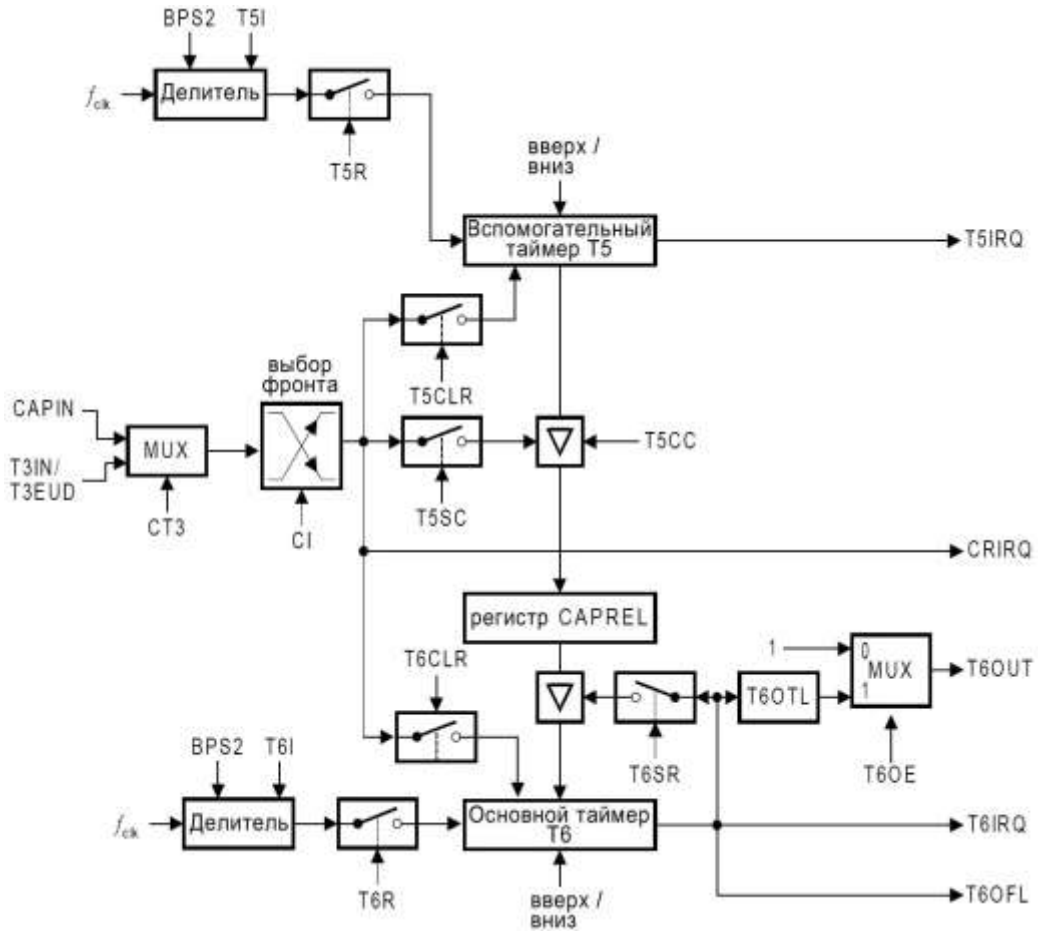


Рисунок 14.35 – Регистр CAPREL блока GPT2 в комбинированном режиме

Таймер T6, который работает в режиме счетчика и считает «вниз» с частотой $f_{CLK}/4$, использует значении регистра CAPREL для загрузки по опустошению. Отсюда следует, что значение в регистре CAPREL представляет собой промежуток времени между двумя опустошениями таймера T6, отсчитанный таймером T6. Поскольку таймер T6 считает в восемь раз быстрее таймера T5, он будет опустошаться восемь раз за время, равное минимально допустимому времени между появлениями двух фронтов сигнала, с которым

работает таймер T5. Поэтому сигнал опустошения таймера T6 генерирует восемь импульсов, и с каждым импульсом выставляется флаг T6IR и переключается бит T6OTL. Значение бита T6OTL может быть выведено на линию T6OUT. Этот сигнал в восемь раз больше фронтов, чем сигнал на линии CAPIN.

Некоторое отклонение выходной частоты возникает в результате того, что таймер T5 подсчитывает фактические временные единицы (например, таймер T5, работая на частоте 1 МГц, может захватить значение $64_H/100_D$ для входного сигнала частотой 10 кГц), в то время как T6OTL может переключаться только по опустошению таймера T6. В приведенном выше примере таймер T6 может считать «вниз» от 64_H, и опустошение произойдет через 101 такт сигнала синхронизации таймера T6. Действительная частота составит 79,2 кГц вместо ожидаемых 80 кГц.

Для корректировки частоты предусмотрен бит коррекции T5CC. Если T5CC = 1_B, содержимое таймера T5 декрементируется на 1 перед захватом. Такой способ позволяет устранить отклонение выходной частоты. После корректировки таймер T5 может захватить $63_H/99_D$ и выходная частота будет 80 кГц.

Примечание – Помимо прочего, сигнал опустошения таймера T6 может использоваться для тактирования таймеров других блоков таймеров посредством сигнала T6OFL.

14.4 Интерфейс блоков GPT1 и GPT2

Интерфейс блоков GPT1 и GPT2 представлен на рисунке 14.36.

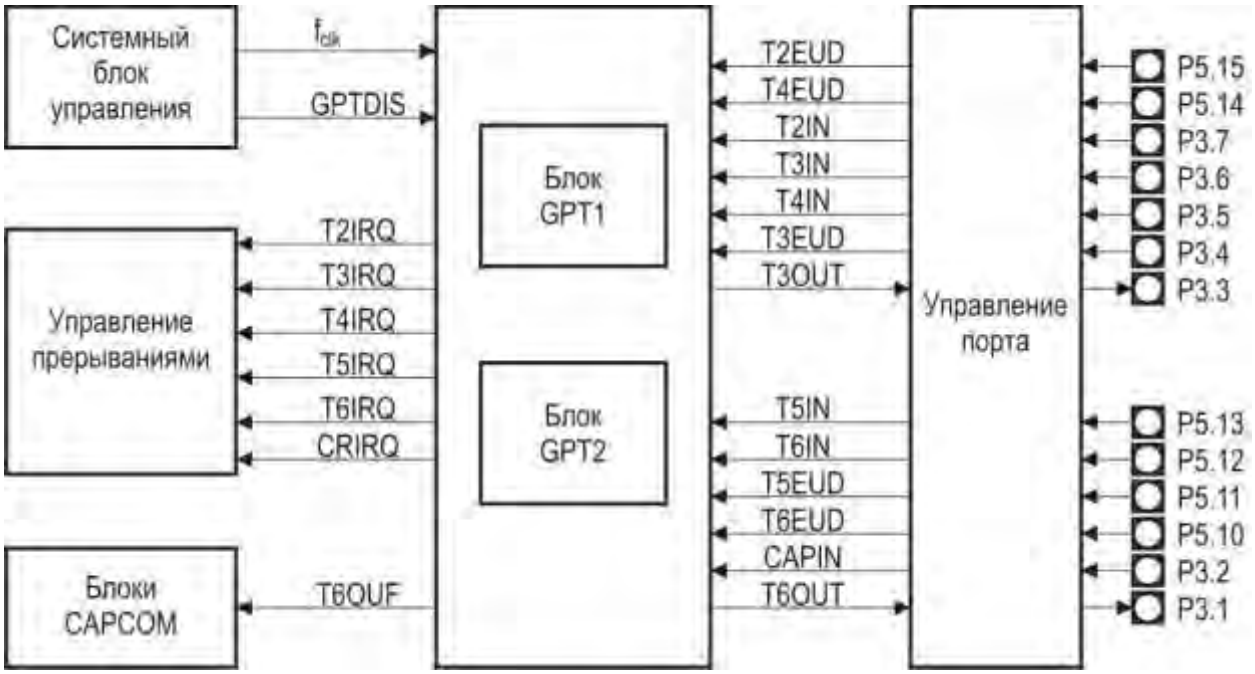


Рисунок 14.36 – Интерфейс блоков GPT1 и GPT2

15 Часы реального времени RTC

Часы реального времени RTC – это модуль, обеспечивающий контроллер информацией в масштабе реального времени. RTC имеет встроенные сигнальные функции, которые могут вызывать периодические обращения к системе. Модуль RTC имеет настраивающийся механизм, позволяющий программно подстраивать тактовый сигнал к частоте 1 Гц, компенсируя девиацию входной частоты, см. рисунки 15.1, 15.2.

RTC тактируется от генератора с частотой 32,768 кГц. Модуль обеспечивает формирование прерывания, при успешном сравнении запрограммированных значений регистров с текущим значением счетчика реального времени. Модуль RTC продолжает работать в то время как контроллер находится в режиме сохранения мощности.

Особенности:

- вход медленного тактового сигнала частотой 32,768 кГц;
- 16-разрядный делитель:
  - перезапускается после сравнения с запрограммированным значением;
  - формирует точный тактовый сигнал, используемый счетчиком реального времени;
- 32-разрядный основной счетчик реального времени:
  - обеспечивает подсчет времени до 136 лет при частоте входного тактового сигнала 1 Гц;
  - два регистра сравнения обеспечивают формирование прерываний при соответствии данных, содержащихся в них, с содержимым счетчика реального времени;
  - возможность формирования независимых прерываний при нахождении соответствия для каждого из сравниваемых регистров.

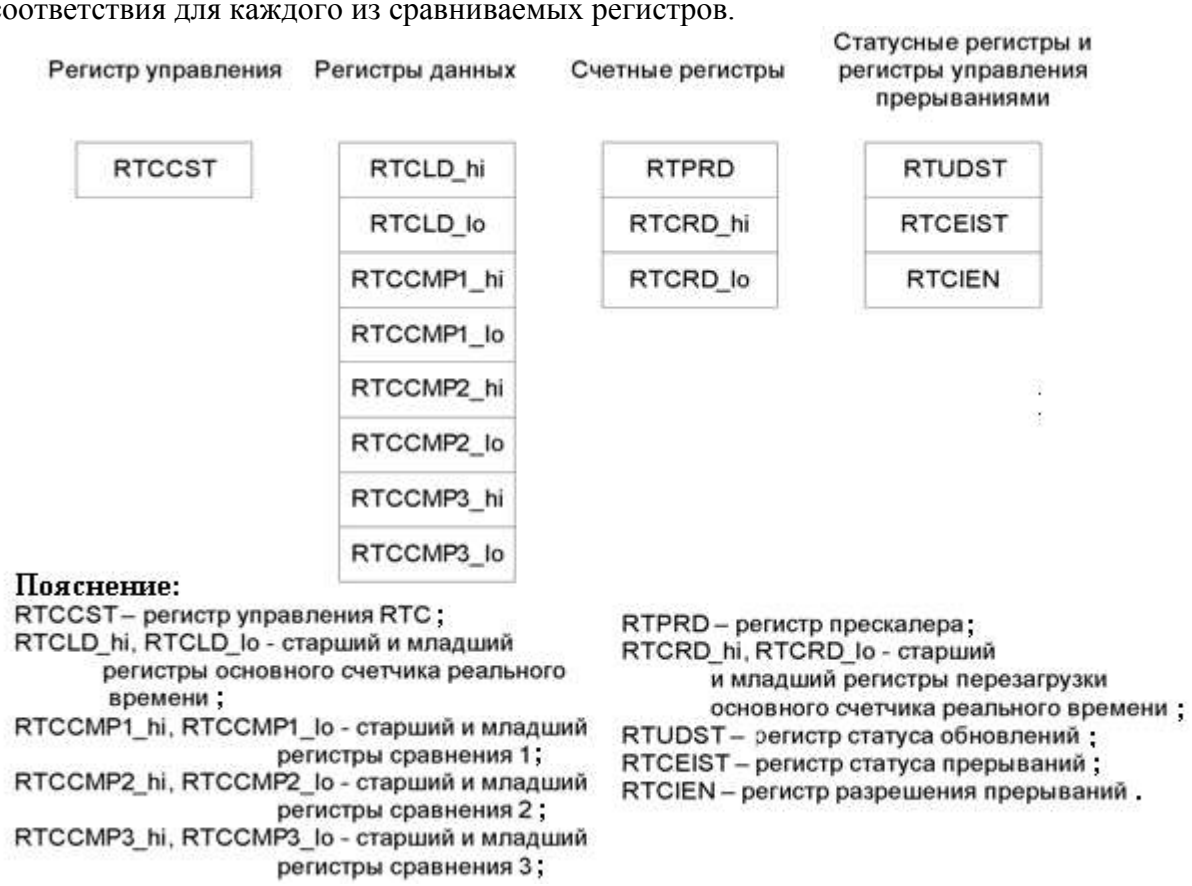


Рисунок 15.1 – Регистры модуля RTC

Некоторые 16-разрядные регистры образуют попарно 32-разрядные регистры:

- 16-разрядные RTCLD_hi, RTCLD_lo – 32-разрядный RTCLD;

- 16-разрядные RTCRD_hi, RTCRD_lo – 32-разрядный RTCRD;
- 16-разрядные RTCCMP1_hi, RTCCMP1_lo – 32-разрядный RTCCMP1;
- 16-разрядные RTCCMP2_hi, RTCCMP2_lo – 32-разрядный RTCCMP2;
- 16-разрядные RTCCMP3_hi, RTCCMP3_lo – 32-разрядный RTCCMP3.

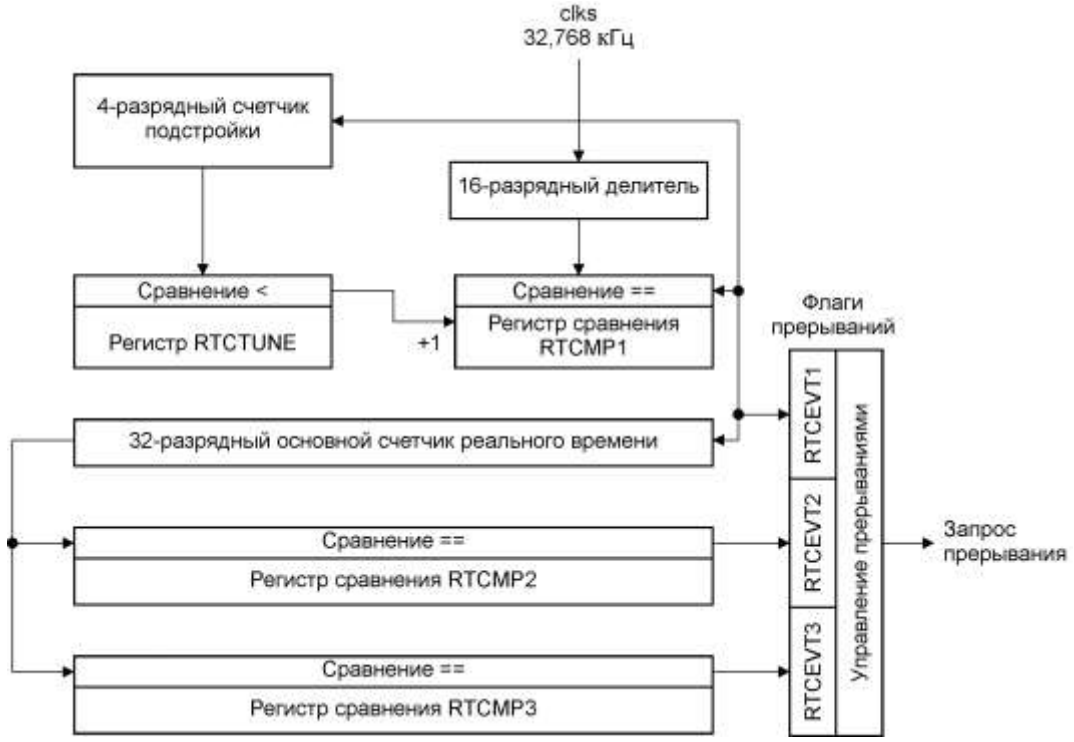


Рисунок 15.2 – Схема модуля RTC

15.1 Работа модуля RTC

Входной тактовый сигнал

Все действия RTC синхронизируются с медленным тактовым сигналом clks на входе. Рекомендуемая частота тактового сигнала clks 32,768 кГц. Входной тактовый сигнал поступает на делитель, который формирует точный тактовый сигнал частотой 1 Гц.

16-разрядный делитель и регистр сравнения RTCCMP1

16-разрядный делитель модуля RTC используется для формирования тактового сигнала для основного счетчика реального времени. Делитель считает входные тактовые импульсы, начиная с 0000_H, и как только его значение соответствует сравнивающемуся значению, находящемуся в регистре сравнения RTCCMP1, счетчик перезапускается заново с числа 0000_H. В момент перехода значения счетчика из 0000_H в 0001_H формируется тактовый импульс, поступающий на основной 32-разрядный счетчик реального времени. Базовое значение деления равно $7DFF_H$. К этому основному значению прибавляется значение смещения, задаваемое в регистре сравнения RTCCMP1. Таким образом, значение, при котором произойдет перезагрузка регистра делителя, вычисляется как RTCCMP1 (значение смещения) + $7DFF_H$ (базовое значение).

В схему встроена возможность задержки формирования тактового импульса для счетчика реального времени. Для этого используется счетчик точной подстройки. Выдача импульса при перезагрузке может подстраиваться в диапазоне между 0/16 и 15/16 тактов входного тактового сигнала с шагом 1/16 такта.

Перезагрузка делителя может быть инициирована программно в любое время установкой бита RTCCST.RTPRST в регистре управления. Запись «1» в этот бит перезапустит счетчик с 0000_H по следующему нарастающему фронту входного тактового сигнала clks. Текущее значение делителя может быть прочитано в любое время.

Регистр 16-разрядного делителя не затрагивается системным сбросом. Сброс делителя и регистра сравнения RTCCMP1 может быть осуществлен лишь с помощью power-on reset.

Счетчик подстройки RTC Tune

Счетчик точной подстройки – 4-разрядный счетчик обратного счета, используемый для более точной подстройки тактового сигнала на выходе 16-разрядного делителя (1 Гц). Счетчик синхронизирован выходным сигналом делителя. Текущее значение счетчика сравнивается со значением в битах RTCTUNE регистра сравнения RTCCMP1. Время установки подстроенного выходного сигнала делителя обеспечивается по истечении 16 тактов RTCOUT. Таким образом, возможности настройки позволяют осуществлять подстройку выходного тактового сигнала в диапазоне ±15 625 ppm с разрешающей способностью ±2 ppm.

RTC делитель и прерывание

При нахождении соответствия между сохраненным в регистре RTCCMP1 и текущим значением делителя устанавливается флаг RTCEIST.RTCEVT1, и если установлен бит RTCIEN.RTCIN1, формируется запрос на прерывание. В зависимости от параметров настроек делителя, прерывание может быть сформировано во время от 1,016 до 0,985 с при частоте входного тактового сигнала 32,768 кГц.

Основной счетчик реального времени

Основной счетчик реального времени представляет собой 32-разрядный счетчик с направлением счета «вверх», с максимальным значением времени 136 лет, при работе от опорного тактового сигнала (формируется делителем) частотой 1 Гц. Счетчик может быть запущен программно, установкой бита RTCCST.RTSTRT. Остановка счетчика возможна только лишь с помощью power-on reset.

Текущее значение счетчика реального времени может быть считано или записано программно в любое время. Однако операция записи синхронизируется с тактовым сигналом счетчика, поэтому новые данные будут записаны в регистр счетчика по следующему нарастающему фронту тактового импульса основного счетчика реального времени. Бит состояния RTCCST.RTUPD будет установлен при инициализации обновления содержимого счетчика, и остается установленным, пока загрузка нового значения в счетчик не будет завершена.

Прерывания

Кроме описанного выше прерывания, формируемого при перезапуске делителя, в схеме модуля RTC предусмотрены дополнительно два условия формирования прерывания.

Текущее значение основного счетчика реального времени непрерывно сравнивается с содержимым каждого из двух 32-разрядных регистров сравнения – RTCCMP2 и RTCCMP3. Как только текущее значение счетчика станет равно значению, сохраненному в любом из регистров, будет установлен соответствующий флаг RTCEIST.RTCEIST2 и RTCEIST.RTCEIST3, соответственно, и выставлен запрос на прерывание, если это разрешено битами RTCIEN.RTCIEN2 для регистра сравнения RTCCMP2 или RTCIEN.RTCIEN3 для регистра сравнения RTCCMP3.

Управление прерываниями

Модуль RTC формирует три различных, независимых прерывания. Кроме этого при выставлении запроса на любое из этих прерываний формируется еще одно, общее для всех, прерывание. Контроль над прерываниями осуществляется с помощью регистров управления прерываниями. В таблице 15.1 представлены типы прерываний и регистры управления соответствующими прерываниями.

Таблица 15.1 – Прерывания модуля RTC и регистры управления прерываниями

Источник прерывания	Регистр управления прерыванием
Совпадение значения регистра сравнения RTCCMP1 с текущим значением делителя	RTC_IC1 адрес – $0F150_H$
Совпадение значения регистра RTCCMP2 с текущим значением основного счетчика реального времени	RTC_IC2 адрес – $0F152_H$
Совпадение значения регистра RTCCMP3 с текущим значением основного счетчика реального времени	RTC_IC3 адрес – $0F154_H$
Формирование любого из прерываний по совпадению значений регистров RTCCMP1, RTCCMP2 или RTCCMP3	RTC_IC адрес – $0F13A_H$

Домен медленного тактового сигнала

16-разрядный делитель и счетчик реального времени работают непосредственно от медленного тактового сигнала, частотой 32,768 кГц, в то время как интерфейс шины, по которой происходит обмен информацией модуля RTC с контроллером, и регистры модуля работают от системного тактового генератора, имеющего более высокую частоту. Поэтому при чтении и записи некоторых регистров, возникает задержка обновления информации, связанная с синхронизацией схемы с двумя тактовыми сигналами.

Регистр состояния обновления RTUDST показывает текущее состояние задержки обновления регистров, тактируемых медленным тактовым сигналом.

При чтении данных из регистров, относящихся к домену медленного тактового сигнала, требуется, чтобы прошло четыре цикла системного тактового сигнала. Это относится к регистрам RTPRD и RTCRD.

Системный сброс reset и power-on reset

Системный сброс инициализирует только регистры, относящиеся к домену системного тактового сигнала. Регистры делителя RTPRD, основной счетчик реального времени RTCRD и регистры сравнения RTCCMP1, RTCCMP2, RTCCMP3 не затрагиваются системным сбросом. Перечисленные регистры инициализируются только power-on reset. Это позволяет продолжать выполнение операций хронометрирования во время системного сброса в обычном режиме. Однако операции обновления информации, связанные с синхронизацией между медленным тактовым сигналом (от генератора 32,768 МГц) и системным тактовым сигналом, отменяются системным сбросом.

Power-on reset останавливает все действия модуля RTC и инициализирует все регистры, устанавливая их в состояния, приведенные в таблице 15.12.

15.2 Регистры модуля RTC

Все регистры модуля RTC имеют разрядность 8, 16 или 32 бита. Однако каждый из 32-разрядных регистров сформирован из двух 16-разрядных, т. к. разрядность шины данных и размерность регистров, размещаемых в памяти контроллера, составляет не более 16 бит.

Регистр управления и статуса модуля RTCCST – 8-разрядный регистр, управляющий основными функциями модуля RTC и предоставляющий основную информацию о состоянии модуля, см. рисунок 15.3, таблицу 15.2.

RTCCST
регистр управления и статуса

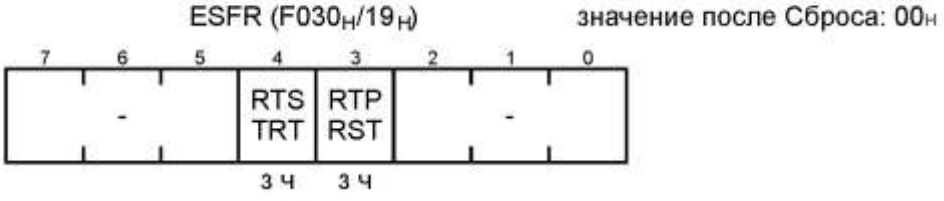


Рисунок 15.3 – Формат регистра RTCCST

Таблица 15.2 – Функциональное назначение полей регистра RTCCST

Поле	Биты	Тип	Описание
	2-0		Зарезервированы
RTPRST	3	Запись Чтение	Перезапуск делителя: 0 Обычные операции. 1 Сброс значения делителя. Когда бит установлен, делитель будет перезапущен по следующему нарастающему фронту входного тактового импульса. После операции перезагрузки бит очищается. Запись «0» игнорируется. Системный сброс очищает бит и завершает ожидание перезапуска делителя.
RTSTRT	4	Запись Чтение	Запуск счетчика реального времени: 0 Счетчик реального времени выключен. 1 Запуск счетчика реального времени. Установка бита запускает делитель и счетчик реального времени. После запуска остановка счетчика программным обеспечением становится невозможной. Установленный бит может быть очищен лишь с помощью power-on reset.
	7-5		Зарезервированы

Регистр состояния обновлений RTUDST – 8-разрядный регистр, предоставляющий информацию о состоянии обновлений содержимого регистров, расположенных в домене медленного тактового сигнала. Запись в этот регистр недоступна. Регистр очищается после системного сброса, см. рисунок 15.4, таблицу 15.3.

RTUDST
регистр состояния обновлений

ESFR ($F032_H/1A_H$) значение после Сброса: 00_H

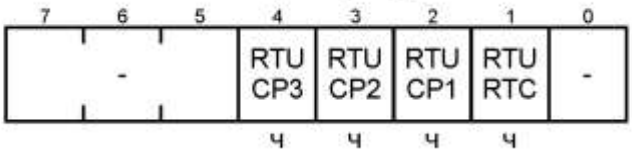


Рисунок 15.4 – Формат регистра RTUDST

Таблица 15.3 – Функциональное назначение полей регистра RTUDST

Поле	Биты	Тип	Описание
1	2	3	4
RTURTC	1	Чтение	Бит статуса процесса записи новых данных в регистр счетчика реального времени: 0 Отсутствует новое значение для загрузки. 1 Ожидание изменения. Установленный бит указывает, что счетчик реального времени загружается новым значением. Бит устанавливается при каждой записи в регистр загрузки счетчика реального времени RTCLD и очищается, как только новое значение было загружено в основной счетчик реального времени.

Окончание таблицы 15.3

1	2	3	4
RTUCP1	2	Чтение	Бит статуса процесса записи новых данных в регистр сравнения RTCCMP1: 0 Отсутствует новое значение для загрузки. 1 Ожидание изменения. Установленный бит указывает, что регистр сравнения RTCCMP1 загружается новым значением. Бит устанавливается при каждой записи в регистр сравнения RTCCMP1 и очищается сразу после записи нового значения в регистр.
RTUCP2	3	Чтение	Бит статуса процесса записи новых данных в регистр сравнения RTCCMP2: 0 Отсутствует новое значение для загрузки. 1 Ожидание изменения. Установленный бит указывает, что регистр сравнения RTCCMP2 загружается новым значением. Бит устанавливается при каждой записи в регистр сравнения RTCCMP2 и очищается сразу после записи нового значения в регистр.
RTUCP3	4	Чтение	Бит статуса процесса записи новых данных в регистр сравнения RTCCMP3: 0 Отсутствует новое значение для загрузки. 1 Ожидание изменения. Установленный бит указывает, что регистр сравнения RTCCMP3 загружается новым значением. Бит устанавливается при каждой записи в регистр сравнения RTCCMP3 и очищается сразу после записи нового значения в регистр.
	7-5, 0		Зарезервированы.

Регистр состояния прерываний RTCEIST – 8-разрядный регистр чтения/записи, предоставляющий статусную информацию состоянии всех прерываний модуля RTC, которые формируются при нахождении соответствия между содержимым регистра сравнения RTCCMP1 и делителем или содержимым регистров сравнения RTCCMP2 или RTCCMP3 и основным счетчиком реального времени. Все биты могут быть очищены путем записи в соответствующий бит «1». Системный сброс очищает биты регистра, см. рисунок 15.5, таблицу 15.4.

RTCEIST
регистр состояния прерываний

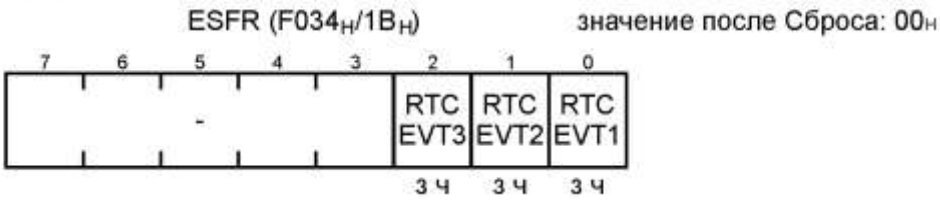


Рисунок 15.5 – Формат регистра RTCEIST

Таблица 15.4 – Функциональное назначение полей регистра RTCEIST

Поле	Биты	Тип	Описание
RTCEVT1	0	Запись Чтение	Флаг соответствия содержимого регистра и делителя: 0 Соответствия не произошло. 1 Выявлено соответствие. Бит устанавливается в случае нахождения соответствия между значением, сохраненным в регистре сравнения RTCCMP1, и текущим значением делителя.
RTCEVT2	1	Запись Чтение	Флаг соответствия содержимого регистра и счетчика реального времени: 0 Соответствия не произошло. 1 Выявлено соответствие. Бит устанавливается в случае нахождения соответствия между значением, сохраненным в регистре сравнения RTCCMP2, и текущим значением основного счетчика реального времени.
RTCEVT3	2	Запись Чтение	Флаг соответствия содержимого регистра и счетчика реального времени: 0 Соответствия не произошло. 1 Выявлено соответствие. Бит устанавливается в случае нахождения соответствия между значением, сохраненным в регистре сравнения RTCCMP3, и текущим значением основного счетчика реального времени.
	7-3		Зарезервированы.

Регистр управления прерываниями RTCIEN – 8-разрядный регистр чтения/записи, разрешающий формирование прерывания при установке флагов прерываний в статусном регистре RTCEIST, см. рисунок 15.6, таблицу 15.5. Системный сброс очищает биты регистра.

RTCIEN
регистр управления прерываниями

ESFR ($F036_H/1C_H$) значение после Сброса: 00_H

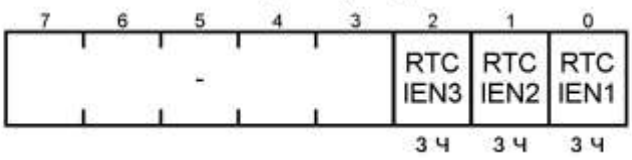


Рисунок 15.6 – Формат регистра RTCIEN

Таблица 15.5 – Функциональное назначение полей регистра RTCIEN

Поле	Биты	Тип	Описание
RTCIEN1	0	Запись Чтение	Бит разрешения прерывания: 0 Прерывание запрещено. 1 Прерывание разрешено. Если бит установлен, прерывание будет сгенерировано, как только бит RTCEVT1 будет установлен (при нахождении соответствия между значением, сохраненным в регистре сравнения RTCCMP1, и значением делителя).
RTCIEN2	1	Запись Чтение	Бит разрешения прерывания: 0 Прерывание запрещено. 1 Прерывание разрешено. Если бит установлен, прерывание будет сгенерировано, как только бит RTCEVT2 будет установлен (при нахождении соответствия между значением, сохраненным в регистре сравнения RTCCMP2, и значением основного счетчика реального времени).
RTCIEN3	2	Запись Чтение	Бит разрешения прерывания: 0 Прерывание запрещено. 1 Прерывание разрешено. Если бит установлен, прерывание будет сгенерировано, как только бит RTCEVT3 будет установлен (при нахождении соответствия между значением, сохраненным в регистре сравнения RTCCMP3, и значением счетчика реального времени).
	7-3		Зарезервированы

Регистр делителя RTPRD – 16-разрядный регистр, возвращает текущее значение делителя, см. рисунок 15.7, таблицу 15.6.

Регистр является только читаемым, запись в этот регистр неэффективна. Системный сброс не затрагивает содержимое регистра. После power-on reset содержимое регистра принимает значение 0000_H.

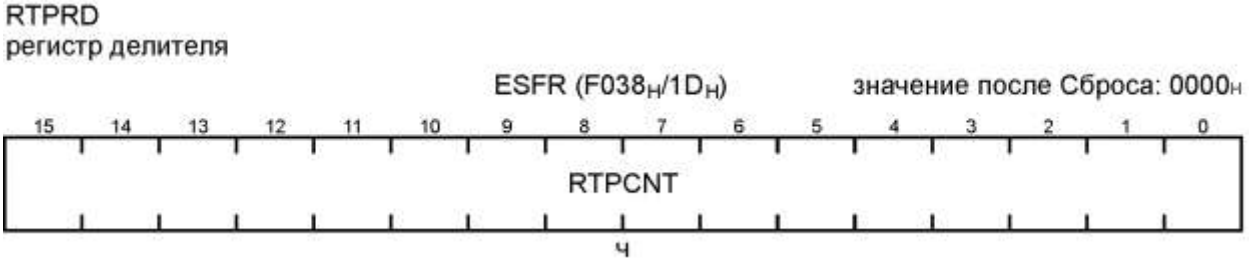


Рисунок 15.7 – Формат регистра RTPRD

Таблица 15.6 – Функциональное назначение полей регистра RTPRD

Поле	Биты	Тип	Описание
RTPCNT	15-0	Чтение	Биты содержат значение делителя. Запись в эти биты неэффективна. Чтение возвращает текущее значение делителя.

Регистр счетчика реального времени RTCRD

Регистр RTCRD – 32-разрядный регистр, возвращает текущее значение счетчика реального времени, см. рисунок 15.8, таблицу 15.7.

Регистр является только читаемым, запись в этот регистр не возможна. Системный сброс не влияет на содержимое регистра. После power-on reset содержимое регистра принимает значение 00000001_H.

32-разрядный регистр RTCRD построен на основе двух 16-разрядных регистров RTCRD_hi (старшие разряды регистра RTCRD) и RTCRD_lo (младшие разряды регистра RTCRD).

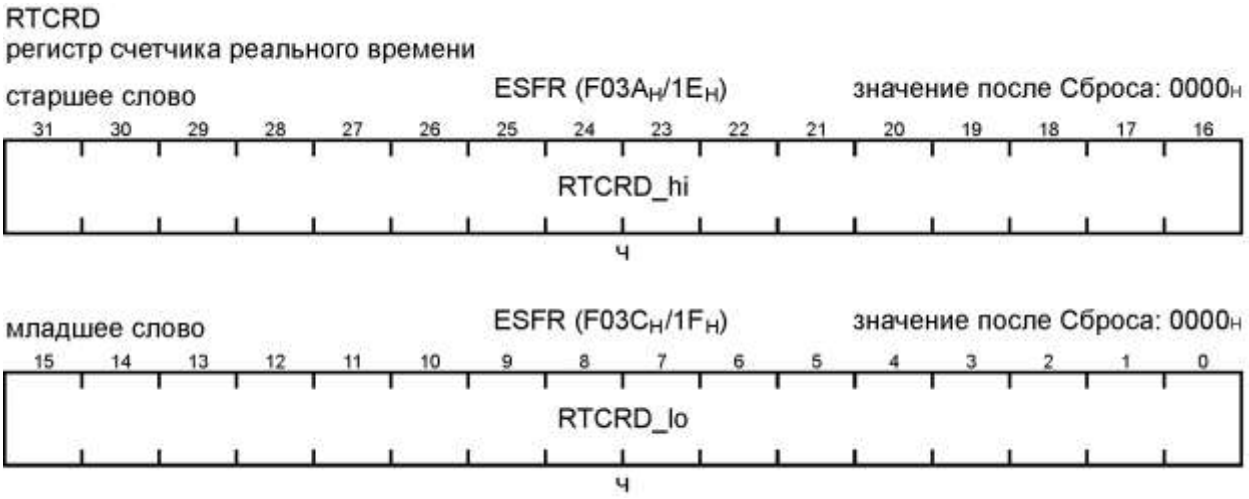


Рисунок 15.8 – Формат регистра RTCRD

Таблица 15.7 – Функциональное назначение полей регистра RTCRD

Поле	Биты	Тип	Описание
RTCRD_lo	15-0	Чтение	Младшие биты регистра RTCRD. Чтение возвращает текущее значение соответствующих разрядов счетчика.
RTCRD_hi	32-16	Чтение	Старшие биты регистра RTCRD. Чтение возвращает текущее значение соответствующих битов счетчика.

Регистр загрузки RTCLD

Регистр загрузки RTCLD – 32-разрядный регистр чтения/записи. Запись в этот регистр вызывает загрузку основного счетчика реального времени записанным значением. Чтение возвращает последнее записанное значение. Обновление содержимого основного счетчика реального времени требует синхронизации между медленным и системным тактовыми сигналами. Флаг RTCCST.RTURTC будет установлен во время записи и очищен после завершения обновления содержимого регистра счетчика реального времени. После записи данных в регистр RTCLD следующая запись в этот регистр игнорируется до тех пор, пока не будет обновлен основной счетчик реального времени текущим содержимым регистра RTCLD. Регистр очищается после системного сброса, при этом ожидание загрузки счетчика реального времени текущим содержимым регистра RTCLD сбрасывается.

32-разрядный регистр RTCLD построен на основе двух 16-разрядных регистров RTCLD_hi (старшие разряды регистра RTCLD) и RTCLD_lo (младшие разряды регистра RTCLD). Каждый из двух регистров может быть записан независимо, см. рисунок 15.9, таблицу 15.8.

.

Примечание – При программировании записи в регистр RTCLD, между записью в младший регистр RTCLD_lo и записью в старший – RTCLD_hi (независимо от последовательности) следует делать промежуток равный времени выполнения не менее 150 команд NOP.

Рисунок 15.9 – Формат регистра RTCLD

Таблица 15.8 – Функциональное назначение полей регистра RTCLD

Поле	Биты	Тип	Описание
RTCLD_lo	15-0	Запись Чтение	Младшие биты регистра RTCLD. Запись в эти биты необходимо производить перед записью старших битов. Чтение возвращает последнее записанное значение.
RTCLD_hi	32-16	Запись Чтение	Старшие биты регистра RTCLD. Запись в эти биты должна производиться после записи в RTCLD_lo. Чтение возвращает последнее записанное значение.

Регистр сравнения RTCCMP1 – 32-разрядный регистр чтения/записи содержит значение, сравниваемое с содержимым делителя, и текущее значение точной подстройки частоты на выходе делителя. Значение регистра не затрагивается системным сбросом. Начало записи в этот регистр устанавливает бит RTUDST.RTUCP1, который остается установленным, пока регистр RTCCMP1 не будет обновлен. После power-on reset регистр принимает значение 00000200$_H$.

32-разрядный регистр RTCCMP1 построен на основе двух 16-разрядных регистров RTCCMP1_hi (старшие разряды регистра RTCCMP1) и RTCMP1_lo (младшие разряды регистра RTCMP1), см. рисунок 15.10, таблицу 15.9.

.

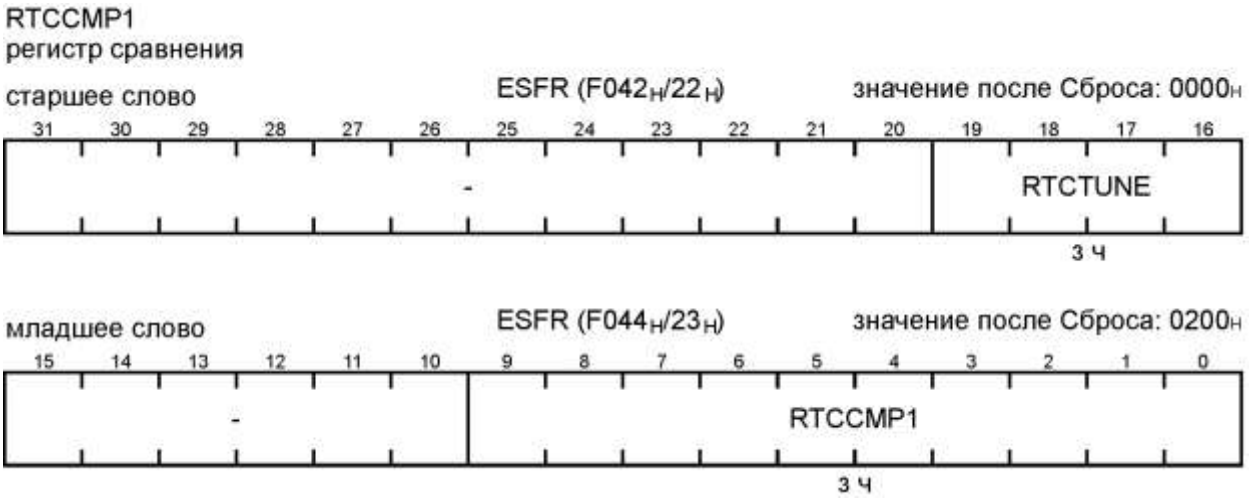


Рисунок 15.10 – Формат регистра RTCCMP1

Таблица 15.9 – Функциональное назначение полей регистра RTCCMP1

Поле	Биты	Тип	Описание
RTCCMP1	9-0	Запись Чтение	Младшие биты регистра. Поле содержит текущее значение смещения для делителя, по достижении которого делитель перезапускается. Смещение добавляется к основному значению $7DFF_H$. Биты должны быть записаны перед записью RTCCMP1_hi. Чтение возвращает последнее записанное значение.
RTCTUNE	19-16	Запись Чтение	Значение подстройки. Старшие биты регистра. Запись в эти биты должна производиться после записи в RTCCMP1_lo.
–	15-10, 31-19	–	Зарезервированы

Регистр сравнения RTCCMP2 – 32-разрядный регистр чтения/записи содержит значение, сравниваемое с содержимым счетчика реального времени. Значение регистра не затрагивается системным сбросом. Начало записи в этот регистр устанавливает бит RTUDST.RTUCP2, который остается установленным, пока регистр RTCCMP2 не будет обновлен. Содержимое регистра не затрагивается системным сбросом. После power-on reset регистр принимает значение $FFFFFFFF_H$.

Во время работы содержимое регистра RTCCMP2 постоянно сравнивается со значением основного счетчика реального времени. При совпадении значений устанавливается флаг – бит RTCEIST.RTCEVT2 и, если установлен бит RTCIEN.RTCIEN2, вырабатывается запрос на прерывание, см. рисунок 15.11, таблицу 15.10.

32-разрядный регистр RTCCMP2 построен на основе двух 16-разрядных регистров RTCCMP2_hi (старшие разряды регистра RTCCMP2) и RTCMP2_lo (младшие разряды регистра RTCMP2).

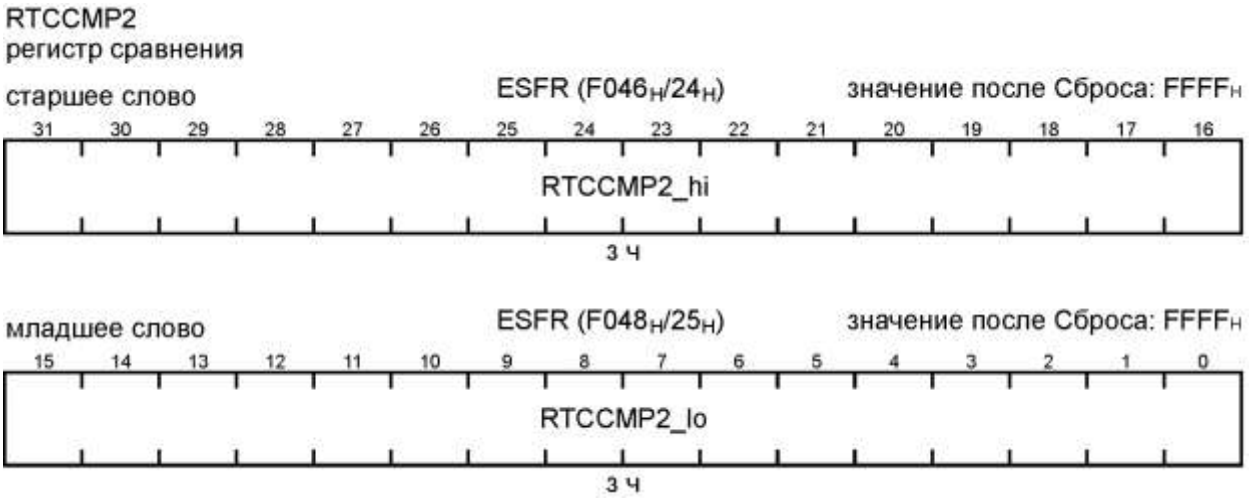


Рисунок 15.11 – Формат регистра RTCCMP2

Таблица 15.10 – Функциональное назначение полей регистра RTCCMP2

Поле	Биты	Тип	Описание
RTCCMP2_lo	15-0	Запись Чтение	Младшие биты регистра RTCCMP2. Запись в эти биты необходимо производить перед записью старших битов. Чтение возвращает последнее записанное значение.
RTCCMP2_hi	32-16	Запись Чтение	Старшие биты регистра RTCCMP2. Запись в эти биты должна производиться после записи в RTCCMP2_lo. Чтение возвращает последнее записанное значение.

Регистр сравнения RTCCMP3 – 32-разрядный регистр чтения/записи содержит значение, сравниваемое с содержимым счетчика реального времени. Значение регистра не затрагивается системным сбросом. Начало записи в этот регистр устанавливает бит RTUDST.RTUCP3, который остается установленным, пока регистр RTCCMP3 не будет обновлен. Содержимое регистра не затрагивается системным сбросом. После power-on reset регистр принимает значение $FFFFFFFF_H$.

Во время работы содержимое регистра RTCCMP3 постоянно сравнивается со значением основного счетчика реального времени. При совпадении значений устанавливается флаг – бит RTCEIST.RTCEVT3 и, если установлен бит RTCIEN.RTCIEN3, вырабатывается запрос на прерывание.

32-разрядный регистр RTCCMP3 построен на основе двух 16-разрядных регистров RTCCMP3_hi (старшие разряды регистра RTCCMP3) и RTCCMP3_lo (младшие разряды регистра RTCCMP3), формат регистра RTCCMP3 представлен на рисунке 15.12, функциональное назначение полей регистра RTCCMP3 – в таблице 15.11.

.

.

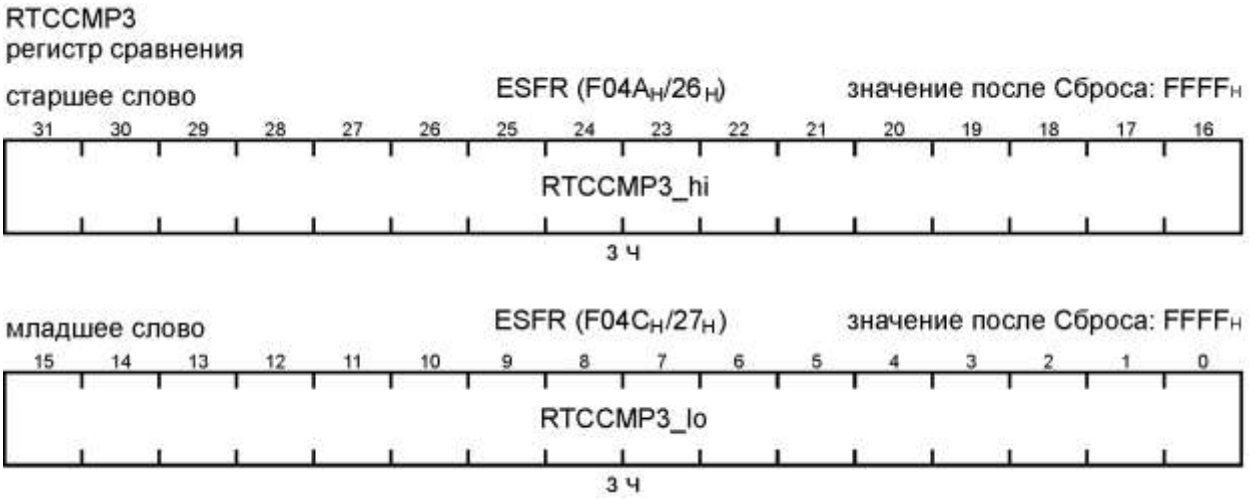


Рисунок 15.12 – Формат регистра RTCCMP3

Таблица 15.11 – Функциональное назначение полей регистра RTCCMP3

Поле	Биты	Тип	Описание
RTCCMP3_lo	15-0	Запись Чтение	Младшие биты регистра RTCCMP3. Запись в эти биты необходимо производить перед записью старшего байта. Чтение возвращает последнее записанное значение.
RTCCMP3_hi	32-16	Запись Чтение	Старшие биты регистра RTCCMP3. Запись в эти биты должна производиться после записи в RTCCMP3_lo. Чтение возвращает последнее записанное значение.

В таблице 15.12 представлены типы сигналов сброса, по которым происходит инициализация регистров RTC.

Таблица 15.12 – Инициализация регистров RTC

Название регистра	Тип инициализации
RTCCST	reset
RTUDST	reset
RTCEIST	reset
RTCIEN	reset
RTPRD	power-on reset
RTCRD_hi	power-on reset
RTCRD_lo	power-on reset
RTCLD_hi	reset
RTCLD_lo	reset
RTCCMP1_hi	power-on reset
RTCCMP1_lo	power-on reset
RTCCMP2_hi	power-on reset
RTCCMP2_lo	power-on reset
RTCCMP3_hi	power-on reset
RTCCMP3_lo	power-on reset

16 Модуль АЦП

В состав контроллера входят два 8/10-разрядных АЦП и схема выборки и хранения. Мультиплексор выбирает между 16 аналоговыми входными каналами (альтернативные функции порта P5) программно (фиксированный канал) или автоматически (режим автоматического сканирования). Есть два режима работы модуля: одиночный АЦП и двойной АЦП, которые можно выбрать программным способом с помощью регистра выбора режима ADC_SWITCH, см. рисунок 16.1, таблицу 16.1.

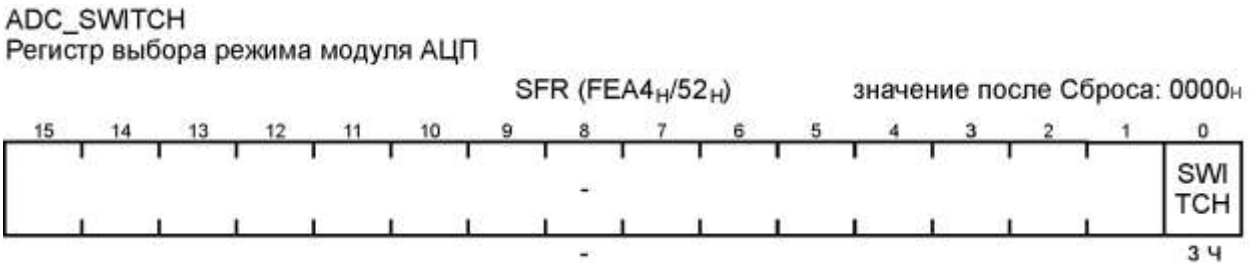


Рисунок 16.1 – Формат регистра ADC_SWITCH

Таблица 16.1 – Функциональное назначение полей регистра ADC_SWITCH

Поле	Биты	Тип	Описание
SWITCH	0	Чтение Запись	Выбор режима работы АЦП. Режим одиночного АЦП. Режим двойного АЦП.

16.1 Режим одиночного АЦП

По умолчанию бит SWITCH регистра ADC_SWITCH сброшен (SWITCH = 0_B) и модуль АЦП работает в одиночном режиме, см. рисунки 16.2, 16.3.

Рассмотрим этот режим подробнее.

Для выполнения большинства требований АЦП поддерживает следующие режимы преобразования:

- Фиксированный канал однократного преобразования. Производит только одно преобразование для выбранного канала.
- Фиксированный канал неоднократного преобразования. Производит неоднократное преобразование выбранного канала.
- Автоматическое сканирование однократного преобразования. Производит только одно преобразование каналов из выбранной группы.
- Автоматическое сканирование неоднократного преобразования. Производит неоднократное преобразование каналов из выбранной группы.
- Режим ожидания чтения ADC_DAT. Автоматическое начало преобразования после того, как предыдущий результат был прочитан.
- Режим инжекции канала. Вставка канала в группу каналов (автоматическое сканирование).

Набор SFR регистров и входных каналов обеспечивает доступ к управлению функциями АЦП и результату преобразования.

Рисунок 16.2 – Регистры и выводы портов в режиме одиночного АЦП

Выводы ∩VCC3 и ∩0V3 являются внешним питанием АЦП. Отдельное (независимое) питание АЦП уменьшает влияние помех от других цифровых сигналов. Опорное напряжение должно быть стабильным во время сброса (± 80 мВ для 8-разрядного преобразования АЦП, ± 20 мВ для 10-разрядного преобразования) и во время любого преобразования (± 10 мВ для 8-разрядного преобразования, ± 2,5 мВ для 10-разрядного преобразования) для достижения максимума точности.

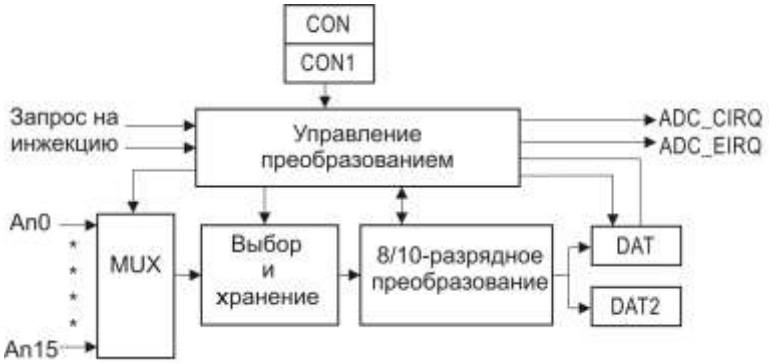


Рисунок 16.3 – Блок-схема АЦП в одиночном режиме

16.2 Выбор режима

Аналоговые входы AN15, …, AN0 являются альтернативными функциями порта P5, который рассчитан только на ввод. Линии порта P5 могут использоваться как аналоговые, так и цифровые входы.

Функциями АЦП управляют два побитно адресуемых регистра управления ADC_CON, см. рисунок 16.4, таблицу 16.2, и ADC_CON1, см. рисунок 16.5, таблицу 16.3. Их биты определяют аналоговый канал (вход), который будет обрабатываться, режим преобразования и также отражают состояния преобразования.

ADC_CON
регистр управления АЦП

SFR ($FFA0_H/D0_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCTC		-		AD CRQ	AD CIN	AD WR	AD BSY	AD ST	-	ADM		ADCH			
з ч		-		з ч ап	з ч	з ч	з ч ап	з ч ап	-	з ч		з ч			

Рисунок 16.4 – Формат регистра ADC_CON

Таблица 16.2 – Функциональное назначение полей регистра ADC_CON

Поле	Биты	Тип	Описание
ADCTC	15-14	Чтение Запись	Управление временем преобразования АЦП (определяет основную частоту преобразования $f_П$): 00 $f_П = f_{ADC}/8$ 01 $f_П = f_{ADC}/4$ 10 $f_П = f_{ADC}/32$ 11 $f_П = f_{ADC}/16$
ADCRQ	11	Чтение Запись Аппаратное влияние	Флаг запроса на инжекцию канала АЦП.
ADCIN	10	Чтение Запись	Разрешение инжекции канала АЦП.
ADWR	9	Чтение Запись	Ожидания управления чтения АЦП.
ADBSY	8	Чтение Аппаратное влияние	Флаг занятости АЦП: 0 АЦП находится в состоянии простоя. 1 АЦП находится в активном состоянии.
ADST	7	Чтение Запись Аппаратное влияние	Стартовый бит АЦП: 0 Остановить начатое преобразование. 1 Начать преобразование.
ADM	5-4	Чтение Запись	Выбор режима работы АЦП: 00 Фиксированный канал однократное преобразование. 01 Фиксированный канал неоднократное преобразование. 10 Автоматическое сканирование однократное преобразование. 11 Автоматическое сканирование неоднократное преобразование.
ADCH	3-0	Чтение Запись	Выбор аналогового канала АЦП.
Примечание – В режиме двойного АЦП использовать три младших бита битового поля ADCH, старший бит этого поля на выбор каналов влиять не будет.			

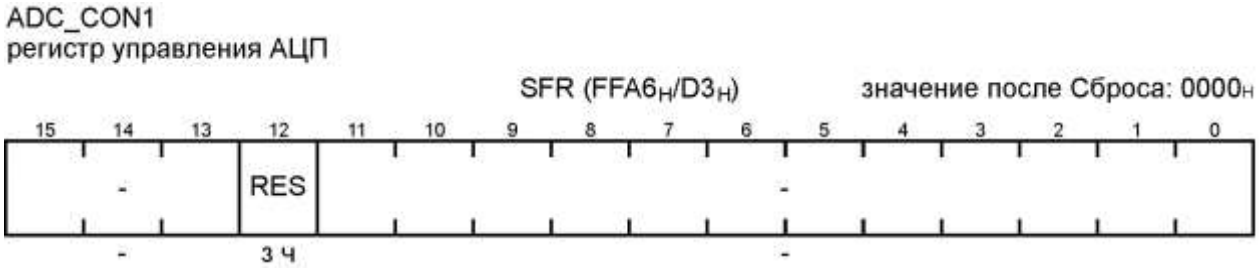


Рисунок 16.5 – Формат регистра ADC_CON1

Таблица 16.3 – Функциональное назначение поля регистра ADC_CON1

Поле	Биты	Тип	Описание
RES	12	Чтение Запись	Бит управления точностью (разрешающей способностью) преобразования: 0 10-битное преобразование (устанавливается по сбросу). 1 8-битное преобразование.

16.3 Работа АЦП

Выбор канала для преобразования

Битовое поле ADCH управляет входной логикой каналов мультиплексора. В однократных режимах оно определяет входной канал, который должен быть преобразован. В режимах автоматического сканирования оно определяет номер самого старшего канала, который будет преобразован в цикле автоматического сканирования.

ADCH может быть изменено во время преобразования. Новое значение будет использоваться после завершения преобразования в режимах фиксированного канала, или после завершения цикла преобразований в режиме автоматического сканирования.

Флаг ADBSY

Флаг занятости АЦП ADBSY устанавливается, если преобразование началось, т. е. установлен бит ADST, и остается установленным, пока преобразование происходит. Бит ADBSY очищен тогда, когда АЦП находится в состоянии простоя, это означает, что преобразование не происходит.

Управление стартом и остановкой АЦП

Бит ADST используется для запуска или остановки АЦП. Однократное преобразование или последовательность преобразований начинается после установки бита ADST. После установки флага занятости ADBSY преобразователь выбирает канал, который указан в битовом поле ADCH. Напряжение на входе канала будет захвачено внутренними средствами во время преобразования. Когда преобразование канала завершилось, результат вместе с номером преобразованного канала заносится в регистр результата, и генерируется запрос на прерывание. Результат преобразования расположен в битовом поле ADRES.

Бит ADST остается установленным, пока не будет сброшен программно или аппаратно. Аппаратная очистка зависит от режимов преобразования:

- В режиме однократного преобразования фиксированного канала бит ADST очищен после завершения преобразования указанного канала.

- В режиме однократного преобразования автоматического сканирования бит ADST очищен после завершения преобразования нулевого канала.

Примечание – В режимах непрерывного преобразования бит ADST аппаратно никогда не очищается.

Программная остановка АЦП выполняется при очистке бита ADST. Реакция АЦП зависит от режима преобразования.

В режиме однократного преобразования фиксированного канала АЦП завершает преобразование и затем останавливается. На работу АЦП не повлияет, если программно не очистить бит ADST.

В режиме неоднократного преобразования фиксированного канала АЦП завершает текущее преобразование и затем останавливается. Это обычный способ завершения преобразования.

В режиме однократного преобразования автоматического сканирования АЦП завершает преобразование текущего канала и затем остановится. На работу АЦП не повлияет, если программно не очистить бит ADST.

В режиме неоднократного преобразования автоматического сканирования АЦП завершает преобразование текущего канала и затем останавливается. Это обычный способ завершения преобразования.

Рестарт АЦП может быть выполнен, если очистить бит ADST, а затем его установить. Эта последовательность действий прервет текущее преобразование и перезапустит АЦП с новыми значениями регистров управления.

Выбор режима преобразования

Битовое поле ADM, смотри таблицу 16.2 с функциональным назначением полей регистра ADC_CON, выбирает режим преобразования АЦП, см. таблицу 16.4.

Таблица 16.4 – Режимы преобразования АЦП

ADM	Описание
00_B	Фиксированный канал однократное преобразование
01_B	Фиксированный канал неоднократное преобразование
10_B	Автоматическое сканирование однократное преобразование
11_B	Автоматическое сканирование неоднократное преобразование

Поле выбора режима ADM и поле выбора канала ADCH возможно изменить во время преобразования. Поле ADM будет использоваться после завершения текущего преобразования. Поле ADCH будет использоваться после завершения текущего преобразования (режимы фиксированных каналов) или после завершения текущей последовательности преобразований (режимы автоматического сканирования).

Управление точностью преобразования

АЦП может производить 10-битное преобразование ($RES = 0_B$) или 8-битное ($RES = 1_B$). В зависимости от требований выбирается менее точное преобразование или более точное преобразование в регистре ADC_CON1.

Результат преобразования

Результат преобразования хранится в регистре результата ADC_DAT, см. рисунок 16.6, или в регистре ADC_DAT2, см. рисунок 16.7, для инжектированного преобразования. Расположение результата зависит от выбранной точности (8 битное или 10-битное преобразование).

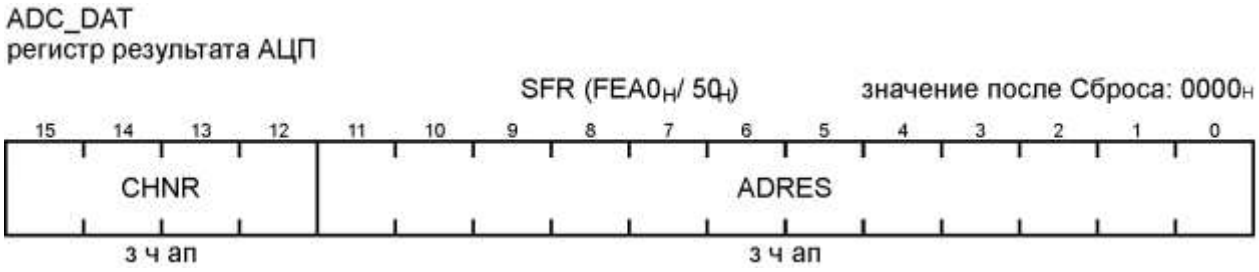


Рисунок 16.6 – Формат регистра ADC_DAT

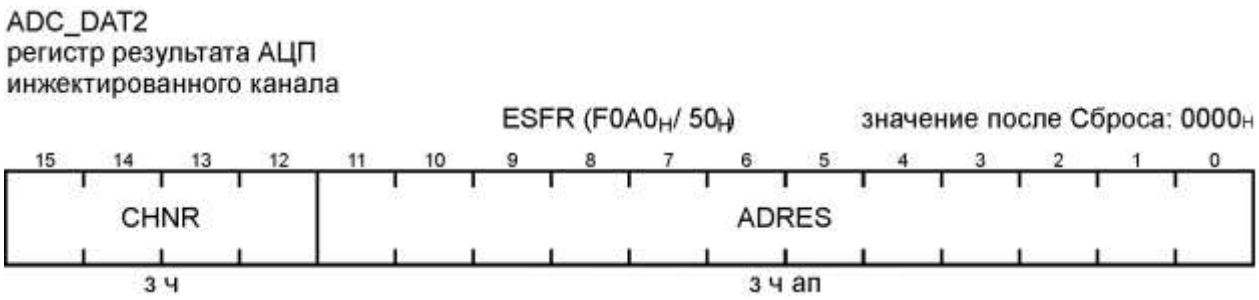


Рисунок 16.7 – Формат регистра ADC_DAT2

Функциональное назначение полей регистров ADC_DAT, ADC_DAT2 представлено в таблице 16.5.

Таблица 16.5 – Функциональное назначение полей регистров ADC_DAT, ADC_DAT2

Поле	Биты	Тип	Описание
CHNR	15-12	Чтение Запись Аппаратное влияние	Номер канала (идентифицирует преобразованный аналоговый канал)
ADRES	11-0	Чтение Запись Аппаратное влияние	Результат преобразования АЦП. Цифровой результат самого последнего преобразования. Результат расположен следующим образом: 8-битное преобразование: ADRES.9 – ADRES.2, 10-битное преобразование: ADRES.9 – ADRES.0. Примечание – Неиспользованные биты ADRES всегда 0.

Режимы преобразования фиксированных каналов

Этот режим можно выбрать программно, путем установки «00» (однократное преобразование) или «01» (неоднократное преобразование) в битовом поле ADM. Преобразование начинается после установки «1» в битовом поле ADST, затем выставляется флаг занятости ADBSY и начинается преобразование определенного канала, который указан в битовом поле ADCH. После завершения преобразования будет выставлен флаг запроса на прерывание ADCIR.

В режиме однократного преобразования преобразователь автоматически останавливается по окончанию преобразования и сбрасывает биты ADBSY и ADCIR.

В режиме неоднократного преобразования каждое новое преобразование начинается автоматически для канала, указанного в битовом поле ADCH. Флаг ADCIR будет установлен после каждого завершенного преобразования. В случае программной установки «0» в бите ADST во время преобразования, преобразователь завершит текущее преобразование, затем преобразователь будет остановлен и бит ADBSY будет сброшен.

Режимы преобразования с автоматическим сканированием

Эти режимы можно выбрать программно, путем установки «10» (однократное преобразование) или «11» (неоднократное преобразование) в битовом поле ADM. В режиме автоматического сканирования преобразователь без каких-либо программных изменений номера канала автоматически последовательно преобразует сигнал на нескольких входных каналах, начиная с канала, указанного в битовом поле ADCH и заканчивая нулевым каналом. Преобразование начинается после установки «1» в бите ADST. После этого выставляется флаг занятости ADBSY и производится преобразование канала, указанного в битовом поле ADCH. После завершения преобразования выставляется флаг запроса на прерывание, и автоматически начинается следующее преобразование для следующего канала, номер которого на единицу меньше. После завершения каждого преобразования выставляется флаг запроса на прерывание ADCIR. После преобразования нулевого канала текущая последовательность считается законченной.

В режиме однократного преобразования преобразователь автоматически останавливается и сбрасывает биты ADBSY и ADST.

В режиме неоднократного преобразования преобразователь автоматически начинает новую последовательность с канала, указанного в битовом поле ADCH.

В случае программного сброса значения бита ADST во время совершения преобразования, преобразователь будет остановлен и будет сброшено значение бита ADBSY после завершения текущей последовательности преобразований.

Пример работы в режиме автоматического сканирования из регистра ADC_DAT представлен на рисунке 16.8.

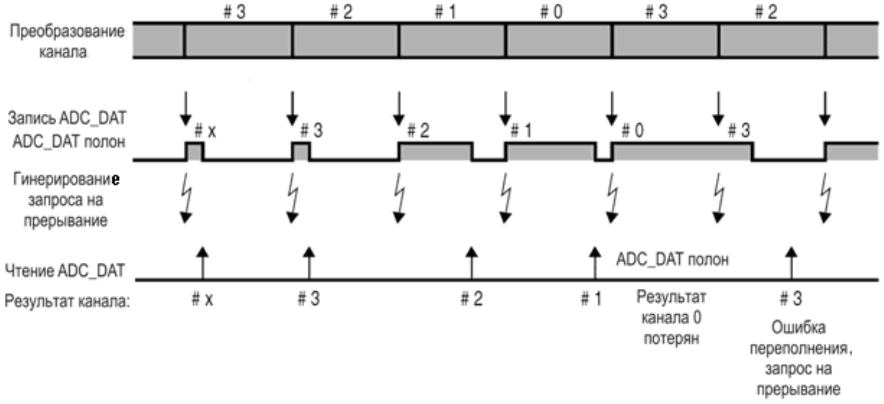


Рисунок 16.8 – Пример работы в режиме автоматического сканирования из регистра ADC_DAT

Режим ожидания чтения

Если значение предыдущего результата АЦП не было прочитано из регистра ADC_DAT до завершения нового преобразования, то оно будет потеряно, и при этом будет выставлен флаг запроса на прерывание (ошибка переполнения ADEIR).

Чтобы избежать прерывания по ошибке и потерь результатов преобразования при работе в режиме неоднократных преобразований, необходимо переключить АЦП в режим ожидания чтения ADDAT, см. рисунок 16.9. Этот режим можно вызвать с помощью установки «1» в бите ADWR.

Если значение ADC_DAT не прочитано до окончания следующего преобразования, результат нового преобразования будет сохранен во временном буфере, и дальнейшие преобразования не будут начаты, при этом значения ADST и ADBSY остаются неизменными, и запрос на прерывание по окончанию преобразования не генерируется. После чтения старого значения из ADDAT содержимое временного буфера переносится в ADC_DAT, при этом генерируется запрос на прерывание. Этот механизм работает как в режиме однократного преобразования, так и в режиме многократного преобразования.

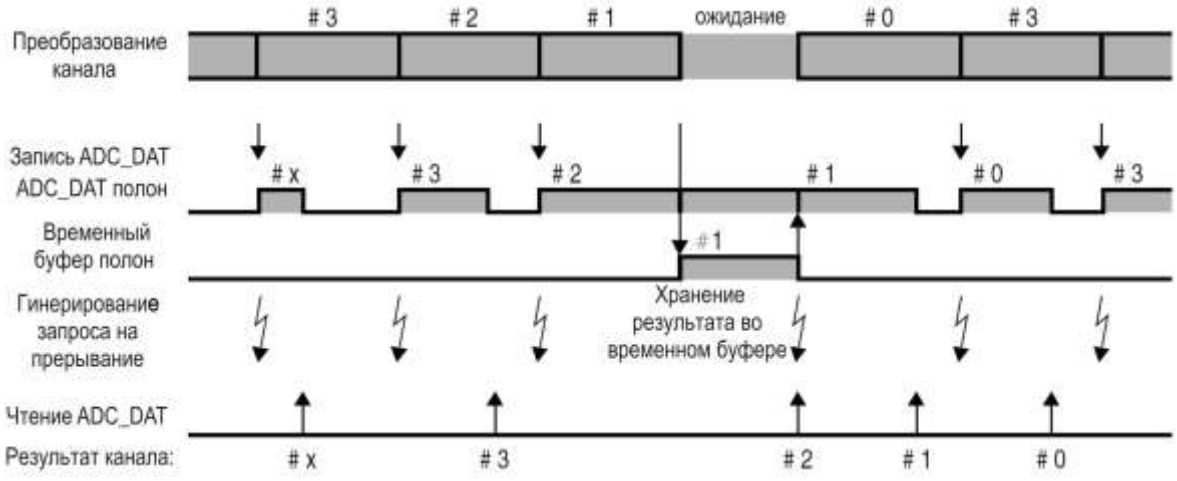


Рисунок 16.9 – Пример работы в режиме ожидания чтения из регистра ADC_DAT

Режим инжекции канала

Режим инжекции каналапозволяет преобразовывать определенный аналоговый канал, когда АЦП работает в неоднократном режиме или в режиме автоматического сканирования, без изменения текущего режима работы. После завершения преобразования напряжения этого канала АЦП продолжает работу в обычном режиме.

Режим инжекции канала разрешен при установке «1» в бите ADCIN и требуется установить режим ожидания чтения ADWR = 1_B. Преобразуемый канал выбирается в битовом поле CHNR регистра ADC_DAT2.

Примечание – При совершении преобразования напряжения инжектированного канала АЦП не изменяет значения битов CHNR регистра ADC_DAT2. Результат записывается в поле ADRES регистра ADC_DAT2. Поскольку номер канала CHNR регистра ADC_DAT2 не записывается в буфер, никогда не следует изменять его значение во время преобразования инжектированного канала, см. рисунок 16.10, в ином случае входной мультиплексор переключится на новый канал.

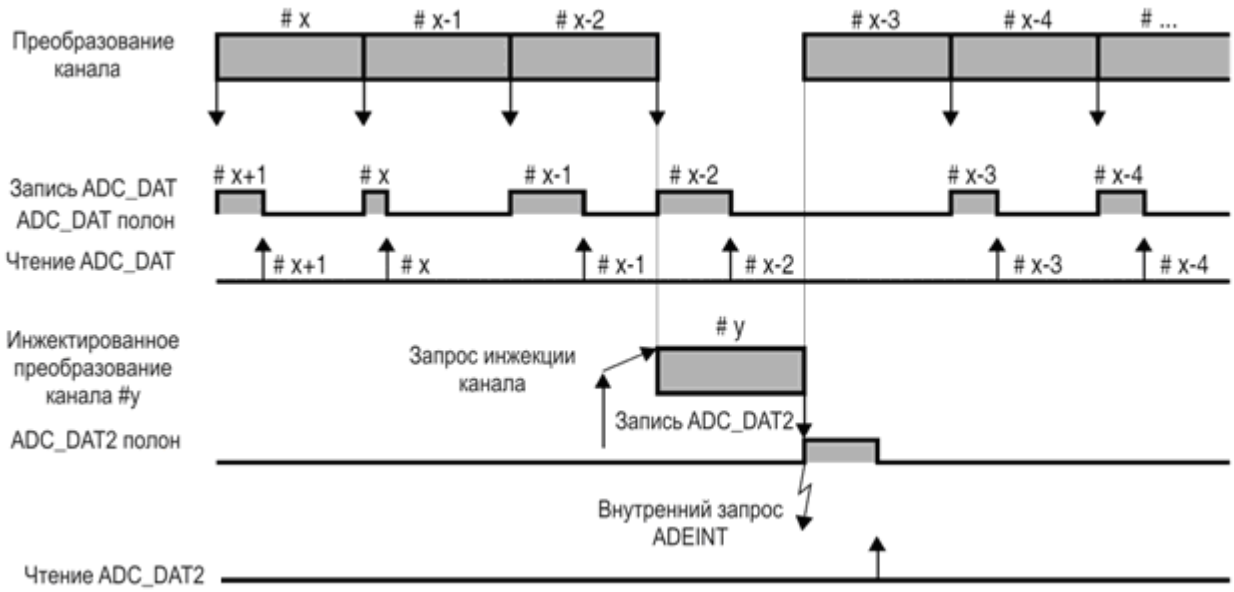


Рисунок 16.10 – Пример работы в режиме инжектированного канала из регистра ADC_ DAT2

Инжекция канала может быть вызвана тремя способами:

- Первый. Программная установка «1» в бите запроса инжекции канала ADCRQ.
- Второй. По событию сравнения или захвата в регистре захвата и сравнения CC31 модуля CAPCOM2. При этом также устанавливается «1» в бите ADCRQ.
- Третий. По совпадению периода таймера T13 блока CAPCOM6. При этом также устанавливается «1» в бите ADCRQ.

Второй способ позволяет производить преобразование в определенный момент времени при достижении счетчиком блока CAPCOM2 определенного значения или при захвате значения регистра CC31.

Примечание – В бите запроса на инжекцию канала ADCRQ будет устанавливаться «1» при любом запросе на прерывание от канала CC31 блока CAPCOM2 или при совпадении периода таймера T13 блока CAPCOM6, независимо разрешен режим инжекции или нет. Рекомендуется всегда очищать бит ADCRQ, прежде чем включать режим инжекции.

После завершения текущего преобразования будет начато инжектированное преобразование выбранного канала. После завершения преобразования результат

записывается в регистр ADC_DAT2 и выставляется флаг запроса на прерывание по завершении преобразования в инжектированном канале ADEIR (по этой причине нужен режим ожидания чтения).

Примечание – Результат инжектированного преобразования записывается непосредственно в регистр ADC_DAT2. Тем временем, если предыдущий результат не читался, он перезаписывается. Если временный буфер полон, стандартные преобразования приостановлены.

Арбитраж преобразований

В то время как АЦП находится в простое, активирование запроса на преобразования немедленно вызывает соответствующее преобразование. Если во время текущего преобразования происходит запрос на выполнение другого преобразования, то действия АЦП зависят от вида преобразования: стандартное или инжектированное.

Примечание – Запрос на преобразование активирован, если соответствующий бит управления ADST или ADCRQ переключился из «0» в «1», то есть бит должен быть сначала очищен, а затем установлен. В таблице 16.6 показаны действия АЦП в различных ситуациях.

Таблица 16.6 – Арбитраж преобразований

Текущее преобразование	Новый запрос на преобразование	
	Стандартное	Инжектированное
Стандартное	Прерывание текущего преобразования и начало нового преобразования, которое запрашивалось	Завершение текущего преобразования, после этого начнется запрашиваемое преобразование.
Инжектированное	Завершение текущего преобразования, после этого начнется запрашиваемое преобразование	Завершение текущего преобразования, после этого начнется запрашиваемое преобразование. Бит ADCRQ для последующего преобразования будет «0».

16.4 Управление временем преобразования

Величина времени преобразования программируется в двух старших битах ADCTC (управление временем преобразования) регистра ADC_CON. Эта величина используется для тактирования преобразования, см. таблицу 16.7.

Таблица 16.7 – Управление временем и определение времени преобразования

Битовое поле ADCTC	Основная частота преобразования $f_П$, МГц	Время преобразования $T_{ПР}$, мкс
00_B	$f_{ADC}/8$	$T_{ADC}\times8\times12 = 4,8$
01_B	$f_{ADC}/4$	$T_{ADC}\times4\times12 = 2,4$
10_B	$f_{ADC}/32$	$T_{ADC}\times32\times12 = 9,6$
11_B	$f_{ADC}/16$	$T_{ADC}\times16\times12 = 19,2$
Примечание – Время преобразования АЦП рассчитано при условии, что внешняя частота XTAL1 равна 40 МГц.		

16.5 Управление прерываниями АЦП

В конце каждого преобразования выставляется флаг запроса на прерывание ADCIR регистра ADC_CIC, см. рисунок 16.11. Этот флаг может использоваться для входа в стандартное прерывание по вектору ADCINT или может использоваться для входа в PEC-обслуживание. Эти действия необходимы для записи результата преобразования во внутренней памяти RAM.

Флаг запроса на прерывание ADCIR регистра ADC_EIC, см. рисунок 16.12, будет выставлен в том случае, если результат предыдущего преобразования не будет сохранен в памяти до окончания нового преобразования (прерывание по ошибки в стандартном режиме), либо если результат преобразования был сохранен в регистре ADC_DAT2 (прерывание по окончанию инжектированного преобразования). Этот флаг может использоваться для входа в стандартное прерывание по вектору ADEINT, или может использоваться для входа в PEC-обслуживание.

Примечание – Для уточнения назначений полей регистров прерываний АЦП следует обратиться к описанию общего регистра управления прерываниями «Арбитраж прерываний» в разделе 10.

ADC_CIC
регистр прерывания АЦП

SFR ($FF98_H/CC_H$) значение после Сброса: $--00_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-							GPX	ADC IR	ADC IE	ILVL				GLVL	
-							3 ч	3 ч ап	3 ч	3 ч				3 ч	

Рисунок 16.11 – Формат регистра ADC_CIC

ADC_EIC
Ошибка АЦП регистр прерывания

SFR ($FF9A_H/CD_H$) значение после Сброса: $--00_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-							GPX	ADC IR	ADC IE	ILVL				GLVL	
-							3 ч	3 ч ап	3 ч	3 ч				3 ч	

Рисунок 16.12 – Формат регистра ADC_EIC

16.6 Режим работы двойного АЦП

Режим работы двойного АЦП характеризуется независимой и одновременной работой двух аналого-цифровых преобразователей. Каждому АЦП будут соответствовать свои аналоговые входы: для АЦП1 – каналы AN7, …, AN0, для АЦП2, см. рисунок 16.13, – каналы AN15, …, AN8, а так же свои регистры, преобразователи и линии прерываний.

Для выбора канала преобразования в режиме двойного АЦП необходимо использовать три младших разряда битового поля ADCH регистров ADC2_CON и ADC2_CON1, см. рисунок 16.14, таблицу 16.8.

Режим двойного АЦП нужно выбирать с помощью регистра ADC_SWITCH. Для этого необходимо установить бит SWITCH (записать «1») в регистр. В режиме двойного АЦП используются дополнительные регистры, которые относятся к работе второго преобразователя АЦП2.

Рисунок 16.13 – Регистры и выводы портов АЦП2

Функциональное назначение всех регистров второго преобразователя, см. таблицы 16.8, 16.9, 16.10, аналогично назначению соответствующих регистров первого. Формат регистра ADC2_CON1 представлен на рисунке 16.15, ADC2_DAT – на рисунке 16.16, ADC2_DAT2 – на рисунке 16.17.

ADC2_CON
регистр управления АЦП2

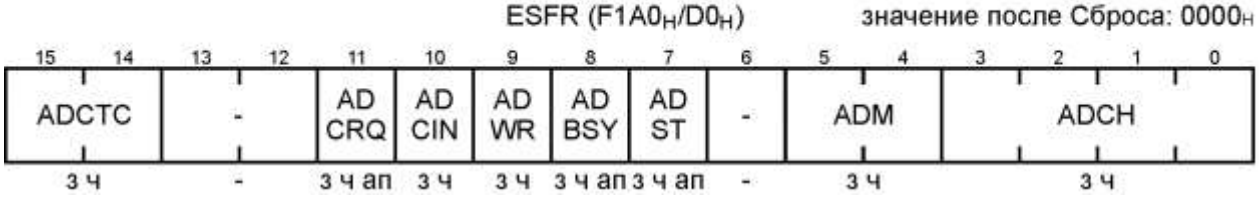


Рисунок 16.14 – Формат регистра ADC2_CON

Таблица 16.8 – Функциональное назначение полей регистра ADC2_CON

Поле	Биты	Тип	Описание
1	2	3	4
ADCTC	15, 14	Чтение Запись	Управление временем преобразования АЦП2 (определяет основную частоту преобразования второго АЦП $f_{П2}$): 00 $f_{П2} = f_{ADC}/8$ 01 $f_{П2} = f_{ADC}/4$ 10 $f_{П2} = f_{ADC}/32$ 11 $f_{П2} = f_{ADC}/16$
ADCRQ	11	Чтение Запись Аппаратное влияние	Флаг запроса на инжекцию канала АЦП2.
ADCIN	10	Чтение Запись	Разрешение инжекции канала АЦП2.
ADWR	9	Чтение Запись	Ожидания управления чтения АЦП2.
ADBSY	8	Чтение Аппаратное влияние	Флаг занятости АЦП2: 0 АЦП находится в состоянии простоя. 1 АЦП находится в активном состоянии.

Окончание таблицы 16.8

1	2	3	4
ADST	7	Чтение Запись Аппаратное влияние	Стартовый бит АЦП2: 0 Остановить начатое преобразование. 1 Начать преобразование.
ADM	5, 4	Чтение Запись	Выбор режима работы АЦП2: 00 Фиксированный канал однократного преобразования. 01 Фиксированный канал неоднократного преобразования. 10 Автоматическое сканирование однократного преобразования. 11 Автоматическое сканирование неоднократного преобразования.
ADCH	3-0	Чтение Запись	Выбор аналогового канала АЦП2. Для выбора каналов преобразования использовать три бита ADCH.2 – ADCH.0. Старший бит поля ADCH влияния на выбор канала не оказывает.

ADC2_CON1
регистр управления АЦП2

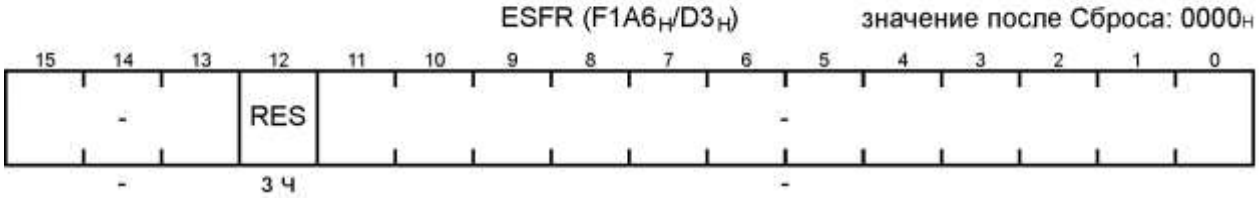


Рисунок 16.15 – Формат регистра ADC2_CON1

Таблица 16.9 – Функциональное назначение поля регистра ADC2_CON1

Поле	Биты	Тип	Описание
RES	12	Чтение Запись	Бит управления точностью (разрешающей способностью) преобразования АЦП2: 0 10-битное преобразование (устанавливается по сбросу). 1 8-битное преобразование.

ADC2_DAT
регистр результата АЦП2

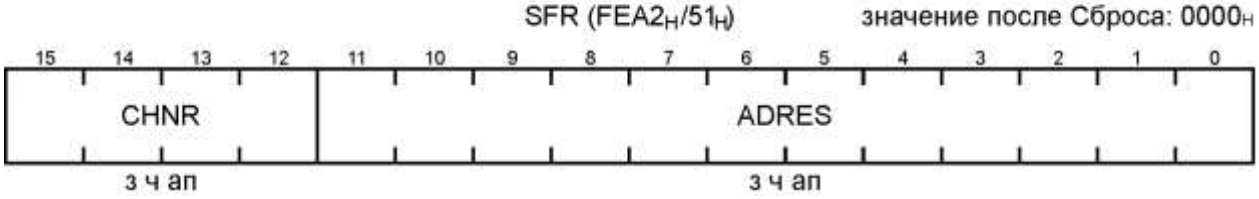


Рисунок 16.16 – Формат регистра ADC2_DAT

ADC2_DAT2
регистр результата АЦП2
инжектированного канала

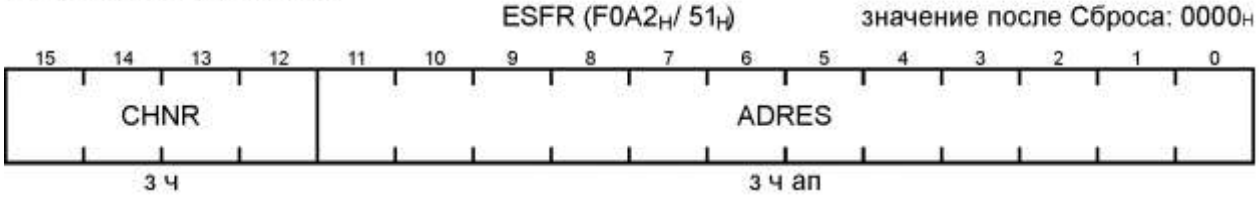


Рисунок 16.17 – Формат регистра ADC2_DAT2

Таблица 16.10 – Функциональное назначение полей регистров ADC2_DAT и ADC2_DAT2

Поле	Биты	Тип	Описание
CHNR	15-12	Чтение Запись Аппаратное влияние	Номер канала (идентифицирует преобразованный аналоговый канал)
ADRES	11-0	Чтение Запись Аппаратное влияние	Результат преобразования АЦП2. Цифровой результат самого последнего преобразования. Результат расположен следующим образом: 8-битное преобразование: ADRES.9 – ADRES.2; 10-битное преобразование: ADRES.9 – ADRES.0. Примечание – Неиспользованные биты ADRES всегда равны нулю.

АЦП 2 в режиме двойного АЦП вырабатывает свои независимые флаги прерывания. Они аналогичны прерываниям АЦП 1 и имеют свои регистры прерываний, см. рисунки 16.18, 16.19.

ADC2_CIC
регистр прерывания АЦП2

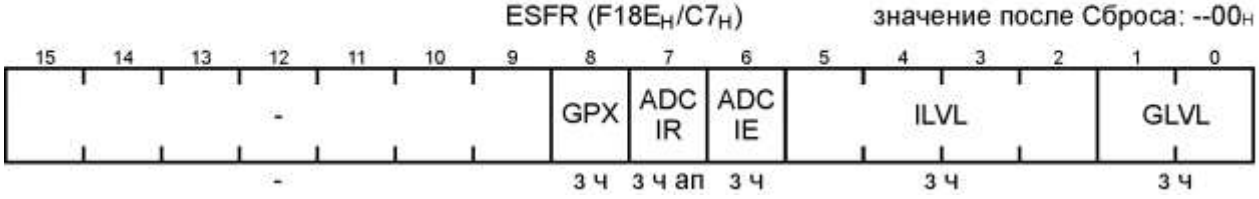


Рисунок 16.18 – Формат регистра ADC2_CIC

ADC2_EIC
Ошибка АЦП2 регистр прерывания

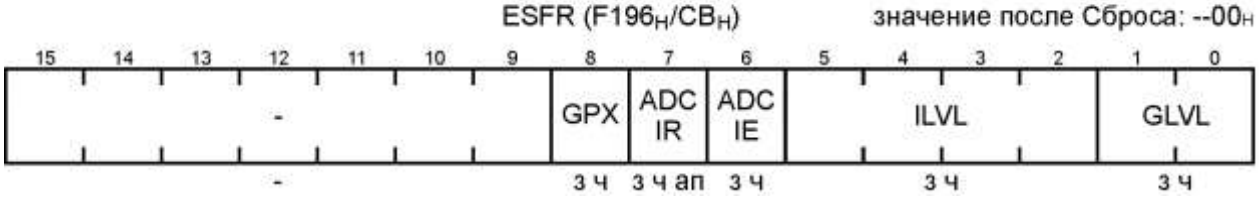


Рисунок 16.19 – Формат регистра ADC2_EIC

Примечание – Для уточнения назначений полей регистров прерываний АЦП следует обратиться к описанию общего регистра управления прерываниями «Арбитраж прерываний» в разделе 10.

16.7 Интерфейс блока АЦП

Блок АЦП подключается к своему окружению разными способами, интерфейс модуля АЦП представлен на рисунке 16.20.

Внутренние подключения

Сигнал захвата/сравнения CC31IO блока CAPCOM2 и сигнал совпадения периода таймера T13 блока CAPCOM6 подключены к АЦП. Они являются источником для инжектированных преобразований как в режиме одиночного АЦП, так и в режиме двойного АЦП. Две линии запроса на прерывание блока АЦП подключены к блоку управления прерываниями в режиме одиночного АЦП и четыре линии запроса на прерывание в режиме двойного АЦП.

Внешние подключения

Аналоговые входные сигналы АЦП подключаются к порту P5 (только вход). Два специальных вывода ∩VCC3 и ∩0V3 обеспечивают питание АЦП.

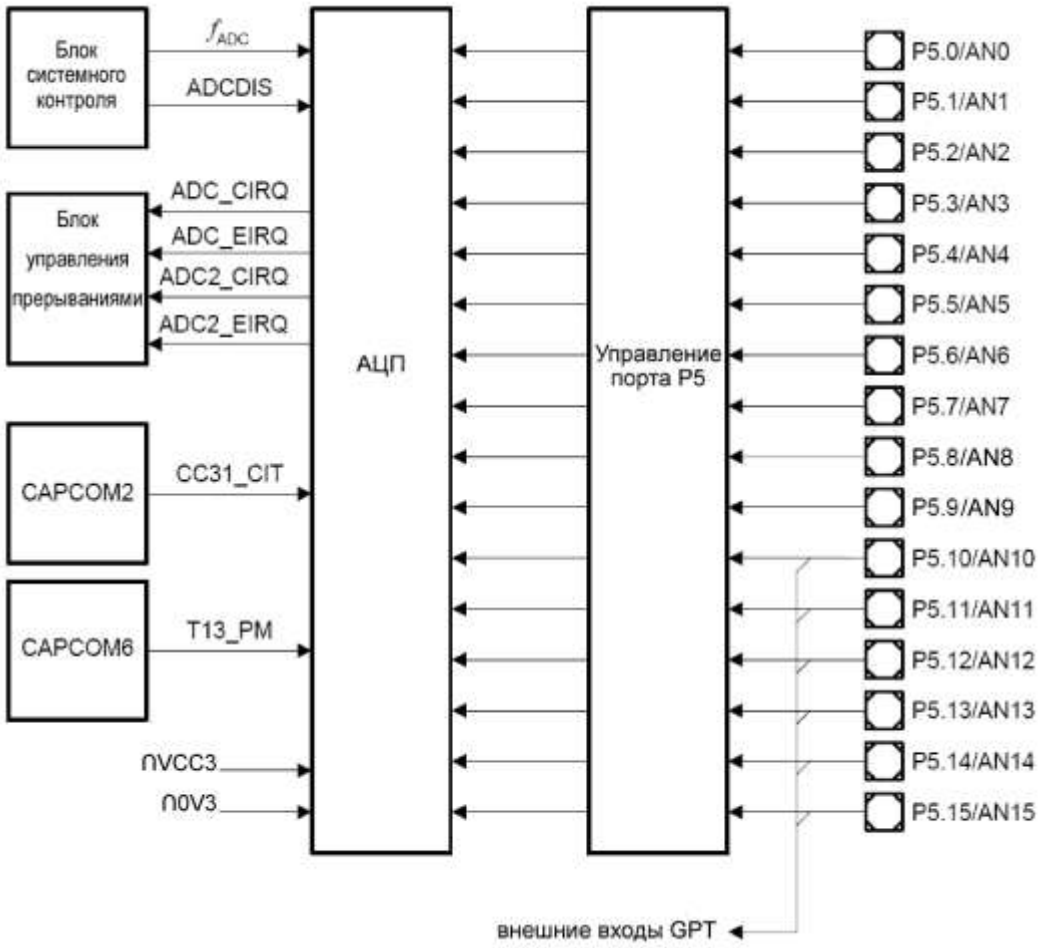
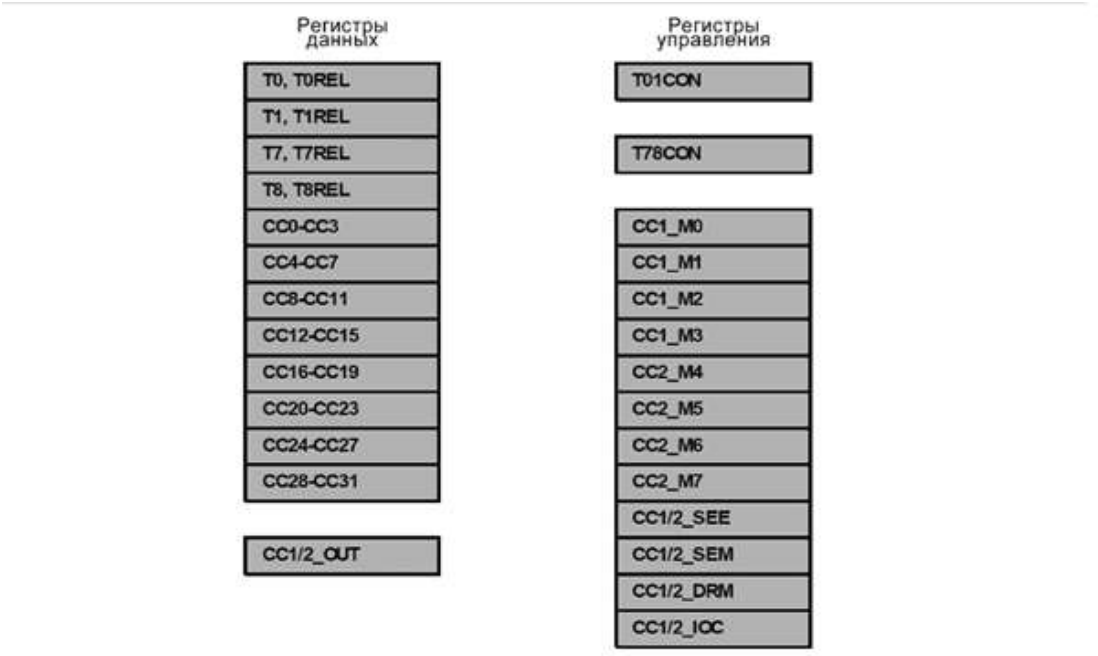


Рисунок 16.20 – Интерфейс модуля АЦП

17 Блоки захвата/сравнения CAPCOM1 и CAPCOM2

В микроконтроллере 1887ВЕ3Т находятся два идентичных блока захвата/сравнения (в дальнейшем – CAPCOM), см. рисунок 17.1, между которыми существует отличие только в подключении к выводам. Каждый CAPCOM реализует 16 каналов захвата/сравнения, которые взаимодействуют с двумя таймерами. Канал CAPCOM может захватывать содержимое таймера при некотором внутреннем или внешнем событии или сравнивать содержимое таймера с заданным значением и соответствующим образом изменять выходные сигналы.



Пояснение:

CC0...15 – CAPCOM1 Регистр 0...15.
CC1_M0...3 – CAPCOM1 Регистр управления режимом захвата/сравнения 0...3.
T01CON – CAPCOM1 Регистр управления таймерами 0/1.
T0, T1 – CAPCOM1 Регистр таймера 0/1
T0/1REL – CAPCOM1 Регистр загрузки таймера 0/1.
CC1/2_SEE – CAPCOM1/2 Регистр запуска ожидания однократного события.
CC1/2_SEM – CAPCOM1/2 Регистр режима однократного события.

CC16...31 – CAPCOM2 Регистр 16...31.
CC2_M4...7 – CAPCOM2 Регистр управления режимом захвата/сравнения 4...7.
T78CON – CAPCOM2 Регистр управления таймерами 7/8.
T7, T8 – CAPCOM2 Регистр таймера 7/8.
T7/8REL – CAPCOM2 Регистр загрузки таймера 7/8.
CC1/2_DRM – CAPCOM1/2 Регистр двухрегистрового режима сравнения.
CC1/2_OUT – CAPCOM1/2 Регистр выводов сравнения.
CC1/2_IOC – CAPCOM1/2 Регистр управления вводом-выводом.

Рисунок 17.1 – Регистры блоков CAPCOM

Каждый блок CAPCOM поддерживает генерирование и контроль временных последовательностей для 16 каналов с минимальными требованиями к программному обеспечению.

CAPCOM, см. рисунок 17.2, обычно используется для выполнения высокоскоростных задач, таких как генерация импульсов и их последовательностей, или запись времени возникновения какого-либо события. Также CAPCOM поддерживает осуществление 16 программно контролируемых прерываний.

Каждый CAPCOM состоит из двух 16-разрядных таймеров (T0/T7, T1/T8), каждый со своим регистром перезагрузки TxREL, где x = 1, 2, и банка 16 регистров захвата/сравнения двойного назначения CCy, где y = 1, …, 15 для CAPCOM1 и y = 16, …, 31 для CAPCOM2.

Тактирование таймеров программируется несколькими значениями делителей для входного (относительно модуля CAPCOM) тактового сигнала f_{CC}, или оно может быть получено с помощью сигналов переполнения/опустошения таймера T6.

T0/T7 могут также работать в режиме счетчиков (внешних входных сигналов) при тактировании внешними событиями.

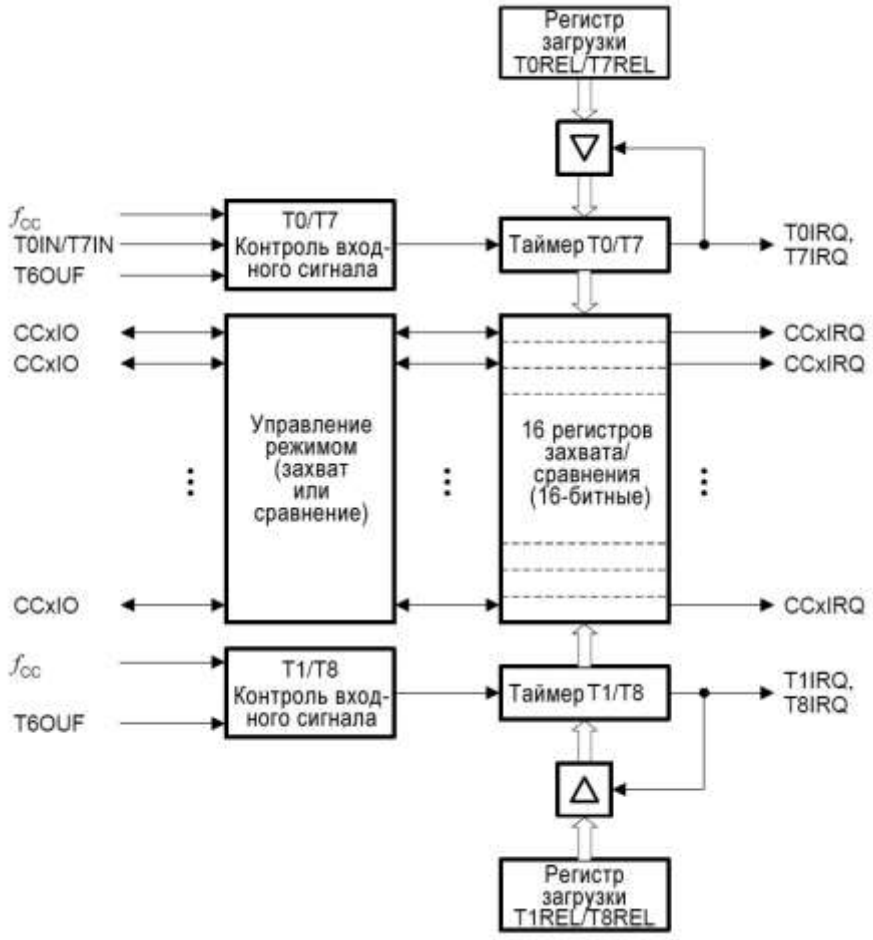
Каждый регистр захвата/сравнения может быть запрограммирован индивидуально для операций захвата или сравнения, и каждый регистр может работать с любым из двух таймеров. Каждый регистр имеет только один, ассоциирующийся с ним канал, который используется как входной сигнал для операции захвата или выходной – для операции сравнения.

Операция захвата служит причиной того, что текущее содержимое таймера запоминается в соответствующем регистре, инициируясь внешним событием.

Операция сравнения инициирует передачу сигнала на соответствующий выход в то время, когда активный таймер инкрементируется и новое значение совпадает с содержащимся значением в регистре захвата/сравнения. Совпадение также активирует соответствующий запрос прерывания.

В режиме сравнения с использованием двух регистров пара регистров контролирует один общий выход.

Выходные сигналы сравнения доступны посредством открывания выходного регистра, и могут также контролировать выходные защелки порта. Путь вывода сигнала может быть задан.



Примечание – Блок CAPCOM1 имеет каналы x = 0, …, 15 (сигналы CCxIO). Блок CAPCOM2 имеет каналы x = 16, …, 31 (сигналы CCxIRQ).

Рисунок 17.2 – Блок CAPCOM

Для переключения выходных сигналов используются два варианта.

- Ступенчатый режим. Выходной сигнал переключается непрерывно за восемь шагов. Интервалы переключения распределены точно по времени. Максимальное разрешение – $8t_{CC}$.

- Неступенчатый режим. Выходной сигнал переключается непосредственно в то же время. Максимальное разрешение $1t_{CC}$.

Основная структура CAPCOM показана на рисунке 17.2.

Блок CAPCOM1 имеет каналы x = 0, …, 15, блок CAPCOM2 имеет каналы x = 16, …, 31 (сигналы CCxIO и CCxIRQ на рисунке 17.2).

17.1 Таймеры блока CAPCOM

Основное назначение таймеров T0 и T1 блока CAPCOM1 и таймеров T7 и T8 блока CAPCOM2 – осуществлять две независимые временные оси для каналов захвата/сравнения каждого блока. С максимальным разрешением $8t_{CC}$ для ступенчатого режима и $1t_{CC}$ для неступенчатого режима.

Структура таймеров показана на рисунке 17.3 (различие между таймерами лишь в выводе, помеченном пунктиром).

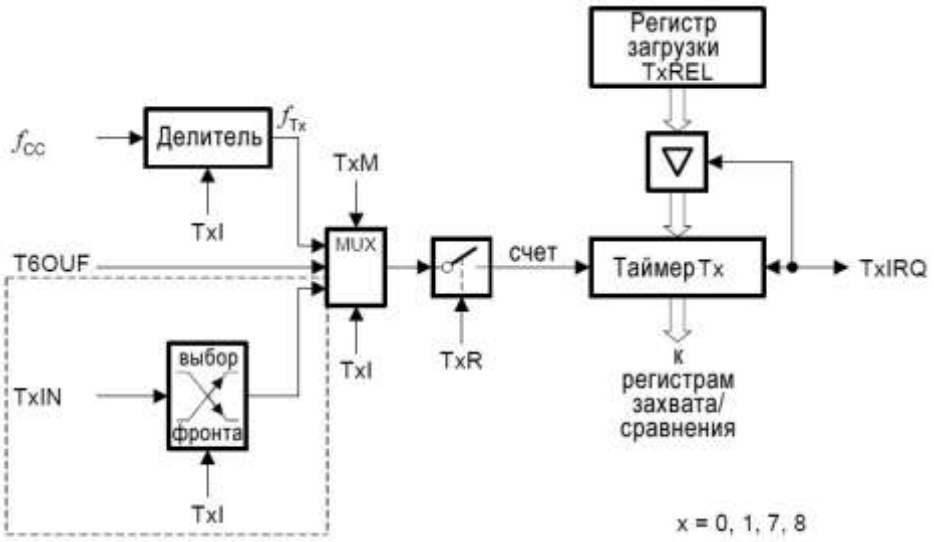


Рисунок 17.3 – Таймеры блока CAPCOM

Примечание – Когда внешний входной сигнал присоединен к входным линиям обоих таймеров T0 и T7, таймеры принимают входной сигнал синхронно. Таким образом, два таймера могут быть рассмотрены как один таймер, чье содержимое может сравниваться с 32-разрядными регистрами.

Функционирование таймеров контролируется посредством побитно адресуемых регистров T01CON и T78CON. Старший байт T01CON контролирует таймер 1, младший байт – таймер 0. Старший байт T78CON контролирует таймер 8, младший байт – таймер 7. Управление всеми четырьмя таймерами идентично (за исключением внешних выводов).

Во всех режимах таймеры считают в сторону увеличения. Текущее значение таймера доступно для ЦП через регистр таймера Tx, который не является побитно адресуемым. Когда ЦП обращается с целью записи к регистру Tx соответствующего таймера непосредственно перед его инкрементом или перезагрузкой, эти действия таймера блокируются до завершения обращения (операция записи, осуществляемая ЦП, является приоритетной).

Формат регистра CC1_T01CON представлен на рисунке 17.4, формат регистра CC2_T78CON – на рисунке 17.5, функциональное назначение полей регистров CC1_T01CON, CC2_T78CON – в таблице 17.1.

CC1_T01CON
регистр управления таймерами 0 и 1

SFR ($FF50_H/A8_H$) значение после Сброса: 0000_H

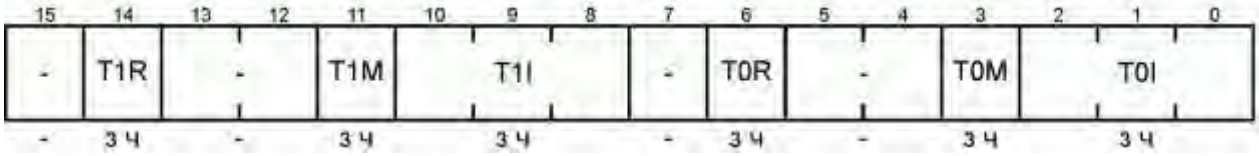


Рисунок 17.4 – Формат регистра CC1_T01CON

CC2_T78CON
регистр управления таймерами 7 и 8

SFR ($FF20_H/90_H$) значение после Сброса: 0000_H

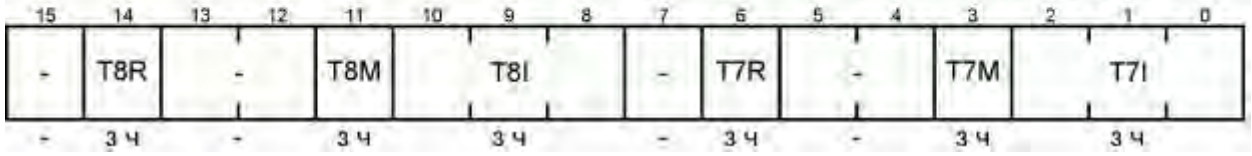


Рисунок 17.5 – Формат регистра CC2_T78CON

Таблица 17.1 – Функциональное назначение полей регистров CC1_T01CON, CC2_T78CON

Поле	Биты	Тип	Описание
TxR	14, 6	Чтение Запись	Управление запуском таймера/счетчика Tx: 0 Таймер/счетчик Tx выключен. 1 Таймер/счетчик Tx включен.
TxM	11, 3	Чтение Запись	Выбор режима таймера/счетчика Tx: 0 Режим таймера. 1 Режим счетчика.
TxI	10-8, 2-0	Чтение Запись	Выбор сигнала тактирования таймера/счетчика Tx. Режим таймера TxM = 0_B: входная частота $f_{Tx} = f_{CC}/2(<TxI>+3)$ или $f_{CC}/2(<TxI>)$ зависит от выбранного режима – ступенчатый или неступенчатый (см. таблицу 17.2). Режим счетчика (TxM = 1_B): 000 Переполнение/опустошение GPT таймера 6. 001 Появление положительного перепада входного сигнала на входе TxIN. 010 Появление отрицательного перепада входного сигнала на входе TxIN. 011 Появление любого перепада входного сигнала на входе TxIN. 1XX Зарезервированное значение. Не использовать! Примечание – Для таймеров 1 и 8 в режиме счетчиков доступно только значение 000_H (при установке других значений таймеры не считают), поскольку у этих таймеров отсутствуют входы TxIN.

Флаги работы таймеров TxR позволяют запускать и останавливать таймеры. Дальнейшее описание режимов работы таймера и операций применимо лишь к состоянию работы таймера, т. е. предполагается, что соответствующий флаг установлен.

Режимы работы таймера Tx, x = 0, 1, 7, 8

В режиме таймера TxM = 0_B входная тактовая частота таймера, получаемая из f_{CC}, делится на программируемый делитель. Каждый таймер имеет свой индивидуальный делитель, управляемый через битовые поля TxI в регистрах контроля таймеров T01CON и T78CON.

Входная частота f_{Tx}, МГц, и разрешающая способность r_{Tx}, мкс, определяются следующими формулами:

Ступенчатый режим:

$$f_{Tx} = \frac{f_{CC}}{2^{(<TxI> + 3)}}, \quad (17.1)$$

$$r_{Tx} = \frac{2^{(<TxI> + 3)}}{f_{CC}} \quad (17.2)$$

Неступенчатый режим:

$$f_{Tx} = \frac{f_{CC}}{2^{<TxI>}}, \quad (17.3)$$

$$r_{Tx} = \frac{2^{<TxI>}}{f_{CC}} \quad (17.4)$$

Когда таймер переполняется и переходит из $FFFF_H$ в 0000_H, в него загружается значение, находящееся в соответствующем этому таймеру регистре TxREL. Это записанное значение определяет период P_{Tx}, мкс, между двумя последовательными переполнениями согласно выражениям:

Ступенчатый режим:

$$P_{Tx} = \frac{(2^{16} - <TxREL>) \times 2^{(<TxI> + 3)}}{f_{CC}} \quad (17.5)$$

Неступенчатый режим:

$$P_{Tx} = \frac{(2^{16} - <TxREL>) \times 2^{<TxI>}}{f_{CC}} \quad (17.6)$$

После запуска таймера с установленным значением посредством выставления флага TxR, первое инкрементирование произойдет в пределах временного интервала, который определяется выбранным разрешением r_{Tx} таймера. Все дальнейшее инкрементирование будет происходить точно в соответствии с определенным ранее разрешением (примеры в таблице 17.2).

Значения для периода таймера начинаются со значения 0000_H. Надо иметь в виду, что некоторые числа могут быть округленными.

Таблица 17.2 – Выбор тактовой частоты для таймера Tx в режиме таймера, f_{CC} = 40 МГц

TxI	Делитель	Входная частота	Разрешение	Период, мс
Ступенчатый режим				
000_B	8	5,0000 МГц	200,0 нс	13,11
001_B	16	2,5000 МГц	400,0 нс	26,21
010_B	32	1,2500 МГц	800,0 нс	52,43
011_B	64	625,0000 кГц	1,6 мкс	104,86
100_B	128	312,5000 кГц	3,2 мкс	209,72
101_B	256	156,2500 кГц	6,4 мкс	419,43
110_B	512	78,1250 кГц	12,8 мкс	838,86
111_B	1024	39,0625 кГц	25,6 мкс	1677,72
Неступенчатый режим				
000_B	1	40,00 МГц	25,0 нс	1,6384
001_B	2	20,00 МГц	50,0 нс	3,2768
010_B	4	10,00 МГц	100,0 нс	6,5536
011_B	8	5,00 МГц	200,0 нс	13,1100
100_B	16	2,50 МГц	400,0 нс	26,2100
101_B	32	1,25 МГц	800,0 нс	52,4300
110_B	64	625,00 кГц	1,6 мкс	104,8600
111_B	128	312,50 кГц	3,2 мкс	209,7200

Режим счетчика

В режиме счетчика TxM = 1_B входная тактовая частота таймера определяется сигналом на соответствующем внешнем выводе T0IN/T7IN или переполнением/ опустошением GPT таймера T6.

Использование внешнего сигнала, приходящего на вывод TxIN в качестве сигнала счета, возможно только для таймеров T0 и T7. Для таймеров T1 и T8 допускается только использование переполнения/опустошения GPT таймера T6, выбирается TxI = 000_B.

Битовые поля T0I/T7I используются для выбора события (появление положительного фронта сигнала, отрицательного фронта или любого) для инкрементирования таймеров T0/T7.

Необходимо учитывать, что для правильной работы в этом режиме внешний сигнал и программирование выводов порта должны соответствовать определенным критериям.

Переполнение таймера и обновление

Когда содержимое таймера равно $FFFF_H$, в то время как происходит очередной счет, генерируется прерывание и таймер обновляется – в него загружается число, хранящееся в регистре TxREL. После этого таймер возобновляет инкрементирование.

Регистр перезагрузки TxREL не является побитно адресуемым. После сброса его значение равно 0000_H.

17.2 Каналы захвата/сравнения

16-битные регистры захвата/сравнения CC0, …, CC15 (CC16, ..., CC31) используются как регистры данных для операций захвата или сравнения, соответствуют таймерам T0/T7 и T1/T7. Регистры не побитно адресуемые.

Функционирование 16 регистров захвата/сравнения блока управляется 16-битным регистром контроля режима (4 бит адресуемый – CC1_M0, ..., CC1_M3, CC2_M4, ..., CC2_M7).

Каждый регистр включает биты выбора режима и выбора таймера для четырех регистров захвата/сравнения.

Регистры управления режимами захвата/сравнения для блока CAPCOM1

Форматы регистров CC1_M0, …, CC1_M3 представлены на рисунках 17.6 – 17.9, соответственно.

Рисунок 17.6 – Формат регистра CC1_M0

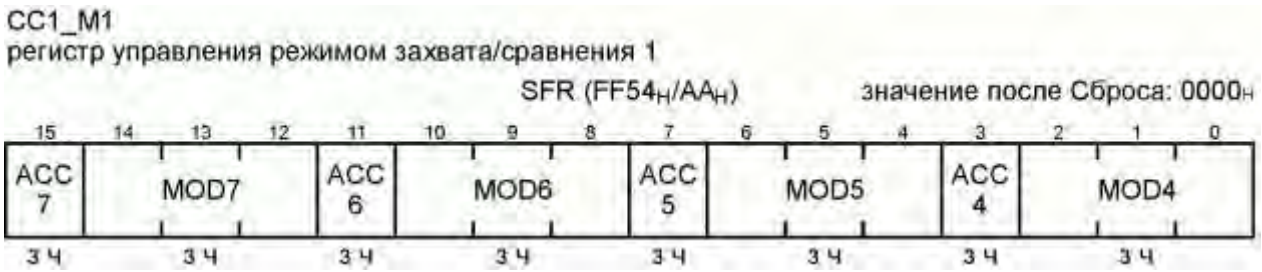


Рисунок 17.7 – Формат регистра CC1_M1

Рисунок 17.8 – Формат регистра CC1_M2

Рисунок 17.9 – Формат регистра CC1_M3

Регистры управления режимами захвата/сравнения для блока CAPCOM2 (CC31, ..., CC16)

Форматы регистров CC2_M4, …, CC2_M7 представлены на рисунках 17.10 – 17.13, соответственно.

Функциональное назначение полей регистров CC1_M0, …, CC1_M3 и CC2_M4, …, CC2_M7 представлено в таблице 17.3.

CC2_M4
регистр управления режимом захвата/сравнения 4

SFR ($FF22_H/91_H$) значение после Сброса: 0000_H

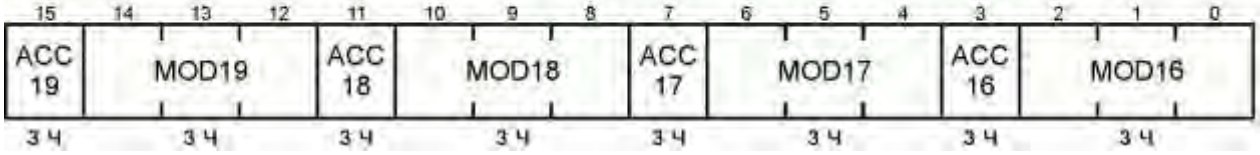


Рисунок 17.10 – Формат регистра CC2_M4

CC2_M5
регистр управления режимом захвата/сравнения 5

SFR ($FF24_H/92_H$) значение после Сброса: 0000_H

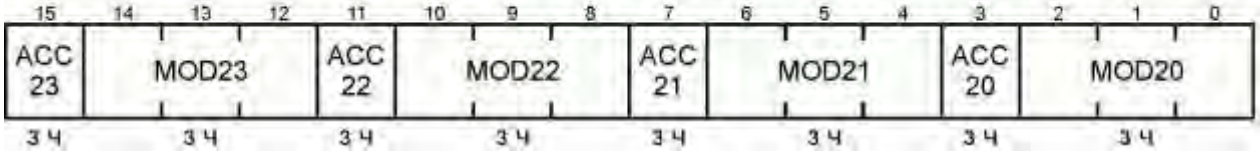


Рисунок 17.11 – Формат регистра CC2_M5

CC2_M6
регистр управления режимом захвата/сравнения 6

SFR ($FF26_H/93_H$) значение после Сброса: 0000_H

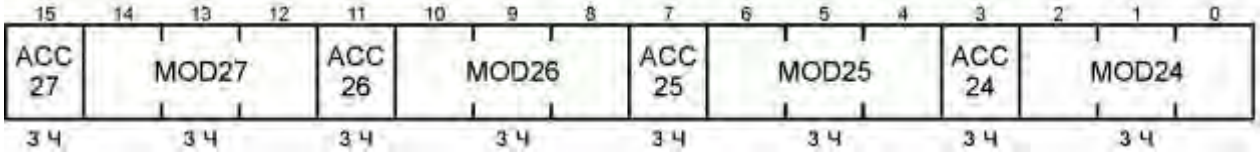


Рисунок 17.12 – Формат регистра CC2_M6

CC2_M7
регистр управления режимом захвата/сравнения 7

SFR ($FF28_H/94_H$) значение после Сброса: 0000_H

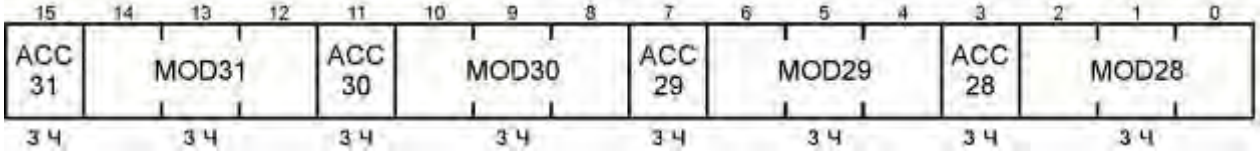


Рисунок 17.13 – Формат регистра CC2_M7

Таблица 17.3 – Функциональное назначение полей регистров CC1_M0, …, CC1_M3 и CC2_M4, …, CC2_M7

Поле	Биты	Тип	Описание
ACCy	15, 11, 7, 3	Чтение Запись	Бит распределения регистра CCy: 0 Регистр CCy выделен для работы с таймером 0 или 7, соответственно. 1 Регистр CCy выделен для работы с таймером 1 или 8, соответственно.
MODy	14-12, 10-8, 6-4, 2-0	Чтение Запись	Выбор режима регистра CCy. Кодировка в таблице 17.4.

Каждый из регистров CCy может быть запрограммирован индивидуально для режима захвата или одного из четырех режимов сравнения и может быть выделен для работы с одним из двух таймеров соответствующего блока CAPCOM. Специальный двухрегистровый режим сравнения объединяет два регистра для работы с одним выходным сигналом. Когда операции захвата или сравнения для одного из регистров CCy запрещены, он может использоваться как обыкновенный регистр для хранения данных.

Таблица 17.4 – Выбор режима захвата или режима сравнения

Режим	MODy	Выбранный режим
Нет операции	000_B	Режимы захвата и сравнения выключены. Соответствующий регистр может быть использован как регистр хранения данных.
Операции захвата	001_B	Захват по переднему фронту сигнала на входе CCyIO.
	010_B	Захват по заднему фронту сигнала на входе CCyIO.
	011_B	Захват по каждому фронту сигнала на входе CCyIO.
Операции сравнения	100_B	Режим сравнения 0. Допускаются только прерывания в течение периода счета таймера. Может быть включен двухрегистровый режим для регистров банка 2.
	101_B	Режим сравнения 1. Допускаются прерывания в течение периода счета таймера. При каждом совпадении инвертируется сигнал на соответствующем выходе. Может быть включен двухрегистровый режим для регистров банка 1.
	110_B	Режим сравнения 2. Допускается только одно прерывание в течение периода счета таймера
	111_B	Режим сравнения 3. Допускается только одно прерывание в течение периода счета таймера. При каждом совпадении сигнал на соответствующем выходе переводится в высокий уровень. При переполнении таймера сигнал на соответствующем выходе переводится в низкий уровень.
Примечание – Захват или сравнение на канале 31 может использоваться для активации канала АЦП (при условии, что АЦП включен).		

Режим захвата

В режиме захвата, см. рисунок 17.14, текущее содержимое таймера записывается в соответствующий регистр захвата/сравнения в ответ на внешнее событие. Это делается, например, для того, чтобы записать время возникновения внешнего события или измерить промежуток времени между двумя внешними событиями.

Событие, вызывающее захват содержимого таймера может быть запрограммировано. Это может быть передний фронт (перепад из «0» в «1») внешнего сигнала, задний фронт (перепад из «1» в «0) или любой фронт. Программирование осуществляется записью поля MODy в соответствующем регистре контроля режима. При появлении внешнего запрограммированного события, содержимое таймера записывается в соответствующий регистр захвата/сравнения, при этом активируется соответствующая линия прерывания CCyIRQ.

Примечание – Вход захвата может использоваться как дополнительный внешний вход прерывания. Операция захвата в этом случае игнорируется.

Для выбора таймера, содержимое которого будет захвачено, программируется бит контроля АССу каждого регистра в регистре контроля режима.

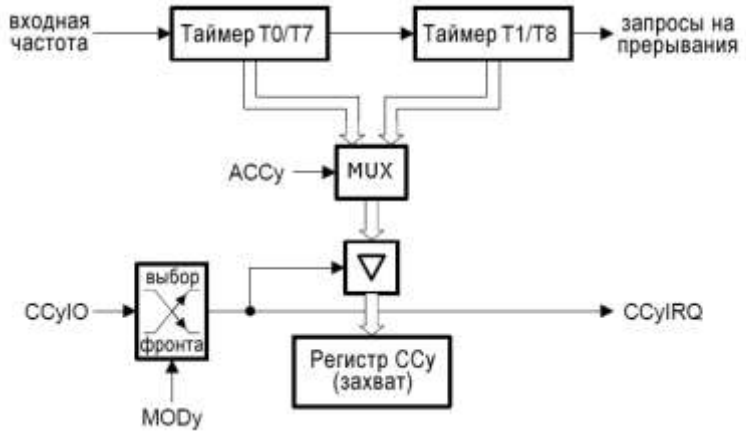


Рисунок 17.14 – Блок-диаграмма режима захвата

Для операции захвата соответствующий вывод должен быть запрограммирован как вход. Для уверенности в том, что фронт входного сигнала определен корректно, уровень входного сигнала до очередного изменения должен держаться высоким или низким определенное время (минимальное число тактов).

Режимы сравнения

Режимы сравнения позволяют реализовывать прерывания и/или изменения выходного сигнала, а также генерирование импульсных последовательностей с минимальной программной загрузкой. Во всех режимах сравнения, 16-битное значение, хранящееся в регистре захвата/сравнения ССу (в дальнейшем называемое «сравниваемое значение»), последовательно сравнивается с содержимым таймеров (T0/T7 или T1/T8). В случае совпадения текущего значения таймера со сравниваемым значением, активируется линия прерывания, ассоциирующаяся с активированным регистром ССу и, в зависимости от режима сравнения, может быть сгенерировано переключение или установка выходного сигнала на соответствующем выводе CCyIO.

Доступны четыре различных режима сравнения, которые выбираются индивидуально для каждого регистра захвата/сравнения полем MODy в соответствующем регистре контроля режима.

Режимы 0 и 2 не влияют на выходные сигналы.

Помимо обычных режимов может применяться двухрегистровый режим сравнения. Двухрегистровый режим сравнения позволяет двум регистрам работать с одним выходом. Два различных сравниваемых значения могут программироваться для контроля последовательности появления фронтов сигнала.

Во всех режимах сравнения компаратор осуществляет сравнение «на совпадение». Это означает, что результат сравнения положительный только в том случае, когда содержимое таймера совпадает с содержимым регистра сравнения. К тому же, компаратор активирован только в момент такта точно после аппаратного инкрементирования таймера. Такой механизм обусловлен тем, чтобы предотвратить повторное сравнение (дающее положительны результат) в случае если таймер не работает с максимально возможной частотой входного тактового сигнала (в любом режиме – захвата или сравнения). В этом случае, содержимое таймера остается неизменным в течение длительного времени. При этом программное изменение содержимого таймера не повлияет на компаратор. В случае если таймер загружен программно таким же значением, что содержится в одном из

регистров сравнения, сравнение не даст положительного результата (т. е. не будет идентифицироваться как «совпадение»). Если регистр сравнения загружен значением меньшим, чем текущее значение таймера, не будет осуществлено никаких действий.

Когда два или более регистров сравнения программируются одним и тем же сравниваемым значением (в ступенчатом режиме прерывания и выходные сигналы генерируются последовательно), то соответствующие им флаги прерываний будут установлены, и выходные сигналы будут сгенерированы после того, как выбранный таймер досчитает до сравниваемого значения. Дальнейшее сравнение события с таким же сравниваемым значением запрещается (полностью) до тех пор, пока таймер не переключится снова или не будет перезагружен программно. После сброса сравниваемое событие для регистра CCy становится возможным, только если выбранный таймер инкрементировался или был перезагружен программно, и был выбран один из режимов сравнения.

Режим сравнения 0

В режиме сравнения 0, см. рисунок 17.15, возможны лишь прерывания, он может быть использован для программного расчета времени. В этом режиме линия прерывания CCyIRQ активируется каждый раз, когда возникает совпадение содержимого регистра сравнения CCy и выбранного таймера. Некоторые из совпадений возможны в пределах одного периода таймера, если сравниваемое значение в регистре CCy изменяется в течениие периода таймера. В этом режиме не оказывается влияние на соответствующий портовый сигнал CCyIO, который при этом может использоваться для обычного ввода-вывода.

Примечание – Если режим 0 программируется для одного из регистров второго банка, то для него может быть разрешен двухрегистровый режим сравнения.

Режим сравнения 1

В режиме сравнения 1, см. рисунок 17.15, оказывается влияние на соответствующий выходной сигнал. Основные функции аналогичны режиму 0. Каждый раз, когда возникает совпадение между содержимым регистра сравнения CCy и выбранным таймером, активируется соответствующая линия прерывания CCyIRQ. К тому же, переключается (инвертируется) соответствующий выходной сигнал. Некоторые из этих совпадений возникают в течение одного периода таймера, если сравниваемое значение в регистре CCy изменяется в течение периода таймера.

Примечание – Если режим 1 программируется для одного из регистров первого банка, то для него может быть разрешен двухрегистровый режим сравнения.

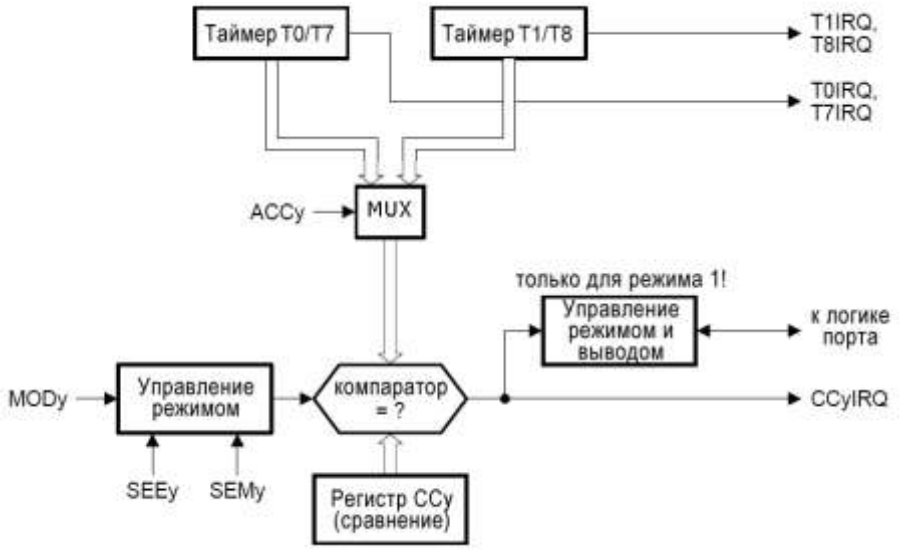


Рисунок 17.15 – Блок-диаграмма режимов сравнения 0 и 1

На рисунке 17.16 показано несколько вариантов для режимов 0 и 1.

Во всех примерах значение, загружаемое в таймер по переполнению, равно $FFF9_H$.

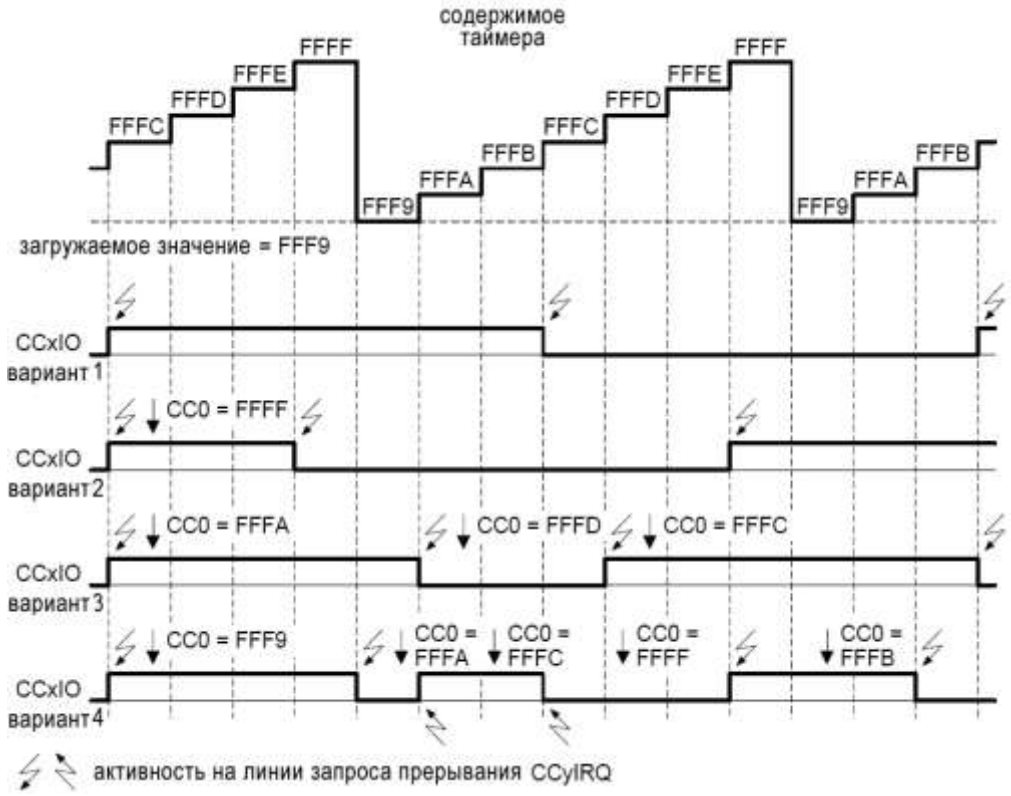


Рисунок 17.16 – Примеры для режимов 0 и 1

Пояснения к рисунку 17.16

Вариант 1. Регистр CCy хранит значение $FFFC_H$. Когда таймер достигает этого значения, возникает совпадение и активируется линия прерывания CCyIRQ. В режиме 0 на этом процесс заканчивается. В режиме 1, помимо этого, происходит переключение выхода соответствующего порта по причине инвертирования выходного сигнала. Если содержимое регистра CCy не изменяется, это действие выполнятся каждый раз, когда таймер достигает запрограммированного сравниваемого значения.

Вариант 2. После совпадения с числом $FFFC_H$, программа загружает в регистр сравнения CCy значение $FFFF_H$. По мере продолжения счета, таймер в итоге достигает нового значения и возникает очередное совпадение, активируя линию прерывания в обоих режимах и переключение выходного сигнала (режим 1). Если после этого сравниваемое значение не изменяется, следующее совпадение возникнет снова по достижении таймером значения $FFFF_H$.

Данные примеры иллюстрируют, что дальнейшие совпадения возможны в пределах текущего периода таймера (это составляет одно из отличий от режимов 2 и 3).

Вариант 3. Новое сравниваемое значение, большее по значению, чем находящееся в таймере на текущий момент, является причиной нового совпадения в пределах текущего периода таймера. Регистр сравнения загружается значением $FFFA_H$ после первого совпадения с $FFFC_H$. Тем временем, таймер уже просчитал это значение. Отсюда следует, что значение будет храниться до тех пор, пока таймер не достигнет значения $FFFA_H$ в следующем периоде своей работы, до получения желаемого совпадения. Загрузка в регистр CCy в этот момент значения, превышающего текущее значение таймера, вызовет совпадение в пределах настоящего периода.

Вариант 4. Сравниваемое значение равно значению, загружаемому в таймер при перезагрузке или максимально возможному $FFFF_H$.

Режим сравнения 2

В режиме 2, см. рисунок 17.17, возможны только прерывания, подобно режиму 0. Главное отличие заключается в том, что только одно совпадение и соответствующее прерывание возможно в течение одного периода таймера.

Когда возникает совпадение в режиме 2 в первый раз за период таймера, активируется линия прерывания CCyIRQ. Все последующие совпадения в текущем периоде таймера запрещаются, даже если в регистр будет записано новое сравниваемое значение, большее по значению, чем текущее значение таймера. Запрет снимается только по перезагрузке таймера. Новое сравниваемое значение, записанное в регистр сравнения после первого совпадения, будет использовано только в следующем периоде таймера.

Режим сравнения 3

Режим 3 основан на режиме 2, см. рисунок 17.17. В этом режиме оказывается влияние на соответствующий вывод порта. В этом режиме возможно только одно совпадение в пределах одного периода таймера.

Когда возникает совпадение в первый раз за текущий период таймера, активируется линия прерывания CCyIRQ, и соответствующий выходной сигнал устанавливается в высокий уровень «1». Все последующие совпадения в текущем периоде таймера запрещаются, даже если в регистр будет записано новое сравниваемое значение, большее по значению, чем текущее значение таймера. Запрет снимается только по перезагрузке таймера. Новое сравниваемое значение, записанное в регистр сравнения после первого совпадения, будет использовано только в следующем периоде таймера.

Сигнал переполнения таймера в этом режиме используется для сброса соответствующего выходного сигнала в низкий уровень «0».

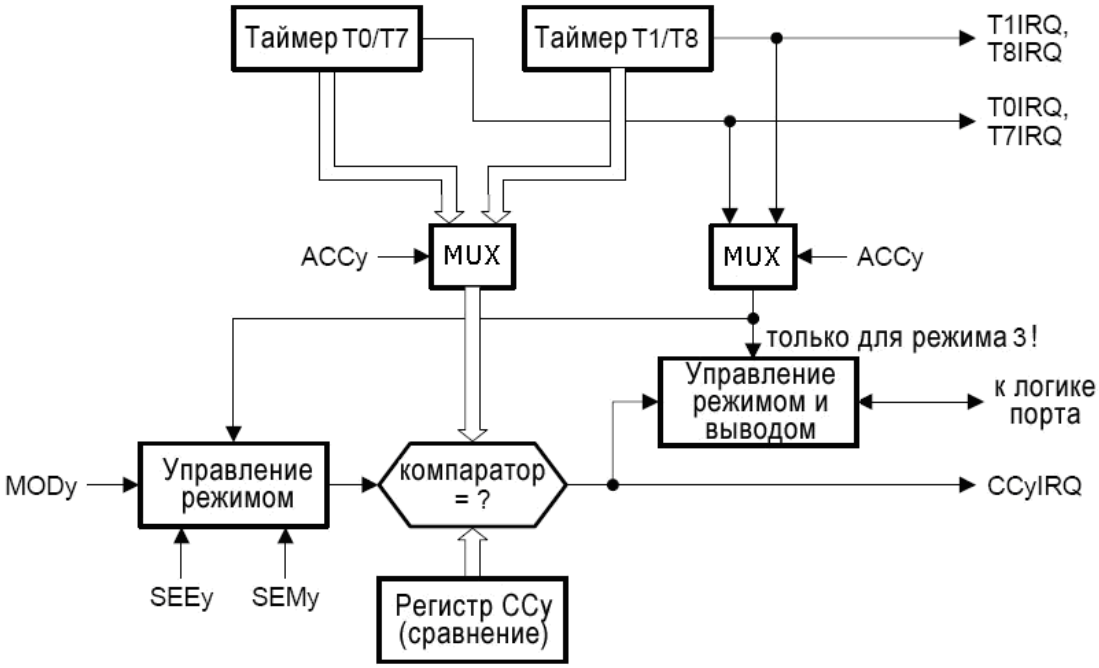


Рисунок 17.17 – Блок-диаграмма режимов сравнения 2 и 3

Примечание – Портовая защелка и сигнал остаются без изменения в режиме 2.

Особое внимание стоит уделить случаю, когда сравниваемое значение равно значению, загружаемому в таймер при перезагрузке. В этом случае сигнал появления совпадения будет стараться установить выходной сигнал в «1», в то время как переполнение таймера будет сбрасывать его в «0». Во избежание конфликта уровней, в случае возникновения такой ситуации выходной сигнал по умолчанию остается без изменений.

Примечание – Когда сравниваемое значение изменяется от значения, превышающего текущее значение таймера, до значения меньшего, чем значение таймера, новое значение не будет использовано до начала следующего периода таймера.

На рисунке 17.18 иллюстрировано несколько примеров режимов 2 и 3.

Во всех примерах значение, загружаемое в таймер при перезагрузке, равно $FFF9_H$.

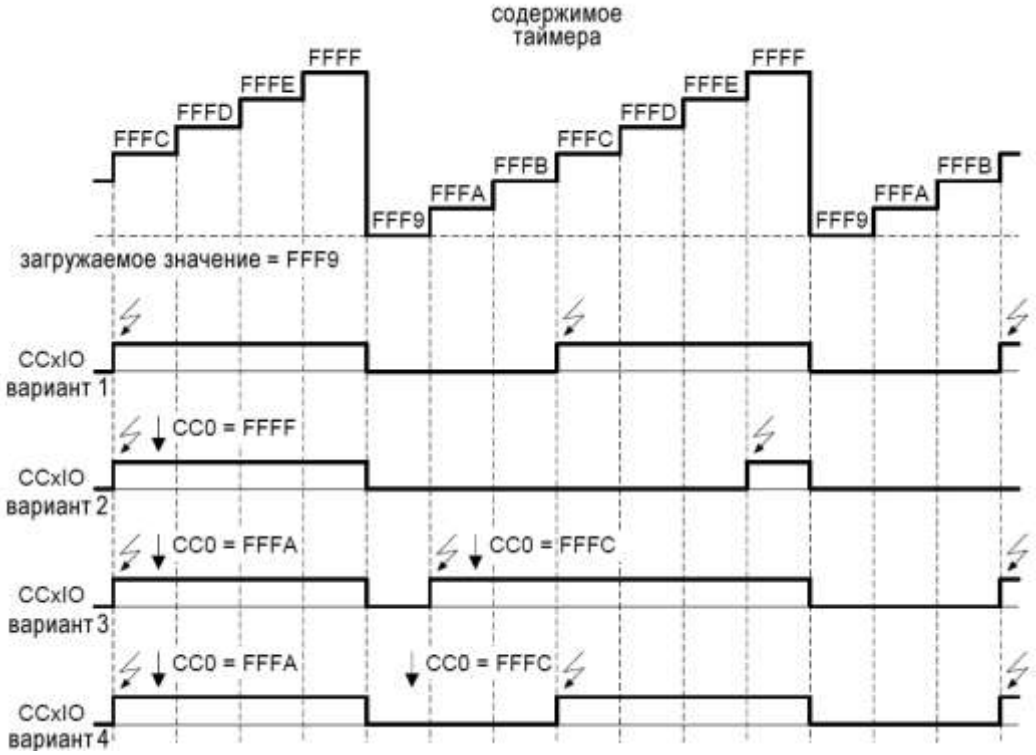


Рисунок 17.18 – Примеры для режимов 2 и 3

Пояснения к рисунку 17.18

Вариант 1. Регистр CCy хранит значение $FFFC_H$. Когда таймер достигает этого значения, возникает совпадение и активируется линия прерывания CCyIRQ. В режиме 2 на этом процесс заканчивается. В режиме 3, помимо этого, происходит переключение соответствующего вывода порта в 1. Таймер продолжает счет и достигает переполнения. В этот момент вывод порта сбрасывается в 0. Следует заметить (хотя это и не показано на диаграмме), что сигнал переполнения таймера активирует соответствующую линию прерывания TxIRQ. Если содержимое регистра CCy не изменилось, вывод порта будет установлен снова в течение последующего периода таймера и сброшен снова по переполнении таймера. Этот метод идеально подходит для формирования ШИМ сигнала с минимальной программной поддержкой. Ширина импульса изменяется в зависимости от сравниваемого значения.

Вариант 2. Процесс сравнения запрещается после появления первого совпадения в пределах одного периода таймера. При первом совпадении с числом $FFFC_H$ генерируется запрос на прерывание и устанавливается вывод порта. Дальнейшие операции сравнения запрещаются. Если теперь сравниваемое значение запишется в регистр CCy, запроса на прерывание не возникнет и состояние вывода порта не изменится, не смотря на то, что новое сравниваемое значение может быть больше, чем текущее значение таймера. Только после переполнения таймера будет обнаружено следующее совпадение. Поскольку программно можно записать новое сравниваемое значение независимо от того больше оно или меньше текущего значения таймера, это создает идеальные условия для формирования последовательности ШИМ. Это дает уверенность в том, что новое

значение (обычно записываемое в регистр сравнения соответственно программе обслуживания прерывания) будет иметь влияние только в течение следующего периода таймера.

Вариант 3. Дальнейшие примеры запрета логики сравнения показаны на рисунке 17.19.

Вариант 4. Новое сравниваемое значение записывается в регистр сравнения до первого появления совпадения в пределах периода таймера. Можно видеть, что изначально запрограммированное совпадение (с $FFFA_H$) не возникнет. Первое совпадение будет обнаружено при сравнении с $FFFC_H$. Тем не менее, важно заметить, что перепрограммирование регистра сравнения бывает асинхронным, т. е. регистр записывается безотносительно текущего состояния таймера. Опасность заключается в том, что последствия этого непредсказуемы. Только в случае, если таймер уже достиг изначально запрограммированного сравниваемого значения $FFFA_H$ к моменту времени, когда происходит перепрограммирование и возникло совпадение, новое значение будет иметь силу в следующем периоде таймера.

Примеры на рисунке 17.19 отражают особые варианты режимов 2 и 3.

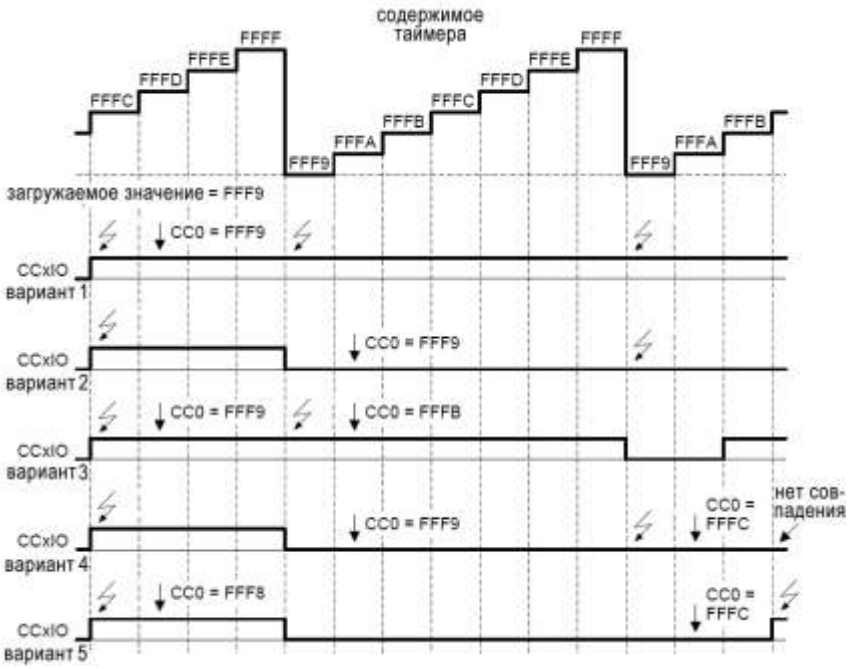


Рисунок 17.19 – Особые случаи в режимах 2 и 3

Пояснения к рисунку 17.19

Вариант 1 иллюстрирует эффект того, что сравниваемое значение равно значению, загружаемому в таймер при переполнении. Прерывание генерируется в обоих режимах. В режиме 3 выходной сигнал остается неизменным – высоким. Установка сравниваемого значения, равного загружаемому в таймер значению при переполнении, допускает стабильное формирование последовательности ШИМ. Важно здесь то, что прерывание при сравнении продолжает генерироваться и может быть использовано для загрузки следующего сравниваемого значения. Таким образом, для данного варианта (вариант 3) нет специальных требований.

Варианты 2, 4 и 5 отражают различные возможности для генерирования сигнала.

Вариант 2 показывает асинхронное перепрограммирование сравниваемого значения, равного значению, загружаемому в таймер при переполнении. В конце текущего периода таймера генерируется прерывание, которое позволяет программе загрузить следующее сравниваемое значение. Неудобство этого метода в том, что, по крайней мере, два периода

таймера пройдут, пока новое очередное сравниваемое значение будет задействовано. Совпадение со значением $FFF9_H$ (загружается в таймер при переполнении) запрещает дальнейшее сравнение до конца периода таймера. Смотри вариант 4 на рисунке 17.19.

Вариант 5. В данном случае регистр сравнения загружен числом $FFF8_H$ (значением меньшим, чем значение загрузки таймера). Таким образом, таймером никогда не достигнется этого значения и совпадения не будет. Выходной сигнал будет установлен в 0 после первого переполнения таймера. Тем не менее, после второго переполнения, программа загрузит в регистр сравнения следующее сравниваемое значение. Так как запрета на сравнение не возникнет (не будет совпадений), новое сравниваемое значение, записанное в текущем периоде, будет обработано.

Двухрегистровый режим сравнения

Двухрегистровый режим, см. блок-диаграмму на рисунке 17.22, дает возможность в последующем упростить программное обеспечение приложений. В этом режиме два регистра сравнения работают вместе для контролирования одного выхода. Этот режим активируется через регистр DRM или специальной комбинацией режимов сравнения для двух регистров.

Для двухрегистрового режима 16 регистров захват/сравнения блока CAPCOM определяются как два банка по 8 регистров. Младший банк регистров – первый, старший банк – второй. Для настоящего режима регистр банка 1 и регистр банка 2 формируют регистровую пару. Оба регистра этой регистровой пары оперируют с выводом, который соответствует регистру из банка 1, при этом вывод, соответствующий регистру из банка 2, переводится в состояние логического нуля.

Соотношения регистров банков показаны в таблице 17.5.

Таблица 17.5 – Регистровые пары для двухрегистрового режима

Блок CAPCOM 1				Блок CAPCOM 2			
Регистровая пара		Исполь-зуемый выход	Управляющее поле в CC1DRM	Регистровая пара		Исполь-зуемый выход	Управляющее поле в CC2DRM
Банк 1	Банк 2			Банк 1	Банк 2		
CC0	CC8	CC0IO	DR0M	CC16	CC24	CC16IO	DR0M
CC1	CC9	CC1IO	DR1M	CC17	CC25	CC17IO	DR1M
CC2	CC10	CC2IO	DR2M	CC18	CC26	CC18IO	DR2M
CC3	CC11	CC3IO	DR3M	CC19	CC27	CC19IO	DR3M
CC4	CC12	CC4IO	DR4M	CC20	CC28	CC20IO	DR4M
CC5	CC13	CC5IO	DR5M	CC21	CC29	CC21IO	DR5M
CC6	CC14	CC6IO	DR6M	CC22	CC30	CC22IO	DR6M
CC7	CC15	CC7IO	DR7M	CC23	CC31	CC23IO	DR7M

Двухрегистровый режим программируется индивидуально для каждой регистровой пары. Выбирается двухрегистровый режим определенной комбинацией режимов каждого регистра пары. Банк 1 должен программироваться на режим 1 (влияние на порт), банк 2 – на режим 0 (только прерывание).

Двухрегистровый режим может контролироваться (может быть разрешен или запрещен) для каждой регистровой пары с помощью битовых полей DRxM в регистре CC1_DRM, см. рисунок 17.20, или CC2_DRM, см. рисунок 17.21, функциональное назначение полей регистров CC1_DRM и CC2_DRM представлено в таблице 17.6.

Рисунок 17.20 – Формат регистра CC1_DRM

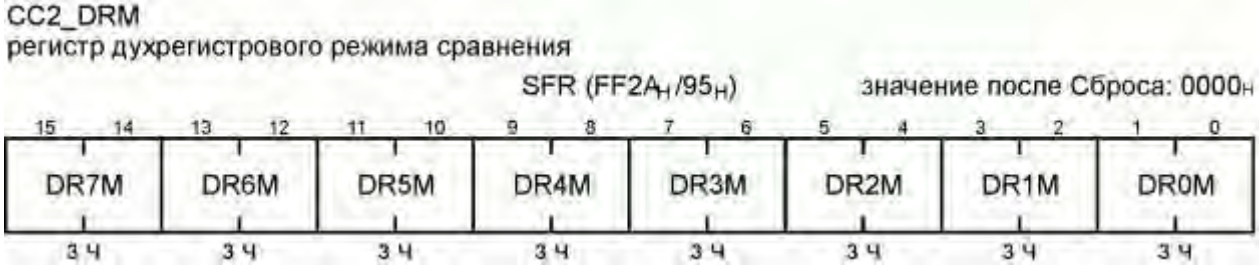


Рисунок 17.21 – Формат регистра CC2_DRM

Таблица 17.6 – Функциональное назначение полей регистров CC1_DRM и CC2_DRM

Поле	Биты	Тип	Описание
DRxM	1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14,	Чтение Запись	Выбор режима сравнения пары x регистров: 00 Двухрегистровый режим включается автоматически при условии, что регистры пары работают в режимах 1 и 0. 01 Двухрегистровый режим выключен. 10 Двухрегистровый режим включен независимо от комбинации режимов регистров. 11 Зарезервировано. Примечание – «x» – номер регистровой пары.

На блок-диаграмме, представленной на рисунке 17.22, банк 2 соотносится с CCz, в то время как банк 1 – с CCy.

Когда возникает совпадение для одного из двух регистров регистровой пары CCy или CCz, активируется соответствующая линия прерывания CCyIRQ или CCzIRQ, и вывод CCyIO, соответствующий банку 1, переключается. Генерируемое прерывание всегда соответствует регистру, с которым возникло совпадение.

Примечание – Если совпадение возникает одновременно для двух регистров регистровой пары, вывод CCyIO переключится только один раз, но при этом будут сгенерированы два запроса на прерывания.

Каждый регистр пары может быть независимо выделен для работы с одним из двух таймеров. Это открывает широкие возможности, например, два таймера могут работать в разных режимах с разным периодом.

Примечание – Сигналы CCzIO (не предназначены для двухрегистрового режима сравнения) могут быть использованы для общего ввода-вывода.

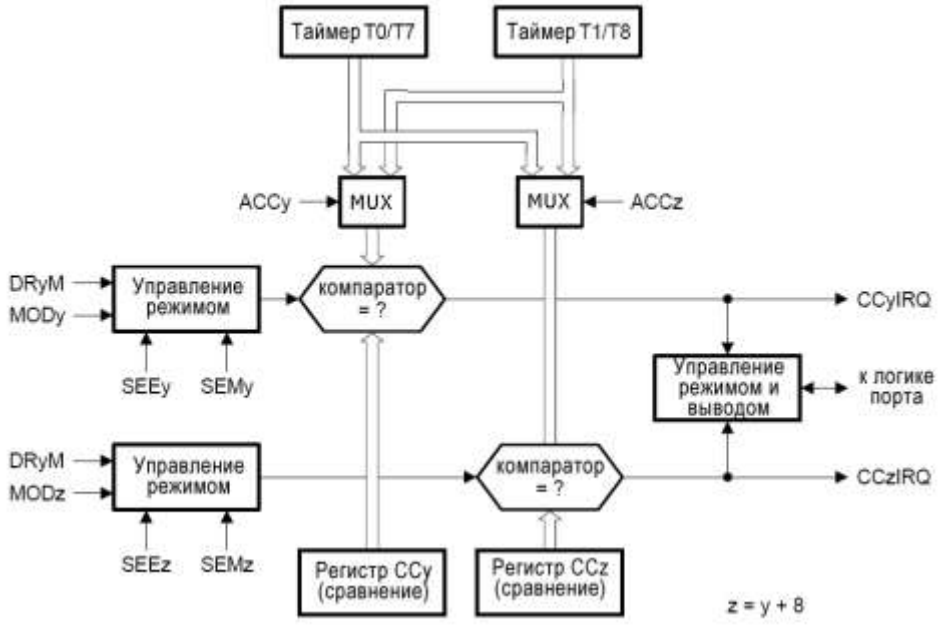


Рисунок 17.22 – Блок-диаграмма двухрегистрового режима

17.3 Генерирование выходных сигналов сравнения

В подразделе рассматривается взаимодействие между блоком CAPCOM и портовой логикой. Схема на рисунке 17.25 показывает внутреннее устойство блока «Управление режимом и выводом».

Каждый выходной сигнал защелкивается в соответствующем ему бите регистра выходной защелки CCx_OUT. Биты обновляются каждый раз при возникновении события для сравнения. Эти биты соединяются с соответствующими выводами порта и являются альтернативными функциями.

Сравниваемые сигналы могут также непосредственно воздействовать на портовую защелку Px. В этом случае портовая защелка должна быть сконфигурирована для работы с соответствующими выводами. Непосредственные опции портовой защелки запрещены в неступенчатом режиме или могут быть запрещены установкой бита PL в регистре CCx_IOC.

Регистр CCx_OUT всегда изменяется параллельно изменениям выводов портовой защелки. Формат регистра CC1_OUT представлен на рисунке 17.23, регистра CC2_OUT – на рисунке 17.24, функциональное назначение полей этих регистров – в таблице 17.7.

CC1_OUT
регистр выводов сравнения

SFR ($FF5C_H/AE_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC15IO	CC14IO	CC13IO	CC12IO	CC11IO	CC10IO	CC9IO	CC8IO	CC7IO	CC6IO	CC5IO	CC4IO	CC3IO	CC2IO	CC1IO	CC0IO
з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап

Рисунок 17.23 – Формат регистра CC1_OUT

CC2_OUT
регистр выводов сравнения

SFR ($FF2C_H$ /96_H) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC 15 IO	CC 14 IO	CC 13 IO	CC 12 IO	CC 11 IO	CC 10 IO	CC 9 IO	CC 8 IO	CC 7 IO	CC 6 IO	CC 5 IO	CC 4 IO	CC 3 IO	CC 2 IO	CC 1 IO	CC 0 IO
з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап

Рисунок 17.24 – Формат регистра CC2_OUT

Таблица 17.7 – Функциональное назначение полей регистров CC1_OUT и CC2_OUT

Поле	Биты	Тип	Описание
CCyIO	15-0	Чтение Запись Аппаратное влияние	Выходы канала y. Альтернативные выводы порта.

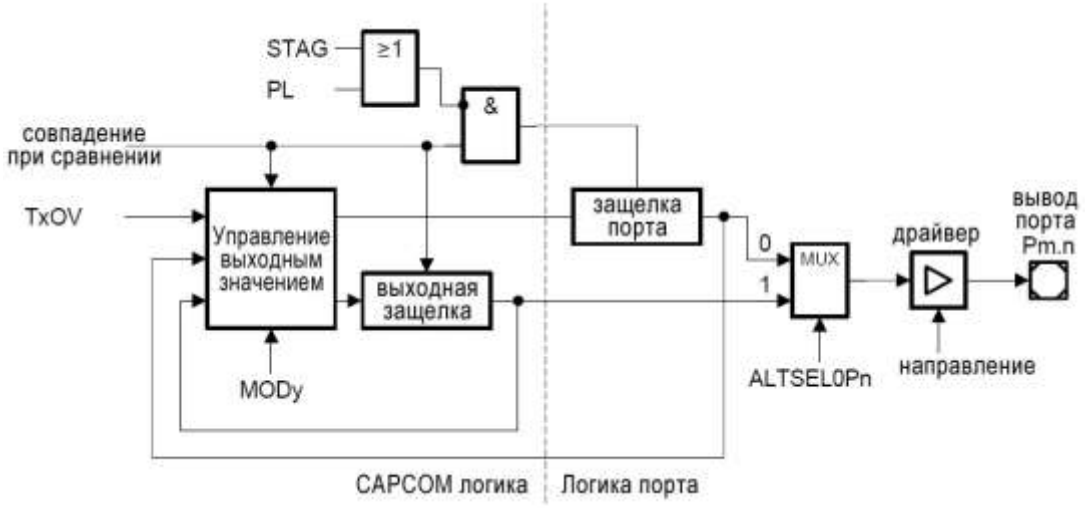


Рисунок 17.25 – Схема блока «Управление режимом и выводом»

Выходные сигналы могут быть высокого (1) и низкого (0) уровня. Блок контроля выходного значения определяет новый уровень сигнала в зависимости от результатов сравнения, сигнал переполнения таймера и текущее состояние портовой и выходной защелок. Для функции переключения вывода (в режиме 1) состояние выходной защелки читается, инвертируется и записывается снова.

Соответствующие выводы, управляющие сигналами портовой защелки или выходными сигналами, выбираются регистром ALTSEL.

Примечание – Если портовая защелка программируется в момент аппаратного переключения, программная запись имеет приоритет. В этом случае аппаратного переключения не происходит.

17.4 Режим однократного срабатывания

Если приложение требует, чтобы только одно событие подлежало сравнению (в пределах настоящего периода таймера), операция однократного срабатывания помогает упростить программирование и исключить необходимость в быстром отклике на событие.

Чтобы произвести однократное срабатывание надлежащим образом, надо программно запретить режим сравнения или записать новое значение, которое должно находиться вне диапазона счета таймера в регистре сравнения, после возникновения

запрограммированного совпадения. Таким образом, для выполнения этой операции, обычно требуется подпрограмма обработки прерываний. Время отклика на прерывание может играть существенную роль, если период таймера очень короткий – операция запрета должна завершиться до того, как таймер досчитает повторно до значения.

Операция однократного срабатывания исключает необходимость программы реагировать на первое совпадение. Завершение операции может произойти до события, и после наступления события никаких действий не потребуется. Аппаратная часть позаботится о том, чтобы сгенерировать только одно событие и затем запретить все дальнейшие совпадения.

Данная опция программируется посредством регистра однократного срабатывания CCx_SEM и регистра разрешения CCx_SEE. В каждом из этих двух регистров, для каждого регистра CCy, выделено по одному биту. Формат регистра CC1_SEM представлен на рисунке 17.26, регистра CC2_SEM – на рисунке 17.27, функциональное назначение полей этих регистров – в таблице 17.8. Формат регистра CC1_SEE представлен на рисунке 17.28, регистра CC2_ SEE – на рисунке 17.29, функциональное назначение полей этих регистров – в таблице 17.9.

CC1_SEM
регистр режима однократного срабатывания

SFR ($FE2C_H$ /16_H) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEM 15	SEM 14	SEM 13	SEM 12	SEM 11	SEM 10	SEM 9	SEM 8	SEM 7	SEM 6	SEM 5	SEM 4	SEM 3	SEM 2	SEM 1	SEM 0
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 17.26 – Формат регистра CC1_SEM

CC2_SEM
регистр режима однократного срабатывания

SFR ($FE28_H$ /14_H) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEM 15	SEM 14	SEM 13	SEM 12	SEM 11	SEM 10	SEM 9	SEM 8	SEM 7	SEM 6	SEM 5	SEM 4	SEM 3	SEM 2	SEM 1	SEM 0
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 17.27 – Формат регистра CC2_SEM

Таблица 17.8 – Функциональное назначение полей регистров CC1_SEM и CC2_SEM

Поле	Биты	Тип	Описание
SEMy	15-0	Чтение Запись	Включение режима однократного срабатывания: 0 Режим однократного срабатывания выключен для канала y. 1 Режим однократного срабатывания включен для канала y.

CC1_SEE
регистр однократного срабатывания

SFR ($FE2E_H/17_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEE 15	SEE 14	SEE1 3	SEE 12	SEE 11	SEE 10	SEE 9	SEE 8	SEE 7	SEE 6	SEE 5	SEE 4	SEE 3	SEE 2	SEE 1	SEE 0
з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап

Рисунок 17.28 – Формат регистра CC1_SEE

CC2_SEE
регистр однократного срабатывания

SFR ($FE2A_H/15_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEE 15	SEE 14	SEE1 3	SEE 12	SEE 11	SEE 10	SEE 9	SEE 8	SEE 7	SEE 6	SEE 5	SEE 4	SEE 3	SEE 2	SEE 1	SEE 0
з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап

Рисунок 17.29 – Формат регистра CC2_SEE

Таблица 17.9 – Функциональное назначение полей регистров CC1_SEE и CC2_SEE

Поле	Биты	Тип	Описание
SEEy	15-0	Чтение Запись	Запуск ожидания события: 0 Ожидание события выключено для канала y. 1 Ожидание события включено для канала y.

Для задания режима однократного срабатывания для регистра CCy, сначала необходимо запрограммировать желаемую операцию сравнения и сравниваемое значение, и только после этого установить соответствующий бит в регистре CCx_SEM для разрешения режима. И, наконец, установить соответствующий бит события в регистре CCx_SEE.

Примечание – Если режим однократного срабатывания для выбранного канала не включен (соответствующий бит в регистре CCx_SEM равен «0»), запуск ожидания события для этого канала невозможен.

Когда возникает запрограммированное совпадение, имеют место все операции выбранного режима сравнения, и аппаратно автоматически запрещаются все последующие совпадения и сбрасывается в «0» бит ожидания события в регистре CCx_SEE. Пока этот бит сброшен, любые операции сравнения запрещены. Для разрешения бит следует установить.

17.5 Операции в ступенчатом и неступенчатом режимах

CAPCOM может работать в двух основных режимах: ступенчатом и неступенчатом. Выбор режима осуществляется через регистр CC1_IOC/CC2_IOC.

Формат регистра CC1_IOC представлен на исунке 17.30, CC2_IOC – на исунке 17.31, функциональное назначение полей регистров CC1_IOC и CC2_IOC представлено в таблице 17.10.

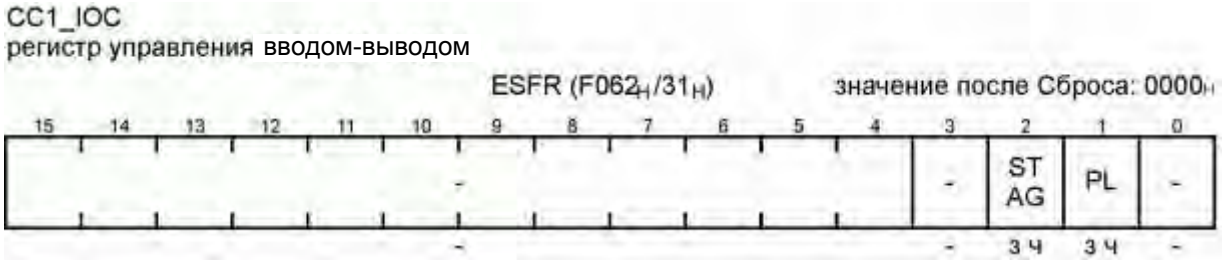


Рисунок 17.30 – Формат регистра CC1_IOC

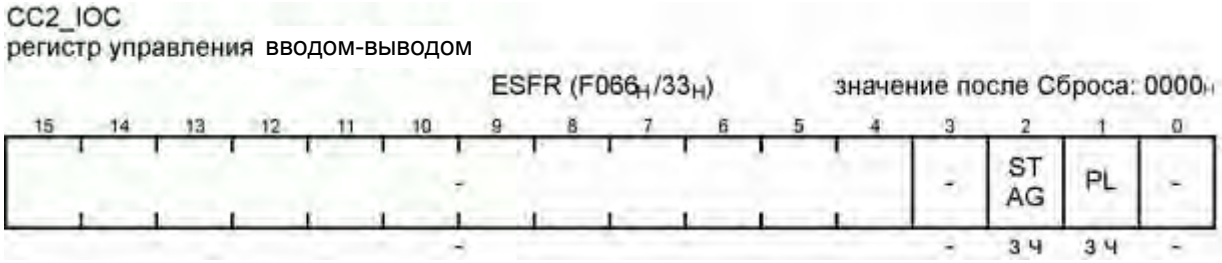


Рисунок 17.31 – Формат регистра CC2_IOC

Таблица 17.10 – Функциональное назначение полей регистров CC1_IOC и CC2_IOC

Поле	Биты	Тип	Описание
STAG	2	Чтение Запись	Включение ступенчатого режима: 0 CAPCOM работает в ступенчатом режиме. 1 CAPCOM работает в неступенчатом режиме.
PL	1	Чтение Запись	Управление блокирования порта: 0 Сигналы сравнения проходят на выводы порта. 1 Проход сигналов сравнения на выводы порта заблокирован.

Примечание – Всякий раз, когда неступенчатый режим активирован STAG = 1_B или блокирование порта включено PL = 1_B, выходной регистр порта не меняет своего значения.

В ступенчатом режиме цикл блока CAPCOM состоит из восьми тактов. При этом выходы различных регистров переключаются не одновременно, а ступенчато через определенные интервалы времени. Такой способ позволяет уменьшить шум и скачки напряжения на выходах.

В неступенчатом режиме цикл блока CAPCOM равен одному такту. При этом все выходы регистров переключаются одновременно, что увеличивает скорость работы и разрешение блока CAPCOM.

Ступенчатый режим

На рисунке 17.32 отражена работа в ступенчатом режиме. В данном примере все регистры CCy программируются для режима 3.

Регистры CC0, CC1 и CC2 программируются на сравниваемое значение $FFFE_H$. Когда таймер досчитывает до $FFFE_H$, компаратор обнаруживает совпадение для всех регистров. Выход CC0IO регистра CC0 переключается в «1» за один цикл после появления совпадения. В то же время, выходы CC1IO и CC2IO не переключаются в этот же момент, а по очереди за два последующих цикла. Такой способ переключения является ступенчатым и распространяется на все регистры до CC7, включительно. Как видно, регистр CC7

переключится семью циклами позже регистра CC1. В примере сравниваемое значение для регистра CC7 равно $FFFD_H$. Значит, выход переключится в последний такт тактовой последовательности блока CAPCOM, в который таймер досчитает до $FFFD_H$.

Когда таймер переполняется, все выходы сбрасываются в 0 (в режиме 3).

Взглянув на регистры с CC8 до CC15, можно заметить, что их выходы переключаются параллельно соответствующим выходам регистров с CC0 до CC7. Фактически ступенчатое переключение происходит параллельно для верхнего и нижнего банков регистров. Таким образом, это гарантирует, что оба сравниваемых сигнала регистровой пары в двухрегистровом режиме будут обработаны одновременно.

В ступенчатом режиме возможно прямое переключение портовой защелки. Это дает возможность использования альтернативных функций соответствующих выводов порта для вывода сравниваемых сигналов.

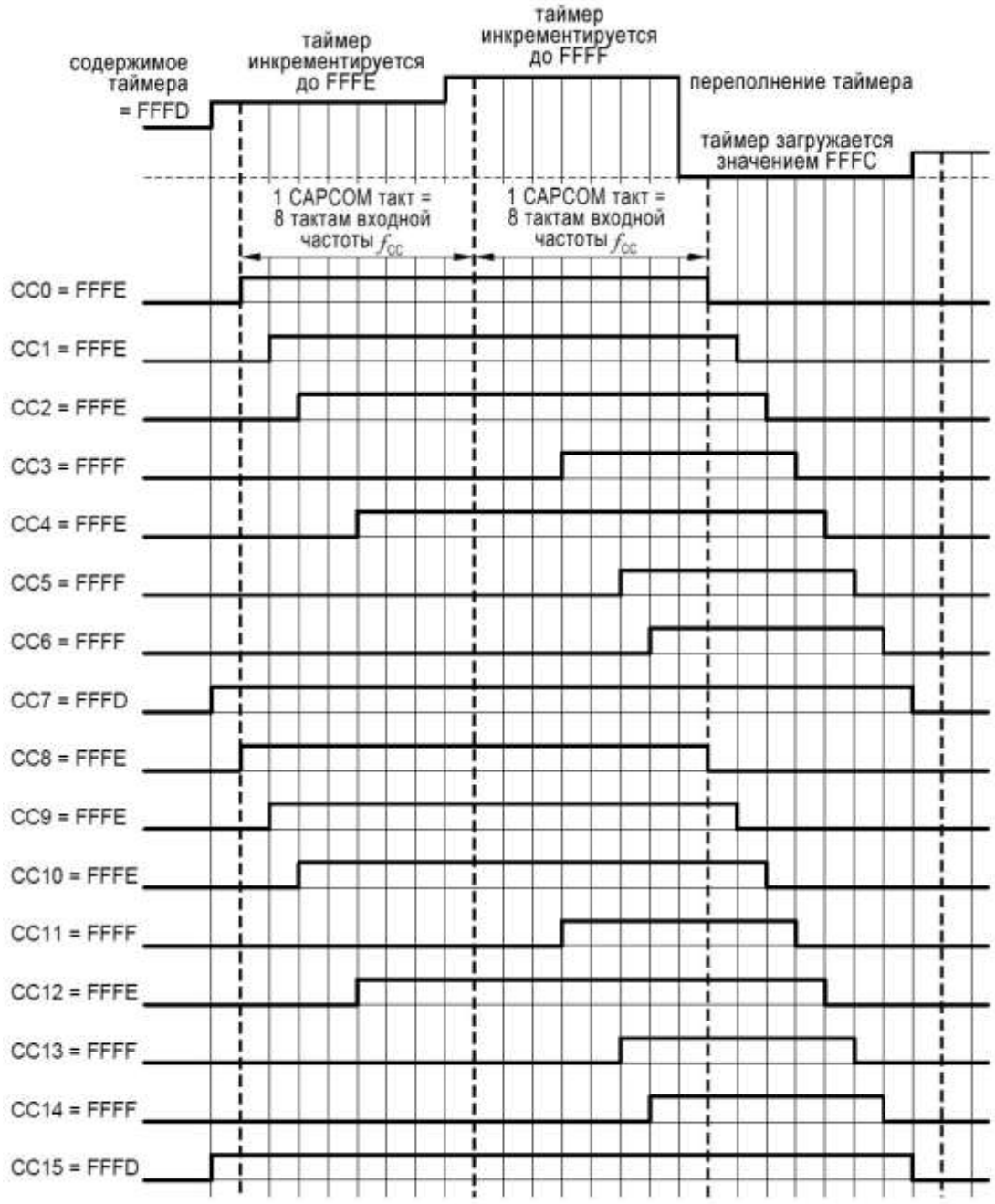


Рисунок 17.32 – Ступенчатый режим

Неступенчатый режим

Неступенчатый режим, см. рисунок 17.33, используется для достижения максимальной скорости и разрешения при работе CAPCOM.

Инкрементирование таймера и сравнение его обновленного содержимого с регистром сравнения происходит за один цикл. Соответствующий выходной сигнал переключается в следующем цикле (параллельно со следующим переключением таймера).

Надо иметь в виду, что переполнение таймера занимает дополнительный цикл для переключения выходного сигнала

Примечание – В неступенчатом режиме прямое переключение портовой защелки запрещено.

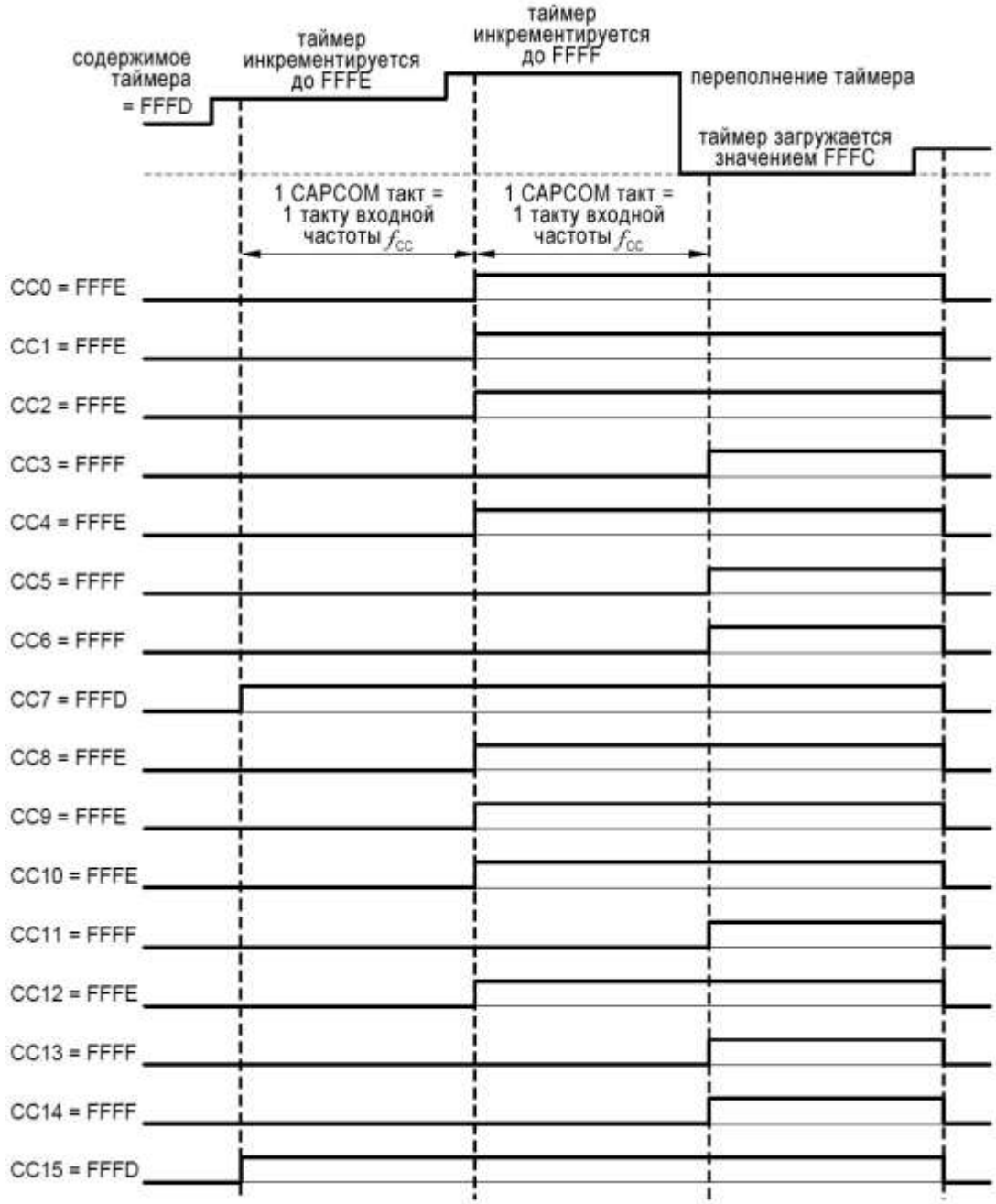


Рисунок 17.33 – Неступенчатый режим

17.6 Требования к внешнему входному сигналу

Внешние входные сигналы блока CAPCOM отслеживаются (замеряются) CAPCOM-логикой, работающей на частоте модуля в двух основных режимах (в ступенчатом и неступенчатом). Для уверенности в том, что уровень сигнала определен корректно, высокий или низкий уровень должен оставаться активным как минимум один период выборки (замера).

Длительность периода выборки равняется одному циклу частоты модуля в неступенчатом режиме, и восьми циклам в ступенчатом режиме. Для распознавания переключения (изменения уровня) сигнала, необходимы две выборки (замера). Если уровень первого замера отличается от уровня второго замера, то это определяется как переключение. Таким образом, для отслеживания внешнего входного сигнала необходимы, минимум, два периода замера. Отсюда следует, что максимальная частота входного сигнала не должна превышать половины частоты модуля в неступенчатом режиме, и 1/16 частоты модуля в ступенчатом режиме, см. таблицу 17.11.

Таблица 17.11 – Требования и ограничения, накладываемые на входной сигнал

Параметр входного сигнала	Неступенчатый режим	Ступенчатый режим
Максимальная частота	$f_{CC} / 2$	$f_{CC} / 16$
Минимальная длительность	$1 / f_{CC}$	$8 / f_{CC}$

При использовании внешнего входного сигнала как сигнала подсчета или сравнения, соответствующий вывод порта должен быть сконфигурирован как вход.

Примечание – Например, для тестирования вывода на захват или сравнение последний может быть сконфигурирован как выход. Соответствующий сигнал может контролироваться программно либо с помощью периферии.

Для вывода сигнала соответствующий вывод порта должен быть сконфигурирован как выход. Сравниваемый выходной сигнал может напрямую переключать портовую защелку, или же выход защелки CCx_OUT выполняет альтернативную функцию порта.

17.7 Интерфейс блоков CAPCOM

Блоки модуля CAPCOM: CAPCOM1, см. рисунок 17.34, и CAPCOM2, см. рисунок 17.35, соединяются со своей периферией по-разному.

Внутренние соединения

Сигналы T6OUF и GPT2 перегрузки/опустошения таймера T6 соединяются с блоками CAPCOM, являясь оптимальными источниками частоты для таймеров CAPCOM.

18 линий прерывания каждого CAPCOM-блока соединены с блоком контроля прерываний.

Примечание – Входные линии порта P2, соединяемые с блоком CAPCOM1, также могут использоваться как независимые входы внешних прерываний.

Внешние соединения

Сигналы захвата/сравнения обоих блоков CAPCOM соединяются с портами ввода-вывода контроллера. В зависимости от выбранной директивы, порты осуществляют захват внешнего сигнала или выводят сигнал сравнения.

Примечание – Сигнал сравнения может также быть получен извне. В этом случае, например, защелкивание может быть сгенерировано программно.

Таймеры T0 и T7 могут тактироваться внешним сигналом. Входной сигнал T7IN таймера блока CAPCOM2 использует вывод порта совместно с выводом ввода-вывода CC15IO блока CAPCOM1. Операции в обоих блоках CAPCOM могут быть объединены:

- вывод сравнения блока CAPCOM1 может использоваться для тактирования таймера T7 блока CAPCOM2;
- сигнал счетного входа блока CAPCOM2 может быть записан блоком CAPCOM1 посредством функции сравнения.

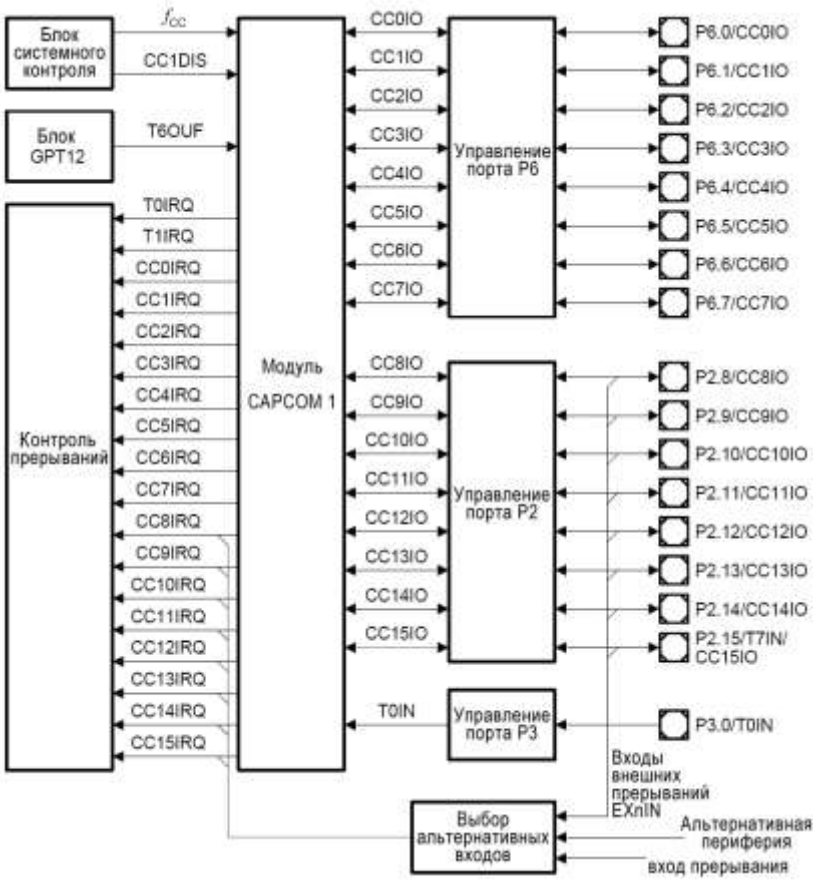


Рисунок 17.34 – Интерфейс блока CAPCOM1

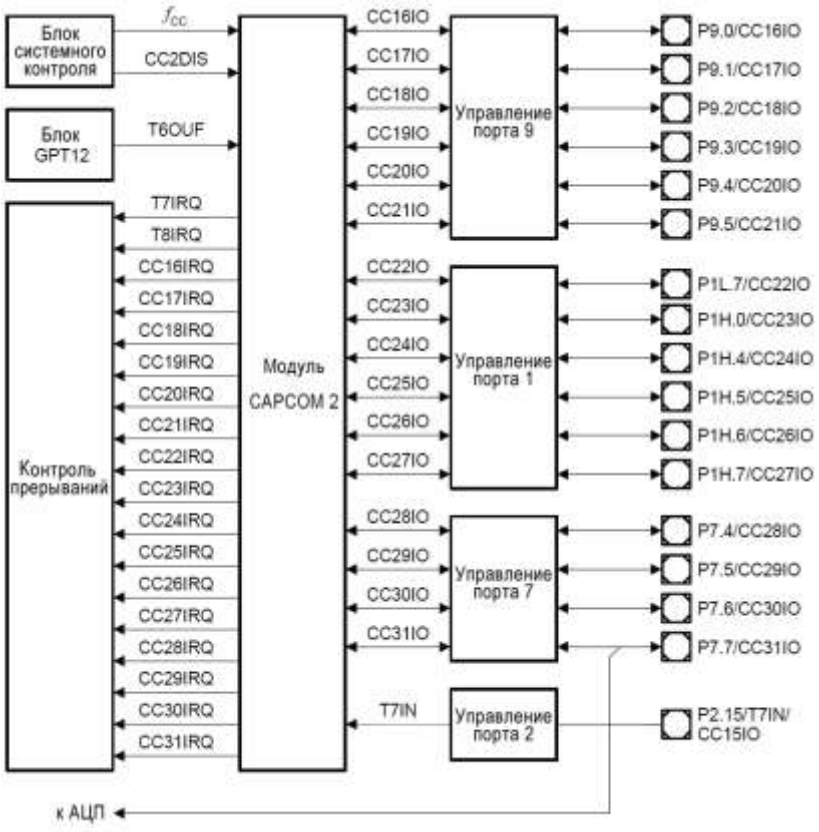


Рисунок 17.35 – Интерфейс блока CAPCOM2

18 Блок захвата/сравнения CAPCOM6

Блок CAPCOM6, см. рисунок 18.1, состоит из блока таймера T12 с тремя каналами захвата/сравнения и блока таймера T13 с одним каналом сравнения. Каналы таймера T12 могут независимо генерировать ШИМ сигналы или считывать значения триггеров захвата.

Специальные режимы работы позволяют также осуществлять управление бесщеточными ДПТ с датчиками Холла или контролем обратной ЭДС.

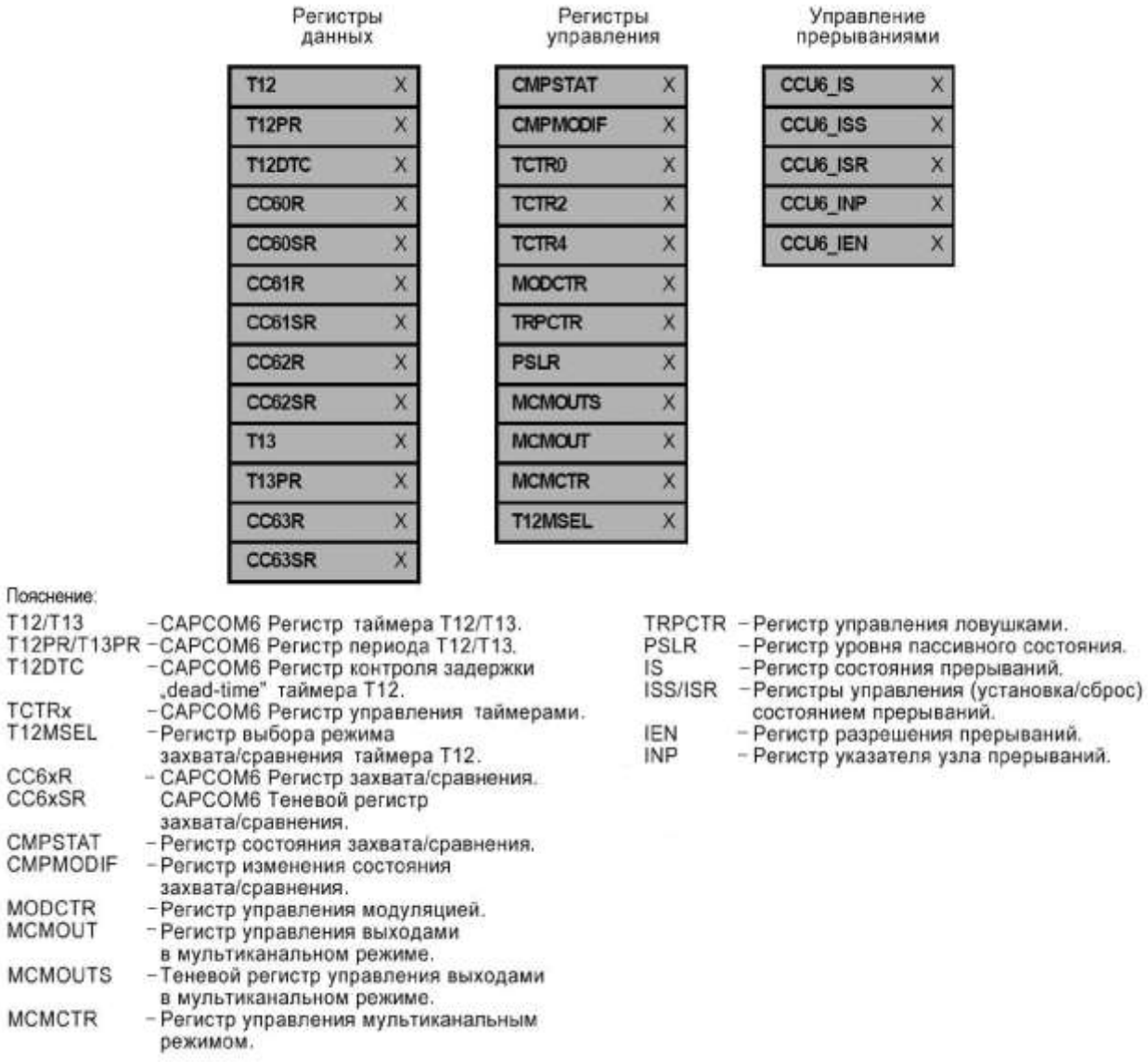


Рисунок 18.1 – SFR регистры, соответствующие блоку CAPCOM6

18.1 Особенности блока CAPCOM6

Структурная схема блока CAPCOM6 представлена на рисунке 18.2.

Особенности блока таймера T12:

- 3 канала захвата/сравнения;
- возможность генерирования трехфазного ШИМ (шесть независимых выходов);
- разрешение 16 бит. Максимальная частота работы счетчика равна внешней тактовой частоте;
- временная задержка «dead-time» для каждого канала (например, для предотвращения короткого замыкания в силовой части, если контроллер работает с двигателем);

- синхронное обновление содержимого регистров T12/T13;
- возможность генерирования пилообразного и симметричного ШИМ;
- поддержка режима однократного срабатывания;
- большое количество источников прерываний;
- режим блокирования.

Особенности блока таймера T13:

- один независимый канал сравнения с одним выходом;
- разрешение 16 бит. Максимальная частота работы счетчика равна внешней тактовой частоте;
- возможность синхронизации с таймером T12;
- генерирование прерываний при совпадениях по периоду и при совпадениях по значению;
- поддержка режима однократного срабатывания.

Дополнительные возможности:

- осуществление коммутации блока с бесщеточным ДПТ;
- отслеживание положения ротора двигателя через датчики Холла;
- фильтр шумов при работе с датчиками Холла;
- автоматическое измерение скорости вращения для блока коммутации;
- встроенный аппарат обработки ошибок;
- скоростное экстренное прекращение работы без задействования ЦПУ посредством внешнего сигнала CTRAP#;
- режимы контроля для многоканальных асинхронных двигателей;
- уровни выходных сигналов могут быть адаптированы под силовую часть.

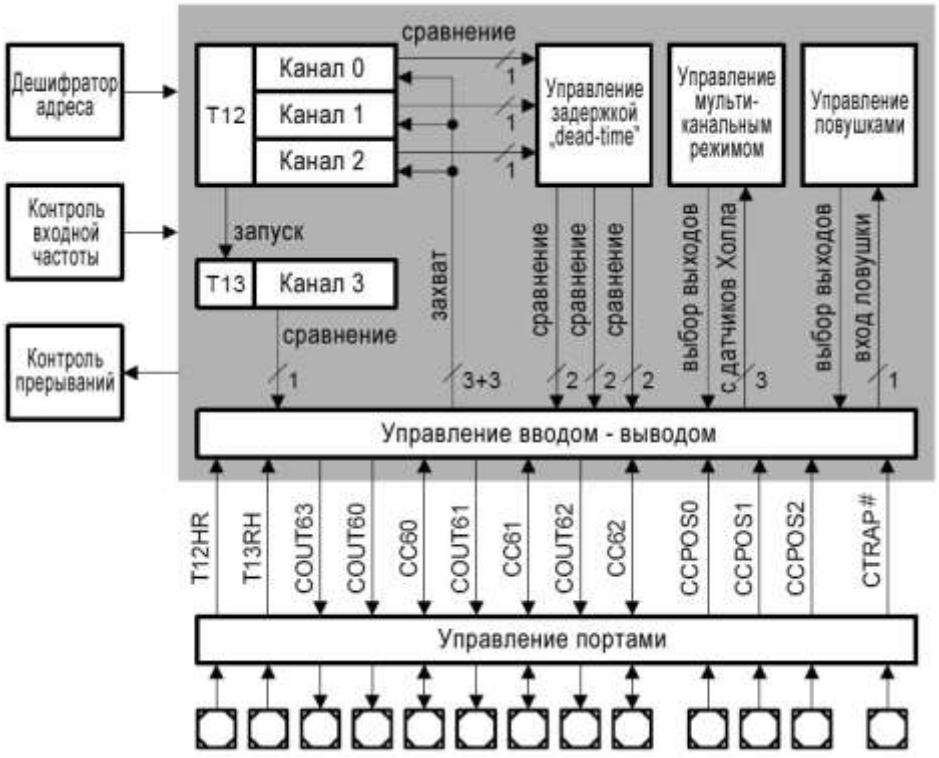


Рисунок 18.2 – Структурная схема блока CAPCOM6

Три канала таймера T12, см. рисунок 18.2, могут работать как в режимах захвата, так и в режиме сравнения. Режимы могут комбинироваться. Канал таймера T13 может работать только в режиме сравнения. Блок управления мультиканальным режимом имеет возможность смешивания сигналов, модулируемых таймерами T12 и T13.

18.2 Блок таймера T12

Блок таймера T12, см. рисунок 18.3, является основным генератором трехфазной ШИМ. 16-битный счетчик соединен с регистрами трех каналов через компараторы, которые генерируют сигнал, когда содержимое счетчика совпадает с содержимым одного из регистров.

Помимо генерирования трехфазной ШИМ, каналы блока таймера T12 имеют возможность захвата, а так же возможность включения временной задержки «dead-time» и блокирования.

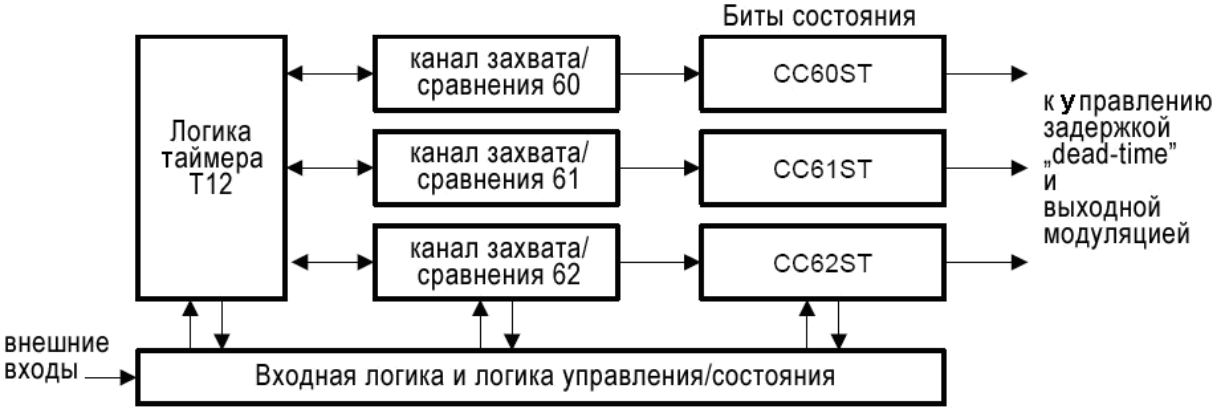


Рисунок 18.3 – Блок таймера T12

На рисунке 18.4 показана подробная схема таймера T12.

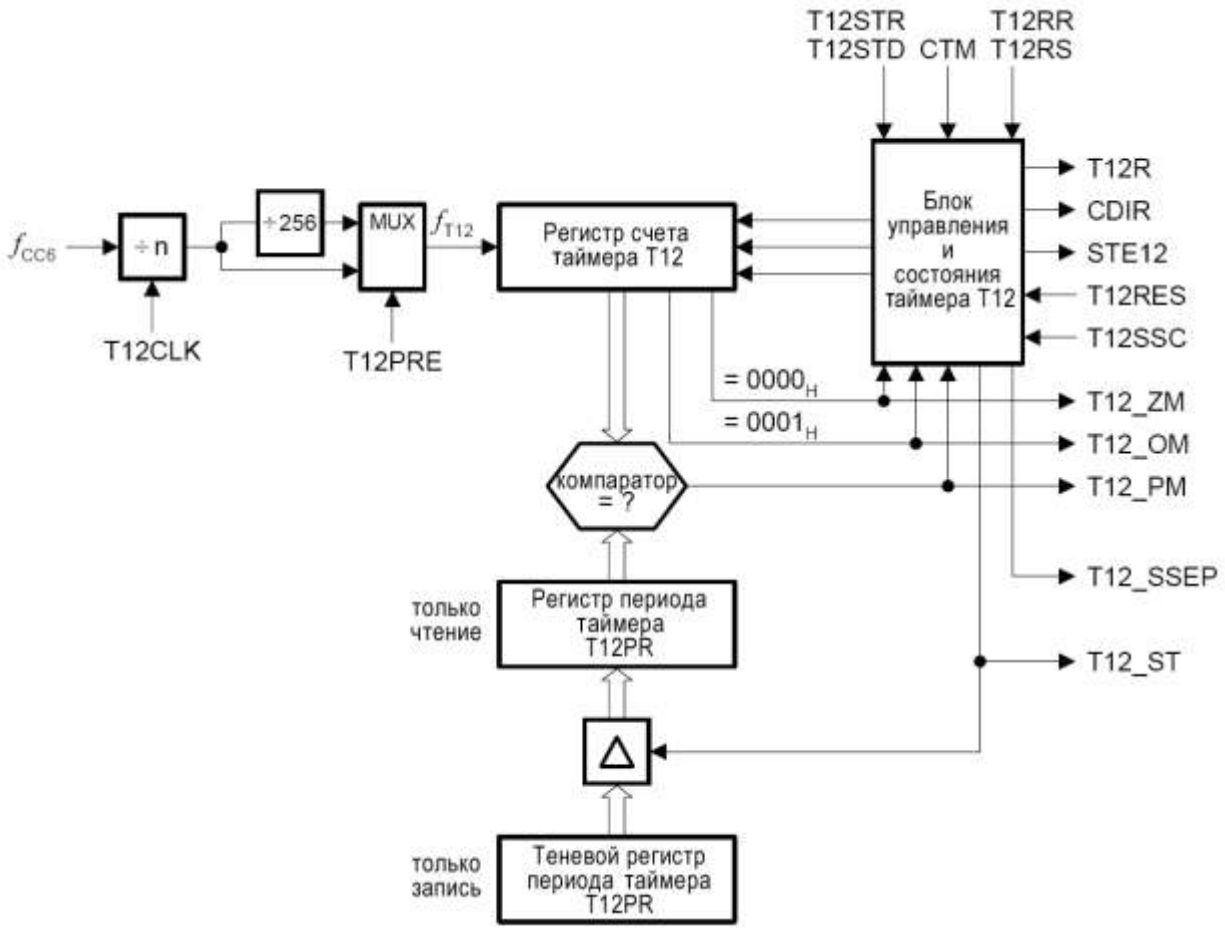


Рисунок 18.4 – Компаратор периода и логика таймера T12

Таймер T12 получает входную тактовую частоту f_{T12} из частоты f_{CC6} посредством программируемого делителя и делителя 1/256. Деление частоты контролируется посредством битовых полей T12CLK и T12PRE. Таймер T12 может считать как «вверх» (инкрементирование), так и «вниз» (декрементирование), в зависимости выбранного режима, см. рисунок 18.5, таблица 18.1. Направление счета показывает флаг CDIR.

Через компаратор таймер T12 соединяется с регистром периода T12PR. Этот регистр определяет максимальное значение (значение периода), до которого считает таймер. В режиме «пилы», по достижении значения периода, таймер сбрасывается в ноль. В «симметричном» режиме по достижении значения периода, направление счета таймера меняется на противоположное (включается декрементирование). Следует помнить, что в «симметричном» режиме значение счетчика таймера T12 превысит значение величины периода на один, прежде чем начнется обратный счет.

В обоих режимах при достижении таймером значения периода генерируется сигнал T12_PM (совпадение по периоду).

Регистр периода CCU6_T12 таймера T12PR, см. рисунок 18.6, таблицу 18.2, получает новое значение периода из теневого регистра периода, который записывается программно. Передача нового значения из теневого регистра в регистр периода T12PR активируется посредством сигнала теневой передачи – T12_ST. Генерирование этого сигнала зависит от режима работы и бита контроля STE12.

Наличие теневого регистра периода и других теневых регистров позволяет генерировать ШИМ сигнал и параллельно программно задавать новые параметры для работы блока CAPCOM6.

Работой таймера T12 (на аппаратном уровне) управляют: сигнал T12_OM совпадения содержимого счетчика таймера с единицей, т. е. со значением 0001_H, и сигнал T12_ZM совпадения с нулем.

Режим работы таймера T12 («пила» на рисунке 18.7 или «симметричный» на рисунке 18.8) выбирается битом CTM.

Бит управления однократным запуском T12SSC осуществляет автоматическую остановку таймера (после включения), по достижении значения периода.

Запуск и остановка таймера T12 управляется битом T12R. Бит программно может быть установлен или сброшен посредством парной установки битов T12RS и T12RR. Также бит T12R может быть сброшен аппаратно.

Теневая передача для регистров таймера T12 (T12_ST) активируется битом STE12. Этот бит программно может быть установлен или сброшен посредством парной установки битов T12STR и T12STD.

Регистр счета таймера T12 хранит текущее значение счета. Этот регистр может быть записан только во время, когда таймер T12 остановлен (в течение времени, когда счетчик таймера включен, запись не возможна). Регистр счета всегда может быть прочитан программно.

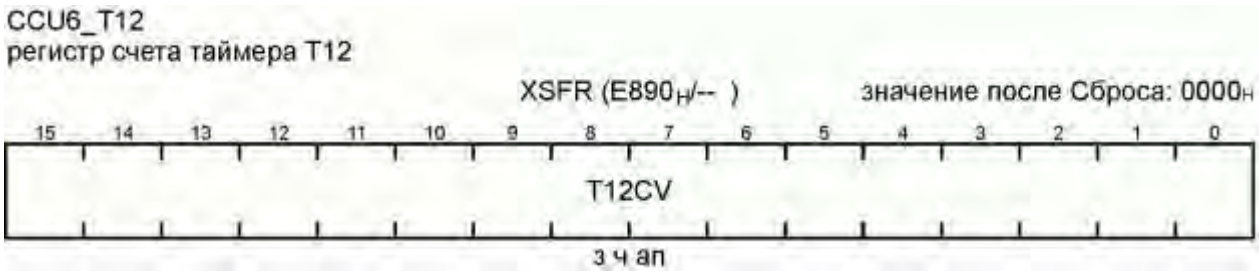


Рисунок 18.5 – Формат регистра CCU6_T12

Таблица 18.1 – Функциональное назначение поля регистра CCU6_T12

Поле	Биты	Тип	Описание
T12CV	15-0	Чтение Запись Аппаратное влияние	16-битное значение счетчика таймера T12

Регистр T12PR содержит значение периода таймера T12. Значение периода сравнивается с текущим значением счетчика таймера T12 и действия, производимые после, зависят от правил счета.

Рисунок 18.6 – Формат регистра CCU6_T12PR

Таблица 18.2 – Функциональное назначение поля регистра CCU6_T12PR

Поле	Биты	Тип	Описание
T12PV	15-0	Чтение Запись	Значение периода таймера T12. Когда значение счетчика таймера сравнивается со значением T12PV, возникает совпадение по периоду. После этого счетчик таймера сбрасывается в ноль в случае режима «пилы» или меняет направление счета (начинает считать «вниз») в случае «симметричного» режима.

Этот регистр имеет теневой регистр с таким же адресом. Запись в регистр периода таймера осуществляется при теневой передаче синхронно со счетом таймера, т. е. новое значение будет записано в момент очередного инкрементирования/декрементирования таймера T12.

Теневой регистр периода таймера доступен только для записи, в то время как основной регистр периода таймера – только для чтения.

Регистр периода таймера T12 всегда может быть прочитан программно.

Операции таймера T12

Входная частота f_{T12} таймера T12, см. таблицу 18.3 получается из частоты f_{CC6} посредством программируемого делителя и делителя 1/256.

Таблица 18.3 – Входная частота таймера T12

T12CLK	Входная частота таймера f_{T12}	
	Дополнительный делитель выключен (T12PRE = 0_B)	Дополнительный делитель включен (T12PRE = 1_B)
1	2	3
000_B	f_{CC6}	f_{CC6} / 256
001_B	f_{CC6} / 2	f_{CC6} / 512
010_B	f_{CC6} / 4	f_{CC6} / 1024

Окончание таблицы 18.3

1	2	3
011_B	f_{CC6} / 8	f_{CC6} / 2048
100_B	f_{CC6} / 16	f_{CC6} / 4096
101_B	f_{CC6} / 32	f_{CC6} / 8192
110_B	f_{CC6} / 64	f_{CC6} / 16384
111_B	f_{CC6} / 128	f_{CC6} / 32768

Период счета таймера определяется значением в регистре периода T12PR и режимом таймера.

В режиме «пилы» период равен:

T_{12PER} = «значение периода» + 1, в тактах таймера T12.

В «симметричном» режиме:

T_{12PER} = («значение периода» + 1) × 2, в тактах таймера T12.

Таймер T12 в режиме «пилы»

Режим «пила» для таймера T12 представлен на рисунке 18.7.

Если с приходом очередного такта f_{T12} возникает совпадение по периоду, счетчик сбрасывается в ноль.

Направление счета всегда «вверх» (инкрементирование), CDIR = 0_B.

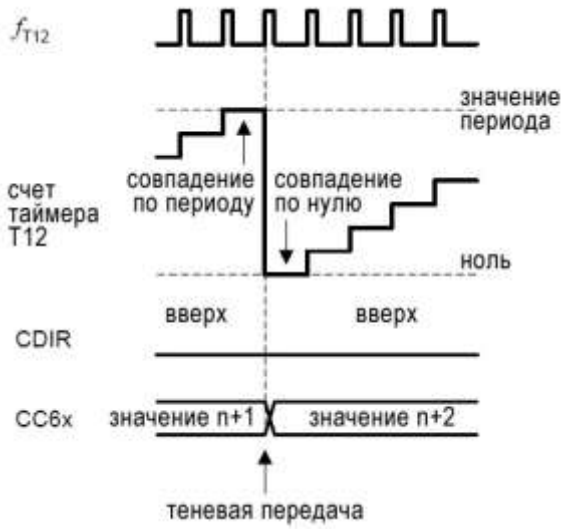


Рисунок 18.7 – Режим «пила» для таймера T12

Таймер T12 в «симметричном» режиме

«Симметричный» режим для таймера T12 представлен на рисунке 18.8.

Если с приходом очередного такта f_{T12} возникает совпадение по периоду при счете «вверх» (инкрементирование), направление счета устанавливается «вниз» (декрементирование) и CDIR принимает значение «1».

Если с приходом очередного такта f_{T12} при счете «вниз» возникает совпадение с единицей (счетчик = 0001_H), направление счета устанавливается «вверх» и CDIR принимает значение «0».

Бит CDIR является индикатором направления счета таймера T12 (если CDIR = 0_B, значит таймер считает «вверх»; если CDIR = 1_B, значит таймер считает «вниз»).

Примечание – Бит CDIR изменяется с каждым последующим тактом инкрементирования/декрементирования таймера после совпадения по периоду или по нулю. Таким образом, таймер продолжает считать в предыдущем направлении еще в течение одного такта до смены направления счета.

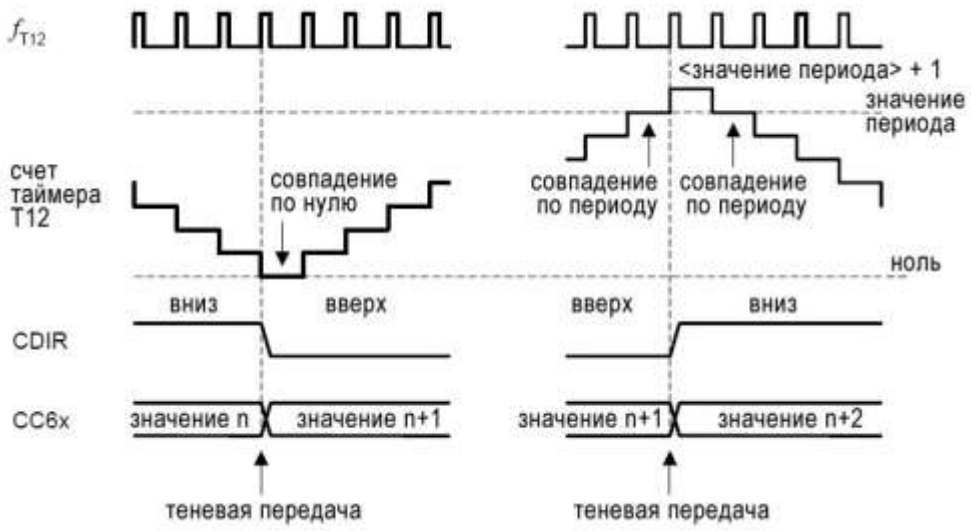


Рисунок 18.8 – «Симметричный» режим для таймера T12

Примечание – На рисунках 18.8 – 18.10, 18.13 – 18.16 показаны варианты изменения сигналов для случаев, когда частота переключения счетчика таймера максимальна и совпадает с f_{CC}, т. е. T12CLK = 000_B и T12PRE = 0_B.

Сигнал теневой передачи T12_ST

Структура дополнительных (теневых) регистров с сигналом теневой передачи включена в состав блока CAPCOM6 для синхронизации изменений (в том числе автоматически на аппаратном уровне) значений регистров с работой счетчиков таймеров T12 и T13 .

Генерация/запрет сигнала теневой передачи управляется битом STE12. Программно этот бит может быть установлен или сброшен посредством установки/сброса бита T12STR.

Запись «1» в бит T12STR (только для записи) устанавливает бит STE12, который в свою очередь активирует сигнал теневой передачи. После установки бита STE12, в случае если таймер запущен, аппаратная часть ожидает очередное инкрементирование/ декрементирование таймера T12 и синхронно с переключением счетчика таймера производит запись в регистры из соответствующих им теневых регистров, после чего бит T12STR автоматически очищается, что влечет за собой очистку бита STE12.

Помимо программной организации теневой передачи имеется возможность аппаратной организации. По умолчанию, в случае если бит T12STD равен «0», аппаратная часть будет генерировать сигнал теневой передачи (выставлением бита STE12) каждый раз, когда будет возникать совпадение по периоду (T12_PM) при счете таймера T12 «вверх» и совпадение с единицей (T12_OM) при счете таймера T12 «вниз». Запись «1» в бит T12STD отключит аппаратное генерирование теневой передачи (для включения следует очистить бит T12STD).

Примечания

1 В режиме работы с датчиками Холла MSEL6x = 1000_B, где x = 0, 1, 2, независимо от состояния бита T12STD, генерирование сигнала аппаратной теневой передачи, т. е. установка бита STE12, не происходит.

2 Программное генерирование теневой передачи (установка бита T12STR) возможно в любой момент времени, независимо от состояния бита T12STD.

В случае если таймер остановлен, запись в регистры происходит сразу после появления сигнала теневой передачи.

Контроль старта/останова и сброса таймера T12

Счетом таймера T12 управляет бит T12R. Счетчик таймера T12 активен только в случае, если бит T12R = 1_B. Программно этот бит может быть установлен посредством записи «1» в бит T12RS (только для записи) или сброшен посредством записи «1» в бит T12RR (только для записи),

Также T12R может быть сброшен аппаратно в режиме однократного срабатывания.

Очистить счетчик таймера T12 (также и во время, когда таймер считает) можно программно, записью «1» в бит T12RES (только для записи). Установка этого бита, обнуляет счетчик (в него записывается значение 0000_H), но не оказывает другого влияния, например, не останавливает его.

Режим однократного срабатывания

В режиме однократного срабатывания, см. рисунок 18.9, бит запуска T12R находится под программным и аппаратным влиянием.

Режим активируется посредством бита T12SSC. Следует помнить, что запись бита T12SSC возможна, только если таймер T12 остановлен.

Если бит T12SSC = 1_B, то счетчик таймера T12 после очередного запуска остановится, отсчитав один период таймера. В режиме «пилы» это произойдет тогда, когда таймер обнулится по достижении заданного значения периода. В «симметричном» режиме период закончится, когда таймер досчитает «вниз» до 0, см. рисунок 18.10.

Для выхода из режима необходимо записать «0» в бит T12SSC.

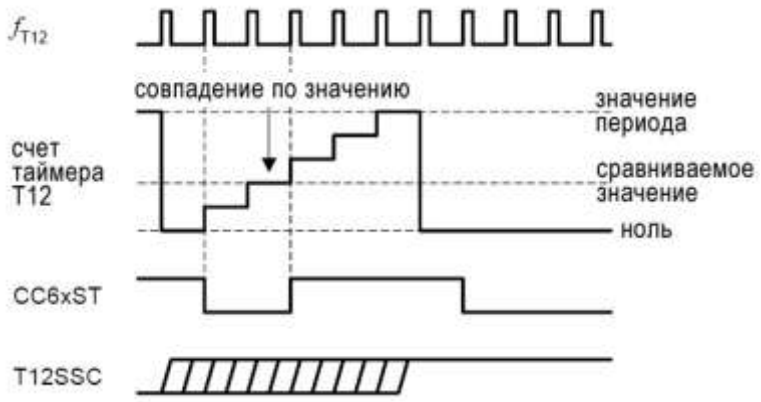


Рисунок 18.9 – Однократное срабатывание в режиме «пилы»

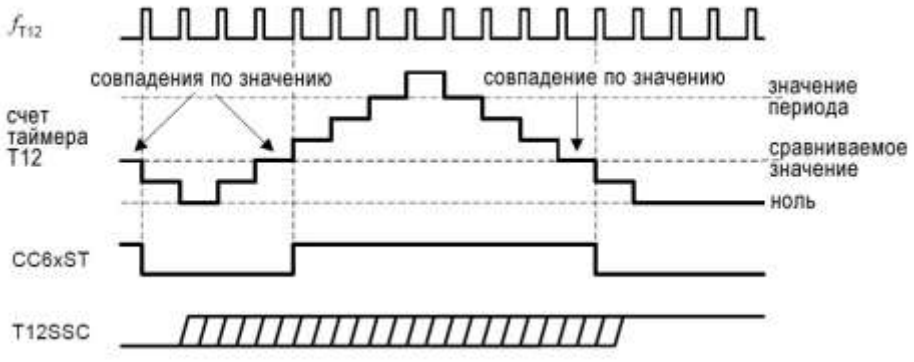


Рисунок 18.10 – Однократное срабатывание в «симметричном» режиме

Режимы сравнения T12

С таймером T12 связаны три независимых канала захвата/сравнения, которые могут выполнять операции захвата или сравнения по отношению к содержимому счетчика таймера T12.

В режиме сравнения, см. рисунок 18.11, три канала могут работать как независимо друг от друга, так и совместно для генерирования трехфазного ШИМ.

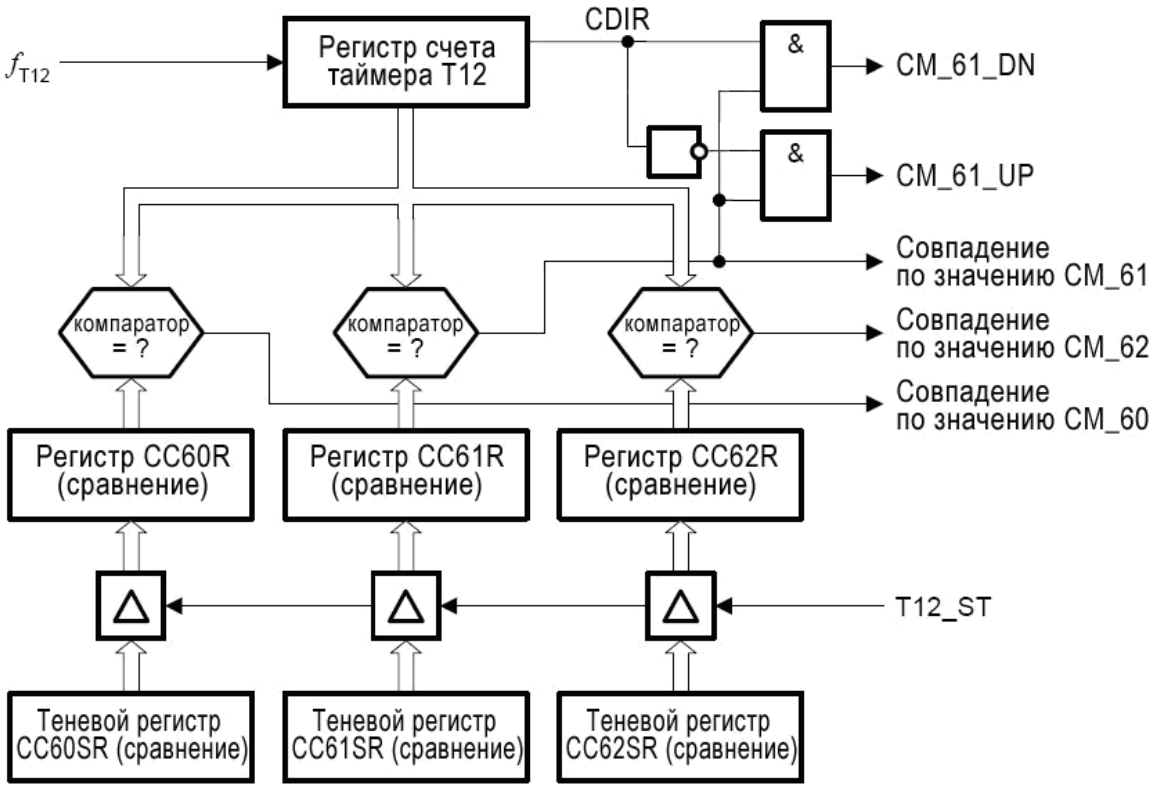


Рисунок 18.11 – Компараторы каналов таймера T12

Каждый канал соединяется с регистром счета таймера T12 посредством его индивидуального компаратора сравнения, который генерирует сигнал совпадения, когда содержимое счетчика совпадает с содержимым соответствующего регистра сравнения.

Каждый канал состоит из компаратора и двух регистров – регистра сравнения CC6xR, содержимое которого загружается в компаратор, и соответствующего ему теневого регистра CC6xSR, который записывается программно и передает значение в регистр сравнения по сигналу теневой передачи T12_ST.

С каждым каналом связан бит состояния CC6xST, который отражает состояние выходной линии канала, см. рисунок 18.12.

Входы логики управления установкой/сбросом битов состояния CC6xST следующие:

- направление счета таймера CDIR;
- бит запуска таймера T12R;
- сигнал совпадения счетчика таймера с нулем T12_ZM;
- сигнал однократного срабатывания T12_SSEP;
- индивидуальные сигналы совпадения по значению в режимах сравнения CM_6x, а также биты контроля режима MSEL6x.

Помимо этого, каждый бит состояния может быть программно установлен установкой бита MCC6xS или сброшен установкой бита MCC6xR.

Примечание – В режиме датчиков Холла подключаются дополнительные входы.

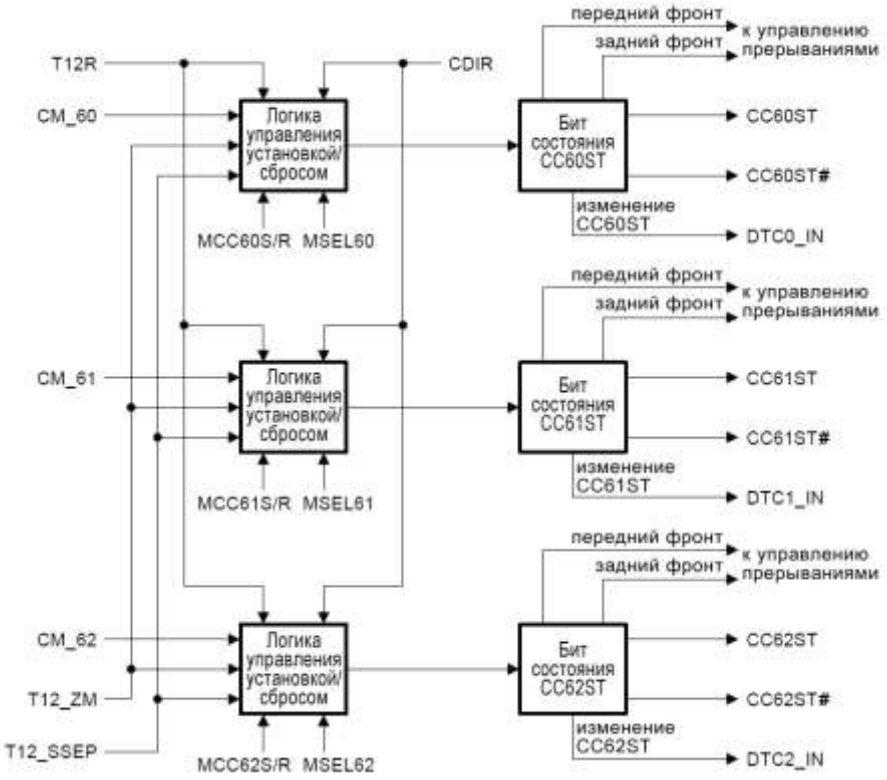


Рисунок 18.12 – Биты состояния

Аппаратное изменение бита состояния CC6xST возможно только во время работы таймера T12 (T12R = 1_B). Поведение бита состояния в режиме «пила» и «симметричном» режиме показано на рисунках 18.13 и 18.14.

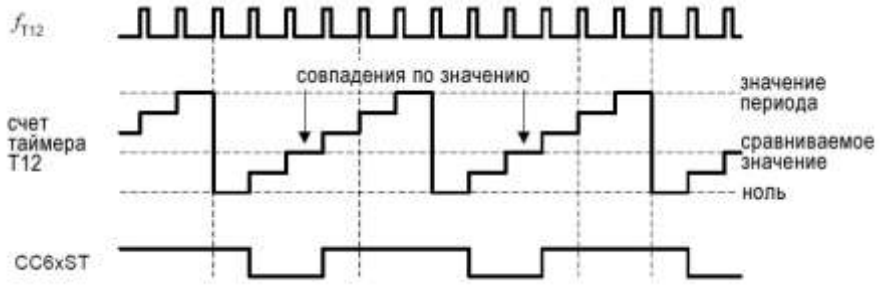


Рисунок 18.13 – Операция сравнения, режим «пила»

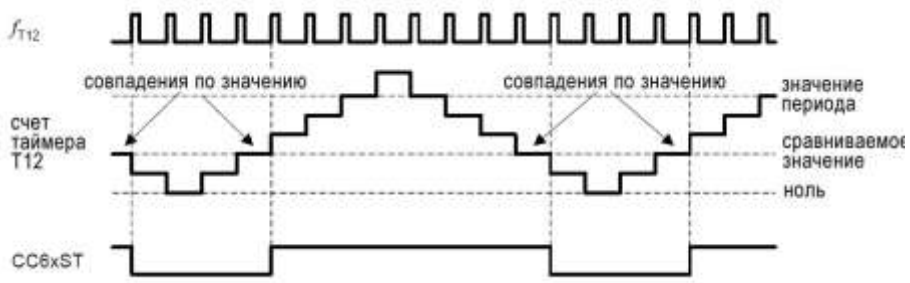


Рисунок 18.14 – Операция сравнения, «симметричный» режим

Бит CC6xST переключается в «1» если:

- при очередном переключении счетчика таймера возникает совпадение по значению, когда счетчик инкрементируется (считает «вверх»);
- при сбросе счетчика таймера в ноль по переполнению («пила») возникает совпадение по значению, равному 0000_H;
- при переключении направления счета счетчика таймера («симметричный») из «вниз» в «вверх» возникает совпадение по значению, равному 0000_H.

Бит CC6xST переключается в «0» если:

- при очередном переключении счетчика таймера возникает совпадение по значению, когда счетчик декрементируется (считает «вниз»);
- при сбросе счетчика таймера в ноль по переполнению («пила») не возникает совпадение по значению, равному 0000_H.

На рисунке 18.15 приведено несколько примеров: здесь предполагается изменение некоторых сравниваемых значений в течение работы таймера. Эти изменения являются следствием программной загрузки теневого регистра CC6xSR. Значения предаются в регистр сравнения CC6xR посредством сигнала аппаратной теневой передачи T12_ST, который считается разрешенным (бит T12STD = 0_B).

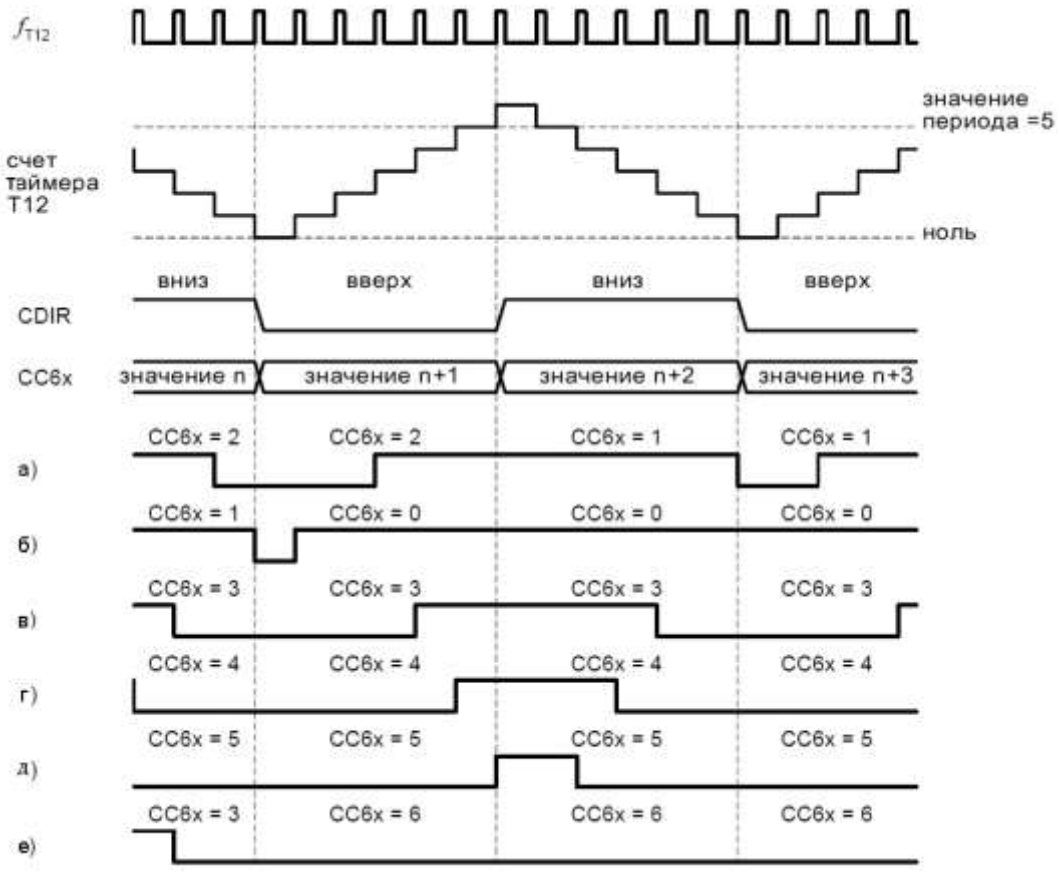


Рисунок 18.15 – Примеры переключения сигналов в режиме сравнения

Пояснения к рисунку 18.15

Пример б) иллюстрирует передачу при 100 % заполнении. Используется сравниваемое значение 0001_H, меняющееся затем на 0000_H. Следует заметить, что отрицательный импульс с длиной в один такт появляется в период, когда вступает в силу новое значение 0000_H. Этот импульс возникает из предыдущего значения 0001_H. В следующий период таймера бит состояния CCxST остается в «1», поддерживая постоянный сигнал. В данном случае имеет место правило «при переключении

направления счета счетчика таймера («симметричный») из «вниз» в «вверх», возникает совпадение по нулю.

Пример e) показывает передачу при 0 % заполнении. Новое сравниваемое значение устанавливается в значение, равное «значение периода» плюс 1, и бит состояния CC6xST остается равным «0».

На рисунке 18.16 иллюстрованы сигналы трех каналов. С применением задержки «dead-time» и контроля выходной модуляции может быть сгенерирован достаточно хороший ШИМ сигнал.

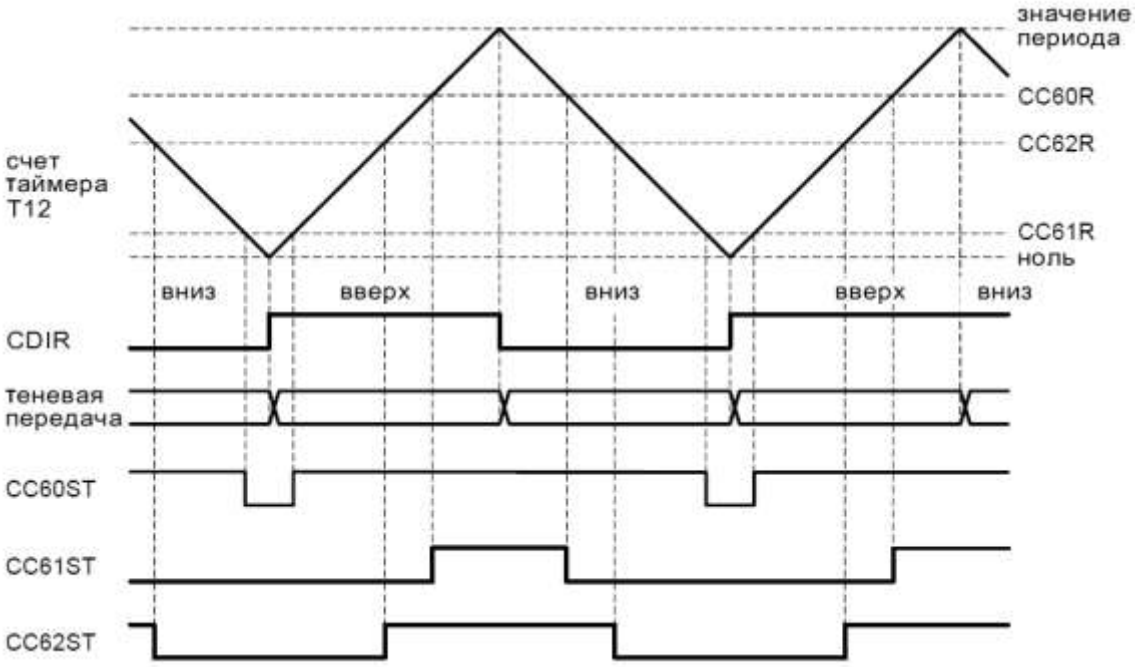


Рисунок 18.16 – Сигналы сравнения трех каналов

Вывод сигнала в режиме сравнения

На рисунке 18.17 отражен в общем виде путь сигнала от бита состояния канала до выхода. Как видно из рисунка, пользователь имеет широкие возможности управления выходным сигналом, который формируется битом состояния CC6xST.

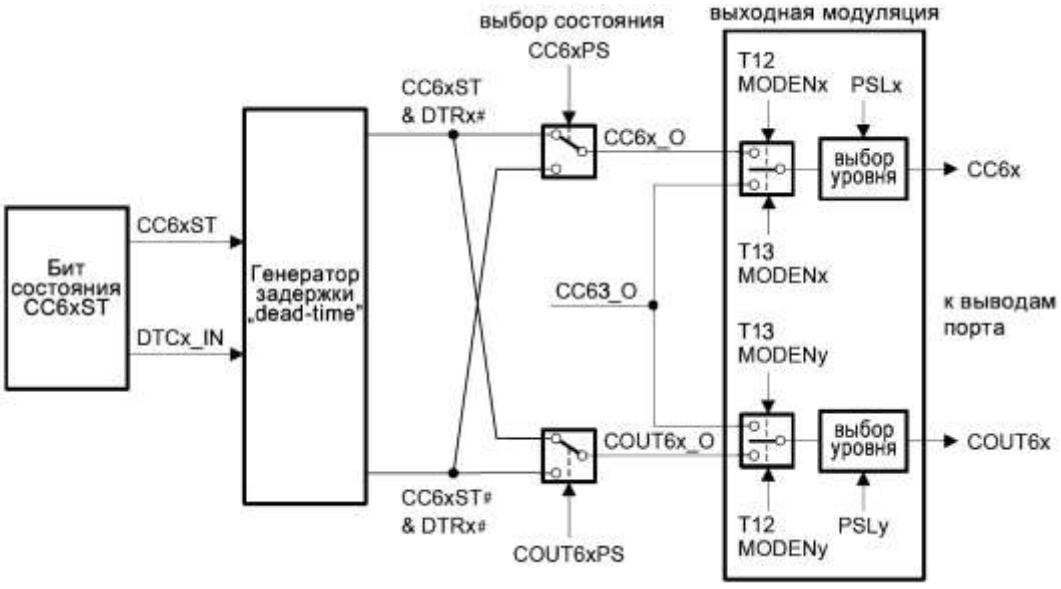


Рисунок 18.17 – Вывод сигнала в режиме сравнения

18.3 Регистры захвата/сравнения

Регистры CC6xR являются регистрами, которые могут быть только прочитаны программно. Запись в эти регистры осуществляется через соответствующие им теневые регистры CC6xSR, которые могут программно читаться и записываться. Перенос значений из теневых регистров в регистры CC6xR происходит по сигналу теневой передачи T12_ST, см. рисунки 18.18 – 18.24, таблицы 18.4 – 18.7.

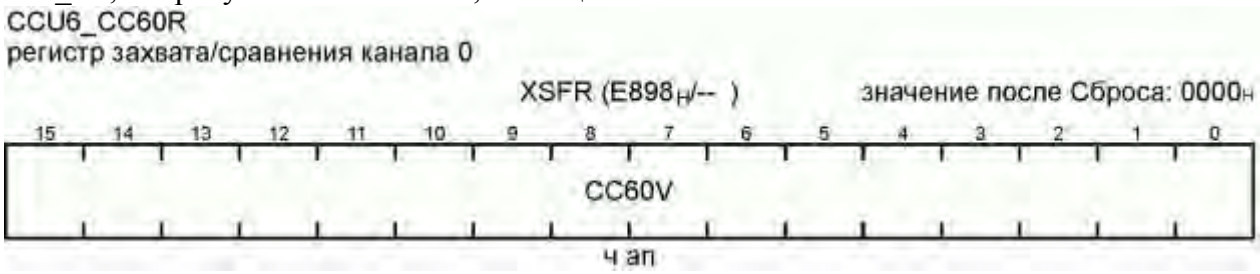


Рисунок 18.18 – Формат регистра CCU6_CC60R

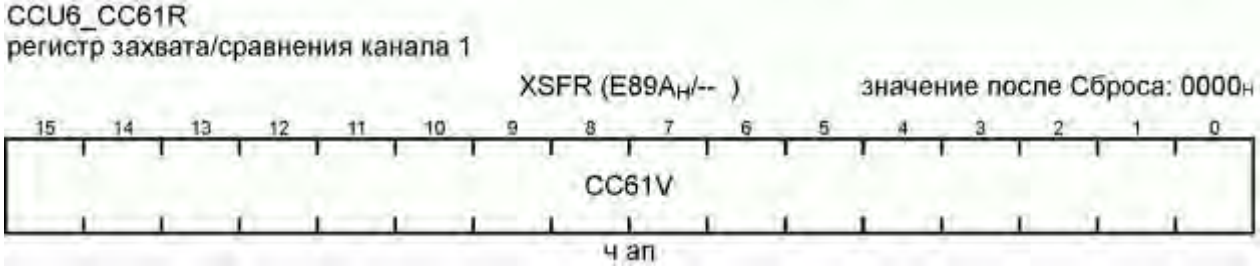


Рисунок 18.19 – Формат регистра CCU6_CC61R

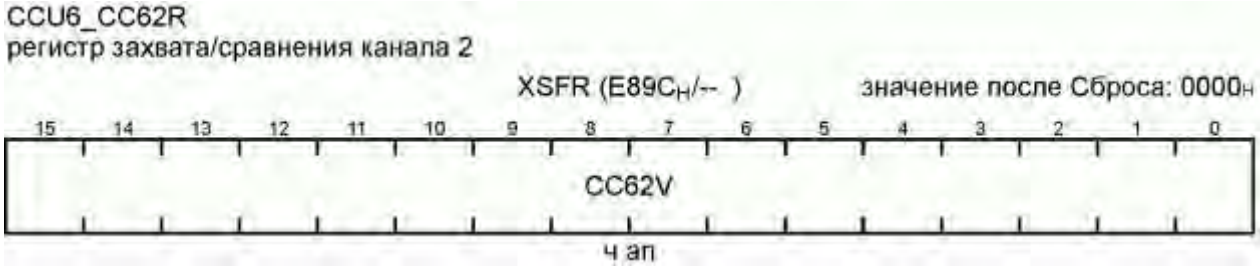


Рисунок 18.20 – Формат регистра CCU6_CC62R

Таблица 18.4 – Функциональное назначение полей регистров CCU6_CC60R, CCU6_CC61R и CCU6_CC62R

Поле	Биты	Тип	Описание
CC6xV x = 0, 1, 2	15-0	Чтение Аппаратное влияние	Сравниваемое значение канала x. В режиме сравнения регистр хранит значение, которое сравнивается с содержимым счетчика таймера T12. В режиме захвата, захваченное значение счетчика таймера T12 может быть считано из этого регистра.

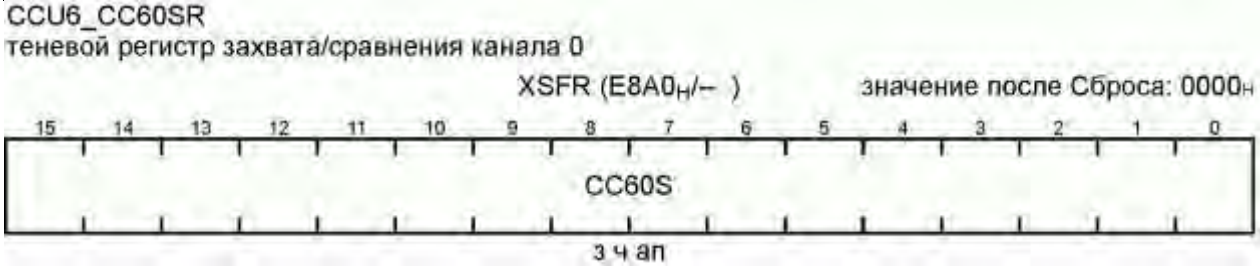


Рисунок 18.21 – Формат регистра CCU6_CC60SR

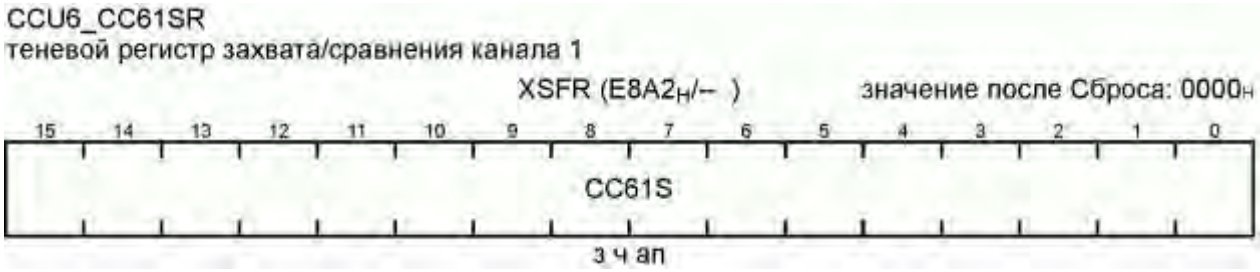


Рисунок 18.22 – Формат регистра CCU6_CC61SR

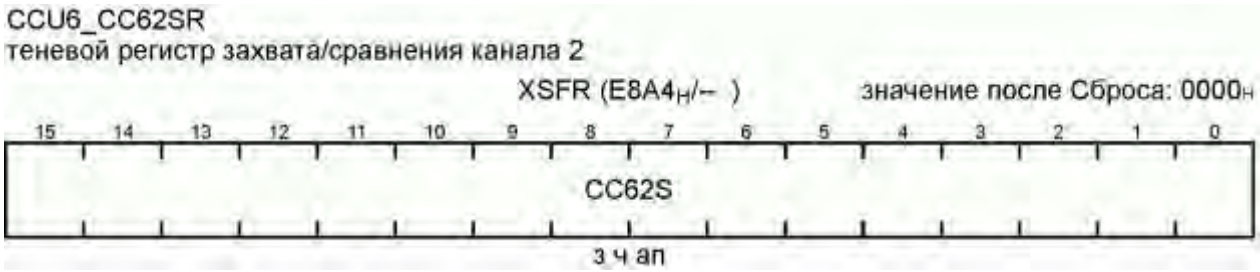


Рисунок 18.23 – Формат регистра CCU6_CC62SR

Таблица 18.5 – Функциональное назначение полей регистров CCU6_CC60SR, CCU6_CC61SR и CCU6_CC62SR

Поле	Биты	Тип	Описание
CC6xS, x = 0, 1, 2	15-0	Чтение Запись Аппаратное влияние	Теневой регистр для сравниваемого значения канала x. В режиме сравнения содержимое регистра переносится в регистр CC6xR по сигналу теневой передачи. В режиме захвата захваченное значение счетчика таймера T12 может быть считано из этого регистра.

Регистры CC6xR в режиме сравнения, x = 0, 1, 2

Значение, записанное в CC6xR, сравнивается (одновременно во всех трех каналах) со значением счетчика таймера T12. На выходе компаратора каждого канала формируется сигнал совпадения (согласно правилам), который затем обрабатывается логикой управления битом состояния канала.

Регистры CC6xR в режиме захвата, x = 0, 1, 2

Когда на входе канала возникает заранее определенное и ожидаемое событие, текущее значение регистра счета таймера T12 захватывается и записывается в регистры CC6xR или CC6xSR (согласно выбранному режиму).

Регистр T12MSEL содержит биты управления выбором режима захвата/сравнения для каждого из трех каналов таймера T12.

Рисунок 18.24 – Формат регистра выбора CCU6_T12MSEL режима захвата/сравнения

Таблица 18.6 – Функциональное назначение полей регистра CCU6_T12MSEL

Поле	Биты	Тип	Описание
MSEL62 MSEL61 MSEL60	11-8 7-4 3-0	Чтение Запись	Выбор режима захвата/сравнения. Биты выбирают режим для каждого канала. Каждый из трех каналов может быть запрограммирован независимо от других (исключение составляет режим датчиков Холла). Кодировка представлена в таблице 18.7.

Таблица 18.7 – Обзор режимов захвата/сравнения

MSEL6x	Выбранный режим				
0000_B	Вывод сигнала сравнения запрещен. Выходы используются для простого ввода-вывода.				
0001_B	Вывод сигнала сравнения на вывод CC6x. Вывод COUT6x используется для простого ввода-вывода.				
0010_B	Вывод сигнала сравнения на вывод COUT6x. Вывод CC6x используется для простого ввода-вывода.				
0011_B	Вывод сигнала сравнения на выводы COUT6x и CC6x.				
	Режим	Вывод	Активный перепад	Содержимое теневого регистра CC6xSR переносится в регистр	Содержимое таймера T12 переносится в регистр
0100_B	1	CC6x	0 – 1	–	CC6xR
		CC6x	1 – 0	–	CC6xSR
0101_B	2	CC6x	0 – 1	CC6xR	CC6xSR
0110_B	3	CC6x	1 – 0	CC6xR	CC6xSR
0111_B	4	CC6x	любой	CC6xR	CC6xSR
1000_B	Режим датчиков Холла				
1001_B	Режим блокирования				
	Режим	Вывод	Активный перепад	Содержимое таймера T12 переносится в регистр	
1010_B	5	CC6x	0 – 1	CC6xR	
		CCPOSx	1 – 0	CC6xSR	
1011_B	6	CC6x	1 – 0	CC6xR	
		CCPOSx	0 – 1	CC6xSR	
1100_B	7	CC6x	0 – 1	CC6xR	
		CCPOSx	0 – 1	CC6xSR	
1101_B	8	CC6x	1 – 0	CC6xR	
		CCPOSx	1 – 0	CC6xSR	
1110_B	9	CC6x	любой	CC6xR	
		CCPOSx	любой	CC6xSR	
1111_B	–	Зарезервировано (никаких действий захвата/сравнения не производится).			
Примечание – Режим датчиков Холла должен программироваться одновременно для трех каналов таймера T12.					

В приложении Ж дана таблица Ж.2 регистров области XSFR, принадлежащих к блоку захвата/сравнения CAPCOM6, с указанием физических адресов и значений после сброса (RESET).

18.4 Управление задержкой «dead-time»

При управлении силовыми ключами, в связи с заметной разницей во времени между открыванием и запиранием последних, возникает необходимость задержки переключения, управляющих сигналов на заранее определенное время, т. е. включение времени «dead-time» в общее время переключения.

Для этого в блоке CAPCOM6 имеется блок управления задержкой «dead-time», который позволяет задерживать переключение сигнала из пассивного в активный уровень (обратное не подлежит управлению).

Структуру блока управления задержкой «dead-time», показанную на рисунке 18.25, имеют все три канала таймера T12. Любое изменение бита CC6xST запускает соответствующий 6-битный декрементирующий счетчик задержки, который тактируется частотой таймера T12.

По импульсу DTCx_IN, сигнализирующему об изменении бита состояния CC6xST, в счетчик задержки загружается значение задержки, хранящееся в регистре T12DTS, и происходит запуск счета.

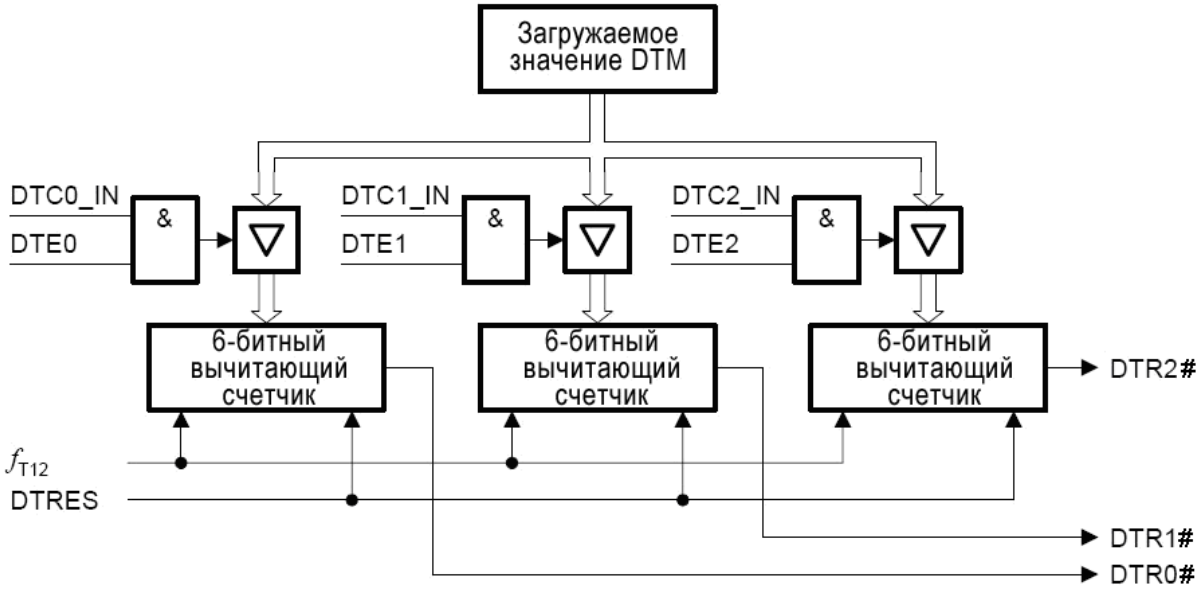


Рисунок 18.25 – Генерация задержки «dead-time»

В течение времени, пока счетчик включен, выходная линия DTRx# находится в «0» (пассивное состояние). Таким образом, значение DTM, загружаемое из регистра T12DTC определяет длительность пассивного состояния выходного сигнала, и, соответственно, задержку «dead-time» в переключении бита статуса из «0» в «1».

Значение DTM является одним для всех каналов.

Когда счетчик задержки достигает нуля, он останавливается, и выходная линия DTRx# переключается в «1».

Каждый из трех счетчиков задержки имеет свой индивидуальный бит управления DTEx для разрешения переключения входа DTCx_IN, т. е. разрешение включения счетчика задержки.

Запись нового значения задержки в регистр счетчика задержки не возможна в течение его работы. Это исключает возможность изменения значения задержки «dead-time» при изменении состояния бита статуса CC6xST.

Примечание – В любой момент времени все три счетчика задержки каналов одновременно могут быть сброшены в ноль и остановлены сигналом DTRES регистра TCTR4.

На рисунке 18.26 показаны переключения сигналов с внесенной задержкой «dead-time».

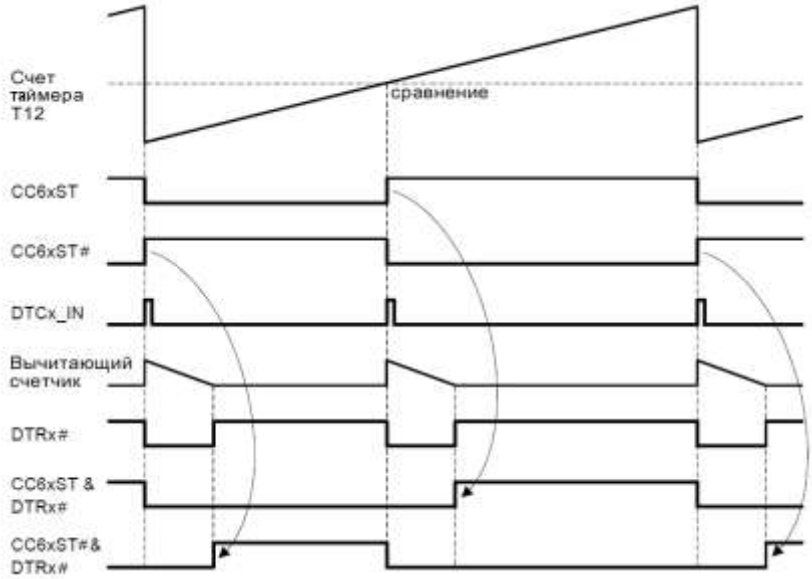


Рисунок 18.26 – Сигналы при внесении задержки «dead-time»

Регистр T12DTS управляет внесением задержки «dead-time» в изменение сигналов каналов таймера T12 (каждый канал программируется индивидуально). Формат регистра CCU6_T12DTC представлен на рисунке 18.27, функциональное назначение полей данного регистра – в таблице 18.8.

CCU6_T12DTC
регистр контроля задержки "dead-time" T12

XSFR ($E894_H$/--) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	DTR 2	DTR 1	DTR 0	-	DTE 2	DTE 1	DTE 0	-	-	DTM					
	ч ап	ч ап	ч ап		з ч	з ч	з ч			з ч					

Рисунок 18.27 – Формат регистра CCU6_T12DTC

Таблица 18.8 – Функциональное назначение полей регистра CCU6_T12DTC

Поле	Биты	Тип	Описание
DTR2 DTR1 DTR0	14 13 12	Чтение Аппаратное влияние	Флаг работы счетчика задержки. Отражает работу счетчика задержки для каждого канала 2, 1, 0, соответственно. 0 Счетчик задержки остановлен. 1 Счетчик задержки запущен.
DTE2 DTE1 DTE0	10 9 8	Чтение Запись	Разрешение генерирования задержки. Разрешение/запрет генерирования задержки для каждого канала 2, 1, 0, соответственно. 0 Запрет генерирования задержки. 1 Разрешение генерирования задержки.
DTM	5-0	Чтение Запись	Величина задержки. DTM определяет программируемую задержку «dead-time» между переключением канала из пассивного состояния в активное (определяет значение, с которого начинает счет счетчик задержки).

18.5 Режим захвата таймера T12

Когда на входе канала x возникает заранее определенное и ожидаемое событие, текущее значение регистра счета таймера T12 захватывается и записывается в регистры CC6xR или CC6xSR, согласно выбранному режиму.

Имеется несколько различных режимов захвата. Во всех режимах используются оба регистра каждого канала: основной и теневой. Это снижает количество обращений к ЦПУ по прерываниям, т. к. обслуживания требует только каждое второе прерывание. Выбор режима осуществляется посредством битовых полей MSEL6x в регистре T12MSEL, см. таблицу 18.7. Каждый канал может быть запрограммирован на свой режим.

Режим захвата 1

На рисунке 18.28 показан режим захвата 1 MSEL = 0100_B. Когда на соответствующем входе CC6x обнаруживается передний фронт (положительный перепад из «0» в «1») сигнала, текущее содержимое счетчика таймера T12 заносится в регистр CC6xR. В случае обнаружения заднего фронта (отрицательного перепада из «1» в «0») сигнала, содержимое таймера T12 заноситься в CC6xSR.

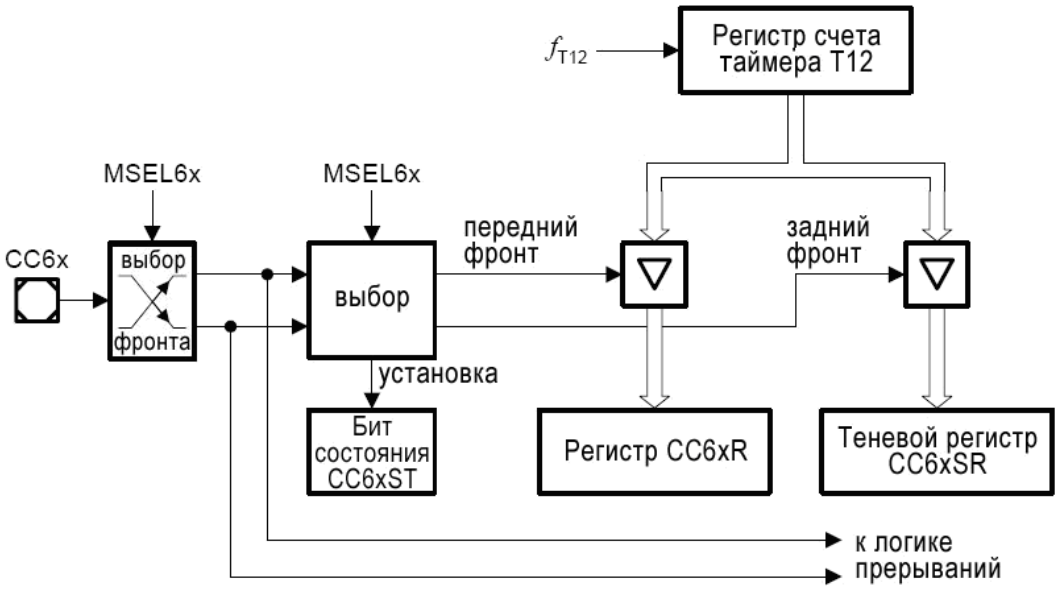


Рисунок 18.28 – Режим захвата 1 таймера T12

Режимы захвата 2, 3 и 4

Функциональная схема режимов захвата 2, 3 и 4 представлена на рисунке 18.29. В каждом из трех режимов, при обнаружении на соответствующем входе CC6x ожидаемого фронта сигнала, текущее состояние теневого регистра CC6xSR заносится в регистр CC6xR, и, синхронно с этим, текущее состояние счетчика таймера T12 – в регистр CC6xSR.

Эти режимы захвата весьма эффективно можно использовать в случаях очень малого интервала времени между последовательно приходящими фронтами входного сигнала.

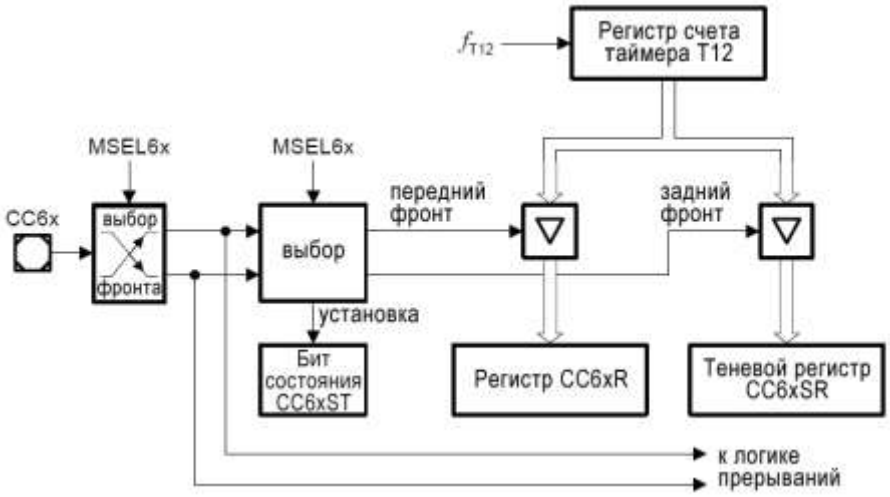


Рисунок 18.29 – Функциональная схема режимов захвата 2, 3 и 4 таймера T12

Режимы захвата 5 – 9

Оставшиеся пять режимов захвата 5, 6, 7, 8 и 9 относятся к мультивходовым режимам, поскольку в них используются по два входа CC6x и CC6POSx, см. рисунок 18.30.

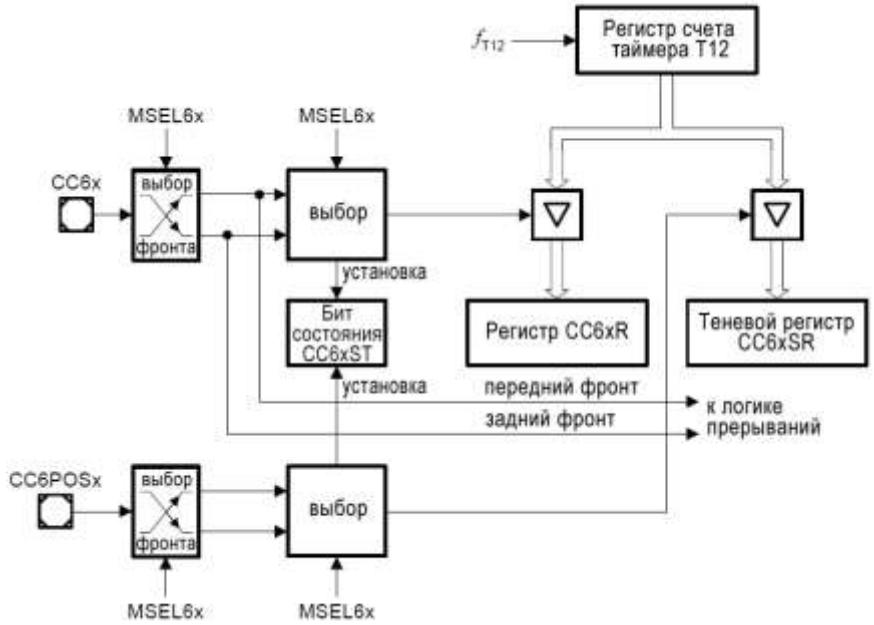


Рисунок 18.30 – Мультивходовый режим захвата таймера T12

В каждом из этих режимов текущее состояние счетчика таймера T12 захватывается в регистр CC6xR в соответствии с ожидаемым событием на входе CC6x, и в регистр CC6xSR – в соответствии с ожидаемым событием на входе CC6POSx.

В каждом режиме захвата, когда происходит ожидаемое событие на входе CC6x и/или CC6POSx (в мультивходовых режимах), бит состояния соответствующего канала CC6xST устанавливается в «1». Сброс бита состояния в любом режиме захвата осуществляется только программно.

При обнаружении ожидаемого события генерирует запрос на прерывание. Независимо от выбранного ожидаемого события все фронты, обнаруженные на выводе CC6x, приводят к генерированию соответствующих прерываний.

Режим блокирования

Режим блокирования может использоваться совместно только с режимами сравнения.

Режим MSE6x = 1001_B представляет собой возможность блокирования изменений бита состояния канала x, согласно уровню сигнала на входе CCPOSx, см. рисунок 18.31.

Если вход CCPOSx становиться равным «0», бит состояния канала x переводится в пассивное состояние и остается равным «0», пока на входе CCPOSx поддерживается «0».

Как только на входе CCPOSx возникает «1», блокирование бита состояния канала x снимается и канал работает в прежнем режиме захвата.

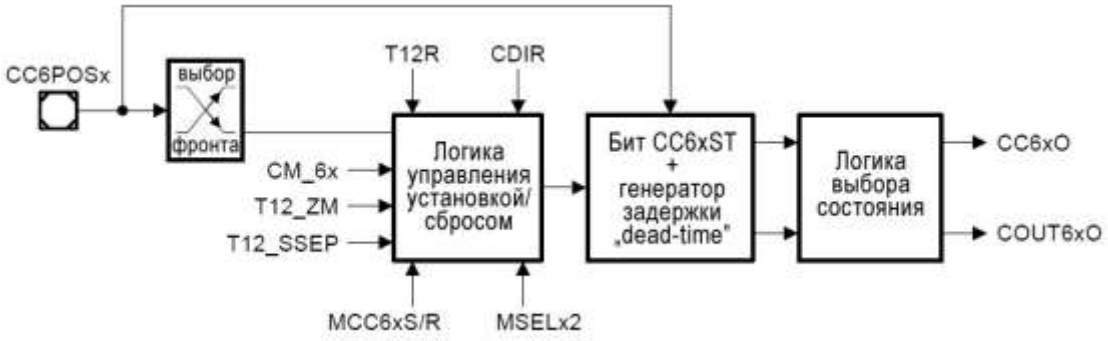


Рисунок 18.31 – Логика работы в режиме блокирования

Примечание – Если выходы канала x контролировались согласно заданному режиму MSELx, и схема работала в режиме сравнения, то запись значения 1001_B в MSELx не внесет изменений в работу, но режим блокирования включится. Чтение соответствующего поля MSELx вернет значение 1001_B. Следует помнить, что схема будет продолжать работать в режиме, который был задан в поле MSELx на момент записи в него значения 1001_B, но при этом будет реагировать на уровень сигнала на входе CCPOSx. Для выключения режима блокирования достаточно записать в поле MSELx значение, соответствующее любому другому режиму работы.

18.6 Блок таймера T13

Таймер 13 подобен таймеру T12, но имеет только один канал, работающий в режиме сравнения, см. рисунок 18.32. 16-битный счетчик (только инкрементирование) соединен с регистром канала через компаратор, который генерирует сигнал, когда содержимое счетчика совпадает с содержимым регистра. Множество функций управления таймером T13 открывает широкие возможности его использования. Помимо этого, таймер T13 может быть синхронизирован с событиями таймера T12.

На рисунке 18.33 показана подробная схема таймера T13. Таймер получает входную тактовую частоту f_{T13} из частоты f_{CC6} посредством программируемого делителя и делителя 1/256. Деление частоты контролируется посредством битовых полей T13CLK и T13PRE. Таймер T13 может считать только «вверх» (инкрементирование), аналогично режиму «пилы» таймера T12.

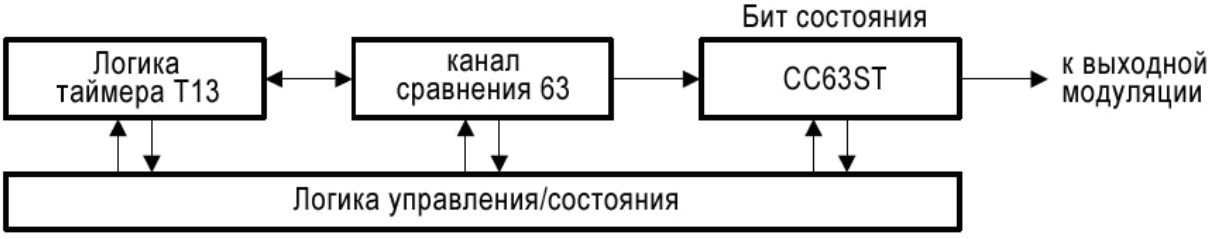


Рисунок 18.32 – Блок таймера T13

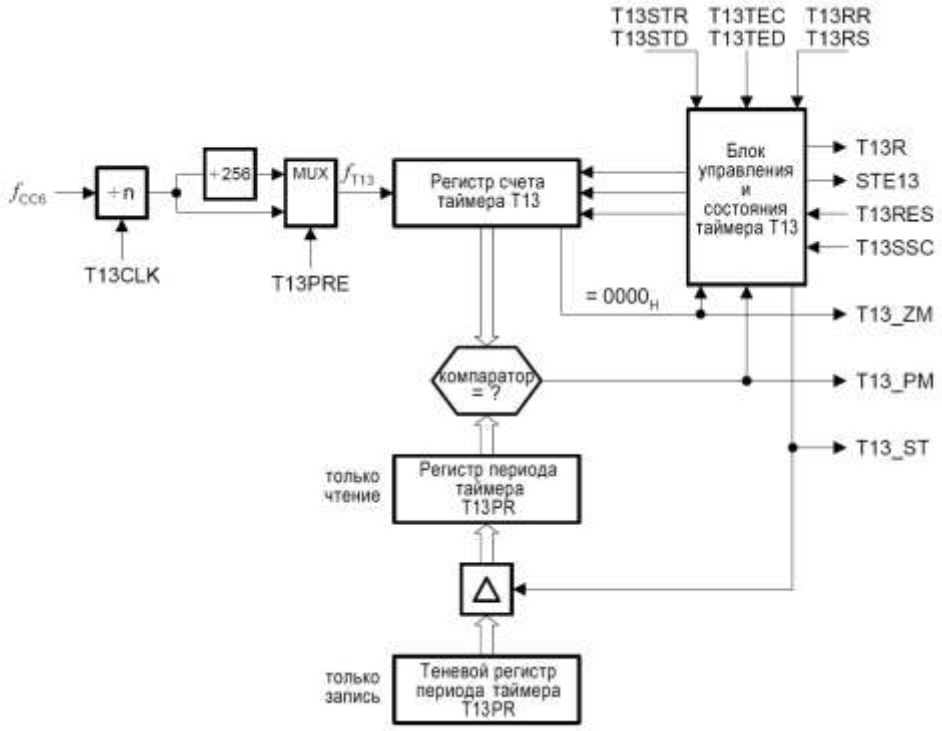


Рисунок 18.33 – Компаратор периода и логика таймера T13

Через компаратор таймер T13 соединен с регистром периода T13PR. Этот регистр определяет максимальное значение (значение периода), до которого считает таймер. По достижении счетчиком таймера значения периода, генерируется сигнал T13_PM (совпадение по периоду) и с приходом очередного такта тактовой частоты счетчик таймера T13 сбрасывается в 0000_H.

Регистр периода таймера T13PR получает новое значение периода из теневого регистра периода T13PS, который записывается программно. Передача нового значения периода из теневого регистра в регистр периода T13PR контролируется посредством сигнала теневой передачи T13_ST. Генерирование этого сигнала зависит от бита контроля STE13.

Работой таймера T13 (на аппаратном уровне) управляет сигнал T13_ZM совпадения содержимого счетчика таймера с нулем.

Бит управления однократным срабатыванием T13SSC осуществляет автоматическую остановку таймера по достижении значения периода.

Запуск и остановка таймера T13 управляется битом T13R. Бит программно может быть установлен или сброшен посредством парной установки битов T13RS и T13RR. Также бит T13R может быть сброшен аппаратно.

Теневая передача для регистров таймера T13 (T13_ST) активируется битом STE13. Этот бит программно может быть установлен или сброшен посредством парной установки битов T13STR и T13STD.

Два битовых поля T13TEC и T13TED, см. рисунки 18.34, 18.35, таблицы 18.9, 18.10, управляют синхронизацией T13 относительно событий таймера T12. T13TEC указывает на событие, по которому включается таймер T13, а T13TED определяет в каком направлении («вверх» или «вниз») при этом считает таймер T12.

Регистр счета таймера T13 хранит текущее значение счета. Этот регистр может быть записан только во время, когда таймер T13 остановлен (в течение времени, когда счетчик таймера включен, запись невозможна). Регистр счета всегда может быть прочитан программно.

Рисунок 18.34 – Формат регистра CCU6_T13

Таблица 18.9 – Функциональное назначение полей регистра CCU6_T13

Поле	Биты	Тип	Описание
T13CV	15-0	Чтение Запись Аппаратное влияние	16-битное значение счетчика таймера T13.

Регистр T13PR содержит значение периода таймера T13, см. рисунок 18.35, таблицу 18.10. Значение периода сравнивается с текущим значением счетчика таймера T13 и действия, производимые после, зависят от правил счета.

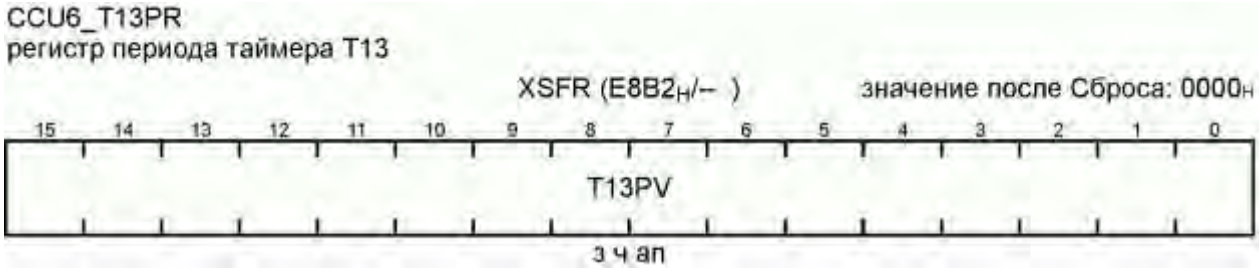


Рисунок 18.35 – Формат регистра CCU6_T13PR

Таблица 18.10 – Функциональное назначение полей регистра CCU6_T13PR

Поле	Биты	Тип	Описание
T13PV	15-0	Чтение Запись	Значение периода таймера T13. Когда значение счетчика таймера сравнивается со значением T13PV, возникает совпадение по периоду. После этого счетчик таймера сбрасывается в ноль.

Этот регистр имеет теневой регистр с таким же адресом. Запись в регистр периода таймера осуществляется при теневой передаче синхронно со счетом таймера, т. е. новое значение будет записано в момент очередного инкрементирования/декрементирования таймера T13.

Теневой регистр периода таймера доступен только для записи, в то время как основной регистр периода таймера – только для чтения.

Регистр периода таймера T13 всегда может быть прочитан программно.

Операции таймера T13

Входная частота f_{T13} таймера T13, см. таблицу 18.11, получается из частоты f_{CC6} посредством программируемого делителя и делителя 1/256.

Таблица 18.11 – Входная частота таймера T13

T13CLK	Входная частота таймера f_{T13}	
	Дополнительный делитель выключен T13PRE = 0_B	Дополнительный делитель включен T13PRE = 1_B
000_B	f_{CC6}	f_{CC6} / 256
001_B	f_{CC6} / 2	f_{CC6} / 512
010_B	f_{CC6} / 4	f_{CC6} / 1024
011_B	f_{CC6} / 8	f_{CC6} / 2048
100_B	f_{CC6} / 16	f_{CC6} / 4096
101_B	f_{CC6} / 32	f_{CC6} / 8192
110_B	f_{CC6} / 64	f_{CC6} / 16384
111_B	f_{CC6} / 128	f_{CC6} / 32768

Период счета таймера определяется значением в регистре периода T13PR и режимом таймера.

$T13_{PER}$ = «значение периода» + 1, в тактах таймера T13.

Как было описано выше, счетчик таймера T13 может считать только «вверх».

Если с приходом очередного такта f_{T13} возникает совпадение по периоду, счетчик сбрасывается в ноль, см. рисунок 18.36.

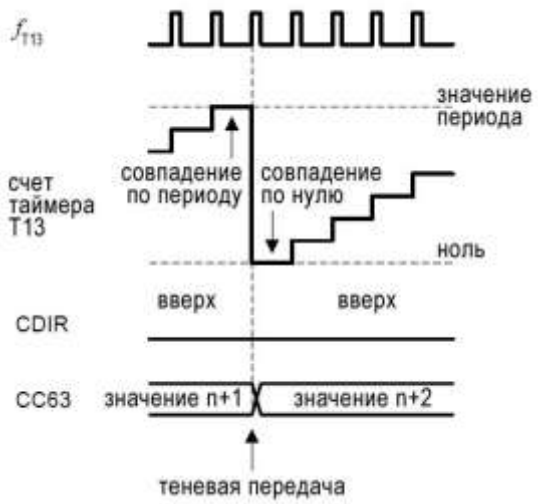


Рисунок 18.36 – Работа таймера T13

Примечание – Здесь и далее на рисунках показаны варианты изменения сигналов для случаев, когда частота переключения счетчика таймера максимальна и совпадает с f_{CC}, т. е. T13CLK = 000_B и T13PRE = 0_B.

Сигнал теневой передачи T13_ST

Генерация/запрет сигнала теневой передачи управляется битом STE13. Программно этот бит может быть установлен или сброшен посредством установки/сброса бита T13STR.

Запись «1» в бит T13STR (только для записи) устанавливает бит STE13, который в свою очередь активирует сигнал теневой передачи. После установки бита STE13, в случае если таймер запущен, аппаратная часть ожидает очередное инкрементирование таймера T13 и синхронно с переключением счетчика таймера производит запись в регистры из соответствующих им теневых регистров, после чего бит T13STR автоматически очищается, что влечет за собой очистку бита STE13.

Помимо программной организации теневой передачи имеется возможность аппаратной организации. По умолчанию, в случае если бит T13STD равен «0», аппаратная

часть будет генерировать сигнал теневой передачи (выставлением бита STE13) каждый раз, когда будет возникать совпадение по периоду T13_PM. Запись «1» в бит T13STD отключит аппаратное генерирование теневой передачи (для включения следует очистить бит T13STD).

Примечание – Программное генерирование теневой передачи (установка бита T13STR) возможно в любой момент времени, независимо от состояния бита T13STD.

В случае если таймер остановлен, запись в регистры происходит сразу после появления сигнала теневой передачи.

Контроль старта/останова и сброса таймера T13

Счетом таймера T13 управляет бит T13R. Счетчик таймера T13 активен только в случае, если бит T13R = 1_B. Программно этот бит может быть установлен посредством записи «1» в бит T13RS (только для записи) или сброшен посредством записи «1» в бит T13RR (только для записи).

Также T13R может быть очищен аппаратно в режиме однократного срабатывания.

Очистить счетчик таймера T13 (также и во время, когда таймер считает) можно программно, записью «1» в бит T13RES (только для записи). Установка этого бита обнуляет счетчик (в него записывается значение 0000_H) синхронно с инкрементированием, но не оказывает другого влияния, например, не останавливает его.

Режим однократного срабатывания

В режиме однократного срабатывания, см. рисунок 18.37, бит запуска T13R находится под программным и аппаратным влиянием.

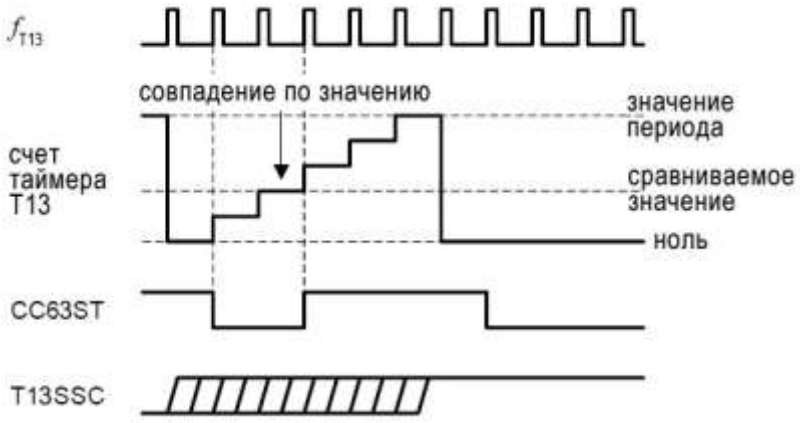


Рисунок 18.37 – Однократное срабатывание

Режим активируется посредством бита T13SSC. Следует помнить, что запись бита T13SSC возможна, только если таймер T13 остановлен.

Если бит T13SSC установлен, то счетчик таймера T13 после запуска остановится, отсчитав один период таймера, см. рисунок 18.37.

Для выхода из режима необходимо записать «0» в бит T13SSC.

Синхронизация таймеров T12 и T13

Таймер T13 может быть синхронизирован с событиями таймера T12. Битовые поля T13TEC и T13TED выбирают событие, по которому будет аппаратно запускаться таймер T13.

Так если обнаруживается выбранное событие, бит T13R устанавливается аппаратно и счетчик таймера T13 запускается. Как вариант, такая возможность в совокупности с режимом однократного срабатывания может использоваться для реализации программируемых задержек желаемых событий, находящихся в зависимости от выбранных событий таймера T12.

На рисунке 18.38 показан пример синхронизации запуска таймера T13 по событию таймера T12. В данном случае событием является совпадение по значению (сравниваемое значение равно 2) в течение счета таймера T12 «вверх». Тактовые частоты T12 и T13 могут отличаться (влияние делителя); на рисунке тактовая частота таймера T13 равна половине тактовой частоты таймера T12.

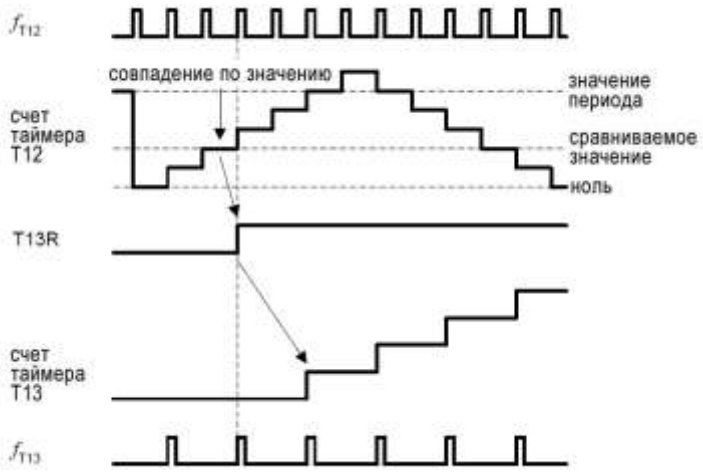


Рисунок 18.38 – Синхронизация запуска таймера T13 по таймеру T12

Битовое поле T13TEC выбирает событие (связанное с таймером T12), по которому следует запустить таймер T13, см. таблицу 18.12, а битовое поле T13TED определяет, какое направление («вверх» или «вниз») счета таймера T12 при этом используется таблица 18.13.

Таблица 18.12 – Выбор события, связанного с таймером T12

T13TEC	Ожидаемое событие
000_B	Не выбрано
001_B	Совпадение по значению канала 0
010_B	Совпадение по значению канала 1
011_B	Совпадение по значению канала 2
100_B	Совпадение по значению любого канала (0, 1, 2)
101_B	Совпадение по периоду таймера T12
110_B	Совпадение по нулю таймера T12
111_B	Любой сигнал с датчиков Холла

Таблица 18.13 – Выбор направления счета таймера T12

T13TED	Желаемое направление счета таймера T12
00_B	Зарезервировано
01_B	Ожидаемое событие обнаруживается, когда таймер T12 считает «верх»
10_B	Ожидаемое событие обнаруживается, когда таймер T12 считает «вниз»
11_B	Ожидаемое событие обнаруживается независимо от направления счета таймера T12

18.7 Режимы сравнения T13

С таймером T13 связан один канал сравнения, см. рисунок 18.39.

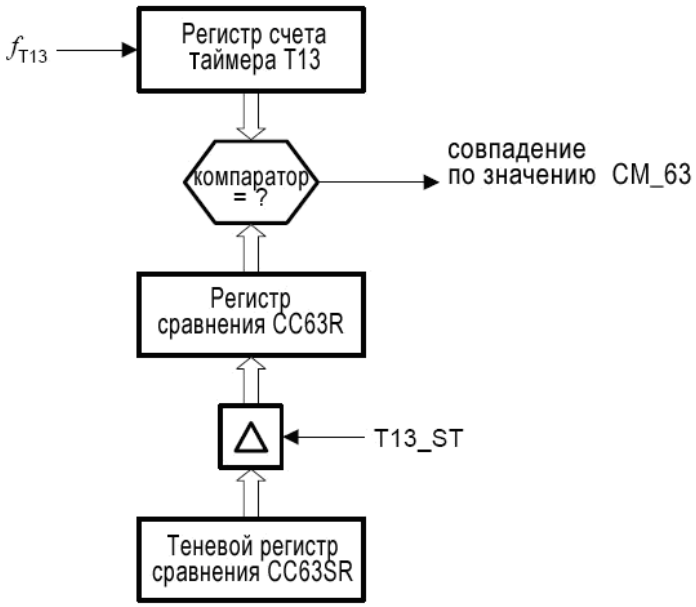


Рисунок 18.39 – Компаратор канала таймера T13

Канал соединен с регистром счета таймера T13 посредством его индивидуального компаратора сравнения, который генерирует сигнал совпадения, когда содержимое счетчика совпадает с содержимым регистра сравнения.

Канал состоит из компаратора и двух регистров – регистра сравнения CC63R, содержимое которого загружается в компаратор, и соответствующего ему теневого регистра CC63SR, который записывается программно и передает значение в регистр сравнения по сигналу теневой передачи T13_ST.

С каналом связан бит состояния CC63ST, который отражает состояние выходной линии канала, см. рисунок 18.40.

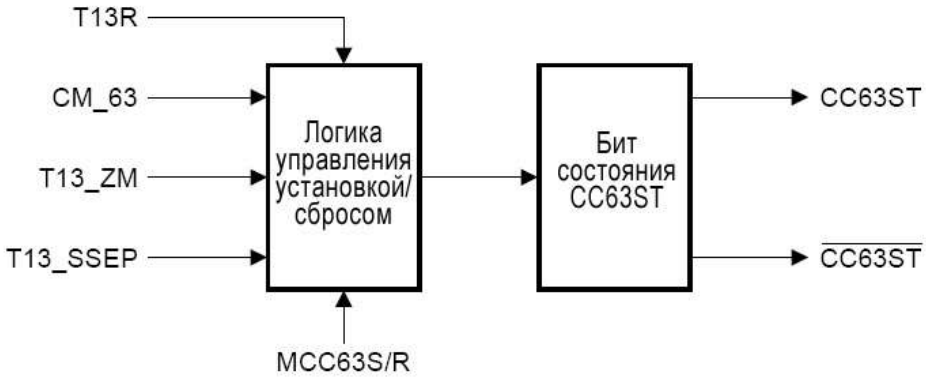


Рисунок 18.40 – Бит состояния канала таймера T13

Входы логики управления установкой/сбросом бита состояния CC63ST следующие:
- бит запуска таймера T13R;
- сигнал совпадения счетчика таймера с нулем T13_ZM;
- сигнал однократного срабатывания T13_SSEP;
- сигнал совпадения по значению CM_63.

Помимо этого, бит состояния может быть программно установлен (установкой бита MCC63S) или сброшен (установкой бита MCC63R).

Аппаратное изменение бита состояния CC63ST возможно только во время работы таймера T13 (T13R = 1_B). Если таймер включен, то бит состояния изменяется по правилам.

Бит CC63ST переключается в «1», если:

- при очередном переключении счетчика таймера, возникает совпадение по значению;

- при сбросе счетчика таймера в ноль по переполнению, возникает совпадение по значению, равному 0000_H.

Бит CC63ST переключается в «0», если при сбросе счетчика таймера в ноль по переполнению не возникает совпадение по значению, равное 0000_H.

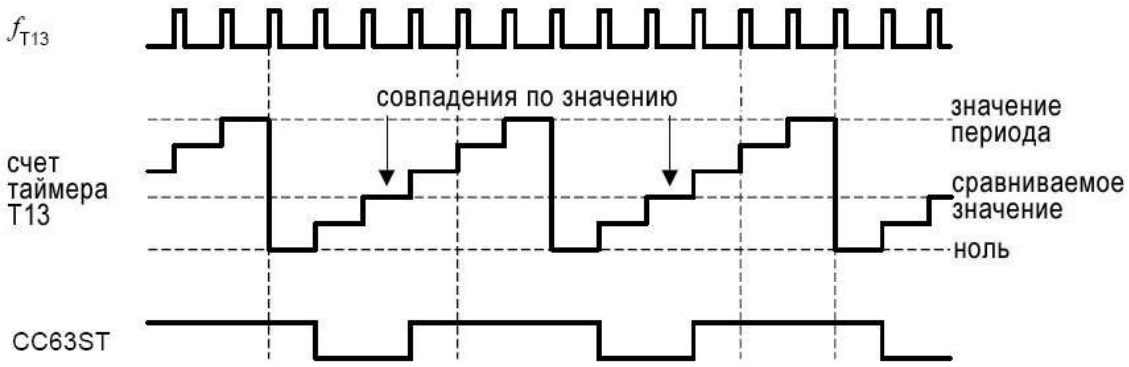


Рисунок 18.41 – Операция сравнения таймера T13

Примечание – Сигналы на рисунке 18.41 аналогичны сигналам таймера T12.

Вывод сигнала в режиме сравнения

На рисунке 18.42 отражен в общем виде путь сигнала от бита состояния канала до выхода. Как видно из рисунка, пользователь имеет широкие возможности управления выходным сигналом, который формируется битом состояния CC63ST.

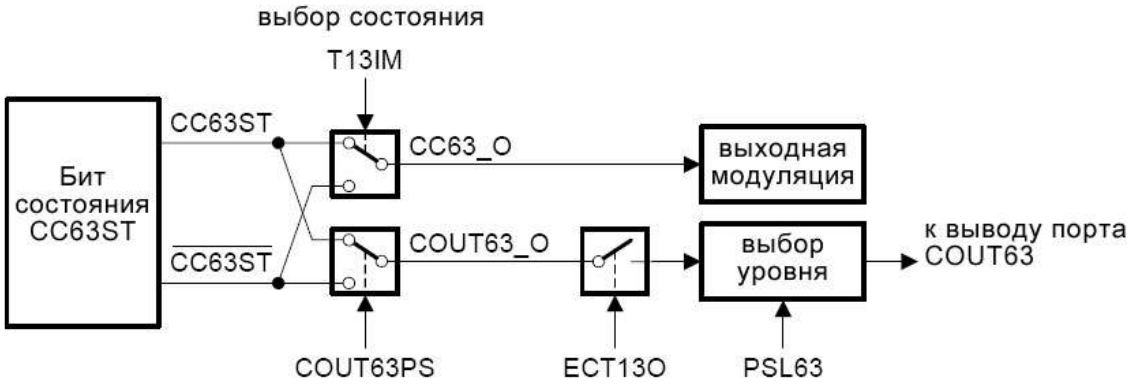


Рисунок 18.42 – Вывод сигнала в режиме сравнения

18.8 Регистры сравнения

Регистр CC63R, см. рисунок 18.43, таблицу 18.14, является регистром, который может быть только программно прочитан. Запись в этот регистр осуществляется через соответствующий ему теневой регистр CC63SR, см. рисунок 18.44, таблицу 18.15, который может программно читаться и записываться. Перенос значения из теневого регистра в регистр CC63R происходит по сигналу теневой передачи T13_ST.

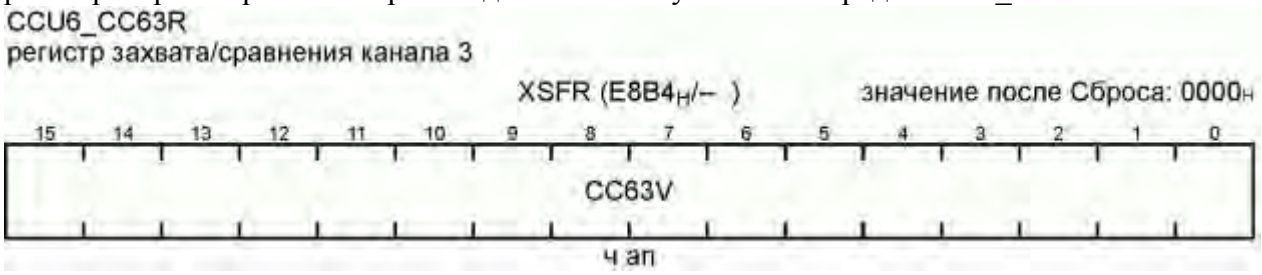


Рисунок 18.43 – Формат регистра CCU6_CC63R

Таблица 18.14 – Функциональное назначение поля регистра CCU6_CC63R

Поле	Биты	Тип	Описание
CC63V	15-0	Чтение Аппаратное влияние	Сравниваемое значение канала 3. Регистр хранит значение, которое сравнивается с содержимым счетчика таймера T13.

Рисунок 18.44 – Формат регистра CCU6_CC63SR

Таблица 18.15 – Функциональное назначение поля регистра CCU6_CC63SR

Поле	Биты	Тип	Описание
CC63S	15-0	Чтение Запись Аппаратное влияние	Теневой регистр для сравниваемого значение канала 3. Содержимое регистра переноситься в регистр CC63R по сигналу теневой передачи.

18.9 Управление таймерами

Функционированием таймеров T12 и T13 управляют регистры TCTR0, TCTR2 и TCTR4, см. рисунки 18.45 – 18.47, таблицы 18.16 – 18.18.

CCU6_TCTR0
регистр управления таймерами 0

XSFR ($E8AC_H$/--) значение после Сброса: 0000_H

15	14	13	12	11	10 9 8	7	6	5	4	3	2 1 0
-	-	STE 13	T13R	T13 PRE	T13CLK	CTM	CDIR	STE 12	T12R	T12 PRE	T12CLK
-	-	ч ап	ч ап	з ч	з ч	з ч	ч ап	ч ап	ч ап	з ч	з ч

Рисунок 18.45 – Формат регистра CCU6_TCTR0

Регистр TCTR2 управляет режимом однократного срабатывания и синхронизацией таймеров T12 и T13, см. рисунок 18.46, таблицу 18.17.

CCU6_TCTR2
регистр управления таймерами 2

XSFR ($E8AE_H$/--) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6 5	4 3 2	1	0
-	-	-	-	-	-	-	-	-	T13TED	T13TEC	T13 SSC	T12 SSC
-	-	-	-	-	-	-	-	-	з ч	з ч	з ч	з ч

Рисунок 18.46 – Формат регистра CCU6_TCTR2

Регистр CCU6_TCTR4, см. рисунок 18.47, таблицу 18.18, предназначен для программного управления запуском/остановкой и сбросом таймеров T12 и T13, теневыми передачами и остановкой счетчиков задержки «dead-time» каналов. Чтение регистра всегда возвращает значение 0000_H.

CCU6_TCTR4
регистр управления таймерами 4

XSFR ($E8A6_H$/--) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13 STD	T13 STR	-	-	-	T13 RES	T13 RS	T13 RR	T12 STD	T12 STR	-	-	DT RES	T12 RES	T12 RS	T12 RR
3	3	-	-	-	3	3	3	3	3	-	-	3	3	3	3

Рисунок 18.47 – Формат регистра CCU6_TCTR4

Таблица 18.16 – Функциональное назначение полей регистра CCU6_TCTR0

Поле	Биты	Тип	Описание
1	2	3	4
STE13[1)]	13	Чтение Аппаратное влияние	Разрешение теневой передачи таймера T13: 0 Теневая передача не требуется. 1 Требуется теневая передача.
T13R[2)]	12	Чтение Аппаратное влияние	Бит работы таймера T13. T13R запускает и останавливает счетчик таймера T13. Этот бит устанавливается программно битом T13RS и сбрасывается программно битом T13RR или аппаратно, согласно правилам режима однократного срабатывания. 0 Таймер T13 остановлен. 1 Таймер T13 запущен.
T13PRE	11	Чтение Запись	Бит включения делителя частоты таймера T13. Позволяет включать и выключать дополнительный делитель 1/256 частоты работы счетчика таймера T13. 0 Дополнительный делитель выключен. 1 Дополнительный делитель включен.
T13CLK	10-8	Чтение Запись	Выбор входной тактовой частоты таймера T13. Поле позволяет выбирать делитель входной частоты f_{CC6} для работы счетчика таймера. Правило: $f_{T13} = f_{CC6} / 2^{<T13CLK>}$ (таблица 18.11)
CTM	7	Чтение Запись	Режим работы таймера T12: 0 Режим «пила». 1 «Симметричный» режим.
CDIR	6	Чтение Аппаратное влияние	Направление счета таймера T12. Показывает текущее направление («вверх» – инкрементирование / «вниз» – декрементирование) счета таймера T12. 0 Счетчик таймера T12 считает «вверх». 1 Счетчик таймера T12 считает «вниз».
STE12[1)]	5	Чтение Аппаратное влияние	Разрешение теневой передачи таймера T12: 0 Теневая передача не требуется. 1 Требуется теневая передача.
T12R[2)]	4	Чтение Аппаратное влияние	Бит работы таймера T12. T12R запускает и останавливает счетчик таймера T12. Этот бит устанавливается программно битом T12RS и сбрасывается программно битом T12RR или аппаратно, согласно правилам режима однократного срабатывания. 0 Таймер T12 остановлен. 1 Таймер T12 запущен.

Окончание таблицы 18.16

1	2	3	4
T12PRE	3	Чтение Запись	Бит включения делителя частоты таймера T12. Позволяет включать и выключать дополнительный делитель 1/256 частоты работы счетчика таймера T12. 0 Дополнительный делитель выключен. 1 Дополнительный делитель включен.
T12CLK	2-0	Чтение Запись	Выбор входной тактовой частоты таймера T12. Поле позволяет выбирать делитель входной частоты f_{CC6} для работы счетчика таймера. Правило: $f_{CC6} / 2^{<T12CLK>}$ (таблица 18.3).
Примечание – При необходимости можно осуществлять запись в битовые поля T12CLK/T13CLK, T12PRE/T13PRE и бит CTM во время работы таймеров. [1] Бит STE12/STE13 очищается аппаратно по окончании теневой передачи. [2] Параллельная установка/сброс T13R/T12R (из TxSSC, TxRR, TxRS) не будет иметь эффекта. Бит T13R/T12R остается без изменений.			

Таблица 18.17 – Функциональное назначение полей регистра CCU6_TCTR2

Поле	Биты	Тип	Описание
T13TED	6-5	Чтение Запись	Выбор направления счета таймера T12. Поле определяет, при каком направлении счета («вверх»/«вниз») таймера T12 детектировать ожидаемое событие, см. таблицу 18.13.
T13TEC	4-2	Чтение Запись	Выбор события, связанного с таймером T12. Поле определяет ожидаемое событие, связанное с таймером T12, при обнаружении которого будет запущен таймер T13.
T13SSC T12SSC	1 0	Чтение Запись	Бит однократного срабатывания таймера T13/T12. Бит управляет включением/выключением режима однократного срабатывания. 0 Режим однократного срабатывания выключен. 1 Режим однократного срабатывания включен.

Таблица 18.18 – Функциональное назначение полей регистра CCU6_TCTR4

Поле	Биты	Тип	Описание
1	2	3	4
T13STD T12STD	15 7	Запись	Бит управления аппаратной теневой передачей таймера T13/T12: - для таймера T12: 0 Аппаратное генерирование сигнала теневой передачи каждый раз, когда возникает совпадение по периоду T12_PM при счете таймера T12 «вверх» и совпадение с единицей (T12_OM) при счете таймера T12 «вниз». 1 Аппаратное генерирование запрещено. - для таймера T13: 0 Аппаратное генерирование сигнала теневой передачи каждый раз, когда возникает совпадение по периоду T13_PM. 1 Аппаратное генерирование запрещено.
T13STR T12STR	14 6	Запись	Бит создания запроса теневой передачи таймера T13/T12: 0 Нет запроса. 1 Запрос теневой передачи. По этому сигналу устанавливается бит STE13/STE12.
T13RES T12RES	10 2	Запись	Сброс таймера T13/T12[1]. Обнуление содержимого счетчика таймера без влияния на его работу. 0 Нет действий. 1 Содержимое счетчика таймера T13/T12 обнуляется.
T13RS T12RS	9 1	Запись	Бит запуска счетчика таймера T13/T12[2]: 0 Бит T13R/T12R не установлен. 1 Бит T13R/T12R устанавливается, и счетчик таймера T13/T12 запускается.
T13RR T12RR	8 0	Запись	Бит остановки счетчика таймера T13/T12[2]: 0 Бит T13R/T12R не очищается. 1 Бит T13R/T12R очищается, и счетчик таймера T13/T12 останавливается.
DTRES	3	Запись	Сброс и остановка счетчиков задержки «dead-time»[1]: 0 Нет действия. 1 Счетчики задержки «dead-time» всех каналов очищаются и останавливаются.

[1] Бит аппаратно очищается после сброса счетчика.

[2] Одновременная установка обоих битов (запуска и остановки) таймера в «1» или «0» игнорируется.

Регистр состояния захвата/сравнения каналов CCU6_CMPSTAT, см. рисунок 18.48, таблицу 18.19, содержит биты состояния, отражающие текущее состояние, и биты управления, определяющие активное/пассивное состояние каналов.

CCU6_CMPSTAT
регистр состояния захвата/сравнения

XSFR ($E8A8_H$/–) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13 IM	C OUT 63PS	C OUT 62PS	CC 62PS	C OUT 61PS	CC 61PS	C OUT 60PS	CC 60PS	-	CC 63ST	CC POS 2	CC POS 1	CC POS 0	CC 62ST	CC 61ST	CC 60ST
з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	-	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап

Рисунок 18.48 – Формат регистра CCU6_CMPSTAT

Таблица 18.19 – Функциональное назначение полей регистра CCU6_CMPSTAT

Поле	Биты	Тип	Описание
1	2	3	4
T13IM	15	Чтение Запись Аппаратное влияние	Бит управления инвертированием модулируемого сигнала канала таймера T13: 0 Сигнал CC63_O подается в прямом виде для дальнейшей модуляции. 1 Сигнал CC63_O подается в инвертированном виде для дальнейшей модуляции.
COUT63PS[1)] COUT62PS CC62PS COUT61PS CC61PS COUT60PS CC60PS	14 13 12 11 10 9 8	Чтение Запись Аппаратное влияние	Управление пассивным состоянием линии вывода сигнала в режимах сравнения. Бит выбирает состояние линии вывода сигнала. В пассивном состоянии на линии удерживается низкий уровень сигнала. 0 Линия вывода сигнала в активном состоянии. Уровень сигнала отражает состояние бита CC6xST. 1 Линия в пассивном состоянии. Поддерживается низкий уровень сигнала.
CC63ST CC62ST CC61ST CC60ST	6 2 1 0	Чтение Запись Аппаратное влияние	Бит состояния канала таймера T13/T12: Бит отражает работу канала в режимах захвата/сравнения. Режимы сравнения: 0 Значение счетчика таймера меньше сравниваемого значения (хранится в регистре CC6xR). 1 Значение счетчика таймера больше сравниваемого значения. Режимы захвата (только для каналов 0, 1, 2): 0 Выбранный и ожидаемый фронт входного сигнала канала еще не обнаружен. 1 Выбранный и ожидаемый фронт входного сигнала канала обнаружен.
CCPOSx	5, 4, 3	Чтение Аппаратное влияние	Биты для сигналов с датчиков Холла[2)]. Биты отражают состояние сигналов, приходящих с датчиков Холла.

[1)] Биты имеют теневые биты. Запись значений осуществляется в теневые биты, а перенос из теневых битов – по сигналу соответствующей теневой передачи.

[2)] Только для режима датчиков Холла (для всех каналов MSELx = 1000_B).

Регистр CCU6_CMPMODIF, см. рисунок 18.49, таблицу 18.20, содержит биты программного управления битами состояния CC6xST каналов, независимо от их состояния и состояния таймеров.

Рисунок 18.49 – Формат регистра CCU6_CMPMODIF

Таблица 18.20 – Функциональное назначение полей регистра CCU6_CMPMODIF

Поле	Биты	Тип	Описание
MCC63R MCC62R MCC61R MCC60R	14 10 9 8	Запись	Сброс бита состояния CC6xST канала x: 0 Нет действий. 1 Бит CC6xST очищается [1].
MCC63S MCC62S MCC61S MCC60S	6 2 1 0	Запись	Установка бита состояния CC6xST канала x: 0 Нет действий. 1 Бит CC6xST устанавливается [1].
[1] Одновременная установка обоих битов MCC6xR и MCC6xS в «1» или «0» инвертирует состояние соответствующего бита состояния CC6xST.			

18.10 Мультиканальный режим

Мультиканальный режим позволяет управлять модуляцией всех шести выходных сигналов таймера T12. Для включения режима необходимо установить бит MODCTR.MCMEN = 1_B.

Регистр CCU6_MODCTR состоит из битов, управляющих комбинацией и модуляцией выходных сигналов каналов обоих таймеров, см. рисунок 18.50, таблицу 18.21.

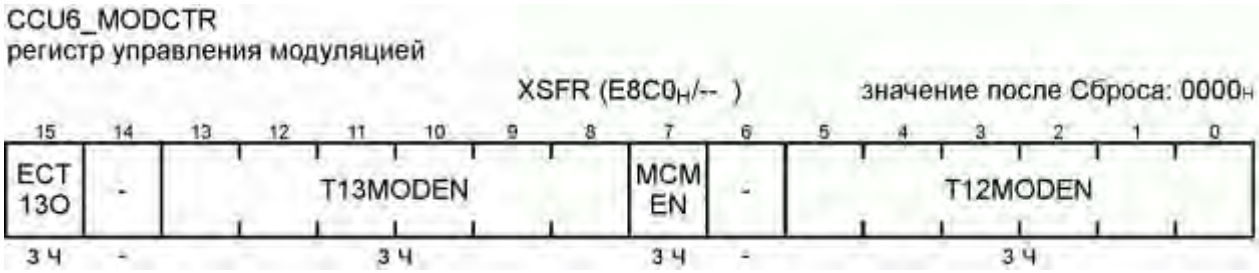


Рисунок 18.50 – Формат регистра CCU6_MODCTR

Таблица 18.21 – Функциональное назначение полей регистра CCU6_MODCTR

Поле	Биты	Тип	Описание
ECT13O	15	Чтение Запись	Разрешение вывода сигнала канала таймера T13: 0 Вывод сигнала запрещен (на выходе поддерживается низкий уровень сигнала). 1 Вывод сигнала разрешен.
T13MODEN T12MODEN	13-8 5-0	Чтение Запись	Выбор выходных сигналов, каналов таймера T12. Биты управляют выбором источников выходных сигналов каналов. 0 Нет источника. 1 Источником является выбранный сигнал, модулируемый таймером T13/T12. Каждый бит поля T13MODEN/T12MODEN управляет соответствующим ему выходом. Соответствие выходов битам полей T13MODEN и NT12MODEN (слева направо) следующее: COUT62, CC62, COUT61, CC61, COUT60, CC60
MCMEN	7	Чтение Запись	Включение мультиканального режима. Бит разрешения управления модуляцией выходных сигналов посредством поля MCMOUT. 0 Мультиканальный режим запрещен. 1 Мультиканальный режим разрешен.

Каждый канал таймера T12 имеет две линии вывода сигнала (к выходам CC6x и COUT6x). Каждая линия имеет два источника выходного сигнала – бит состояния CC6xST и бит состояния CC63ST, который помимо использования каналом 3 идет на каждую из шести выходных линий каналов 0, 1 и 2.

Поля T13MODEN и T12MODEN управляют выбором источников выходных сигналов каналов. После сброса содержимое обоих полей равно нулю, вследствие чего на всех выходах каналов поддерживается низкий уровень сигнала.

Примечание – Следует помнить, что битовые поля T13MODEN и T12MODEN должны быть заданы (независимо от состояния бита MCMEN) и их значения не должны побитно совпадать. Если биты (обоих полей), управляющие одним выводом, находятся в одинаковом состоянии (оба равны «0» или «1»), на соответствующем им выходе будет поддерживаться низкий уровень сигнала, независимо от уровня сигнала источников.

Если мультиканальный режим включен, биты поля MCMOUT.MCMP управляют выходными линиями каналов, открывая или закрывая их для вывода сигналов. Открытыми (отражающими изменения битов состояний CC6xST каналов) будут линии вывода сигнала, управляющие биты которых находятся в состоянии «1». Закрытые линии поддерживают низкий уровень сигнала.

Битовое поле MCMP имеет свое теневое битовое поле MCMPS. Передача значения из поля MCMPS в поле MCMP может происходить как программно (в любой момент, когда это необходимо, асинхронно по отношению к работе таймеров) по сигналу теневой передачи STRMCM, так и по аппаратному сигналу теневой передачи MCM_ST, который синхронизируется по событиям таймеров T12 и T13, что позволяет избежать появления нежелательного импульса из-за асинхронности работы счетчиков таймеров.

На рисунке 18.51 показан путь формирования сигнала теневой передачи MCM_ST.

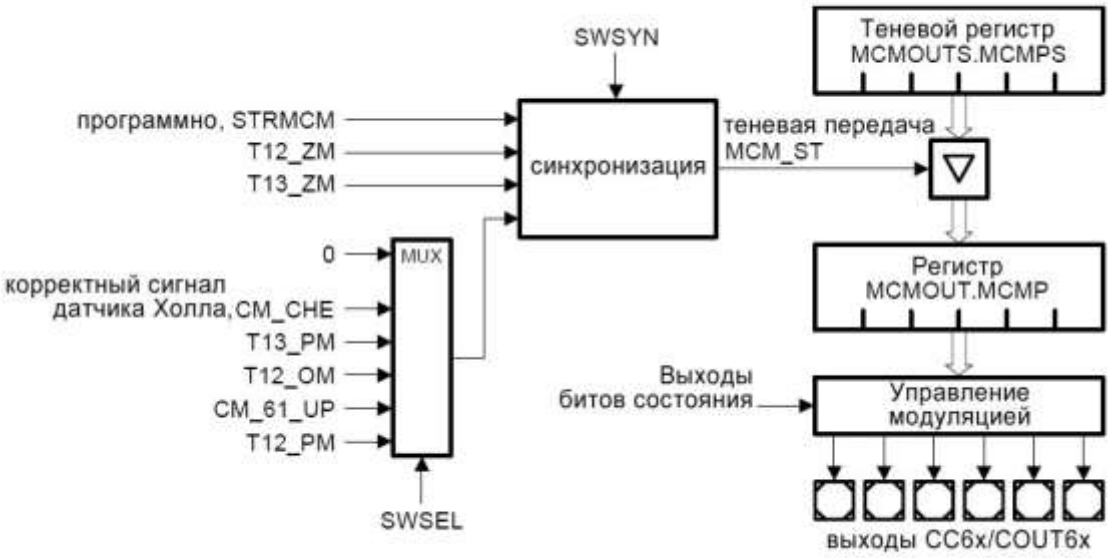


Рисунок 18.51 – Мультиканальный режим

Управление осуществляется посредством регистра CCU6_MCMCTR, см. рисунок 18.52, таблицу 18.22.

CCU6_MCMCTR
регистр управления мультиканальным режимом

XSFR ($E8CE_H$/--) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	SWSYN		-	SWSEL		
-	-	-	-	-	-	-	-	-	-	3 Ч		-	3 Ч		

Рисунок 18.52 – Формат регистра CCU6_MCMCTR

Таблица 18.22 – Функциональное назначение полей регистра CCU6_MCMCTR

Поле	Биты	Тип	Описание
SWSYN	5-4	Чтение Запись	Поле выбора сигнала синхронизации. Появление выбранного полем SWSYN сигнала активирует сигнал теневой передачи MCM_ST в случае, если бит R регистра MCMOUT равен «1».
SWSEL	2-0	Чтение Запись	Поле выбора события для теневой передачи. Поле SWSEL определяет сигнал, по которому будет установлен флаг R.

Примечание – Аппаратное генерирование сигнала теневой передачи MCM_ST возможно лишь в случае, если бит MCMEN = 1_B.

Битовое поле SWEL указывает одно из событий, см. таблицу 18.23, появление которого должно активировать теневую передачу из теневого поля MCMPS в поле MCMP синхронно с работой одного из таймеров или асинхронно по отношению к ним, см. таблицу 18.24.

Таблица 18.23 – Выбор события активирования теневой передачи в мультиканальном режиме

SWSEL	Выбранное событие
000_B	Событие не выбрано
001_B	Корректный сигнал датчика Холла CM_CHE
010_B	Совпадение по периоду таймера T13 (T13_PM)
011_B	Совпадение счетчика таймера T12 с единицей (T12_OM) при счете «вниз»
100_B	Совпадение по значению для канала 1 при счете таймера T12 «вверх» (CM_61_UP)
101_B	Совпадение по периоду таймера T12 (T12_PM) при счете «вверх»
$11x_B$	Зарезервировано

Таблица 18.24 – Выбор источника синхронизации

SWSEL	Источник синхронизации
00_B	Нет синхронизации
01_B	Синхронизация по сигналу совпадения с нулем таймера T13 (T13_ZM)
10_B	Синхронизация по сигналу совпадения с нулем таймера T12 (T12_ZM)
11_B	Зарезервировано

В случае, когда не требуется аппаратная синхронизация появления сигнала теневой передачи MCM_ST с работой таймеров SWSEL = 00_B, теневая передача будет осуществлена сразу при появлении выбранного события (поле SWSYN).

В случае, когда требуется синхронизация теневой передачи с работой одного из таймеров (определяется полем SWSEL), появление выбранного события (поле SWSYN) установит флаг ожидания R в регистре MCMOUT. Как только счетчик выбранного таймера обнулиться/достигнет нуля, выставится флаг совпадения с нулем таймера и вместе с этим будет сформирован сигнал теневой передачи MCM_ST. Таким образом, произойдет теневая передача содержимого поля MCMPS в поле MCMP, после чего флаг R будет аппаратно сброшен.

В случае если к моменту появления сигнала совпадения счетчика выбранного таймера с нулем (T12_ZM или T13_ZM) флаг R не будет выставлен (= «0»), сигнал MCM_ST не появится, и теневая передача не произойдет.

В случае аппаратной синхронизации, фактически, обновление содержимого поля MCMP будет происходить в начале каждого периода выбранного таймера.

18.11 Режим работы с датчиками Холла

Выбирается записью 1000_B во все поля MSELx. Бит MCMEN следует установить в «1».

Сигналы, формируемые блоком CAPCOM6, помимо прочего, могут использоваться как управляющие сигналы управления бесщеточным ДПТ. Для формирования сигналов управления, как правило, необходимо знать положение ротора двигателя, для чего могут использоваться датчики Холла. Блок CAPCOM6 имеет возможность подключения трех датчиков на входы CC6POS0, CC6POS1 и CC6POS2, что дает возможность аппаратного управления двигателем на основе информации, полученной с датчиков.

Как известно, между положением ротора двигателя и сигналами управления имеется строгое соотношение. Когда ротор достигает заданного положения (согласно информации с датчиков), блок CAPCOM6 формирует сигналы управления, которые контролируют модуляцию выходных сигналов, которые передаются к силовой части управления двигателем для поддержания необходимой скорости вращения ротора.

Поскольку виды модуляции различны для разных двигателей, желательна высокая гибкость в формировании управляющих сигналов.

Блок, отвечающий за управление формированием необходимых сигналов, имеет в своем составе регистр MCMOUT с соответствующим теневым регистром MCMOUTS.

Регистр хранит информацию о текущей поле CURH и ожидаемой (поле EXPH) комбинации сигналов с датчиков, а также управляющие сигналы MCMP.

Для отслеживания положения ротора двигателя CAPCOM6 опрашивает входы, соединенные с датчиками с частотой f_{CC6}. Как только обнаруживается очередная новая (ожидаемая) комбинация сигналов с датчиков, формируются новые управляющие сигналы.

Для защиты от шумов на входах, соединенных с датчиками, и для повышения точности, в блоке CAPCOM6 реализована возможность вставлять однократную задержку после каждого обнаружения новой комбинации сигналов на входах и до считывания этой комбинации блоком.

Помимо этого, блок CAPCOM6 реализует сравнение комбинаций сигналов с датчиков с комбинацией, записанной в регистровом поле CURH, для отслеживания и пропуска коротких неинформативных импульсов.

Для хранения комбинаций управляющих сигналов и для более гибкого управления выходной модуляцией, в блоке CAPCOM6 имеется двухрегистровая структура MCMOUTS-MCMOUT (теневой регистр-основной регистр), теневые передачи MCM_ST которой генерируются в зависимости от заданных условий.

Логика работы с датчиками Холла

На рисунке 18.53 показана двухрегистровая структура и логика сравнения.

В теневой регистр MCMOUTS программно записываются: управляющее значение MCMPS, текущая комбинация CURHS и ожидаемая комбинация EXPHS сигналов с датчиков. Регистр хранит эти значения до появления очередной ожидаемой комбинации сигналов с датчиков. При этом сигналы управления модуляцией передаются в блок управления выходной модуляцией, состояние на входах сигналов с датчиков обрабатывается компаратором.

Появление сигнала выборки HCRDY активирует захват комбинации сигналов со входов. Захваченная комбинация передается на компаратор. Если происходит совпадение принятой комбинации с ожидаемой EXPH, то генерируется сигнал CM_CHE (корректный сигнал датчика), если с текущей CURH, то сигнал CM_CHE не генерируется. В случае несовпадения принятой комбинации ни с одной из EXPH и CURH, генерируется сигнал CH_WHE (некорректный сигнал датчика).

Каждое появление сигнала CM_CHE генерирует сигнал теневой передачи, по которому происходит передача содержимого теневых полей CURHS и EXPHS в поля

CURH и EXPH, вследствие чего, текущая и ожидаемая комбинации сигналов обновляются. После этого программно можно записать новые значения в теневые поля. Сигнал теневой передачи может также быть сгенерирован установкой бита STRHP регистра MCMOUTS.

Помимо этого, появление сигнала CM_CHE, при необходимости, может генерировать сигнал теневой передачи MCM_ST, если заданы соответствующие комбинации SWSEL и SWSYN, что повлечет за собой передачу содержимого MCMOUTS в MCMOUT и, соответственно, изменение комбинации управляющих сигналов выходной модуляции. После чего в поле MCMOUTS может быть программно записано новое значение, которое попадет в поле MCMOUT при следующем появлении сигнала MCM_ST.

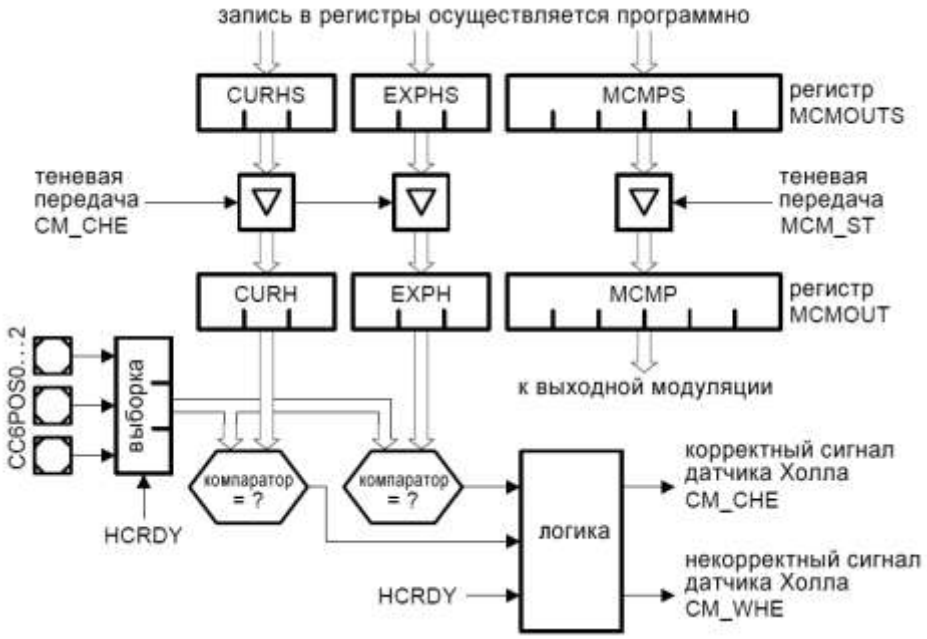


Рисунок 18.53 – Логика работы с датчиками Холла

Фильтрация сигналов с датчиков

Для подавления шумов на входах CC6POSx используется аппаратный фильтр.

Входы датчиков опрашиваются с тактовой частотой f_{CC6}, когда обнаруживается изменение уровня сигнала на одном из трех входов.

Фильтр шумов реализован на базе счетчика задержки «dead-time» DTC0 нулевого канала и детектора изменения фронтов входных сигналов на входах CC6POSx.

Как только возникает изменение уровня сигнала на одном из входов, генерируется сигнал, который запускает счетчик задержки «dead-time» DTC0. Когда счетчик достигает нуля, его выходной сигнал DTC0_O становится равным «1». Этот сигнал в режиме работы с датчиками Холла используется как сигнал выборки HCRDY. Появление сигнала выборки инициирует захват комбинации сигналов со входов CC6POSx, см. рисунок 18.54.

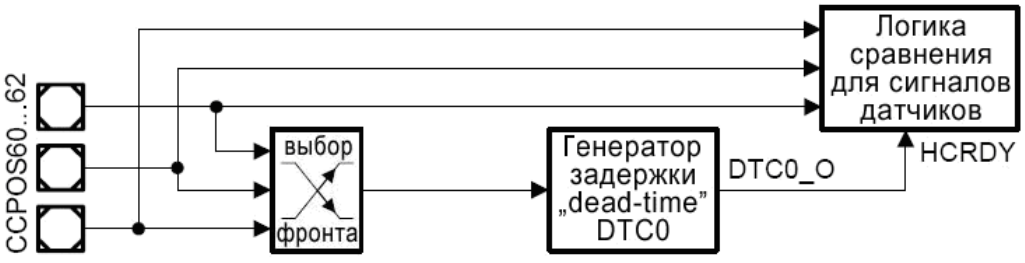


Рисунок 18.54 – Логика работы фильтра шумов

Таким образом, может быть устранено большинство шумов на линиях сигналов приходящих с датчиков. При обнаружении изменения уровня входного сигнала на любом из входов, комбинация с входов считывается через некоторое точно определенное время, отсчитываемое счетчиком DTC0. Затем она сравнивается с текущей и ожидаемой комбинациями. Если возникает совпадение с текущей комбинацией (CURH), пришедший сигнал считается шумом, и никаких дальнейших действий не предпринимается. При совпадении с ожидаемой комбинацией (EXPH), пришедший сигнал считается корректным и генерируется сигнал CM_CHE.

В случае несовпадения ни с одной из комбинаций CURH и EXPH генерируется сигнал CM_WHE (некорректный сигнал датчика).

Управление бесщеточным двигателем постоянного тока на основе блока таймера T12

Учитывая вышесказанное, на основе блока таймера T12 программно-аппаратными средствами можно построить систему управления бесщеточным двигателем постоянного тока.

Для перехода в соответствующий режим во все битовые поля MSELx трех каналов T12 следует записать в 1000_B.

Следует помнить, что в этом режиме канал 0 работает на захват, в то время как каналы 1 и 2 – на сравнение.

Из блока сравнения комбинаций сигналов с датчиков выходит сигнал CM_CHE, см. рисунок 18.55.

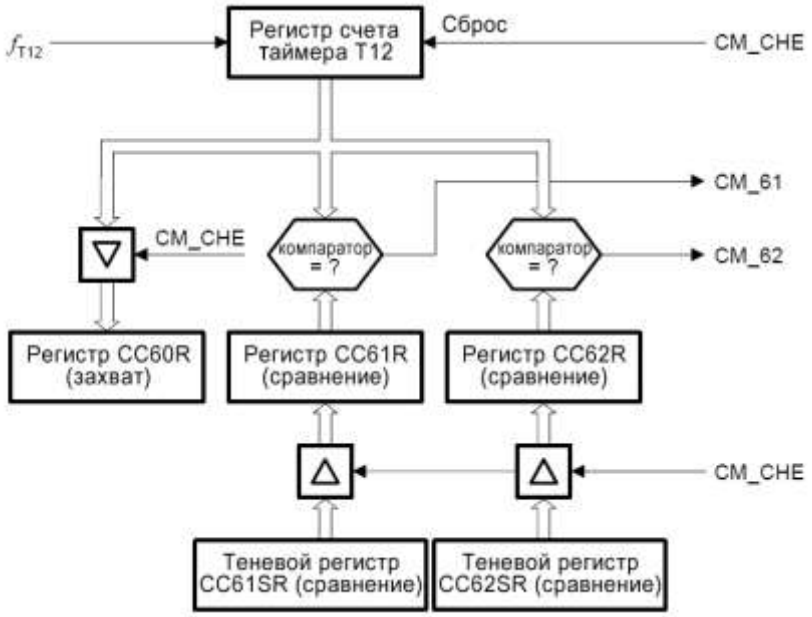


Рисунок 18.55 – Таймер T12 в режиме работы с датчиками Холла

Появление этого сигнала, помимо прочего, инициирует следующие действия:

- передачу нового сравниваемого значения из теневого регистра CC6xSR в соответствующий ему регистр сравнения CC6xR (для каналов 1 и 2);
- захват текущего состояния таймера T12 в регистр CC60R (для канала 0);
- сброс счетчика таймера T12 в 0000_H и запись нового значения периода таймера из теневого регистра периода.

Примечание – В этом режиме аппаратно не генерируется сигнал теневой передачи T12ST. Теневые биты, такие как биты PSLy, не будут переданы в их основные регистры. Для программирования основных регистров, запись в них должна осуществляться программно.

В целом, взаимодействие блока CAPCOM6 с ДПТ осуществляется, см. рисунок 18.56, следующим образом:

- После обнаружения корректной комбинации сигналов с датчиков, значение счетчика таймера T12 захватывается нулевым каналом (показывает текущую скорость двигателя), после чего таймер T12 сбрасывается в ноль и продолжает счет. По достижении таймером сравниваемого значения канала 1, генерируется (если разрешено) сигнал теневой передачи MCM_ST и новые значение управляющих модуляцией битов вступают в силу. Помимо этого, время счета таймера T12 до момента достижения их сравниваемого значения канала 1 может использоваться для фазовой задержки (необходима в случае работы с обратной ЭДС вместо датчиков Холла) от момента прихода сигналов с датчиков до момента переключения выходных сигналов.

- Отслеживание комбинаций сигналов с датчиков, которым необходима шумовая фильтрация, и управление выходной модуляцией могут быть засинхронизированы с работой источника модуляции.

- Сравниваемое значение в канале 2 может использоваться как флаг блокировки работы, сигнализирующий о том, что текущая скорость вращения двигателя значительно ниже желаемого значения, что может быть следствием повышения нагрузки на валу. В этом случае модуляция выходных сигналов таймером T12 должна быть прекращена (например, записью нуля в T12MODENx).

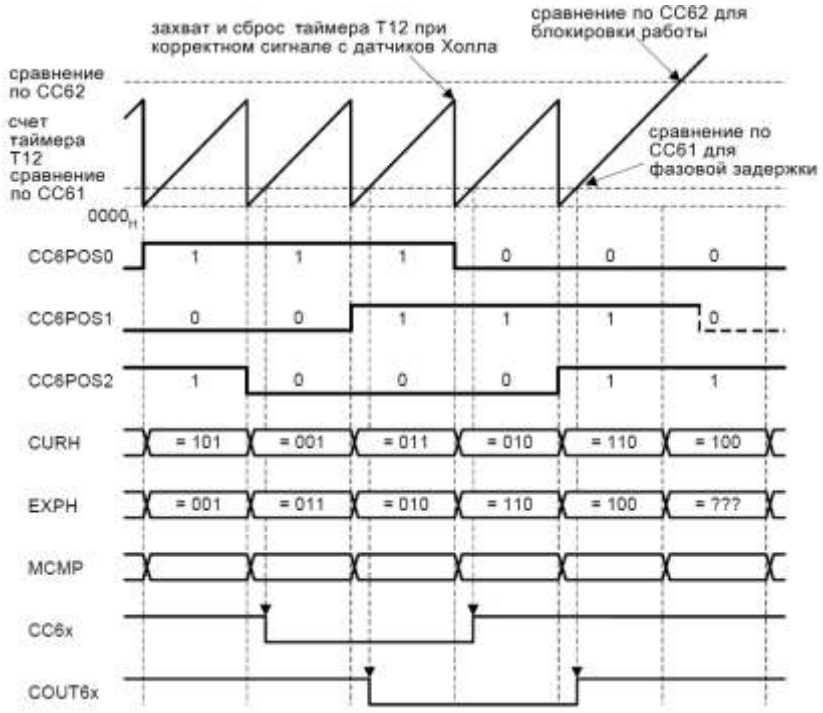


Рисунок 18.56 – Пример управления бесщеточным ДПТ (все MSEL6x = 1000_B)

Флаги, отражающие работу с датчиками Холла

Регистр состояния прерываний IS содержит биты, которые являются программно и аппаратно управляемыми флагами, отражающими работу блока CAPCOM6.

Флаг CHE в регистре IS устанавливается в «1», когда генерируется сигнал CH_CHE. Также, этот флаг может быть установлен программно, установкой бита SCHE в регистре установки прерываний ISS. Если бит разрешения ENCHE регистра разрешения прерываний IEN установлен в «1», то флаг CHE может сгенерировать запрос на прерывание ЦП. Для сброса флага CHE необходима программная запись «1» в бит RCHE в регистре сброса прерываний ISR.

Работа флага WHE аналогична работе флага CHE, см. рисунок 18.57.

Следует заметить, что для флагов CHE, WHE запрос на прерывание генерируется появлением передних фронтов сигналов CH_CHE и CH_WHE, соответственно. Это означает, что прерывание может быть сгенерировано, даже если флаг уже установлен. Таким образом, нет необходимости сбрасывать флаг для последующих прерываний.

Каждое появление сигнала MCM_ST генерирует прерывание.

Флаг IDLE устанавливается аппаратно, если это разрешено битом ENIDLE, когда появляется сигнал CM_WHE. Также можно установить флаг программно, записью «1» в бит SIDLE. Пока бит IDLE установлен, поле MCMP очищено и выходы блока CAPCOM6 находятся в пассивном состоянии. Чтобы вернуться к работе, флаг IDLE должен быть сброшен программно установкой в «1» бита RIDLE. Для полного выхода из режима Idle следует программно записать необходимый регистр, сгенерировав теневые передачи между полями MCMOUTS и MCMOUT, CURHS и CURHS, а также EXPHS и EXPH записью «1» в биты STRMCM и STRHP в регистре MCMOUTS. В таком случае, выход из режима Idle происходит под программным управлением, но в тоже время может быть синхронизирован с ШИМ сигналом.

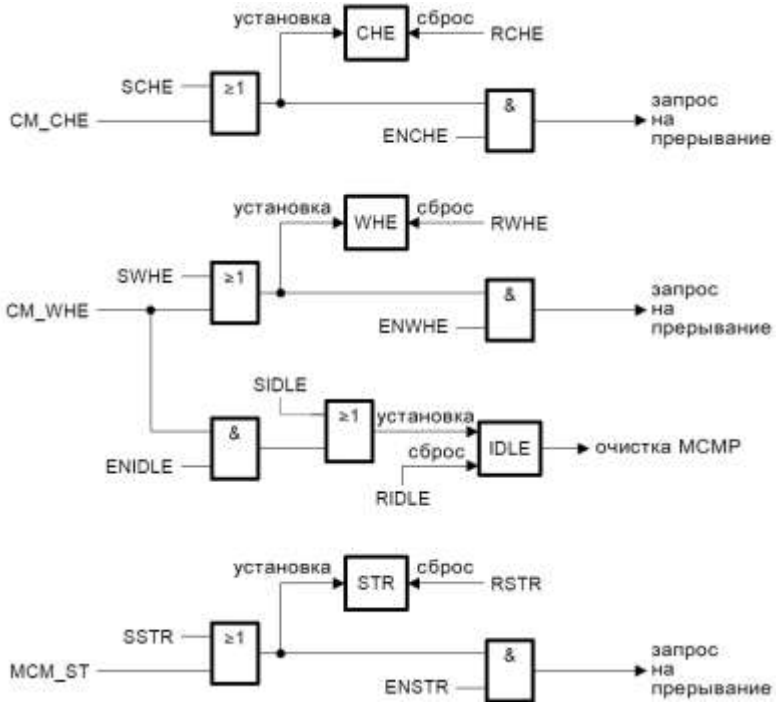


Рисунок 18.57 – Логика работы флагов в режиме датчиков Холла

Регистры режима работы с датчиками Холла

Основные регистры, используемые при работе с датчиками Холла – регистр MCMOUT и его теневой регистр MCMOUTS, см. рисунки 18.58, 18.59, таблицы 18.25, 18.26.

CCU6_MCMOUTS

теневой регистр управления выходами в мультиканальном режиме

XSFR ($E8CA_H$/–) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STR HP	-	CURHS			EXPHS			STR MCM	-	MCMPS					
3	-	3 Ч			3 Ч			3	-	3 Ч					

Рисунок 18.58 – Формат регистра CCU6_MCMOUTS

Таблица 18.25 – Функциональное назначение полей регистра CCU6_MCMOUTS

Поле	Биты	Тип	Описание
STRHP	15	Запись	Запрос программной теневой передачи для записи комбинаций сигналов в битовые поля CURH и EXPH из теневых полей CURHS и EXPHS. Теневая передача инициируется сразу после установки STRHP, после чего бит аппаратно очищается. Чтение этого бита всегда возвращает «0». 0 Теневая передача инициируется только аппаратно по сигналу CM_CHE. 1 Программная теневая передача.
CURHS	13-11	Запись	Теневое поле текущей комбинации сигналов датчиков. Содержимое этого поля записывается в поле CURH по сигналу теневой передачи.
EXPHS	10-8	Чтение Запись	Теневое поле ожидаемой комбинации сигналов датчиков. Содержимое этого поля записывается в поле EXPH по сигналу теневой передачи.
STRMCM	7	Запись	Запрос программной теневой передачи для записи комбинации значений, управляющих выходной модуляцией битов в поле MCMP из теневого поля MCMPS. После установки STRMCM теневая передача инициируется синхронно с очередным переключением счетчика таймера T12, после чего бит аппаратно очищается. Чтение этого бита всегда возвращает «0». 0 Теневая передача инициируется только аппаратно по сигналу MCM_ST. 1 Программная теневая передача.
MCMPS	5-0	Чтение Запись	Теневое поле комбинации значений, управляющих выходной модуляцией битов. Содержимое этого поля записывается в поле MCMP по сигналу теневой передачи.

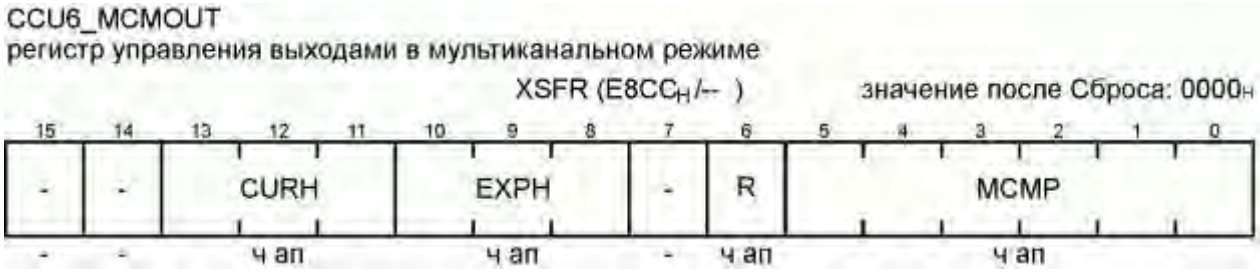


Рисунок 18.59 – Формат регистра CCU6_MCMOUT

Таблица 18.26 – Функциональное назначение полей регистра CCU6_MCMOUT

Поле	Биты	Тип	Описание
CURH[1)]	13-11	Чтение Аппаратное влияние	Поле текущей комбинации сигналов датчиков. Это поле получает значение из теневого поля CURHS по сигналу теневой передачи и хранит значение текущей комбинации сигналов, приходящих с датчиков Холла.
EXPH[1)]	10-8	Чтение Аппаратное влияние	Поле ожидаемой комбинации сигналов датчиков. Это поле получает значение из теневого поля EXPHS по сигналу теневой передачи и хранит значение ожидаемой комбинации сигналов, приходящих с датчиков Холла.
R	6	Чтение Аппаратное влияние	Флаг ожидания. Флаг ожидания синхронизированной теневой передачи из моля MCMPS в поле MCMP (см. рисунок 18.53). По окончании передачи аппаратно сбрасывается. Также бит R равен «0» в случае, когда MCMEN = 0_B. 0 Теневая передача не ожидается. 1 Теневая передача ожидается и еще не произошла.
MCMP[2)]	5-0	Чтение Аппаратное влияние	Поле комбинации управляющих выходной модуляцией битов. Это поле получает значение из теневого поля MCMPS по сигналу теневой передачи и хранит значение комбинации управляющих выходной модуляцией сигналов. Побитно. 0 На соответствующей выходной линии поддерживается низкий уровень. 1 Соответствующая выходная линия активна. MCMP.5 – MCMP .0 соответствует (побитно слева направо) COUT62, CC62, COUT61, CC61, COUT60, CC60.

[1)] Биты в полях EXPH и CURH соответствуют сигналам с датчиков Холла на входах CCPOSx, x = 0, 1, 2. EXPH.2, EXPH.1, EXPH.0 и CURH.2, CURH.1, CURH.0 соответствуют побитно слева направо CCPOS2, CCPOS1, CCPOS0.

[2)] Поле очищается в то время как бит IS.IDLE = 1_B.

18.12 Ловушки

Ловушки позволяют выходным сигналам реагировать на состояние входа CTRAP#. Это можно использовать для экстренной остановки работы, если CTRAP# становится активным.

Флаг ловушки TRPF отражает состояние попадания в ловушку. Помимо аппаратной установки флага TRPF по сигналу CTRAP# имеется возможность его программной установки.

Бит ловушки TRPS определяет состояние на выходных линиях сигналов и управляет выходом из состояния ловушки.

Когда возникает аппаратное или программное состояние попадания в ловушку, устанавливается флаг ловушки TRPF, установка которого переводит бит ловушки TRPS в

«1». Сигнал бита TRPS идет в блок выходной модуляции. В случае если TRPS = 1_B, выбранные выходные линии сигналов переводятся в пассивное состояние и на них поддерживается низкий уровень до тех пор, пока TRPS = 1_B. Каждая выходная линия сигнала (всех каналов таймера T12 и канала таймера T13) имеет свой управляющий бит ловушки.

Выход из состояния ловушки может осуществляться различными путями:

- немедленно, по сигналу CTRAP#, когда он становиться неактивным;
- под программным контролем;
- синхронно с генерацией выходного сигнала таймерами T12 или T13.

Наилучший путь выхода из состояния ловушки – программный.

На рисунке 18.60 показана структурная схема блока обработки состояний ловушки.

Когда входной сигнал CTRAP# становится равным «0» (активный уровень), выставляется флаг TRPF, если TRPPEN = 1_B (находится в регистре состояний прерываний IS). Также флаг TRPF может быть выставлен программно, установкой бита STRPF регистра установки прерываний ISS.

Установка флага TRPF повлечет за собой аппаратную установку бита TRPS регистра IS посредством блока управления установкой/сбросом.

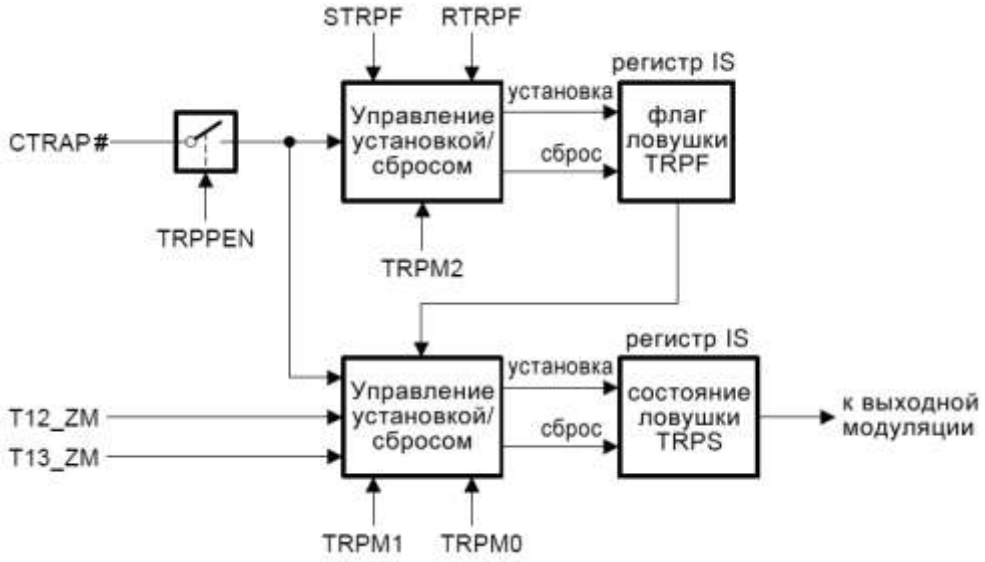


Рисунок 18.60 – Блок обработки состояния ловушки

При условии, что TRPPEN = 1_B и CTRAP# = 0_B, TPRF и TRPS остаются равными «1» и не могут быть очищены.

Сброс TRPF контролируется битом управления режимом ловушки TRPM2 (расположен в регистре управления ловушками TRPCTR). Если TRPM2 = 0_B, см. рисунок 18.61, флаг TRPF сбрасывается аппаратно, как только CTRAP# становится неактивным, т. е. равным «1». В случае если TRPM2 = 1_B, флаг TRPF должен быть сброшен программно после того, как CTRAP# перейдет в неактивное состояние.

Сброс TRPS контролируется битами управления режимом ловушки TRPM1 и TRPM0 регистра TRPCTR. Сброс TRPS выводит блок CAPCOM6 из состояния ловушки и переводит его в нормальный режим работы.

В зависимости от комбинации битов TRPM1 и TRPM0 выход из состояния ловушки происходит по-разному:

- немедленно после очистки флага TRPF;
- синхронно с периодом счета таймера T12 или T13 только после очистки флага TRPF.

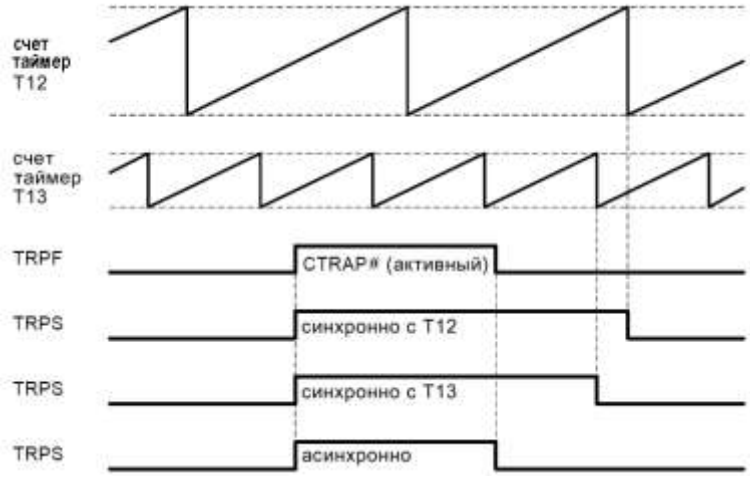


Рисунок 18.61 – Синхронизация в состоянии ловушки при TRPM2 = 0_B

Регистр ловушки

Регистр CCU6_TRPCTR, см. рисунок 18.62, таблицу 18.27, управляет функционированием ловушки. Он содержит независимые биты, управляющие разрешением возможности попадания в ловушку каждой линии каналов выходных сигналов, и биты управления режимами ловушки.

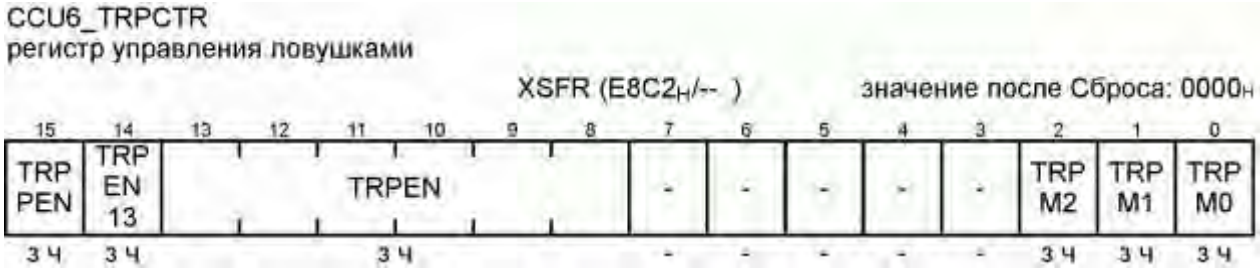


Рисунок 18.62– Формат регистра CCU6_TRPCTR

Таблица 18.27 – Функциональное назначение полей регистра CCU6_TRPCTR

Поле	Биты	Тип	Описание
1	2	3	4
TRPPEN	15	Чтение Запись	Бит разрешения аппаратного срабатывания ловушки по сигналу CTRAP#: 0 Запрет срабатывания ловушки по сигналу CTRAP#. Состояние ловушки может быть инициировано только программно. 1 Разрешение срабатывания ловушки по сигналу CTRAP#. Состояние ловушки может быть инициировано как падением в низкий уровень сигнала CTRAP#, так и программно.
TRPEN13	14	Чтение Запись	Бит разрешения аппаратного срабатывания ловушки для канала таймера T13: 0 Запрет срабатывания ловушки независимо от состояния бита TRPS. 1 Разрешение срабатывания ловушки. В канале таймера T13 будет поддерживаться низкий уровень выходного сигнала, пока бит TRPS равен «1».

Окончание таблицы 18.27

1	2	3	4
TRPEN	13-8	Чтение Запись	Биты управления функциями ловушки для каждого из трех каналов таймера T12. Каждая линия выходного сигнала может независимо от других иметь разрешение или запрет на попадание в ловушку. Побитно. 0 Запрет попадания в ловушку. Линия функционирует независимо от бита TRPS. 1 Разрешение попадания в ловушку. На линии поддерживается низкий уровень сигнала, пока бит TRPS равен «1». TRPEN.5 – TRPEN.0 соответствует побитно слева направо COUT62, CC62, COUT61, CC61, COUT60, CC60.
TRPM2	2	Чтение Запись	Бит 2 управления режимом ловушки. Бит указывает, будет ли флаг TRPF сброшен аппаратно или же программно. 0 Флаг TRPF сбрасывается аппаратно, как только CTRAP# становиться неактивным, т. е. равным «1». 1 Флаг TRPF должен быть сброшен программно после того, как CTRAP# перейдет в неактивное состояние.
TRPM1, TRPM0	1, 0	Чтение Запись	Биты 1, 0 управления режимом ловушки. Комбинация битов управляет выходом из состояния ловушки, т. е. сбросом бита TRPS. После того, как флаг TRPF будет сброшен, бит TRPF будет очищен согласно правилу, которое определяется комбинацией битов TRPM1 и TRPM2. 00 Синхронизация по таймеру T12. Сброс бита TRPS произойдет, когда счетчик таймера T12 обнулится (T12_ZM). 01 Синхронизация по таймеру T13. Сброс бита TRPS произойдет, когда счетчик таймера T13 обнулится (T13_ZM). 10 Зарезервировано. 11 Нет синхронизации. Сброс бита TRPS произойдет сразу же после сброса флага TRPF.

18.13 Управление выходной модуляцией

Последний блок в цепи обработки сигналов – это блок логики управления выходной модуляцией. В дальнейшем шесть выходов таймера T12 (CC6x, COUT6x) будут рассмотрены отдельно от выхода CC63 таймера T13.

На рисунке 18.63 представлены шесть блоков управления модуляцией. К каждому из блоков подходят модулируемые сигналы и сигналы управления модуляцией:

- модулируемые сигналы CC6x_O и COUT6x_O приходят с логики выбора состояния, см. рисунок 18.17;

- модулируемый сигнал CC63_O, генерируемый в канале таймера T13, приходит с выхода логики выбора состояния, см. рисунки 18.42 и 18.17;

- сигнал управления модуляцией MCMPy приходит с регистра MCMOUT в случае, если включен мультиканальный режим, бит MCMEN = 1_B;

- бит ловушки TRPS.

Примечание – Сигналы CC6x_O/COUT6x_O, CC63_O и TRPS имеют индивидуальное управление для каждого из шести блоков. Сигналы MCMPy имеют один общий сигнал управления MCMEN.

Выход каждого из шести блоков управления модуляцией соединен с блоком выбора уровня, который определяет, будет ли на выход передаваться модулируемый сигнал или на выходе будет удерживаться низкий уровень сигнала, независимо от уровня модулируемого сигнала.

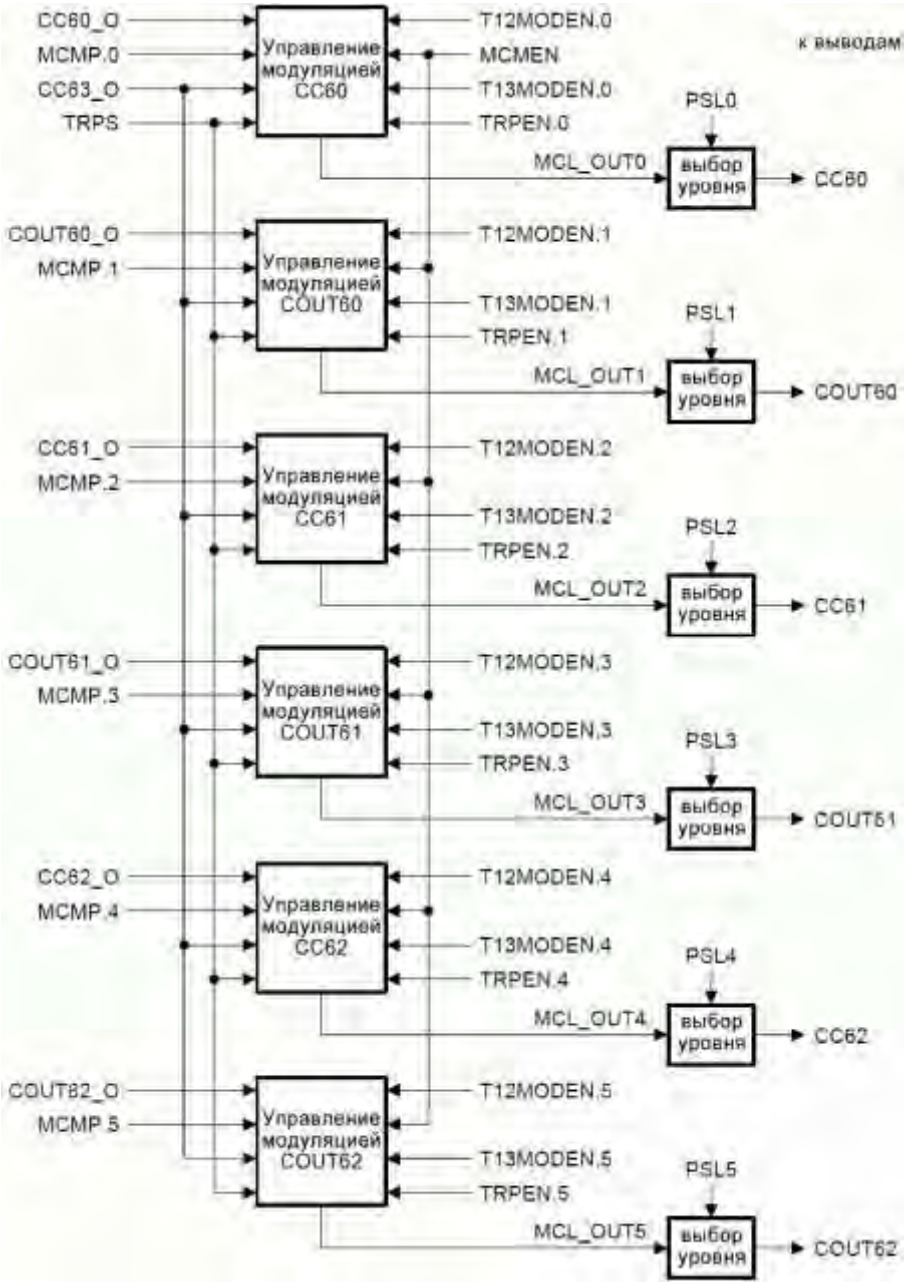


Рисунок 18.63 – Выходная модуляция

На рисунках 18.64, 18.65 представлена подробная схема одного из блоков управления модуляцией. Логика, объединяющая различные сигналы, построена так, чтобы

на линию MCL_OUTy мог передаваться только один модулируемый сигнал, разрешенный соответствующими сигналами управления.

Бит уровня выходного сигнала PSLy регистра уровня пассивного состояния PSLR, см. рисунок 18.66, таблицу 18.28, управляет выводом сигнала MCL_OUTy. Если бит PSLy равен «0», то соответствующий выходной сигнал MCLOUTy выводится в прямом виде, если PSLy равен «1», то выводится в инвертированном виде.

Биты PSLy имеют теневые биты, запись из которых происходит по сигналу теневой передачи T12_ST таймера T12. Обращение к регистру PSLR с целью записи/чтения происходит аналогично обращению к другим двухрегистровым (основной регистр – теневой регистр) структурам блока CAPCOM6.

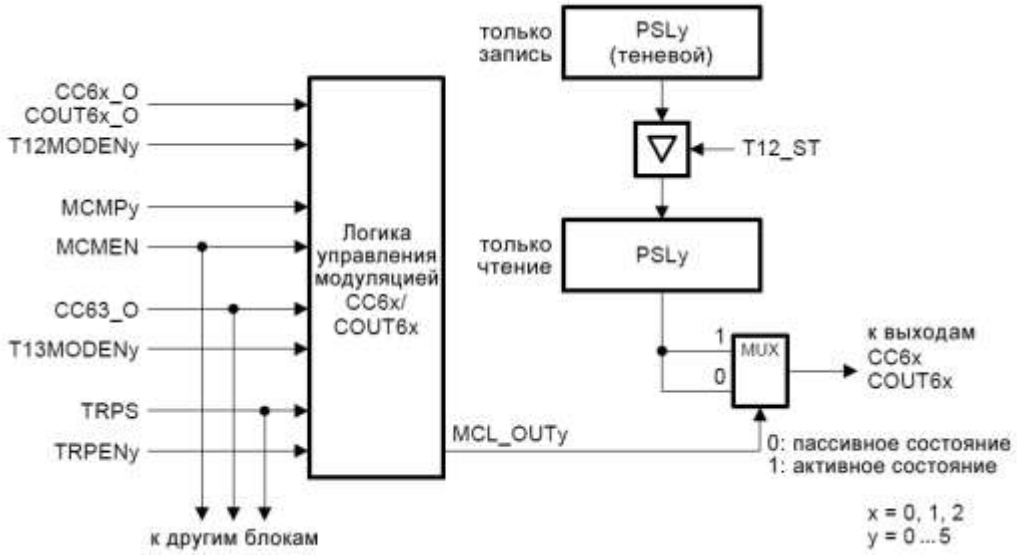


Рисунок 18.64 – Выходная модуляция в каналах таймера T12

Для канала таймера T13 построение схемы управления выводом модулируемого сигнала аналогично схеме каналов таймера T12.

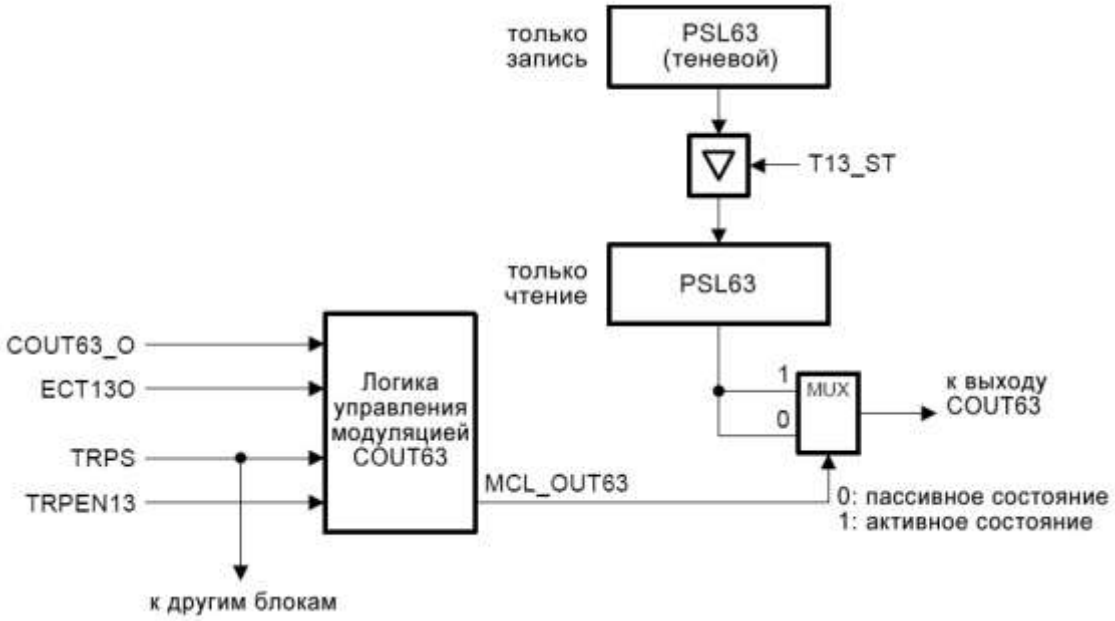


Рисунок 18.65 – Выходная модуляция в канале таймера T13

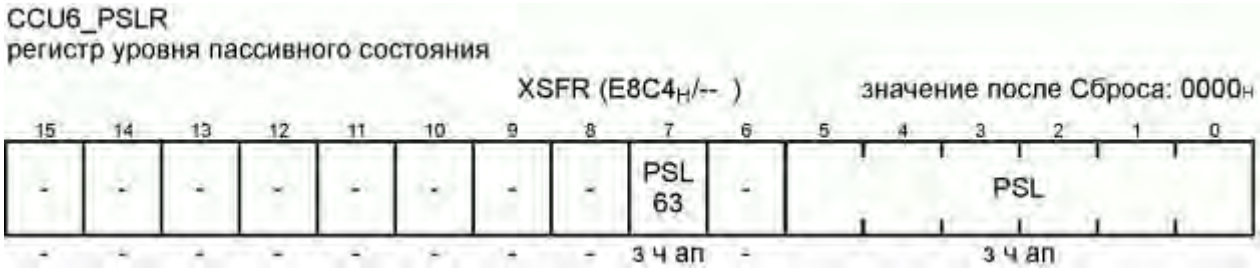


Рисунок 18.66 – Формат регистра CCU6_PSLR

Таблица 18.28 – Функциональное назначение полей регистра CCU6_PSLR

Поле	Биты	Тип	Описание
PSL63	7	Чтение Запись Аппаратное влияние	Бит уровня выходного сигнала канала таймера T13. Бит определяет, в каком виде будет передан на выход модулируемый сигнал. 0 Модулируемый сигнал передается в прямом виде. 1 Модулируемый сигнал передается в инвертированном виде.
PSL	5:0	Чтение Запись Аппаратное влияние	Поле управления уровнем выходных сигналов каналов таймера T12. Поле побитно определяет, в каком виде будет передан на выход соответствующий модулируемый сигнал. Побитно. 0 Модулируемый сигнал передается в прямом виде. 1 Модулируемый сигнал передается в инвертированном виде. PSLR.5-PSLR.0 соответствует (побитно слева направо) COUT62, CC62, COUT61, CC61, COUT60, CC60.

18.14 Двухрегистровые структуры и теневая передача

На рисунках 18.67 и 18.68 показаны двухрегистровые (основной регистр – теневой регистр) структуры и сигналы теневых передач каналов таймеров T12 и T13.

Наличие теневых регистров позволяет генерировать ШИМ сигнал и параллельно программно задавать новые параметры модуляции, которые могут вступать в силу сразу с приходом очередного переключения счетчика таймера.

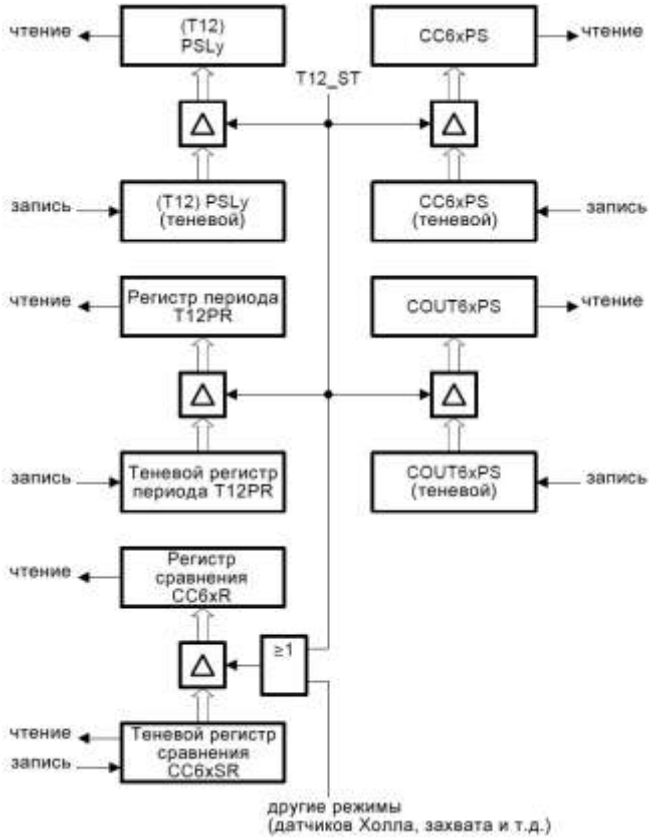


Рисунок 18.67 – Двухрегистровые структуры и сигнал теневой передачи каналов таймера T12

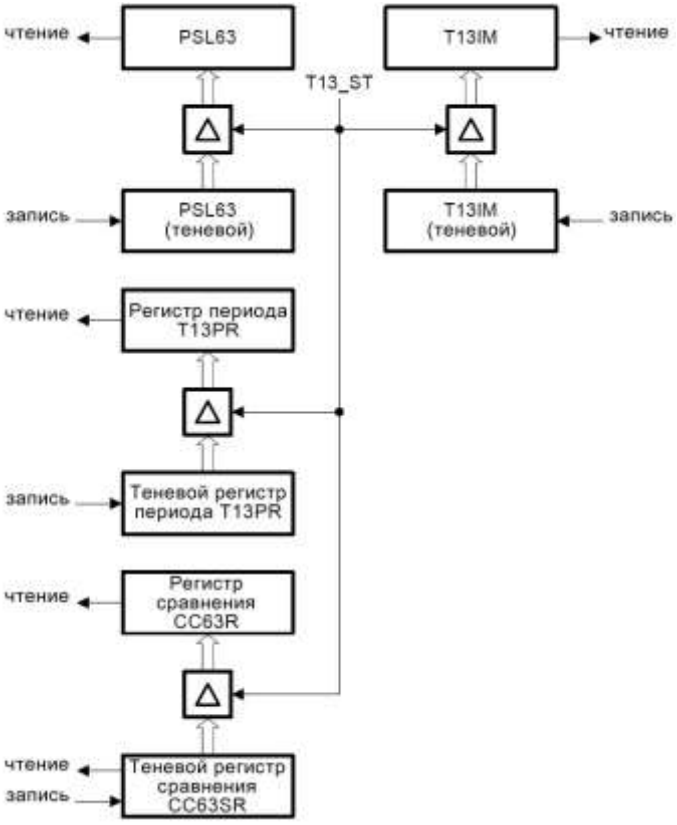


Рисунок 18.68 – Двухрегистровые структуры и сигнал теневой передачи канала таймера T13

18.15 Прерывания

Общая схема взаимодействия регистров прерываний показана на рисунке 18.69.

14 источников прерываний объединены попарно логикой управления прерываниями, имеющей четыре выхода I0, ..., I3. К каждому из четырех выходов подходит по семь линий прерываний (одна линия от каждой пары). Управление передачей сигналов запросов прерываний на один из четырех выходов осуществляется посредством регистра указателя узла прерываний INP.

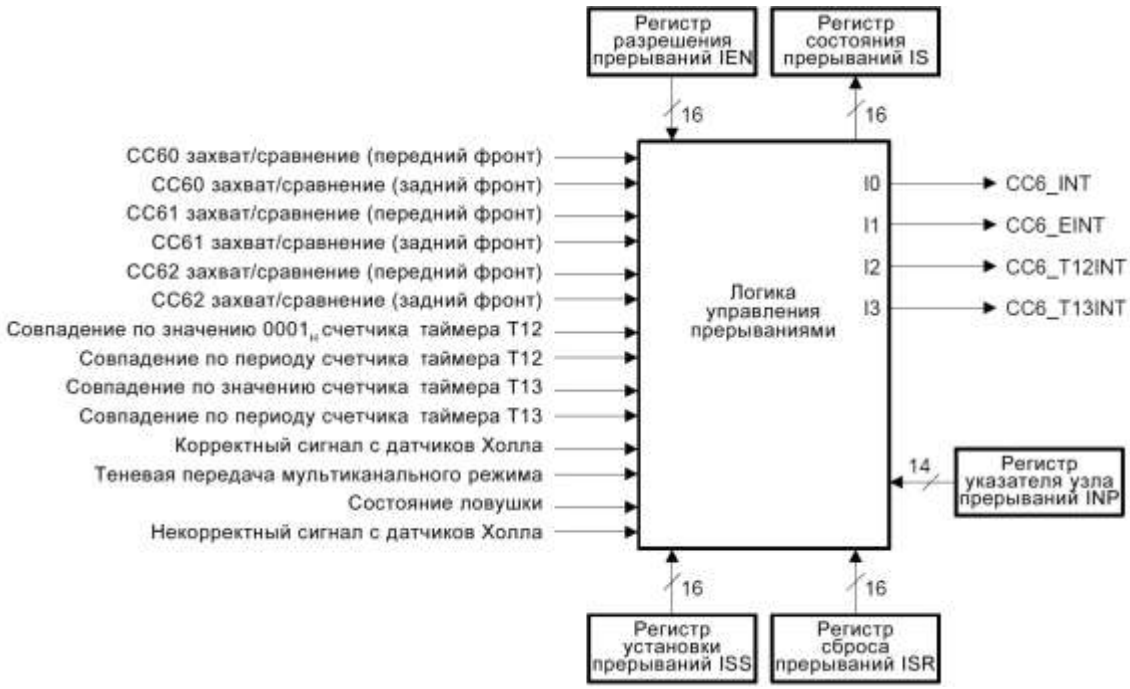


Рисунок 18.69 – Блок обработки прерываний

Подробная схема одного из семи узлов (одной из семи пар) прерываний показана на рисунке 18.70.

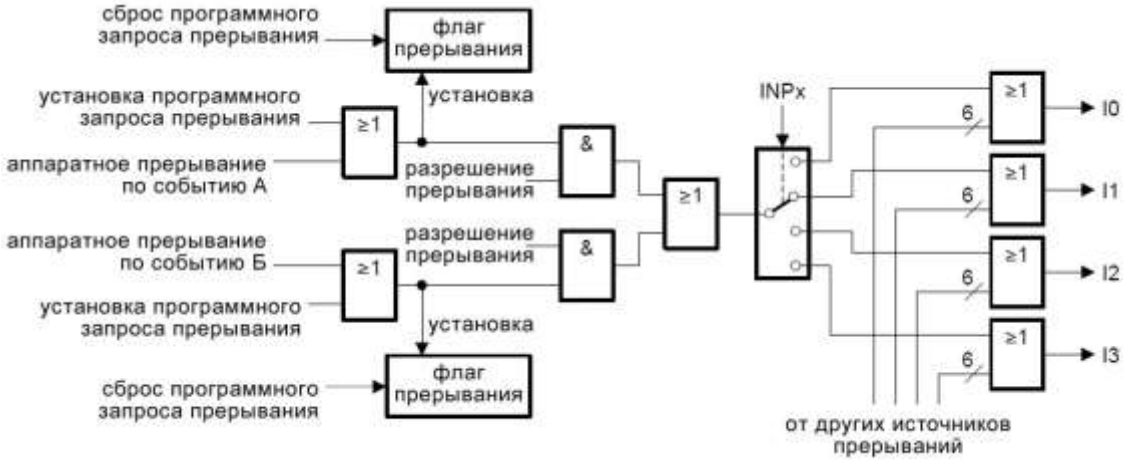


Рисунок 18.70 – Структура узла управления парой прерываний

Установка флагов прерываний в регистре IS может осуществляться как аппаратно (прерывания по событиям А и В), так и программно (установка соответствующего бита в регистре ISS).

Если разрешено соответствующими битами регистра IEN, запросы на прерывания генерируются каждый раз, когда возникает аппаратное и/или программное прерывание независимо от состояний соответствующих флагов прерывания в регистре IS. Оба запроса на прерывание объединены по «ИЛИ». Соответствующее поле регистра INP указывает, на какой из четырех выходов I0, …, I3 передать запрос на прерывание.

Флаги прерываний в регистре IS могут быть сброшены только программно, записью соответствующих битов в регистре ISR.

Регистры прерываний

Регистр IS содержит флаги прерываний. Этот регистр доступен только для чтения. Программная запись в регистр невозможна. Программно можно устанавливать или сбрасывать побитно флаги прерываний только посредством регистров ISS (установка битов) и ISR (сброс битов), см. рисунки 18.71 – 18.73, таблицы 18.29 – 18.31.

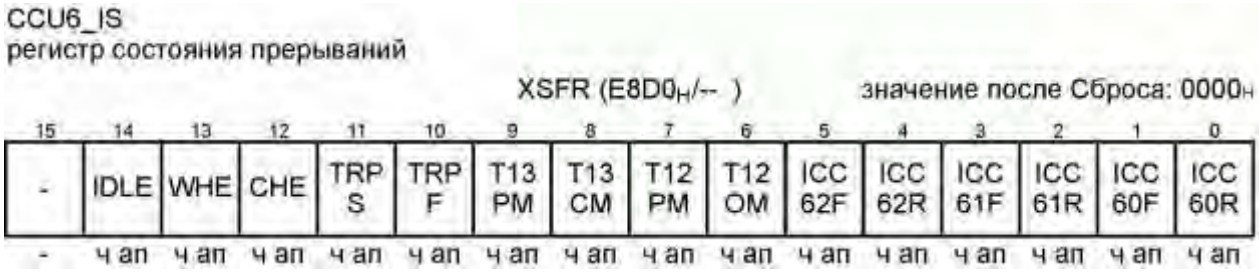


Рисунок 18.71 – Формат регистра CCU6_IS

Таблица 18.29 – Функциональное назначение полей регистра CCU6_IS

Поле	Биты	Тип	Описание
1	2	3	4
IDLE	14	Чтение Аппаратное влияние	Флаг режима ожидания Idle. Если разрешено ENIDLE = 1_B, этот флаг устанавливается совместно WHE. 0 Бездействие. 1 Поле MCMP очищается, на выходах каналов таймера T12 поддерживается низкий уровень сигнала.
WHE	13	Чтение Аппаратное влияние	Флаг некорректного сигнала датчика Холла: 0 Бездействие. 1 Обнаружение некорректной комбинации сигналов датчиков.
CHE	12	Чтение Аппаратное влияние	Флаг корректного сигнала датчика Холла: 0 Бездействие. 1 Обнаружение корректной комбинации сигналов датчиков.
TRPS	11	Чтение Аппаратное влияние	Флаг перехода в состояние ловушки: 0 Попадания в ловушку нет (флаг TRPF не выставлен). 1 Переход в состояние попадания в ловушку.
TRPF	10	Чтение Аппаратное влияние	Флаг попадания в ловушку: 0 Аппаратного или программного попадания в ловушку нет. 1 Аппаратное (CTRAP# = 0_B) или программное попадание в ловушку.

Окончание таблицы 18.29

1	2	3	4
T13PM	9	Чтение Аппаратное влияние	Флаг совпадения по периоду счетчика таймера T13: 0 Совпадения по периоду еще не было. 1 Совпадение по периоду произошло.
T13CM	8	Чтение Аппаратное влияние	Флаг совпадения по значению счетчика таймера T13: 0 Совпадения по значению еще не было. 1 Совпадение по значению произошло.
T12PM	7	Чтение Аппаратное влияние	Флаг совпадения по периоду при счете «вверх» счетчика таймера T12: 0 Совпадения по периоду еще не было. 1 Совпадение по периоду (при счете «вверх») произошло.
T12OM	6	Чтение Аппаратное влияние	Флаг совпадения с единицей (при счете «вниз») счетчика таймера T12: 0 Совпадения с единицей еще не было. 1 Совпадение с единицей (при счете «вниз») произошло.
ICC62F ICC61F ICC60F	5 3 1	Чтение Аппаратное влияние	Флаг события 1: - появление ожидаемого заднего фронта сигнала в случае режима захвата; - совпадения по значению при счете «вниз» счетчика таймера T12 в случае режима сравнения. В режиме сравнения флаг будет выставлен, когда возникнет совпадение по значению в течение времени декрементирования счетчика таймера T12. В режиме захвата флаг будет выставлен, когда на входе CC6x, x = 0, 1, 2, появится ожидаемый задний фронт (перепад из «1» в «0») входного сигнала. 0 Событие не произошло. 1 Ожидаемое событие произошло.
ICC62R ICC61R ICC60R	4 2 0	Чтение Аппаратное влияние	Флаг события 2: - появление ожидаемого переднего фронта сигнала в случае режима захвата; - совпадения по значению при счете «вверх» счетчика таймера T12 в случае режима сравнения. В режиме сравнения флаг будет выставлен, когда возникнет совпадение по значению, в течение времени инкрементирования счетчика таймера T12. В режиме захвата флаг будет выставлен, когда на входе CC6x, x = 0, 1, 2, появится ожидаемый передний фронт (перепад из «0» в «1») входного сигнала. 0 Событие не произошло. 1 Ожидаемое событие произошло.

Примечание – В режиме сравнения (и режиме датчиков Холла) аппаратные прерывания таймера могут генерироваться только в течение работы таймера TxR = 1_B. В режиме захвата, аппаратные прерывания могут генерироваться также и во время, когда таймер T12 остановлен.

Регистры ISS и ISR программно могут быть только записаны. Чтение битов этих регистров всегда возвращает нули. Запись «1» в биты регистра ISS устанавливает соответствующие флаги в регистре IS и (если разрешено соответствующими битами регистра IEN) создает запросы на прерывания (аналогично аппаратным запросам). Запись «1» в биты регистра ISR сбрасывает соответствующие флаги в регистре IS.

Биты регистров ISS и ISR очищаются аппаратно. Запись «0» в любой бит регистров ISS и ISR невозможна.

CCU6_ISS
регистр установки состояния прерываний

XSFR (E8D0$_H$/--) значение после Сброса: 0000$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	S IDLE	S WHE	S CHE	-	S TRPF	S T13PM	S T13CM	S T12PM	S T12OM	S CC62F	S CC62R	S CC61F	S CC61R	S CC60F	S CC60R
-	3	3	3	-	3	3	3	3	3	3	3	3	3	3	3

Рисунок 18.72 – Формат регистра CCU6_ISS

Таблица 18.30 – Функциональное назначение полей регистра CCU6_ISS

Поле	Биты	Тип	Описание
SIDLE	14	Запись	Установка флага режима ожидания Idle
SWHE	13	Запись	Установка флага некорректного сигнала датчика Холла
SCHE	12	Запись	Установка флага корректного сигнала датчика Холла
STRPF	10	Запись	Установка флага перехода в состояние ловушки
ST13PM	9	Запись	Установка флага совпадения по периоду счетчика таймера T13
ST13CM	8	Запись	Установка флага совпадения по значению счетчика таймера T13
ST12PM	7	Запись	Установка флага совпадения по периоду при счете «вверх» счетчика таймера T12
ST12OM	6	Запись	Установка флага совпадения с единицей при счете «вниз» счетчика таймера T12
SCC62F SCC61F SCC60F	5 3 1	Запись	Установка флага события 1
SCC62R SCC61R SCC60R	4 2 0	Запись	Установка флага события 2

CCU6_ISR
регистр сброса состояния прерываний

XSFR (E8D4$_H$/--) значение после Сброса: 0000$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	R IDLE	R WHE	R CHE	-	R TRPF	R T13PM	R T13CM	R T12PM	R T12OM	R CC62F	R CC62R	R CC61F	R CC61R	R CC60F	R CC60R
-	3	3	3	-	3	3	3	3	3	3	3	3	3	3	3

Рисунок 18.73 – Формат регистра CCU6_ISR

Таблица 18.31 – Функциональное назначение полей регистра CCU6_ISR

Поле	Биты	Тип	Описание
RIDLE	14	Запись	Сброс флага режима ожидания Idle
RWHE	13	Запись	Сброс флага некорректного сигнала датчика Холла
RCHE	12	Запись	Сброс флага корректного сигнала датчика Холла
RTRPF	10	Запись	Сброс флага перехода в состояние ловушки
RT13PM	9	Запись	Сброс флага совпадения по периоду счетчика таймера T13
RT13CM	8	Запись	Сброс флага совпадения по значению счетчика таймера T13
RT12PM	7	Запись	Сброс флага совпадения по периоду при счете «вверх» счетчика таймера T12
RT12OM	6	Запись	Сброс флага совпадения с единицей при счете «вниз» счетчика таймера T12
RCC62F RCC61F RCC60F	5 3 1	Запись	Сброс флага события 1
RCC62R RCC61R RCC60R	4 2 0	Запись	Сброс флага события 2

Примечание – Установка/сброс битов в регистрах ISS и ISR может осуществляться посредством бит-операций (к примеру, BSET), логических операций (например, OR) или одиночной операции перемещения (например, выполнение посредством PEC).

Регистр CCU6_IEN, см. рисунок 18.74, таблицу 18.32, содержит биты разрешения прерываний и биты управления для выполнения аппаратного перехода в режим Idle в случае появления некорректного сигнала датчика CH_WHE.

Установка битов в «1» в этом регистре разрешает формирование соответствующих запросов на прерывания.

Очистка битов (запись в биты «0») запрещает формирование запросов на прерывание.

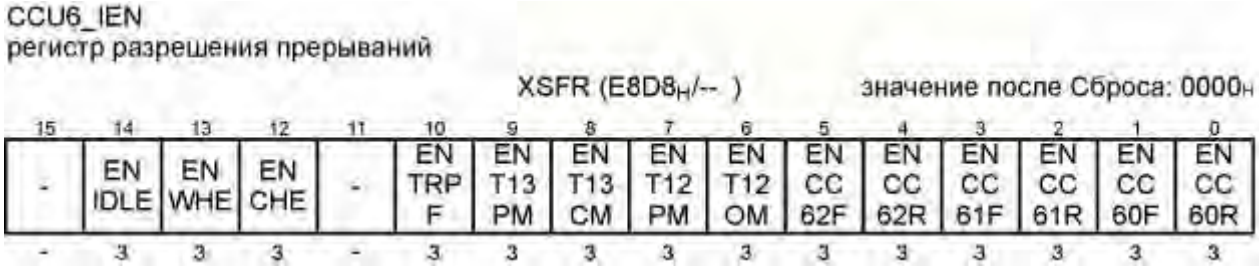


Рисунок 18.74 – Формат регистра CCU6_IEN

Таблица 18.32 – Функциональное назначение полей регистра CCU6_IEN

Поле	Биты	Тип	Описание
ENIDLE	14	Чтение Запись	Разрешение установки флага режима ожидания Idle
ENWHE	13	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала некорректного сигнала датчика Холла
ENCHE	12	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала корректного сигнала датчика Холла
ENTRPF	10	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала перехода в состояние ловушки
ENT13PM	9	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду счетчика таймера T13
ENT13CM	8	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по значению счетчика таймера T13
ENT12PM	7	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду при счете «вверх» счетчика таймера T12
ENT12OM	6	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения с единицей при счете «вниз» счетчика таймера T12
ENCC62F ENCC61F ENCC60F	5 3 1	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала события 1
ENCC62R ENCC61R ENCC60R	4 2 0	Чтение Запись	Разрешение формирования запроса на прерывания при появлении сигнала события 2

Источники прерываний модуля CAPCOM6 могут быть собраны программно в четыре узла прерываний (выходы I0, I1, I2, I3). Управляет этим регистр указателя узла прерываний INP, см. рисунок 18.75, таблицу 18.33.

Каждое битовое поле регистра INP указывает на один из четырех выходов (I0, …, I3), на который следует передавать запрос на прерывание от пары источников. Кодировки его битовых полей в таблице 18.34.

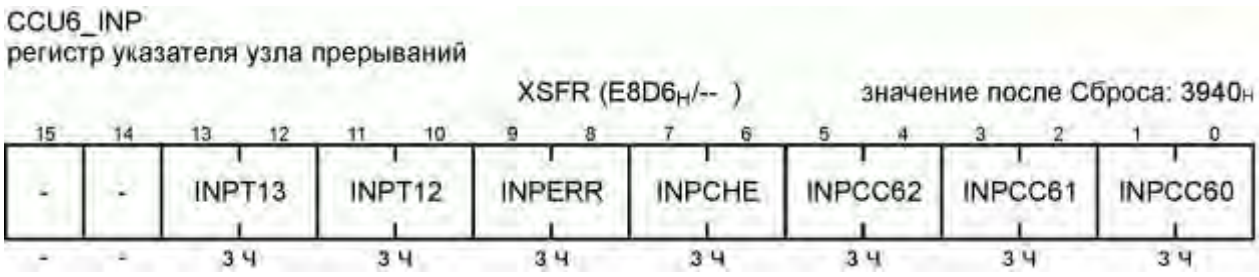


Рисунок 18.75 – Формат регистра CCU6_INP

Таблица 18.33 – Функциональное назначение полей регистра CCU6_INP

Поле	Биты	Тип	Описание
INPT13	13, 12	Чтение Запись	Указатель узла прерываний таймера T13. Источники формирования запроса на прерывание: - T13CM; - T13PM.
INPT12	11, 10	Чтение Запись	Указатель узла прерываний таймера T12. Источники формирования запроса на прерывание: - T12OM; - T12PM.
INPERR	9, 8	Чтение Запись	Указатель узла прерываний ошибок. Источники формирования запроса на прерывание: - TRPF; - WHE.
INPCHE	7, 6	Чтение Запись	Указатель узла прерываний теневых передач Источники формирования запроса на прерывание: - CHE; - MCM_ST.
INPCC62 INPCC61 INPCC60	5, 4 3, 2 1, 0	Чтение Запись	Указатель узла прерываний канала x (x = 0, 1, 2). Источники формирования запроса на прерывание: - CC6xR; - CC6xF.

Таблица 18.34 – Кодировка для полей регистра INP

Комбинация сигналов в поле INPxx	Выбранный выход
00_B	I0
01_B	I1
10_B	I2
11_B	I3

В таблице 18.35 отражены соответствия «по умолчанию» источников запросов на прерывания по узлам (выходам) и их соответствующим регистрам.

Таблица 18.35 – Соответствия «по умолчанию» источников запросов на прерывания по узлам (выходам) и их соответствующим регистрам

Источник прерывания	Выход	Управляющий регистр прерываний	Адрес регистра
Прерывания канал 0	I0	CCU6_IC	$F140_H$
Прерывания канал 1	I0		
Прерывания канал 2	I0		
Прерывания по корректному сигналу датчика	I1	CCU6_EIC	$F188_H$
Прерывания при возникновении ошибок	I1		
Прерывания таймера T12	I2	CCU6_T12IC	$F190_H$
Прерывания таймера T13	I3	CCU6_T13IC	$F198_H$

Все регистры контроля прерываний имеют однотипную структуру.

Рисунок 18.76 – Формат регистра CCU_xIC

Примечание – Для уточнения описания битовых полей следует обратиться к описанию регистра управления прерываниями.

18.16 Интерфейс блока CAPCOM6

Внутренние соединения

Четыре выходные линии запросов на прерывания блока CAPCOM6 соединены с блоком контроля прерываний, см. рисунок 18.77.

Сигнал совпадения по периоду таймера T13 (T13_PM) соединен с АЦП, как возможный источник переключений для входных преобразований.

Внешние соединения

Сигналы блока CAPCOM соединяются с портами ввода-вывода контроллера. Эти порты могут осуществлять захват сигналов из внешних источников, выводить сигналы сравнения во внешнюю цепь или принимать входные сигналы управления.

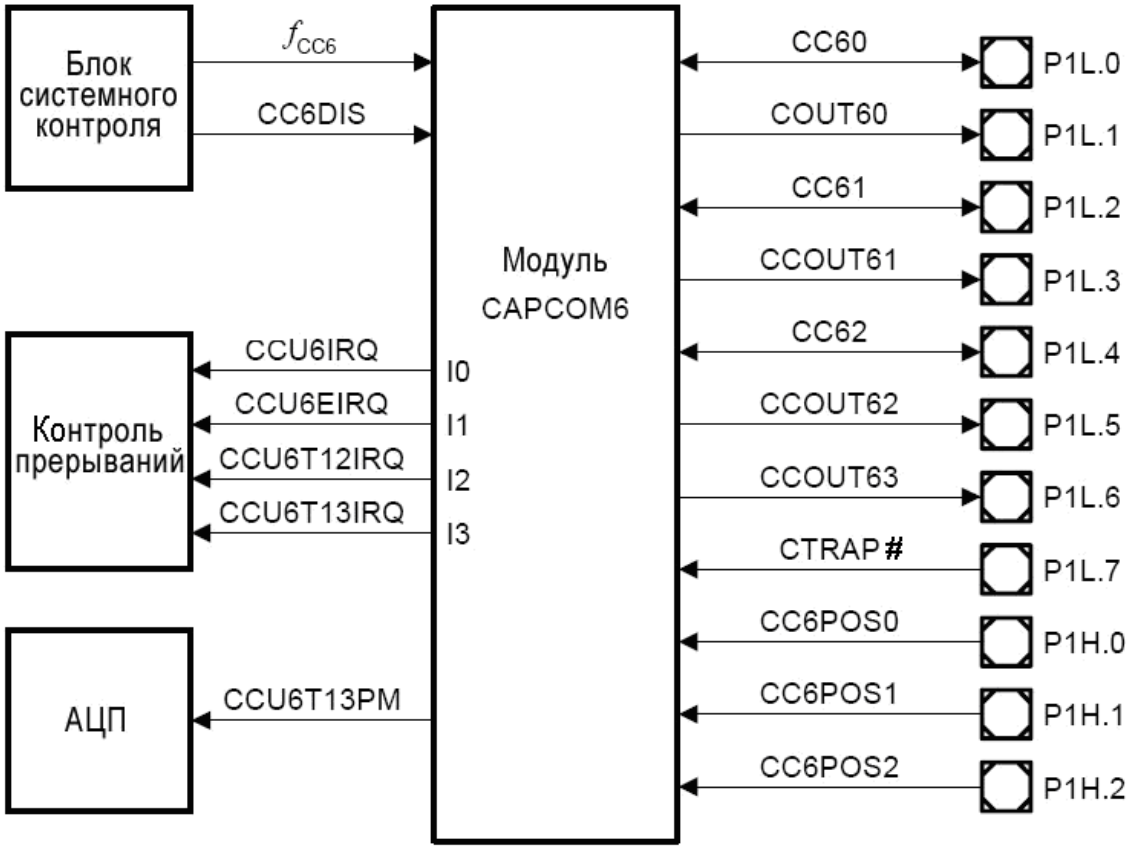


Рисунок 18.77 – Интерфейс блока CAPCOM6

19 Сторожевой таймер

Сторожевой таймер WDT позволяет перезагрузить контроллер при зависании программы управления или аппаратном сбое. При зависании программы управления сторожевой таймер сообщает о переполнении и инициализируется WDT-сброс.

Если WDT-сброс разрешен (по умолчанию) и программа регулярно реализует его, сторожевой таймер следит за выполнением программы. Также сторожевой таймер срабатывает, если программная ошибка возникает вследствие аппаратного сбоя. Это препятствует ложному срабатыванию по истечении установленного времени.

WDT-сброс сбрасывает центральный процессор, контроллер прерываний, контроллер внешней шины, блок управления и, собственно, сторожевой таймер.

Если WDT-сброс запрещен, при переполнении генерируется только прерывание. Это позволяет использовать сторожевой таймер как генератор периодических (по переполнению таймера) прерываний. Сторожевой таймер запускается командой SRVWDT.

Примечания

1 При переполнении сторожевой таймер автоматически перезагружается.

2 В случае разрешенного WDT-сброса генерируемый сброс имеет приоритет над стандартным механизмом перезагрузки.

19.1 Регистры сторожевого таймера

Сторожевой таймер состоит из двух 16-разрядных регистров: регистра управления WDTCON для инициализации и сброса и регистра таймера для хранения значения счета.

Операции сторожевого таймера управляются побитно адресуемым регистром управления WDTCON. Этот регистр устанавливает значение для перезагрузки старшего байта таймера, выбирает коэффициент деления входной частоты, а также выставляет флаг сброса.

WDTCON
регистр управления сторожевым таймером

SFR ($FFAE_H/D7_H$) значение после Сброса: $008x^{1)}_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTREL								WDT PRE	TIM EN	-	-	-	SW R	WDT R	WDT IN
-								з ч	з ч	-	-	-	ч ап	ч ап	з ч

1) x = $0xx1_B$

Рисунок 19.1 – Формат регистра WDTCON

Таблица 19.1 – Функциональное назначение полей регистра

Поле	Биты	Тип	Описание
1	2	3	4
WDTIN	0	Запись Чтение	Выбор делителя входной частоты (совместно с WDTPRE). Комбинации в таблице 19.2.
WDTR	1	Чтение Аппаратное влияние	Флаг WDT-сброса.

Окончание таблицы 19.1

1	2	3	4
SWR	2	Чтение Аппаратное влияние	Флаг программного сброса.
TIMEN	6	Запись Чтение	Бит разрешения таймера. Если этот бит установлен, то WDT-сброс запрещается после выполнения команды DISWDT. При переполнении сторожевого таймера генерируется запрос на прерывание.
WDTPRE	7	Запись Чтение	Выбор делителя входной частоты (совместно с WDTIN). Комбинации в таблице 19.2.
WDTREL	15-8	–	Значение перезагрузки сторожевого таймера.

Текущее значение счета сторожевого таймера хранится в регистре таймера (неадресуемый побитно), доступном только для чтения.

16-разрядный сторожевой таймер реализуется на основе объединения двух 8-разрядных таймеров. Старшие восемь битов сторожевого таймера программно доступны, и в них можно записать желаемое значение, что дает возможность изменения установленного времени срабатывания. При каждом таком программировании младшие восемь битов очищаются.

Сторожевой таймер считывает «вверх» (инкрементируется) с частотой, получаемой из частоты f_{PER} шины PDBUS, деленной на выбранный регистром WDTCON делитель.

Таблица 19.2 – Комбинации значений полей выбора делителя входной частоты

WDTIN	WDTPRE	Делитель
0_B	0_B	2
0_B	1_B	4
1_B	0_B	128
1_B	1_B	256

19.2 Функционирование сторожевого таймера

После любого сброса сторожевой таймер начинает считать «вверх» от значения 0000_H с частотой f_{WDT} (тактовая частота WDT), установленной по умолчанию $f_{WDT} = f_{PER}/256$.

Установленная по умолчанию тактовая частота f_{WDT} может быть изменена программируемым делителем.

Режимы работы сторожевого таймера

Режим WDT-сброса

Если сторожевой таймер не блокируется командой DISWDT, он продолжает счет, даже во время режима Idle. Если таймер достигает значения $FFFF_H$ (и нет обращения к таймеру посредством команды SRVWDT), он переполняется, что приводит к генерированию WDT-сброса и запроса прерывания WDTINT.

Происходит сброс контроллера и, собственно, сторожевого таймера.

Режим WDT-прерывания

Если установлен флаг TIMEN, то WDT-сброс запрещается после выполнения команды DISWDT. При переполнении сторожевого таймера генерируется запрос на прерывание.

Режим выключенного сторожевого таймера WDT

Режим возможен, если TIMEN = 0_B (регистр WDTCON) и генерирование WDT-сброса было остановлено исполнением команды DISWDT. В этом режиме сторожевой таймер не считает. Ни WDT-сброс, ни WDT-прерывание не будут сгенерированы. Команда DISWDT является защищенной 32-разрядной командой и может быть выполнена только в промежуток времени от сброса до выставления флага окончания инициализации (EINIT) или команды SRVWDT (программирование сторожевого таймера). Любая из этих двух команд блокирует выполнение DISWDT. WDT-сброс не завершит текущий цикл внешней шины до начала внутреннего сброса.

Избегать переполнения WDT следует программно, при помощи защищенной 32-разрядной команды SRVWDT. Периодическая загрузка в счетчик сторожевого таймера значения WDTREL из регистра WDTCON будет очищать младший байт регистра счетчика таймера и позволять вести непрерывный счет без переполнения счетчика. После каждой такой перезагрузки счетчик будет вести счет от значения (<WDTREL> × 2^8).

Примечание – Выполнение команды SRVWDT не зависит от выполнения команд EINIT и DISWDT.

Команда SRVWDT декодируется таким образом, что вероятность непреднамеренного обращения к сторожевому таймеру сведена к минимуму.

Период счета WDT до переполнения может быть запрограммирован двумя способами – изменением частоты тактирования счетчика посредством программируемого делителя входной частоты (биты WDTPRE и WDTIN регистра WDTCON), см. таблицу 19.2, и перезагрузкой счетчика значением WDTREL регистра WDTCON.

Период счета от перезагрузки счетчика до переполнения может быть вычислен

$$P_{WDT} = \frac{2^{(1 + <WDTPRE> + <WDTIN> \times 6)} \times (2^{16} - <WDTREL> \times 2^8)}{f_{PER}} \qquad (19.1)$$

Примечание – Желательно перезаписывать содержимое WDTCON каждый раз перед началом обращения к сторожевому таймеру.

В таблице 19.3 представлены комбинации флагов-индикаторов сброса

Таблица 19.3 – Комбинации флагов-индикаторов сброса

Событие	Флаг индикации сброса [1)]	
	SWR	WDTR
Аппаратный сброс	0	0
Программный сброс	1	0
WDT-сброс	0	1
Аппаратный сброс и программный сброс	0	0
Аппаратный сброс и WDT-сброс	0	0
Аппаратный сброс, программный сброс и WDT-сброс	0	0
Программный сброс и WDT-сброс	1	1

[1)] «1» – флаг установлен; «0» – флаг сброшен.

Аппаратный сброс

При аппаратном сбросе программный сброс и WDT-сброс блокируются и не обнаруживаются. Ни один из флагов SWR и WDTR не выставляется.

Программный сброс

Флаг SWR выставляется после сброса, произошедшего в результате выполнения команды SRST.

WDT-сброс

Флаг WDTR выставляется после сброса, произошедшего в результате переполнения счетчика сторожевого таймера.

20 Асинхронно/синхронные последовательные интерфейсы ASC0, ASC1

Асинхронно/синхронный последовательный интерфейс обеспечивает передачу данных между микроконтроллером и другими микроконтроллерами, микропроцессорами и периферией. Микроконтроллер 1887ВЕ3Т содержит два модуля последовательного интерфейса ASC0 и ASC1. Далее будет дано описание режимов работы только для одного модуля последовательного интерфейса – ASCx. Модуль ASC1 идентичен ASC0, за исключением адресации регистров, входящих в его структуру.

Особенности интерфейса:

- полнодуплексные асинхронные режимы:
  - 8- или 9-битные фреймы данных, LSB первый;
  - генерация/проверка бита четности;
  - один или два стоповых бита;
- скорость пересылки данных до 1,25 Мбод (частота f_{PER} = 20 МГц);
- многопроцессорный режим для автоматического определения байта адреса/данных;
- полудуплексный 8-битный синхронный режим:
  - скорость пересылки данных до 2,5 Мбод (частота f_{PER} = 20 МГц);
- двойная буферизация при передаче/приеме;
- генерация прерываний;
- определение ошибок четности, фрейма, переполнения.

На рисунке 20.1 показана связь ASC-интерфейса с ядром и выходными портами.

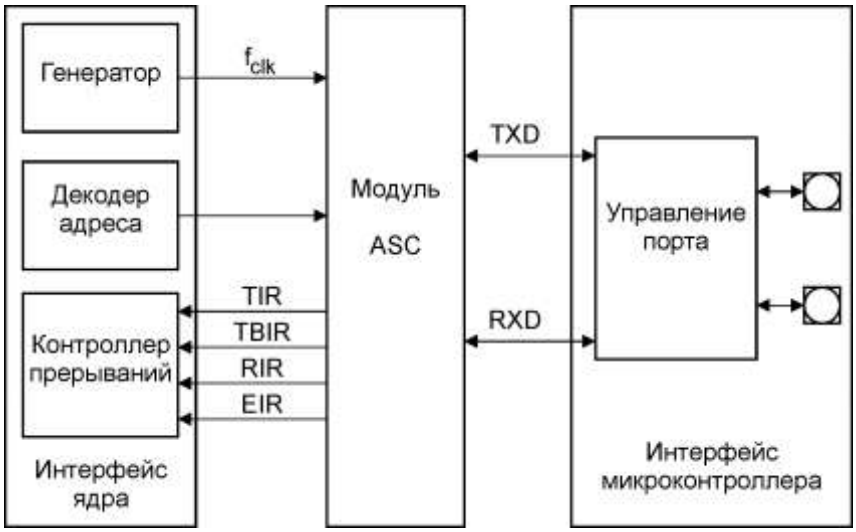


Рисунок 20.1 – ASC-интерфейс

Рисунок 20.2 – Регистры модулей ASC0 и ASC1

Все регистры модулей ASC0 и ASC1 расположены в адресном пространстве SFR/ESFR. Их адреса могут быть найдены в списке SFR.

Регистры S0CON и S1CON предназначены для управления режимами работы ASC0 и ASC1, соответственно.

20.1 Краткий обзор режимов работы

На рисунке 20.3 показана блок-схема интерфейса ASC и режимы его работы: асинхронный и синхронный режимы.

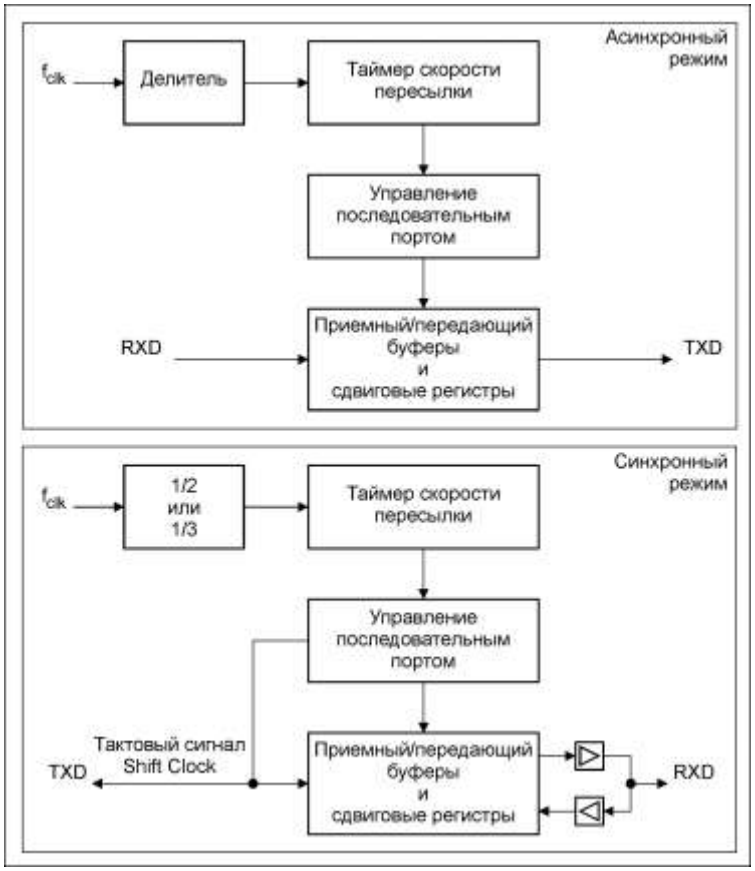


Рисунок 20.3 – Блок-схема интерфейса ASC

20.2 Общие операции

ASC поддерживает полнодуплексную асинхронную передачу до 1,25 Мбод и полудуплексную синхронную передачу до 2,5 Мбод (частота тактового генератора f_{PER} = 20 МГц). В синхронном режиме данные передаются или принимаются синхронно с тактовым сигналом, генерируемым микроконтроллером. В асинхронном режиме поддерживается 8 или 9-битовая передача данных, могут быть выбраны количество стоповых битов и бит четности. Обнаружение ошибок четности, синхронизации и ошибки переполнения увеличивают надежность передачи данных. При передаче и приеме данных используется двойная буферизация. Для использования в многопроцессорных системах поддерживается механизм распознавания байта адреса и байтов данных. 13-битовый таймер скорости пересылки данных с универсальной входной схемой делителя тактового генератора обеспечивает последовательный тактовый сигнал.

Передача начинается после записи в передающий буферный регистр SxTBUF. Выбранный операционный режим определяет число информационных разрядов, которые будут переданы, таким образом биты, записанные с девятой по 15 позиции регистра SxTBUF, никогда не передаются. После передачи данных буферный регистр очищается. Передача данных – с двойной буферизацией, поэтому новые данные могут быть записаны в буферный регистр прежде, чем передача предыдущих данных будет завершена. Это позволяет осуществлять передачу данных друг за другом без перерывов между ними.

Прием данных разрешается установкой бита REN в регистре SxCON. После завершения приема принятые данные могут быть прочитаны из приемного буферного регистра SxRBUF. Регистр SxRBUF является только читаемым, запись в данный регистр любого значения недействительна. Полученный бит четности также может считаться, если это разрешено операционным режимом. Старшие биты регистра SxRBUF, недействительные в выбранном операционном режиме, будут читаться как нули.

Прием данных – с двойной буферизацией, поэтому прием вторых данных может быть начат прежде, чем полученные ранее данные будут считаны из приемного буферного регистра. Обнаружение ошибки переполнения поддерживается во всех режимах и разрешается установкой бита OEN в регистре SxCON. Когда обнаружение ошибки переполнения разрешено, флаг состояния SxCON_OE и флаг запроса прерывания по ошибке переполнения EIR будут установлены в том случае, если буферный регистр не был прочтен до завершения получения следующих данных.

Режим loop-back (петлевой режим) используется для получения передаваемых в текущий момент данных в приемный буфер. Этот режим разрешается установкой бита SxCON_LB в регистре SxCON. Данный режим применяется для тестирования подпрограммы последовательной передачи данных на ранних стадиях разработки без необходимости создания внешней сети. В режиме loop-back необходимо использовать альтернативные функции выводов порта P3.

Примечание – Последовательная передача или прием данных возможны только в том случае, если установлен бит разрешения работы генератора скорости пересылки (baud rate) R регистра SxCON. В ином случае последовательный интерфейс не используется. Не следует программировать поле M регистра SxCON одной из зарезервированных комбинаций, чтобы избежать непредсказуемого поведения последовательного интерфейса.

Операционным режимом последовательного интерфейса управляет поле M в S0CON регистре, см. рисунки 20.4, 20.5, таблицу 20.1. Этот регистр содержит служебные биты для разрешения режимов определения ошибок и биты флагов состояния ошибок.

S0CON
регистр управления ASC0

SFR-b ($FFB0_H$/ $D8_H$) значение после Сброса:0000_H

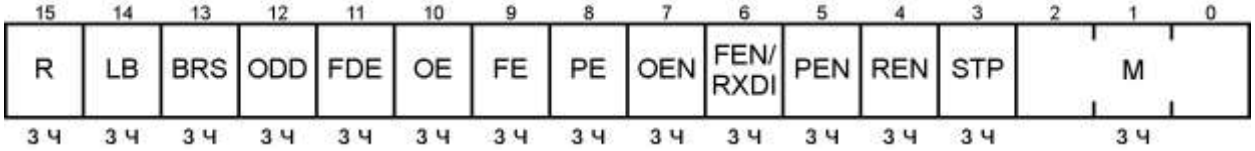


Рисунок 20.4 – Формат регистра S0CON

S1CON
регистр управления ASC1

ESFR-b ($F1B8_H$/ DC_H) значение после Сброса:0000_H

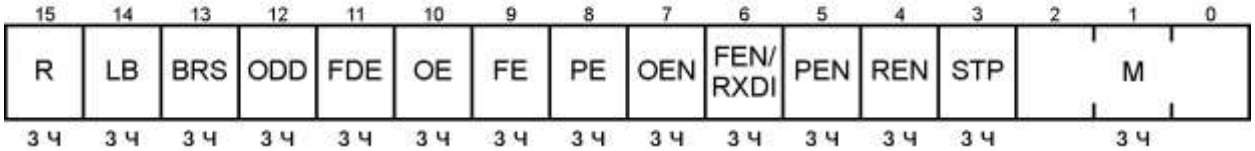


Рисунок 20.5 – Формат регистра S1CON

Таблица 20.1 – Функциональное назначение полей регистров S0CON и S1CON

Поле	Биты	Тип	Описание
1	2	3	4
M	2-0	Запись Чтение	Управление режимом работы ASC: 000 8-разрядные данные, синхронный режим. 001 8-разрядные данные, асинхронный режим. 010 Зарезервировано. 011 7-разрядные данные и четность, асинхронный режим. 100 9-разрядные данные, асинхронный режим. 101 8-разрядные данные и бит пробуждения, асинхронный режим. 110 Зарезервировано. 111 8-разрядные данные и четность, асинхронный режим.
STP	3	Запись Чтение	Выбор количества стоповых бит: 0 Один стоповый бит. 1 Два стоповых бита.
REN	4	Запись Чтение	Бит разрешения приема: 0 Прием запрещен. 1 Прием разрешен.
PEN	5	Запись Чтение	Бит включения проверки четности. Все асинхронные режимы. 0 Игнорировать проверку четности. 1 Проверять четность.
FEN/RXDI	6	Запись Чтение	Проверка ошибки фрейма (только асинхронный режим): 0 Игнорировать ошибки фрейма. 1 Проверять ошибки фрейма.

Окончание таблицы 20.1

1	2	3	4
OEN	7	Запись Чтение	Проверка ошибки переполнения: 0 Игнорировать ошибки переполнения. 1 Проверять ошибки переполнения.
PE	8	Запись Чтение	Флаг ошибки четности. Устанавливается аппаратными средствами при ошибке четности, если FEN = 1_B. Должен быть очищен программно.
FE	9	Запись Чтение	Флаг ошибки фрейма Устанавливается аппаратными средствами при ошибке фрейма, если OEN = 1_B. Должен быть очищен программно.
OE	10	Запись Чтение	Флаг ошибки переполнения. Устанавливается аппаратными средствами при ошибке переполнения, если OEN = 1_B. Должен быть очищен программно.
FDE	11	Запись Чтение	Бит включения дробного делителя: 0 Дробный делитель выключен. 1 Дробный делитель включен и используется как делитель для генератора скорости передачи.
ODD	12	Запись Чтение	Бит выбора четности: 0 По четным (проверка будет установлена на четное количество «1» в данных). 1 По нечетным (проверка будет установлена на нечетное количество «1» в данных).
BRS	13	Запись Чтение	Выбор скорости пересылки данных (baudrate) 0 Деление частоты генератора на запрограммированное значение + постоянная величина (зависящая от режима). 1 Дополнительное уменьшение частоты тактового генератора до 2/3.
LB	14	Запись Чтение	Бит разрешения режима loopback: 0 Режим стандартной передачи/получения данных. 1 Включен режим loopback.
R	15	Запись Чтение	Бит работы генератора baudrate: 0 Генератор baudrate отключен (ASC выключен). 1 Генератор baudrate включен.

Асинхронные операции

Асинхронный режим поддерживает полнодуплексную коммуникацию, в которой и передатчик и приемник используют одинаковый формат кадра данных и одинаковую скорость пересылки данных. Данные передаются в линии TXD и принимаются на линии RXD. На рисунке 20.6 представлена схема работы интерфейса ASC в асинхронном режиме.

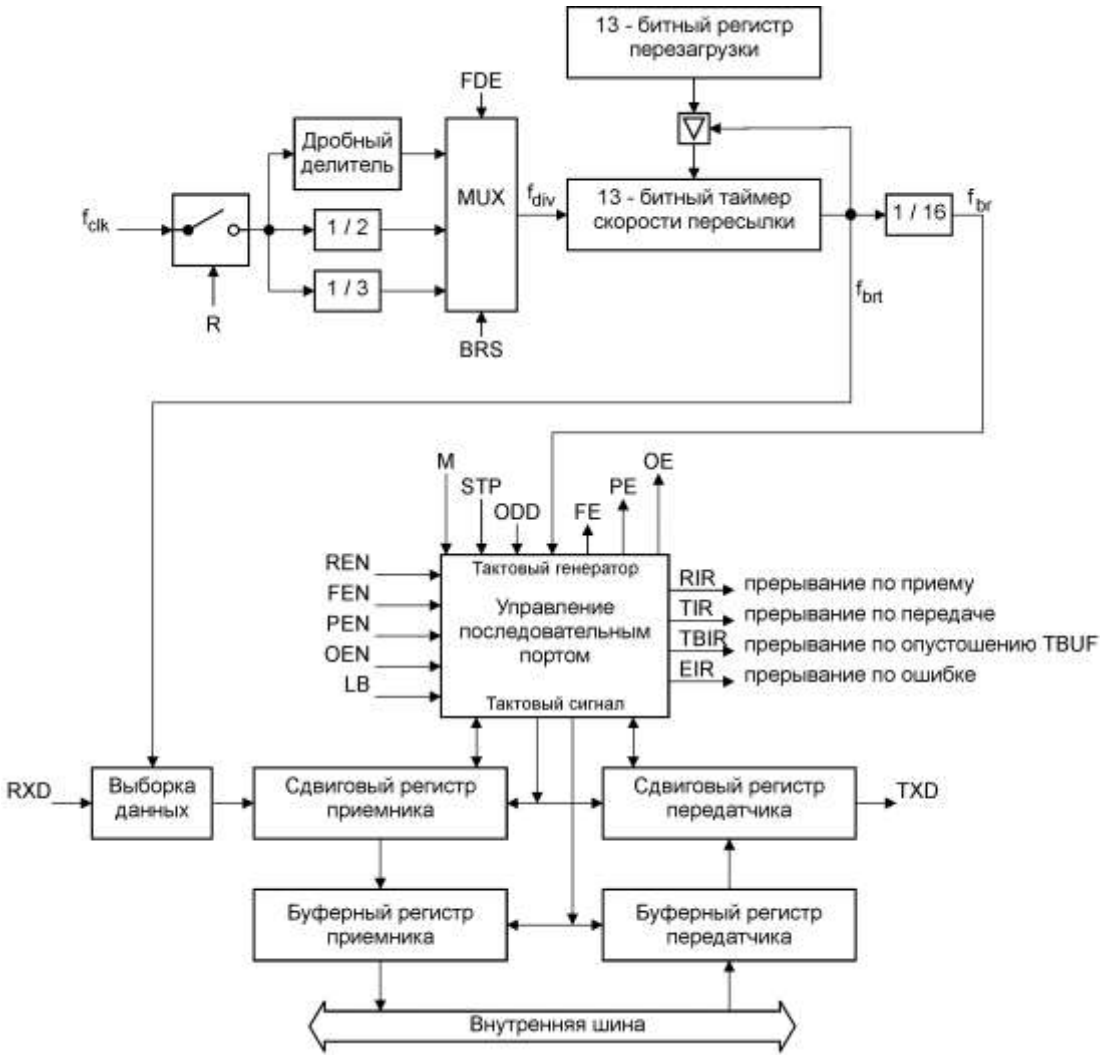


Рисунок 20.6 – Асинхронный режим работы интерфейса ASC

Асинхронные фреймы (кадры) данных

8-разрядные фреймы данных

8-разрядные фреймы данных состоят из любых восьми информационных бит данных D7, …, D0 (SxCON_M = 001_B) или из семи бит данных D6, …, D0 плюс автоматически генерируемого бита четности (SxCON_M = 011_B), см. рисунок 20.7. Бит четности может быть нечетным или четным, в зависимости от бита SxCON_ODD. Бит проверки на четность будет установлен, если сумма по модулю 2 семи информационных разрядов будет 1. Бит проверки на нечетность будет в этом случае очищен. Проверка четности разрешается установкой разряда SxCON_PEN (проверка всегда выключена в 8-битовом режиме данных). Флаг ошибки четности SxCON_PE будет установлен наряду с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит самой четности будет сохранен в разряде SxRBUF.7.

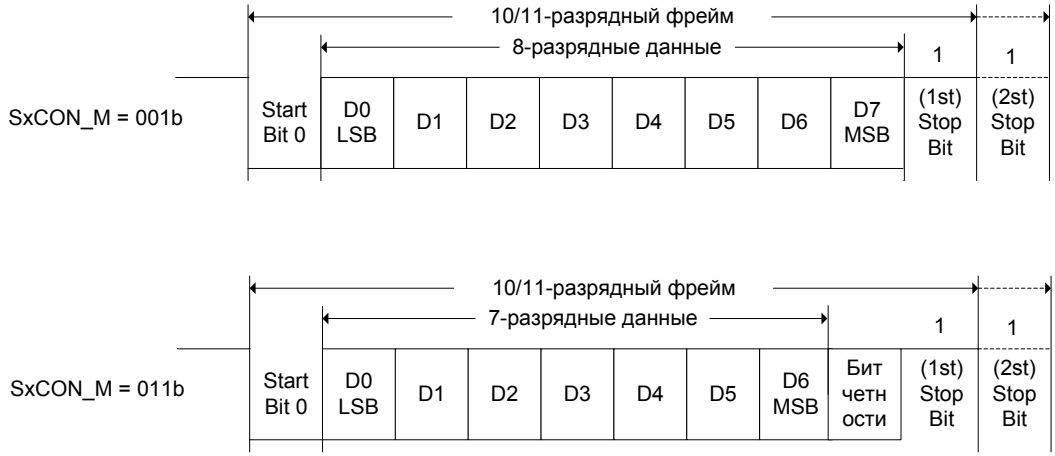


Рисунок 20.7 – Асинхронный 8-битный фрейм

9-разрядные фреймы данных

9-разрядные фреймы данных, см. рисунок 20.8, состоят из любых девяти информационных бит D8, …, D0 (SxCON_M = 100_B) или из восьми бит данных плюс автоматически генерируемого бита четности SxCON_M = 101_B. Четность может быть четной или нечетной, в зависимости от бита SxCON_ODD. Бит проверки на четность будет установлен, если сумма по модулю 2 восьми информационных разрядов будет 1. Бит проверки на нечетность будет в этом случае очищен. Проверка четности разрешается установкой разряда SxCON_PEN (проверка всегда выключена в 9-битовом режиме данных и режиме пробуждения). Флаг ошибки четности SxCON_PE будет установлен наряду с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит самой четности будет сохранен в разряде SxRBUF.7.

Рисунок 20.8 – Асинхронный 9-битный фрейм

В режиме пробуждения принимаемый фрейм передается в приемный буферный регистр, если только 9-й бит (бит побуждения) – «1». Если этот бит будет «0», запрос прерывания по приему не будет активирован и никакие данные не будут переданы.

Данный режим может быть использован в многопроцессорных системах.

Когда ведущий процессор хочет передать блок данных одному или нескольким ведомым, вначале посылается байт адреса, идентифицирующий подчиненный микроконтроллер. Байт адреса отличается от байта данных тем, что для адреса девятый бит устанавливается в «1», а для данных – в «0». Поэтому никакой ведомый микроконтроллер не будет прерван байтом данных. Байт адреса прервет только всех ведомых, работающих в режиме 8-битовых данных плюс бит пробуждения. Подчиненный микроконтроллер, для которого предназначается передача, переключится в 9-битный режим данных (путем очистки бита SxCON_M.0). После этого ведомый получит байт

данных, а все остальные подчиненные останутся в режиме 8 бит данных плюс бит пробуждения, и, таким образом, проигнорируют следующий байт данных.

Асинхронная передача

Асинхронная передача начинается после переполнения счетчика делителя на 16 таймера скорости пересылки данных f_{BR}, если бит SxCON_R установлен и данные были загружены в SxTBUF, см. рисунки 20.9, 20.10, таблицу 20.2. Переданный фрейм состоит из трех базовых элементов:

- стартового бита;
- поля данных (восемь или девять битов, LSB первый, включая бит четности, если выбрано);
- разделитель (один или два стоповых бита).

Передача данных с двойной буферизацией. Когда передатчик в состоянии простоя, передаваемые данные, загруженные в буферный регистр передатчика, немедленно перемещаются в передающий сдвиговый регистр, таким образом, освобождая буфер передатчика для следующих данных. После очистки буфера выставляется флаг запроса прерывания по опустошению буферного регистра TBIR. После этого буферный регистр загружается следующими данными, в то время как передача предыдущих данных продолжается. Запрос на прерывание по передаче TIR будет активизирован перед передачей последнего бита фрейма, т. е. перед первым или вторым стоповым битом данные будут стерты из сдвигового регистра.

S0TBUF
буферный регистр передатчика ASC0

SFR ($FEB0_H$/ 58_H) значение после Сброса:0000_H

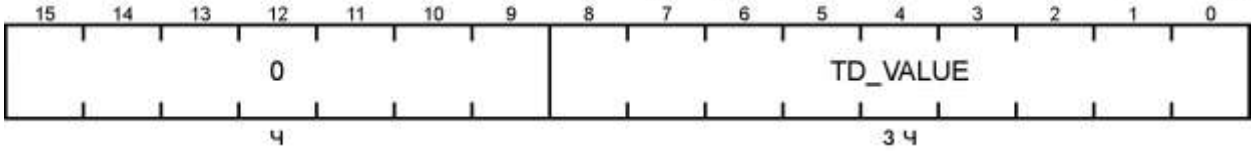


Рисунок 20.9 – Формат регистра S0TBUF

S1TBUF
буферный регистр передатчика ASC1

SFR ($FEA6_H$/ 53_H) значение после Сброса:0000_H

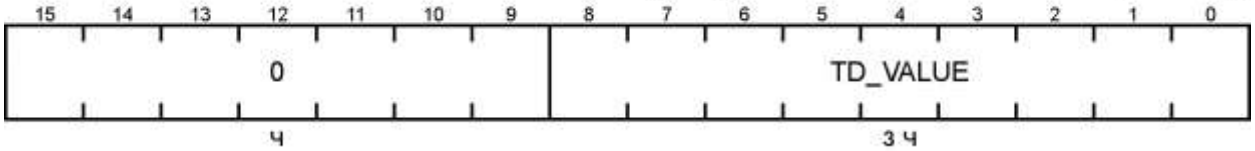


Рисунок 20.10 – Формат регистра S1TBUF

Таблица 20.2 – Функциональное назначение полей регистров S0TBUF и S1TBUF

Поле	Биты	Тип	Описание
TD_VALUE	8-0	Запись Чтение	Значение данных буферного регистра передатчика. Содержит данные, которые будут переданы в синхронном или асинхронном режимах ASC. Передача с двойной буферизацией.
0	15-9	Чтение	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна.

Примечание – Для передачи данных вывод TXD должен быть сконфигурирован как альтернативный выход порта.

Асинхронный прием

Асинхронный прием инициализируется падающим фронтом на линии RXD, при условии, что биты SxCON_R и SxCON_REN установлены. Измерение значения на входе данных RXD производится 16 раз за один такт генератора скорости пересылки данных. Значение большинства битов определяются на седьмом, восьмом или девятом измерении значения на входе. Такая схема позволяет добиваться безошибочных результатов.

Если обнаруженное значение не «0», когда бит начала выбран, принимаемый канал сбрасывается и ждет следующего переключения «1» в «0». Если стартовый бит оказывается действительным, то принимающий канал продолжает производить выборку и сдвиги данных, входящих во фрейм, в приемный сдвиговый регистр.

Когда последний стоповый бит принят, содержимое приемного сдвигового регистра передается в приемный буферный регистр данных SxRBUF приемника, см. рисунки 20.11, 20.12, таблицу 20.3. Одновременно с этим вырабатывается запрос на прерывание по приему, активируется линия RIR после девятого измерения в последнем стоповом бите вне зависимости от правильности принятого значения последнего стопового бита. После этого приемный канал переходит в режим ожидания следующего стартового бита (отрицательного фронта).

Примечание – Приемный вход RXD должен быть сконфигурирован как вход.

Асинхронный прием прекращается очисткой бита SxCON_REN. В этом случае прием данных запрещен, а принимаемые в текущий момент осуществляются в полном объеме, включая генерацию флага запроса прерывания по приему и прерыванию по ошибке.

Примечание – В режиме пробуждения фрейм данных передается в буферный регистр, только если девятый бит является битом пробуждения, т. е. «1». В противном случае запрос на прерывание не формируется, никакие данные не будут переданы.

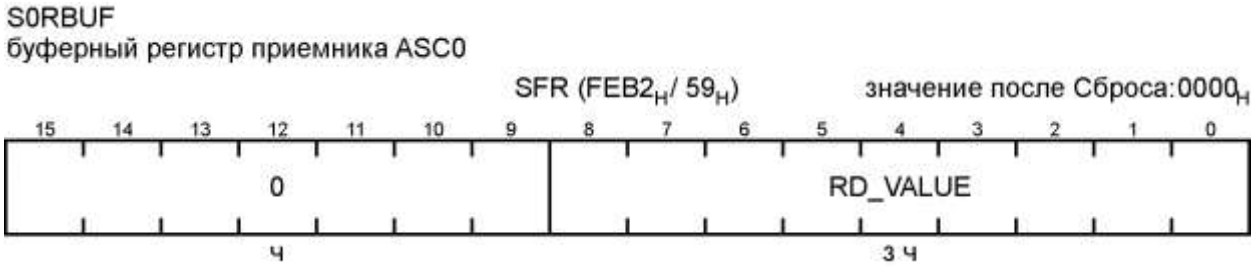


Рисунок 20.11 – Формат регистра S0RBUF

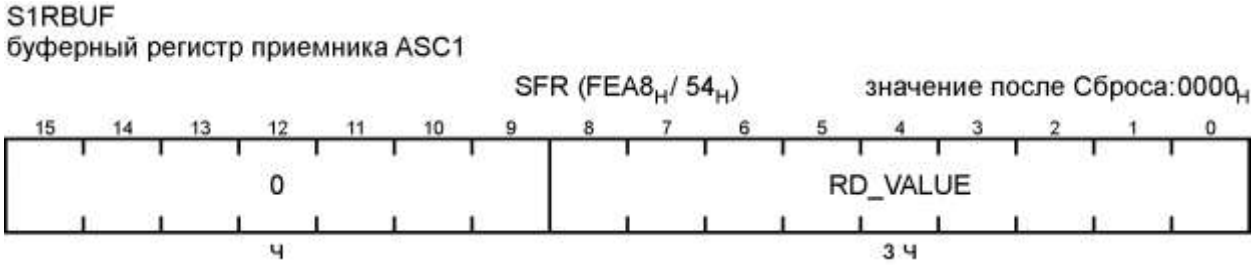


Рисунок 20.12 – Формат регистра S1RBUF

Таблица 20.3 – Функциональное назначение полей регистров S0RBUF и S1RBUF

Поле	Биты	Тип	Описание
RD_VALUE	8-0	Запись Чтение	Значение данных буферного регистра приемника. Содержит полученные данные и, в зависимости от выбранного режима, бит четности. В асинхронном режиме с CON_M = 011_B (7-битные данные плюс бит четности) полученный бит четности записывается в RD7. В асинхронном режиме с CON_M = 111_B (8-битные данные плюс бит четности) полученный бит четности записывается в RD8.
0	15-9	Чтение	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна.

Синхронные операции

Синхронный режим, см. рисунок 20.13, поддерживает полудуплексную связь, в основном, для простого расширения ввода-вывода через сдвиговые регистры. Данные передаются и принимаются через вывод RXD, при этом тактовый сигнал выводится через TXD. Эти сигналы являются альтернативными функциями порта. Для работы в синхронном режиме необходимо установить 000_B в битовом поле SxCON_M.

8 бит данных передаются и получаются синхронно тактовыми импульсами, генерируемыми внутренним генератором скорости пересылки baudrate. Тактовый генератор выдает тактовый сигнал только в случае передачи или приема данных.

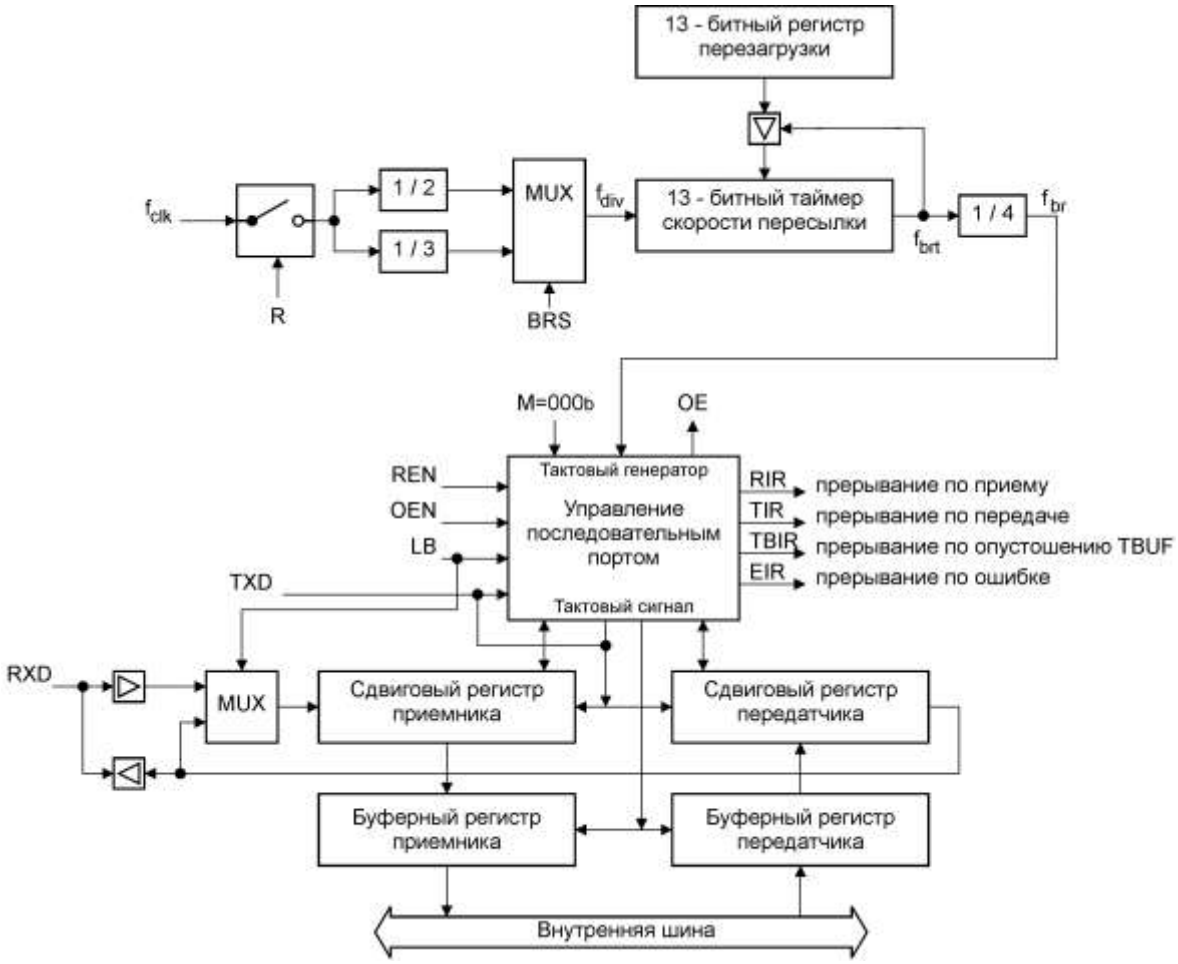


Рисунок 20.13 – Синхронный режим работы интерфейса ASC

Синхронная передача

Синхронная передача начинается в пределах четырех тактов, после того как данные были загружены в буферный регистр передачи SxTBUF, при условии, что SxCON_R установлен и SxCON_REN очищен (полудуплексный режим, нет приема). Исключение: в петлевом режиме (бит SxCON_LB установлен) SxCON_REN должен быть установлен для приема передаваемого байта. Передача данных с двойной буферизацией. Когда передатчик находится в состоянии простоя, данные, загруженные в буферный регистр SxTBUF, немедленно перемещаются в передающий сдвиговый регистр, таким образом, освобождая буферный регистр SxTBUF для следующих данных. Очистка буферного регистра SxTBUF сопровождается выставлением флага запроса на прерывание по очистке передающего буферного регистра TBIR. После этого буферный регистр SxTBUF может быть загружен следующими данными, в то время как передача предыдущих данных продолжается. Это позволяет осуществлять передачу данных без пауз. Информационные разряды передаются синхронно тактовому сигналу, выводимому на линию TXD. После завершения передачи 8 бита данных выставляется высокий уровень напряжения на выводах RXD и TXD, а также, устанавливается флаг запроса на прерывание по передаче TIR и прекращается передача данных.

Примечание – Для правильной работы в этом режиме необходимо правильно сконфигурировать выводы – вывод TXD должен быть настроен для дополнительного вывода данных, чтобы обеспечить вывод тактового сигнала, а RXD – должен быть сконфигурирован как вывод во время передачи данных.

Синхронный прием

Синхронный прием инициализируется установкой бита SxCON_REN. Если бит SxCON_R установлен, данные принимаются на RXD и поступают в сдвиговый регистр по импульсам тактового генератора, выводимым на TXD. После получения последнего бита данные из сдвигового регистра передаются в буферный приемный регистр SxRBUF. Вместе с этим будет установлен флаг запроса на прерывание по приему RIR, после этого бит SxCON_REN очищается.

Примечание – Вывод TXD должен быть сконфигурирован как выход.

Синхронный прием данных прекращается после очистки бита SxCON_REN. После этого будет завершен текущий прием данных, включая генерацию запросов на прерывание по ошибке (в случае необходимости). Запись нового значения в буфер передачи во время приема данных не приведет к началу передачи.

Если предыдущие данные не были прочитаны из буферного регистра до завершения передачи следующего байта, будет выставлен флаг запроса на прерывание EIR, а также установлен флаг SxCON_OE, если разрешена установка флага в бите SxCON_OEN.

Синхронизация

На рисунке 20.14 представлена временная диаграмма работы интерфейса в синхронном режиме передачи и приема данных. В неактивном состоянии уровень сигнала на выходе тактового генератора высокий. С началом синхронной передачи байта данных данные сдвигаются на линию RXD по падающему фронту тактового сигнала. В режиме синхронного приема данные принимаются на линии RXD по возрастающему фронту тактового сигнала. Между двумя последовательными передачами или приемами данных вставлен один тактовый цикл задержки.

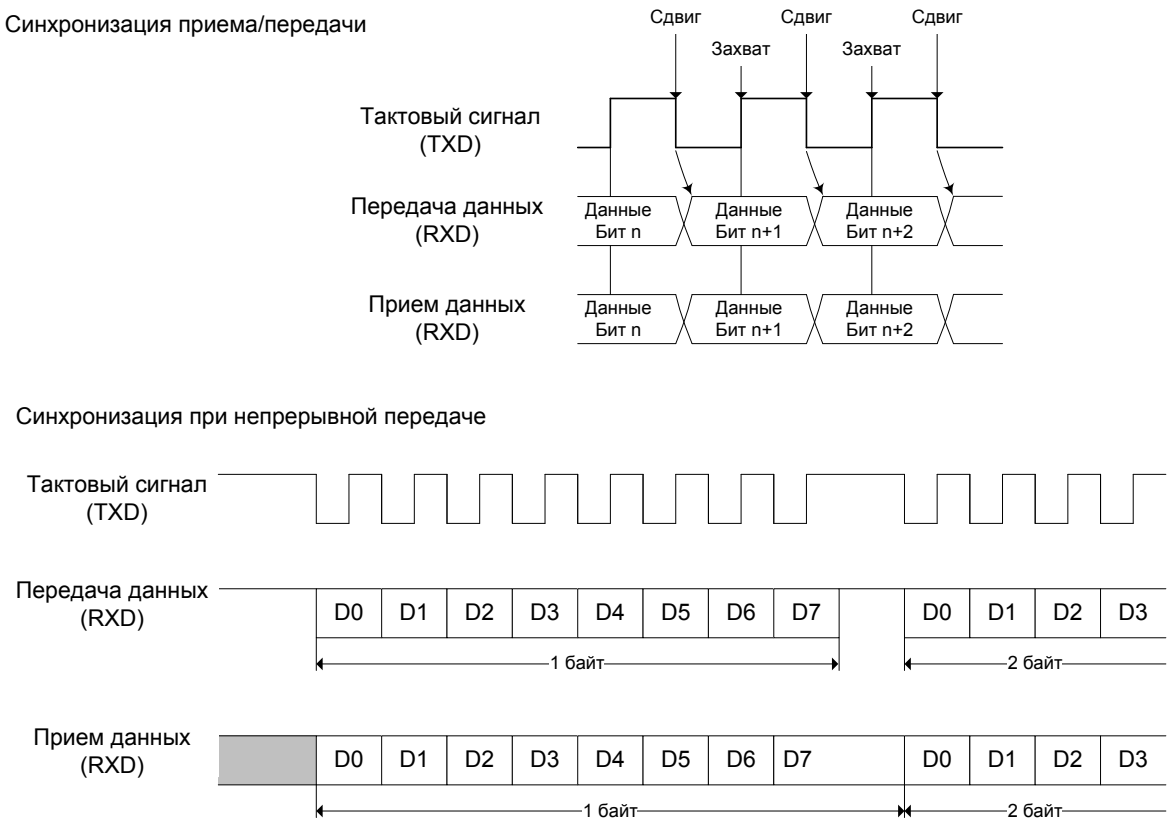


Рисунок 20.14 – Временная диаграмма работы интерфейса ASC в синхронном режиме приема/передачи

Генератор скорости пересылки данных

Последовательный интерфейс ASC имеет свой собственный специализированный 13-разрядный генератор скорости передачи с возможностью перезагрузки, позволяющий управлять скоростью передачи данных независимо от других таймеров.

Контроллер скорости передачи синхронизирован с тактовым сигналом, полученным делением входного опорного сигнала f_{CLK}. Счет таймера скорости пересылки данных происходит в обратном направлении и может быть начат или остановлен через бит управления SxCON_R. Каждое обнуление значения таймера скорости обеспечивает один тактовый импульс последовательному каналу. Таймер перезагружается значением, сохраненным в 13-битовом регистре перезагрузки, после каждого обнуления. Сформированный тактовый сигнал f_{BR} делится дополнительно на коэффициент, формируя тактовый сигнал скорости пересылки данных (±16 в асинхронных режимах и ±4 в синхронном режиме). Также результирующую частоту можно уменьшить, установив бит SxCON_BRS, при этом частота тактового сигнала уменьшается на 1/3. В асинхронных режимах работы интерфейса доступен дробный делитель частоты, который позволяет выбор отношений деления n/512 с n = 0, …, 511. Таким образом, скорость пересылки данных определяется тактовым генератором модуля, значением перезагрузки SxBG, содержимым SxFDV, значением бита SxCON_BRS и операционным режимом (асинхронный или синхронный).

Регистр SxBG, см. рисунки 20.15, 20.16, таблицу 20.4, является двухфункциональным регистром. Чтение SxBG возвращает содержимое таймера, биты 15, …, 13 возвращают 0, в то время как запись в регистр изменяет значения регистра перезагрузки, при этом биты 15, …, 13 игнорируются.

Автоперезагрузка таймера содержимым выполняется каждый раз, когда SxCON_BG установлен. Однако если SxCON_R очищен во время операции записи в SxCON_BG, значение таймера не будет перезагружено до тех пор, пока не будет разрешена работа

тактового генератора baudrate. Для правильной установки скорости передачи данные в регистр SxBG необходимо записывать только при сброшенном бите SxCON_R. Запись данных в регистр SxBG при установленном бите SxCON_R может привести к непредсказуемому поведению ASC.

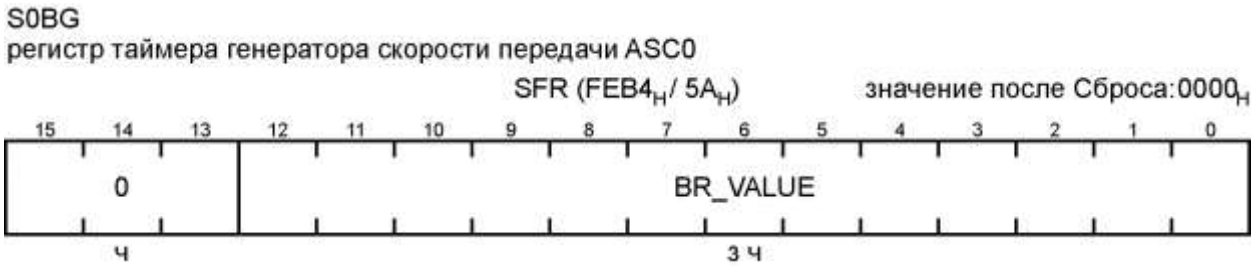


Рисунок 20.15 – Формат регистра S0BG

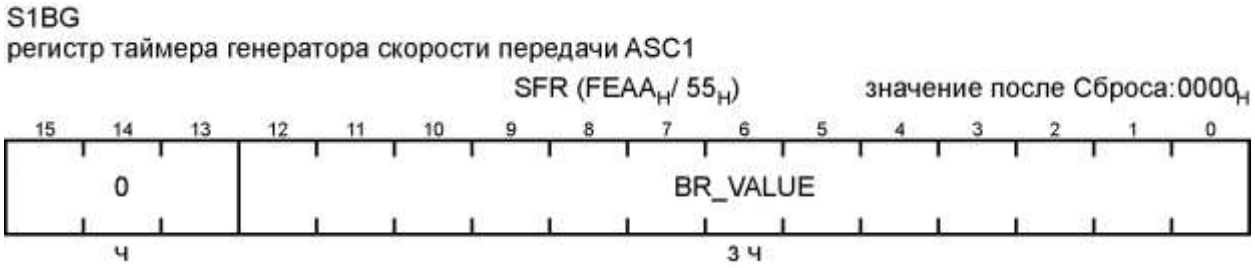


Рисунок 20.16 – Формат регистра S1BG

Таблица 20.4 – Функциональное назначение полей регистров S0BG и S1BG

Поле	Биты	Тип	Описание
BR_VALUE	12-0	Запись Чтение	Значение перезагружаемых данных. Чтение возвращает 13-битовое содержимое регистра Запись загружает значение в таймер. Примечание – BG должен быть записан только при CON_R = 0_B.
0	15-13	Чтение	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна.

Скорость пересылки данных в асинхронном режиме

Во время асинхронного приема данных измерение значения на входе данных RXD производится 16 раз за один такт генератора скорости пересылки данных. Значение принятого бита определяется на седьмом, восьмом или девятом измерении значения на входе. Схема делителя сигнала тактового генератора, которая формирует тактовый сигнал для 13-битового таймера скорости пересылки данных, расширена дробной схемой делителя, позволяющей точнее производить подстройку скорости пересылки данных и расширяющей диапазон скоростей пересылки данных.

Скорость пересылки данных зависит от следующих сигналов, битов и значений регистров:

- входного тактового сигнала f_{CLK};
- битов SxCON_FDE и SxCON_BRS, определяющих частоту тактового сигнала f_{DIV}, формируемого из входного тактового сигнала;
- значения регистра дробного делителя SxFDV, рисунки 20.17, 20.18, таблицу 20.5, если бит SxCON_FDE установлен (дробный делитель включен);
- значения 13-битового перезагружаемого регистра SxBG.

Тактовый сигнал скорости пересылки данных f_{BR} получен из тактового сигнала генератора f_{DIV} путем деления на 16. Регистр дробного делителя SxFDV, см. рисунки

20.17, 20.18, таблицу 20.5, содержит 9-битовое значение (дробный делитель используется только в асинхронных режимах).

S0FDV
регистр дробного делителя ASC0

SFR ($FEB6_H$/ $5B_H$) значение после Сброса:0000_H

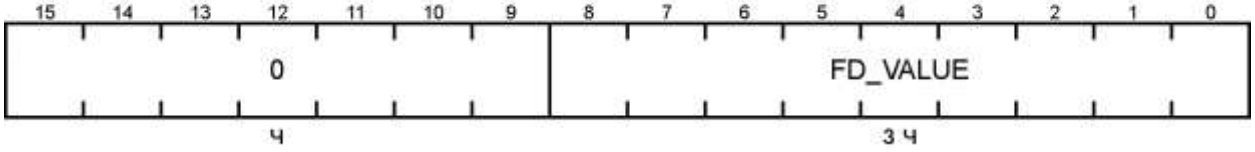


Рисунок 20.17 – Формат регистра S0FDV

S1FDV
регистр дробного делителя ASC1

SFR ($FEAC_H$/ 56_H) значение после Сброса:0000_H

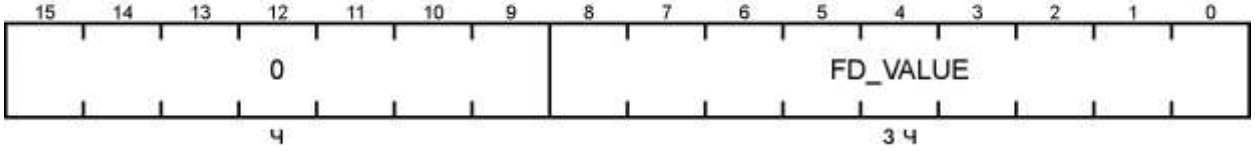


Рисунок 20.18 – Формат регистра S1FDV

Таблица 20.5 – Функциональное назначение полей регистров S0FDV и S1FDV

Поле	Биты	Тип	Описание
FD_VALUE	8-0	Запись Чтение	Значение дробного делителя. Определяет дробное отношение n/512, где n = 0, …, 512. При n = 0 дробный делитель выключен $f_{DIV} = f_{CLK}$.
0	15-9	Чтение	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна.

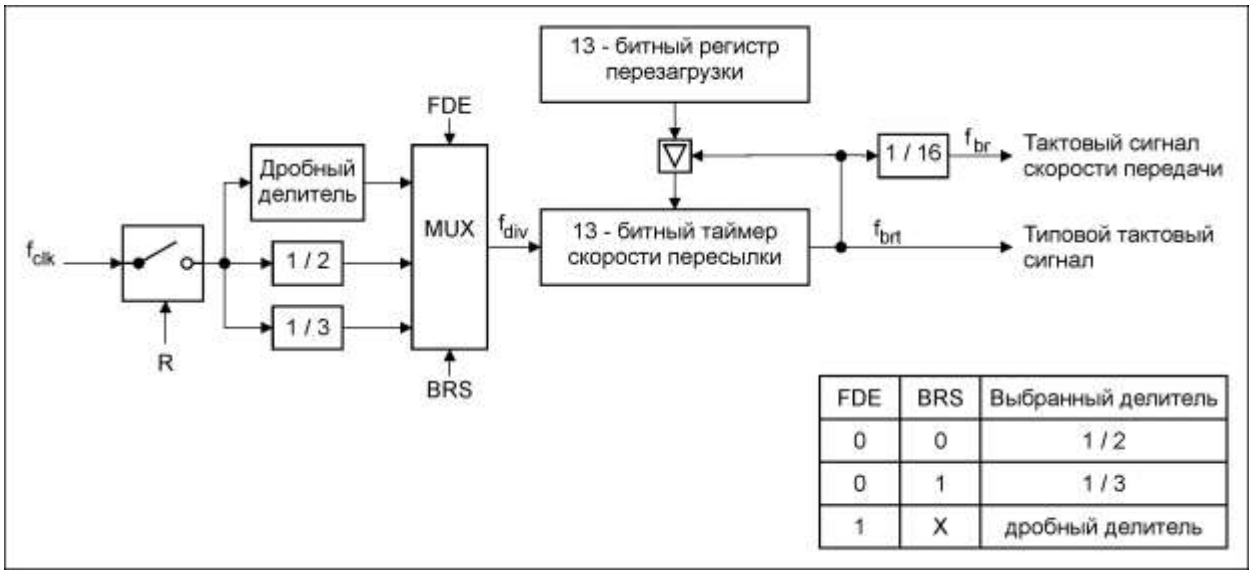


Рисунок 20.19 – Генератор скорости передачи в асинхронном режиме

Скорость пересылки для асинхронного режима, когда используется фиксированный тактовый сигнал SxCON_FDE = 0_B, и необходимые значения перезагрузки для данной скорости определяется по формулам из таблицы 20.6.

Таблица 20.6 – Формулы для определения скорости пересылки данных при использовании фиксированного тактового сигнала

FDE	BRS	BG	Формула	
0	0	0, …, 8191	$baudrate = \frac{f_{CLK}}{32 \times (BG + 1)}$,	(20.1)
			$BG = \frac{f_{CLK}}{32 \times baudrate} - 1$	(20.2)
	1		$baudrate = \frac{f_{CLK}}{48 \times (BG + 1}$,	(20.3)
			$BG = \frac{f_{CLK}}{48 \times baudrate} - 1$	(20.4)

Максимальная скорость пересылки данных, которая может быть достигнута для асинхронных режимов, используя два фиксированных делителя и входной тактовый сигнал модуля частотой 60 МГц, должна быть 1,875 Мбод. В таблице 20.7 представлены скорости пересылки данных вместе с необходимыми значениями перезагрузки и ошибками отклонений от рассчитанных скоростей.

Таблица 20.7 – Типовые скорости пересылки данных при соответствующих значениях регистров перезагрузки и отклонения скоростей

Скорость пересылки (baudrate)	$BRS = 0_B$, f_{CLK} = 60 МГц		$BRS = 1_B$, f_{CLK} = 60 МГц	
	Ошибка отклонения, %	Перезагружаемое значение	Ошибка отклонения, %	Перезагружаемое значение
1,875 Мбод	–	0000_H	–	–
1,250 Мбод	–	–	–	000_H
10,200 Кбод	0,7 / –0,4	0060_H / 0061_H	0,2 / –1,4	0040_H / 0041_H
9600 бод	0,2 / –0,4	$00C2_H$ / $00C3_H$	0,2 / –0,6	0081_H / 0082_H
4800 бод	0,2 / –0,1	0185_H / 0186_H	0,2 / –0,2	0104_H / 0105_H
2400 бод	0,0 / –0,1	$020C_H$ / $030D_H$	0,2 / –0,0	0207_H / 0208_H
1200 бод	0,0 / –0,0	0619_H / $061A_H$	0,1 / –0,0	0410_H / 0411_H

Примечание – Бит SxCON_FDE должен быть очищен для того, чтобы достигнуть скоростей пересылки данных, приведенных в таблице 20.7. Ошибки отклонения, данные в таблице 20.7, являются округленными. При использовании скоростей передачи микроконтроллера будет обеспечена корректная передача данных без ошибок отклонения.

Использование дробного делителя

Для включения дробного делителя необходимо установить бит SxCON_FDE. Тактовый сигнал f_{DIV} формируется из f_{CLK}. При этом f_{CLK} делится на дробь n/512 для любого значения n от 0 до 511. Если n = 0, то отношение делителя 1, т. е. $f_{DIV} = f_{CLK}$. Таким образом, дробный делитель позволяет программировать скорость передачи данных с намного большей точностью, чем двумя основными делителями частоты.

В таблице 20.8 представлены формулы для расчета скорости пересылки данных в асинхронном режиме при использовании дробного делителя. В таблице 20.9 содержатся данные о типовых скоростях пересылки и возможном отклонении от заданных скоростей.

Таблица 20.8 – Формулы расчета скорости пересылки данных при использовании дробного делителя

FDE	BRS	BG	FDV	Формула
1	–	1, …, 8191	1, …, 511	$\text{baudrate} = \frac{\text{FDV}}{512} \times \frac{f_{CLK}}{16 \times (BG + 1)}$ (20.5)
			0	$\text{baudrate} = \frac{f_{CLK}}{16 \times (\text{BG} + 1)}$ (20.6)

Таблица 20.9 – Типовые скорости пересылки данных при использовании дробного делителя и возможные отклонения

f_{CLK}, МГц	Требуемая скорость, Кбод	BG	FDV	Итоговая скорость, Кбод	Отклонение, %
40	115,2	15_H	$1F7_H$	115,214	0,01
	56,6	26_H	$1F7_H$	56,607	0,01
	37,4	41_H	$1F7_H$	37,404	0,01
	19,2	83_H	$1F7_H$	19,202	0,01
60	115,2	20_H	$1F7_H$	115,214	0,01
	56,6	41_H	$1F7_H$	56,607	0,01
	37,4	61_H	$1FD_H$	37,415	0,04
	19,2	$C5_H$	$1FD_H$	19,207	0,04

Скорость пересылки данных в синхронном режиме

В синхронном режиме скорость пересылки данных получается из сигнала генератора скорости пересылки делением на четыре.

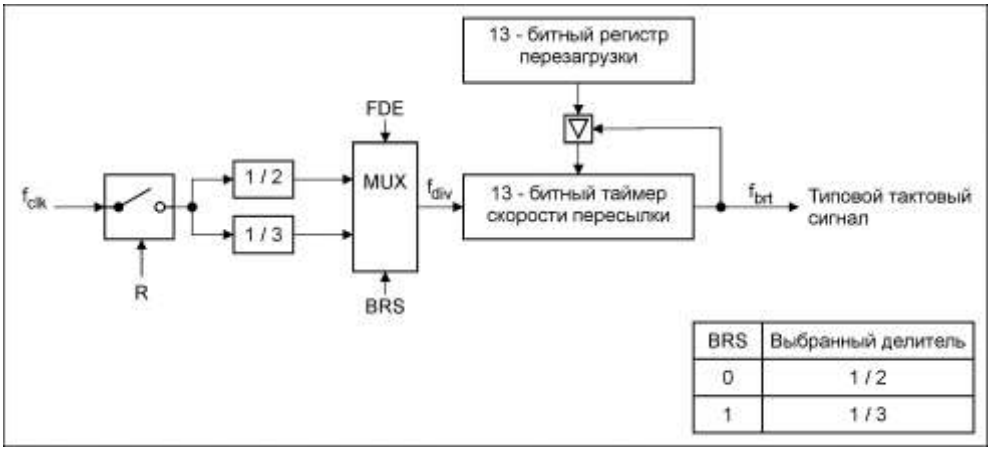


Рисунок 20.20 – Схема формирования скорости передачи в синхронном режиме

Скорость пересылки данных для синхронного режима может быть определена по формулам в таблице 20.10.

Таблица 20.10 – Формулы для определения скорости передачи данных (baudrate) в синхронном режиме

BRS	BG	Формула	
0	1, …, 8191	$baudrate = \frac{f_{CLK}}{8 \times (BG + 1)}$,	20.7)
		$baudrate = \frac{f_{CLK}}{8 \times (BG + 1)} - 1$	(20.8)
1		$baudrate = \frac{f_{CLK}}{12 \times (BG + 1)}$,	(20.9)
		$baudrate = \frac{f_{CLK}}{12 \times (BG + 1)} - 1$	(20.10)

Максимальная скорость пересылки данных, которая может быть достигнута в синхронном режиме, используя входной тактовый сигнал 60 МГц, должна быть 7,5 Мбод.

Аппаратные возможности обнаружения ошибок

Для повышения надежности последовательного обмена данными ASC может выставлять запрос на прерывание по ошибке, указывающий на наличие ошибок. При этом три флага состояния ошибок регистра SxCON указывают на тип произошедшей ошибки. После завершения приема линия запроса прерывания по ошибке EIR будет активирована одновременно с запросом на прерывание по приему, если выполнено одно или несколько следующих условий:

- при установленном бите обнаружения ошибок SxCON_FEN не обнаружена «1» в любом из ожидаемых стоповых битов. После этого будет установлен флаг SxCON_FE, указывающий на ошибку формата передаваемых данных (только для асинхронного режима);

- при включенном режиме проверки четности SxCON_PEN = 1_B в случае обнаружения неправильного результата. Также будет установлен флаг SxCON_PE, индицируя ошибку четности (только для асинхронного режима);

- при разрешенном обнаружении ошибки переполнения SxCON_OEN = 1_B, если полученные данные не считались из приемного буфера до окончания следующего цикла получения данных. При этом устанавливается флаг SxCON_OE, указывающий на ошибку переполнения (асинхронный и синхронный режимы).

Прерывания

Интерфейс ASC обеспечивает четыре источника прерываний. Линия TIC указывает на прерывание по передаче, TBIC индицирует прерывание по освобождению передающего буфера (готовность к загрузке следующих данных в регистр SxTBUF), RIC указывает на прерывание по приему, EIR указывает на наличие ошибок. Выходные линии прерываний TBIR, TIR, RIR и EIR активны в течение двух тактов f_{CLK}.

Причина запроса прерывания по ошибке (ошибка синхронизации кадров, ошибка четности и ошибка переполнения) может быть определена по статусным флагам FE, PE или OE, расположенным в регистре SxCON.

Примечание – В отличие от запроса прерывания по ошибке EIR, флаги состояния ошибки FE, PE и OE не сбрасываются автоматически. Они должны быть очищены программно.

При нормальной работе (отсутствуют прерывания по ошибке) ASC обеспечивает три типа запроса на прерывание, предназначенных для управления обменом данными:

- TBIR устанавливается, когда данные перемещаются из буферного регистра в сдвиговый регистр;

- TIR устанавливается перед передачей последнего бита в асинхронном режиме или после передачи восьмого бита в синхронном режиме;

- RIR устанавливается, когда полученные данные перемещаются в приемный буферный регистр SxRBUF.

Если нет необходимости получения данных, передатчик обслуживается при помощи двух прерываний. В случае одиночных передач достаточно использовать только прерывание TIR, показывающее окончание передачи данных.

При многократных последовательных передачах необходимо обеспечить загрузку следующих данных до окончания передачи текущих, т. е. когда последний бит кадра был передан. Такой режим работы можно организовать, используя прерывание TBIR, вырабатываемое во время очистки буферного регистра. В этом случае новые данные будут загружены в буферный регистр передатчика еще до передачи последнего бита текущих данных, т. е. передача будет осуществляться без паузы.

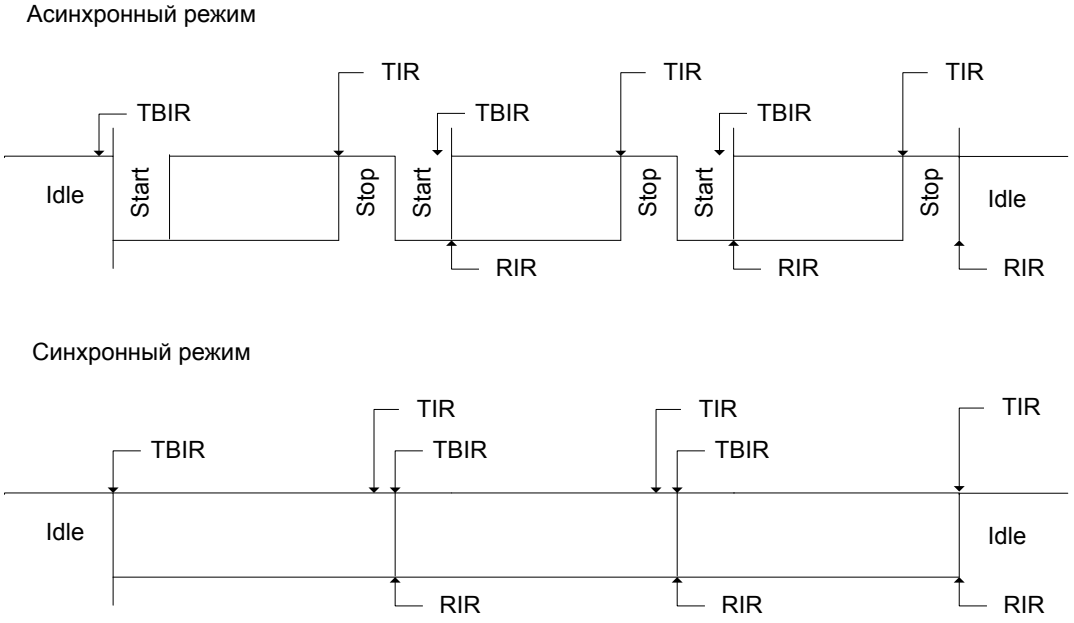


Рисунок 20.21 – Генерация прерываний ASC

Как видно из рисунка 20.21, флаг TBIR может использоваться в качестве запускающего механизма для подпрограммы перезагрузки буферного регистра. Поэтому программное обеспечение должно обязательно использовать флаг TIR, выставляющийся в конце передачи блока данных, чтобы гарантировать, что все данные были полностью переданы.

21 Высокоскоростные синхронные последовательные интерфейсы SSC0 и SSC1

Высокоскоростные последовательные интерфейсы SSC0 и SSC1 поддерживают полнодуплексный и полудуплексный последовательный синхронный обмен данными на скоростях до 30 Мбод при частоте входного тактового сигнала 60 МГц. Последовательный тактовый сигнал может быть сформирован непосредственно модулем SSC (в режиме ведущего – «master») или может быть принят от внешнего источника (в режиме ведомого – «slave»). Размер данных, направление передачи, полярность тактового сигнала и фаза программируются. Это позволяет обеспечивать передачу данных с SPI-совместимыми устройствами. Прием и передача – с двойной буферизацией. 16-разрядный контроллер скорости передачи данных позволяет формировать независимый тактовый сигнал.

В состав контроллера входят два одинаковых блока последовательной передачи данных – SSC0 и SSC1, отличающиеся лишь адресами регистров и выходными портами. Поэтому будет дано описание работы лишь для одного интерфейса.

Особенности интерфейса SSC:

- master или slave оперативные режимы;
- полнодуплексный или полудуплексный обмен данными;
- гибкий формат данных:
  - программирование числа бит данных: от 2 до 16 бит;
  - программирование сдвига: LSB или MSB сдвигаются первыми;
- программирование полярности тактового сигнала: низкий или высокий уровень на линии при отсутствии тактового сигнала;
- программирование фазы: данные сдвигаются или захватываются по положительному или отрицательному фронту тактового сигнала.
  - генератор скорости передачи данных до 10 Мбод в master-режиме и до 5 Мбод в slawe-режиме (частота f_{PER} = 20 МГц);
- генерирование прерываний:
  - после передачи данных;
  - после приема данных;
  - при обнаружении ошибок (приема, фазы, скорости, передачи).

На рисунке 21.1 представлена функциональная схема интерфейса SSC.

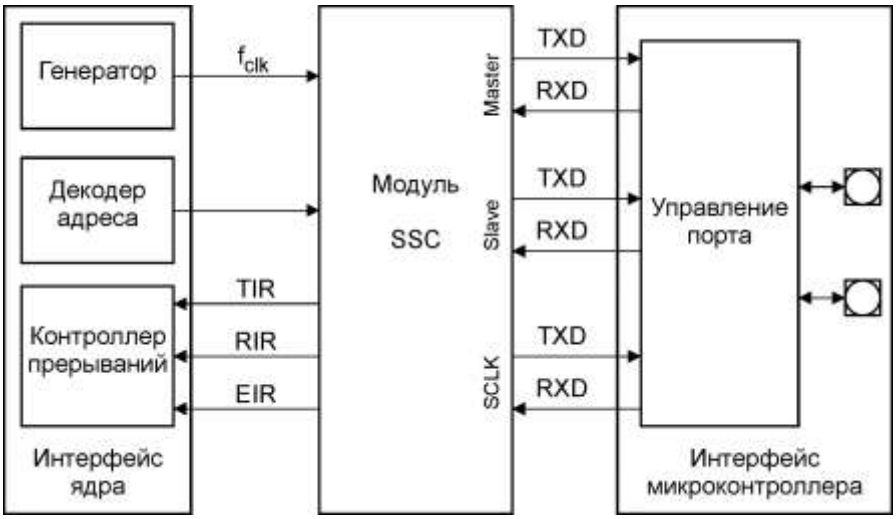


Рисунок 21.1 – Функциональная схема интерфейса SSC

Рисунок 21.2 – Регистры высокоскоростных последовательных интерфейсов SSC0 и SSC1

Все регистры расположены в адресном пространстве SFR/ESFR. Соответствующие адреса регистров могут быть найдены в списке регистров.

21.1 Операции

Блок-схема высокоскоростного синхронного последовательного интерфейса SSC представлена на рисунке 21.3.

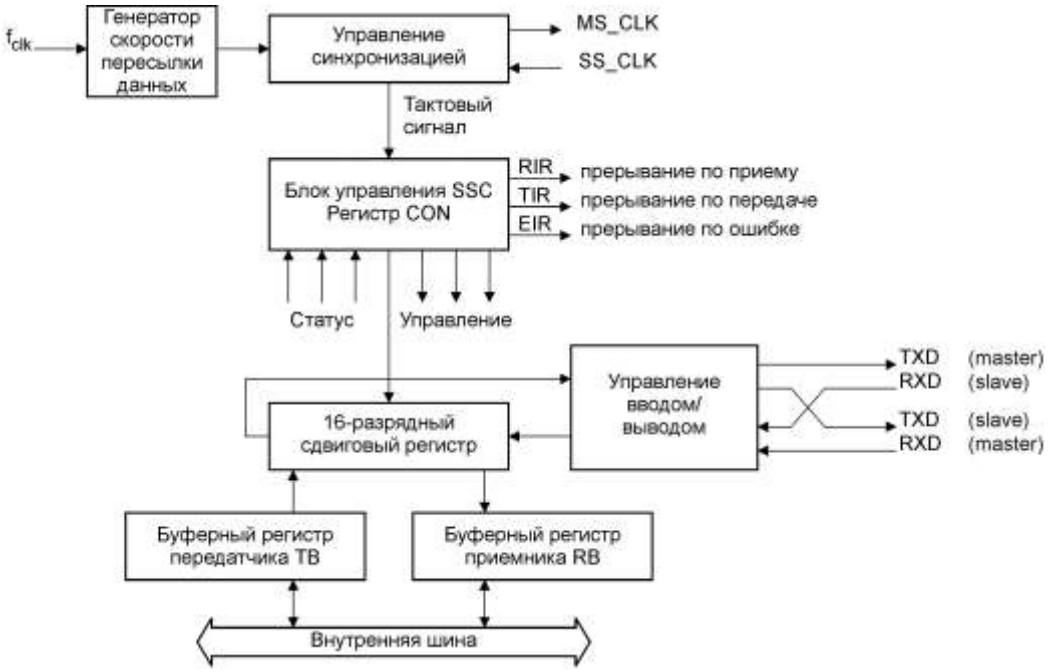


Рисунок 21.3 – Блок-схема высокоскоростного синхронного последовательного интерфейса SSC

Высокоскоростной синхронный последовательный интерфейс может быть сконфигурирован очень гибким способом, поэтому он может сопрягаться с другими синхронными последовательными интерфейсами, может использоваться для соединений master/slave или работать совместно с широко используемым интерфейсом SPI. Данные передаются или принимаются по линиям TXD или RXD, обычно подключаемым к выводам MTSR (master передает/slave принимает) и MRST (master принимает/slave

передает). Сигнал тактового генератора выводится на линию MS_CLK или вводится через линию SS_CLK. Обе эти линии подключаются к SCLK и являются дополнительными функциями выводов порта.

Выбор режима

Операционным режимом последовательного интерфейса управляет регистр управления SSCxCON, см. рисунки 21.4 – 21.7, таблицы 21.1, 21.2 (здесь и далее по тексту под «x» подразумевается номер регистра 0 или 1). Этот регистр имеет две функции:

- во время программирования SSC отключен битом SSCxCON_EN = 0_B, обеспечивает доступ к ряду служебных битов;
- во время работы SSC включен битом SSCxCON_EN = 1_B, обеспечивает доступ к флагам состояния.

Регистр конфигурации

Этот регистр содержит биты управления режимами и биты разрешения проверок на ошибки, а также статусные флаги для идентификации ошибок. В зависимости от состояния бита EN, доступны либо функции управления, либо флаги ошибок.

Режим программирования SSCxCON_EN = 0_B

SSC0CON
регистр управления

SFR ($FFB2_H$/ $D9_H$) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	MS	0	AR EN	BEN	PEN	REN	TEN	LB	PO	PH	HB	BM			
З Ч	З Ч	Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч			

Рисунок 21.4 – Формат регистра SSC0CON

SSC1CON
регистр управления

SFR ($FF5E_H$/ AF_H) значение после Сброса:0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	MS	0	AR EN	BEN	PEN	REN	TEN	LB	PO	PH	HB	BM			
З Ч	З Ч	Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч			

Рисунок 21.5 – Формат регистра SSC1CON

Таблица 21.1 – Функциональное назначение полей регистров SSC0CON и SSC1CON

Поле	Биты	Тип	Описание
1	2	3	4
BM	3-0	Чтение Запись	Выбор ширины данных: 0000 Зарезервировано. 0001 – 1111 Ширина передаваемых данных 2, …, 16 бит (BM+1).

Окончание таблицы 21.1

1	2	3	4
HB	4	Чтение Запись	Бит управления типом заголовка: 0 Первый передается/принимается LSB. 1 Первый передается/принимается MSB.
PH	5	Чтение Запись	Бит управления фазой тактового сигнала 0 Передача по переднему фронту, захват по заднему фронту. 1 Передача по заднему фронту, захват по переднему фронту.
PO	6	Чтение Запись	Бит управления полярностью тактового сигнала: 0 Состояние ожидания при низком уровне напряжения. Передний фронт – переход из «0» в «1». 1 Состояние ожидания при высоком уровне напряжения. Передний фронт – переход из «1» в «0».
LB	7	Чтение Запись	Бит управления режимом loopback: 0 Нормальный выход. 1 Вход приемника подключен к выходу передатчика (полудуплексный режим).
TEN	8	Чтение Запись	Бит разрешения определения ошибки передачи: 0 Игнорировать ошибки передачи. 1 Проверять ошибки передачи.
REN	9	Чтение Запись	Бит разрешения определения ошибки приема: 0 Игнорировать ошибки приема. 1 Проверять ошибки приема.
PEN	10	Чтение Запись	Бит разрешения определения ошибки фазы: 0 Игнорировать ошибки фазы. 1 Проверять ошибки фазы.
BEN	11	Чтение Запись	Бит разрешения определения ошибки скорости передачи. 0 Игнорировать ошибки скорости передачи. 1 Проверять ошибки скорости передачи.
AREN	12	Чтение Запись	Бит разрешения автоматического перезапуска SSC: 0 Не требуется дополнительных действий при определении ошибки скорости передачи. 1 Автоматический перезапуск SSC при определении ошибки скорости передачи.
MS	14	Чтение Запись	Бит выбора режима master: 0 Режим slave, работа с внешним тактовым сигналом на входе SCLK. 1 Режим master, генерация собственного тактового сигнала и вывод его через SCLK.
EN	15	Чтение Запись	Бит разрешения работы SSC = 0_B. Передача и прием данных отключены, имеется доступ к битам управления.
0	13	Чтение	Зарезервирован. При чтении возвращает значение «0». Всегда должен записываться как «0».

Операционный режим SSCxCON_EN = 1_B

SSC0CON
регистр управления

SFR ($FFB2_H$/ $D9_H$) значение после Сброса:0000_H

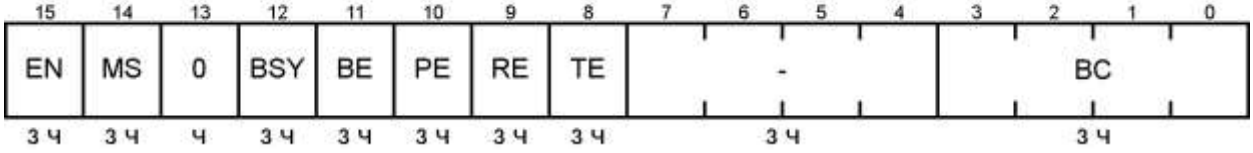


Рисунок 21.6 – Формат регистра SSC0CON

SSC1CON
регистр управления

SFR ($FF5E_H$/ AF_H) значение после Сброса:0000_H

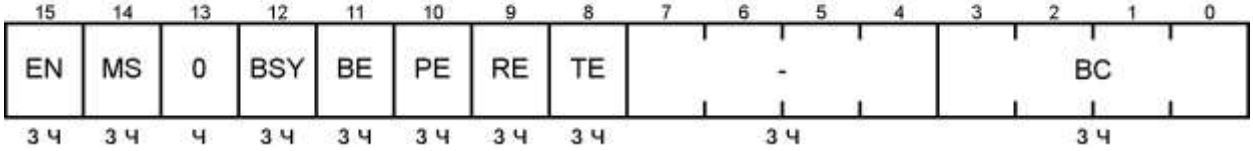


Рисунок 21.7 – Формат регистра SSC1CON

Таблица 21.2 – Функциональное назначение полей регистров SSC0CON и SSC1CON

Поле	Биты	Тип	Описание
1	2	3	4
BC	3-0	Чтение Запись	Поле отсчета битов. Изменяет свое значение на единицу после каждого переданного бита. Примечание – Не производить запись в это поле!
TE	8	Чтение Запись	Флаг ошибки передачи: 0 Нет ошибки. 1 Передача началась, но при этом не было изменено значение в slave буфере передачи.
RE	9	Чтение Запись	Флаг ошибки приема: 0 Нет ошибки. 1 Прием был завершен до того, как приемный буфер был прочитан.
PE	10	Чтение Запись	Флаг ошибки фазы: 0 Нет ошибки. 1 Принимаемые данные изменяются во время считывания значения.
BE	11	Чтение Запись	Флаг ошибки скорости передачи: 0 Нет ошибки. 1 Фактическая и ожидаемая скорость передачи данных отличается больше чем диапазон (0,5 – 2) раз.

Окончание таблицы 21.2

1	2	3	4
BSY	12	Чтение Запись	Флаг занятости. Устанавливается во время передачи. Не записывать!!!
MS	14	Чтение Запись	Бит выбора режима master: 0 Режим slave, работа с внешним тактовым сигналом на входе SCLK. 1 Режим master, генерация собственного тактового сигнала и вывод его через SCLK.
EN	15	Чтение Запись	Бит разрешения работы SSC = 1_B. Передача и прием данных разрешены, доступны флаги состояния и управление M/S.
–	7-4	–	Зарезервированы.
0	13	Чтение	Зарезервирован. При чтении возвращает значение «0». Всегда должен записываться как «0».

При записи в SSCxCON необходимо в зарезервированные биты записывать нулевые значения.

Регистр сдвига подключен с помощью логики управления и к выводу передачи, и к выводу приема. Передача и прием данных могут синхронизироваться и осуществляться одновременно, т. е. число переданных бит такое же, как и полученных. Передаваемые данные записываются в буфер передатчика SSCxTB и сразу же перемещаются в сдвиговый регистр, если он пуст. SSC master SSCxCON_MS = 1_B немедленно начинает передачу, в то время как SSC slave SSCxCON_MS = 1_B ждет поступления тактового сигнала. Когда передача начата, устанавливается флаг занятости интерфейса SSC SSCxCON_BSY = 1_B и флаг запроса на прерывание по передаче TIR, указывающий, что регистр SSCxTB может быть вновь загружен. После передачи запрограммированного числа битов (2, …, 16) содержимое сдвигового регистра перемещается в приемный буферный регистр SSCxRB, и выставляется запрос на прерывание по приему RIR. При отсутствии дальнейших передач данных SSCxTB пуст, бит SSCxCON_BSY сбрасывается. Флаг SSCxCON_BSY устанавливается аппаратно, поэтому нельзя программно менять значение этого бита.

Буферный регистр передатчика

Буферный регистр передатчика SSCxTB содержит данные, которые необходимо передать, см. рисунки 21.8, 21.9, таблицу 21.3.

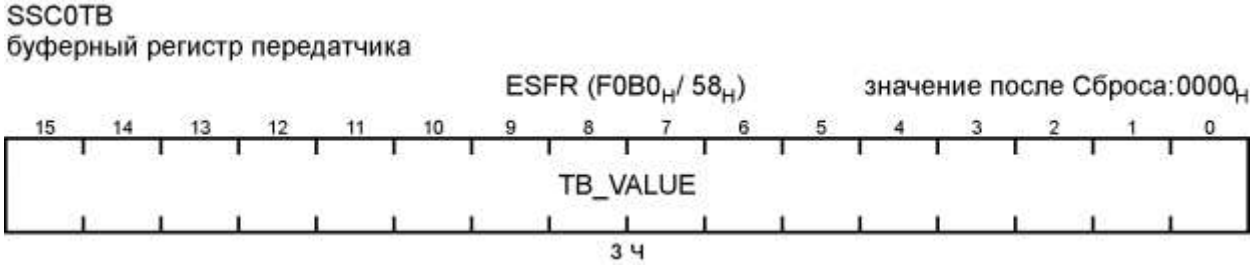


Рисунок 21.8 – Формат регистра SSC0TB

SSC1TB
буферный регистр передатчика

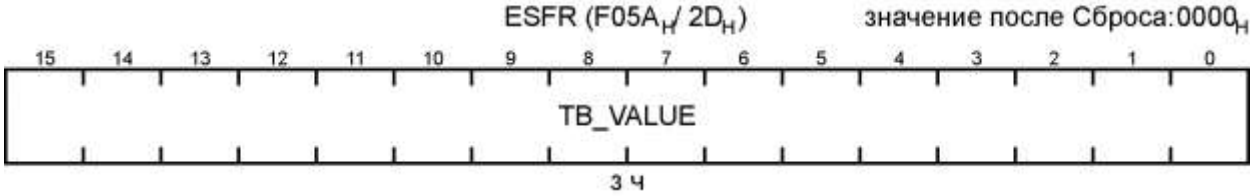


Рисунок 21.9 – Формат регистра SSC1TB

Таблица 21.3 – Функциональное назначение полей регистров SSC0TB и SSC1TB

Поле	Биты	Тип	Описание
TB_VALUE	15-0	Чтение Запись	Содержит значение данных, которое будет передано. Невыбранные биты SSCxTB игнорируются во время передачи.

Буферный регистр приемника

Буферный регистр приемника SSCxRB, см. рисунки 21.10, 21.11, таблицу 21.4, содержит принятые данные.

SSC0RB
буферный регистр приемника

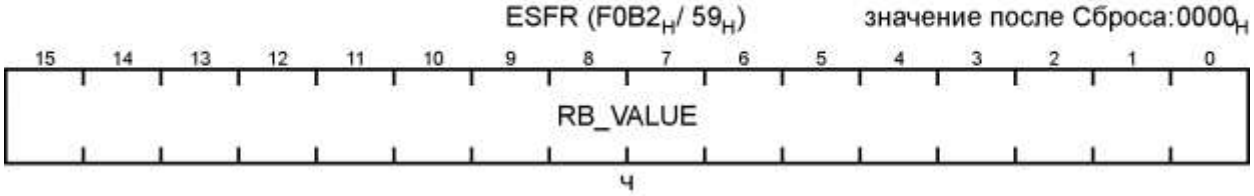


Рисунок 21.10 – Формат регистра SSC0RB

SSC1RB
буферный регистр приемника

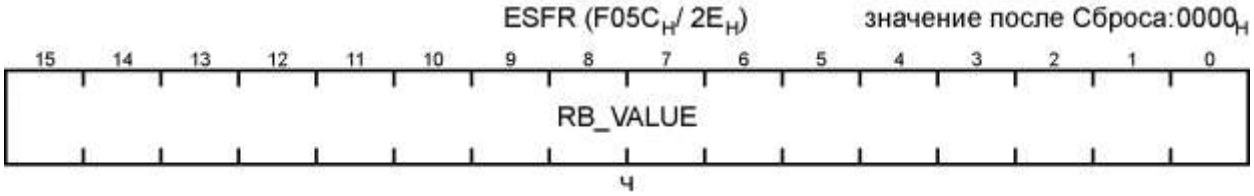


Рисунок 21.11 – Формат регистра SSC1RB

Таблица 21.4 – Функциональное назначение полей регистров SSC0RB и SSC1RB

Поле	Биты	Тип	Описание
RD_VALUE	15-0	Чтение	Содержит значение принятых данных. Невыбранные биты SSCxRB являются не действительными и должны быть проигнорированы.

Примечание – При использовании нескольких модулей последовательного интерфейса только один модуль SSC может быть включен в режиме master.

Настройку последовательной передачи данных можно производить в широком диапазоне:

- данные могут содержать от 2 до 16 битов;
- передача может начинаться с LSB или MSB;
- на линии тактового сигнала в режиме ожидания может быть как низкий, так и высокий уровень напряжения;

- данные могут передаваться по переднему или по заднему фронту тактового сигнала;

- скорость передачи данных может быть установлена от 457,76 бод до 30 Мбод (при частоте входного тактового сигнала f_{CLK} = 60 МГц);

- тактовый сигнал может как генерироваться модулем SSC (в режиме master), так и приниматься от внешнего источника (в режиме slave).

Все эти особенности позволяют адаптировать SSC к широкому диапазону приложений, в которых требуется последовательная передача данных.

Возможность выбора ширины данных поддерживает передачу кадров любой длины, от 2-битовых посылок до 16-разрядных. Передача данных, начиная с LSB SSCxCON_HB = 0_B, позволяет производить обмен данными с устройствами, поддерживающими ASC в синхронном режиме (C166, 8051). Начало передачи с MSB позволяет осуществлять обмен по SPI интерфейсу.

Вне зависимости от того выбран MSB или LSB, передаваемые данные всегда выстраиваются в регистрах SSCxTB и SSCxRB в правильном порядке, при этом LSB передаваемых данных размещается в бите 0. С помощью логики сдвигового регистра биты данных перераспределяются для передачи. Значения в невыбранных битах SSCxTB игнорируются, а значения в невыбранных битах SSCxRB не будут действительными и должны игнорироваться программой получателя.

Управление тактовыми импульсами позволяет адаптировать режим приема и передачи SSC к различным последовательным интерфейсам. Один из фронтов тактового сигнала (передний или задний) используется для передачи данных, в то время как другой используется для захвата данных. Изменяя значение бита SSCxCON_PH, можно установить тип фронта (передний или задний), используемый для каждой функции. Изменяя значение бита SSCxCON_PO, можно установить уровень напряжения на тактовом выходе в режиме ожидания, см. рисунок 21.12.

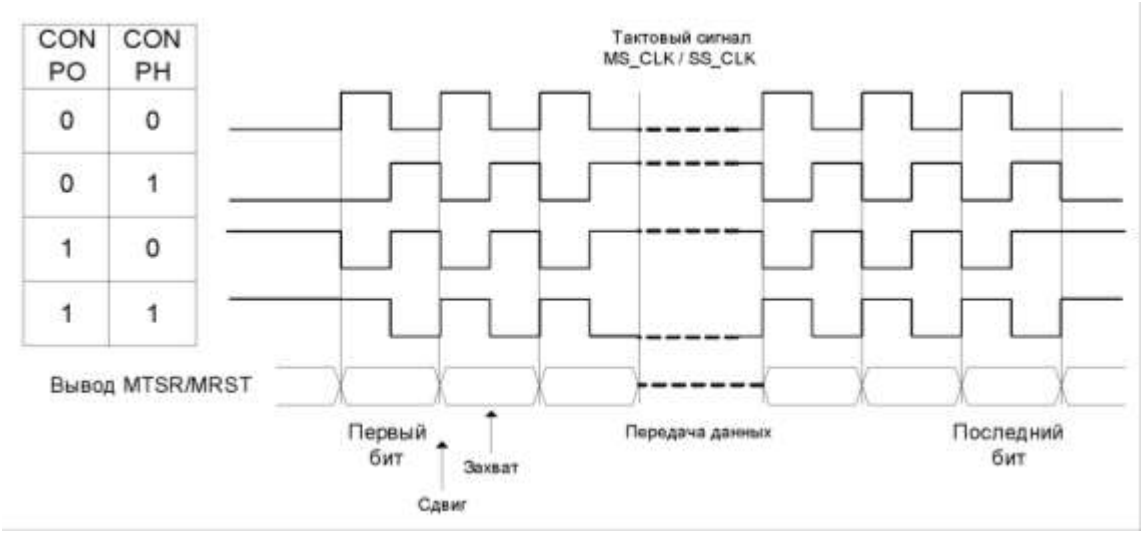


Рисунок 21.12 – Полярность и фаза тактового сигнала

Полнодуплексные операции

Различные устройства подключены к трем линиям. Режимы работы линий определяет всегда устройство-master. Линия, подключенная к выводу MTSR, является линией передачи данных, линия приема данных подключается к входу MRST, линия тактового сигнала подключается к SCLK. Тактовый сигнал может выдавать только то устройство, которое работает в режиме master. Все остальные устройства работают в режиме slave, используют внешний тактовый сигнал, поэтому их выводы SCLK должны быть сконфигурированы для ввода тактового сигнала. Вывод сдвигового регистра

передачи master подключен к линии, соединенной с входами приемных сдвиговых регистров устройств-slave. Выходы передающих сдвиговых регистров устройств-slave подключены к линии приемного сдвигового регистра master. Внешние соединения являются жестко определенными, и поэтому для каждого конкретного устройства функция и направление этих выводов определяются с помощью выбора режима (master или slave).

Примечание – Направление передачи, показанное на рисунке 21.13, используется как для MSB старта, так и для LSB старта.

Во время настройки одно устройство должно быть установлено в режим master, а все остальные – в режим slave. Инициализация включает установку операционного режима SSC устройства, а также включение функций соответствующих выводов портов.

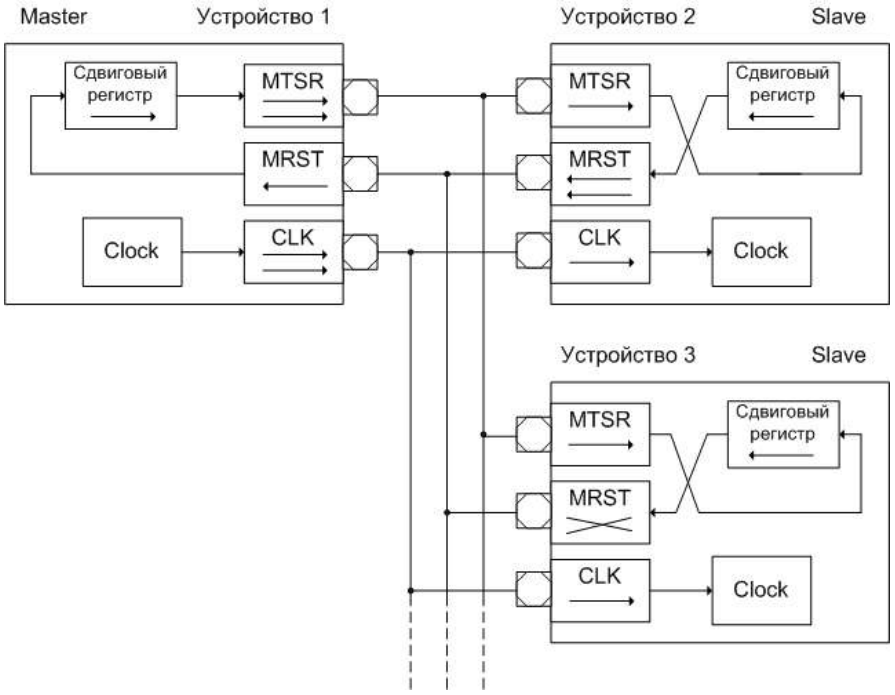


Рисунок 21.13 – Интерфейс SSC в полнодуплексном режиме работы

Все выходные линии MRST из всех slave-устройств должны быть объединены и подключены к линии приема данных. Во время передачи данных каждый slave может передавать данные из своего сдвигового регистра. Существуют два способа для преодоления конфликтов на линии между slave-устройствами:

- только одно slave-устройство включает драйвер передачи для MRST. Для всех других устройств MRST программируется на вход, таким образом, только один slave сможет выставлять данные на линию. Master выбирает slave-устройство, от которого ожидает поступление данных, либо с помощью независимых линий выбора, либо с помощью специальных команд, посланных на slave-устройство. После этого выбранное slave-устройство переключает линию MRST на выход и находится в таком состоянии до тех пор, пока не получит сигнала или команды отключения;

- slave-устройства используют входы с открытыми коллекторами на MRST. В этом случае соединение происходит по формуле wired-AND. При этом линия получения нуждается во внешнем pullup-устройстве. При этом если все не выбранные для передачи slave-устройства выставляют «1» на выходе, исключается искажение посылаемых данных. Master выбирает «активный» slave либо с помощью специальной команды, либо с помощью независимых линий выбора.

После выполнения необходимой инициализации последовательный интерфейс может быть включен. Для master-устройства линия тактового генератора перейдет в

запрограммированное состояние: «0» или «1». Линия данных перейдет в «0» или «1», пока первая передача не начнется. После передачи линия данных будет оставаться на логическом уровне последнего переданного информационного разряда.

Когда последовательный интерфейс включен, master может инициализировать первую передачу данных при записи данных в регистр SSCxTB. Это значение скопируется в регистр сдвига (если он пустой), и выбранный первый бит данных будет перемещен на линию TXD по следующему импульсу тактового сигнала контроллера скорости передачи (передача начинается, только если SSCxCON_EN = 1_B). В зависимости от выбранной фазы импульс тактового сигнала также будет сгенерирован на линии MS_CLK. В то же самое время, по противоположному фронту синхроимпульса, master-устройство захватит данные, обнаруженные на входной линии RXD. Таким образом, происходит обмен передаваемых данных с принимаемыми данными. Поскольку тактовый сигнал получается всеми slave-устройствами, содержимое их сдвиговых регистров будет сдвинуто синхронно со сдвиговым регистром master-устройства – сдвиг выходных данных, содержащихся в регистрах, и сдвиг данных, обнаруженных на входной линии. После прохождения предварительно запрограммированного числа тактовых импульсов (через выбор ширины данных), данные, переданные master-устройством, содержатся во всех сдвиговых регистрах slave-устройств, а регистр передачи master-устройства содержит данные только от выбранного slave-устройства. После этого данные из сдвигового регистра копируются в приемный буферный регистр RB и активируется линия прерывания RIR. Как только данные из буферного регистра будут переданы в регистр передачи slave-устройства, первый бит (MSB или LSB) сразу поступает на выход RXD, не дожидаясь первого тактового импульса от master-устройства. Это необходимо по причине того, что передний фронт тактового сигнала может использоваться для приема данных, так как возможна различная выбранная фаза. Разряд SSCxCON.BSY не будет установлен до тех пор, пока не появится передний фронт синхроимпульса.

Примечание – Прием и передача данных в SSC всегда происходят одновременно, независимо от правильности передаваемых или принимаемых данных. В этом заключается одно из отличий SSC от интерфейса ASC.

Операции в полудуплексном режиме

В полудуплексном режиме и для приема и для передачи данных используется только одна линия. Линия обмена данными соединяет и MTSR и MRST каждого из устройств. Линия тактовых сигналов подключена к SCLK-выводам.

Master-устройство управляет передачей данных при помощи собственного тактового сигнала. Slave-устройство использует внешний тактовый сигнал. Данные могут передаваться между произвольными устройствами, т. к. все выводы для приема и передачи объединены и подключены к одной линии.

Подобно дуплексному режиму, есть два способа избежать конфликтов во время передачи данных в полудуплексном режиме:

- разрешение на включение передачи дается только одному передающему устройству;
- использование режима с открытым коллектором для всех непередающих устройств.

Так как входы и выходы данных соединены вместе, то передаваемые устройством данные подаются на его собственный вход MRST в режиме master, MTSR в режиме slave. Поэтому имеется возможность для определения искажения данных на общей линии передачи.

Рисунок 21.14 иллюстрирует работу интерфейса SSC в полудуплексном режиме работы.

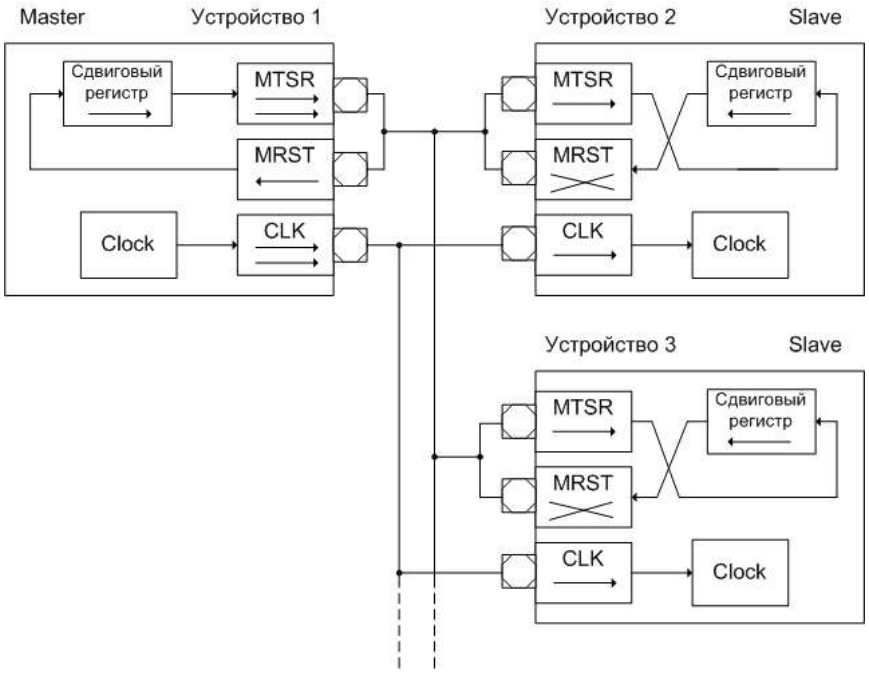


Рисунок 21.14 – Интерфейс SSC в полудуплексном режиме работы

Непрерывные передачи

Когда флаг запроса прерывания по передаче установлен, это указывает на то, что буфер передачи TB пустой и готов к загрузке следующих передаваемых данных. В том случае, когда загрузка данных была произведена до окончания текущей передачи, сразу после завершения новые данные передаются в сдвиговый передающий регистр и следующая передача начнется без какой-либо дополнительной задержки. В этом случае на линии отсутствует пауза между двумя пакетами данных. Например, передача двух байтов выглядит как передача слова. Этот режим может быть использован для передачи в случае, когда устройству требуется более 16 битов в одном пакете. Также он используется для передачи данных между 8-разрядными и 16-разрядными устройствами по последовательной шине.

Примечание – Эта особенность используется только в том случае, когда длина данных кратна выбранной базовой длине передачи.

Генерация скорости передачи данных

Последовательный интерфейс SSCx имеет свой собственный специализированный 16-разрядный контроллер скорости передачи с возможностью перезагрузки содержимым 16-разрядного регистра, позволяющий формировать синхросигнал с необходимой частотой независимо от таймеров. На рисунке 21.15 представлен контроллер скорости передачи данных.

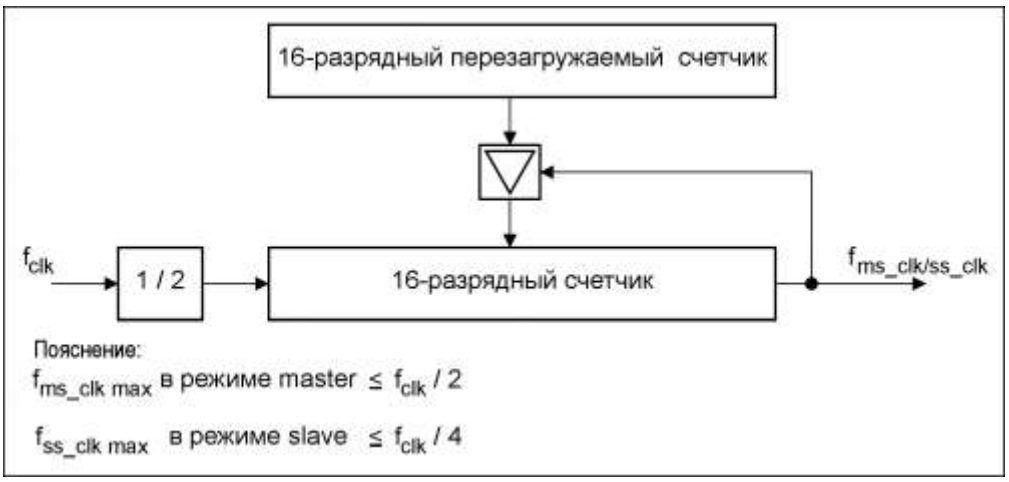


Рисунок 21.15 – Генератор скорости передачи данных

Генератор скорости передачи синхронизируется от внешнего тактового сигнала f_{CLK}. Отсчет таймера ведется в обратном направлении. Регистр SSCxBR имеет двойную функцию – регистр скорости пересылки/перезагрузки регистр. Чтение SSCxBR, в то время как интерфейс SSCx выключен, возвращает запрограммированное значение перезагрузки. В этом режиме значение перезагрузки может быть записано в регистр.

Перезагружаемый регистр генератора скорости пересылки

Регистр BR, см. рисунки 21.16, 21.17, таблицу 21.5, содержит 16-разрядное перезагружаемое значение для таймера скорости передачи.

SSC0BR
регистр скорости пересылки

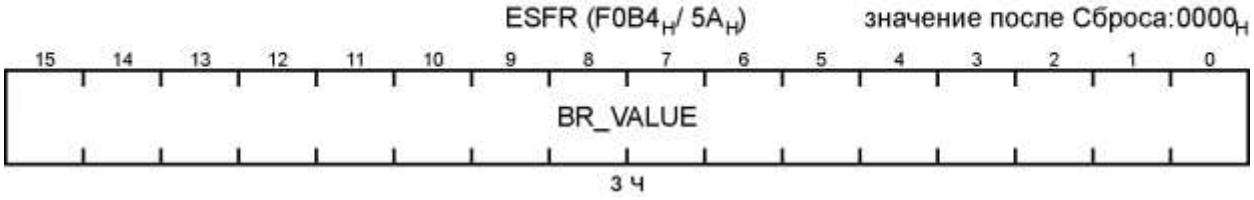


Рисунок 21.16 – Формат регистра SSC0BR

SSC1BR
регистр скорости пересылки

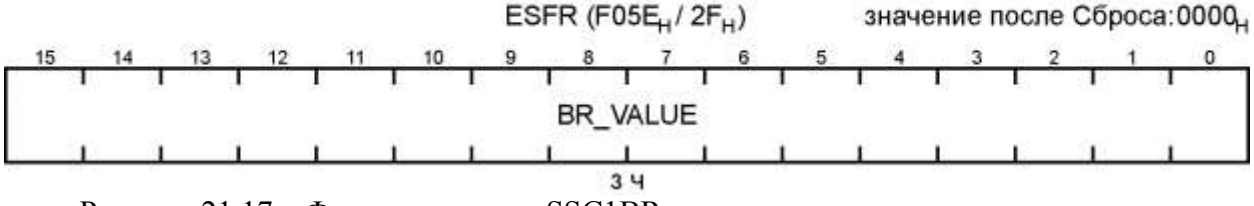


Рисунок 21.17 – Формат регистра SSC1BR

Таблица 21.5 – Функциональное назначение полей регистров SSC0BR и SSC1BR

Поле	Биты	Тип	Описание
BR_VALUE	15-0	Чтение	Чтение возвращает записанное ранее значение перезагрузки. Запись загружает в таймер скорости передачи перезагружаемое значение регистра BR_VALUE.

Примечание – При рабочем интерфейсе SSC запись в регистр SSCxBR нежелательна.

Формулы (21.1), (21.2) позволяют вычислить скорость пересылки данных для данного значения перезагрузки или необходимое значение перезагрузки для данной скорости пересылки.

$$\text{baudrate} = \frac{f_{CLK}}{2 \times (
 + 1)}, \quad (21.1)$$

$$BR = \frac{f_{CLK}}{2 \times \text{baudrate}} - 1 \quad (21.2)$$

«BR» представляет собой содержимое перезагружаемого регистра, взятое как 16-разрядное целое число без знака, а скорость пересылки, представленная на рисунке 21.15, равна f_{MC_CLK/SS_CLK}.

Максимальная скорость пересылки данных, которая может быть, используя тактовый генератор 60 МГц, равна 30 Мбод в режиме master (при $
 = 0000_H$) или 15 Мбод в режиме slave (при $
 = 0000_H$).

В таблице 21.6 приведен список некоторых возможных скоростей пересылки данных вместе с необходимыми значениями перезагрузки и отклонениями от этих скоростей при значении тактового генератора 60 МГц.

Таблица 21.6 – Типовые скорости пересылки данных SSCx (f_{MC_CLK/SS_CLK} = 60 МГц)

Значение перезагрузки	Скорость пересылки f_{MC_CLK/SS_CLK}	Отклонение, %
$003B_H$	500 Кбод	0
0095_H	200 Кбод	0
$012B_H$	100 Кбод	0
$FFFF_H$	457,76 бод	0

Механизм определения ошибок

Интерфейс SSC может обнаружить четыре различных типа ошибок. Ошибка приема и ошибка фазы определяются во всех режимах работы. Ошибка передачи и ошибка скорости передачи определяются только в режиме slave. При детектировании ошибки, устанавливается соответствующий ей флаг и активизируется линия EIR. Если до этого флаг был установлен, также генерируется запрос на прерывание по ошибке (EIR). После этого программа обслуживания прерываний по ошибке проверяет флаги ошибок для определения типа имевшей место ошибки. Флаги автоматически не сбрасываются и поэтому должны быть очищены программно перед выходом из подпрограммы обслуживания прерываний.

Структура формирования прерываний по ошибкам показана на рисунке 21.18.

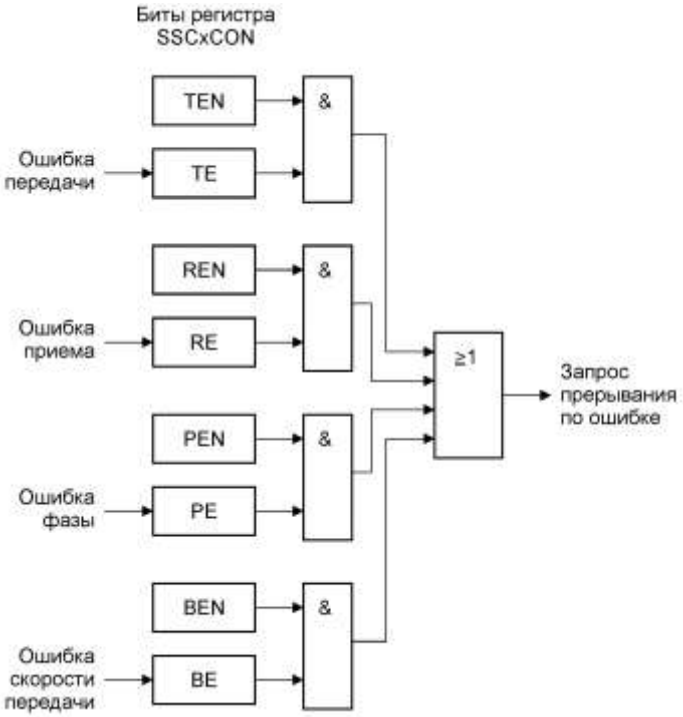


Рисунок 21.18 – Интерфейс SSCx. Управление прерыванием по ошибке

Примечание – Подпрограмма обработки прерываний должна очищать сопоставленный флаг ошибки для предотвращения повторного запроса прерывания.

Ошибка приема

Определяется в режиме master или slave, если предыдущие данные не были считаны из буфера приема SSCxRB до завершения приема новых данных. В этом случае устанавливается флаг SSCxCON.RE. Запрос прерывания по ошибке вырабатывается в том случае, если установлен бит SSCxCON.REN. После этого в приемный буфер SSCxRB будет записано новое значение данных и старое значение будет потеряно.

Ошибка фазы

Определяется в режиме master или slave в том случае, когда сигнал на входе MRST (в режиме master) или входе MTSR (в режиме slave) изменяет свое значение в интервале, начинающемся за один такт ЦПУ до прихода фронта тактового сигнала передачи и заканчивающемся через два такта после прихода фронта. При этом устанавливается флаг SSCxCON.PE.

Ошибка скорости передачи

Определяется в slave режиме в том случае, когда входной тактовый сигнал отклоняется от запрограммированного более чем на 100 %, т. е. либо в два раза больше ожидаемой скорости, либо менее половины ожидаемой скорости. В этом случае устанавливается флаг ошибки SSCxCON.BE и, если установлен бит SSCxCON.BEN, выставляется запрос на прерывание по ошибке EIR. Использование этой функции позволяет определять дополнительные ложные или пропущенные такты на линии передачи тактового сигнала.

Примечание – В том случае, когда эта ошибка имеет место, при установленной «1» в бите SSCxCON.REN автоматически производится сброс блока SSCx. Это необходимо для повторной инициализации SSCx в том случае, когда определяется слишком много или слишком мало тактовых импульсов.

Ошибка передачи

Определяется в slave режиме в том случае, когда master инициализировал передачу данных (master послал тактовый сигнал), а значение в буфере передачи SSCxTB slave-устройства не было изменено с последней передачи. При этом устанавливается флаг

SSCxCON.TE и, если установлен бит SSCxCON.TEN, запрос на прерывание по ошибке EIR. Если передача начинается без записи нового значения в буфер передачи, slave передаст старое содержимое сдвигового регистра, которое обычно заполнено данными, принятыми во время последней передачи. Это может привести к искажению данных на линии передачи в полудуплексном режиме, если slave не выбран для передачи. Этот режим требует, чтобы все slave-устройства, не выбранные для передачи, выставляли на линию только единицы для корректной передачи данных выбранным slave-устройством, т. е. в буфер передачи должно быть записано значение $FFFF_H$.

Причина запроса прерывания по ошибке может быть идентифицирована флагами состояния ошибок в регистре управления SSCxCON.

Примечание – В отличие от запроса прерывания по ошибке EIR, статусные флаги SSCxCON.TE, SSCxCON.RE, SSCxCON.BE не сбрасываются автоматически, поэтому они должны быть очищены программно перед выходом из подпрограммы обслуживания прерываний.

22 Модуль интерфейса I2C

Модуль интерфейса I2C (далее – модуль I2C) обеспечивает полную поддержку двухпроводного последовательного синхронного интерфейса I2C/SMBUS. Результат такой совместимости – легкое соединение со многими запоминающими устройствами и устройствами ввода-вывода, включая EEPROM, SRAM, счетчики, АЦП, ЦАП, периферийные устройства.

Функциональные возможности модуля:

Совместимость с SMBus (Version 1.1 и 2.0), ACCESS.Bus, I2C (Version 2.1).

Поддержка стандартного Standard, скоростного F/S и высокоскоростного Hs режимов.

Программирование действий Master или Slave.

Возможность подключения к шине нескольких ведущих устройств (режим Multi-Master).

Один определяемый программно 7/10-битный адрес для Slave.

Возможность одновременного обращения ко всем устройствам шины, так называемый «общий вызов» (global call).

Особые возможности SMBus:

- отслеживание времени ожидания линии SCL;
- наличие функции PEC (Packet Error Checking – отслеживание ошибок в пакетах данных) с использованием стандартного полинома SMBus;
- возможность обращения одного или нескольких Slave к Master с получением отклика от последнего.

Возможность последовательного опроса.

Функционирование под управлением прерываний.

На рисунке 22.1 показаны регистры, связанные с модулем I2C.

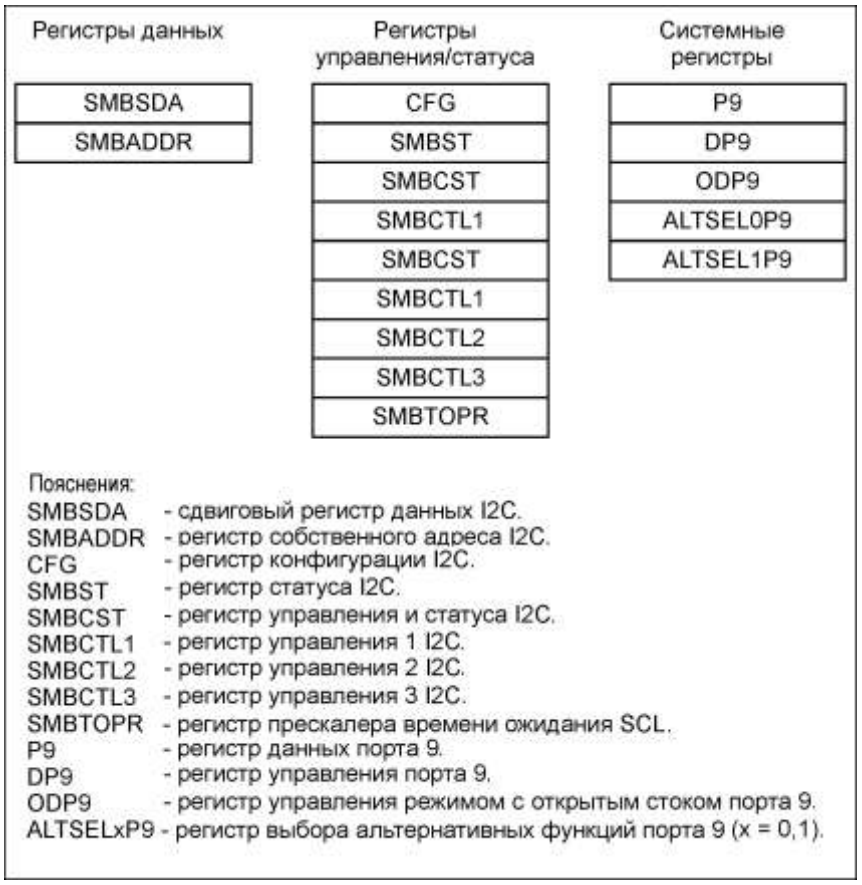


Рисунок 22.1 – Регистры, связанные с модулем I2C

22.1 Протокол шины

Протокол I2C использует двухпроводной интерфейс для двусторонней связи между устройствами, подключенными к шине. Двунаправленная шина состоит из двух линий: линии данных (Serial Data Line – SDA) и линии тактового сигнала (Serial Clock Line – SCL). Эти линии подключены к источнику питания через подтягивающие резисторы. Шинные формирователи любых устройств, подключаемых к шине, выполняются по схеме с открытым коллектором или открытым стоком. Устройства могут выставить только низкий уровень на соответствующей линии. Следовательно, обе линии SDA и SCL реализуют функцию «монтажное И».

Протокол поддерживает режим Multi-Master – одно или несколько устройств, подключенных к шине, могут контролировать шину. Каждому устройству, подключенному к шине, присваивается собственный адрес, оно может выступать как передатчик или приемник, хотя некоторые периферийные устройства являются только приемниками.

Операции с данными

Устройство, которое начинает передачу данных, становится Master. Master генерирует тактовый сигнал SCL, а также инициирует и завершает передачу данных по шине.

Один бит данных передается по SDA в течение каждого импульса на SCL, см. рисунок 22.2. Данные стабильны, пока уровень сигнала на линии SCL высокий, могут меняться пока уровень сигнала на линии SCL низкий.

Каждая передача данных состоит из начального состояния «Старт», ряда передач байтов данных и состояния «Стоп» при окончании передачи данных. Первым передается старший разряд (Most Significant Bit – MSB). После каждого байта (8 бит) должен следовать сигнал подтверждения ACK.

Рисунок 22.2 – Передача бита данных

Slave может увеличивать паузу между тактовыми импульсами, удерживая на линии SCL сигнал низкого уровня, пока происходит обработка принятых данных или подготовка данных для следующей передачи. Этот процесс может происходить после передачи любого бита.

Состояния «Старт» и «Стоп»

Master формирует состояния «Старт» и «Стоп», см. рисунок 22.3. После того, как было сформировано состояние «Старт», шина считается занятой и другие ведущие не должны пытаться управлять ей. Такой статус шина сохраняет до тех пор, пока не будет сформировано состояние «Стоп».

Состояние «Старт» S – отрицательный перепад на SDA, когда сигнал на SCL высокого уровня.

Состояние «Стоп» P – положительный перепад на SDA, когда сигнал на SCL высокого уровня.

Состояние «Повторного старта» Sr, см. рисунок 22.4, – может быть сформировано в середине передачи, чтобы разрешить другому Slave обратиться к Master или чтобы изменить направление передачи данных без потери контроля над шиной.

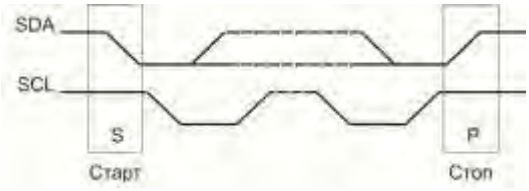


Рисунок 22.3 – Состояния «Старт» и «Стоп»

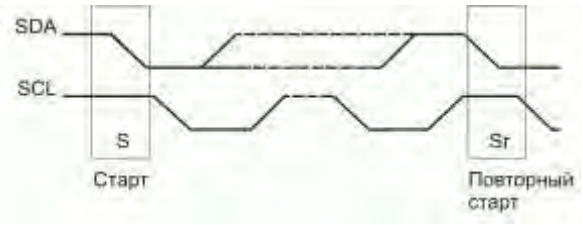


Рисунок 22.4 – Состояние «Повторный старт»

Цикл подтверждения

Цикл подтверждения, см. рисунки 22.5 и 22.6, включает в себя два сигнала:

- тактовый импульс подтверждения, который передается Master с каждым байтом данных;
- сигнал подтверждения ACK, посылаемый приемным устройством.

Рисунок 22.5 – Операции с данными

Во время девятого тактового импульса при передаче данных Master генерирует импульс подтверждения. Передатчик освобождает линию SDA (удерживает ее в высоком уровне), чтобы позволить приемнику отправить сигнал подтверждения. В течение импульса подтверждения приемник должен сформировать сигнал подтверждения, указывая тем самым на корректный прием последнего байта данных и готовность к приему следующего байта данных.

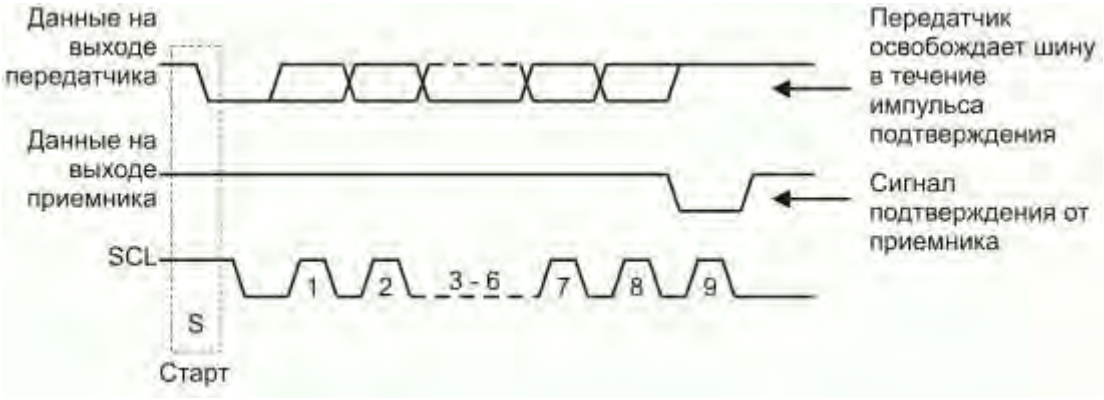


Рисунок 22.6 – Цикл подтверждения

Правило «Подтверждение после каждого байта»

Master генерирует импульс подтверждения после каждого переданного байта, приемник должен формировать сигнал подтверждения после каждого принятого байта.

Существует два исключения из правила «Подтверждение после каждого байта»:

- когда Master является приемником, то при завершении передачи данных во время девятого тактового импульса он формирует инверсный сигнал подтверждения NACK;
- если возникли проблемы при приеме или приемник переполнен, он формирует инверсный сигнал подтверждения NACK, показывая, что не может принять дополнительные данные.

Формат передачи данных с 7-битной адресацией

На рисунке 22.7 показана завершенная передача данных. Каждому устройству, подключенному к шине, присваивается уникальный 7-битный адрес. Первые семь бит, передаваемые после состояния «Старт» – адрес Slave, восьмой бит R/W# указывает направление передачи данных (передача Slave-устройством – чтение, передача Slave-устройству – запись).

Рисунок 22.7 – Завершенная передача данных

Каждое устройство, подключенное к шине, сравнивает принятый адрес со своим собственным адресом. Если адрес устройства совпадает с принятым им адресом, то устройством формируется сигнал подтверждения. В зависимости от значения бита R/W# 1_B – чтение, 0_B – запись, адресуемое устройство становится Slave-передатчиком или Slave-приемником.

Протокол I2C предоставляет возможность одновременного обращения к устройствам, подключенным к шине при помощи адреса общего вызова. В первом передаваемом байте определяется специальный адрес общего вызова 00_H, во втором – значение общего вызова. Slave, готовые к приему данных, формируют сигнал подтверждения, остальные игнорируют общий вызов.

Протокол I2C также допускает работу в режиме Alert-отклика (Slave Transmit Alert Response Mode). Любой Slave (или несколько Slave) может послать сигнал запроса SMBAlert,

с целью передать данные к Master. Master, получив сигнал SMBAlert, в ответ выдает на шину специальный адрес отклика 0001100_B с битом R/W#, равным 1_B (чтение).

Устройство (устройства), пославшее сигнал SMBAlert, получив адрес отклика, выдает на шину свой 7-битный адрес (последний бит передаваемого байта может быть равным как 0_B, так и 1_B). Если несколько Slave выдают свои адреса на шину, то передачу осуществляет устройство, выигравшее арбитраж.

Как только в ответ на появление адреса отклика одним из устройств будет выдан на шину корректный адрес, сигнал SMBAlert должен быть сброшен программно (поддержка режима Alert-отклика является уникальной особенностью устройств SMBus).

Арбитраж

Процесс распределения приоритетов (арбитраж) на линии SDA происходит в то время, когда сигнал на линии SCL высокого уровня. Арбитраж возникает в случае, когда два Master одновременно сформировали состояние «Старт», и продолжается до тех пор, пока одно устройство удерживает высокий уровень на SDA, в то время как другое удерживает низкий уровень на SDA (уровень сигнала на SCL – высокий). Master, удерживающий высокий уровень на SDA, теряет приоритет. Если Master теряет приоритет во время передачи адреса Slave (первый байт, следующий после формирования состояния «Старт»), то он переключается в режим «Slave приемник» и начинает сравнивать передаваемые адреса со своим адресом. Приоритет также может быть потерян в режиме «Master приемник» в течение цикла подтверждения или в режиме «Slave передатчик» во время получения сигнала отклика.

В случае потери приоритета устанавливаются соответствующие биты в статусном регистре SMBST (SMBST.MODE) и генерируется прерывание.

Синхронизация тактового сигнала

В случае, когда два и более Master пытаются одновременно начать передачу, необходима синхронизация их тактовых сигналов. Задача синхронизации решается просто благодаря тому, что все устройства подключаются к линии SCL по схеме «монтажное И». В результате длительность тактовых импульсов на линии SCL определяется тактовым сигналом с наименьшей длительностью импульсов, а пауза между тактовыми импульсами – тактовым сигналом с наибольшей паузой между импульсами.

Формат передачи данных с 10-битной адресацией

10-битная адресация, см. рисунок 22.8, позволяет использовать на шине I2C 1024 адреса Slave. После формирования состояния «Старт» передается зарезервированный адрес $11110XX_B$. 10-битная адресация полностью совместима с 7-битной адресацией. Таким образом, Slave с величиной адреса 7 бит и 10 бит могут быть подключены к одной и той же шине.

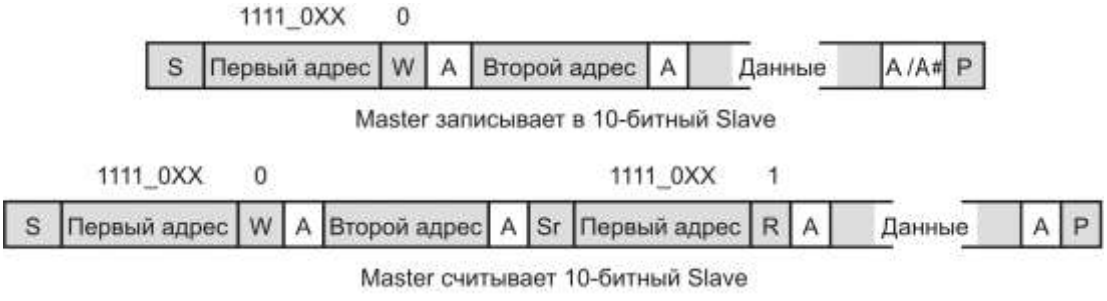


Рисунок 22.8 – 10-битная адресация Slave

Высокоскоростной режим Hs

По сравнению с режимом «Быстрый/Стандартный» F/S в высокоскоростном режиме повышается скорость передачи данных по шине. В режиме «Быстрый/Стандартный» скорость передачи ограничена 400 кГц и 100 кГц, соответственно, а в высокоскоростном режиме – 3,4 МГц, таким образом, пропускная способность увеличивается почти в 10 раз.

Шина переходит в высокоскоростной режим из режима «Быстрый/Стандартный» с помощью следующих действий:

Формирование состояния «Старт» S

8-битный адрес Master $00001YYY_B$, где YYY – уникальный адрес каждого мастера, подключенного к шине.

Инверсный сигнал подтверждения A#, передаваемый для Slave.

Так как адрес для каждого Master является уникальным, то процесс распределения приоритетов происходит только в момент передачи адреса Master. После передачи адреса один Master получает приоритет в работе с шиной. Таким образом, арбитраж и синхронизация тактового сигнала не выполняются в высокоскоростном режиме.

После выполнения вышеуказанных действий устройства, поддерживающие высокоскоростной режим, подключаются к шине. Далее Master формирует состояние «Повторного старта» Sr, затем передается адрес Slave и бит направления передачи R/W#.

Формат операций с данными в высокоскоростном режиме Hs аналогичен режиму F/S, таким образом, режим Hs полностью совместим с устройствами, работающими в режиме F/S.

Завершение режима Hs, см. рисунок 22.9, производится формированием состояния «Стоп» P, после чего все устройства переключаются в режим F/S.

Рисунок 22.9 – Высокоскоростной режим

22.2 Функциональное описание

На рисунке 22.10 показано строение модуля I2C.

Входные и выходные каскады линий SDA и SCL

Выводы SDA и SCL используются для подключения модуля к одноименным линиям шины I2C. Входные каскады, подключенные к выводам SDA и SCL, содержат фильтры, подавляющие помехи. В режиме F/S эти фильтры подавляют все входные сигналы с длительностью меньше, чем период тактового сигнала. Выходные каскады содержат устройства с предустановкой, выполненные по схеме с открытым стоком. Работа входных и выходных каскадов разрешается при разрешении работы модуля I2C (установлен бит SMBCTL2.ENABLE).

Регистр адреса и компаратор

В регистр адреса SMBADDR может быть записан 7-битный адрес, на который I2C отвечает, когда является Slave-устройством на шине. Установка старшего бита SAEN регистра SMBADDR разрешает распознавание адреса Slave.

Компаратор адреса сравнивает принятый 7-битный адрес с адресом, записанным в регистр SMBADDR. При разрешении общего вызова, когда установлен бит SMBCTL1.GCMEN, первый принятый байт сравнивается с величиной 00_H.

В режиме Alert-отклика, если установлен бит SMBCTL1.SMBARE, первый принятый байт сравнивается с величиной 0001100_B.

При 10-битной адресации (одновременно установлены биты SMBADDR.SAEN и SMBCTL3.S10EN) старшие 5 бит первого принятого адреса сравниваются с величиной 11110B, младшие 2 бита принятого адреса сравниваются с величиной, записанной в

SMBCTL3S10.2 – SMBCTL3S10.1. Старший бит второго принятого адреса сравнивается с SMBCTL3.SA10.0, младшие 7 бит с SMBADDR.ADDR.

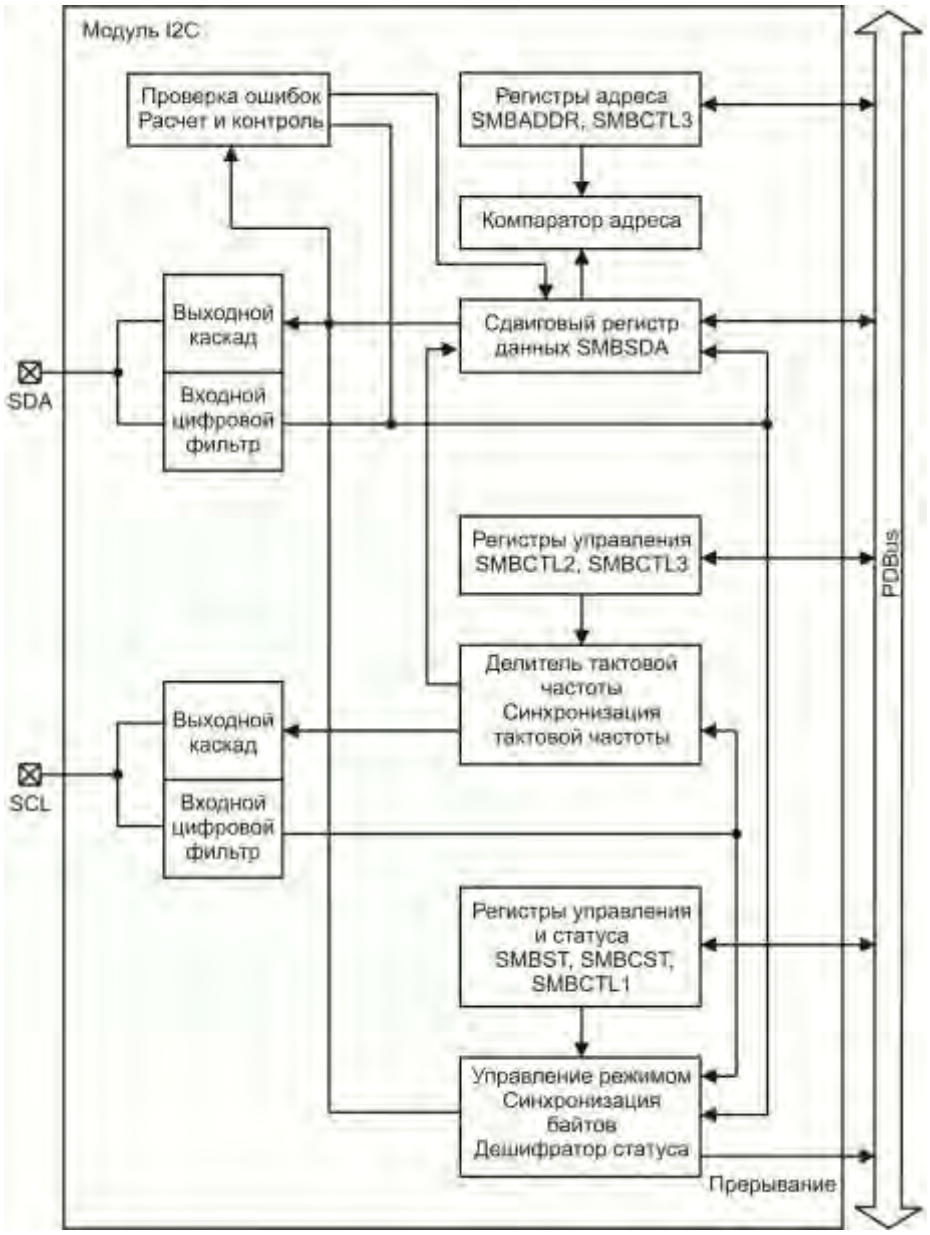


Рисунок 22.10 – Строение модуля I2C

Сдвиговый регистр данных

Сдвиговый регистр данных SMBSDA содержит байт данных, который может быть как принятым байтом данных, так и байтом данных, который необходимо передать. Данные одновременно принимаются и передаются, следовательно, SMBSDA всегда содержит последний байт данных, которые передавались по шине. Поэтому даже если произошла потеря приоритета, то корректные данные будут сохраняться в SMBSDA до момента окончания операции с данными. Данные передаются по заднему фронту SCL (первым передается MSB), принимаются – по переднему фронту SCL.

Процесс распределения приоритетов проходит во время передачи адреса, формирования состояний «Старт» и «Стоп». Если принятые данные отличаются от переданных, например, передана «1», а принят «0», то приоритет будет потерян.

Генерация тактового сигнала. Синхронизация

Последовательный тактовый сигнал, выходной сигнал I2C в режиме Master, формируется генератором из системного тактового сигнала.

В режиме F/S используется 7-битный прескалер. Значение старших 6 бит определяется полем SMBCTL2.SCLFRQ, младший бит всегда равен нулю (деление производится только на четное число). Минимальный коэффициент деления равен восьми, максимальный – 128.

Блок синхронизации тактового сигнала производит синхронизацию генератора тактового сигнала и выходного сигнала SCL с тактовым сигналом других устройств, подключенных к шине.

В режиме Hs коэффициент деления определяется величиной, записанной в поле SMBCTL3.HSCDIV. Прескалер уменьшен до 4 бит, значение младшего бита равно нулю.

Управление режимом и статус

Флаг прерывания находится в регистре SMBST. Также этот регистр содержит биты, показывающие режим работы I2C (Master или Slave, приемник или передатчик).

Регистр SMBCST является регистром управления и статуса, он показывает статус шины, возврат после ошибки.

Регистр SMBCTL1 управляет формированием состояний «Старт», «Повторный старт», «Стоп», а также сигнала подтверждения приема данных.

В регистрах SMBCTL2 и SMBCLT3 содержатся коэффициенты деления тактового сигнала, которые используются при работе I2C в режиме Master. Также регистр SMBCLT3 управляет 10-битной адресацией.

Функционирование

I2C поддерживает следующие режимы работы:

- режим «Неадресованный Slave»;
- режим «Master передатчик»;
- режим «Master приемник»;
- режим «Slave приемник»;
- режим «Slave передатчик».

Инициализация

Для начала работы шины I2C требуется произвести инициализацию. Для этого необходимо выполнение нижеследующих действий.

Разрешить работу I2C установкой бита SMBCTL2.ENABLE.

При работе в режиме Master для выбора периода тактового сигнала SCL необходимо записать нужный коэффициент деления в поле SMBCTL2.SCLFRQ.

В режиме Hs необходимо записать коэффициент деления в поле SMBCTL3.HSDIV.

При работе в режиме Slave значения в полях SMBCTL2.SCLFRQ и SMBCTL3.HSDIV не учитываются.

Для работы в режиме Slave необходимо записать в регистр SMBADDR значение адреса Slave и разрешить распознавание адреса установкой бита SMBADDR.SAEN.

Для разрешения расширенной 10-битной адресации записать значение старших разрядов адреса в поле SMBCTL3.S10AD и установить бит SMBCTL3.S10EN.

Установить бит SMBCTL1.GCMEN для разрешения распознавания адреса общего вызова (дополнительная функция).

Установить бит SMBCTL1.SMBARE для распознавания адреса отклика (дополнительная функция).

Если установлены требования к времени ожидания шины – записать соответствующую величину в биты SMBCST.TOCDIV и в регистр SMBTOPR для проверки времени активной работы линии SCL (максимум 25 мс). Запись в биты SMBCST.TOCDIV величины, отличной от нуля, автоматически разрешает проверку времени ожидания шины.

Для разрешения прерывания I2C установить бит SMBCTL1.INTEN.

Режим «Неадресованный Slave»

Режим «Неадресованный Slave» (общий режим) является режимом работы по умолчанию SMBST.MODE = IDLE (00_H). После разрешения работы I2C шина начинает работать в режиме «Неадресованный Slave». I2C непрерывно отслеживает состояние шины и при формировании состояния «Старт» или «Повторный старт» переходит в режим «Slave приемник». Из этого режима есть возможность перейти в режим «Master передатчик» после успешного формирования состояния «Старт».

I2C возвращается в режим «Неадресованный Slave» при выполнении следующих условий:

- состояние «Старт» не было успешно сформировано, так как другое устройство удерживает линию SCL в низком уровне;
- произошла потеря приоритета при передаче пакета данных в режиме «Master передатчик» или при передаче бита R/W# в режиме «Master приемник»;
- произошла потеря приоритета во время отклика;
- отправлен инверсный сигнал подтверждения в режиме «Slave приемник» (было запрещено распознавание адреса или адрес Slave не совпал);
- получен инверсный сигнал подтверждения в режиме «Slave передатчик»;
- сформировано состояние «Стоп»;
- ошибочное состояние на шине;
- сброс I2C;
- запрещение работы I2C.

Режим «Master передатчик»

Переход в режим «Master передатчик» производится после формирования сигнала «Старт». Первый байт, следующий после формирования сигнала «Старт», передается Master-устройством и содержит адрес Slave и бит направления передачи.

В зависимости от значения бита направления передачи R/W# I2C может как остаться в режиме «Master передатчик», так и перейти в режим «Master приемник», если R/W# = 1_B. Вместо последующей передачи адреса Slave, I2C может передать адрес Master ($00001XXX_B$) для перехода в высокоскоростной режим Hs.

I2C может перейти в режим Master с помощью установки бита SMBCTL1.START. I2C проверяет, свободна ли шина (бит SMBCST.BB = 0_B), и формирует состояние «Старт». Если шина занята (SMBCST.BB = 1_B), то состояние «Старт» будет сформировано в тот момент, когда шина будет свободна и сигнал на линии SCL будет в высоком уровне.

При успешном формировании состояния «Старт», бит SMBCTL1.START станет равным нулю, будет установлен флаг SMBST.INT и линия SCL будет удерживаться в низком уровне, пока флаг не будет сброшен (SMBST.MODE = 01_H – код статуса шины STDONE). Прерывание будет сгенерировано в случае, если установлен бит SMBCTL1.INTEN.

Передача адреса и данных:

Если бит SMBST.INT установлен, необходимо программно записать адрес Slave и направление передачи в сдвиговый регистр данных SMBSDA.

После записи в регистр SMBSDA необходимо сбросить флаг прерывания SMBST.INT – очистить бит SMBCTL1.CLRST.

Линия SCL будет освобождена после сброса флага SMBST.INT и времени установки данных. Затем содержимое регистра SMBSDA будет передано на шину.

После передачи всех битов данных и получения сигнала подтверждения (после девятого тактового импульса), бит подтверждения анализируется системой, биты SMBST.MODE показывают код статуса шины I2C в зависимости от принятого адреса.

В течение передачи данных на линиях SDA и SCL отслеживаются потенциальные конфликты с другими устройствами, подключенными к шине. Если зафиксирован конфликт, передача данных прекращается. Код статуса шины определяется как SRAAPA (SMBST.MODE = 11_H), если потерян приоритет в режиме «Slave приемник» или как IDLARL (SMBST.MODE = 03_H), если потерян приоритет в режиме «Неадресованный Slave».

Если бит направления передачи R/W# = 1_B, I2C переходит в режим «Master приемник», если не произошло ошибок.

Если выбрано требуемое направление передачи и передача адреса Slave успешно завершена (код статуса шины не MTADNA SMBST.MODE = 05_H и не BERROR SMBST.MODE = $1F_H$), то устанавливается флаг SMBST.INT, показывающий необходимость записи первого байта данных в регистр SMBSDA. Пока установлен флаг SMBST.INT, SCL удерживается в низком уровне, прерывание будет сгенерировано, если установлен бит SMBCTL1.INTEN.

В случае, когда после передачи данных Slave приемник отправляет инверсный сигнал подтверждения, на SCL будет удерживаться низкий уровень, прерывание будет сгенерировано, если установлен бит SMBCTL1.INTEN (код статуса шины MRDANA SMBST.MODE = $0B_H$).

Отслеживание ошибок в пакетах данных

При работе I2C в режиме «Master передатчик», см. таблицу 22.1, рисунок 22.11, после установки бита SMBCST.PECNEXT произойдет передача содержимого регистра Packet Error Checking в регистр SMBSDA и начало передачи байта Cyclic Redundancy Check (CRC) для Slave-устройства. Это произойдет после передачи последнего байта данных и до того, как будут сформированы состояния «Старт» и «Повторный старт».

Таблица 22.1 – Режим F/S «Master передатчик», действия и статус

Код	Название	Описание	Данные		Управление SMBCTL1				Следующее действие
			R/W	SMBSDA	clrst	ask	stop	start	
1	2	3	4	5	6	7	8	9	10
01_H	STDONE	Сформировано состояние «Старт»	W	Адрес Master	1	0	0	0	Передача кода Master, переход в режим Hs
				SADR/ RW = 0_B					Передача SADR/RW – ACK или NACK от Slave
02_H	RSDONE	Сформировано состояние Повторный «Старт»	W	SADR/ RW = 0_B	1	0	0	0	Передача SADR/RW – ACK или NACK от Slave
				SADR/ RW = 1_B					Передача SADR/RW – ACK или NACK от Slave, I2C переходит в режим «Master приемник»
03_H	IDLARL	Потеря приоритета, переход в режим «Неадресованный Slave»	–		1	0	0	0	Режим «Неадресованный Slave»

Окончание таблицы 22.1

1	2	3	4	5	6	7	8	9	10
04_H	MTADPA	Отправлен адрес Slave, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
05_H	MTADNA	Отправлен адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	Состояние «Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
06_H	MTDAPA	Отправлен байт данных, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Старт»
07_H	MTDANA	Отправлен байт данных, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»

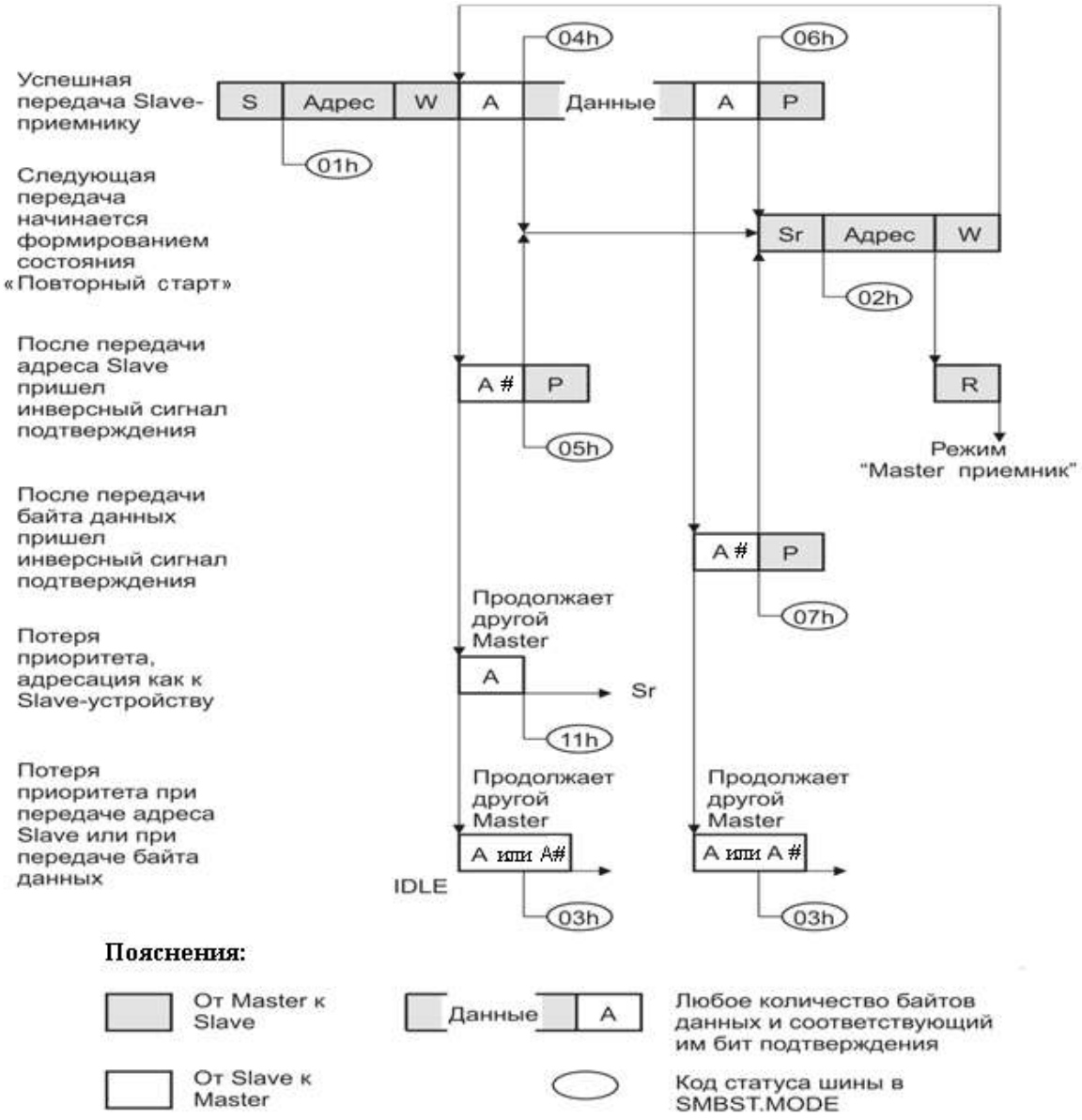


Рисунок 22.11 – Режим F/S «Master передатчик», действия и статус

Повторный старт

I2C при работе в режиме «Master передатчик» осуществляет полный контроль над шиной и может адресовать другого Slave, изменить направление передачи, без потери контроля над шиной.

Для формирования состояния «Повторный старт» необходимо:

- установить бит SMBCTL1.START;
- в режиме «Master приемник» считать последний элемент данных из регистра SMBSDA;
- сбросить ожидаемый флаг прерывания установкой бита SMBCTL1.CLRST;
- впоследствии линия SCL будет освобождена, будет сгенерировано состояние «Повторный старт» и прерывание I2C (код статуса шины RSDONE).

Выход из режима «Master передатчик»

Выход из режима «Master передатчик» осуществляется формированием сигнала «Стоп». Чтобы сформировать сигнал «Стоп» необходимо:

- установить бит SMBCTL1.STOP;
- в режиме «Master приемник» считать последний элемент данных из регистра SMBSDA;

- сбросить ожидаемый флаг прерывания установкой бита SMBCTL1.CLRST.

Выполнение данных условий вызывает немедленное формирование I2C состояния «Стоп» и обнуление бита SMBCTL1.STOP. Состояние «Стоп» может быть сформировано только в том случае, когда I2C является активным Master-устройством шины SMBST.MODE = 01_H – $0B_H$.

Высокоскоростной режим Hs «Master передатчик»

Вход в высокоскоростной режим «Master передатчик» осуществляется формированием состояния «Старт» с последующей передачей адреса Master ($00001XXX_B$) вместо адреса Slave. После того, как адрес Master будет предан, установится флаг SMBST.INT, сгенерируется прерывание (если был установлен бит SMBCTL1.INTEN). После успешной передачи адреса Master код статуса шины станет HMTOK. В этот момент I2C входит в режим высокоскоростной режим «Master передатчик», см. рисунок 22.12, таблицу 22.2.

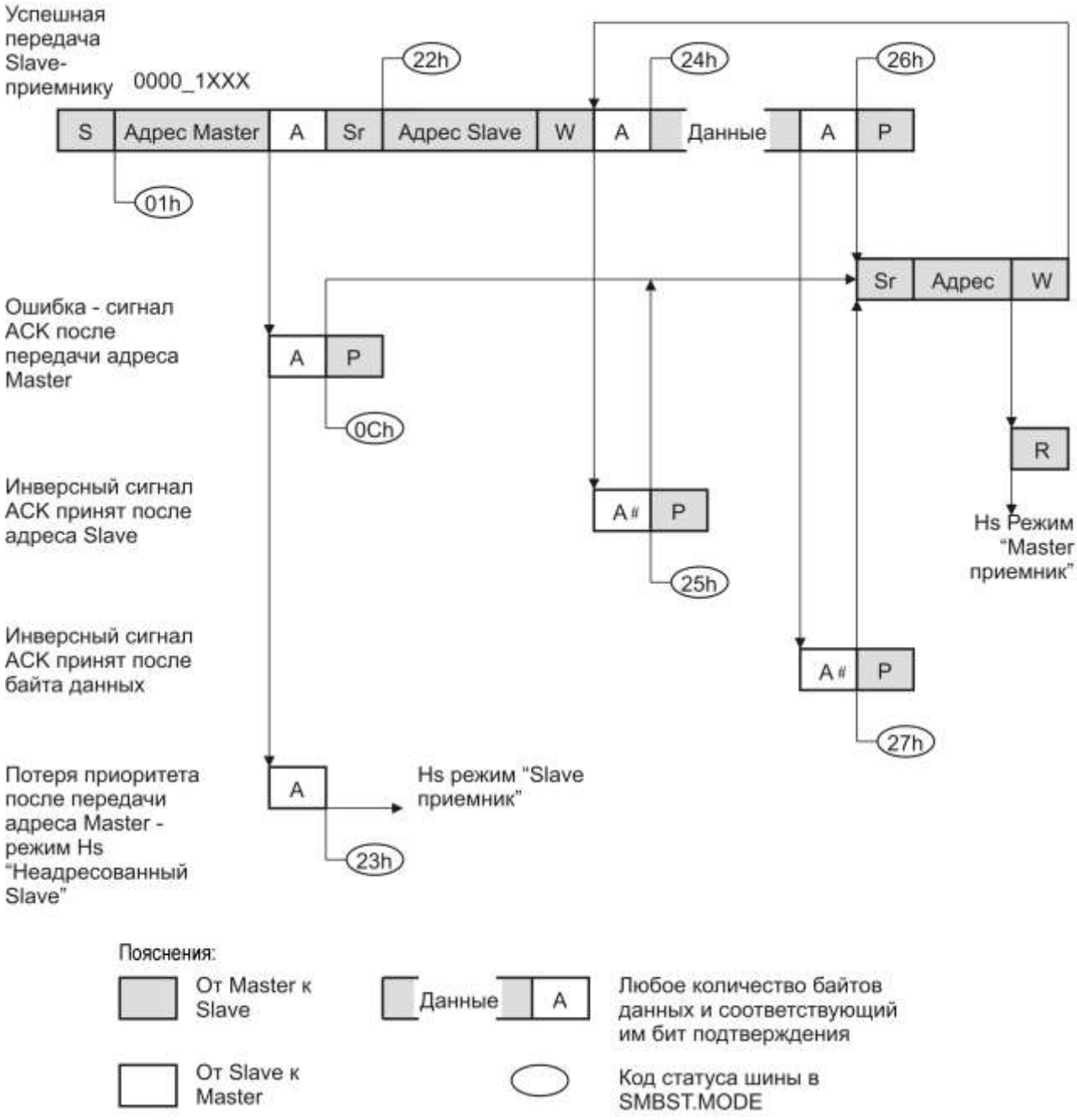


Рисунок 22.12 – Режим Hs «Master передатчик», действия и статус

Таблица 22.2 – Режим Hs «Master передатчик», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
$0C_H$	MTMCER	Передан код Master, найдена ошибка (сигнал ACK)	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
21_H	HMTMCOK	Адрес Master успешно передан, переход в режим Hs	–		1	0	0	1	«Повторный старт»
22_H	HRSDONE	Сформирован «Повторный старт»	W	SADR/ RW = 0	1	0	0	0	Передача SADR/RW – сигнал ACK или NACK от Slave
				SADR/ RW = 0					Передача SADR/RW – сигнал ACK или NACK от Slave, I2C перейдет в режим Hs «Master приемник»
23_H	HIDLARL	Потеря приоритета, переход в режим Hs «Неадресованный Slave»	–		1	0	0	0	Режим «Неадресованный Slave»
24_H	HMTADPA	Отправлен адрес Slave, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
25_H	HMTADNA	Отправлен адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
26_H	HMTDAPA	Отправлен байт данных, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
27_H	HMTDANA	Отправлен байт данных, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»

Необходимо сформировать состояние «Повторный старт» установкой бита SMBCTL1.START и сбросить флаг прерывания установкой бита SMBCTL1.CLRST.

После формирования сигнала «Повторный старт» установится флаг SMBST.INT, код статуса в SMBST.MODE станет HRSDONE. Затем последует передача адреса и данных (см. «Передача адреса и данных»).

Режим «Master приемник»

Вход в режим «Master приемник» осуществляется после успешной передачи адреса Slave с битом направления передачи, равным единице R/W# = 1_B. В режиме «Master приемник» I2C считывает с Slave-устройства, подключенного к шине, и, тем не менее, продолжает контролировать линию SDA. I2C продолжает генерировать импульсы SCL и подтверждать каждый принятый байт данных, см. рисунок 22.13, таблицу 22.3.

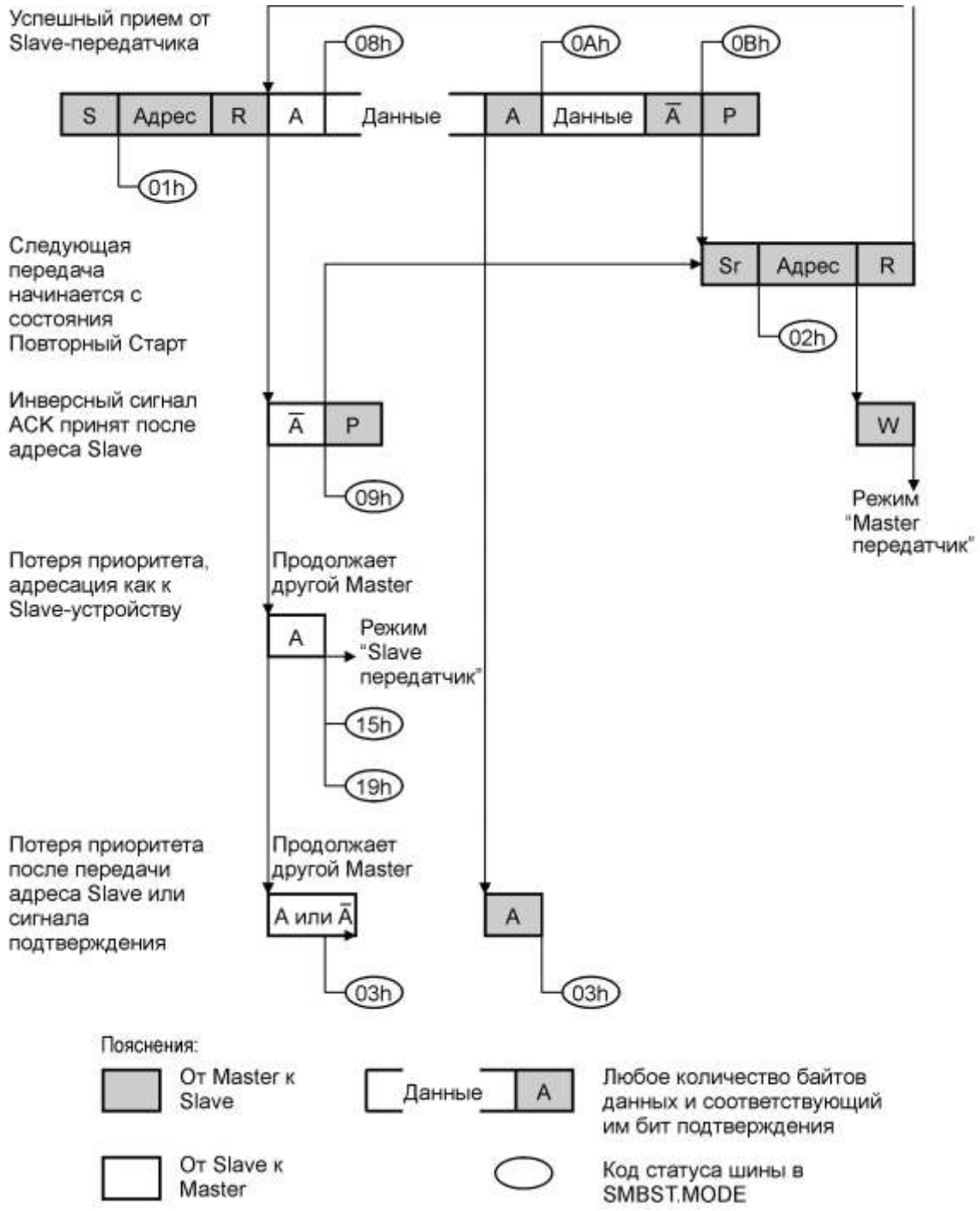


Рисунок 22.13 – Режим F/S «Master приемник»

Таблица 22.3 – Режим F/S «Master приемник»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
08_H	MRADPA	Передан адрес Slave, сигнал ACK	–		1	0	0	0	Принят байт данных, сформирован сигнал ACK
					1	1	0	0	Принят байт данных, сформирован сигнал NACK
09_H	MRADNA	Передан адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
$0A_H$	MRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$0B_H$	MRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»

После каждого принятого байта устанавливается бит SMBST.INT и данные программно считываются из регистра SMBSDA. Линия SCL удерживается в низком уровне, пока установлен флаг SMBST.INT. Установка бита SMBCTL1.CLRST очищает бит SMBST.INT, разрешая прием следующего байта данных.

Протокол шины устанавливает, что состояния «Повторный старт» или «Стоп» не должны формироваться, пока работа ведется в режиме «Master приемник» и I2C уже не является единственным устройством, контролирующем SDA. Для восстановления полного контроля над шиной в соответствии с протоколом шины Master-приемнику необходимо отправить инверсный сигнал подтверждения после приема последнего байта данных.

Для этого необходимо очистить бит SMBCTL1.ACK до сброса флага SMBST.INT во время приема предпоследнего байта. В то же самое время должен быть установлен бит SMBCST.PECNEXT, если есть запрос на PEC. Когда флаг SMBST.INT будет сброшен, линия SCL станет свободной и будет принят последний байт данных, в течение девятого тактового импульса SCL будет отправлен инверсный сигнал ACK. Затем I2C вернется в режим «Master передатчик» и сможет сформировать на шине состояния «Повторный старт» или «Стоп».

Проверка ошибок пакета данных

При разрешении проверки ошибок пакета данных PEC, последний байт, принятый от передающего Slave-устройства, будет байт PEC. Если результат вычислений CRC отличен от нуля, то будет установлен бит SMBCST.PECFAULT, указывающий на ошибку.

Высокоскоростной режим «Master приемник»

Вход в высокоскоростной режим «Master приемник» осуществляется, если после передачи адреса Master следует адрес Slave с битом направления передачи, равным единице ($R/W\# = 1_B$), а затем формируется состояние «Повторный старт». После входа I2C в

высокоскоростной режим «Master приемник» устанавливается флаг SMBST.INT, коды статуса шины показаны в таблице 22.4, на рисунке 22.14.

Таблица 22.4 – Высокоскоростной режим Hs «Master приемник», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
28_H	HMRADPA	Передан адрес Slave, сигнал ACK	–		1	0	0	0	Принят байт данных, сформирован сигнал ACK
					1	1	0	0	Принят байт данных, сформирован сигнал NACK
29_H	HMRADNA	Передан адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
$2A_H$	HMRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$2B_H$	HMRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»

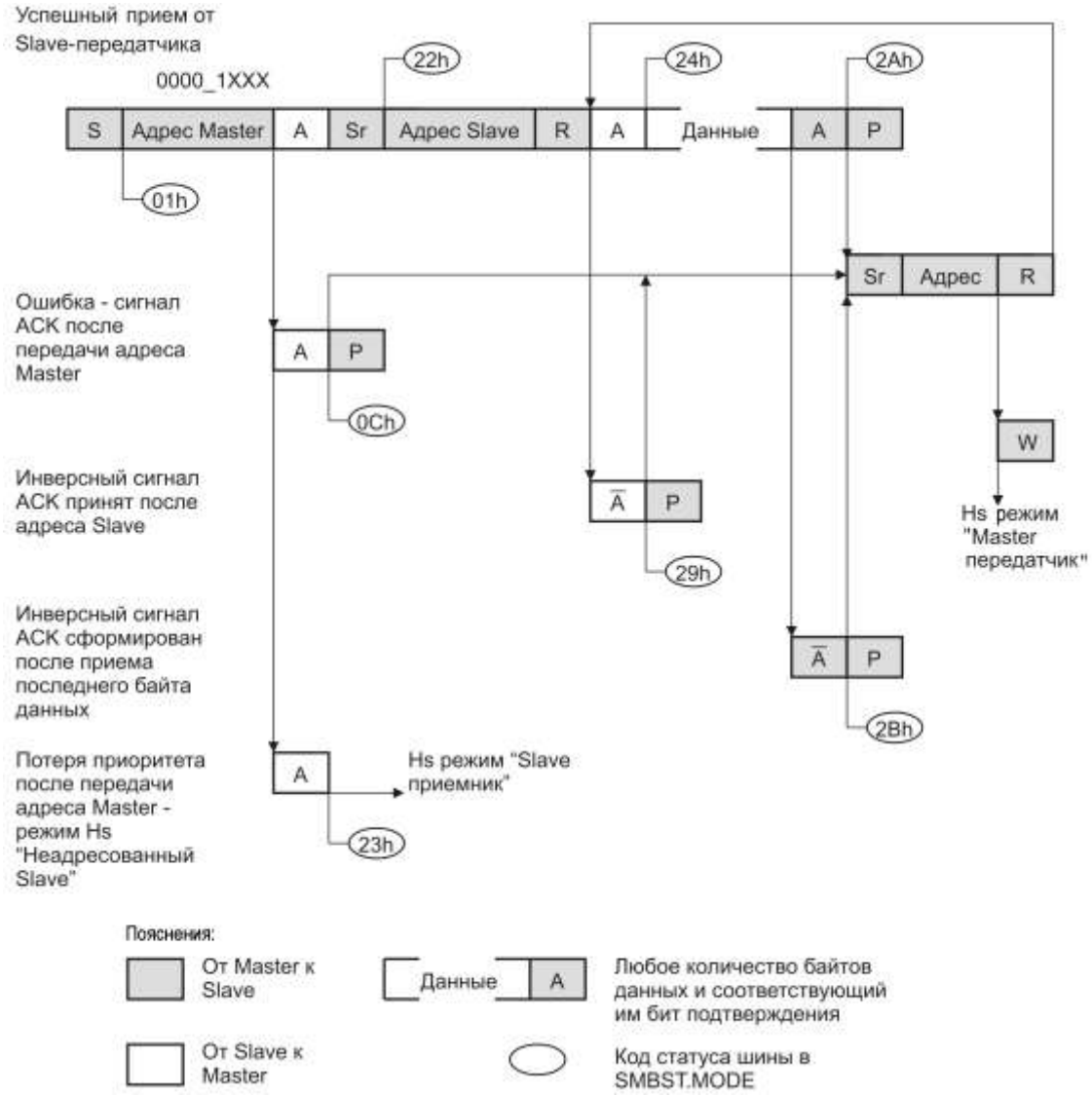


Рисунок 22.14 – Режим Hs «Master приемник»

Режим «Slave приемник»

В режиме «Slave приемник» данные принимаются от Master-передатчика. Сигнал подтверждения формируется после приема каждого байта.

Вход в режим «Slave приемник»

После разрешения на работу I2C продолжает контролировать шину. После обнаружения состояния «Старт», I2C входит в режим «Slave приемник», см. таблицу 22.5, рисунок 22.15, и начинает прием 7-битного адреса и бита направления передачи R/W# от Master. Переход в режим «Slave приемник» также может произойти из режима «Master передатчик» в случае потери приоритета при передаче адреса или данных.

После приема полного адреса I2C сравнивает принятый адрес:

- с полем SMBADDR.ADDR, если установлен бит SMBADDR.SAEN;
- с адресом общего вызова 00_H, если установлен бит SMBCTL1.GCMEN;
- с адресом отклика 0001100_B, если установлен бит SMBCTL1.SMBARE.

Сигнал подтверждения формируется в течение девятого тактового импульса на SCL, если принятый адрес совпал с адресом, записанным в регистр, адресом общего вызова или адресом отклика. После того, как произошло совпадение, устанавливается флаг SMBST.INT, изменяется значение SMBST.MODE. Если установлен бит SMBCTL1.INTEN, то генерируется прерывание. В регистре SMBSDA содержится принятый байт, включающий адрес Slave и бит направления передачи.

В зависимости от принятого бита направления передачи I2C может перейти в режим «Slave передатчик» R/W# = 1_B или остаться в режиме «Slave приемник» R/W# = 0_B.

После каждого принятого байта данных устанавливается флаг SMBST.INT, показывая необходимость считать данные из регистра SMBSDA, линия SCL удерживается в низком уровне. Необходимо программно считать принятый байт данных, а затем установить бит SMBCTL1.CLRST, чтобы сбросить флаг SMBST.INT и освободить линию SCL.

Режим «Slave приемник» – 10-битная адресация

Для использования внешней (10-битной) адресации необходимо установить бит SMBCTL2.S10EN и бит SMBADDR.SAEN. Старшие биты адреса необходимо записать в поле SMBCTL3.S10AD. После формирования сигнала «Старт» и приема первого байта, I2C сравнивает содержимое сдвигового регистра данных с величиной $11110YY_B$, где YY – являются двумя наиболее значащими битами адреса Slave, хранимыми в двух старших битах поля SMBCTL3.S10AD. Бит направления передачи должен быть равен нулю R/W# = 0_B, показывая Slave-устройству то, что будет принята вторая часть 10-битного адреса.

Если две величины равны, то I2C формирует сигнал подтверждения и продолжает принимать младшие биты адреса. После приема первого байта не генерируется прерывание.

Master-передатчик затем передает второй байт 10-битного адреса Slave. Старший бит принятого адреса сравнивается с младшим битом поля SMBCTL3.S10AD, а младшие 7 бит принятого адреса сравниваются с величиной, определяемой полем SMBADDR.SADDR. Если величины равны, то формируется сигнал подтверждения, при этом установится флаг SMBST.INT, код статуса шины станет соответственно SRADPA или SRAPAA.

Проверка ошибок пакета данных

При разрешенной проверке ошибок пакета данных PEC последний байт, принятый от Master, будет байтом PEC. Если результат вычислений CRC не равен нулю, будет установлен бит SMBCST.PECFAULT, указывая на произошедшую ошибку, будет передан инверсный сигнал подтверждения. Необходимо точно знать количество байтов, передаваемых Master-устройством шины, и при чтении предпоследнего байта из регистра SMBSDA необходимо установить бит SMBCST.PECNEXT, а затем установить SMBCTL1.CLRST.

Выход из режима «Slave приемник»

Если Slave приемник больше не может принимать данные от Master, необходимо установить бит SMBCTL1.ACK до того, как будет установлен бит SMBCTL1.CLRST, сбрасывающий флаг SMBST.INT. Это послужит причиной тому, что после принятия последнего байта данных от Master, будет сформирован инверсный сигнал подтверждения. После принятия последнего байта данных установится флаг SMBST.INT. Затем необходимо программно считать последний принятый байт из регистра SMBSDA и записать единицу в бит SMBCTL1.CLRST, чтобы сбросить флаг прерывания SMBST.INT и освободить шину SCL.

Таблица 22.5 – Режим F/S «Slave приемник», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
10_H	SRADPA	Принят адрес Slave, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
11_H	SRAAPA	Принят адрес Slave после потери приоритета, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
12_H	SRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
13_H	SRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	0	Режим «Неадресованный Slave»
					1	0	0	1	Режим «Неадресованый Slave», формирование сигнала «Старт» после того, как шина станет свободна
$1C_H$	SSTOP	Режим Slave, обнаружено состояние «Стоп»	–		1	0	0	0	Режим «Неадресованный Slave»
					1	0	0	1	Режим «Неадресованый Slave», формирование сигнала «Старт» после того, как шина станет свободна
$1D_H$	SGADPA	Принят адрес общего вызова, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$1E_H$	SGAAPA	Принят адрес общего вызова после потери приоритета, сигнал ACK			1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK

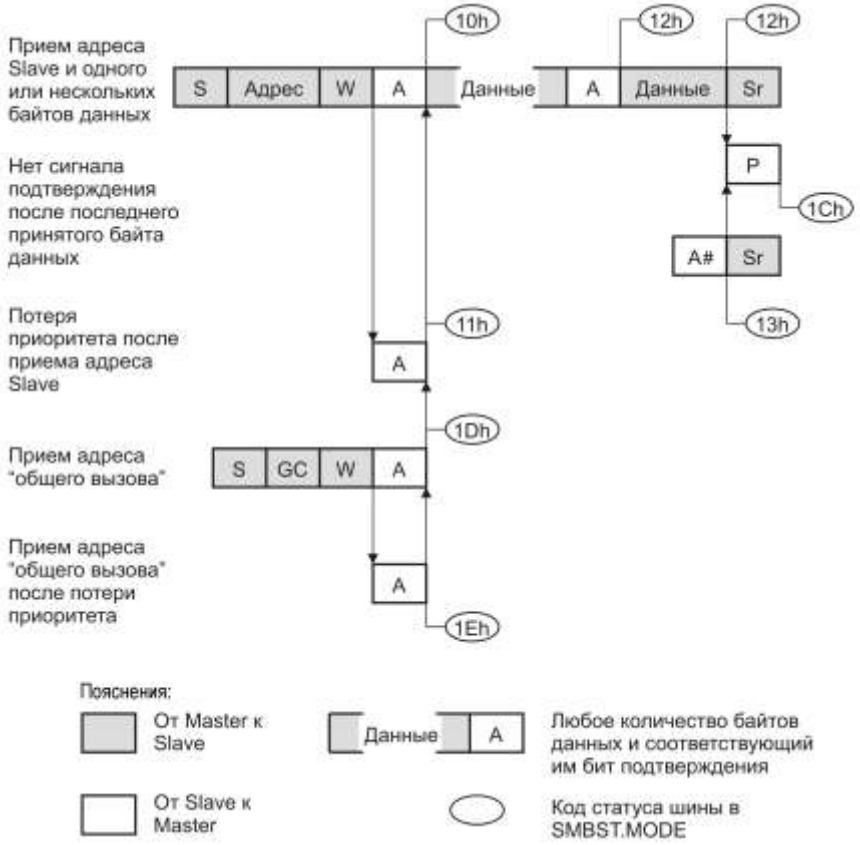


Рисунок 22.15 – Режим F/S «Slave приемник», действия и статус

Высокоскоростной режим «Slave приемник»

Вход в режим Hs «Slave приемник» осуществляется после приема адреса Master ($00001XXX_B$). После адреса Master следует состояние «Повторный старт» и адрес Slave, включающий бит направления, равный нулю $R/W\# = 0_B$. После приема адреса Slave I2C производит сравнение адресов, см. рисунок 22.16. Более подробные действия указаны в таблице 22.6.

Таблица 22.6 – Режим Hs «Slave приемник»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
30_H	HSRADPA	Принят адрес Slave, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
32_H	HSRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
33_H	HSRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	0	«Неадресованный Slave»
					1	0	0	1	«Неадресованный Slave», «Старт» после освобождения шины

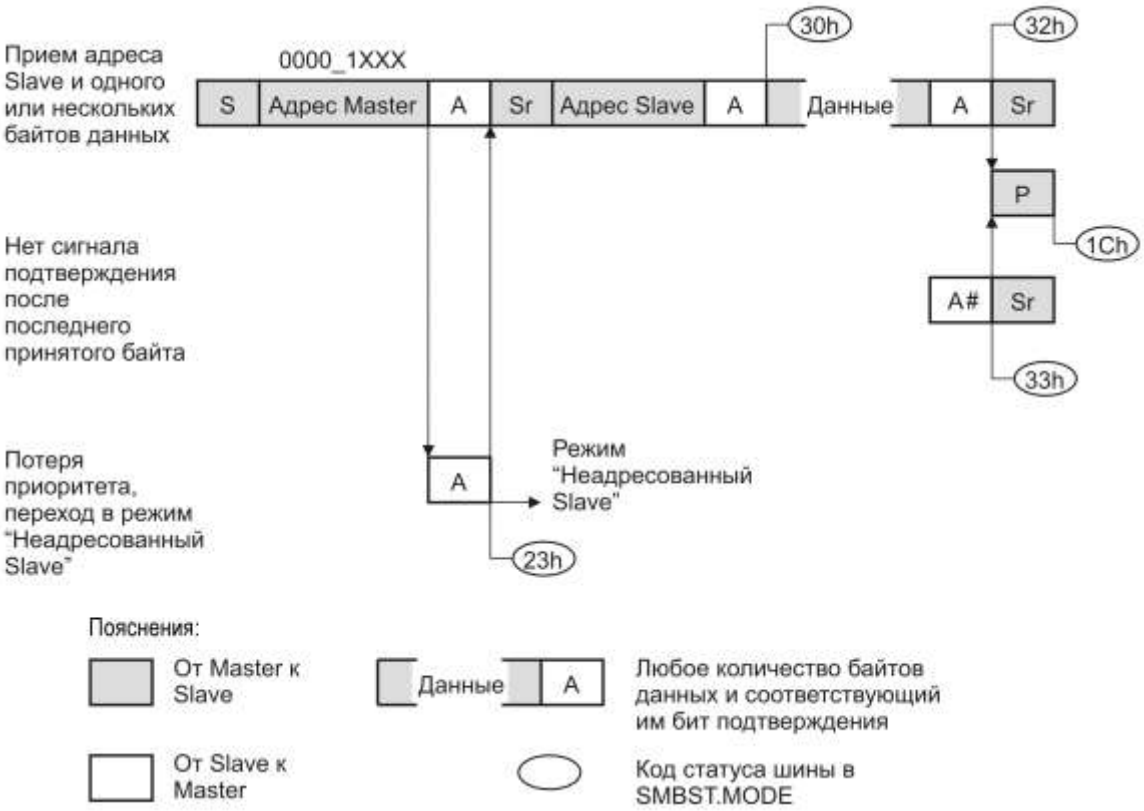


Рисунок 22.16 – Режим Hs «Slave приемник»

Режим «Slave передатчик»

В режиме «Slave передатчик» любое количество байтов данных может быть передано от I2C Master-приемнику. I2C проверяет бит подтверждения в течение девятого тактового импульса при передаче данных.

Вход в режим «Slave передатчик»

Вход в режим «Slave передатчик» осуществляется из режима «Slave приемник» после адресации I2C «Master-передатчиком». Бит направления передачи, следующий после адреса Slave, должен быть равным единице R/W# = 1_B. После этого установится флаг SMBST.INT, показывая необходимость записи данных в регистр SMBSDA. Линия SCL будет удерживаться в низком уровне, пока флаг SMBST.INT не будет сброшен. Когда первый байт передаваемых данных будет записан в регистр SMBSDA, необходимо программно установить бит SMBCTL1.CLRST, который сбросит флаг SMBST.INT. Затем линия SCL станет свободной, и будет передан байт данных.

Передача данных в этом режиме сходна с передачей в режиме «Master передатчик». Флаг SMBST.INT устанавливается после каждой успешно завершенной передачи данных, SMBST.MODE содержит соответствующий код статуса шины. Каждый следующий байт данных необходимо записывать в регистр SMBSDA, если SMBST.MODE не содержит код статуса шины STDANA, показывающий, что Master не может принять дополнительных данных.

Выход из режима «Slave передатчик» может быть осуществлен только с помощью Master-приемника. В этом случае Master-приемник формирует инверсный сигнал подтверждения после последнего принятого байта. После обнаружения инверсного сигнала подтверждения, I2C переходит в режим «Неадресованный Slave» (SMBST.MODE = IDLE) и ждет формирования на шине сигнала «Старт», см. таблицу 22.7.

Таблица 22.7 – Режим F/S «Slave передатчик»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
14_H	STADTA	Принят адрес Slave, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
15_H	STADPA	Принят адрес Slave после потери приоритета, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
16_H	STDAPA	Передача байта данных, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
17_H	STDANA	Передача байта данных, сигнал NACK	–		1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна
18_H	SATADP	Принят адрес отклика, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
19_H	SATAAP	Принят адрес отклика после потери приоритета, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
$1A_H$	SATDAP		W	Данные	1	X	0	0	Передача байта данных
$1B_H$	SATDAN				1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна

Режим «Slave передатчик» – 10-битная адресация

Для использования в режиме «Slave передатчик» 10-битной адресации должен быть записан адрес Slave и разрешена 10-битная адресация, см. «Режим «Slave приемник» и рисунок 22.17.

Сначала I2C входит в режим «Slave приемник», чтобы принять полный 10-битный адрес Slave. Флаг SMBST.INT не будет установлен, линия SCL не будет удерживаться в низком уровне, статус кода шины в SMBST.MODE не изменится.

После формирования состояния «Повторный старт» последует второй байт адреса Slave, а затем – повторная передача старших битов адреса с битом направления передачи, равным единице R/W# = 1_B. Если принятый адрес совпадет с адресом, записанным в регистрах, то будет установлен флаг SMBST.INT и I2C перейдет в режим «Slave передатчик», код статуса шины в поле SMBST.MODE будет STADPA или STAAPA.

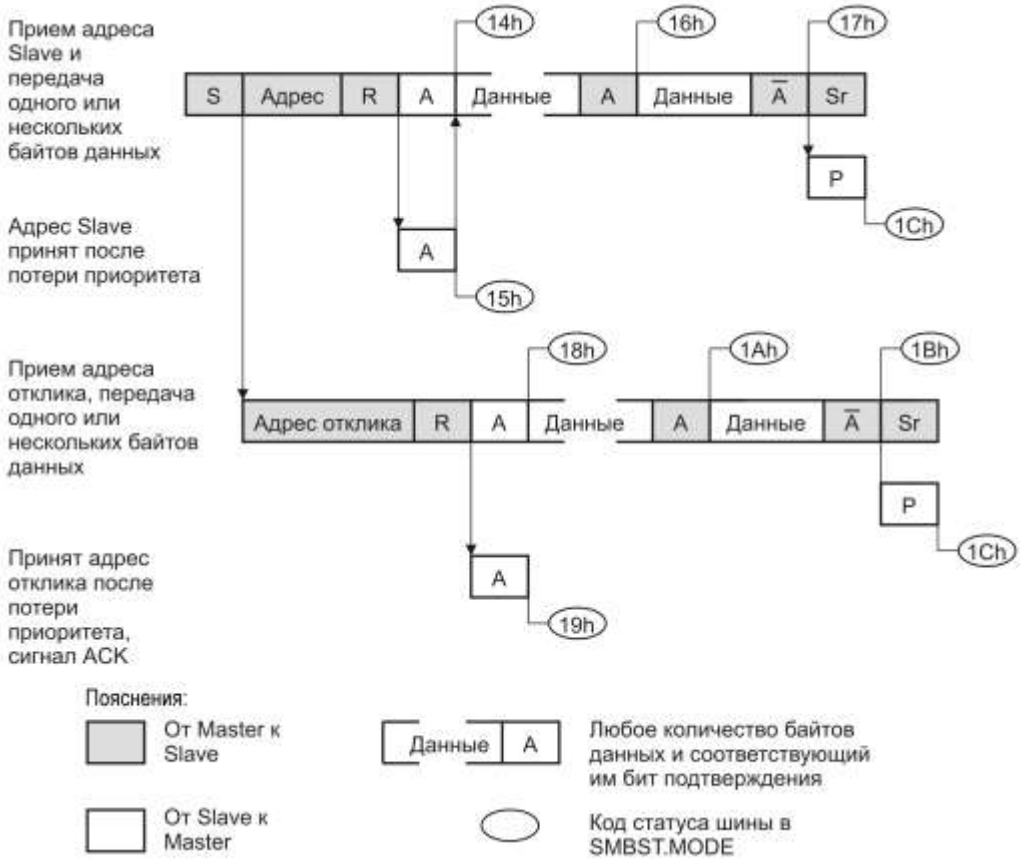


Рисунок 22.17 – Режим F/S «Slave передатчик»

Проверка ошибок пакета данных

При разрешении проверки ошибок пакета данных (PEC) последний байт, переданный Master-устройством, будет байтом PEC. Необходимо установить бит SMBCST.PECNEXT для записи байта PEC в регистр SMBSDA.

Режим Alert-отклика

Любой Slave (или несколько Slave) может послать сигнал запроса SMBAlert#, с целью передать данные к Master. Master, получив сигнал SMBAlert#, в ответ, инициализирует цикл отклика – выдает на шину специальный адрес отклика 0001100_B с битом R /W#, равным 1_B (чтение).

Устройство (устройства), пославшее сигнал SMBAlert# и получившее адрес отклика, выдает на шину свой 7-битный адрес (последний бит передаваемого байта может быть равным как 0_B, так и 1_B). Если несколько Slave выдают свои адреса на шину, то передачу осуществляет устройство, выигравшее арбитраж.

Master, который не послал адрес отклика (не инициализировал цикл отклика) в ответ на сигнал SMBAlert#, может сделать это через некоторое время.

Переход в режим Alert-отклика осуществляется после появления на шине SMBus адреса отклика 0001100_B, если установлен бит SMBCTL1.SMBARE.

Этот режим является единственным случаем, когда при функционировании I2C в качестве Slave используется арбитраж при передаче адреса.

Высокоскоростной режим «Slave передатчик»

После передачи адреса Master $00001XXX_B$ осуществляется вход в высокоскоростной режим. Затем следует формирование состояния «Повторный старт», передача адреса Slave с битом направления передачи, равным единице R/W# = 1_B. После этого I2C переходит в режим «Slave передатчик». Работа в Hs режиме «Slave передатчик» почти идентична работе в режиме F/S, отличие заключается только в более высокой скорости передачи данных и в кодах статуса шины, см. таблицу 22.8, рисунок 22.18.

Таблица 22.8 – Режим Hs «Slave передатчик»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
34_H	HSTADPA	Принят адрес Slave, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
36_H	HSTDAPA	Передача байта данных, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
37_H	HSTDANA	Передача байта данных, сигнал NACK	–		1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна

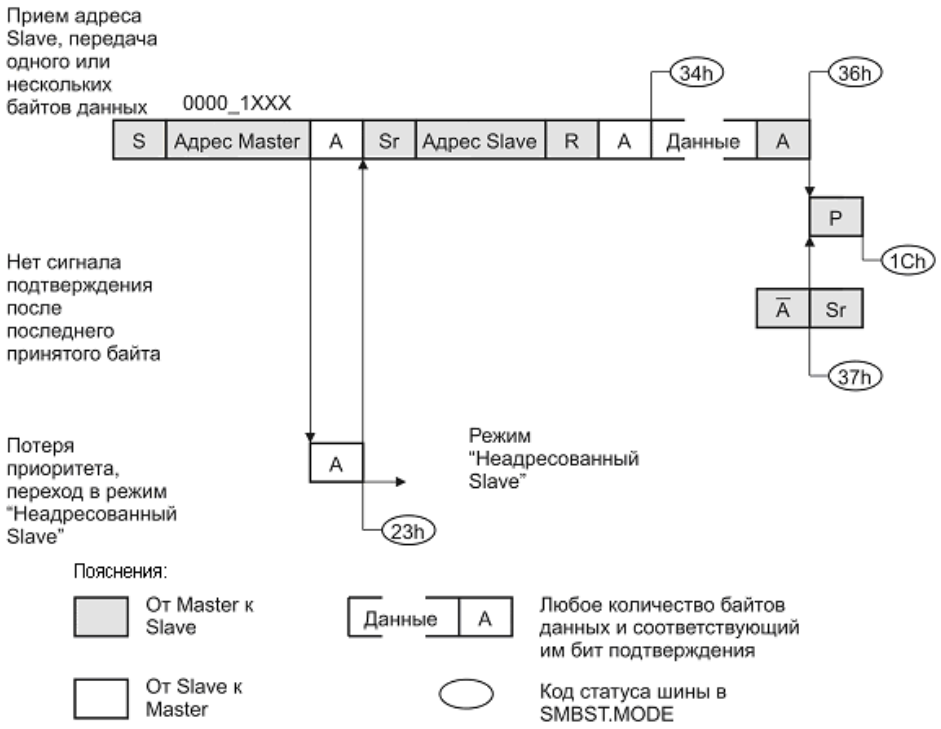


Рисунок 22.18 – Режим Hs «Slave передатчик»

Арбитраж и обнаружение ошибок на шине

Арбитраж

Приоритет в режиме «Master передатчик» может быть потерян в случаях, когда два Master одновременно формируют состояние «Старт» и начинают передачу данных. I2C по-разному реагирует на потерю приоритета в случаях, когда она произошла при передаче байта адреса или при передаче байта данных.

В случае потери приоритета при передаче байта адреса, I2C переходит в режим «Slave приемник» и начинает отслеживать передачу своего адреса Slave на шине. Если адреса совпали, I2C остается в режиме Slave. Если адреса не совпали, I2C переходит в режим «Неадресованный Slave».

Как только произошла потеря приоритета при передаче байта данных, I2C переходит в режим «Неадресованный Slave».

Обнаружение ошибок на шине

Ошибки на шине случаются, если во время передачи адреса, данных, сигнала подтверждения обнаруживаются состояния «Старт» или «Стоп», см. таблицу 22.9. При нахождении ошибки на шине, I2C действует следующим образом:

Код статуса шины в SMBST.MODE становится BERROR ($1F_H$).

Генерируется прерывание, если был установлен бит SMBCTL1.INTEN.

I2C переходит из данного режима в режим «Неадресованный Slave».

Линии SDA и SCL становятся свободны.

Таблица 22.9 – Ошибки на шине, действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
00_H	IDLE	Информация о режиме отсутствует	–		–				Ожидание завершения текущей передачи данных
$1F_H$	BERROR	Обнаружены некорректные состояния «Старт» или «Стоп»	–		1	0	0	0	Режим «Неадресованный Slave»

Исправление ошибок на шине

В некоторых случаях I2C и другое устройство, подключенное к шине, могут обнаружить ошибку и вызвать простой шины. В этом случае может быть не завершено формирование состояния «Старт» и I2C перестанет функционировать.

Для возврата из этого состояния необходимо выполнить следующие действия:

- запретить и снова разрешить работу I2C (стачала обнулить бит SMBCTL2.ENABLE, а затем записать туда единицу);

- во время простоя проверить, не подключен ли другой активный Master к шине (бит SMBCST.BB остается равным «0»).

В этот момент некоторые Slave могут не распознать ошибку на шине. Для исправления ошибки I2C становится Master-устройством шины, формируя состояние «Старт» и передавая адрес. Затем I2C формирует состояние «Стоп», чтобы синхронизировать все Slave-устройства.

Режим Idle и режим Halt

В режимах Halt или Idle системный тактовый сигнал остановлен и на шине не производятся операции с данными. Таким образом, вход в режимы Halt или Idle не должен осуществляться при работе I2C в качестве Master или Slave при передаче или приеме данных. Следовательно, I2C должна работать в режиме «Неадресованный Slave» при входе в режимы Halt или Idle (биты SMBST.MODE читаются как 000_B). Это гарантирует то, что I2C не сформирует сигнал подтверждения при передаче адреса и при дальнейших действиях на шине.

При входе шины в режим Idle или Halt работа I2C запрещается (обнуляется бит SMBCTL2.ENABLE). При этом регистры SMBCTL1, SMBST и SMBCST очищаются, генерация тактовых сигналов приостанавливается, чтобы гарантировать правильное возобновление работы I2C в дальнейшем.

Конфигурация частоты тактового сигнала SCL

Когда I2C работает в режиме Master-устройства шины, то можно программно задавать частоту тактового сигнала на линии SCL.

Режим F/S

Величина, содержащаяся в поле SMBCTL2.SCLFRQ, определяет период тактового сигнала SCL относительно системного тактового сигнала. Также учитывается, что длительность тактовых импульсов на линии SCL определяется тактовым сигналом Master-устройства с наименьшей длительностью импульсов, а длительность паузы между тактовыми импульсами определяется тактовым сигналом Master-устройства с наибольшей паузой между импульсами.

Режим Hs

Величина, содержащаяся в поле SMBCTL3.HSDIV, определяет коэффициент деления тактового сигнала SCL относительно системного тактового сигнала. В соответствии с протоколом шины, синхронизация тактового сигнала не применяется при работе в режиме Master, то есть другое Master-устройство не может уменьшить длительность тактового импульса. Величина в SMBCTL3.HSDIV определяет длительность тактового импульса на SCL, длительность паузы между тактовыми импульсами равна удвоенной длительности импульсов.

Время ожидания на линии SCL

Спецификация SMBus определяет наименьшее время ожидания на линии как TTIMEOUT = 25 мс. Если пауза между двумя тактовыми импульсами превысила TTIMEOUT, то устройство должно прервать текущую передачу. Master должен сформировать состояние «Старт» в процессе передачи или после ее окончания. Slave должен освободить шину. Устройства, обнаружившие данное состояние, должны восстановить свои соединения и ожидать формирования состояния «Старт» в пределах 10 мс.

Функциональное описание счетчика времени ожидания SCL

I2C содержит счетчик времени ожидания для обнаружения и уведомления о простое на шине, см. рисунок 22.19.

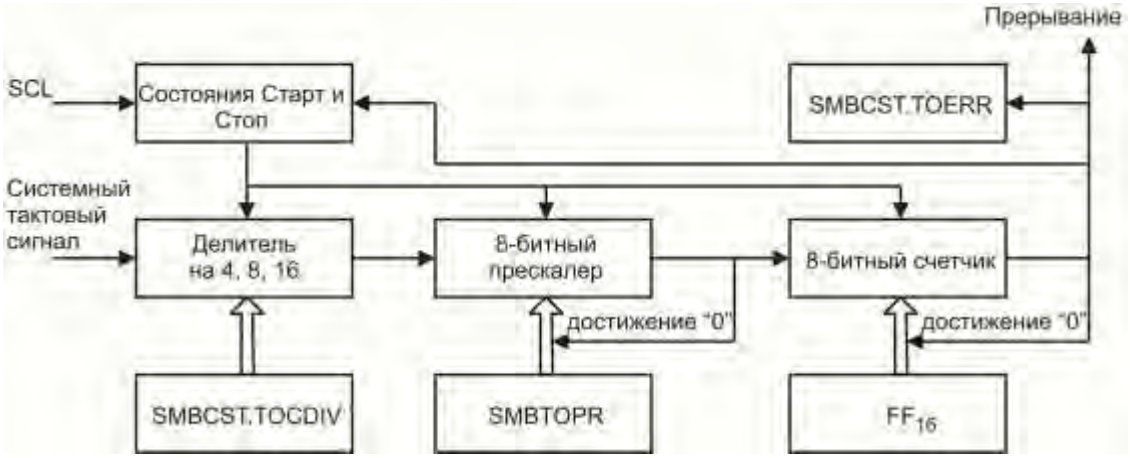


Рисунок 22.19 – Обнаружение времени ожидания

Счетчик времени ожидания включает в себя делитель, 8-битный программируемый прескалер и 8-битный основной счетчик. Все элементы счетчика времени ожидания начинают работу по отрицательному фронту на SCL (если работа счетчика разрешена). Положительный фронт на SCL сбрасывает значения делителя, прескалера и основного счетчика. Прескалер считает «вниз», начиная с величины, записанной в регистр SMBTOPR. После достижения нуля счетчиком прескалера, он перезагружает значение из SMBTOPR. Основной счетчик считает «вниз» от величины FF_H. Каждое достижения нуля счетчиком прескалера декрементирует значение основного счетчика. Переход основного счетчика от нуля к FF_H вызывает остановку основного счетчика, прескалера и делителя, при этом устанавливается флаг SMBCST.TOERR. Дополнительно устанавливается флаг SMBST.INT. Прерывание генерируется, если был установлен бит SMBCTL1.INTEN.

Период времени ожидания определяется следующим выражением:

$$T_{сист} \times Div \times (SMBTOPR + 1) \times 256, \quad (22.1)$$

где $T_{сист}$ – период системного тактового сигнала,

Div – величина, определяемая SMBCST.TOCDIV (4, 8 или 16).

Подключение к порту и конфигурация

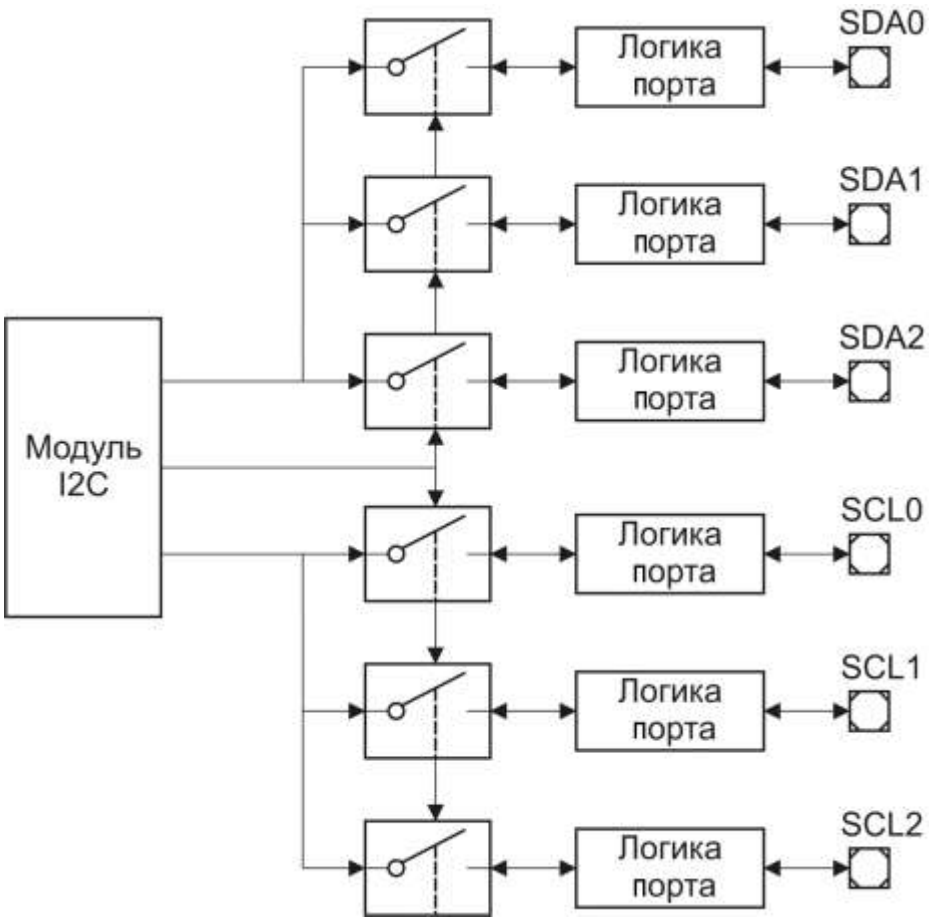


Рисунок 22.20 – Выбор подключения модуля I2C к порту

Модуль I2C обеспечивает подключение трех пар сигналов SCL/SDA к различным выводам схемы. Для управления подключением используется регистр конфигурации CFG. Принцип подключения показан на рисунке 22.20, характеристики I2C – на рисунке 22.21.

Конфигурация выводов

В таблице 22.10 показана установка регистров для конфигурации выводов I2C для работы в режимах Master и Slave.

Таблица 22.10 – Выбор и установка регистров для конфигурации выводов I2C

Линии порта	Регистр управления альтернативными функциями	Регистр управления порта P9	Регистр управления режимом с открытым стоком порта P9
Шина 1			
P9.0/ SDA0	ALTSEL0P9.0 = 1 и ALTSEL1P9.0 = X	DP9.0 = 1	ODP9.0 = 1
P9.1/ SCL0	ALTSEL0P9.1 = 1 и ALTSEL1P9.1 = 0	DP9.1 = 1	ODP9.1 = 1
Шина 2			
P9.2/ SDA1	ALTSEL0P9.2 = 1 и ALTSEL1P9.2 = 0	DP9.2 = 1	ODP9.2 = 1
P9.3/ SCL1	ALTSEL0P9.3 = 1 и ALTSEL1P9.3 = 0	DP9.3 = 1	ODP9.3 = 1
Шина 3			
P9.4/ SDA2	ALTSEL0P9.4 = 1 и ALTSEL1P9.4 = X	DP9.4 = 1	ODP9.4 = 1
P9.5/ SCL2	ALTSEL0P9.5 = 1 и ALTSEL1P9.5 = X	DP9.5 = 1	ODP9.5 = 1

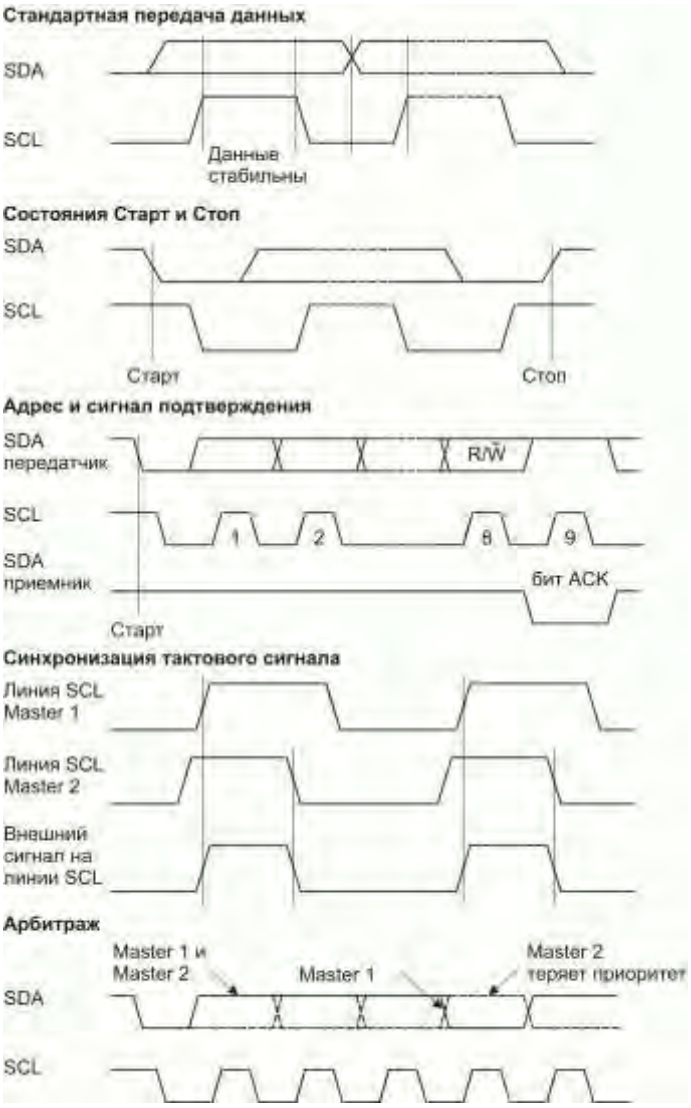


Рисунок 22.21 – Характеристики I2C

22.3 Описание регистров

Регистр SMBSDA, см. рисунок 22.22, таблицу 22.11 является 8-битным сдвиговым регистром, используемым для передачи и приема данных. Наиболее значимый бит MSB передается (принимается) первым, а наименее значимый бит LSB передается (принимается) последним.

SMBSDA
сдвиговый регистр данных I2C

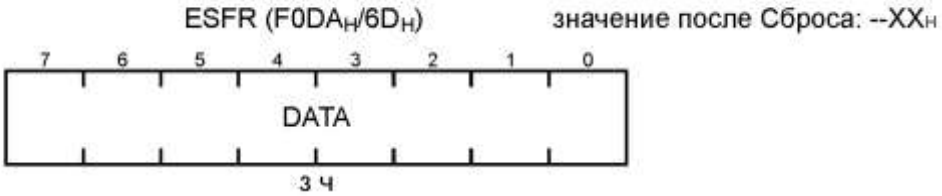


Рисунок 22.22 – Формат регистра SMBSDA

Таблица 22.11 – Функциональное назначение полей регистра SMBSDA

Поле	Биты	Тип	Описание
DATA	7-0	Запись Чтение	Запись в SMBSDA возможна только в случае, когда установлен флаг SMBST.INT. Попытки записи регистра в других случаях будут проигнорированы. Регистр SMBSDA не очищается после сброса, а содержит случайные данные до того, как будет записан программно или до приема и сдвига данных

Регистр SMBST, см. рисунок 22.23, таблицу 22.12, является 8-битным регистром, содержащим текущее значение статуса I2C. Регистр только для чтения. После сброса и во время запрещения работы I2C, SMBST принимает значение 00_H, режим работы I2C представлен в в таблице 22.13.

SMBST
регистр статуса I2C

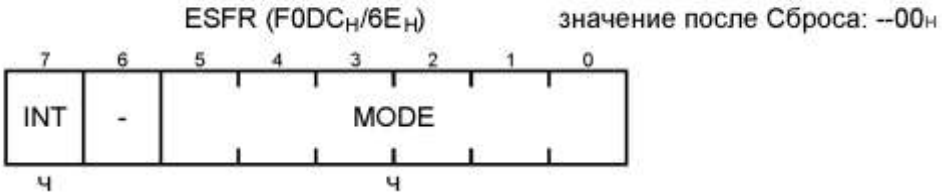


Рисунок 22.23 – Формат регистра SMBST

Таблица 22.12 – Функциональное назначение полей регистра SMBST

Поле	Биты	Тип	Описание
INT	7	Чтение	Флаг прерывания: 0 Устанавливается после записи «1» в бит SMBCTL1.CRST или при запрещении работы I2C при обнулении бита SMBCTL2.ENABLE. 1 Устанавливается после девятого тактового импульса на SCL (SCL – в низком уровне), в любое время разрешено программное вмешательство.
MODE	5-0	Чтение	Текущий статус и режим работы I2C. Расшифровка кода статуса шины приведена в таблице 22.10.

Таблица 22.13 – Режимы работы модуля I2C

Режим	Код	Мнемонический код	Описание
1	2	3	4
Общий	00_H	IDLE	Idle, не доступна достоверная информация о статусе
F/S «Master»	01_H	STDONE	Сформировано состояние «Старт»
	02_H	RSDONE	Сформировано состояние «Повторный старт»
	03_H	IDLARL	Потеря приоритета, вход в режим «Неадресованный Slave»
F/S «Master передатчик»	04_H	MTADPA	Отправлен адрес Slave, сигнал ACK
	05_H	MTADNA	Отправлен адрес Slave, сигнал NACK
	06_H	MTDAPA	Отправлен байт данных, сигнал ACK
	07_H	MTDANA	Отправлен байт данных, сигнал NACK
F/S «Master приемник»	08_H	MRADPA	Отправлен адрес Slave, сигнал ACK
	09_H	MRADNA	Отправлен адрес Slave, сигнал NACK
	$0A_H$	MRDAPA	Принят байт данных, сигнал ACK
	$0B_H$	MRDANA	Принят байт данных, сигнал NACK
F/S «Master»	$0C_H$	MTMCER	Адрес Master передан с ошибкой (сигнал ACK)
–	$0D_H – 0F_H$	–	Не используются
F/S «Slave приемник»	10_H	SRADPA	Принят адрес Slave, сигнал ACK
	11_H	SRAAPA	Принят адрес Slave после потери приоритета, сигнал ACK
	12_H	SRDAPA	Принят байт данных, сигнал ACK
	13_H	SRDANA	Принят байт данных, сигнал NACK
F/S «Slave передатчик»	14_H	STADPA	Принят адрес Slave, сигнал ACK
	15_H	STAAPA	Принят адрес Slave после потери приоритета, сигнал ACK
	16_H	STDAPA	Отправлен байт данных, сигнал ACK
	17_H	STDANA	Отправлен байт данных, сигнал NACK
F/S «Slave передатчик» отклик	18_H	SATADP	Принят адрес отклика, сигнал ACK
	19_H	SATAAP	Принят адрес отклика после потери приоритета, сигнал ACK
	$1A_H$	SATDAP	Отправлены данные отклика, сигнал ACK
	$1B_H$	SATDAN	Отправлены данные отклика, сигнал NACK
F/S «Slave»	$1C_H$	SSTOP	В режиме Slave зафиксировано состояние «Стоп»
	$1D_H$	SGADPA	Принят адрес общего вызова, сигнал ACK
	$1E_H$	SDAAPA	Принят адрес общего вызова после потери приоритета, сигнал ACK
Общий	$1F_H$	BERROR	Ошибка на шине (некорректное формирование состояний «Старт» или «Стоп»)
Hs «Master»	20_H	–	Не используется
	21_H	HMTMCOK	Адрес Master передан – переход в режим Hs
	22_H	HRSDONE	Формирование состояния «Повторный старт»
	23_H	HIDLARL	Потеря приоритета, переход в режим Hs «Неадресованный Slave»
Hs «Master передатчик»	24_H	HMTADPA	Отправлен адрес Slave, сигнал ACK
	25_H	HMTADNA	Отправлен адрес Slave, сигнал NACK
	26_H	HMTDAPA	Отправлен байт данных, сигнал ACK
	27_H	HMTDANA	Отправлен байт данных, сигнал NACK

Окончание таблицы 22.13

1	2	3	4
Hs «Master приемник»	28_H	HMRADPA	Отправлен адрес Slave, сигнал ACK
	29_H	HMRADNA	Отправлен адрес Slave, сигнал NACK
	$2A_H$	HMRDAPA	Принят байт данных, сигнал ACK
	$2B_H$	HMRDANA	Принят байт данных, сигнал NACK
–	$2C_H – 2F_H$	–	Не используются
Hs «Slave приемник»	30_H	HSRADPA	Принят адрес Slave, сигнал ACK
	31_H	–	Не используется
	32_H	HSRDAPA	Принят байт данных, сигнал ACK
	33_H	HSRDANA	Принят байт данных, сигнал NACK
Hs «Slave приемник»	34_H	HSTADPA	Принят адрес Slave, сигнал ACK
	35_H	–	Не используется
	36_H	HSTDAPA	Отправлен байт данных, сигнал ACK
	37_H	HSTDANA	Отправлен байт данных, сигнал NACK
–	$38_H – 3F_H$	–	Не используется

Условия выставления флага прерывания:

- при работе в режимах «Master передатчик», «Master приемник», «Slave приемник» или «Slave передатчик»;
- после обнаружения совпадения адреса (совпадение адреса Slave, адреса отклика, адреса общего вызова); необходимо программно проверять содержимое регистра SMBSDA для определения, какое совпадение произошло;
- после успешного формирования состояний «Старт» или «Повторный старт»;
- если был получен инверсного сигнала подтверждения NACK;
- при успешном формировании состояний «Стоп» или «Повторный старт»;
- после обнаружения ошибки на шине.

Линия SCL удерживается в низком уровне, пока установлен флаг прерывания INT регистра SMBST. Следующие условия устанавливают флаг прерывания, но не приводят к удерживанию SCL в низком уровне (ожидание на линии SCL):

- состояние «Стоп» в режиме Slave поле MODE = SSTOP ($1C_H$);
- потеря приоритета приводит к входу модуля I2C в режим «Неадресованный Slave» поле MODE = IDLARL (03_H) или MODE = HIDLARL (23_H);
- передан байт данных, инверсный сигнал подтверждения NACK (17_H).

Регистр SMBCST, см. рисунок 22.24, таблицу 22.14, является 8-битным регистром для чтения/записи, содержащим значение статуса модуля I2C. После сброса и во время запрещения работы I2C, все биты регистра становятся равными нулю.

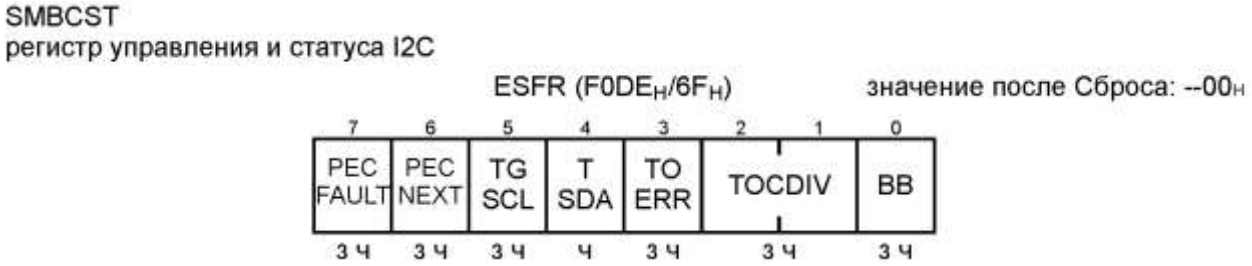


Рисунок 22.24 – Формат регистра SMBCST

Таблица 22.14 – Функциональное назначение полей регистра SMBCST

Поле	Биты	Тип	Описание
PECFAULT	7	Запись Чтение	Ошибка пакета данных: 1 При последней передаче произошла ошибка пакета данных (значение регистра Packet Error Checking Register не равно нулю).
PECNEXT	6	Запись Чтение	Следующий байт PEC: 1 Следующий байт на шине будет байтом PEC
TGSCL	5	Запись Чтение	Удерживание SCL в неактивном уровне. Разрешает удерживание SCL в неактивном уровне на период исправления ошибки. 1 Удерживает SCL в неактивном уровне на один цикл. Попытка записи в этот бит при высоком уровне сигнала на SDA игнорируется. Бит очищается при завершении удерживания SCL.
TSDA	4	Чтение	Тест SDA. Содержит текущее значение SDA. Этот бит может быть использован при исправлении ошибки на шине. Этот бит только для чтения, попытка записи в него игнорируется.
TOERR	3	Запись Чтение	Ошибка ожидания на шине3. 0 Бит очищается при записи «1» в бит SMBCTL1.CLRST. 1 Зафиксировано состояние ожидания на линии SCL. Бит устанавливается при достижении нуля основным счетчиком времени ожидания.
TOCDIV	2-1	Запись Чтение	Эти биты определяют коэффициент деления для прескалера времени ожидания SCL. 00 Запрещение работы, нет тактового сигнала. 01 Деление на 4. 10 Деление на 8. 11 Деление на 16.
BB	0	Запись Чтение	Шина занята: 0 Бит обнуляется при обнаружении состояния «Стоп» или при запрещении работы I2C. 1 Устанавливается, когда шина активна (линии SDA и SCL в низком уровне) или про формировании сигнала «Старт». Показывает, что шина занята.

Регистр SMBCTL1, см. рисунок 22.25, таблицу 22.15, является 8-битным регистром конфигурации и управления I2C. После сброса и при запрещении работы модуля I2C SMBCTL2.ENABLE = 0_B регистр принимает значение 00_H.

SMBCTL1
регистр управления 1 I2C

ESFR ($F0E0_H/70_H$) значение после Сброса: $--00_H$

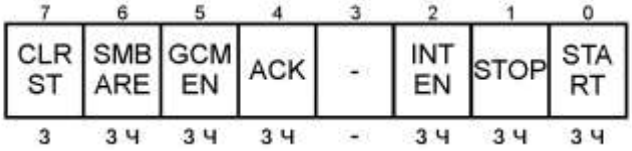


Рисунок 22.25 – Формат регистра SMBCTL1

Таблица 22.15 – Функциональное назначение полей регистра SMBCTL1

Поле	Биты	Тип	Описание
CLRST	7	Запись	Очистка статуса 1 Очищает бит статусный бит SMBST.INT. Запись «0» в этот бит не имеет эффекта, этот бит всегда читается как «0»
SMBARE	6	Запись Чтение	Разрешение распознавания адреса отклика 0 Модул I2C не отвечает на адрес отклика на шине. Бит очищается при выходе модуля I2C из режимов Halt или Idle 1 Разрешение распознавания среди принятых адресов адреса отклика 0001100_B в режиме Slave
GCMEN	5	Запись Чтение	Разрешение распознавания адреса общего вызова 0 I2C не распознает адрес общего вызова. Бит очищается при выходе модуля I2C из режимов Halt или Idle 1 Разрешение распознавания среди принятых адресов адреса общего вызова 00_H в режиме Slave
ACK	4	Запись Чтение	Бит подтверждения. ACK игнорируется в режиме передатчика. 1 Сообщает передающему устройству о необходимости остановки передачи данных.
INTEN	2	Запись Чтение	Разрешение прерывания: 0 Прерывание запрещено. 1 Прерывание разрешено.
STOP	1	Запись Чтение	«Стоп». 1 При работе в режиме Master – формирование состояния «Стоп», что завершит или прервет текущую передачу данных. Бит очищается после завершения формирования состояния «Стоп».

Окончание таблицы 22.15

1	2	3	4
START	0	Запись Чтение	«Старт»: 1 Устанавливается при необходимости формирования состояния «Старт» на шине. Бит очищается при завершении формирования состояния «Старт» или после обнаружения ошибки на шине SMBST.MODE = $1F_H$.

Формат регистра SMBADDR представлен на рисунке 22.26, функциональное назначение полей регистра SMBADDR – в таблице 22.16.

SMBADDR
регистр собственного адреса I2C

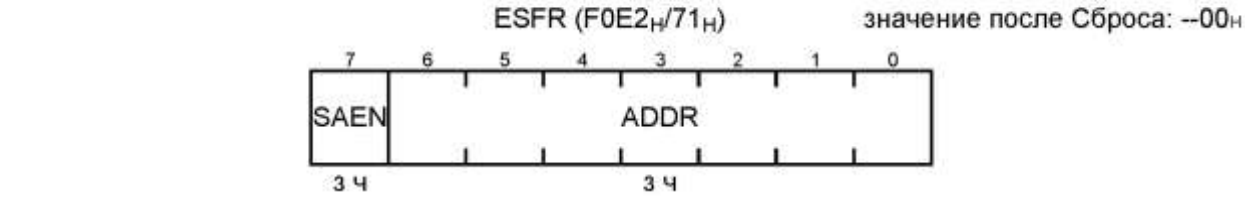


Рисунок 22.26 – Формат регистра SMBADDR

Таблица 22.16 – Функциональное назначение полей регистра SMBADDR

Поле	Биты	Тип	Описание
SAEN	7	Запись Чтение	Разрешение адреса Slave 1 Показывает, что поле ADDR содержит значение адреса, разрешает сравнение ADDR и принятого байта адреса
ADDR	6-0	Запись Чтение	Собственный адрес. Содержит 7-битный адрес I2C. При работе I2C в режиме Slave первые 7 бит, принятые после состояния «Старт», сравниваются с этим полем. Если значение ADDR совпадает с принятым адресом, то SMBST.MODE принимает значение 001_B.

Формат регистра SMBCTL2 представлен на рисунке 22.27, функциональное назначение полей регистра SMBADDR – в таблице 22.17.

SMBCTL2
регистр управления 2 I2C

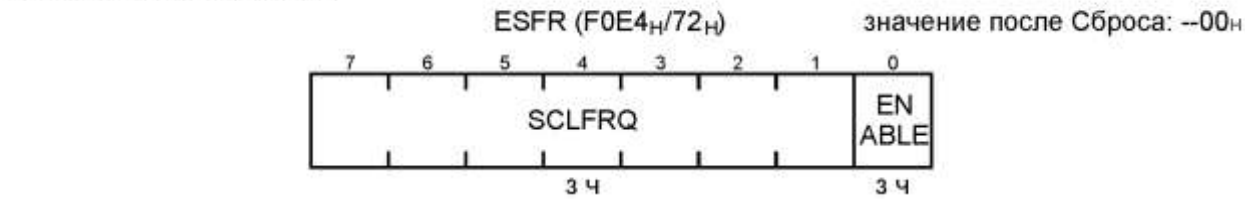


Рисунок 22.27 – Формат регистра SMBCTL2

Таблица 22.17 – Функциональное назначение полей регистра SMBCTL2

Поле	Биты	Тип	Описание
SCLFRQ	7-1	Запись Чтение	Частота SCL. Это поле определяет значение периода SCL при работе I2C в режиме Master. В поле SCLFRQ может быть записано значение от 4 до 127. При выборе значения меньше 4, оно автоматически дополняется до 4.
ENABLE	0	Запись Чтение	Разрешение работы: 0 Запрещение работы I2C. При очищении бита ENABLE регистры SMBCTL1, SMBST, SMBCST обнуляются, прекращается генерация тактового сигнала. 1 Разрешение работы I2C.

SMBTOPR
регистр прескалера времени ожидания SCL

ESFR ($F0E6_H/73_H$) значение после Сброса: --00_H

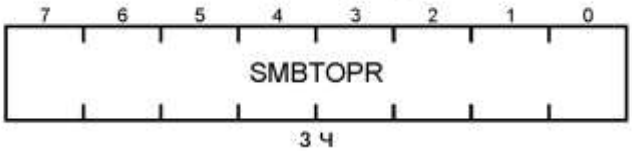


Рисунок 22.28 – Формат регистра SMBTOPR

Таблица 22.18 – Функциональное назначение полей регистра SMBTOPR

Поле	Биты	Тип	Описание
SMBTOPR	7-0	Запись Чтение	Прескалер времени ожидания SCL Содержит значение прескалера времени ожидания. После сброса принимает значение 00_H.

Регистр SMBCTL3 является 8-битным регистром для чтения/записи для управления 10-битной адресацией, делителем тактового сигнала в режиме Hs. После сброса регистр SMBCTL3 принимает значение 00_H.

SMBCTL3
регистр управления 3 I2C

ESFR ($F0E8_H/74_H$) значение после Сброса: --00_H

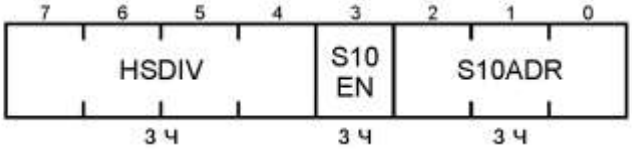


Рисунок 22.29 – Формат регистра SMBCTL3

Таблица 22.19 – Функциональное назначение полей регистра SMBCTL3

Поле	Биты	Тип	Описание
HSDIV	7-4	Запись Чтение	Делитель тактового сигнала SCL в режиме Hs. Определяет значение периода SCL при работе I2C в высокоскоростном режиме. В поле HSDIV может быть записано значение от 2 до 16. При записи числа, меньшего 2, к начальному значению по умолчанию добавляется 2. То есть, при выборе значения делителя, равного 1, результирующим будет значение, равное 3.
S10EN	3	Запись Чтение	Разрешение 10-битной адресации: 1 Разрешает распознавание 10-битного адреса Slave. Для распознавания 10-битного адреса требуется одновременная установка битов S10EN и SMBADDR.SAEN.
S10ADR	2-0	Запись Чтение	10-битный адрес Slave. Содержит старшие биты 10-битного адреса Slave. Старшие 5 бит первого принятого адреса сравниваются с величиной 1110_B, младшие 3 бита – с величиной S10ADR.2 – S10ADR.1. Старший бит второго принятого адреса сравнивается с величиной S10ADR.0, младшие 7 бит – со значением поля SMBADDR.SADR.

CFG
регистр конфигурации I2C

ESFR ($F0EA_H/75_H$) значение после Сброса: $--00_H$

7	6	5	4	3	2	1	0
-	SCL EN2	SCL EN1	SCL EN0	-	SDA EN2	SDA EN1	SDA EN0
-	З Ч	З Ч	З Ч	-	З Ч	З Ч	З Ч

Рисунок 22.30 – Формат регистра CFG

Таблица 22.20 – Функциональное назначение полей регистра CFG

Поле	Биты	Тип	Описание
SCLENx, x = 2, 1, 0	6-4	Запись Чтение	Бит разрешения для линии тактового сигнала SCLx. Этот бит определяет, к каким выходам модуля I2C будет подключена линия тактового сигнала. 0 Выход SCLx не подключен. 1 Выход SCLx соединен с линией тактового сигнала модуля I2C.
SDAENx, x = 2, 1, 0	2-0	Запись Чтение	Бит разрешения для линии данных SDAx. Этот бит определяет, к каким выходам модуля I2C будет подключена линия данных. 0 Выход SDAx не подключен. 1 Выход SCLx соединен с линией данных модуля I2C.

23 Модуль последовательного интерфейса связи CAN

Ведение в протокол CAN

Controller Area Network (CAN) – последовательный интерфейс связи (далее – модуль CAN) эффективно поддерживающий распределенное управление в реальном масштабе времени с высокой помехозащищенностью, см. рисунок 23.1. Протокол связи полностью определен Robert Bosch GmbH в спецификациях CAN 1.2, CAN 2.0A, CAN 2.0B (активная и пассивная версии).

Протокол CAN оптимизирован для систем, в которых должно передаваться относительно небольшое количество информации (по сравнению с Ethernet или USB) к любому или всем узлам сети. Множественный доступ с опросом состояния шины позволяет каждому узлу получить доступ к шине с учетом приоритетов. Неадресатная структура сообщений позволяет организовать многоабонентскую доставку данных с сокращением трафика шины. Быстрая устойчивая передача информации, с системой контроля ошибок, позволяет отключать неисправные узлы от шины, что гарантирует доставку критических по времени сообщений.

Область применения CAN протокола: от высокоскоростных сетей связи до электропроводов в автомобиле. Высокая скорость передачи данных до 1 Мбит/с, высокая помехозащищенность протокола, защита от неисправности узлов делают шину CAN подходящей для индустриальных приложений управления типа DeviceNet.

CAN имеет асинхронную последовательную структуру шины с одним логическим сегментом сети. CAN сеть может состоять из двух или более узлов, с возможностью подключения/отключения узлов от шины без перенастройки других устройств.

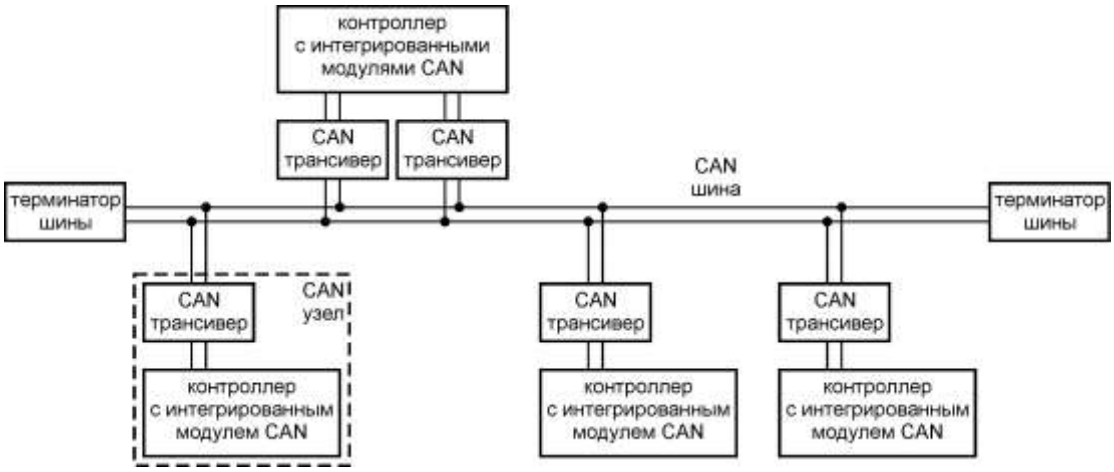


Рисунок 23.1 – Общая структура CAN сети

Логика шины работает по механизму «монтажное И», в котором рецессивный (recessive) бит соответствует логической единице «1», а доминантный (dominant) – логическому нулю «0». Пока ни один узел не формирует доминантный бит, шина находится в рецессивном состоянии. Появление на шине доминантного бита (выставленного одним или несколькими узлами) создает доминантное состояние шины. Отсюда следует, что при выборе среды передачи данных необходимо точно определить, какое состояние будет доминантным, а какое – рецессивным. Одним из наиболее распространенных и «дешевых» вариантов является пара скрученных проводов. Линии шины тогда называются CANH и CANL и могут быть подключены непосредственно к устройствам. Не существует никакого дополнительного стандарта на среду передачи данных.

При использовании в качестве линии связи пары скрученных проводов с нагрузочными резисторами на концах можно получить максимальную скорость передачи

данных 1 Мбит/с при длине линии до 40 м. Для линий связи протяженностью более 40 м необходимо снизить скорость передачи данных (для линии в 1 000 м скорость шины должна быть не более 40 Кбит/с). Из-за дифференциального характера линии связи шина CAN мало чувствительна к электромагнитным помехам. Экранирование шины значительно снизит воздействие внешнего электромагнитного поля, что особенно важно для высокоскоростных режимов работы.

Двоичная информация кодируется, при этом доминантным является низкий уровень, рецессивным – высокий уровень. Для гарантированной синхронизации данных всеми узлами шины используется наполняющий бит. При последовательной передаче пяти бит одинаковой полярности передатчик вставляет один дополнительный бит противоположной полярности перед передачей остальных битов. Приемник также проверяет полярность и удаляет дополнительные биты.

В CAN протоколе при передаче данных приемные узлы не адресуются, а указывается идентификатор передатчика. С помощью идентификатора указывается содержание сообщения (например, применительно автомобиля – обороты, температура двигателя и т. д.). Идентификатор дополнительно указывает приоритет сообщения. Более высокий приоритет у идентификатора, имеющего меньшее бинарное значение.

При коллективном доступе к шине используется неразрушающий арбитраж с опросом состояния шины. Перед началом передачи данных узел проверяет состояние шины (отсутствие активности на шине). При начале передачи сообщения узел становится управляющим шины, все остальные узлы переходят в режим приема. Каждый узел выдает подтверждение приема, проверяет идентификатор сообщения, обрабатывает или удаляет принятые данные. Если два или более узлов начинают передачу данных одновременно, поразрядный арбитраж позволяет избежать конфликта на шине. Каждый узел выдает на шину свой идентификатор (старший бит формируется первым) и контролирует ее состояние. Если узел посылает «1», а читает «0», то арбитраж потерян и узел переключается в режим приема. Это происходит тогда, когда идентификатор конкурирующего узла имеет меньшее бинарное значение. Таким образом, узел с высоким приоритетом выигрывает арбитраж, без необходимости повторять сообщение. Все остальные узлы будут пытаться передать сообщение после освобождения шины. Данный механизм не позволяет передавать одновременно сообщения с одинаковым идентификатором, избегая, таким образом, дальнейших ошибок.

Оригинальная спецификация в версии CAN 2.0B (так называемая расширенная версия CAN) определяет возможность идентификатора иметь длину 11 или 29 бит.

Функциональные возможности модуля CAN

Модуль CAN выполняет все функции для приема и передачи сообщений на шине. Данные, предназначенные для передачи, записываются в соответствующие регистры. Состояние модуля CAN и возникшие ошибки проверяются чтением регистров состояния. Любое сообщение, передаваемое по шине, проверяется на наличие ошибок, согласуется по фильтрам и сохраняется в регистрах приемного буфера при выполнении всех условий.

CAN модуль поддерживает следующие типы сообщений:

- сообщения данных;
- удаленный запрос данных;
- сообщение об ошибке;
- сообщение о перезагрузке.

Существует два типа сообщений данных:

- стандартное сообщение;
- расширенное сообщение.

Стандартное сообщение данных

Стандартное сообщение данных формируется, когда узел желает передать данные, см. рисунок 23.2.

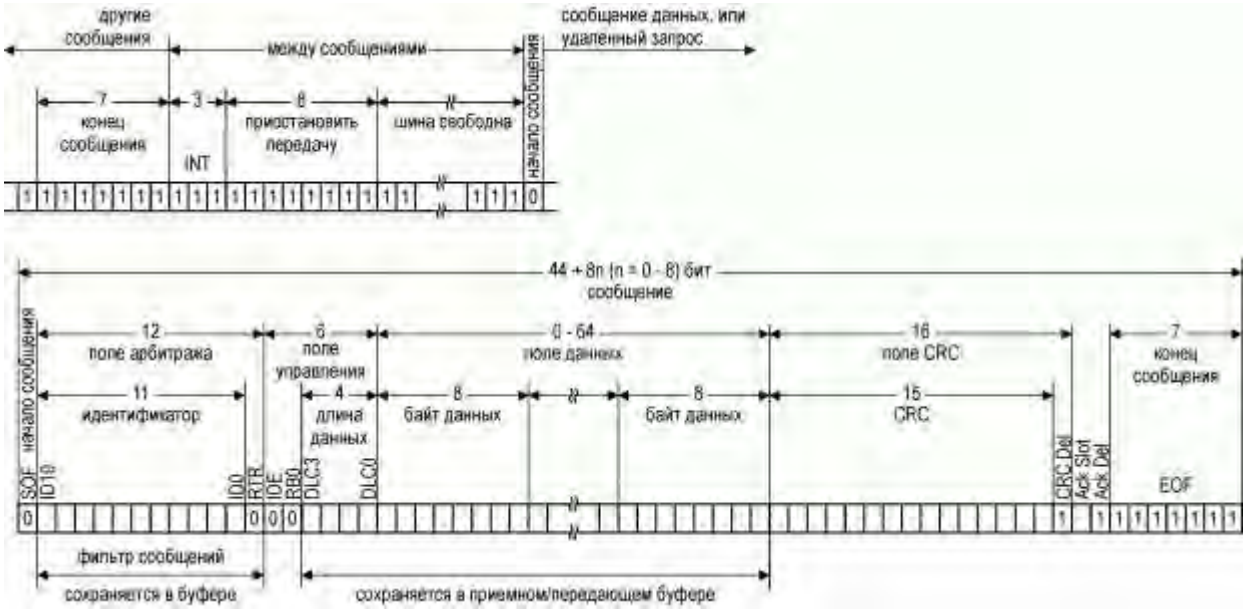


Рисунок 23.2 – Стандартное сообщение данных

Стандартное сообщение данных имеет в своем составе:

- бит SOF – доминантный «0» бит начала сообщения для жесткой синхронизации всех узлов;
- поле арбитража 12 бит, включающее:
  - поле ID идентификатора 11 бит;
  - бит RTR передачи по удаленному запросу ($RTR = 0_B$ соответствует сообщению данных, $RTR = 1_B$ соответствует удаленному запросу);
- поле управления 6 бит, включающее:
  - бит IDE указатель расширенного идентификатора ($IDE = 0_B$ соответствует стандартному идентификатору, $IDE = 1_B$ соответствует расширенному идентификатору);
  - бит RB0 резервный доминантный бит;
  - поле DLC числа байт данных 4 бита, которое указывает, сколько байт данных содержится в сообщении (допустимые значения – от 0 до 8, другие значения использоваться не могут);
- поле данных от 0 до 64 бит, содержащее целое число байт данных;
- поле контрольной суммы CRC 16 бит, включающее:
  - поле CRC 15 бит, используемое для обнаружения возможных ошибок передачи данных;
  - бит CRC Del рецессивный разделитель CRC;
- поле подтверждения 2 бита, включающее:
  - бит ACK Slot подтверждения передачи, передающий узел выдает рецессивный бит, а любой узел, который принял сообщение без ошибок, заменяет его сформированным доминантным битом;
  - бит ACK Del рецессивный разделитель подтверждения;
- поле EOF конца сообщения 7 бит.

Между передачами двух любых сообщений шина должна оставаться в рецессивном состоянии, как минимум, в течение времени появления 3 битов (поле INT простоя). Если после появления трех рецессивных битов (поля INT) ни один узел не начал передачу, шина переходит в состояние бездействия (режим Idle) и находится в рецессивном состоянии до появления доминантного бита сообщения.

Расширенное сообщение данных

Расширенное сообщение данных формируется, когда узел желает передать данные, см. рисунок 23.3.

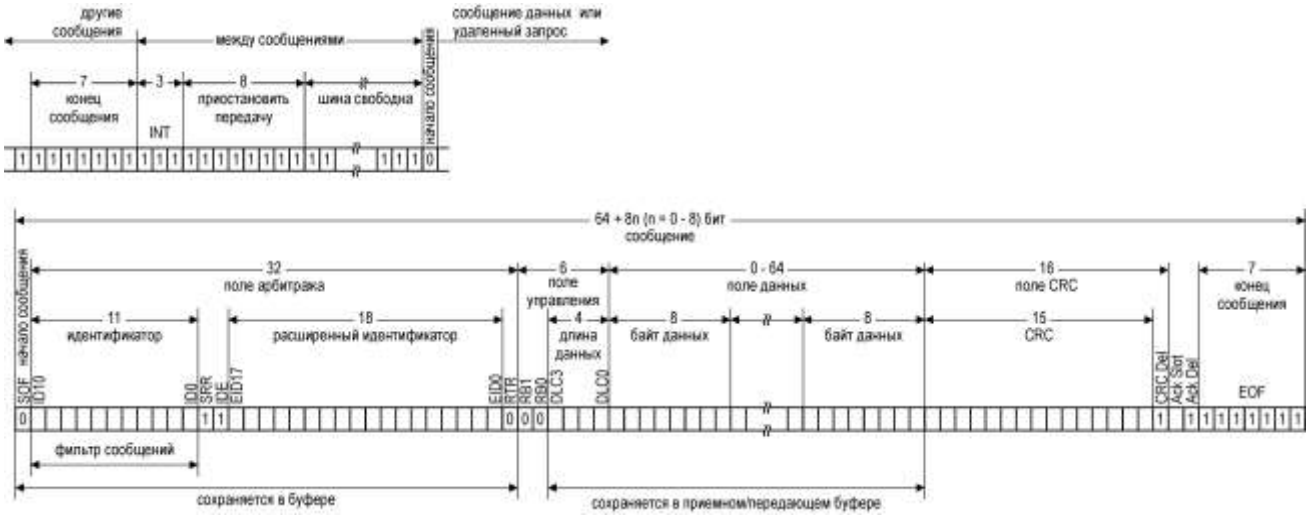


Рисунок 23.3 – Расширенное сообщение данных

Расширенное сообщение данных имеет в своем составе:

- бит SOF – доминантный «0» бит начала сообщения для жесткой синхронизации всех узлов;
- поле арбитража 38 бит, включающее:
  - поле стандартного идентификатора 11 бит;
  - бит SRR заменитель удаленного запроса;
- бит IDE указатель расширенного идентификатора (рецессивный, что соответствует расширенному идентификатору);
  - поле расширенного идентификатора 18 бит;
- бит RTR передачи по удаленному запросу (RTR = 0_B соответствует сообщению данных, RTR = 1_B соответствует удаленному запросу);
- поле управления 6 бит, включающее:
  - бит RB0 резервный доминантный бит;
  - бит RB1 резервный доминантный бит;
- поле DLC числа байт данных 4 бита, которое указывает, сколько байт данных содержится в сообщении (допустимые значения – от 0 до 8, другие значения использоваться не могут);
- поле данных от 0 до 64 бит, содержащее целое число байт данных;
- поле контрольной суммы CRC 16 бит, включающее:
  - поле CRC 15 бит используемое для обнаружения возможных ошибок передачи данных;
  - бит CRC Del рецессивный разделитель CRC;
- поле подтверждения 2 бита, включающее:
  - бит ACK Slot подтверждения передачи (передающий узел выдает рецессивный бит, а любой узел, который принял сообщение без ошибок, заменяет его сформированным доминантным битом);
  - бит ACK Del рецессивный разделитель подтверждения;
- поле EOF конца сообщения 7 бит.

Удаленный запрос данных

Удаленный запрос данных формируется, когда узлу требуются данные другого узла. Узел назначения посылает удаленный запрос с идентификатором источника. Соответствующий узел источника, распознавший свой идентификатор, посылает стандартное или расширенное сообщение в ответ на запрос.

Удаленный запрос данных существует в стандартном и расширенном вариантах. На рисунке 23.4 представлен вариант удаленного запроса со стандартным идентификатором. Имеются только два отличия содержимого удаленного запроса от сообщения данных:

- бит RTR в удаленном запросе передается в рецессивном состоянии;
- поле данных отсутствует (в сообщении не передается никаких данных, значение в поле DLC любое в пределах от 0 до 8).

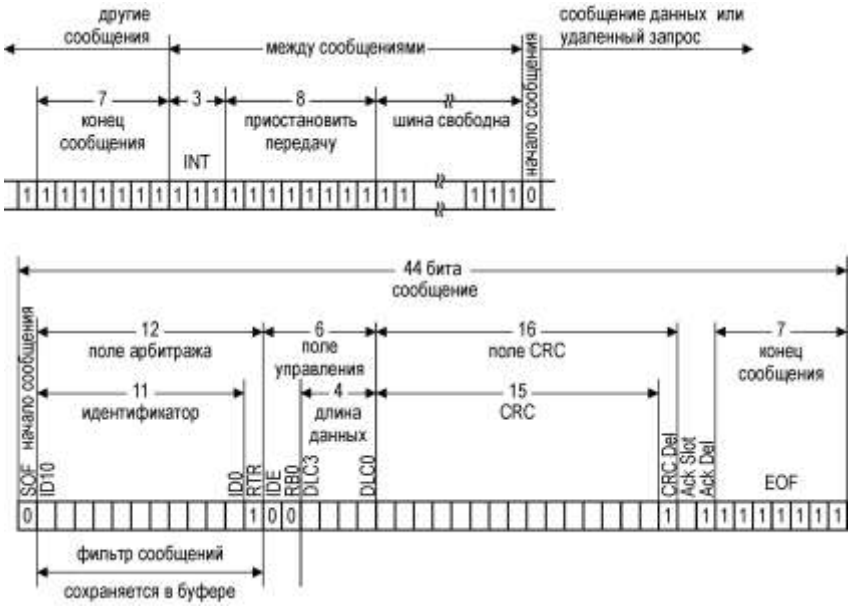


Рисунок 23.4 – Удаленный запрос данных (стандартный формат)

В самом маловероятном случае, когда одновременно формируется удаленный запрос и устройство пытается передать данные с одинаковыми идентификаторами, арбитраж будет выигран устройством, передающим данные, из-за доминантного состояния бита RTR.

Узел, который посылал запрос, получает данные немедленно.

Сообщение об ошибке

Формируется любым узлом, который обнаруживает ошибку на шине.

Сообщение об ошибке может формироваться как активным, так и пассивным узлом.

Если ошибку обнаружил активный узел, см. рисунок 23.5, тогда он прерывает передачу текущего сообщения, формируя флаг активной ошибки. Флаг активной ошибки состоит из 6 последовательных доминантных битов, которые нарушают правила бит-стаффинга (правила наполнения и передачи битов на шине). Остальные узлы также обнаруживают ошибку и начинают формировать сообщение об ошибке. Таким образом, поле флага ошибки может содержать от 6 до 12 доминантных битов. Ошибка сопровождается разделителем ошибки, состоящим из восьми рецессивных битов и позволяющим перезапускать связь с шиной.

После перехода шины в нормальное состояние узлы возобновляют передачу данных.

Если ошибку обнаружил пассивный узел, тогда он формирует флаг пассивной ошибки, состоящий из шести последовательных рецессивных битов, и затем разделитель ошибки. Таким образом, сообщение о пассивной ошибке состоит из 14 рецессивных битов. Это не нарушает правила бит-стаффинга на шине и не оказывает влияния на передачи других узлов. Исключение составляет узел, который передает данные узлу, обнаружившему ошибку. В этом случае правила бит-стаффинга нарушаются, и передача данных прекращается. После передачи пассивной ошибки, узел должен ожидать шесть последовательных рецессивных битов для восстановления связи с шиной.

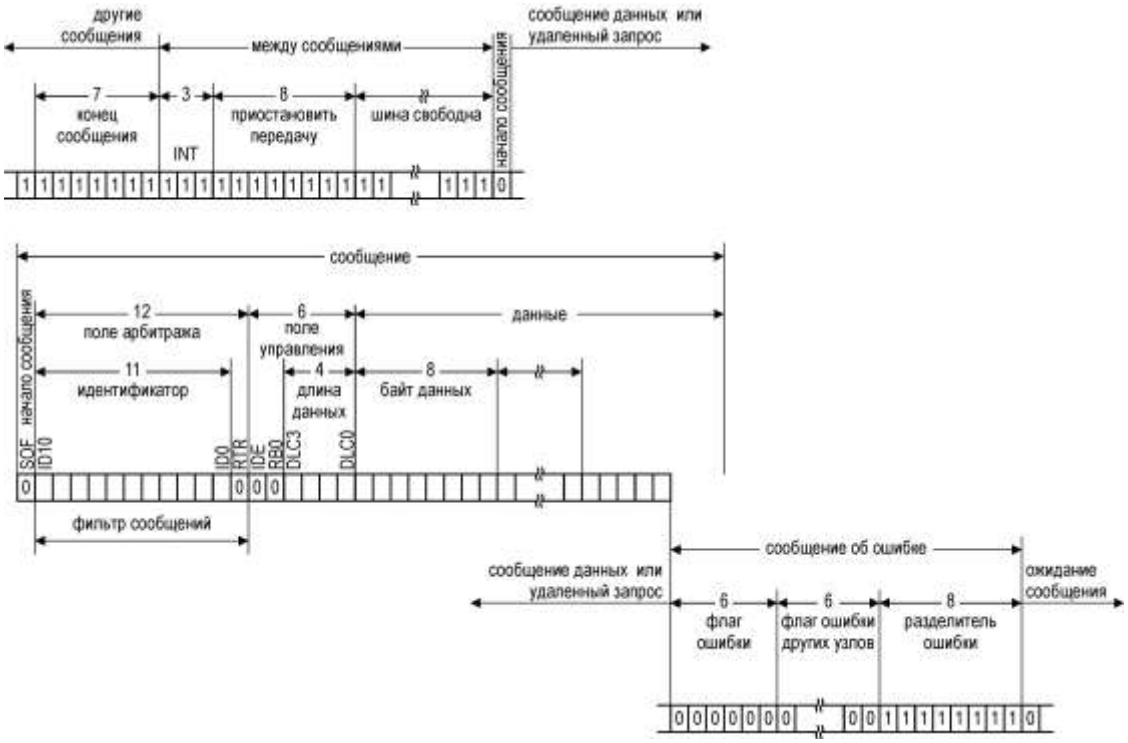


Рисунок 23.5 – Сообщение об активной ошибке

Сообщение о перезагрузке

Формат сообщения о перезагрузке, см. рисунок 23.6, аналогичен формату сообщения об ошибке, но может быть сформирован только, когда шина простаивает.

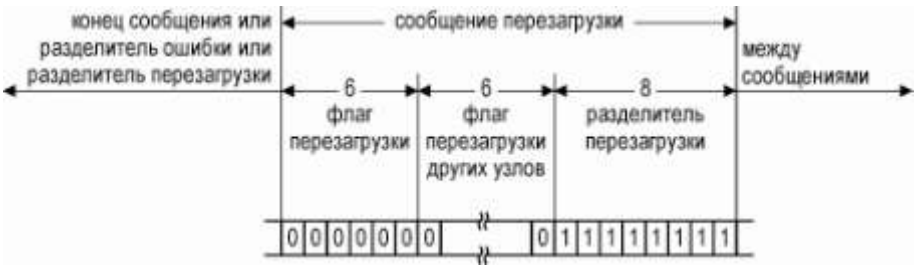


Рисунок 23.6 – Сообщение о перезагрузке

Флаг перезагрузки состоит из 6 последовательных доминантных битов. Другие узлы обнаруживают перезагрузку и начинают формировать ее самостоятельно. Поэтому на шине, во время выполнения перезагрузки, может быть до12 доминантных битов.

Разделитель перезагрузки состоит из 8 последовательных рецессивных битов.

Узел может сформировать сообщение о перезагрузке в двух случаях:

- между сообщениями обнаружен доминантный бит, что является ненормальным во время простоя шины;
- для задержки передачи нового сообщения.

Узел может последовательно сформировать не более двух сообщений перезагрузки.

Простой шины

Между сообщениями шина находится в рецессивном состоянии. Для выполнения условий «простой шины» необходимо, чтобы было получено, как минимум, 3 рецессивных бита после завершения передачи/приема очередного сообщения.

23.1 ISO/OSI модель

ISO/OSI эталонная модель, см. рисунок 23.7, используется для того, чтобы определить уровни протокола системы связи. На самом верхнем уровне модели находятся приложения, которые должны связываться друг с другом. На самом низком уровне модели расположена физическая среда, обеспечивающая электрическую сигнализацию связи.

Верхние уровни сетевой модели CAN реализуются программно. В технических требованиях к CAN протоколу нет никаких дополнительных требований к содержанию сообщений.

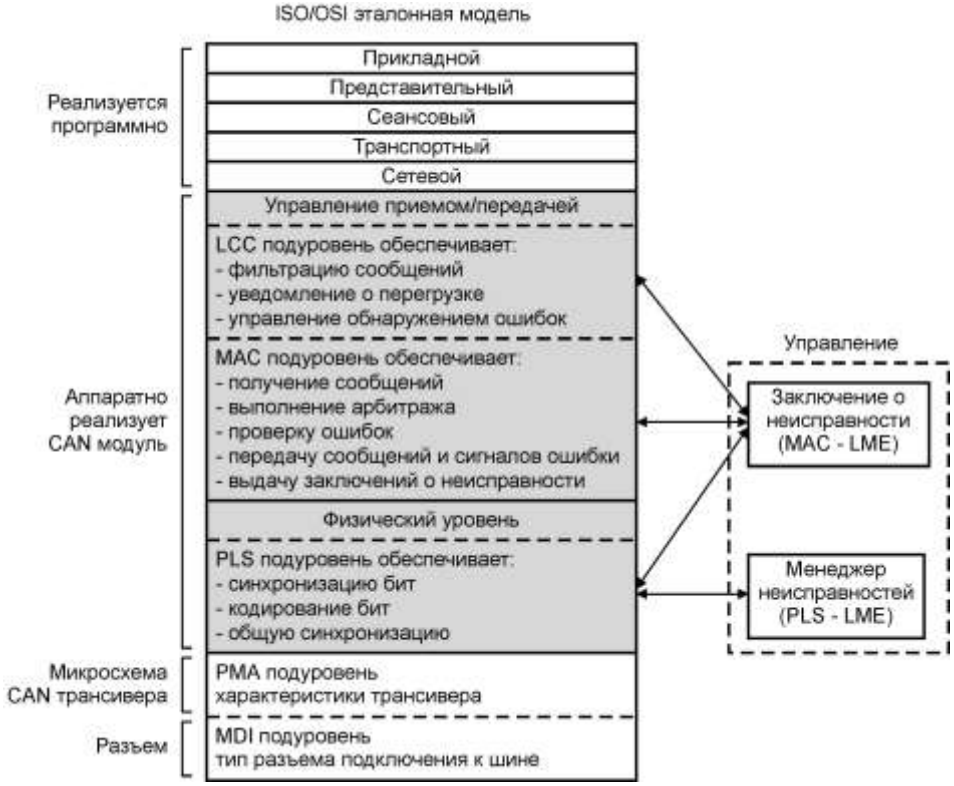


Рисунок 23.7 – ISO/OSI модель

Полнофункциональный CAN модуль поддерживает два нижних уровня эталонной модели сети:

- управление приемом/передачей данных:
  - управление логической связью – LCC подуровень,
  - среднее управление доступом – MAC подуровень;
- физический уровень: физическая сигнализация – PLS подуровень.

LCC подуровень обеспечивает:

- фильтрацию сообщений;
- уведомление о перегрузке;
- управление обнаружением ошибок.

MAC подуровень обеспечивает управление средой передачи:

- получение сообщений;
- выполнение арбитража;
- проверку ошибок;
- передачу сообщений и сигналов ошибки;
- выдачу заключений о неисправности.

MAC подуровень получает сообщения от LCC для передачи по шине и передает сообщения, принятые с шины в LCC подуровень. В пределах подуровня MAC определяется отсутствие активности на шине CAN для начала передачи сообщения.

Заключение о неисправности является механизмом самоконтроля при постоянном возникновении кратковременных ошибок.

Физический уровень определяет фактическую передачу бит между различными устройствами с выполнением всех электрических требований.

PLS подуровень определяет физическую структуру сигнала и контролирует:

- синхронизацию бит;
- кодирование бит;
- общую синхронизацию.

Низшие уровни протокола определяют фактический тип интерфейса связи: витая пара, волоконный световод, т. д. В пределах одной сети физический уровень должен быть одинаков для всех узлов. Характеристики драйверов физического уровня не определены в спецификации на CAN протокол, чтобы позволить оптимизировать среду передачи для конкретных приложений.

23.2 Модуль интерфейса CAN

В микроконтроллере 1887ВЕ3Т находится модуль интерфейса CAN, который включает в себя два идентичных независимых блока CAN (в дальнейшем CAN узел 0 и CAN узел 1), с общим ОЗУ, между которыми есть отличие лишь в подключении к выводам, см. рисунок 23.8.

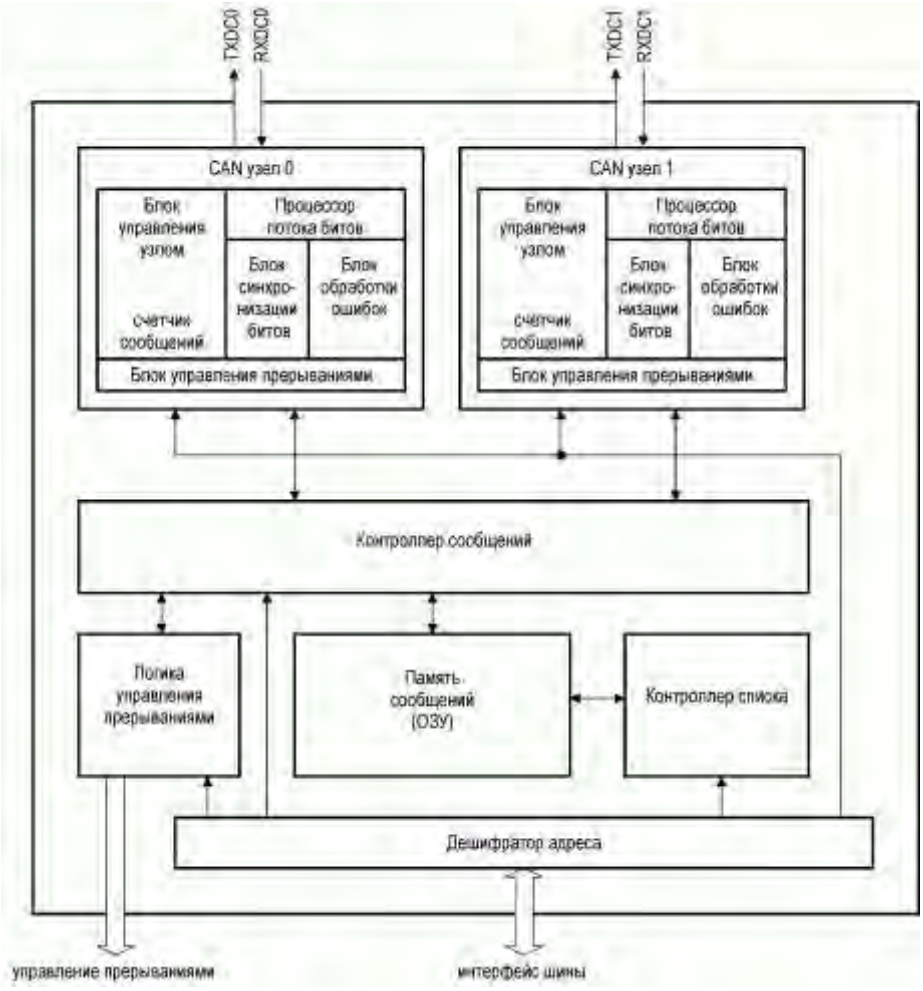


Рисунок 23.8 – Модуль интерфейса CAN

Помимо стандартной передачи сообщений, когда каждый узел передает на свои выходы, в контроллере реализовано логическое «И» передач CAN узлов (вывод результата сложения сигналов передачи обоих узлов на один выход), см. рисунок 23.9.

Пояснения:

NCRx – регистр управления узла x.
NSRx – регистр состояния узла x.
NIPRx – регистр указателя прерываний узла x.
NPCRx – регистр управления портом узла x.
NBTRx – регистр синхронизации битов узла x.
NECNTx – регистр счетчика ошибок узла x.
NFCRx – регистр счетчика сообщений узла x.
MOCTRn – регистр управления областью сообщения n.
MOSTATn – регистр состояния области сообщения n.
MOIPRn – регистр указателя прерываний области сообщения n.
MOFCRn – регистр управления функционированием области сообщения n.
MOFGPRn – регистр указателя FIFO/шлюза области сообщения n.
MOAMRn – регистр маски области сообщения n.
MOARn – регистр арбитража области сообщения n.
MODATALn – регистр (младший) данных области сообщения n.
MODATAHn – регистр (старший) данных области сообщения n.

CLC – регистр управления частотой.
ID – регистр идентификации.
FDR – регистр делителя.
PANCTR – регистр панели команд.
MCR – регистр управления.
MITR – регистр прерываний.
MSPND0 – регистр 0 ждущих прерываний.
MSPND1 – регистр 1 ждущих прерываний.
MSID0 – регистр 0 индекса сообщения.
MSID1 – регистр 1 индекса сообщения.
MSIMASK – регистр маски индекса сообщения узла x.
LIST0 – регистр списка №0 нераспределенных областей сообщений.
LIST1 – регистр списка №1 узла 0.
LIST2 – регистр списка №2 узла 1.
LIST3 – регистр свободного списка №3.
LIST4 – регистр свободного списка №4.
LIST5 – регистр свободного списка №5.
LIST6 – регистр свободного списка №6.
LIST7 – регистр свободного списка №7.

1) x – номер CAN узла равен 0 или 1.
2) n – номер области сообщения, n = 0 ... 31.

Рисунок 23.9 – Регистры модуля CAN

Характеристики модуля CAN:

- соответствие ISO 11898;
- функционирование согласно спецификации CAN 2.0B активная версия;
- управляющие регистры для каждого CAN узла;
- программируемая скорость передачи информации до 1 Мбит/с;
- гибкий и полный контроль передачи сообщений и обработки ошибок;
- 32 области сообщений (message object);
- полнофункциональная реализация CAN, в которой любая область сообщения может быть:
  - выделена для любого из узлов;
  - сконфигурирована как область «для передачи» или область «для приема»;
  - сконфигурирована с 11- или 29-битным идентификатором;
  - сконфигурирована для удаленных запросов;

- расширенная фильтрация входящих сообщений:

- каждая область сообщения имеет свою индивидуальную маску для фильтрации принимаемых сообщений;

- каждая область сообщения может быть сконфигурирована как для приема только стандартного или только расширенного сообщения, так и для приема сообщений обоих типов;

- области сообщений могут быть объединены в классы, каждый из которых может иметь один из четырех уровней приоритета для приема и передачи;

- распределение приоритета сообщений при передаче согласно правилам арбитража (см. спецификацию CAN 2.0B) или согласно их позициям в списке;

- расширенные возможности областей сообщений:

  - области сообщений могут быть объединены для построения FIFO структур с произвольными размерами, ограниченными только количеством областей сообщений;

  - области сообщений CAN узлов могут соединяться попарно с образованием, так называемого, шлюза, что позволяет автоматическую передачу сообщений между CAN шинами. Между CAN узлами может быть сформировано произвольное число шлюзов;

- расширенное управление данными:

  - организация областей сообщений в двухсвязные списки;

  - переорганизация списков возможна в любой момент времени, также во время функционирования CAN узлов;

  - управляемый «Контроллер списка» отвечает за организацию структуры списка и отслеживает его наполнение;

  - FIFO области сообщений организуются в списки, размеры которых в любой момент могут быть изменены;

  - жестко распределенные команды делают модуль CAN совместимым с TwinCAN устройствами, которые не имеют в своей структуре списков;

- расширенное обслуживание прерываний:

  - доступно до 16 линий прерываний. Каждый запрос прерывания может быть индивидуально скоммутирован на любую из 16 линий;

  - информация о последующей обработке сообщения отражается в специальном регистре, состоящем из битов уведомления.

23.3 CAN узел

CAN узел состоит из нескольких функциональных блоков, показанных на рисунке 23.8.

Процессор потока битов

Процессор потока битов управляет операциями с сообщениями данных, удаленного запроса, ошибки и перезагрузки (соответствует стандарту ISO 11898), также выполняет роль преобразователя между последовательным потоком данных и регистрами ввода-вывода.

Блок синхронизации битов

Блок синхронизации битов определяет длительность времени бита и положение точки выборки, согласно запрограммированным установкам, управляет вставкой задержки и ошибкой сдвига фазы. Также отвечает за ресинхронизацию.

Блок обработки ошибок

Блок обработки ошибок управляет счетчиками ошибок приема и передачи. В зависимости от состояния обоих счетчиков, CAN узел может находиться в состоянии активной ошибки, пассивной ошибки или быть отключенным от шины.

Блок управления узлом

Блок управления узлом координирует работу CAN узла:

- разрешает/запрещает действия узла на шине;
- разрешает/запрещает и генерирует различные события, касающиеся работы узла (например, «ошибка на шине», «успешное завершение передачи»), которые приводят к формированию запросов на прерывания;
- управляет счетчиком сообщений.

Блок управления прерываниями

Блок управления прерываниями управляет генерированием прерываний в течение работы CAN узла.

Контроллер сообщений

Контроллер сообщений управляет обменом сообщениями между CAN узлами и памятью сообщений и выполняет следующие функции:

- фильтрацию входящих сообщений для определения корректной области сообщения для сохранения полученных данных;
- определение области сообщения, содержимое которой будет передано в первую очередь (для каждого узла индивидуально);
- передачу содержимого области сообщения к CAN узлу, с параллельной вставкой в сообщение битов управления и состояния;
- осуществление буферизации FIFO и функционирования шлюза;
- объединение битов уведомления ждущих обработки сообщений.

Контроллер списка

Контроллер списка выполняет все операции по модификации списков областей сообщений. Только контроллер списка может изменять структуру списка. Распределение и перераспределение областей сообщений возможно посредством командного интерфейса (панели команд), остальные операции конечный автомат контроллера списка выполняет автономно.

Управление прерываниями

На рисунке 23.10 показана структура формирования запроса на прерывание.

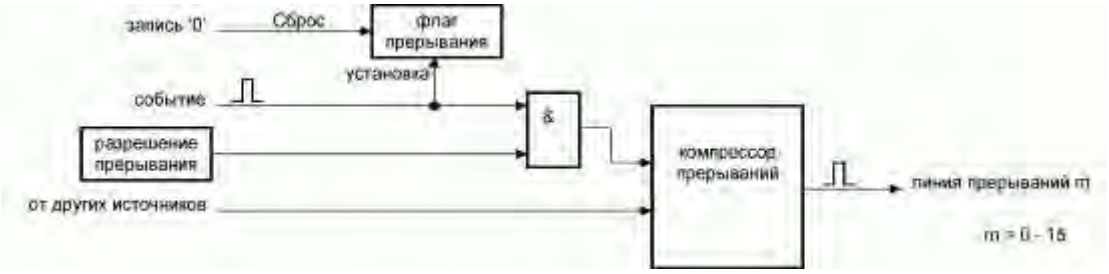


Рисунок 23.10 – Структура формирования запроса на прерывание

Событие, по которому должен быть сгенерирован запрос на прерывание, устанавливает флаг прерывания и, если это разрешено, формирует запрос на прерывание на одной, выбранной из 16, линии прерываний. Импульс запроса на прерывание генерируется независимо от состояния флага прерывания. Флаг прерывания может быть сброшен программно, записью «0».

Если к одной линии прерываний подключены несколько источников прерываний, то появление импульса от любого источника сформирует запрос на прерывание.

Логика управления прерываниями использует схему компрессии прерываний, см. рисунок 23.11.

Источниками прерываний являются:

- CAN узлы, 8 источников по четыре для каждого узла;
- области сообщений, 64 источника по два для каждой области;
- программное прерывание, один источник (регистр MITR).

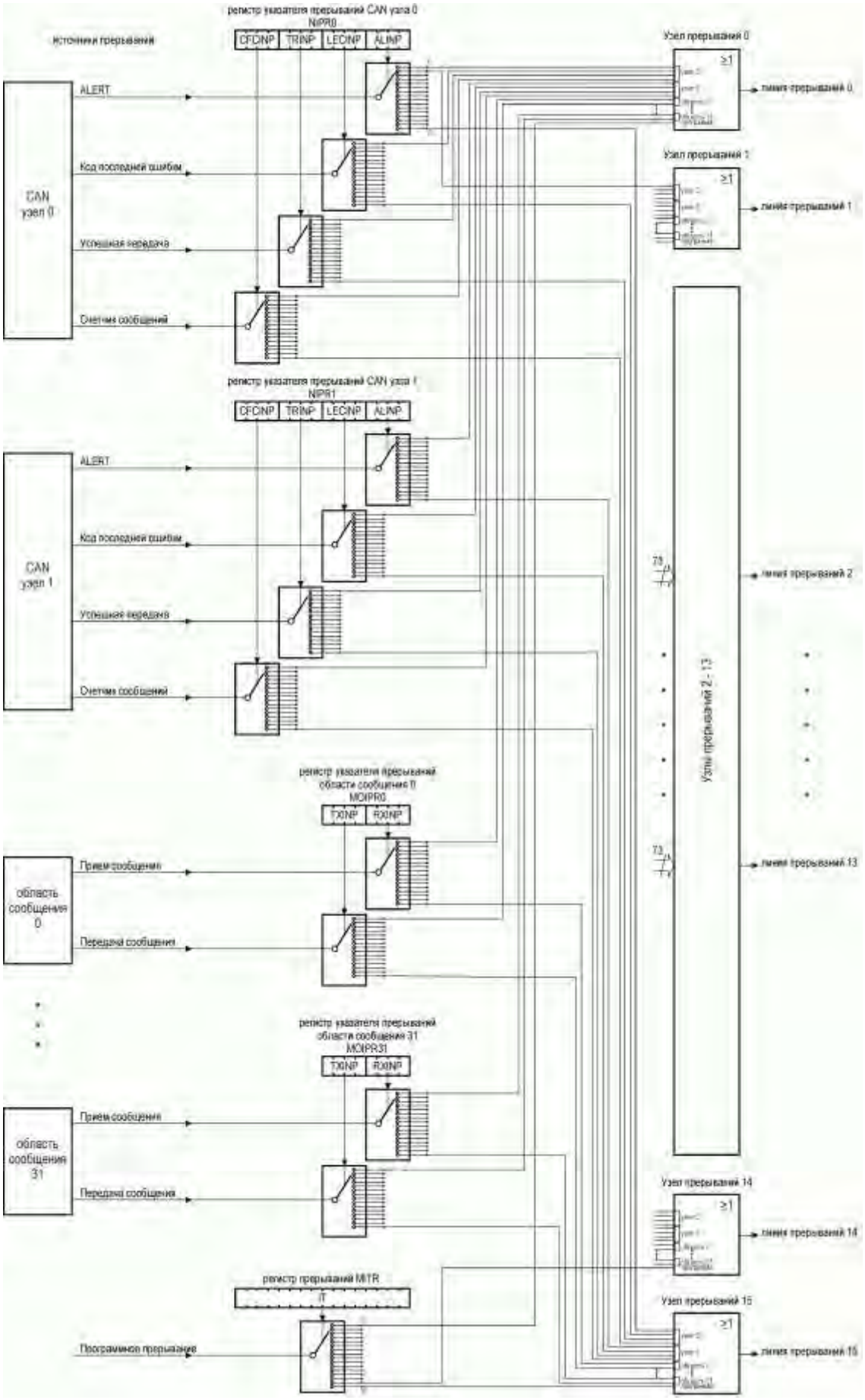


Рисунок 23.11 – Компрессор прерываний

Каждый аппаратный источник прерывания управляется 4 битами указателя прерываний, который определяет для него одну из 16 линий прерываний, что позволяет коммутировать на одну линию несколько источников прерываний.

Формат регистров NIPR0 и NIPR1 представлен на рисунке 23.12, функциональное назначение полей данных регистров – в таблице 23.1, коды выбора линии прерываний для полей ALINP, LECINP, TRINP, CFCINP регистров NIPR0 и NIPR1 – в таблице 23.2.

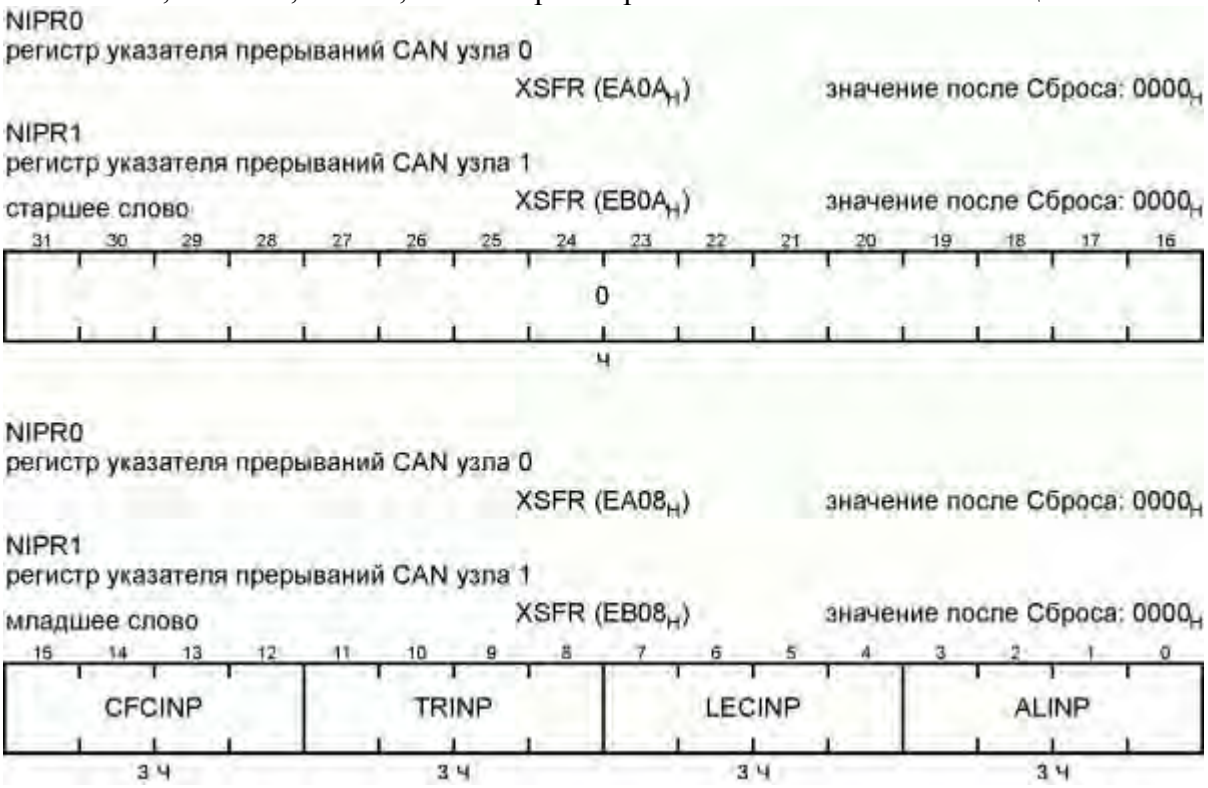


Рисунок 23.12 – Формат регистров NIPR0 и NIPR1

Таблица 23.1 – Функциональное назначение полей регистров NIPR0, NIPR1

Поле	Биты	Тип	Описание
1	2	3	4
ALINP	3-0	Чтение Запись	Указатель узла для ALERT прерывания. Это битовое поле выбирает выходную линию прерываний (одну из 16) для ALERT прерывания CAN узла. Кодировка в таблице 23.2.
LECINP	7-4	Чтение Запись	Указатель узла для прерывания кода последней ошибки. Это битовое поле выбирает выходную линию прерываний (одну из 16) для прерывания кода последней ошибки CAN узла. Кодировка в таблице 23.2.
TRINP	11-8	Чтение Запись	Указатель узла для прерывания после успешной пересылки сообщения. Это битовое поле выбирает выходную линию прерываний (одну из 16) для прерывания, формируемого после успешной пересылки сообщения CAN узла. Кодировка в таблице 23.2.
CFCINP	15:12	Чтение Запись	Указатель узла для прерывания счетчика сообщений. Это битовое поле выбирает выходную линию прерываний (одну из 16) для прерывания счетчика сообщений CAN узла. Кодировка в таблице 23.2.
0	31-16	Чтение	Зарезервировано При чтении возвращает «0». При записи писать «0».

Таблица 23.2 – Коды выбора линии прерываний для полей ALINP, LECINP, TRINP, CFCINP регистров NIPR0 и NIPR1

Код	Номер выходной линии прерываний
0000_B	0
0001_B	1
0010_B	2
0011_B	3
0100_B	4
0101_B	5
0110_B	6
0111_B	7
1000_B	8
1001_B	9
1010_B	10
1011_B	11
1100_B	12
1101_B	13
1110_B	14
1111_B	15

Формат регистров MOIPRn, где n = 0 – 31, представлен на рисунке 23.13, функциональное назначение полей регистров MOIPRn – в таблице 23.3, коды выбора линии прерываний для полей TXINP и RXINP регистров MOIPRn – в таблице 23.4.

*) Адрес – в таблице 23.4.

Рисунок 23.13 – Формат регистров MOIPRn, где n = 0 – 31

Таблица 23.3 – Функциональное назначение полей регистров MOIPRn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
RXINP	3-0	Чтение Запись	Указатель прерывания приема. Это битовое поле выбирает выходную линию прерываний (одну из 16) для прерывания, формируемого областью сообщения n после приема сообщения. Кодировка в таблице 23.4.

Окончание таблицы 23.3

1	2	3	4
TXINP	7-4	Чтение Запись	Указатель прерывания передачи. Это битовое поле выбирает выходную линию прерываний (одну из 16) для прерывания, формируемого областью сообщения n после передачи сообщения. Кодировка в таблице 23.4.
MPN	15-8	Чтение Запись	Номер ждущего сообщения. Это битовое поле указывает позицию флага в регистре ждущих прерываний MSPND0/ MSPND1, который выставляется в ответ на формирование прерывания областью сообщения n по окончании приема/передачи сообщения.
CFCVAL	31-16	Чтение Запись Аппаратное влияние	Значение счетчика сообщений. После того, как полученное сообщение сохранено в области сообщения n или после успешной передачи данных из области сообщения n, значение битового поля счетчика сообщений CFC (регистра NFCR) копируется в битовое поле CFCVAL.

Таблица 23.4 – Коды выбора линии прерываний для полей TXINP и RXINP регистров MOIPRn, где n = 0 – 31

Код	Номер выходной линии прерываний
0000_B	0
0001_B	1
0010_B	2
0011_B	3
0100_B	4
0101_B	5
0110_B	6
0111_B	7
1000_B	8
1001_B	9
1010_B	10
1011_B	11
1100_B	12
1101_B	13
1110_B	14
1111_B	15

Формат регистра MITR представлен на рисунке 23.14, функциональное назначение полей регистра MITR – в таблице 23.5.

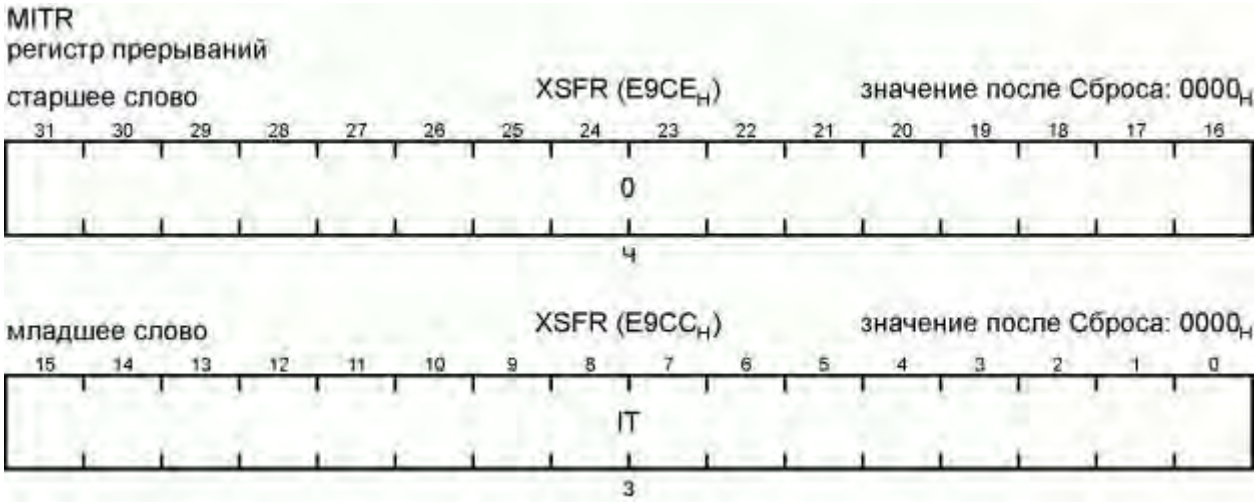


Рисунок 23.14 – Формат регистра MITR

Таблица 23.5 – Функциональное назначение полей регистра MITR

Поле	Биты	Тип	Описание
IT	15-0	Запись	Генерирование прерывания. Для того чтобы на желаемой (одной из 16) линии прерываний сформировался запрос на прерывание, необходимо записать «1» в бит, соответствующий этой линии прерываний. Чтобы сформировать запросы на прерывания одновременно на нескольких линиях прерываний, необходимо одновременно записать «1» в биты, соответствующие этим линиям прерываний. Номер позиции бита в поле IT совпадает с номером соответствующей ему линии прерываний. Запись «0» в биты поля IT игнорируется. Чтение регистра всегда возвращает нули.
0	31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Когда область сообщения n генерирует запрос на прерывание по окончании приема или передачи сообщения, запрос передается на линию прерываний, выбранную в битовом поле RXINP или TXINP регистра MOIPRn области сообщения n. Если количество областей сообщений больше, чем количество линий прерываний, то на одну линию могут приходить несколько запросов прерываний. Для разрешения конфликтов на линиях прерываний в модуле CAN предусмотрен механизм распределения приоритетов для областей сообщений.

23.4 Управление модулем CAN

В случаях, когда модуль CAN не используется, он может быть принудительно выключен установкой бита DISR в регистре CLC, см. рисунок 23.15, таблицу 23.6, при этом бит состояния DISS того же регистра всегда показывает, выключен ли модуль CAN DISS = 1_B или включен DISS = 0_B.

По умолчанию состояние модуля CAN после сброса – «выключен» DISS = 1_B.

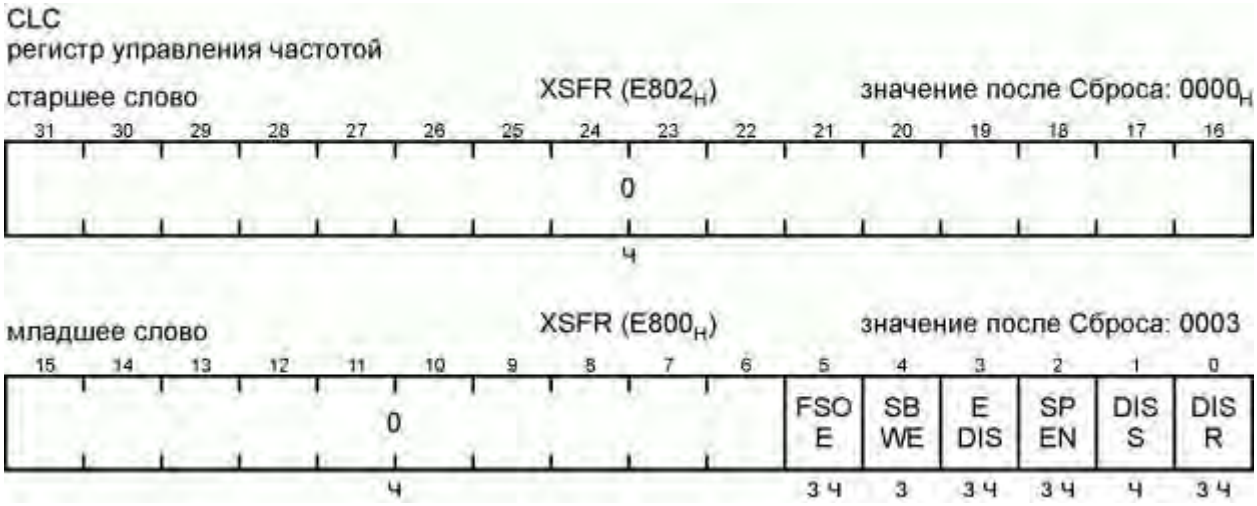


Рисунок 23.15 – Формат регистра CLC

Когда выключенный модуль CAN включается записью «0» в CLC.DISR, бит состояния DISS меняется из «1» в «0». Следует программно проверять состояние модуля CAN чтением бита DISS перед началом программирования регистров модуля.

Примечание – После сброса тактирование модуля CAN внешней частотой f_{CLC} должно быть разрешено записью «0» в DISR до начала задания тактовой частоты таймера модуля CAN в регистре делителя FDR.

Как уже было сказано выше, для выключения модуля CAN следует записать «1» в бит DISR регистр CLC.

При выключении модуль CAN, в первую очередь, завершает все текущие операции. После того, как все блоки переходят в состоянии готовности к выключению, модуль CAN утверждает состояние бита DISS и отключает внутреннее тактирование. После этого логика управления частотой контроллера отключает подачу тактового сигнала к модулю CAN.

Таблица 23.6 – Функциональное назначение полей регистра CLC

Поле	Биты	Тип	Описание
1	2	3	4
DISR	0	Чтение Запись	Бит запроса выключения модуля. Управляет включением/выключением модуля CAN. 0 Выключение модуля не требуется. 1 Требуется выключение модуля. После записи «1» в DISR начинается процесс выключения модуля CAN.
DISS	1	Чтение	Бит состояния модуля. Отражает текущее состояние модуля CAN. 0 Модуль находится в рабочем состоянии. 1 Модуль выключен.
SPEN[1)]	2	Чтение Запись	Бит разрешения режима приостановки работы (Suspend) для системы отладки схемы. Разрешает переход модуля CAN в режим приостановки работы. 0 Режим запрещен (переход модуля в режим приостановки работы запрещен). 1 Режим разрешен (переход модуля в режим приостановки работы разрешен). Бит SPEN может быть доступен для записи, только если бит SBWE = 1_B.
EDIS	3	Чтение Запись	Бит внешнего обращения. Управляет ответной реакцией модуля CAN на внешнее указание перехода в режим сна. 0 Режим сна разрешен. 1 Режим сна запрещен. Примечание – Режим сна для данного модуля CAN недоступен.
SBWE[1)]	4	Запись	Бит разрешения для битов управления приостановкой работы для системы отладки схемы. Определяет возможность записи в биты SPEN и FSOE. 0 Запись в биты запрещена. 1 Запись в биты разрешена. Бит SBWE доступен только для записи, но при этом значение, записываемое в этот бит, не сохраняется. Чтение SBWE всегда возвращает «0».
FSOE[1)]	5	Чтение Запись	Бит разрешения быстрого выключения модуля CAN. Разрешает/запрещает возможность быстрого выключения модуля для системы отладки схемы. 0 Выключение модуля запрещено. 1 Выключение модуля разрешено.

Окончание таблицы 23.6

1	2	3	4
0	15-6, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».
Примечание – Когда модуль CAN находится в выключенном состоянии, только регистр CLC доступен для записи/чтения. Доступ к остальным регистрам модуля запрещен. _____ 1) Биты доступны только в отладочном режиме работы схемы.			

Делитель частоты

Делитель частоты модуля CAN позволяет генерировать и выводить f_{OUT}, программируемую внутреннюю частоту f_{CAN} из внешней частоты f_{CLC} делением на любое число от 0 до 1023. Делитель частоты контролируется регистром делителя FDR, см. рисунок 23.16, таблицу 23.7, и может быть сконфигурирован для работы в двух режимах:

- нормальном;
- дробного деления.

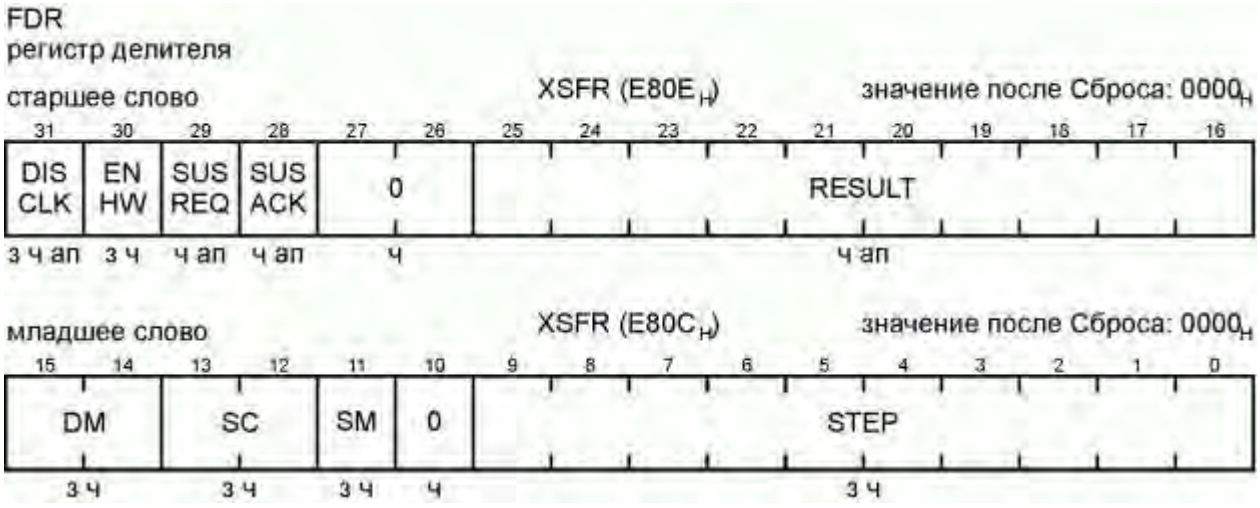


Рисунок 23.16 – Формат регистра FDR

Таблица 23.7 – Функциональное назначение полей регистра FDR

Поле	Биты	Тип	Описание
1	2	3	4
STEP	9-0	Чтение Запись	Значение шага. В нормальном режиме работы делителя в STEP хранится значение перезагрузки для битового поля RESULT. В режиме дробного деления в STEP хранится значение, прибавляемое к значению битового поля RESULT.
SM[1)]	11	Чтение Запись	Бит выбора режима приостановки работы (Suspend): 0 Промежуточный режим приостановки работы. 1 Основной режим приостановки работы.
SC[1)]	13-12	Чтение Запись	Управление в режиме приостановки работы. Определяет поведение делителя в режиме приостановки работы. 0 0 Нормальное генерирование тактовой частоты. 0 1 Генерирование тактовой частоты останавливается. f_{CAN} не генерируется. Значение RESULT не изменяется, за исключением случая, когда DM = 01_B или DM = 10_B. 1 0 Генерирование тактовой частоты останавливается. f_{CAN} не генерируется. В RESULT загружается значение $3FF_H$. 1 1 Так же как при SC = 10_B, но значение сигнала сброса внешнего делителя RED равно «1» (независимо от состояния битового поля DM).
DM	15-14	Чтение Запись	Выбор режима делителя: 0 0 Делитель выключен. Частота f_{CAN} не генерируется. Сигнал сброса внешнего делителя RED равен «1». Значение поля RESULT не изменяется (остается равным значению после сброса). 0 1 Делитель в нормальном режиме работы. 1 0 Делитель в режиме дробного деления. 1 1 Делитель выключен. Частота f_{CAN} не генерируется. Значение поля RESULT не изменяется.
RESULT	25-16	Чтение Аппаратное влияние	Результат: - в нормальном режиме RESULT работает как перезагружаемый счетчик с инкрементированием на 1; - в режиме дробного делителя, RESULT работает как счетчик с инкрементированием на значение, хранящееся в битовом поле STEP; - если DM = 01_B или DM = 10_B, в RESULT загружается значение $3FF_H$ (максимальное значение счетчика).

Окончание таблицы 23.7

1	2	3	4
SUSACK[1)]	28	Чтение Аппаратное влияние	Бит подтверждения режима приостановки работы (Suspend): 0 Режим приостановки не подтверждается. 1 Режим приостановки подтверждается. Примечание – Модуль CAN переходит в режим приостановки работы, когда оба бита SUSREQ и SUSACK равны «1».
SUSREQ[1)]	29	Чтение Аппаратное влияние	Бит запроса включения режима приостановки работы: 0 Включение режима приостановки работы не требуется. 1 Требуется включение режима приостановки работы. Примечание – Модуль CAN переходит в режим приостановки работы, когда оба бита SUSREQ и SUSACK равны «1».
ENHW	30	Чтение Запись	Бит разрешения аппаратного контроля частоты. Управляет подачей внешней тактовой частоты и битом DISCLK. 0 Бит DISCLK не может быть очищен аппаратно в ответ на появление высокого уровня сигнала разрешения внешней тактовой частоты. 1 Бит DISCLK может быть очищен аппаратно в ответ на появление высокого уровня сигнала разрешения внешней тактовой частоты. Примечание – Сигнал разрешения внешней тактовой частоты для делителя модуля CAN аппаратно удерживается равным «0».
DISCLK	31	Чтение Запись Аппаратное влияние	Бит запрета тактирования. Аппаратно управляемый запрет выходного сигнала f_{CAN} делителя. 0 Генерирование частоты f_{CAN} разрешено, согласно установкам (см. битовое поле DM). 1 Делитель выключен. Частота f_{CAN} не генерируется. В случае конфликта между аппаратным сбросом и программной установкой бита DISCLK приоритет остается за программой. Операций записи/чтения желательно избегать.
0	10, 27-26	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

[1)] Биты доступны только в отладочном режиме работы схемы.

Нормальный режим делителя

В нормальном режиме FDR.DM = 01_B делитель работает как обычный перезагружаемый счетчик. Шаг инкрементирования равен 1 (RESULT = RESULT + 1). Каждый раз, при переполнении RESULT = $3FF_H$, формируется импульс тактовой частоты f_{OUT}, счетчик сбрасывается, и в него загружается значение, хранящееся в FDR.STEP.

Выходная частота определяется по формуле

$$f_{OUT} = f_{IN} \times 1/n, \quad (23.1)$$

где n = 1024 – STEP (в десятичном формате).

Для получения частоты $f_{OUT} = f_{IN}$ значение STEP должно быть $3FF_H$ (1023 в десятичном формате). На рисунке 23.17 показано формирование тактовой частоты f_{OUT} при значении STEP = $3FD_H$ (1021). После подстановки значения 1021 в формулу получим:

$$f_{OUT} = f_{IN} \times 1/(1024 - 1021) = f_{IN} / 3. \quad (23.2)$$

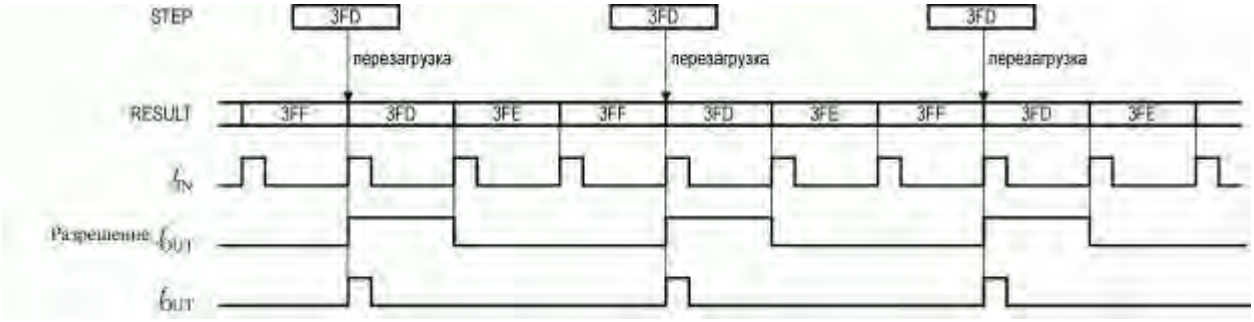


Рисунок 23.17 – Формирование тактовой частоты в нормальном режиме

Режим дробного деления

В режиме дробного деления FDR.DM = 10_B выходная частота f_{OUT} формируется из входной частоты f_{IN} умножением на дробь n/1024, где n может принимать значения от 0 до 1023. В целом, режим дробного деления позволяет программировать частоту с более высокой точностью, чем нормальный режим.

В режиме дробного деления делитель также работает как перезагружаемый счетчик, но шаг инкрементирования равен значению STEP (RESULT = RESULT + STEP). Если результат сложения значений RESULT и STEP превышает $3FF_H$, возникает переполнение счетчика и генерируется импульс тактовой частоты f_{OUT}. Значение, на которое результат сложения превысил $3FF_H$, загружается в счетчик (RESULT) и счет продолжается.

Следует помнить, что в режиме дробного деления частота f_{OUT} может иметь джиттер периода, не превышающий одного периода f_{IN}.

Выходная частота определяется по формуле

$$f_{OUT} = f_{IN} \times n/1024, \quad (23.3)$$

где n от 0 до 1023.

На рисунке 23.18 показано формирование тактовой частоты f_{OUT} при значении STEP = 234_H (564 в десятичном формате). После подстановки значения 564 в формулу (23.3) получим:

$$f_{OUT} = f_{IN} \times 564/1024 = 0{,}55 \times f_{IN}$$

Сигнал тактовой частоты f_{OUT} получается из логического «И» сигналов разрешения FOUT и FIN.

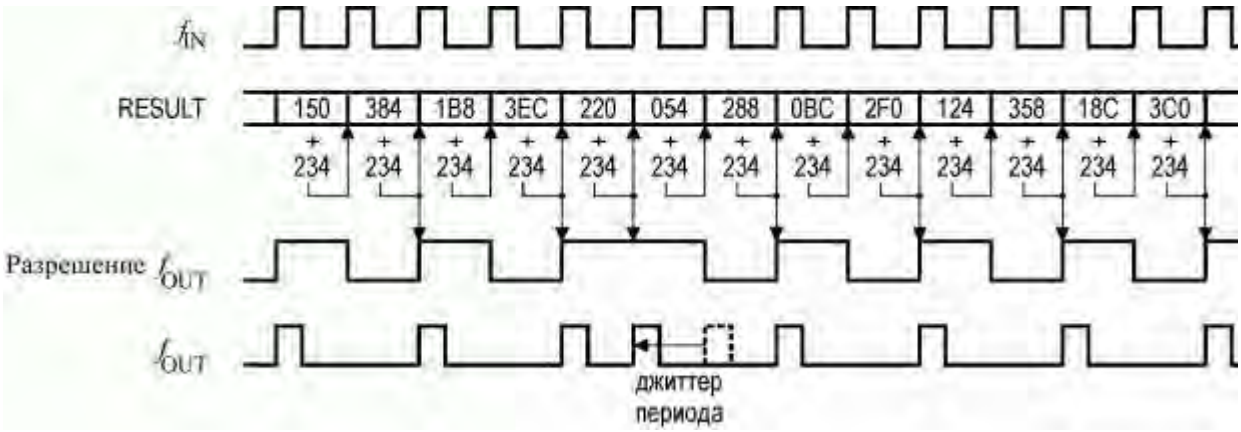


Рисунок 23.18 – Формирование тактовой частоты в режиме дробного деления

Таблица 23.8 – Сводная таблица сигналов управления функционирования делителя

SC	DM	DISS	RESULT	FOUT	Режим делителя
–	00_B	1	Без изменений	Не формируется	Выключен
	01_B	0	Загрузка[1)]	Формируется	Нормальный
	10_B				Дробный делитель
	11_B		Без изменений	Не формируется	Выключен

[1)] Каждый раз при записи в поле DM значения 01_B или 10_B, значение RESULT устанавливается в $3FF_H$.

Управление выводами для приема

С каждым CAN узлом связано 8 входных линий. Программно выбрать линию для приема сообщений можно посредством поля RXSEL регистра NPCR0/NPCR1 (для каждого CAN узла свой регистр).

Управление CAN узлами

Каждый CAN узел имеет свою собственную логику управления и выдачи информации о состоянии, и может быть сконфигурирован и работать независимо от другого узла.

Примечание – Далее в описании номера CAN узлов будут опущены. В названиях регистров под «x» будет подразумеваться номер CAN узла (0 и 1). Под «n» будет подразумеваться номер области сообщения (0 – 31).

Поскольку CAN узел 0 и CAN узел 1 идентичны, все нижесказанное справедливо для каждого из узлов.

Каждый их двух CAN узлов имеет свою группу регистров. Регистры в группах идентичны и отличаются только адресами, см. рисунки 23.19, 23.20, таблицы 23.9, 23.10.

NCR0
регистр управления CAN узла 0

XSFR ($EA02_H$) значение после Сброса: 0000_H

NCR1
регистр управления CAN узла 1

старшее слово XSFR ($EB02_H$) значение после Сброса: 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
Ч															

NCR0
регистр управления CAN узла 0

XSFR ($EA00_H$) значение после Сброса: 0001_H

NCR1
регистр управления CAN узла 1

младшее слово XSFR ($EB00_H$) значение после Сброса: 0001_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							SUS EN	CAL M	CCE	0	CAN DIS	ALI E	LEC IE	TRIE	INIT
Ч							З Ч	З Ч	З Ч	Ч	З Ч	З Ч	З Ч	З Ч	З Ч ап

Рисунок 23.19 – Формат регистров NCR0 и NCR1

NSR0
регистр состояния CAN узла 0

XSFR ($EA06_H$) значение после Сброса: 0000_H

NSR1
регистр состояния CAN узла 1

старшее слово XSFR ($EB06_H$) значение после Сброса: 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
Ч															

NSR0
регистр состояния CAN узла 0

XSFR ($EA04_H$) значение после Сброса: 0000_H

NSR1
регистр состояния CAN узла 1

младшее слово XSFR ($EB04_H$) значение после Сброса: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					SUS ACK	LOE	LLE	B OFF	E WRN	AL ERT	RX OK	TX OK	LEC		
Ч					Ч ап	З Ч ап	З Ч ап	Ч ап	Ч ап	З Ч ап	З Ч ап	З Ч ап	З Ч ал		

Рисунок 23.20 – Формат регистров NSR0 и NSR1

Таблица 23.9 – Функциональное назначение полей регистров NCR0 и NCR1

Поле	Биты	Тип	Описание
1	2	3	4
INIT	0	Чтение Запись Аппаратное влияние	Инициализация узла: 0 Сброс бита INIT разрешает участие узла в трафике CAN шины. Если на момент сброса бита INIT узел находился в состоянии «отключен от шины», идет процесс выхода их этого состояния (независимо от бита INIT). После получения необходимого количества последовательностей битов, узел выходит из состояния «отключен от шины» и снова включается в трафик. Если на момент сброса бита INIT, узел не находился в состоянии «отключен от шины», то узел ожидает обнаружение последовательности 11 рецессивных бит на шине и включается в трафик. 1 Установка бита INIT прекращает участие узла в трафике CAN шины. Все текущие передачи останавливаются, и линии передач переходят в рецессивное состояние. Если на момент установки бита INIT узел находился в состоянии «отключен от шины», процесс выхода их этого состояния продолжается. Если после завершения процесса выхода из состояния «отключен от шины» (после обнаружения 128 последовательностей – каждая из 11 рецессивных битов) бит INIT остается активным, узел выходит из состояния «отключен от шины», но остается неактивным до тех пор, пока INIT = 1_B. Бит INIT автоматически устанавливается, если узел переходит в состояние «отключен от шины».
TRIE	1	Чтение Запись	Разрешение прерывания узла после пересылки сообщения. После успешной пересылки сообщения (передача/ прием), если разрешено, генерируется прерывание. 0 Формирование прерывания запрещено. 1 Формирование прерывания разрешено. Битовое поле TRINP регистра NIPRx выбирает, какая из линий прерываний соответствует данному типу прерывания.
LECIE	2	Чтение Запись	Разрешение прерывания узла при возникновении ошибки кода последней ошибки. Прерывание формируется каждый раз при изменении/ обновлении битового поля LEC регистра NSRx, если LEC > 0 (ошибка CAN протокола). 0 Формирование прерывания кода последней ошибки запрещено. 1 Формирование прерывания кода последней ошибки разрешено. Битовое поле LECINP регистра NIPRx выбирает, какая из линий прерываний соответствует данному типу прерывания.

Продолжение таблицы 23.9

1	2	3	4
ALIE	3	Чтение Запись	Разрешение ALERT прерывания узла: 0 Формирование прерывания ALERT запрещено. 1 Формирование прерывания ALERT разрешено. ALERT прерывание, если разрешено, может быть сформировано любым из следующих событий: - изменение состояния бита BOFF регистра NSRx; - изменение состояния бита EWRN регистра NSRx; - ошибка длины списка, которая также выставляет бит LLE регистра NSR; - ошибка элемента списка, которая также выставляет бит LOE регистра NSR; - бит INIT выставлен аппаратно. Битовое поле ALINP регистра NIPRx выбирает, какая из линий прерываний соответствует данному типу прерывания.
CANDIS	4	Чтение Запись	Выключение узла. Установка этого бита выключает CAN узел. В первую очередь, узел переходит в состояние «простоя» или «отключен от шины», после чего автоматически устанавливается бит IDLE и, если разрешено (ALIE = 1_B), генерируется ALERT прерывание.
CCE	6	Чтение Запись	Разрешение изменения конфигурации: 0 Регистры синхронизации битов NBTRx, управления портом NPCRx и счетчика ошибок NECNTx узла доступны только для чтения. Все попытки записи в эти регистры игнорируются. 1 Регистры NBTRx, NPCRx и NECNTx доступны как для чтения, так и для записи.
CALM	7	Чтение Запись	Режим анализа CAN. Если этот бит установлен, модуль CAN функционирует в режиме анализа. Это означает, что сообщения могут только приниматься и не могут быть переданы. Бит подтверждения не посылается после успешного приема сообщения. Флаг активной ошибки посылается рецессивным вместо доминантного. На линии отправки сообщений поддерживается рецессивный «1» уровень. Бит CALM может быть установлен только в течение времени, когда установлен бит INIT.

Окончание таблицы 23.9

1	2	3	4
SUSEN	8	Чтение Запись	Разрешение режима приостановки. Бит позволяет перевести CAN узел в режим приостановки посредством отладочной системы OCDS. 0 Переход в режим приостановки невозможен. 1 Переход в режим приостановки возможен. По мере того, как CAN узел перейдет в состояние «простоя» или «отключен от шины», бит INIT будет автоматически установлен в «1». Действующее состояние бита INIT остается без изменений. Бит SUSEN сбрасывается по сигналу «Сброса» системы OCDS.
0	5, 15-9, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Таблица 23.10 – Функциональное назначение полей регистров NSR0 и NSR1

Поле	Биты	Тип	Описание
1	2	3	4
LEC	2-0	Чтение Запись Аппаратное влияние	Код последней ошибки. Показывает тип последней (из обнаруженных) CAN ошибки (кодировка в таблице 23.11). Примечание – Битовое поле LEC может быть изменено программно, только если в него записывается код «111_B», в остальных случаях запись игнорируется.
TXOK	3	Чтение Запись Аппаратное влияние	Успешная передача сообщения: 0 С момента сброса флага TXOK не было осуществлено ни одной успешной передачи. 1 Сообщение передано успешно (т. е. без ошибок и с получением бита подтверждения от другого узла). Флаг TXOK должен сбрасываться программно (записью «0»). Запись «1» в бит TXOK игнорируется.
RXOK	4	Чтение Запись Аппаратное влияние	Успешный прием сообщения: 0 С момента сброса флага RXOK не было получено ни одного сообщения. 1 Сообщение получено успешно. Флаг RXOK должен сбрасываться программно (записью «0»). Запись «1» в бит RXOK игнорируется
EWRN	6	Чтение Аппаратное влияние	Флаг критического количества ошибок: 0 Лимит ошибок еще не достигнут. 1 По крайней мере, один из счетчиков ошибок (REC, TEC) достиг лимита ошибок заданного битовым полем EWRNLVL.

Окончание таблицы 23.10

1	2	3	4
ALERT	5	Чтение Запись Аппаратное влияние	ALERT предупреждение. Флаг выставляется в следующих случаях (не взаимоисключающих): - изменение бита BOFF регистра NSRx; - изменение бита LOE регистра NSRx; - ошибка длины списка, которая также выставляет бит LLE регистра NSR; - ошибка элемента списка, которая также выставляет бит LOE регистра NSR; - бит INIT выставлен аппаратно. Флаг ALERT должен сбрасываться программно (записью «0»). Запись «1» в бит ALERT игнорируется.
BOFF	7	Чтение Аппаратное влияние	Флаг состояния «отключен от шины»: 0 Контроллер CAN не находится в состоянии «отключен от шины». 1 Контроллер CAN находится в состоянии «отключен от шины».
LLE	8	Чтение Запись Аппаратное влияние	Флаг ошибки длины списка: 0 С момента сброса флага LLE не было обнаружено ни одной ошибки длины списка. 1 Ошибка длины списка обнаружена при приеме сообщения. В списке, который принадлежит принимающему CAN узлу, количество элементов отличается от указанного в битовом поле SIZE. Флаг LLE должен сбрасываться программно (записью «0»). Запись «1» в бит LLE игнорируется.
LOE	9	Чтение Запись Аппаратное влияние	Флаг ошибки элемента списка: 0 С момента сброса флага LOE не было обнаружено ни одной ошибки элемента списка. 1 Ошибка элемента списка обнаружена при приеме сообщения. Обнаружена операция внесения в список области сообщения с несоответствующим номером списка LIST в регистре MOCTRn. Флаг LOE должен сбрасываться программно записью «0». Запись «1» в бит LOE игнорируется.
SUSACK	10	Чтение Запись Аппаратное влияние	Бит подтверждения перехода узла CAN в режим приостановки: 0 CAN узел не находится в режиме приостановки или ожидает перехода в режим приостановки, но еще не достиг состояния «простоя» или «отключен от шины». 1 CAN узел находится в режиме приостановки. CAN узел неактивен (бит INIT аппаратно установлен в «1») из-за использования его системой OCDS.
0	15-11, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0»

Таблица 23.11 – Коды битового поля LEC

Значение LEC	Значение
1	2
000_B	Ошибок нет. На шине не было обнаружено ни одной ошибки с момента передачи последнего сообщения до настоящего времени.
001_B	Ошибка стаффинга (заполнения, STUFF ERROR). Может быть обнаружена во время передачи шестого бита из последовательности шести одинаковых бит в поле сообщения, которое должно быть кодировано методом разрядного заполнения (заключается в том, что после передачи пяти битов одинаковой полярности, шестой бит должен иметь противоположную полярность и вставляться передатчиком в поток данных автоматически, приемник пропускает этот бит).
010_B	Ошибка формы (FORM ERROR). Обнаруживается, если: - в битовом поле фиксированного формата содержится количество битов, отличающееся от установленного; - на месте рецессивного бита находятся доминантный или наоборот. Исключение – для приемника доминантный бит в течение последнего бита поля «конец кадра» не интерпретируется как ошибка формы.
011_B	Ошибка подтверждения (ACKNOWLEGMENT ERROR). Обнаруживается передатчиком всякий раз, когда он не обнаруживает доминантный бит АСК в «области подтверждения».
100_B	Разрядная ошибка или ошибка бита 1 (BIT 1 ERROR). Узел, который передает данные на шину, осуществляет мониторинг шины. Ошибка бита 1 имеет место, если при передаче рецессивного «1» бита (за исключением битов полей арбитража и подтверждения) на шине обнаруживается доминантный «0» бит.
101_B	Разрядная ошибка или ошибка бита 0 (BIT 0 ERROR). Ошибка возникает в случаях: - во время передачи сообщения (или бита подтверждения, флага активной ошибки, флага перезагрузки), узел передает доминантный бит «0», но на шине обнаруживается рецессивный «1»; - во время выхода из состояния «отключен от шины» при каждом обнаружении последовательности из 11 рецессивных битов. В этом случае, ЦП может использовать код 101_B для отслеживания длительного простоя шины.
110_B	Ошибка циклического избыточного кода (CRC ERROR). Передатчик, по установленному алгоритму, вычисляет значение контрольной суммы (собственно CRC) для передаваемых данных и вставляет ее в сообщение. Приемник, после получения данных, вычисляет CRC по тому же алгоритму, что и передатчик и сравнивает вычисленное значение с принятым значением CRC. В случае не совпадения возникает ошибка.
111_B	ЦП производит запись в LEC. ЦП может записать код 111_B в LEC в любой момент времени. Попытки записи в LEC любого другого значения игнорируются.

Режим конфигурации включается установкой в «1» бита CCE регистра управления CAN узла NCRx. Режим конфигурации позволяет изменять параметры синхронизации битов и состояния счетчиков ошибок.

Режим анализа включается установкой в «1» бита CALM регистра NCRx. В режиме анализа сообщения данных и удаленные запросы отслеживаются без участия узла в операциях на шине (выход TXDC узла находиться в рецессивном состоянии). Входящие удаленные запросы сохраняются в соответствующих областях «для передачи», а входящие сообщения данных – в соответствующих областях «для приема».

Так же в режиме анализа полная информация о конфигурации полученных сообщений сохраняется в соответствующих областях сообщений и может быть обработана ЦП. В ответ на входящие сообщения не выдается подтверждение, и не генерируются сообщения об ошибках. На удаленные запросы не выдаются сообщения данных, а сами сообщения данных не могут быть переданы установкой бита запроса передачи TXRQ регистра состояния области сообщения MOSTATn. Прерывания после приема генерируются (если это разрешено) для всех принятых сообщений, не содержащих ошибок.

Конфигурация прерываний определяется логикой управления узла посредством битов TRIE, ALIE и LECIE регистра NCRx:

- если бит управления TRIE = 1_B, прерывание после передачи генерируется, когда содержимое регистра NCRx меняется (после каждой успешной передачи сообщения);
- если бит управления ALIE = 1_B, прерывание ошибки генерируется, когда устанавливается состояние «отключен от шины» или превышено допустимое количество ошибок, или обнаруживается «недобор» данных;
- если бит управления LECIE = 1_B, прерывание кода последней ошибки генерируется, когда в битовое поле LEC регистра NSRx аппаратно записывается значение кода ошибки больше нуля. В таблице 23.11 представлениы коды битового поля LEC.

Регистр состояния CAN узла NSRx отражает текущее состояние, содержит информацию о передачах и ошибках соответствующего узла.

Счетчик сообщений может использоваться для проверки последовательности передаваемых битов области сообщения или получения информации о завершении передачи/приема сообщения соответствующего CAN узла. Подсчет сообщений осуществляется 16-разрядным счетчиком, который управляется регистром счетчика сообщений CAN узла NFCRx. Битовые поля CFMOD и CFSEL определяют режим работы и событие для инкрементирования счетчика.

Блок синхронизации битов

Формат регистров NBTR0 и NBTR1 представлен на рисунке 23.21, функциональное назначение полей регистров NBTR0, NBTR1 – в таблице 23.12.

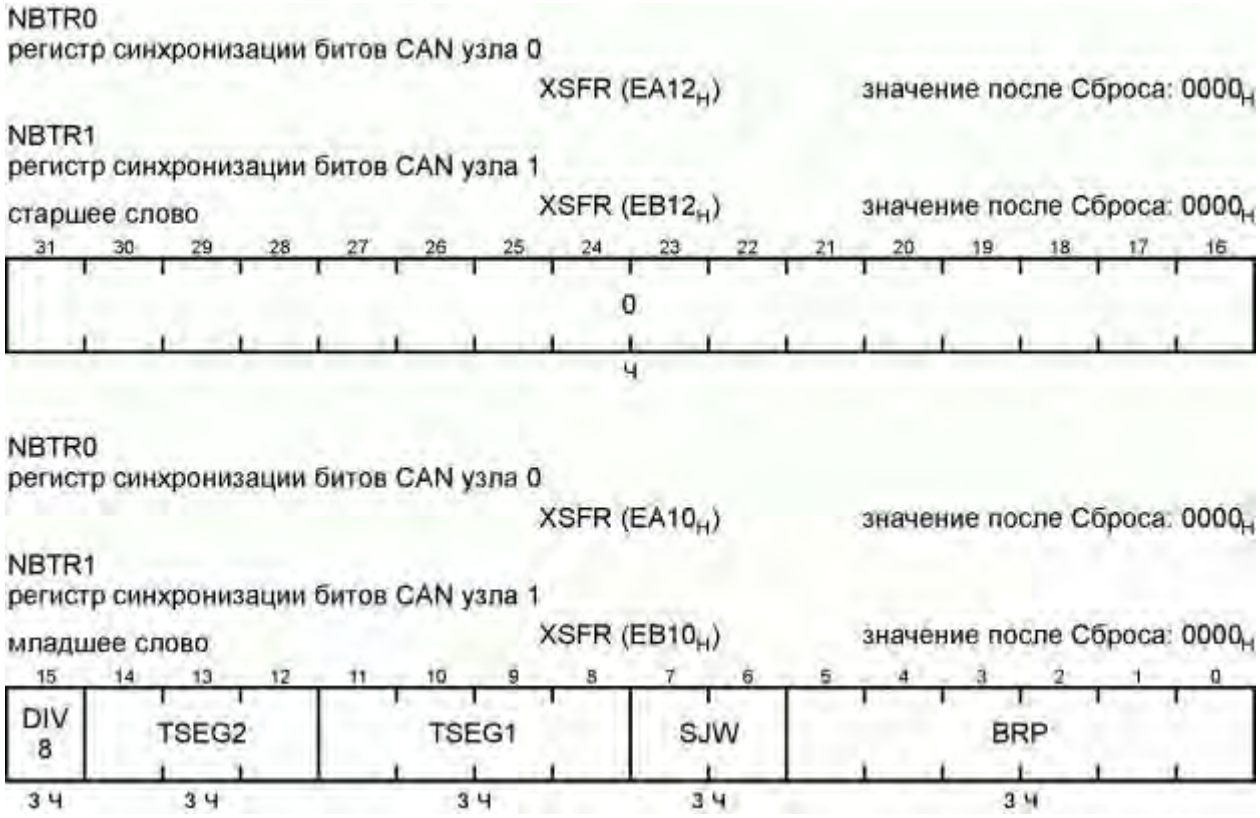


Рисунок 23.21 – Формат регистров NBTR0 и NBTR1

Таблица 23.12 – Функциональное назначение полей регистров NBTR0 и NBTR1

Поле	Биты	Тип	Описание
1	2	3	4
BRP	5-0	Чтение Запись	Скорости передачи. Если DIV8 = 0_B, тогда длительность одного кванта времени равна (BRP + 1) тактам частоты. Если DIV8 = 1_B, тогда длительность одного кванта времени равна 8×(BRP + 1) тактам частоты.
SJW	7-6	Чтение Запись	Ширина перехода ресинхронизации. Длительность ширины перехода ресинхронизации равна (SJW + 1) квантам времени.
TSEG1	11-8	Чтение Запись	Параметр 1. Временной сегмент от сегмента синхронизации до точки выборки, определяемый пользователем и включающий в себя сегмент распространения. Длительность сегмента равна (TSEG1 + 1) квантам времени и может быть увеличена за счет ресинхронизации. Допустимые значения для TSEG1 от двух до 15.

Окончание таблицы 23.12

1	2	3	4
TSEG2	14-12	чтение запись	Параметр 2. Временной сегмент от точки выборки до точки передачи, определяемый пользователем. Длительность сегмента равна (TSEG2 + 1) квантам времени и может быть уменьшена за счет ресинхронизации. Допустимые значения для TSEG1 от 1 до 7.
DIV8	15	чтение запись	Делитель частоты на 8: 0 Длительность кванта времени равна (BRP + 1) тактам частоты. 1 Длительность кванта времени равна 8×(BRP + 1) тактам частоты.
0	31-16	чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».
Примечание – Регистр NBTRx может быть записан, если только бит CCE регистра NCRx установлен.			

Согласно стандарта ISO 11898, время передачи одного бита разделено на сегменты, которые, в свою очередь, составлены из целочисленных отрезков времени, называемых квантами времени t_q, см. рисунок 23.22. Квант времени – фиксированная единица времени, получаемая из частоты синхронизации и делителя модуля CAN.

Сегмент синхронизации T_{Sync}. Позволяет синхронизировать начало обмена данными между передатчиком и приемником. Длительность сегмента всегда равна одному кванту времени t_q.

Сегмент распространения T_{Prop}. Используется для компенсации физического времени запаздывания сигнала в пределах сети. Длительность сегмента рассчитывается с учетом времени прохождения сигнала от передатчика к приемнику и обратно, входной задержки компаратора и задержки выхода драйвера и может составлять от 1 до 8 квантов времени.

Рисунок 23.22 – Структура одного бита

Сегменты буфера фазы 1 и буфера фазы 2 (T_{b1} и T_{b2}), расположенные до и после точки выборки, используются для компенсации смещения фазы тактовых частот источника и приемника, обнаруживаемой после появления сегмента синхронизации, а также для оптимального расположения точки выборки полученного бита.

Точка выборки – момент, когда читается состояние шины для определения принятого бита. Как правило, длительность временного интервала от начала бита до точки выборки составляет (60 – 70) % времени бита, в зависимости от системных параметров.

Сегмент распространения и сегмент буфера фазы 1 вместе составляют сегмент параметра 1 (T_{Seg1}), который определяется битовым полем TSEG1 регистра синхронизации битов NBTRx (может быть записан, только если бит CCE регистра NCRx установлен). Согласно стандарту ISO, минимальная длительность сегмента параметра 1 должна составлять 3 кванта времени.

Сегмент параметра 2 (T_{Seg2}) определяется битовым полем TSEG2 регистра NBTRx и охватывает сегмент буфера фазы 2. Минимальная длительность сегмента параметра 2 составляет 2 кванта времени.

Согласно стандарту ISO, минимальная длительность одного бита, получающаяся сложением сегментов T_{Sync}, T_{Seg1} и T_{Seg2} не должна быть менее 8 квантов времени.

Максимальная длительность бита – 25 квантов времени.

Примечание – Минимальное номинальное время передачи одного бита составляет 1 мкс, что соответствует скорости передачи 1 Мбит/с.

Формулы вычисления значений сегментов и времени одного бита:

$$t_q = (BRP + 1) / f_{CAN}, \text{ если } DIV8 = 0_B, \text{ или} \quad (23.4)$$

$$t_q = 8 \times (BRP + 1) / f_{CAN}, \text{ если } DIV8 = 1_B, \quad (23.5)$$

$$T_{Sync} = 1 \times t_q, \quad (23.6)$$

$$T_{Seg1} = (TSEG1 + 1) \times t_q \text{ (минимум 3 } t_q), \quad (23.7)$$

$$T_{Seg2} = (TSEG2 + 1) \times t_q \text{ (минимум 2 } t_q), \quad (23.8)$$

$$\text{Время бита} = T_{Sync} + T_{Seg1} + T_{Seg2} \text{ (минимум 8 } t_q). \quad (23.9)$$

Чтобы компенсировать смещение фазы между частотами генераторов различных узлов шины, каждый CAN модуль должен синхронизироваться по фронту смены уровня сигнала на шине от рецессивного к доминантному. Как только фронт обнаруживается, логика синхронизации сравнивает его текущее размещение с ожидаемым и выполняет настройку значений параметров 1 и 2 (T_{Seg1} и T_{Seg2}).

Есть два механизма синхронизации:

- аппаратная (жесткая синхронизация);
- ресинхронизация (синхронизация с восстановлением тактовых интервалов).

Аппаратная синхронизация выполняется по каждому фронту смены уровня сигнала на шине от рецессивного к доминантному. При аппаратной синхронизации временные интервалы сегментов, из которых складываются времена битов, не изменяются в течение всего сообщения.

Ресинхронизация выполняется автоматическим удлинением сегмента параметра 1 или укорачиванием сегмента параметра 2. Максимальное значение изменения сегментов параметров колеблется в пределах от 1 до 4 квантов времени. Синхронизация выполняется только при появлении фронта смены уровня сигнала на шине от рецессивного к доминантному. Фиксированное значение максимального числа последовательных бит одинаковой полярности гарантирует своевременное восстановление синхронизации. Смещение фазы фронта смены уровня сигнала на шине отслеживается относительно сегмента синхронизации и измеряется в квантах времени t_q.

Если величина фазового смещения меньше или равна запрограммированному значению ширины перехода ресинхронизации T_{SJW}, выполняется аппаратная синхронизация.

Если величина смещения фазы больше, чем ширина перехода ресинхронизации T_{SJW} и фазовое смещение положительно, то удлиняется сегмент параметра 1.

Если величина смещения фазы больше, чем ширина перехода ресинхронизации T_{SJW} и фазовое смещение отрицательно, то укорачивается сегмент параметра 2.

Значение T_{SJW} определяется полем SJW регистра NBTRx по формуле

$$T_{SJW} = (SJW + 1) \times t_q . \quad (23.10)$$

Помимо прочего, должны соблюдаться следующие правила:

$$T_{Seg1} \geq T_{SJW} + T_{Prop}, \quad (23.11)$$
$$T_{Seg2} \geq T_{SJW}. \quad (23.12)$$

Соотношения между максимальным отклонением частоты f_{CAN} и сегментами буферов фаз и шириной перехода ресинхронизации следующие:

$$df_{CAN} \leq \text{миним.}[T_{b1}, T_{b2}] / 2 \times (13 \times \text{время бита} - T_{b2}), \quad (23.13)$$
$$df_{CAN} \leq T_{SJW} / 20 \times \text{время бита}. \quad (23.14)$$

В итоге имеем:

- сегмент синхронизации составляет 1 квант времени;
- сегмент распространения – от 1 до 8 квантов времени;
- сегмент буфера фазы 1 – от 1 до 8 квантов времени;
- сегмент буфера фазы 2 выбирается равным двум квантам времени или равным сегменту буфера фазы 1, если его значение более 2 квантов времени;
- ширина перехода ресинхронизации может составлять максимально 4 кванта времени, однако в типовых приложениях достаточно 1.

Корректные значения параметров синхронизации битов должны быть записаны в регистр NBTRx до окончания инициализации (сброс бита INIT регистра управления CAN узла NCRx), т. е. до начала работы CAN узла.

Примечание – Следует помнить, что регистр NBTRx может быть записан, только если бит CCE регистра NCRx установлен.

Процессор потока битов

Процессор потока битов формирует (на основе содержимого областей сообщений) сообщения данных и удаленные запросы непосредственно перед отправкой на шину CAN. Процессор потока управляет генератором CRC и добавляет контрольную сумму к сообщению. После вставки битов начала SOF и конца EOF сообщения, процессор потока начинает передачу сообщения по правилам арбитража шины CAN. В течение всего времени передачи сообщения процессор потока битов ведет мониторинг шины. Если обнаруживается несовпадение текущего (определяемого мониторингом) и ожидаемого (выдаваемого CAN узлом) уровня напряжения на шине, генерируется ошибка и соответствующий ей запрос на прерывание. Код возникшей ошибки отражается в битовом поле LEC регистра состояния CAN узла NSRx.

Корректность получаемых данных проверяется и подтверждается или не подтверждается кодом CRC. В случае не подтверждения возникает ошибка, генерируется запрос на прерывание, и код ошибки выставляется в регистре NSRx, кроме этого на шину выдается сообщение об ошибке.

После получения сообщения, не содержащего ошибок, и разбиения его на идентификатор и пакет данных, полученная информация записывается в буфер блока обработки сообщений, формируется соответствующее прерывание, и обновляются регистры состояния.

Блок обработки ошибок

Блок обработки ошибок CAN узла предназначен для выявления ошибок в работе устройств узла. В составе блока есть два счетчика: счетчик ошибок приема REC и счетчик ошибок передачи TEC. Счетчикам соответствуют битовые поля REC и TEC регистра счетчика ошибок CAN узла NECNTx, см. рисунок 23.23, таблицу 23.13. Инкрементированием и декрементированием счетчиков управляет процессор потока битов. В зависимости от значений счетчиков ошибок, CAN узел может находиться в одном из трех состояний:

- активной ошибки;

- пассивной ошибки;
- отключен от шины.

Узел в состоянии «активной ошибки» присоединен к шине и посылает флаг активной ошибки при обнаружении ошибок.

Узел в состоянии «пассивной ошибки» не должен посылать флаг активной ошибки. Он подключен к шине, но при обнаружении ошибок посылает флаг пассивной ошибки. После передачи узел в состоянии «пассивной ошибки» будет ждать инициализации дальнейшей передачи.

Узел в состоянии «отключения от шины» не может работать с шиной (выходные передатчики отключены).

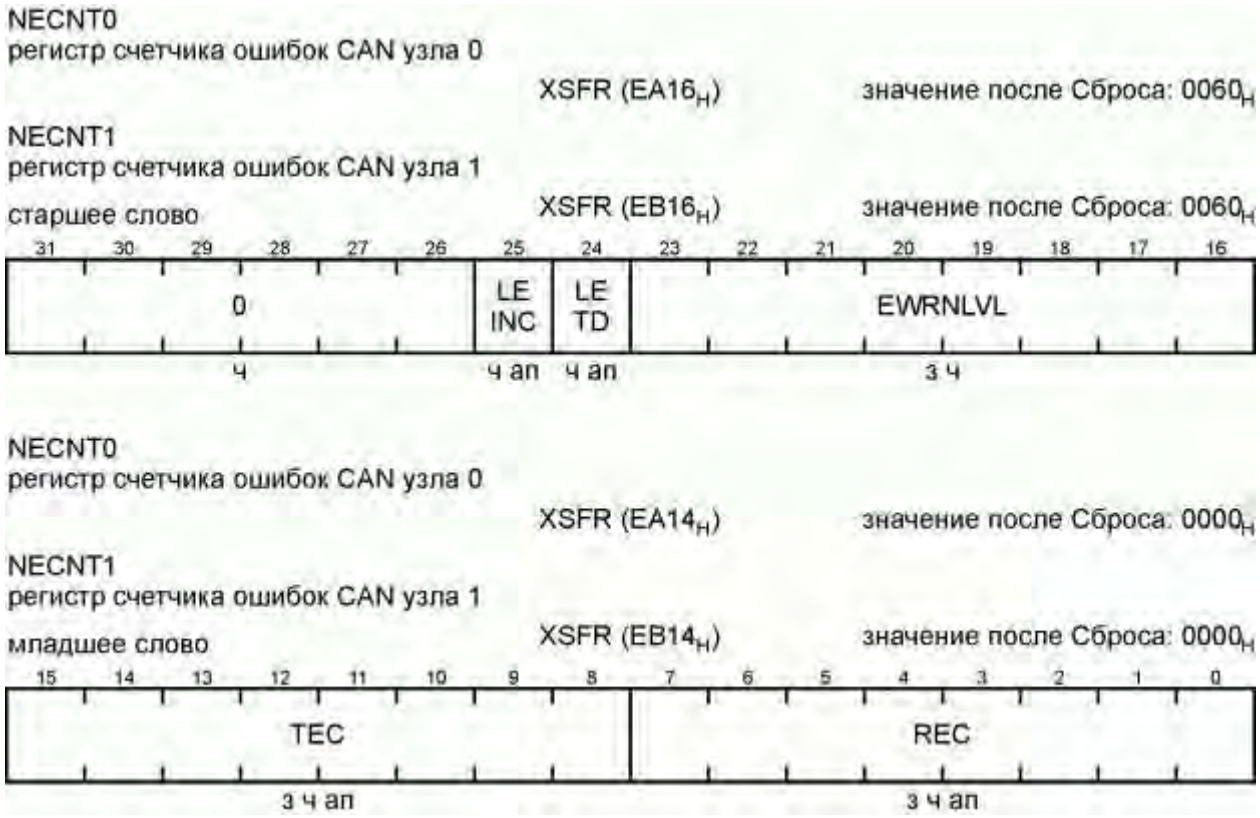


Рисунок 23.23 – Формат регистров NECNT0 и NECNT1 при разных значениях после сброса

Таблица 23.13 – Функциональное назначение полей регистров NECNT0 и NECNT1

Поле	Биты	Тип	Описание
1	2	3	4
REC	7-0	Чтение Запись Аппаратное влияние	Счетчик ошибок приема. Поле хранит значение счетчика ошибок приема сообщений CAN узла.
TEC	15-8	Чтение Запись Аппаратное влияние	Счетчик ошибок передачи. Поле хранит значение счетчика ошибок передачи сообщений CAN узла.

Окончание таблицы 23.13

1	2	3	4
EWRNLVL	23-16	Чтение Запись	Промежуточный лимит ошибок. Поле определяет лимит ошибок (по умолчанию – 96), по достижении которого выставляется соответствующий флаг ошибки EWRN.
LETD	24	Чтение Аппаратное влияние	Последняя ошибка передачи: 0 Последняя ошибка была обнаружена при приеме сообщения (с инкрементированием REC). 1 Последняя ошибка была обнаружена при передаче сообщения (с инкрементированием TEC).
LEINC	25	Чтение Аппаратное влияние	Инкрементирование при последней ошибке: 0 Последняя ошибка приводит к инкрементированию счетчика ошибок на 1. 1 Последняя ошибка приводит к инкрементированию счетчика ошибок на 8.
0	31-26	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Существует 5 различных типов ошибок (не взаимоисключающих).

Разрядная ошибка или ошибка бита BIT ERROR

Разрядная ошибка или ошибка бита BIT ERROR – узел, который передает данные на шину, осуществляет мониторинг шины. Ошибка бита имеет место, если значение бита на шине отличается от переданного значения. Исключение – посылка рецессивного бита в течение битового потока поля арбитража или поля подтверждения. Передатчик, посылающий флаг пассивной ошибки и обнаруживший доминантный бит, не интерпретирует это как ошибку бита.

Ошибка стаффинга (заполнения) STUFF ERROR

Ошибка стаффинга (заполнения) STUFF ERROR может быть обнаружена во время передачи шестого бита из последовательности шести одинаковых бит в поле сообщения, которое должно быть кодировано методом разрядного заполнения. Метод разрядного заполнения заключается в том, что после передачи 5 бит одинаковой полярности, шестой бит должен иметь противоположную полярность и вставляться передатчиком в поток данных автоматически, приемник пропускает этот бит.

Ошибка циклического избыточного кода CRC ERROR

Ошибка циклического избыточного кода CRC ERROR передатчик по установленному алгоритму, вычисляет значение контрольной суммы (собственно CRC) для передаваемых данных и вставляет ее в сообщение. Приемник после получения данных вычисляет CRC по тому же алгоритму, что и передатчик и сравнивает вычисленное значение с принятым значением CRC. В случае несовпадения возникает ошибка.

Ошибка формы FORM ERROR

Ошибка формы FORM ERROR обнаруживается, если:

- в битовом поле фиксированного формата содержится количество битов, отличающееся от установленного количества;

- на месте рецессивного бита находится доминантный бит или наоборот.

Примечание – Для приемника доминантный бит в течение последнего бита поля «конец кадра» не интерпретируется как ошибка формы.

Ошибка подтверждения ACKNOWLEGMENT ERROR

Ошибка подтверждения ACKNOWLEGMENT ERROR обнаруживается передатчиком всякий раз, когда он не обнаруживает доминантный бит ACK в «области подтверждения».

Всякий раз, когда каким-либо узлом обнаружена ошибка бита, ошибка стаффинга, ошибка формы или ошибка подтверждения, со следующего бита начинается передача флага ошибки соответствующим узлом.

Всякий раз, когда обнаружена ошибка CRC, передача флага ошибки начинается с бита, следующего после разделителя подтверждения, если не передается флаг ошибки другого условия.

Как уже было сказано выше, имеются два счетчика ошибок.

Эти счетчики функционируют согласно перечисленным ниже правилам (при передаче сообщения может применяться более одного правила одновременно).

1 Когда принимающий узел обнаруживает ошибку, счетчик ошибок приема увеличивается на единицу, за исключением разрядной ошибки во время передачи флага активной ошибки или флага перезагрузки.

2 Когда принимающий узел обнаруживает появление доминантного бита в качестве первого после передачи флага ошибки, счетчик ошибок приема увеличивается на восемь.

3 Когда передающий узел посылает флаг ошибки, счетчик ошибок передачи увеличивается на восемь.

Исключения, при которых значение счетчика ошибок передачи не изменяется:

- узел находится в состоянии «пассивной ошибки» и обнаруживает ошибку подтверждения (не обнаружен доминантный бит ACK) в течение передачи флага пассивной ошибки;

- передающий узел посылает флаг ошибки вследствие обнаружения ошибки стаффинга во время арбитража из-за того, что бит, находящийся перед битом RTR, должен быть рецессивным, был передан как рецессивный, но детектируется как доминантный.

4 Если передающий узел обнаруживает разрядную ошибку при передаче флага активной ошибки или флага перезагрузки, счетчик ошибок передачи увеличивается на восемь.

5 Если принимающий узел обнаруживает разрядную ошибку во время передачи флага активной ошибки или флага перезагрузки, счетчик ошибок приема увеличивается на восемь.

6 Любой узел сети допускает до семи последовательных доминантных бит после передачи флага активной ошибки, флага пассивной ошибки или флага перезагрузки.

Счетчик ошибок передачи увеличивается на восемь (в случае передачи) и счетчик ошибок приема увеличивается на восемь (в случае приема), если:

- обнаружена последовательность из 14 доминантных битов (в случае флага активной ошибки или флага перезагрузки);

- обнаружено восемь доминантных битов вслед за флагом пассивной ошибки;

- обнаружена последовательность (любая) из восьми доминантных битов при передаче.

7 После успешной передачи сообщения (получение бита ACK и отсутствие ошибок), счетчик ошибок передачи уменьшается на единицу, если его значение не равно нулю.

8 После успешного приема сообщения (прием без ошибок, соответствующее значение поля ACK Slot и отправка бита ACK), счетчик ошибок приема уменьшается на 1, если его значение было между единицей и 127. Если в счетчике ошибок приема ноль, счетчик остается без изменений. Если значение счетчика ошибок приема больше чем 127, он примет значение между 119 и 127.

9 Узел находится в состоянии «пассивной ошибки», когда один из счетчиков (ошибок передачи/приема) больше или равен 128. Возникновение ошибки, вследствие чего узел принял состояние «пассивной ошибки», является причиной того, что узел передает флаг активной ошибки.

10 Узел переходит в состояние «отключен от шины», если значение счетчика ошибок передачи равно или больше 256.

11 Узел, находившийся в состоянии «пассивной ошибки», снова переходит в состояние активной ошибки, если оба счетчика (ошибок передачи/приема) меньше или равны 127.

12 Узлу, который находится в состоянии «отключен от шины», разрешается перейти в состояние «активной ошибки», с установкой обоих счетчиков в «0», после того, как на шине будут обнаружены 128 последовательностей из 11 рецессивных битов.

Примечания

1 Величина счетчика ошибки большая, чем 96, указывает на серьезные нарушения на шине. Это можно использовать для проверки состояния шины.

2 Возможна ситуация, когда после запуска узла, он окажется единственным на шине (других узлов нет или они отключены от шины). В этом случае, если узел начнет передачу, то по ее окончании он не получит подтверждения. Это будет воспринято как ошибка передачи, и сообщение будет передано заново. Вследствие возникновения такой ситуации узел может перейти в состояние «пассивной ошибки», но не может перейти в состояние «отключен от шины».

О том, что CAN узел находится в состоянии «отключен от шины» сигнализирует флаг BOFF регистра NSRx.

Флаг EWRN регистра NSRx устанавливается, когда хотя бы один из счетчиков достиг или превысил лимит ошибок, определенный в битовом поле EWRNLVL регистра NECNTx. Как только значения обоих счетчиков не будут превышать лимит ошибок, флаг EWRN сбросится.

Счетчик сообщений

Каждый CAN узел имеет в своем составе 16-разрядный счетчик сообщений/синхросчетчик для подсчета количества принятых и переданных сообщений. Счетчик управляется регистром счетчика сообщений CAN узла NFCRx, см. рисунок 23.24, таблицу 23.14.

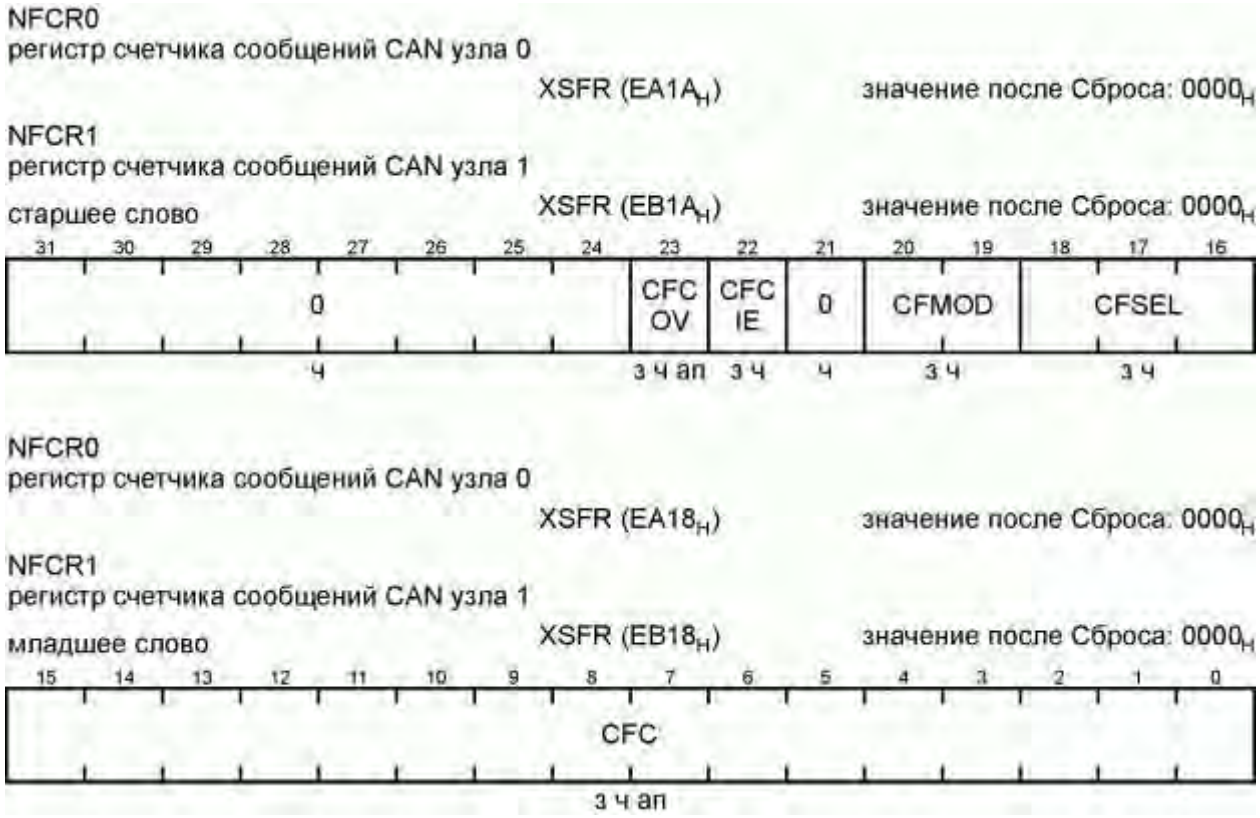


Рисунок 23.24 – Формат регистров NFCR0 и NFCR1

Таблица 23.14 – Функциональное назначение полей регистров NFCR0 и NFCR1

Поле	Биты	Тип	Описание
1	2	3	4
CFC	15-0	Чтение Запись Аппаратное влияние	Счетчик сообщений: - в режиме счетчика сообщений CFMOD = 00_B CFC хранит значение счетчика сообщений; - в режиме подсчета битов CFMOD = 01_B CFC хранит значение счетчика, который инкрементируется с началом очередного бита, начиная от начала передачи сообщения; - в режиме синхросчетчика CFMOD = 10_B CFC всегда хранит количество тактов частоты f_{CLC} (результат замера) минус один. Например, если значение CFC равно 34 в режиме измерения CFSEL = 000_B, то это означает, что между появлениями передних фронтов двух доминантных битов (при приеме сообщения) прошло 35 тактов частоты f_{CLC}.
CFSEL	18-16	Чтение Запись	Выбор объекта счета. Режим счетчика сообщений: - если CFSEL.0 установлен, CFC инкрементируется каждый раз при получении сообщения, не имеющего соответствующей области сообщения; - если CFSEL.1 установлен, CFC инкрементируется каждый раз при получении сообщения, имеющего соответствующую область сообщения; - если CFSEL.2 установлен, CFC инкрементируется после каждой успешной отправки сообщения. Режим подсчета битов: 000 Счетчик сообщений инкрементируется каждый раз с началом очередного бита, начиная от начала передачи сообщения. Значение счетчика сохраняется в CFC. Режим синхросчетчика. Доступные режимы сведены в таблицу 23.15. Если CFCIE установлен, то при изменении CFC генерируются прерывания соответствующего CAN узла.
CFMOD	20-19	Чтение Запись	Режим счетчика сообщений: 00 Режим счетчика сообщений. Счетчик сообщений инкрементируется после каждого успешного приема/передачи сообщения. 01 Режим подсчета битов. 10 Режим синхросчетчика. Счетчик сообщений используется для анализа синхронизации битов.

Окончание таблицы 23.14

1	2	3	4
CFCIE	22	Чтение Запись	Разрешение прерывания счетчика сообщений. Бит разрешения генерирования прерывания счетчика сообщения при его переполнении. 0 Генерирование прерывания запрещено. 1 Генерирование прерывания разрешено. Линия прерываний (одна из 16) для прерывания счетчика сообщений выбирается битовым полем CFCINP регистра NIPRx.
CFCOV	23	Чтение Запись Аппаратное влияние	Флаг переполнения счетчика сообщений. Устанавливается при переполнении счетчика сообщений. 0 Нет переполнения счетчика. 1 Счетчик переполнился. В режиме синхросчетчика, CFCOV устанавливается при изменении CFC. Если разрешено CFCIE = 1_B, при переполнении счетчика генерируется прерывание. Флаг CFCOV сбрасывается программно.
0	21, 31-24	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Таблица 23.15 – Режимы синхросчетчика

CFSEL	Описание
000_B	Всякий раз, когда доминантный фронт (перепад сигнала из «1» в «0») обнаруживается на принимающем входе, время, измеренное в тактах частоты f_{CLC} между этим фронтом и новым доминантным фронтом, сохраняется в CFC.
001_B	Всякий раз, когда рецессивный фронт (перепад сигнала из «0» в «1») обнаруживается на принимающем входе, время, измеренное в тактах частоты f_{CLC} между этим фронтом и новым рецессивным фронтом, сохраняется в CFC.
010_B	Всякий раз, когда принимается доминантный фронт как результат переданного доминантного фронта, время, измеренное в тактах частоты f_{CLC} между этими фронтами, сохраняется в CFC.
011_B	Всякий раз, когда принимается рецессивный фронт как результат переданного рецессивного фронта, время, измеренное в тактах частоты f_{CLC} между этими фронтами, сохраняется в CFC.
100_B	Всякий раз, когда на принимающем входе обнаруживается доминантный фронт, предназначенный для синхронизации, время, измеренное в тактах частоты f_{CLC} между этим фронтом и новой точкой выборки, сохраняется в CFC.
101_B	Всякий раз в момент точки выборки время, измеренное в тактах частоты f_{CLC} между началом нового бита и началом предыдущего бита, сохраняется в CFC.11 – CFC.0. Также в момент точки выборки в CFC.15 – CFC.12 сохраняется дополнительная информация: - CFC.15: передаваемое значение текущего времени бита; - CFC.14: принимаемое значение выборки текущего времени бита; - CFC.13, CFC.12: информация CAN шины подробно – в таблице 23.16.
110_B	Зарезервировано
111_B	Зарезервировано

Таблица 23.16 – Информация о состоянии CAN шины

CFC.13, CFC.12	Состояние шины
00_B	Нет бита. CAN шина находится в состоянии простоя, осуществляет (де-) стаффинг бита или имеет место один из сегментов: SOF, SRR, CRC, разделители, первые 6 битов EOF, IFS.
01_B	Новый бит. Этот код представляет первый бит нового сегмента сообщения. Текущий бит является первым битом одного из следующих сегментов: 10-й бит стандартного идентификатора (только при передаче), RTR, зарезервированные биты, IDE, DLC, седьмой бит каждого байта данных и первый бит расширения идентификатора ID.
10_B	Бит. Этот код представляет бит (не первый бит сегмента, который определяется как «Новый» бит) в пределах сегмента (длина сегмента должна быть более одного бита) сообщения. Текущий бит является битом одного из следующих сегментов: идентификатор (за исключением первого бита стандартного идентификатора и первого бита расширения идентификатора), DLC (3 LSB) и биты с 6 по 0 каждого байта данных.
11_B	Завершающий бит. Текущий бит является битом одного из следующих сегментов: бит подтверждения, последний бит EOF, сообщение об активной/пассивной ошибке, сообщение о перезагрузке. Два или более следующих друг за другом «завершающих» бита формируют сигнал ошибки сообщения.

Битовое поле CFSEL регистра NFCRx определяет один из трех режимов работы счетчика.

Режим подсчета сообщений

После успешной передачи и/или успешного приема сообщения, содержимое счетчика копируется в битовое поле CFCVAL регистра MOIPRn (регистр указателя прерываний области сообщения n) области сообщения, участвующей в пересылке данных. После чего счетчик сообщений инкрементируется.

Режим подсчета битов

Счетчик инкрементируется с началом очередного бита. С началом передачи/приема сообщения значение счетчика захватывается и сохраняется в битовом поле CFC регистра NFCRx. После окончания передачи/приема сообщения сохраненное значение копируется в битовое поле CFCVAL регистра MOIPRn области сообщения, участвующей в пересылке данных.

Режим синхросчетчика

Используется для отслеживания скорости потока битов и анализа синхронизации битов.

Прерывания CAN узла

Каждый CAN узел имеет четыре типа прерываний, каждое от своего источника, см. рисунок 23.25. Источники прерываний:

- успешная передача или прием сообщения;
- код последней ошибки;
- ALERT – состояние, возникающее в случаях:

а) когда хотя бы один из счетчиков ошибок CAN узла достигает значения своего лимита;

б) изменяется состояние «отключен от шины»;

в) возникает ошибка длины списка или ошибка списка областей;

- переполнение счетчика сообщений.

В дополнение к аппаратным прерываниям, есть возможность программного генерирования прерываний с использованием регистра прерываний MITR. Запись «1» в n-й разряд битового поля IT регистра MITR генерирует сигнал запроса прерывания на соответствующей ему n-ой линии прерываний. Запись в регистр MITR с установкой нескольких битов в битовом поле IT приводит к параллельному генерированию запросов прерываний на соответствующих установленным битам линиях прерываний.

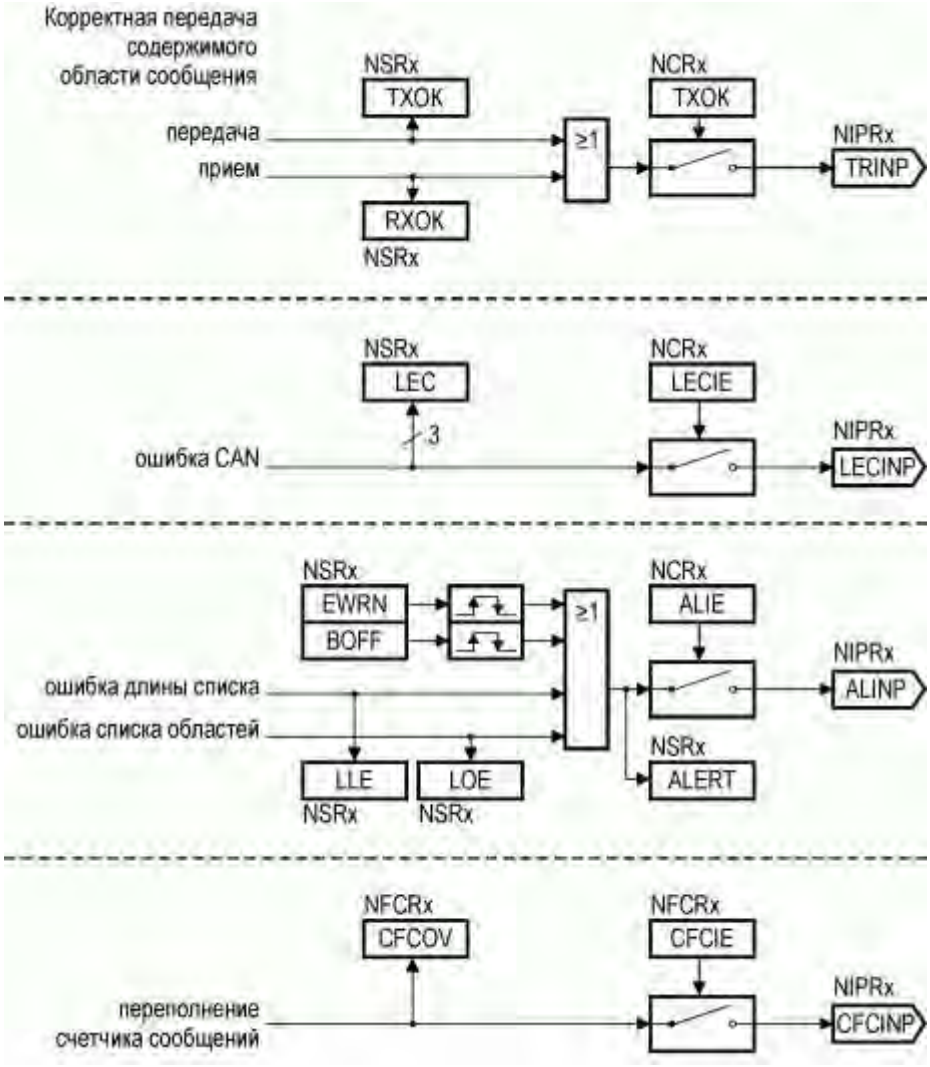
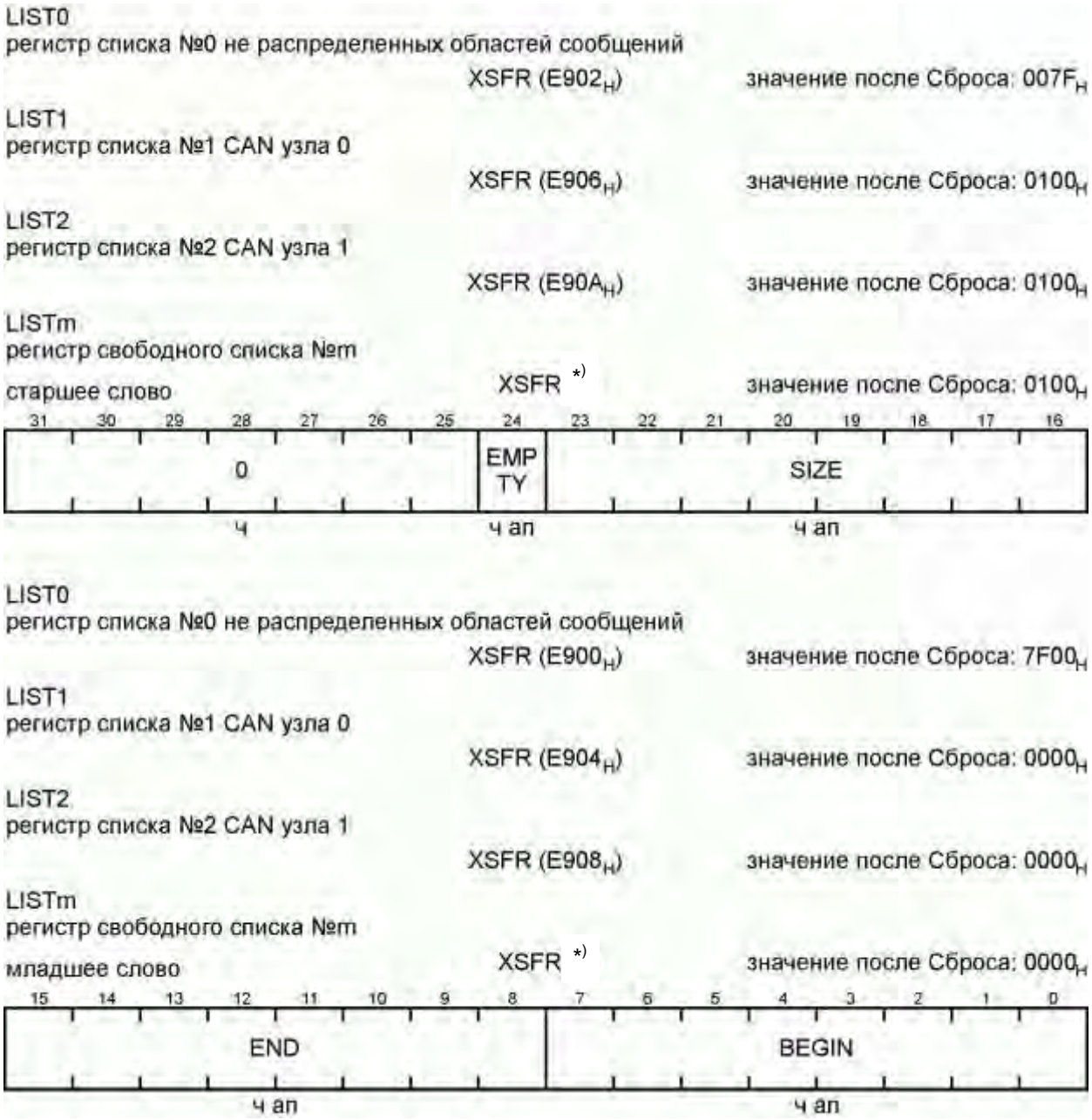


Рисунок 23.25 – Прерывания CAN узла

Структура списка областей сообщений

Области сообщений модуля CAN могут быть организованы в восемь двухсвязных списков. Каждая область сообщения может быть отнесена к одному из списков, каждый CAN узел имеет свой список и соответствующий регистр списка: LIST1 – для списка № 1 узла 0, LIST2 – для списка № 2 узла 1, и имеет указатели на предшествующую ей и следующую за ней области. Не распределенные между CAN узлами области сообщений организуются в отдельный список № 0, который управляется регистром LIST0. Остальные пять списков, от списка № 3 до списка № 7, являются свободными и управляются регистрами LIST3 – LIST7, смотри рисунок 23.26, таблицу 23.17. Адреса регистров LISTm свободных списков, где m = 3, …, 7 представлены в таблице 23.18.

.



*) Адреса – в таблице 23.18.

Рисунок 23.26 – Формат регистров LISTm, где m = 0, …, 7

Таблица 23.17 – Функциональное назначение полей регистров LISTm, где m = 0, …, 7

Поле	Биты	Тип	Описание
1	2	3	4
BEGIN	7-0	Чтение Аппаратное влияние	Номер первого элемента списка. Номер области сообщения, расположенной в начале списка.
END	15-8	Чтение Аппаратное влияние	Номер последнего элемента списка. Номер области сообщения, расположенной в конце списка.

Окончание таблицы 23.17

1	2	3	4
SIZE	23-16	Чтение Аппаратное влияние	Размер списка. Количество областей сообщений, занесенных в список Значение SIZE на единицу меньше количества элементов списка. Если список пуст, значение SIZE равно «0_H».
EMPTY	24	Чтение Аппаратное влияние	Индикатор пустого списка. Бит EMPTY является индикатором того, что список пуст. 0 По крайней мере, один элемент присутствует в списке. 1 Список пуст.
0	31-25	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Таблица 23.18 – Адреса регистров LISTm свободных списков, где m = 3, …, 7

m	Регистр		Адрес
3	Регистр свободного списка № 3 LIST3	Младшее слово	$E90C_H$
		Старшее слово	$E90E_H$
4	Регистр свободного списка № 4 LIST4	Младшее слово	$E910_H$
		Старшее слово	$E912_H$
5	Регистр свободного списка № 5 LIST5	Младшее слово	$E914_H$
		Старшее слово	$E916_H$
6	Регистр свободного списка № 6 LIST6	Младшее слово	$E918_H$
		Старшее слово	$E91A_H$
7	Регистр свободного списка № 7 LIST7	Младшее слово	$E91C_H$
		Старшее слово	$E91E_H$

На рисунке 23.27 представлен вариант, когда области сообщений с номерами 3, 5 и 16 занесены в список № 2, принадлежащий CAN узлу 1 (соответствующий регистр списка – LIST2).

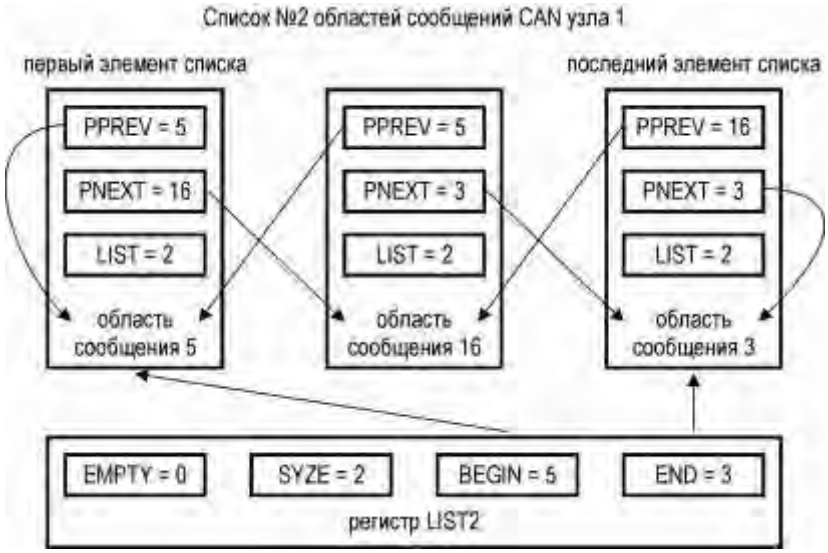


Рисунок 23.27 – Пример списка областей сообщений

Значение битового поля BEGIN регистра LIST(x+1) указывает на первый элемент списка (к примеру, на рисунке 23.27 это область сообщения 5). Значение битового поля END указывает на последний элемент списка, на рисунке 23.27 это область сообщения 3. Количество элементов списка (количество областей сообщений в списке) отражается в битовом поле SIZE (значение SIZE на единицу меньше количества элементов списка – так в нашем случае на рисунке 23.27, SYZE = 2_H для трех областей сообщений). Бит EMPTY является индикатором заполнения списка. Если EMPTY = 1_B, значит лист пустой (в случае на рисунке 23.27, EMPTY = 0_B, потому что список не пустой).

Каждая область сообщения n имеет указатель (битовое поле PNEXT регистра состояния области сообщения n MOSTATn) на следующую по списку область сообщения, и указатель (битовое поле PPREV регистра состояния области сообщения MOSTATn) на предыдущую по списку область сообщения.

PPREV первой по списку области сообщения указывает на эту же область, в примере на рисунке 23.27 область сообщения 5 является первой в списке и поэтому ее указатель PPREV = 5_H.

PNEXT последней по списку области сообщения указывает на эту же область, в примере на рисунке 23.27 область сообщения 3 является последней в списке и поэтому ее указатель PNEXT = 3_H.

Битовое поле LIST регистра MOSTATn указывает номер списка, к которому относится область сообщения n. Для случая, рассмотренного на рисунке 23.27, значение LIST всех трех областей сообщений будет равно 2.

Если битовому полю LIST регистра MOSTATn области сообщения n присвоено значение 0, то эта область относится к списку № 0 нераспределенных областей сообщений.

После сброса все области сообщений считаются нераспределенными и битовые поля LIST, в соответствующих им регистрах, равны 0. По умолчанию, порядок элементов списка № 0 следующий: область сообщений (n – 1) является предыдущей области сообщения n, а область сообщения (n + 1) – последующей.

Если предположить, что все области сообщения занесены в один список в порядке возрастания их номеров, то значения битовых полей PNEXT и PPREV регистров MOSTATn будут соответствовать указанным значениям в таблице 23.19.

Таблица 23.19 – Пояснительная таблица к описанию регистра MOSTATn

Область сообщения	PNEXT	PPREV	Значение после сброса
0	1_H	0_H	$0100\ 0000_H$
1	2_H	0_H	$0200\ 0000_H$
2	3_H	1_H	$0301\ 0000_H$
3	4_H	2_H	$0402\ 0000_H$
4	5_H	3_H	$0503\ 0000_H$
5	6_H	4_H	$0604\ 0000_H$
6	7_H	5_H	$0705\ 0000_H$
7	8_H	6_H	$0806\ 0000_H$
8	9_H	7_H	$0907\ 0000_H$
9	10_H	8_H	$0A08\ 0000_H$
10	11_H	9_H	$0B09\ 0000_H$
11	12_H	10_H	$0C0A\ 0000_H$
12	13_H	11_H	$0D0B\ 0000_H$
13	14_H	12_H	$0E0C\ 0000_H$
14	15_H	13_H	$0F0D\ 0000_H$
15	16_H	14_H	$100E\ 0000_H$
16	17_H	15_H	$111F\ 0000_H$
17	18_H	16_H	$1210\ 0000_H$
18	19_H	17_H	$1311\ 0000_H$
19	20_H	18_H	$1412\ 0000_H$
20	21_H	19_H	$1513\ 0000_H$
21	22_H	20_H	$1614\ 0000_H$
22	23_H	21_H	$1715\ 0000_H$
23	24_H	22_H	$1816\ 0000_H$
24	25_H	23_H	$1917\ 0000_H$
25	26_H	24_H	$1A18\ 0000_H$
26	27_H	25_H	$1B19\ 0000_H$
27	28_H	26_H	$1C1A\ 0000_H$
28	29_H	27_H	$1D1B\ 0000_H$
29	30_H	28_H	$1E1C\ 0000_H$
30	31_H	29_H	$1F1D\ 0000_H$
31	31_H	30_H	$201E\ 0000_H$

Подключение списков областей сообщений

CAN узел может оперировать только с теми областями сообщений, которые занесены в принадлежащий ему список. Так CAN узлу 0 принадлежит список № 1, а CAN узлу 1 принадлежит список № 2.

В модуле CAN имеется восемь списков, см. рисунок 23.28:

- список № 0 нераспределенных сообщений;
- список № 1 CAN узла 0;
- список № 2 CAN узла 1;
- списки № 3 – № 7 – свободные списки, не принадлежащие ни одному из двух CAN узлов.

Области сообщений, распределенные в списки № 3 – № 7, не могут быть использованы CAN узлами.

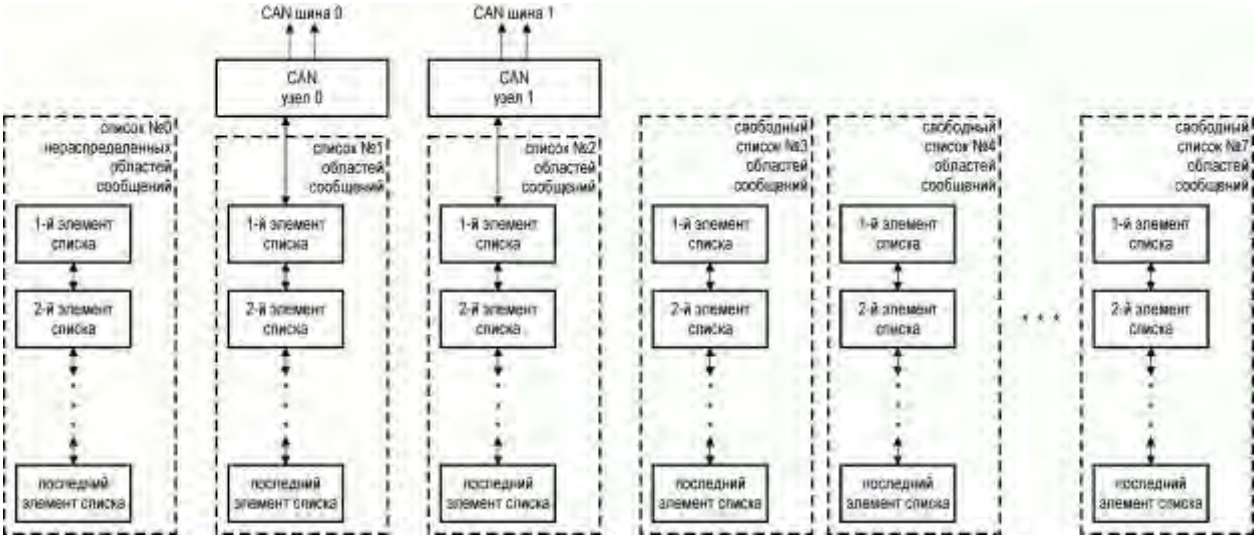


Рисунок 23.28 – Списки модуля CAN

Механизмы FIFO и шлюза оперируют с областями сообщений, независимо от их распределения по спискам, что, ко всему прочему, дает возможность работы со всеми восемью списками. Следовательно, при использовании механизмов FIFO и шлюза следует следить за тем, действительно ли используемые области сообщений принадлежат желаемым спискам.

Панель команд списка

Для просмотра структуры списка областей сообщений CAN узла достаточно обратиться к соответствующим регистрам LIST(x+1) и MOSTATn.

Структура списка управляется и изменяется посредством контроллера списка, который, в свою очередь, управляется панелью команд, основное назначение которой – упрощение внесения изменений в структуру списка, отслеживание этих изменений и проверка их корректности.

Панель команд имеет регистр PANCTR, см. рисунок 23.29, таблицу 23.20, полностью определяющий действия контроллера списка по построению и/или реорганизации списков областей сообщений.

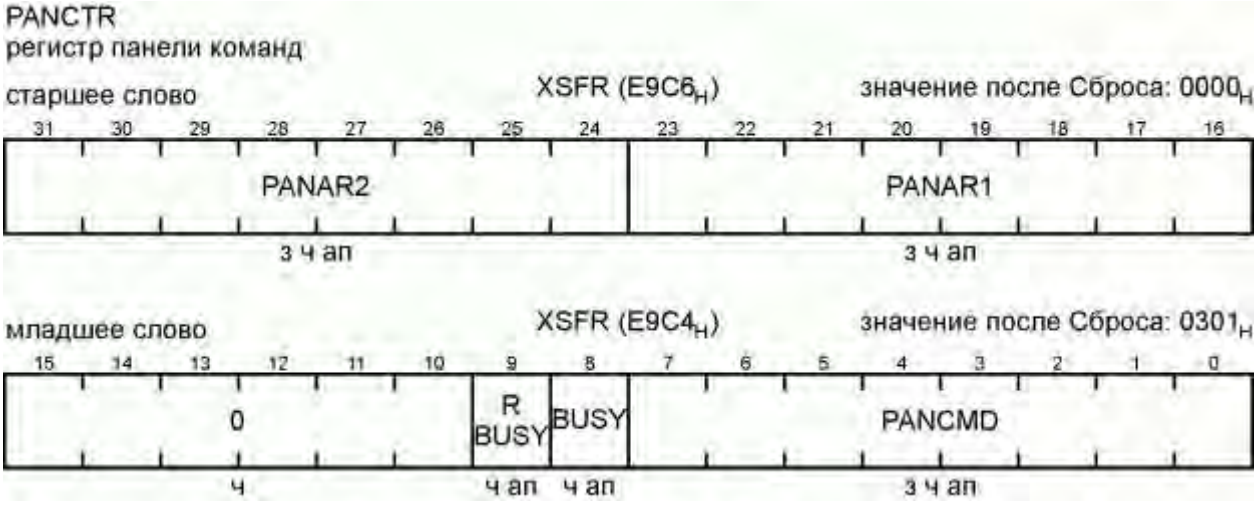


Рисунок 23.29 – Формат регистра PANCTR

Таблица 23.20 – Функциональное назначение полей регистра PANCTR

Поле	Биты	Тип	Описание
PANCMD	7–0	Чтение Запись Аппаратное влияние	Панель команд. Это битовое поле предназначено для записи кодов команд. После выполнения команды код «нет операции» автоматически записывается в PANCMD. Допустимые коды команд сведены в таблицу 23.21.
BUSY	8	Чтение Аппаратное влияние	Флаг занятости панели: 0 Панель готова для записи команды. 1 Панель занята, выполняется команда.
RBUSY	9	Чтение Аппаратное влияние	Флаг занятости панелей аргументов: 0 Изменение битовых полей PANAR1 и PANAR2 контроллером списка не планируется. 1 Выполняется команда списка BUSY = 1_B, результаты выполнения которой будут записаны в битовые поля PANAR1 и PANAR2, но пока не доступны.
PANAR1	23–16	Чтение Запись Аппаратное влияние	Панель аргумента 1. Описание в таблице 23.21.
PANAR2	31–24	Чтение Запись Аппаратное влияние	Панель аргумента 2. Описание в таблице 23.21.
0	15–10	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Операция панели состоит из кода команды PANCMD и двух панелей аргументов PANAR1 и PANAR2. Допустимые коды команды PANCMD представлены в таблице 23.21. Команды, которые имеют возвращаемое значение, записывают это значение в битовое поле PANAR1. Команды, которые возвращают флаг ошибки, записывают его в 31-й бит регистра PANCTR (7-й бит битового поля PANAR2).

Таблица 23.21 – Допустимые коды команды PANCMD

PANCMD	PANAR2	PANAR1	Описание команды
1	2	3	4
00_H	–	–	Нет операции. Запись 00_H безрезультатна. Никаких действий не предпринимается.

Продолжение таблицы 23.21

1	2	3	4
01_H	Результат: 7-й бит – ошибка, 6-й бит – не определен	–	Инициализация списков. Запуск инициализации для очистки битовых полей CTRL и LIST всех областей сообщений. Регистры списков LIST0 – LIST8 устанавливаются в свои значения после сброса. Это приводит к переносу всех областей сообщений в список № 0 нераспределенных областей сообщений. Инициализация списков требует, чтобы биты INIT и CCE регистра NCRx были установлены для обоих CAN узлов. 7-й бит битового поля PANAR2 сигнализирует о результате операции: 0 Инициализация завершена успешно. 1 Не все биты INIT и CCE были установлены. Инициализация не завершена. Команда инициализации списков автоматически запускается при каждом сбросе модуля CAN, за исключением случая, когда все регистры областей сообщений уже сброшены.
02_H	Аргумент: номер списка	Аргумент: номер области сообщения	Статическое распределение области сообщения. Независимо от того, какому списку принадлежала область сообщения, она переносится в указанный список (битовое поле PANAR2) и добавляется в его конец. Эта команда так же используется для дераспределения (перенос области сообщения в список № 0 нераспределенных областей сообщений) области сообщения. Для этого значение в битовом поле PANAR2 должно быть равно 0_H.
03_H	Аргумент: номер списка Результат: 7-й бит – ошибка, 6-й бит – не определен	Результат: номер области сообщения	Динамическое распределение области сообщения. Занесение области сообщения, являющейся первым элементом списка №0 нераспределенных областей сообщений в указанный (битовое поле PANAR2) список. Область сообщения добавляется в конец списка. Номер (по списку) добавленной области сообщения возвращается в битовом поле PANAR1. 7-й бит битового поля PANAR2 сигнализирует о результате операции: 0 Занесение в список прошло успешно. 1 Команда не выполнена, поскольку список № 0 пуст.
04_H [1)]	Аргумент: новый номер области сообщения	Аргумент: текущий номер области сообщения	Перемещение до указанной позиции. Перенос области сообщения (номер области указывается в поле PANAR1) из текущей позиции на позицию, предшествующую занимаемой области сообщения, указанной в битовом поле PANAR2.

Окончание таблицы 23.21

1	2	3	4
05_H [2)]	Аргумент: новый номер области сообщения Результат: 7-й бит– ошибка, 6-й бит – не определен	Результат: номер добавленной области сообщения	Занесение в список до указанной позиции. Перенос области сообщения из списка № 0 (выбирается первый по списку элемент) на позицию, предшествующую занимаемой области сообщения, указанной в битовом поле PANAR2. Номер добавленной области сообщения возвращается в битовом поле PANAR1. 7-й бит битового поля PANAR2 сигнализирует о результате операции: 0 Занесение в список прошло успешно. 1 Команда не выполнена, поскольку список № 0 пуст.
06_H [1)]	Аргумент: новый номер области сообщения	Аргумент: текущий номер области сообщения	Перемещение на позицию, после указанной позиции. Перенос области сообщения (номер области указывается в поле PANAR1) из текущей позиции на позицию, следующую за занимаемой областью сообщения с номером, указанным в PANAR2.
07_H [2)]	Аргумент: новый номер области сообщения Результат: 7-й бит – ошибка, 6-й бит – не определен	Результат: номер добавленной области сообщения	Занесение в список на позицию, после указанной позиции. Перенос области сообщения из списка № 0 (выбирается первый по списку элемент) на позицию, следующую за занимаемой областью сообщения, с номером указанным в битовом поле PANAR2. Номер добавленной области сообщения возвращается в битовом поле PANAR1. 7-й бит битового поля PANAR2 сигнализирует о результате операции: 0 Занесение в список прошло успешно. 1 Команда не выполнена, поскольку список № 0 пуст.
08_H – FF_H	–	–	Зарезервировано.

[1)] Перемещаемая область сообщения и область сообщения до/после позиции которой происходит перемещение, могут принадлежать как одному списку, так и разным. В случае если области сообщений принадлежат разным спискам, то исключение элемента из одного списка и занесение в другой происходит автоматически. Для перемещения области сообщений достаточно знать ее номер (0, …, 31).

[2)] Для перемещения области сообщения из списка № 0 достаточно знать номер области сообщения, до/после позиции которой происходит перемещение. Занесение нового элемента в список, которому принадлежит область сообщения, до/после позиции которой происходит перемещение, осуществляется автоматически, с выдачей номера области сообщения (0, …, 31), заносимой в список.

Панель команд запускается записью соответствующей команды в битовое поле PANCMD регистра PANCTR. Соответствующие аргументы команд должны быть записаны в битовые поля PANAR1 и PANAR2 до записи кода команды в поле PANCMD.

Примечание – Запись новых значений в битовые поля PANAR1 и PANAR2 не изменяет сразу их содержимого. Новые значения, сначала, попадают в специальный теневой регистр. Далее, одновременно с записью кода команды в битовое поле PANCMD, новые значения, хранящиеся в теневом регистре, переносятся в битовые поля PANAR1 и PANAR2.

С записью корректного кода команды выставляется флаг BUSY регистра PANCTR и в дальнейшем все попытки записи в регистр PANCTR игнорируются. Флаг BUSY остается активным, а панель команд заблокированной до тех пор, пока не завершится выполнение записанной команды.

После сброса, контроллер списка формирует список № 0 нераспределенных областей сообщений. Во время этой операции флаг BUSY = 1_B и все обращения к ОЗУ областей сообщений запрещены. ОЗУ становиться доступным только после сброса флага BUSY.

Примечание – ОЗУ областей сообщений автоматически инициализируется после сброса контроллером списка для обеспечения каждой области сообщения корректным указателем на список. По окончании этой операции флаг BUSY сбрасывается.

В случае появления команды динамического распределения, по которой какой-либо элемент забирается из списка № 0 и переноситься в другой указанный список, наряду с битом BUSY устанавливается бит RBUSY (RBUSY = BUSY = 1_B). Это указывает на то, что значения битовых полей PANAR1 и PANAR2 будут обновлены контроллером списка, следующим образом:

- номер области сообщения, забираемой из списка № 0 нераспределенных областей сообщений, записывается в PANAR1;

- если бит ERR = 1_B (7-й бит поля PANAR2), значит, список № 0 пуст и выполнение команды завершается; если бит ERR = 0_B, значит, список № 0 не пуст и команда выполняется.

Результаты выполнения команды динамического распределения записываются до того, как контроллер списка начнет процесс распределения. Как только результаты станут доступны, бит RBUSY снова становиться равным «0» (т. е. неактивным), в то время как бит BUSY остается активным (= 1_B) до завершения выполнения команды. Это позволяет пользователю запрограммировать настройки желаемой области сообщения, в то время как контроллер списка распределяет области. Во время операций со списками доступ к областям сообщений не ограничен, но следует помнить, что любой доступ к регистрам ОЗУ областей сообщений в течение процесса распределения областей вносит задержку (в процесс), равную длительности доступа.

По завершении процесса выполнения команды, флаг BUSY сбрасывается и запись в регистр PANCTR снова становиться возможной.

Код команды «нет операции» автоматически записывается в битовое поле PANCMD.

Новая команда может быть записана в любое время, когда бит BUSY = 0_B.

Все битовые поля регистра PANCTR, исключая биты BUSY и RBUSY, могут быть записаны программно, что делает возможным сохранять и восстанавливать значения регистра PANCTR, если панель команд используется независимой подпрограммой обработки прерываний. Если возникает такая ситуация, то любые задачи, которые используют панель команд и которые могут прерывать выполнение других задач, тоже использующих панель команд, будут опрашивать состояние флага BUSY. До тех пор, пока флаг BUSY будет оставаться активным, содержимое регистра PANCTR будет сохранено в соответствующей области памяти до операции восстановления. Как только подпрограмма обработки прерываний закончится, содержимое регистра PANCTR будет восстановлено.

До того, как область сообщения, занесенная в список активного CAN узла, будет перенесена на другую позицию этого же списка или перенесена в другой список, бит MSGVAL регистра MOSTATn области сообщения n должен быть очищен.

23.5 Анализ работы CAN узла

Доступны три режима анализа:
- режим общего анализа;
- режим внутренней петли;
- режим анализа синхронизации битов.

Режим общего анализа

Этот режим позволяет осуществлять независимый мониторинг работы CAN узла, не затрагивая CAN шину. Режим общего анализа выбирается битом CALM регистра NCRx.

В режиме общего анализа выходы CAN узла находятся в рецессивном состоянии. Узел может получать сообщения данных, сообщения удаленных запросов и сообщения об ошибках, но работа узла на передачу запрещена. Полученные сообщения данных/удаленных запросов остаются без подтверждения (бит подтверждения остается в рецессивном состоянии), но принимаются и сохраняются (при совпадении идентификаторов) в соответствующих областях сообщений.

Режим внутренней петли

Это режим позволяет проводить внутреннее тестирование модуля CAN, а также отладку управляющей программы без доступа к внешней CAN шине.

Внутренняя петля состоит из внутренней CAN шины (внутри модуля CAN) и переключателя выбора шины для каждого CAN узла, см. рисунок 23.30. С помощью переключателя каждый CAN узел может быть подключен либо к внутренней CAN шине (режим внутренней петли) или к внешней CAN шине (нормальный режим работы). Если выбран режим внутренней петли, то на внешнем передающем выводе CAN узла поддерживается рецессивный уровень сигнала, а состояние принимающего вывода игнорируется.

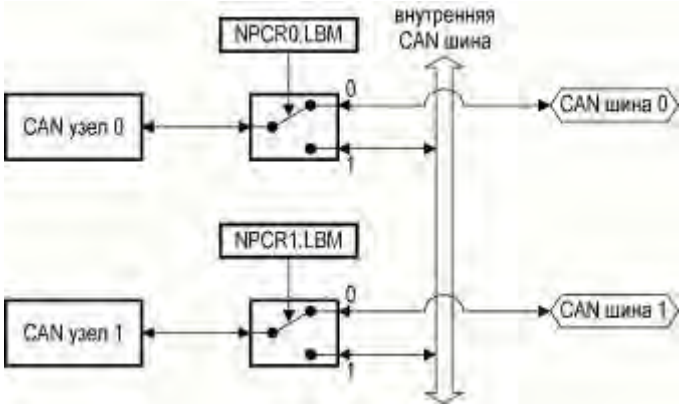


Рисунок 23.30 – Режим внутренней петли

Режим внутренней петли для CAN узла может быть активирован битом LBM регистра NPCRx.

Если оба CAN узла функционируют в режиме внутренней петли, они взаимодействуют друг с другом посредством внутренней CAN шины.

Режим анализа синхронизации битов

Детальный анализ синхронизации битов для CAN узла может быть осуществлен посредством режима анализа счетчика сообщений этого CAN узла. Анализ синхронизации при помощи счетчика сообщений может быть использован для автоматического детектирования потока данных, а также для временного анализа CAN шины.

Режим анализа синхронизации битов для CAN узла x выбирается, когда CFMOD = 10_B (регистр NFCRx). Режим анализа синхронизации битов не оказывает влияния на работу CAN узла.

Результаты измерений в режиме анализа синхронизации битов записываются в битовое поле CFC регистра NFCRx. Каждый раз, когда битовое поле CFC изменяется (в режиме анализа синхронизации битов), устанавливается флаг CFCOV регистра NFCRx. Если разрешено и флаг CFCIE регистра NFCRx установлен, может быть сгенерирован запрос на прерывание.

Автоматический контроль скорости потока битов

Для автоматического контроля скорости потока битов на CAN шине необходимо измерять время между появлениями фронтов передаваемых доминантных битов. Эти измерения осуществляются автоматически, если CFSEL = 000_B (регистр NFCRx). Время (измеренное в тактах частоты f_{CLC}) между появлениями фронтов двух доминантных битов сохраняется в битовом поле CFC.

Анализ синхронизации

Анализ синхронизации времени бита осуществляется, если CFSEL = 010_B (регистр NFCRx). Время между фронтом первого доминантного сигнала и точкой выборки измеряется и сохраняется в битовом поле CFC. Смещение синхронизации бита может быть получено из этого времени, как только появится первый фронт сигнала после точки выборки. Существует только одна синхронизация между последовательными точками выборки.

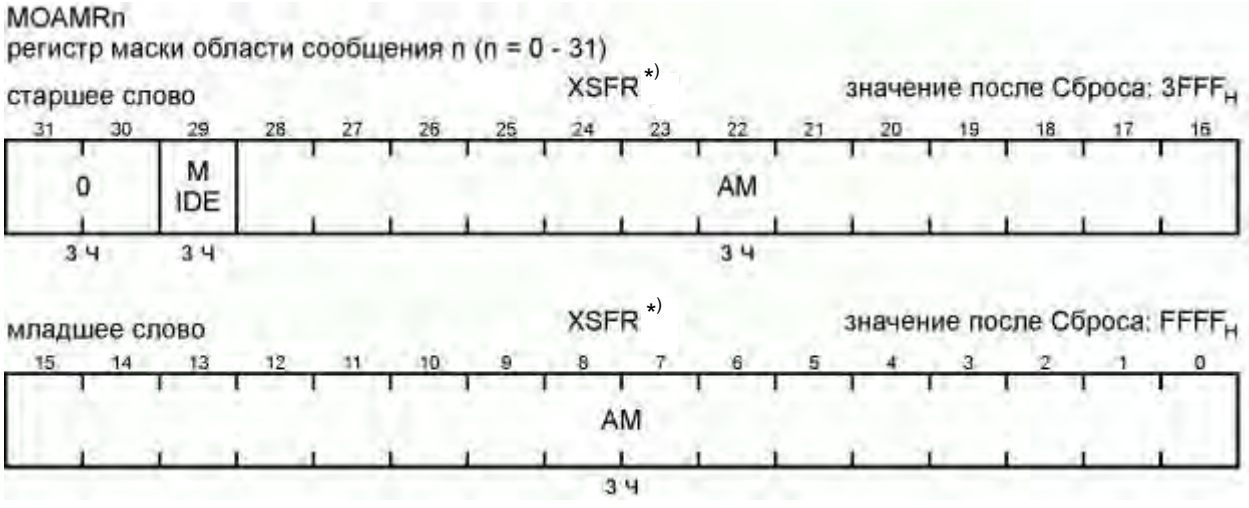
Анализ синхронизации, например, может быть использован во время приема первого сообщения для более точной настройки скорости потока битов.

Измерение задержки драйвера

Задержка между появлением на шине переднего фронта сигнала передаваемого бита и приемом этого фронта измеряется, когда CFSEL = 011_B (доминантный – доминантный) и CFSEL = 100_B (рецессивный – рецессивный). Эта задержка показывает, какое время необходимо для определения значения нового бита в зависимости от физической реализации CAN шины.

23.6 Фильтрация сообщений

Фильтрация используется в модуле CAN для контроля приема и передачи сообщений, см. рисунки 23.31, 23.32, таблицы 23.22, 23.23, 23.24.

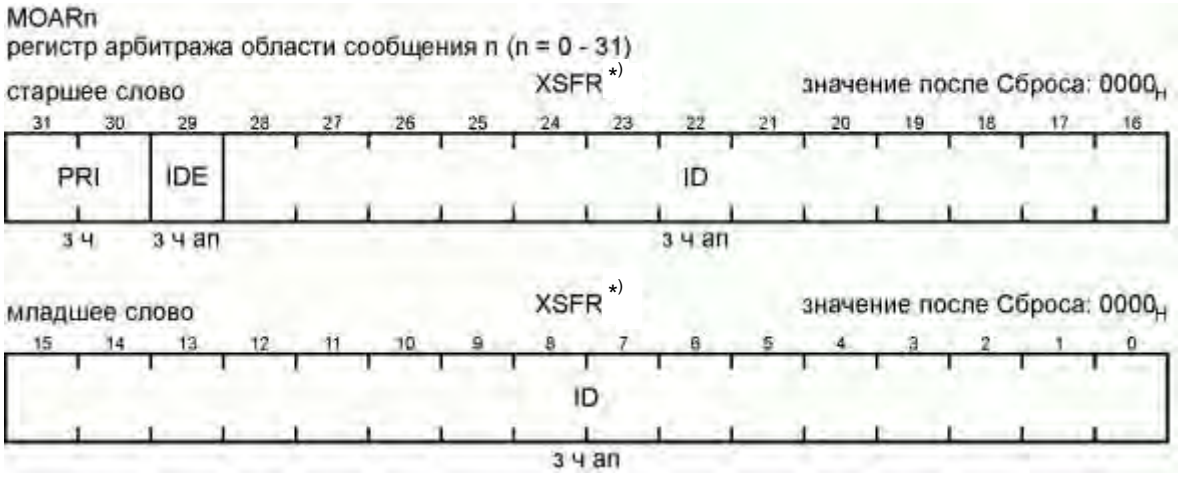


*) Адреса – в таблице 23.40.

Рисунок 23.31 – Формат регистров MOAMRn, где n = 0 – 31

Таблица 23.22 – Функциональное назначение полей регистров MOAMRn, где n = 0 – 31

Поле	Биты	Тип	Описание
AM	28-0	Чтение Запись	Маска идентификатора. 29 битная маска идентификатора для фильтрации принимаемых сообщений. В случае приема стандартного идентификатора используется битовое поле AM.28 – AM.8 (при этом состояние битов AM.17 – AM.0 любое) В случае приема расширенного идентификатора используется битовое поле AM.28 – AM.0
MIDE	29	Чтение Запись	Маска бита IDE 0 Область сообщения n может получать как стандартные, так и расширенные сообщения. 1 Область сообщения n может получать только те сообщения, в которых IDE = 1_B.
0	31-30	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».



*) Адреса – в таблице 23.40.

Рисунок 23.32 – Формат регистров MOARn, где n = 0 – 31

Таблица 23.23 – Функциональное назначение полей регистра MOARn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
ID	28-0	Чтение Запись Аппаратное влияние	Идентификатор области сообщения. Битовое поле хранит идентификатор (стандартный или расширенный) сообщения. В случае стандартного идентификатора используются биты ID.28 – ID.18 (при этом состояние битов ID.17 – ID.0 любое). В случае расширенного идентификатора используются биты ID.28 – ID.0.
IDE	29	Чтение Запись Аппаратное влияние	Бит расширения идентификатора: 0 Область сообщения поддерживает стандартные сообщения с 11-битными идентификаторами. 1 Область сообщения поддерживает расширенные сообщения с 29-битными идентификаторами.

Окончание таблицы 23.23

1	2	3	4
PRI	31-30	Чтение Запись	Класс приоритета. Это битовое поле определяет один из четырех классов приоритета для области сообщения. Меньший номер класса, указанный в PRI, соответствует большему приоритету. В случае если несколько областей сообщений имеют один класс приоритета, тогда приоритет среди них определяется согласно значениям идентификаторов и позициям в списке. Помимо прочего, битовое поле PRI определяет метод фильтрации для передачи сообщений: 00 Зарезервировано. 01 Фильтрация на основе положения в списке. Область сообщения имеет приоритет для передачи/приема в случае, если выше по списку нет других областей сообщений, ждущих передачу MSGVAL & TXEN0 & TXEN1 = 1_B. 10 Фильтрация на основе значения идентификатора. Область сообщения имеет приоритет (для передачи/приема) в случае, если в списке нет других областей сообщений с более высоким приоритетом, определяемым полем «ID + IDE + DIR», согласно правилам арбитража CAN шины (подробнее в таблице 23.24). 11 Фильтрация на основе положения в списке (аналогично PRI = 01_B).

Таблица 23.24 – Распределение приоритетов областей сообщений на основе правил арбитража

Установка приоритетов областей сообщений A и B (A имеет больший приоритет, чем B)	Пояснение
1	2
A.MOAR.28 – A.MOAR.18 < B.MOAR.28 – B.MOAR.18 (11-битный стандартный идентификатор области A меньше по числовому значению, чем 11-битный стандартный идентификатор области B)	Сообщение со стандартным идентификатором, имеющим меньшее значение, обладает более высоким приоритетом. MOAR.28 – старший бит стандартного идентификатора (в случае 11 битного идентификатора). MOAR.18 – младший бит стандартного идентификатора
A.MOAR.28 – A.MOAR.18 = B.MOAR.28 – B.MOAR.18 A.MOAR.IDE = 0_B – стандартный идентификатор, B.MOAR.IDE = 1_B – расширенный идентификатор	При равенстве значений идентификаторов, стандартный идентификатор имеет приоритет перед расширенным идентификатором.

Окончание таблицы 23.24

1	2
A.MOAR.28 – A.MOAR.18 = B.MOAR.28 – B.MOAR.18 A.MOAR.IDE = B.MOAR.IDE = 0_B A.MOAR.DIR = 1_B – сообщение данных, B.MOAR.DIR = 0_B – сообщение удаленного запроса	При равенстве значений идентификаторов, стандартное сообщение данных имеет приоритет перед стандартным сообщением удаленного запроса.
A.MOAR.28 – A.MOAR.18 = B.MOAR.28 – B.MOAR.18 A.MOAR.IDE = B.MOAR.IDE = 1_B A.MOAR.DIR = 1_B – сообщение данных, B.MOAR.DIR = 0_B – сообщение удаленного запроса	При равенстве значений идентификаторов, расширенное сообщение данных имеет приоритет перед расширенным сообщением удаленного запроса.
A.MOAR.28 – A.MOAR.0 < B.MOAR.28 – B.MOAR.0 A.MOAR.IDE = B.MOAR.IDE = 1_B – 29-битный идентификатор	Сообщение с расширенным идентификатором, имеющим меньшее значение, обладает более высоким приоритетом. MOAR.28 – старший бит расширенного идентификатора в случае 29-битного идентификатора, состоящего из стандартного идентификатора MOAR.28 – MOAR.18 и расширения MOAR.17 – MOAR.0. MOAR.0 – младший бит расширенного идентификатора.

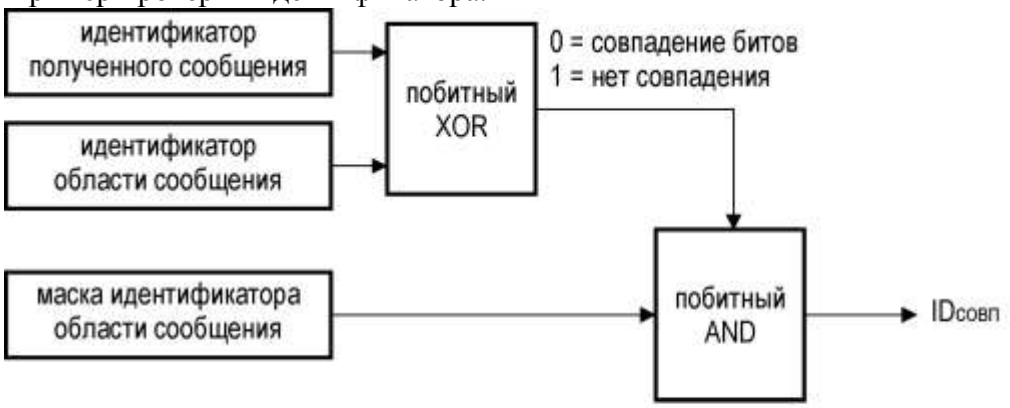
Фильтрация при получении сообщений

При получении CAN узлом сообщения, определяется область сообщения, в которой будут сохранены получаемые данные в случае успешного приема.

Область сообщения считается корректной для приема, если одновременно соблюдаются условия:

- область сообщения распределена в список областей сообщений CAN узла, которым принимается сообщение;
- флаг MSGVAL установлен (регистр MOSTATn);
- флаг RXEN установлен (регистр MOSTATn);
- бит DIR (регистр MOSTATn) равен биту RTR принимаемого сообщения. Если бит DIR = 1_B (область-для-передачи), область сообщения может принять только сообщение удаленного запроса. Если бит DIR = 0_B (область-для-приема), область сообщения может принять только сообщение данных;
- если бит MIDE = 1_B (регистр MOAMRn), бит IDE получаемого сообщения оказывает следующее влияние:
  - если IDE = 1_B (регистра MOARn), бит IDE принимаемого сообщения должен быть равен «1» (указатель расширенного идентификатора);
  - если IDE = 0_B (регистра MOARn), бит IDE принимаемого сообщения должен быть равен «0» (указатель стандартного идентификатора);
  - если бит MIDE = 0_B (регистр MOAMRn), значение бита IDE принимаемого сообщения неважно. В этом случае допускаются сообщения как со стандартным, так и с расширенным идентификатором;

- идентификатор полученного сообщения полностью (побитно) совпадает с идентификатором, хранящимся в регистре MOARn области сообщения n, за исключением битов, закрытых маской регистра MOAMRn, значение которых не важно. На рисунке 23.33 показан пример проверки идентификатора.



Примечание -
IDсовп = 0: идентификатор ID полученного сообщения совпал с ID области сообщения.
IDсовп > 0: идентификатор ID полученного сообщения не совпал с ID области сообщения.

Рисунок 23.33 – Проверка идентификатора полученного сообщения

Среди всех областей сообщений, которые отвечают указанным выше критериям, для сохранения полученного сообщения выбирается область с наивысшим приоритетом.

Для областей сообщений установлена схема определения приоритета.

Область сообщения A (далее здесь OCA) имеет приоритет над областью сообщения B (OCB), если выполняются следующие условия:

- OCA имеет более высокий класс приоритета, чем OCB (MOARA.PRI ≤ MOARB.PRI);
- в списке областей сообщений OCA занимает позицию, предшествующую позиции OCB (в случае MOARA.PRI = MOARB.PRI).

Фильтрация при передаче сообщений

Когда требуется передача содержимого какой-либо области сообщения, в соответствующих управляющих регистрах выставляются флаги, указывающие на необходимость передачи. Может возникнуть ситуация, когда передачи требуют одновременно несколько областей сообщений. Для выбора области сообщения, содержимое которой будет передано в первую очередь, существует система приоритетов.

Область сообщения считается корректной для передачи, если одновременно соблюдаются условия:

- область сообщения распределена в список областей сообщений CAN узла;
- флаг MSGVAL установлен (регистр MOSTATn);
- флаг TXRQ установлен (регистр MOSTATn);
- флаги TXEN0 и TXEN1 установлены (регистр MOSTATn).

Среди всех областей сообщений, которые отвечают указанным выше критериям, для передачи выбирается область с наивысшим приоритетом.

Схема определения приоритета следующая.

Область сообщения A (далее здесь OCA) и область сообщения B (OCB) являются областями, одновременно требующими передачи. В списке областей сообщений OCA находится перед OCB.

Если OCA и OCB имеют один класс приоритета (MOARA.PRI = MOARB.PRI), OCA будет иметь приоритет над OCB, если будут соблюдаться следующие условия:

- PRI = 10_B и сообщение, хранящееся в OCA, имеет приоритет (согласно правилам арбитража) над сообщением, хранящимся в OCB;
- PRI = 01_B или PRI = 11_B (приоритет по положению в списке).

Область сообщения, являющаяся корректной для передачи и имеющая приоритет, будет передана первой. Остальные области сообщений будут переданы по очереди, согласно их приоритетов, см. рисунок 23.34, аналогично описанному выше.

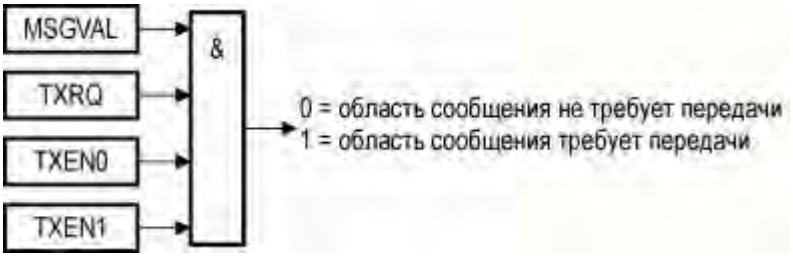


Рисунок 23.34 – Формирование запроса на передачу области сообщения

Заключительные операции при работе с сообщениями

После успешного прима или успешной передачи сообщения, ЦП получает уведомление о завершении операции для определения дальнейших действий, связанных с областью сообщения. Заключительная операция модуля CAN состоит из двух частей:

- формирование прерывания;
- сбор прерываний, ожидающих обработки в общую структуру.

Прерывания областей сообщений

После успешного сохранения принятого сообщения в области сообщения или успешной передачи, формируется соответствующее прерывание. Каждая область сообщения может формировать прерывания. Каждое такое прерывание после распределения попадает на одну из 16 выходных линий прерываний. Прерывания приема (после сохранения сообщения) также формируются после операций FIFO и шлюзовых операций. Флаги TXPND и RXPND (MOSTATn) всегда устанавливаются после успешной операции передачи/приема, независимо от состояния соответствующих флагов разрешения прерываний.

Область сообщения может формировать FIFO прерывания. Если флаг OVIE регистра MOFCRn установлен, формирование FIFO прерывания будет зависеть от настоящего типа области сообщения.

В случае если область сообщения является базовой областью FIFO приема (MOFCRn.MMC = 0001_B), выходная линия прерываний (одна из 16) для этой области определяется битовым полем TXINP регистра MOIPRn.

В случае если область сообщения является базовой областью FIFO передачи (MOFCRn.MMC = 0010_B), выходная линия прерываний (одна из 16) для этой области определяется битовым полем RXINP регистра MOIPRn, см. рисунок 23.35.

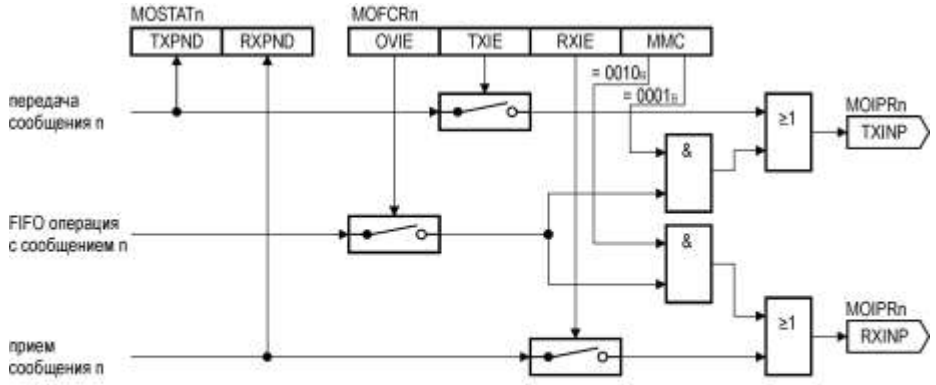


Рисунок 23.35 – Распределение прерываний

Ждущие сообщения

Когда генерируется запрос на прерывание (после приема/передачи сообщения), в одном из двух регистров ждущих прерываний MSPND0 или MSPND1 выставляется флаг ждущего сообщения. Два 32 разрядных регистра образуют область из 64 битов – по два бита (один бит для операций приема и один бит для операций передачи) для каждой из областей сообщений. Позиция флага ждущего сообщения определяется двумя демультиплексорами.

На рисунке 23.36 верхний 1-битный демультиплексор выбирает один из двух регистров MSPND0 или MSPND1, а нижний 5-разрядный – позицию флага в выбранном регистре. Регистры представлены на рисунках 23.37 – 23.40, в таблицах 23.25 – 23.29.

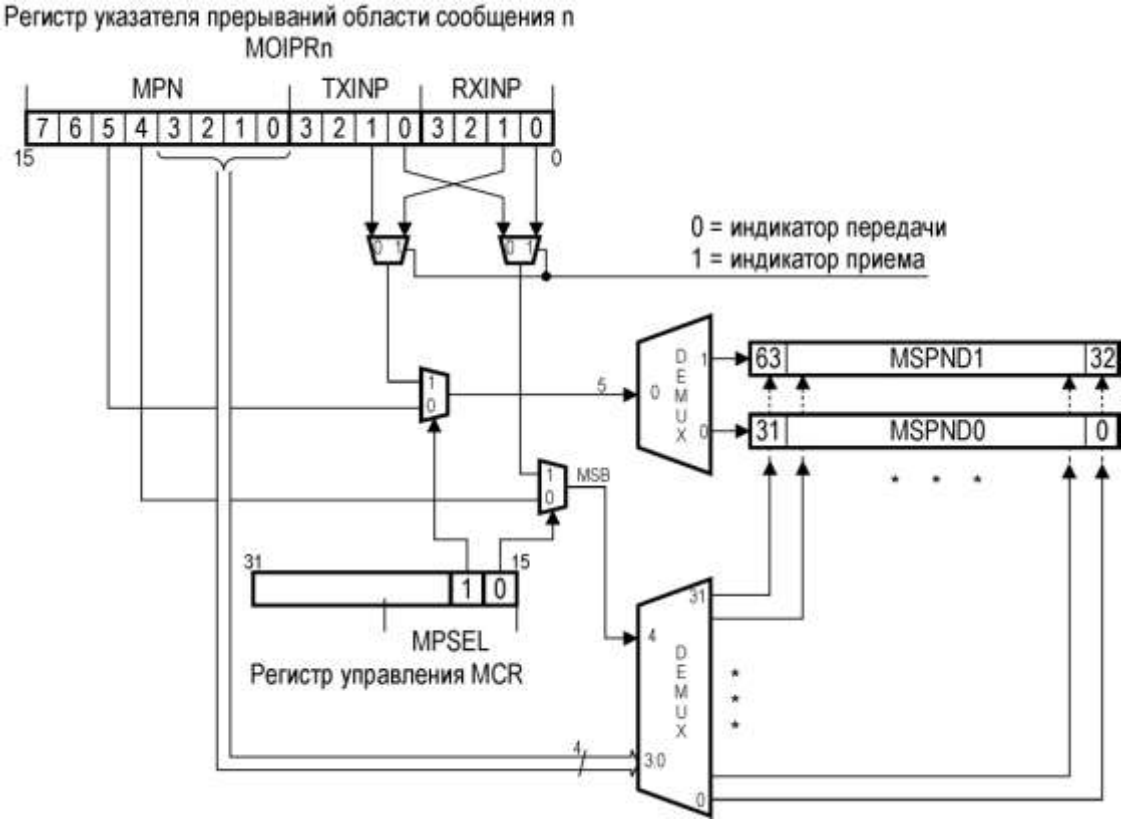


Рисунок 23.36 – Механизм выбора и установки флагов ждущих сообщений

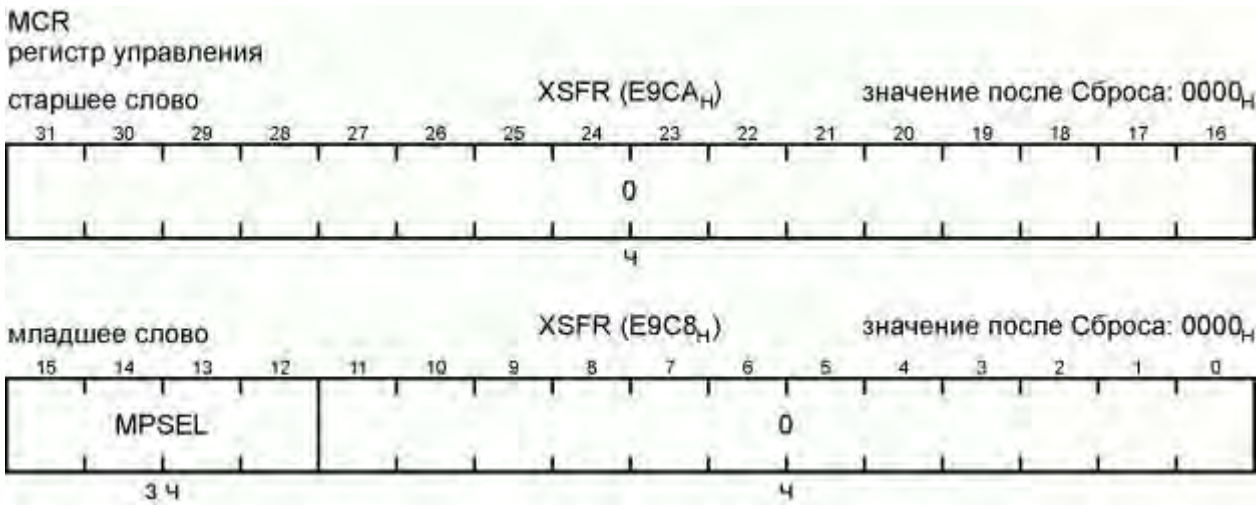


Рисунок 23.37 – Формат регистра MCR

Таблица 23.25 – Функциональное назначение полей регистра MCR

Поле	Биты	Тип	Описание
MPSEL	15-12	Чтение Запись	Выбор ждущего сообщения. Это битовое поле позволяет выбрать позицию для бита ждущего сообщения после его приема/передачи комбинацией битовых полей RXINP, TXINP и MPN регистра MOIPRn.
0	11-0, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

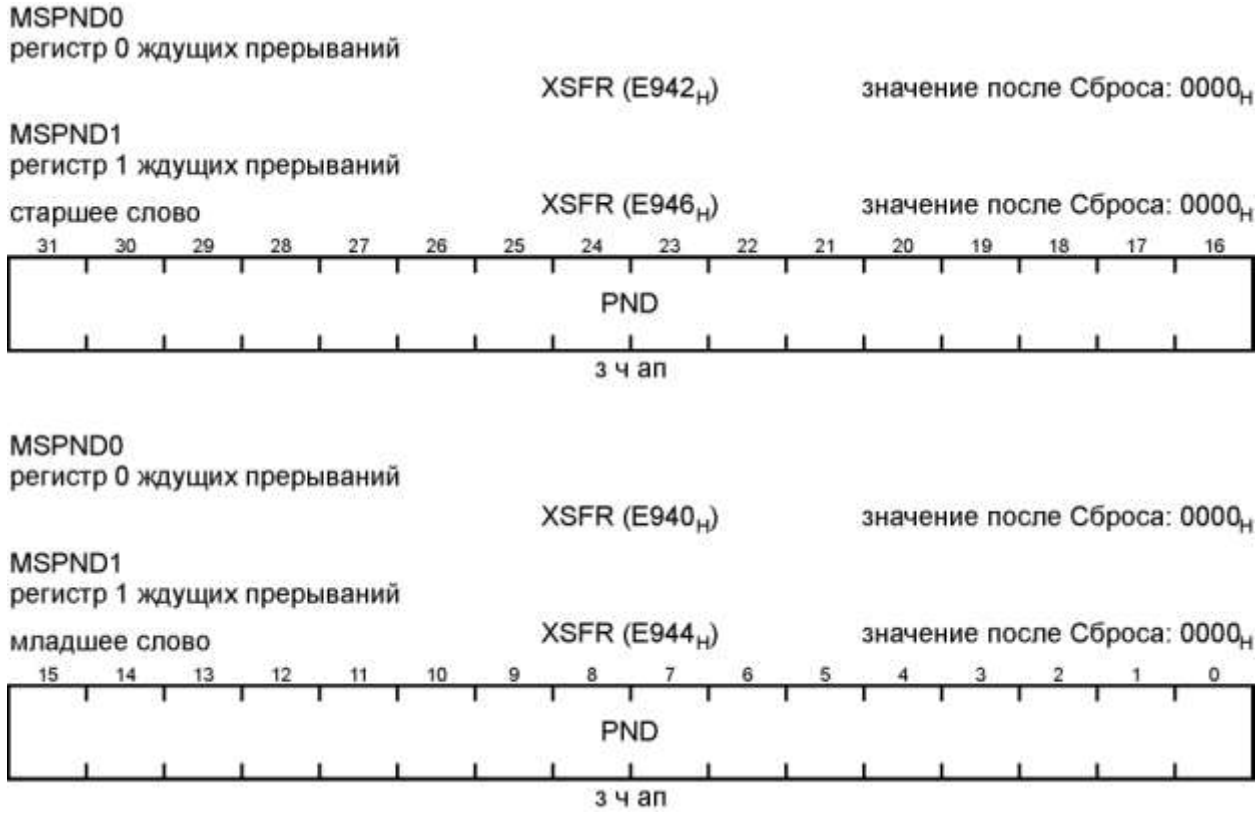


Рисунок 23.38 – Формат регистров MSPND0 и MSPND1

Таблица 23.26 – Функциональное назначение полей регистров MSPND0 и MSPND1

Поле	Биты	Тип	Описание
PND	31-0	Чтение Запись Аппаратное влияние	Поле ожидания. Когда возникает запрос на прерывание, область сообщения n устанавливает соответствующий ей бит в поле PND одного из регистров MSPND0 или MSPND1. Выбор регистра MSPND0 или MSPND1 определяется битом MPN.5, а позиция устанавливаемого бита определяется битами MPN.4 – MPN.0 регистра MOIPRn. Установленные биты могут быть очищены программно, записью «0». Запись «1» в биты регистра игнорируется.

Каждому регистру MSPND0 и MSPND1 соответствует регистр индекса сообщения MSID0/1. Регистр MSID0 или MSID1 показывает индекс активного (установленного) бита ожидания, занимающего низшую из всех позиций активных битов поля PND регистров MSPND0, MSPND1, см. рисунок 23.39, таблицу 23.27.

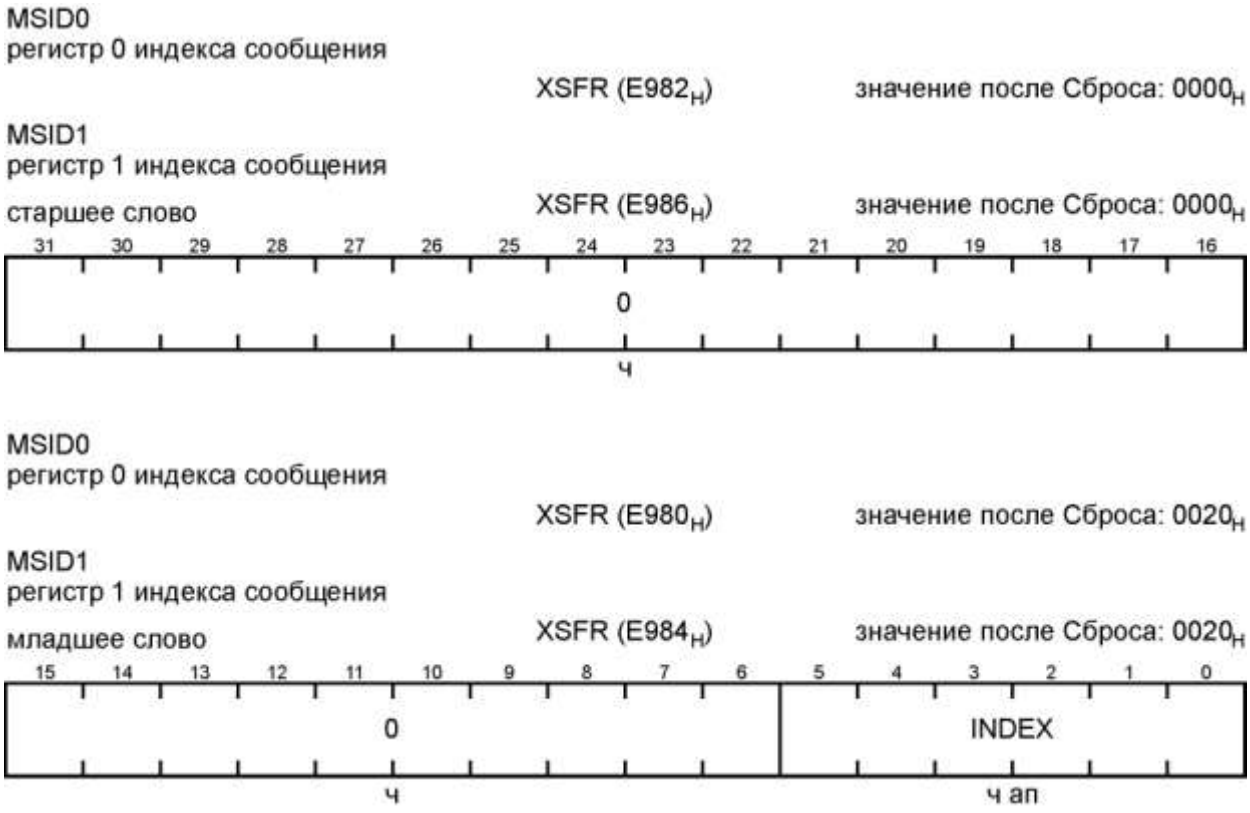


Рисунок 23.39 – Формат регистров MSID0 и MSID1

Таблица 23.27 – Функциональное назначение полей регистров MSID0 и MSID1

Поле	Биты	Тип	Описание
INDEX	5-0	Чтение Аппаратное Влияние	Индекс бита. Показывает индекс активного бита i, удовлетворяющего условиям: MSPND0/1.i & IM.i = 1; i = 0 или MSPND0/1.(i–1) – MSPND0/1.0 & IM.(i–1)–IM.0 = 0_B. Если в регистре MSPND0 или MSPND1 нет битов, удовлетворяющих этим условиям, тогда чтение битового поля INDEX возвращает значение 100000_B. Битовое поле INDEX показывает позицию первого бита ожидания регистра MSPND0 или MSPND1, в котором обслуживаются только те биты, которые не закрыты маской регистра MSIMASK.
0	15-6, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Регистр маски индекса сообщения MSIMASK содержит маску для регистра MSPND0 или MSPND 1, в котором обслуживаются только те биты, которые не закрыты маской.

Регистр MSIMASK, см. рисунок 23.40, таблицу 23.28, используется для обоих регистров MSPND0, MSPND1 и соответствующих им регистров MSID0, MSID1. Возможны два варианта определения позиции флага в регистрах MSPND0, MSPND1.

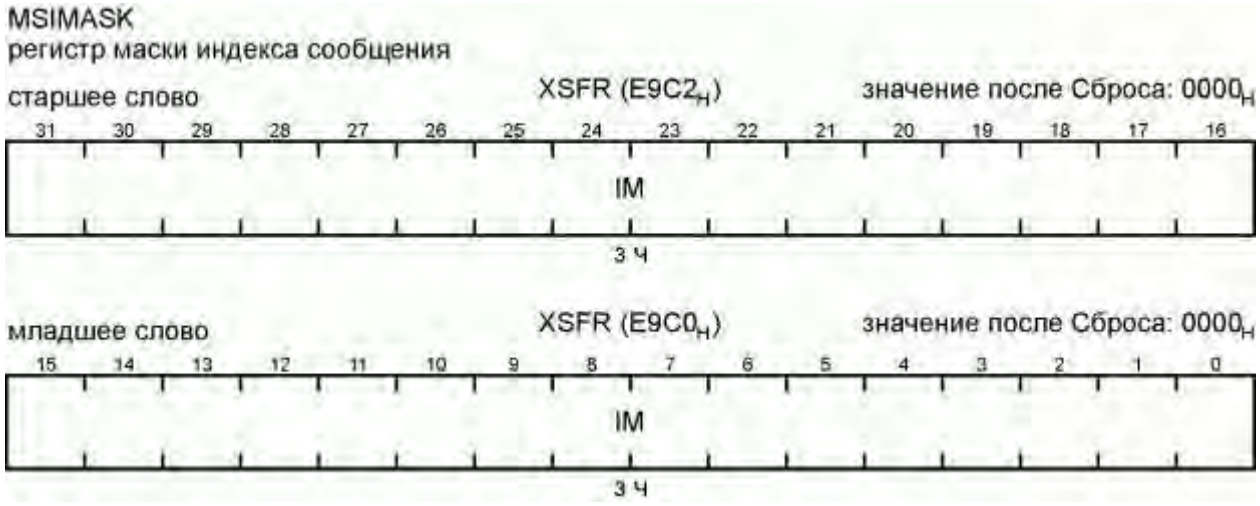


Рисунок 23.40 – Формат регистра MSIMASK

Таблица 23.28 – Функциональное назначение поля регистра MSIMASK

Поле	Биты	Тип	Описание
IM	31-0	Чтение Запись	Маска индекса. При вычислении индекса бита во внимание принимаются только те биты регистра MSPND0/MSPND1, для которых соответствующие биты поля IM равны «1».

В первом варианте битовое поле MPSEL = 0000_B (регистр MCR) и установка флага ждущего сообщения происходит следующим образом:

- 5-й бит поля MPN (MPN.5) выбирает регистр MSPND0 или MSPND1, в котором будет установлен флаг ждущего сообщения;

- младшие 5 битов поля MPN (MPN.4 – MPN.0) с помощью демультиплексора выбирают позицию флага (31-0), который будет установлен в выбранном регистре.

Во втором варианте при определении позиции флага ждущего сообщения принимаются в расчет значения битового поля MPSEL (регистр MCR) и битовых полей указателей узлов прерываний (для приема – MOIPRn.RXINP, для передачи – MOIPRn.TXINP). При этом для флагов операций приема и передачи используются разные биты выбранного регистра MSPND0 или MSPND1.

Таким образом, при MPSEL = 1111_B, установка флага ждущего сообщения происходит следующим образом:

- при передаче 1-й бит поля TXINP (TXINP.1 регистра MOIPRn) определяет регистр MSPND0 или MSPND1, в котором будет установлен флаг ждущего сообщения. При приеме регистр определяется 1-м битом поля RXINP (RXINP.1 регистра MOIPRn);

- позиция флага (31-0) в выбранном регистре выбирается младшим битом поля TXINP (TXINP.0 при передаче) или младшим битом поля RXINP (RXINP.0 при приеме) и четырьмя младшими битами поля MPN (MPN.3 – MPN.0).

Общие замечания

Регистры MSPND0, MSPND1 могут быть записаны программно. Биты, в которые записываются «1», остаются без изменений, а биты, в которые записываются «0», очищаются. Такой механизм записи позволяет очищать любой бит регистра независимо и за один такт записи, что позволяет избежать конфликта между одновременной аппаратной установкой и программной очисткой битов регистра.

Каждый регистр MSPND0, MSPND1 связан с соответствующим регистром индекса сообщения MSID0, MSID1, который отражает позицию самого младшего бита из всех установленных (равных «1») в регистре MSPND0 или MSPND1. Регистры MSID0 и MSID1 доступны только для чтения и обновляются незамедлительно после изменения (аппаратного, либо программного) содержимого соответствующих регистров MSPND0 и MSPND1.

23.7 Операции с данными областей сообщений

Прием сообщения

После получения сообщение сохраняется в области сообщения в соответствии со схемой, показанной на рисунке 23.41. Помимо сохранения данных в области сообщения, модуль CAN осуществляет обмен данными с ЦП.

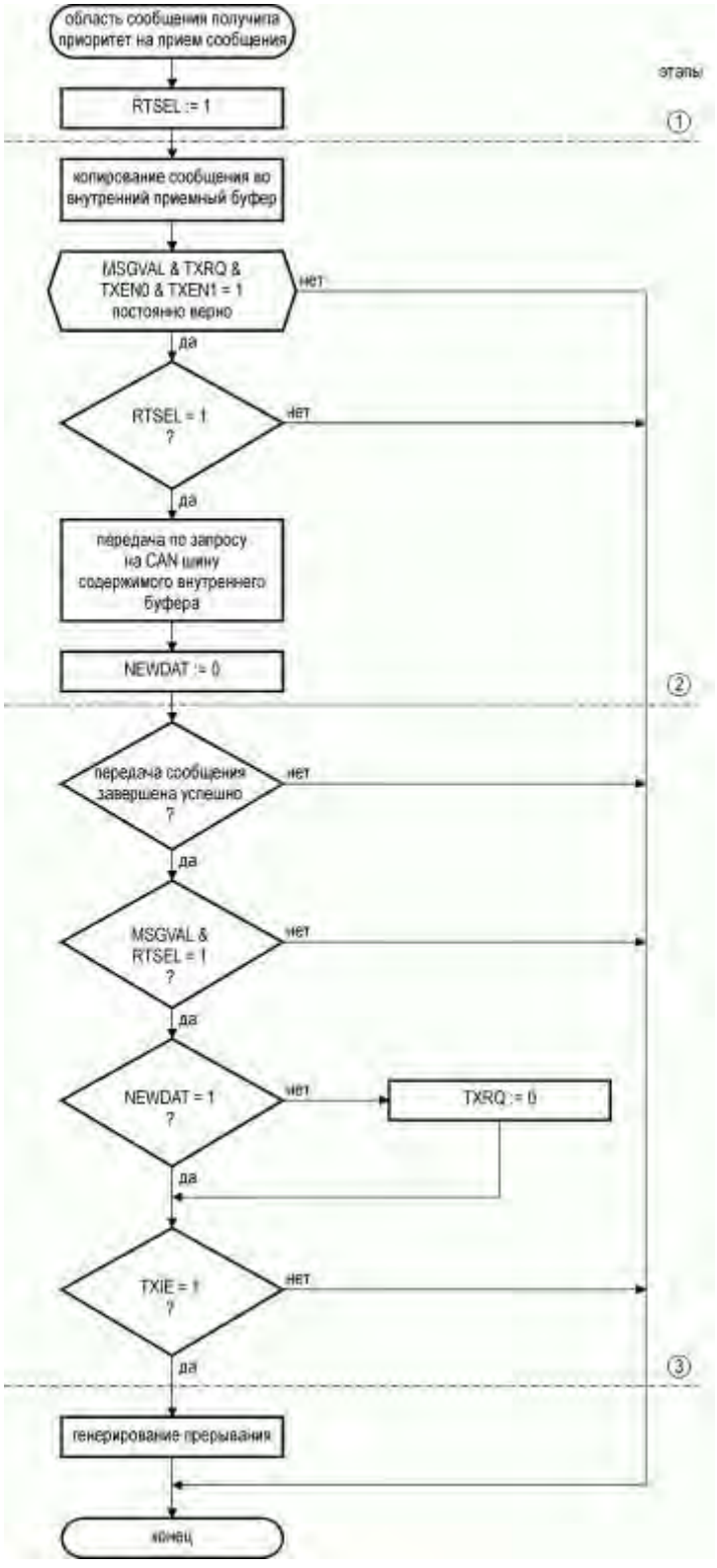


Рисунок 23.41 – Прием сообщения

Регистры управления и состояния областей сообщений

В состав каждой области сообщения входят девять 32-разрядных регистра. Расположение регистров представлено на рисунке 23.41, где для примера взята 5-я область сообщения. Расположение регистров остальных областей сообщений идентично. Адреса регистров всех областей сообщений собраны в таблице 23.40.

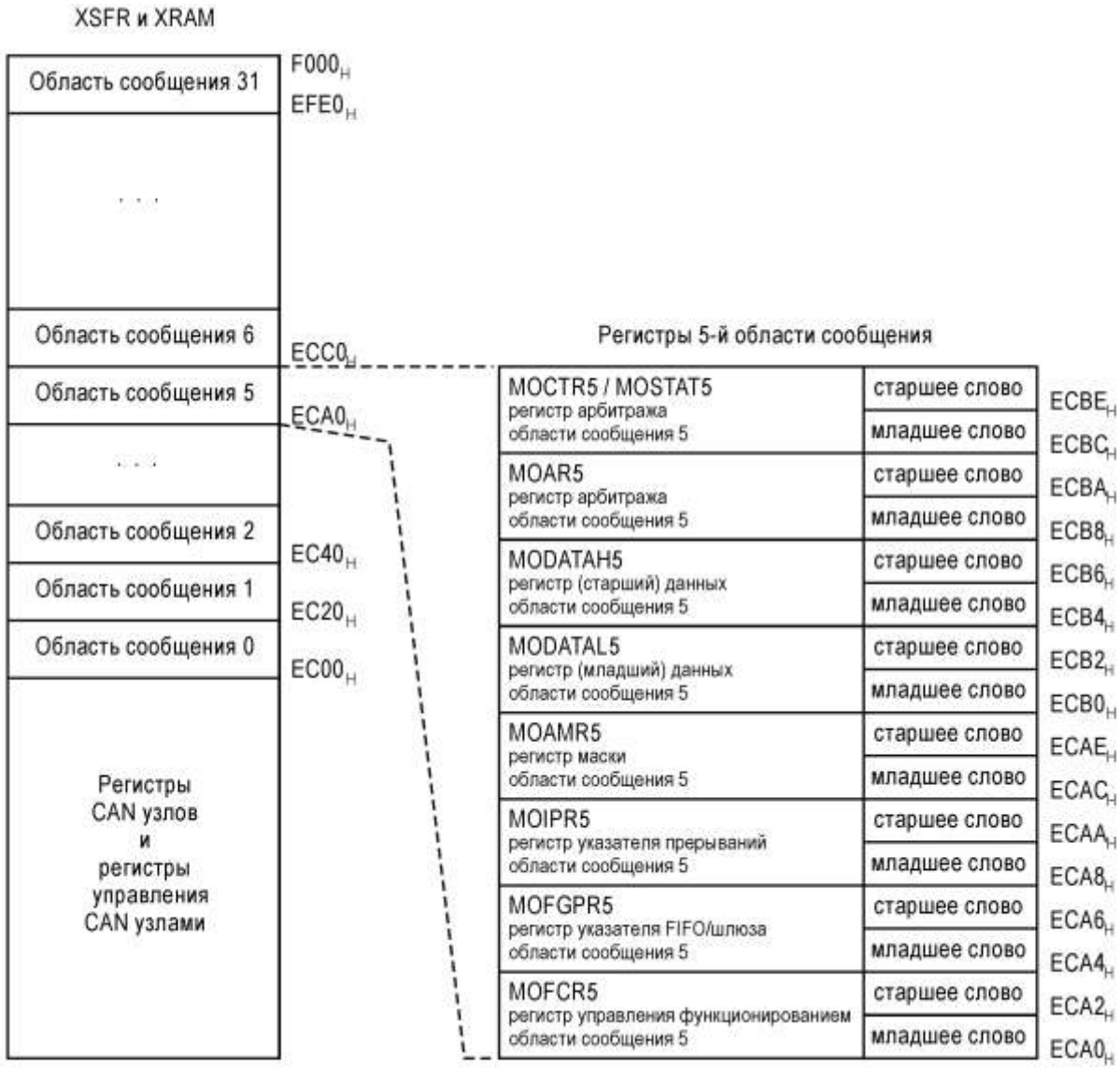
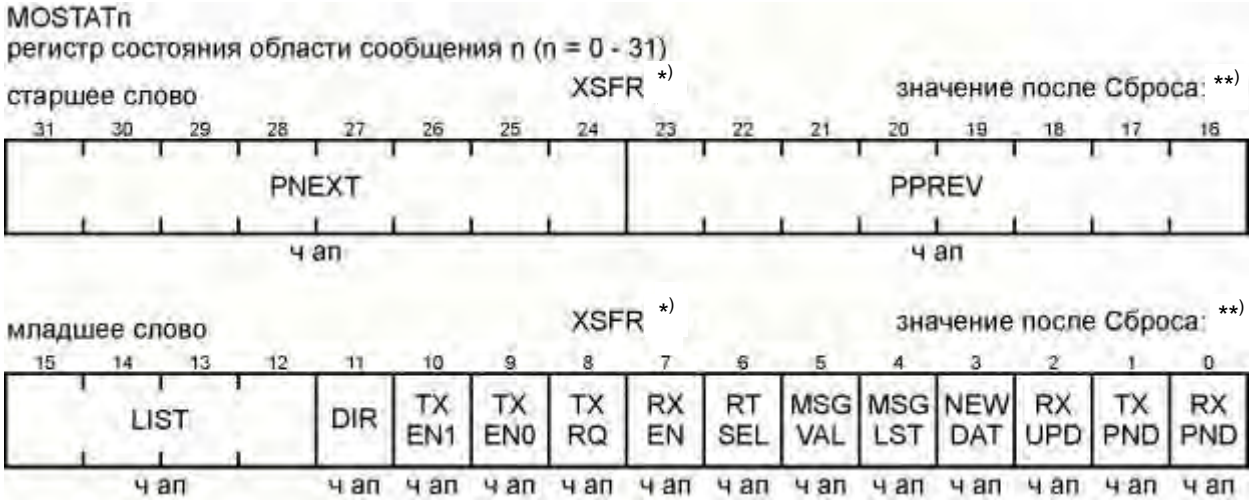


Рисунок 23.42 – Структура областей сообщений

Регистры состояния области сообщения MOSTATn доступны только для чтения и содержат информацию о состоянии списка, в том числе, номер списка, которому принадлежит текущая область сообщения, и номера предшествующей и следующей за ней областей.

Регистры управления областью сообщения MOCTRn и регистры состояния области сообщения MOSTATn расположены по одному адресу. Регистры MOCTRn доступны только для записи и позволяют программно устанавливать и сбрасывать соответствующие управляющие биты, см. рисунки 23.43, 23.44, таблицы 23.29 – 23.31.



*) Адреса регистров – в таблице 23.40.
**) Значения регистров – в таблице 23.41.

Рисунок 23.43 – Формат регистров MOSTATn, где n = 0 – 31

Таблица 23.29 – Функциональное назначение полей регистров MOSTATn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
RXPND	0	Чтение Аппаратное влияние	Флаг ожидания приема: 0 Не было получено ни одного сообщения. 1 Было получено (непосредственно с CAN шины или копированием через шлюз) сообщение и записано в область сообщения n. Флаг RXPND не сбрасывается аппаратно и должен сбрасываться программно.
TXPND	1	Чтение Аппаратное влияние	Флаг ожидания передачи: 0 Не было получено ни одного сообщения. 1 Сообщение из области сообщения n было передано посредством CAN шины. Флаг TXPND должен устанавливаться программно. Сброс флага TXPND происходит аппаратно.
RXUPD	2	Чтение Аппаратное влияние	Флаг изменения: 0 Никаких изменений не происходит. 1 В текущий момент происходит изменение идентификатора сообщения, DLC и данных области сообщения.

Продолжение таблицы 23.29

1	2	3	4
NEWDAT	3	Чтение Аппаратное влияние	Флаг завершенных изменений: 0 Содержимое области сообщения n не изменялось с момента сброса флага NEWDAT. 1 Содержимое области сообщения n было изменено. Флаг NEWDAT выставляется аппаратно после того, как принятое сообщение было сохранено в области сообщения n. Флаг NEWDAT сбрасывается аппаратно после начала передачи содержимого области сообщения n. Для предотвращения аппаратного сброса флага TXRQ в конце текущей передачи данных области сообщения n, флаг NEWDAT должен быть установлен программно после начала передачи.
MSGLST	4	Чтение Аппаратное влияние	Флаг потери сообщения: 0 Потерянных сообщений нет. 1 Было потеряно сообщение. В область сообщения n были записаны новые данные, в то время как был выставлен флаг NEWDAT, что повлекло за собой потерю ранее хранившегося сообщения.
MSGVAL	5	Чтение Аппаратное влияние	Флаг корректности области сообщения: 0 Область сообщения n некорректна. 1 Область сообщения n корректна. В работе CAN узла принимают участие только корректные области сообщений.

Продолжение таблицы 23.29

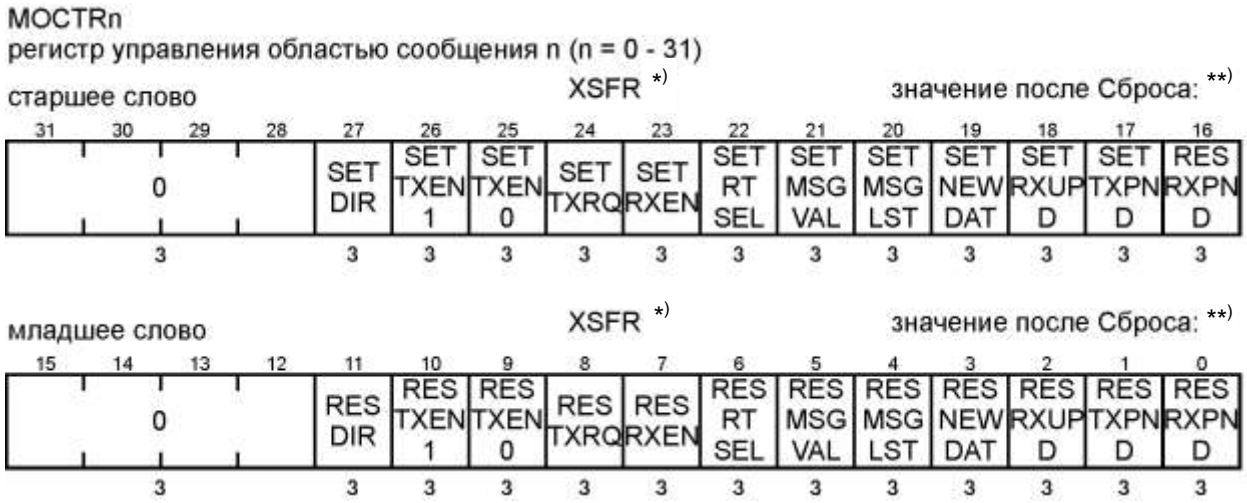
1	2	3	4
RTSEL	6	Чтение Аппаратное влияние	Флаг распределения области сообщения: 0 Область сообщения n не выбрана для операций приема/передачи. 1 Область сообщения n выбрана для операций приема/передачи. Прием сообщения: Флаг RTSEL устанавливается аппаратно, когда полученное сообщение прошло проверку идентификации для сохранения в области сообщения n. До того, как принятое сообщение будет сохранено в области сообщения n, проверяется состояние флага RTSEL (установлен ли). Это используется ЦП следующим образом – сбрасывая программно флаг RTSEL, ЦП может запрещать запись полученного сообщения в область сообщения n. Передача сообщения: Если область сообщения n прошла проверку идентификации и содержимое готово к отправке, устанавливается флаг RTSEL. До того, как будет начата отправка сообщения, проверяется состояние флага RTSEL (установлен ли) и состояние флага NEWDAT (сброшен ли). Также проверяется, чтобы флаг RTSEL оставался установленным до окончания передачи сообщения. Проверка состояния флага RTSEL может потребоваться для того, чтобы избежать конфликта, возникающего при одновременной попытке передать и изменить содержимое области сообщения n. В остальных случаях состояние флага RTSEL игнорируется. Флаг RTSEL не имеет отношения к фильтрации сообщений и не сбрасывается аппаратно.
RXEN	7	Чтение Аппаратное влияние	Флаг разрешения приема: 0 Область сообщения n не доступна для приема сообщений 1 Область сообщения n доступна для приема сообщений. Состояние флага RXEN важно только при фильтрации принимаемых сообщений.

Продолжение таблицы 23.29

1	2	3	4
TXRQ	8	Чтение Аппаратное влияние	Запрос на передачу сообщения: 0 Передача содержимого области сообщения n не требуется. 1 Требуется передача содержимого области сообщения n. Запрос на передачу содержимого области сообщения имеет силу только в случае, если установлены флаги TXRQ, TXEN0, TXEN1 и MSGVAL. TXRQ устанавливается аппаратно в случае получения корректного удаленного запроса. После успешной передачи сообщения и, если флаг NEWDAT повторно не выставлен программно, флаг TXRQ аппаратно сбрасывается.
TXEN0	9	Чтение Аппаратное влияние	Флаг 0 разрешения передачи: 0 Область сообщения n не доступна для передачи сообщения. 1 Область сообщения n доступна для передачи сообщения. Содержимое области сообщения n может быть передано только при условии, что оба флага TXEN0 и TXEN1 установлены. Пользователь может сбросить флаг TXEN0 для запрета передачи сообщения, которое в текущий момент корректируется, или же для запрета автоматической передачи сообщения в ответ на удаленный запрос.
TXEN1	10	Чтение Аппаратное влияние	Флаг 1 разрешения передачи: 0 Область сообщения n не доступна для передачи сообщения. 1 Область сообщения n доступна для передачи сообщения. Содержимое области сообщения n может быть передано только при условии, что оба флага TXEN0 и TXEN1 установлены. Модуль CAN использует флаг TXEN1 для выбора активной области сообщения при FIFO передачах.
DIR	11	Чтение Аппаратное влияние	Флаг распределения: 0 Область для приема. При TXRQ = 1_B для передачи формируется сообщение удаленного запроса с идентификатором области сообщения n. Полученное в ответ на запрос сообщение данных сохраняется в области сообщения n. 1 Область для приема. Если TXRQ = 1_B, содержимое области сообщения предназначается для передачи в качестве сообщения данных. При получении сообщения удаленного запроса с корректным идентификатором устанавливается флаг TXRQ.

Окончание таблицы 23.29

1	2	3	4
LIST	15-12	Чтение Аппаратное влияние	Номер списка. Битовое поле отражает номер списка областей сообщений, которому принадлежит область сообщения n. Битовое поле LIST изменяется аппаратно каждый раз, когда происходит перераспределение области сообщения n посредством панели команд.
PPREV	23-16	Чтение Аппаратное влияние	Указатель на предыдущий элемент списка. Битовое поле хранит номер области сообщения, предшествующей настоящей области сообщения n в списке.
PNEXT	31-24	Чтение Аппаратное влияние	Указатель на следующий элемент списка. Битовое поле хранит номер области сообщения, следующей за настоящей областью сообщения n в списке.



*) Адреса в – таблице 23.40.
**) Значения – в таблице 23.41.

Рисунок 23.44 – Формат регистров MOCTRn, где n = 0 – 31

Таблица 23.30 – Функциональное назначение полей регистров MOCTRn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
RESRXPND, SETRXPND	0, 16	запись	Сброс/установка бита RXPND регистра MOSTATn
RESTXPND, SETTXPND	1, 17	запись	Сброс/установка бита TXPND регистра MOSTATn
RESRXUPD, SETRXUPD	2, 18	запись	Сброс/установка бита RXUPD регистра MOSTATn
RESNEWDAT, SETNEWDAT	3, 19	запись	Сброс/установка бита NEWDAT регистра MOSTATn
RESMSGLST, SETMSGLST	4, 20	запись	Сброс/установка бита MSGLST регистра MOSTATn
RESMSGVAL, SETMSGVAL	5, 21	запись	Сброс/установка бита MSGVAL регистра MOSTATn

Окончание таблицы 23.30

1	2	3	4
RESRTSEL, SETRTSEL	6, 22	запись	Сброс/установка бита RTSEL регистра MOSTATn
RESRXEN, SETRXEN	7, 23	запись	Сброс/установка бита RXEN регистра MOSTATn
RESTXRQ, SETTXRQ	8, 24	запись	Сброс/установка бита TXRQ регистра MOSTATn
RESTXEN0, SETTXEN0	9, 25	запись	Сброс/установка бита TXEN0 регистра MOSTATn
RESTXEN1, SETTXEN1	10, 26	запись	Сброс/установка бита TXEN1 регистра MOSTATn
RESDIR, SETDIR	11, 27	запись	Сброс/установка бита DIR регистра MOSTATn
0	15-12, 31-28	чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».
Примечание – Правила сброса/установки для битов регистров MOCTRn, где n = 0 – 31, сведены в таблицу 23.31.			

Таблица 23.31 – Состояния битов сброса/установки регистров MOCTRn и их влияние на соответствующие биты регистров MOSTATn, где n = 0 – 31

Бит RESy[1)]	Бит SETy[1)]	Действие, оказываемое на бит «y», регистра MOSTATn
0	0 Нет записи	Без изменений
Нет записи 1	0 1	
1	0 Нет записи	Сброс бита
0 Нет записи	1	Установка бита

[1)] Параметр «y» подразумевает вторую часть названия бита RXPND, TXPND и т. д. до DIR.

Флаг корректности области сообщения MSGVAL

При приеме сообщения информация сохраняется в области сообщения, только в том случае, если бит MSGVAL = 1_B, регистр MOSTATn. Если ЦП очищает бит MSGVAL, модуль CAN останавливает запись в область сообщения, и далее область сообщения может быть реконфигурирована центральным процессором с последующей записью в нее информации без участия модуля CAN.

Флаг распределения области сообщения RTSEL

Реконфигурация области сообщения центральным процессором во время работы модуля CAN (например, сброс MSGVAL, изменение области сообщения и повторная установка MSGVAL) происходят следующим образом:

1 Область сообщения получает приоритет.

2 ЦП очищает MSGVAL для реконфигурации области сообщения.

3 После реконфигурации ЦП снова устанавливает MSGVAL.

4 Завершение получения сообщения. Если MSGVAL установлен, полученные данные сохраняются в области сообщения, генерируется запрос на прерывание, и (если сконфигурировано) производятся шлюзовые и FIFO операции.

После реконфигурации области сообщения, дальнейшее сохранение данных, см. шаг 4 выше, может быть нежелательным. Запретить запись данных в область сообщения можно посредством бита RTSEL.

После получения областью сообщения n приоритета, ее бит RTSELn устанавливается модулем CAN, открывая, таким образом, область сообщения n для записи. После приема сообщения, модуль CAN дополнительно проверяет возможность записи в область сообщения n, а именно – установлен ли все еще бит RTSELn. И только в том случае, если RTSELn = 1_B, полученные данные сохраняются в области сообщения n (вместе со всеми последующими действиями, такими как прерывания области сообщения, шлюзовые и FIFO операции, установка/ сброс флагов).

Если во время операций модуля CAN область сообщения становиться некорректной (сброс флага MSGVAL), флаг RTSEL должен быть сброшен до того, как флаг MSGVAL будет установлен снова, или, по крайней мере, одновременно с ним. Это необходимо для предотвращения сохранения старой информации в области сообщения. Итак, реконфигурация области сообщения должна происходить следующим образом:

1 Сброс флага MSGVAL.

2 Реконфигурация области сообщения, пока MSGVAL = 0_B.

3 Сброс флага RTSEL и далее установка флага MSGVAL.

Флаг разрешения приема RXEN

Полученное с CAN шины сообщение может быть сохранено в области сообщения только в случае RXEN = 1_B. Модуль CAN проверяет состояние флага RXEN только во время фильтрации принимаемого сообщения. После того, как сообщение принято, состояние флага не имеет значения и не оказывает влияния на дальнейшее сохранение данных в области сообщения.

Флаг RXEN позволяет управлять закрытием области сообщения: после сброса RXEN полученное сообщение сохраняется в области сообщения, которая получила приоритет, но в сохранении последующих сообщений эта область не принимает участия.

Флаги изменения RXUPD, завершенных изменений NEWDAT и потери сообщения MSGLST

Индикатором процесса сохранения (изменения) данных в области сообщения является флаг RXUPD, который выставляется с началом процесса сохранения (изменения) и сбрасывается с его окончанием.

После сохранения полученного сообщения (идентификатора, бита IDE, кода длины данных; поля данных, в случае сообщения данных) выставляется флаг NEWDAT. Если к моменту выставления (завершение сохранения/изменения данных) флаг NEWDAT был уже установлен, выставляется флаг MSGLST, который говорит о том, что произошла потеря данных.

Флаги RXUPD и NEWDAT позволяют произвести чтение корректных данных из области сообщения во время текущих операций модуля CAN. Рекомендуемая последовательность шагов следующая:

1 Сброс флага NEWDAT.

2 Чтение данных (идентификатор, данные и т. д.) из области сообщения.

3 Проверка флагов. Оба флага NEWDAT и RXUPD должны быть равны «0». В случае не выполнения этого условия – возвращение к шагу 1.

4 Если условие шага 3 выполнено, содержимое области сообщения корректно и не используется модулем CAN в течение операции чтения.

Поведение флагов RXUPD, NEWDAT и MSGLST идентично как для сообщений данных, так и для сообщений удаленных запросов.

Передача данных

Алгоритм передачи сообщений показан на рисунке 23.45. Одновременно с копированием данных: идентификатора, бита IDE, бита RTR, равного биту DIR, кода длины данных и собственно данных, из области сообщения, содержимое которой должно быть

передано во внутренний передающий буфер соответствующего CAN узла, для контроля соблюдения четкой последовательности выполнения всех операций выставляются флаги состояния.

Процессы передачи сообщений данных и сообщений удаленных запросов идентичны, см. рисунок 23.45.

Флаги потери сообщения MSGVAL, разрешения передачи TXEN0, TXEN1 и запрос на передачу сообщения TXRQ

Сообщение может быть передано только в случае, когда все четыре бита MSGVAL, TXEN0, TXEN1, TXRQ равны 1_B, см. рисунок 23.34, таблицу 23.32.

Таблица 23.32 – Описание битов, управляющих передачей сообщений

Бит (флаг)	Описание
MSGVAL	Флаг корректности области сообщения. Основной бит доступа к области сообщения.
TXRQ	Запрос на передачу сообщения. Бит устанавливается, когда требуется передача содержимого области сообщения. По окончании успешной передачи сообщения, этот бит сбрасывается аппаратно, за исключением случая, когда возникает необходимость передачи новых данных (бит NEWDAT = 1_B). Если установлен бит STT регистра MOFCRn, бит TXRQ очищается, когда данные из области сообщения копируются в передающий буфер CAN узла. Полученный удаленный запрос (по окончании получения сообщения удаленного запроса) устанавливает бит TXRQ для формирования запроса на передачу запрашиваемых данных.
TXEN0	Флаг 0 разрешения передачи. Этот флаг может быть временно сброшен программно для запрета передачи содержимого области сообщения, в которую записываются новые данные. Это позволяет избежать передачи некорректных смешанных данных. Если TXEN0 = 0_B, допускается передача сообщения удаленного запроса, но передача сообщения данных станет возможной только после программной установки флага TXEN0.
TXEN1	Флаг 1 разрешения передачи. Этот флаг используется при FIFO передачах для выбора активной области в структуре FIFO. Для областей сообщений, не являющихся элементами FIFO структуры, флаг TXEN1 может быть или постоянно установленным (= 1_B), или использоваться как второй независимый бит разрешения передачи.

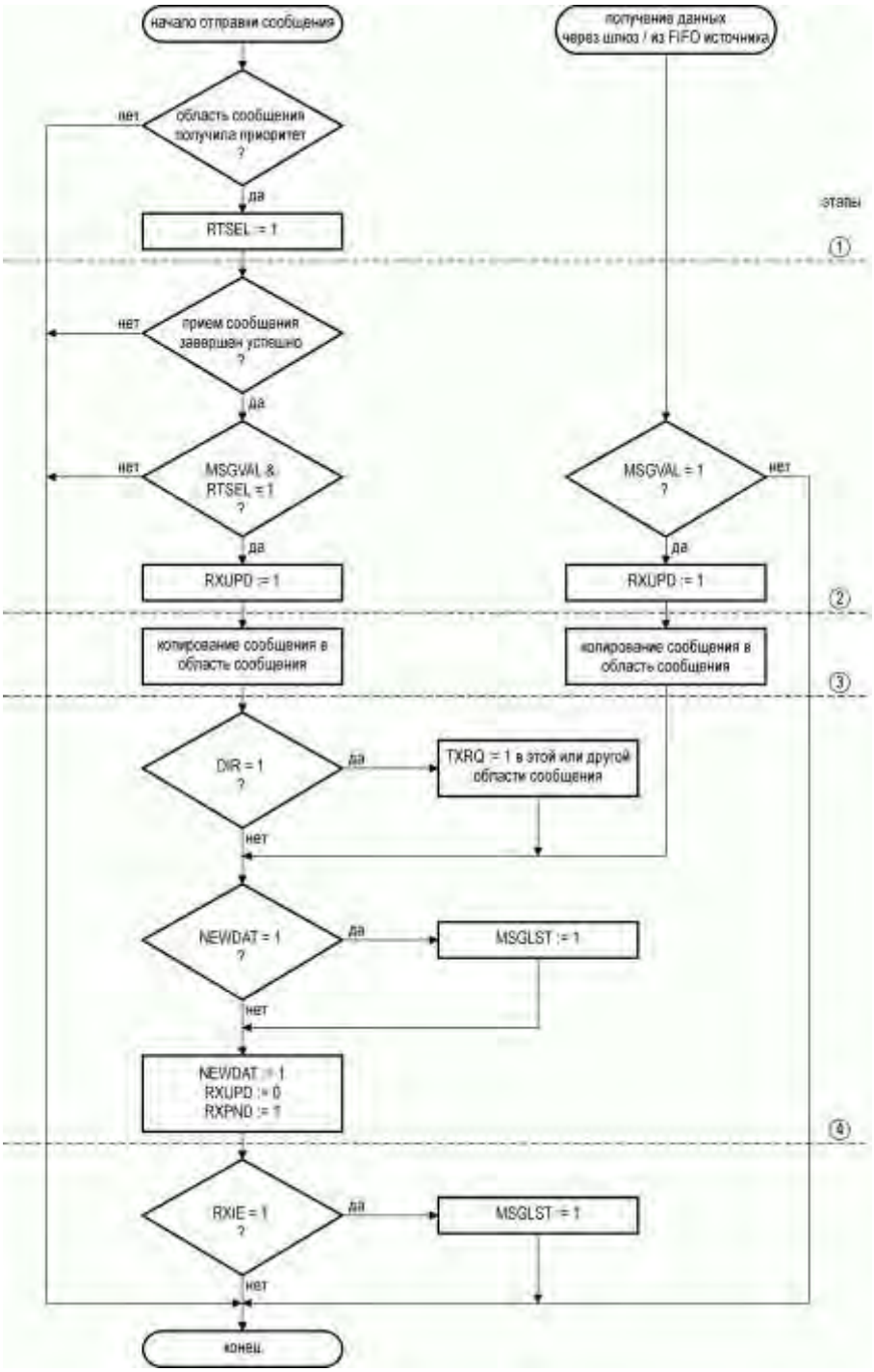


Рисунок 23.45 – Передача сообщения

Флаг распределения области сообщения RTSEL

Флаг RTSEL выставляется после того, как область сообщения получает приоритет для передачи своего содержимого.

Когда данные области сообщения копируются в передающий буфер, флаг RTSEL проверяется, и сообщение передается только в случае, если RTSEL = 1_B. После успешной передачи сообщения, флаг RTSEL проверяется снова, и если RTSEL = 1_B, осуществляются дальнейшие операции.

Для полной и завершенной реконфигурации корректной области сообщения должны быть выполнены следующие шаги:

1 Очистка бита MSGVAL.

2 Реконфигурация области сообщения пока MSGVAL = 0_B.

3 Сброс флага RTSEL и установка флага MSGVAL.

Сброс флага RTSEL гарантирует как полное отключение области сообщения от текущей передачи, так и то, что никакие операции (копирование данных в передающий буфер, включая сброс флага NEWDAT, очистка бита TXRQ, прерывание сообщения и т. д.), относящиеся к старой конфигурации этой области сообщения, не повлияют на новую конфигурацию после установки корректности области (флаг MSGVAL).

Флаг завершенных изменений NEWDAT

После завершения передачи содержимого области сообщения в передающий буфер CAN узла, флаг NEWDAT аппаратно сбрасывается, тем самым обозначая, что область сообщения открыта для записи новых данных.

Если после успешной передачи сообщения (на CAN шину) бит NEWDAT все еще остается равным «0» (в область сообщения не были записаны новые данные), флаг TXRQ автоматически аппаратно сбрасывается. Если же бит NEWDAT был снова установлен программно (потому что требуется передача новых данных), флаг TXRQ не сбрасывается, тем самым разрешая передачу новых данных.

23.8 Функции области сообщения

Регистр управления функционированием области сообщения n MOFCRn включает в себя биты, которые конфигурируют действия области сообщения, и хранит код длины данных сообщения, см. рисунок 23.46, таблицу 23.33 .

*) Адреса – в таблице 23.40.

Рисунок 23.46 – Формат регистров MOFCRn, где n = 0 – 31

Таблица 23.33 – Функциональное назначение полей регистров MOFCRn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
MMC	3-0	Чтение Запись	Выбор типа области сообщения: 0000 Стандартная область сообщения. 0001 Базовая область сообщения FIFO приема. 0010 Базовая область сообщения FIFO передачи. 0011 Вспомогательная область сообщения для FIFO передачи. 0100 Шлюзовая область-источник[1]. Комбинации 0101_B – 1111_B зарезервированы.

Продолжение таблицы 23.33

1	2	3	4
GDFS	8	Чтение Запись	Флаг пересылки сообщения через шлюз: 0 Флаг TXRQ области-приемник не изменяется. 1 Флаг TXRQ шлюзовой области-приемник[2)] устанавливается после успешной пересылки сообщения из шлюзовой области-источник в шлюзовую область-приемник. Флаг GDFS применим только к шлюзовой области-источник и игнорируется остальными.
IDC	9	Чтение Запись	Флаг копирования идентификатора: 0 Идентификатор шлюзовой области-источник не копируется. 1 Идентификатор шлюзовой области-источник (после сохранения полученного сообщения в области-источник) копируется в шлюзовую область-приемник. Флаг IDC применим только к шлюзовой области-источник и игнорируется остальными.
DLCC	10	Чтение Запись	Флаг копирования кода длины данных: 0 Код длины данных не копируется. 1 Код длины данных шлюзовой области-источник (после сохранения полученного сообщения в области-источник) копируется в шлюзовую область-приемник. Флаг DLCC применим только к шлюзовой области-источник и игнорируется остальными.
DATC	11	Чтение Запись	Флаг копирования данных. 0 Поле данных не копируется. 1 Поля данных регистров MODATALn и MODATAHn шлюзовой области-источника (после сохранения полученного сообщения в области-источнике) копируется в шлюзовую область-приемник. Флаг DATC применим только к шлюзовой области-источник и игнорируется остальными
RXIE	16	Чтение Запись	Бит разрешения прерывания приема. Бит разрешает формирование прерывания областью сообщения n после приема сообщения. Прерывание генерируется после получения сообщения (независимо от того, было ли сообщение получено посредством CAN шины или через шлюз). 0 Формирование прерывания запрещено. 1 Формирование прерывания разрешено. Битовое поле RXINP регистра MOIPRn выбирает выходную линию прерывания (одну из 16) для данного типа прерывания.

Продолжение таблицы 23.33

1	2	3	4
TXIE	17	Чтение Запись	Бит разрешения прерывания передачи. Бит разрешает формирование прерывания областью сообщения n после передачи сообщения. Прерывание генерируется после успешной передачи сообщения. 0 Формирование прерывания запрещено. 1 Формирование прерывания разрешено. Битовое поле TXINP регистра MOIPRn выбирает выходную линию прерывания (одну из 16) для данного типа прерывания.
OVIE	18	Чтение Запись	Флаг разрешения прерывания переполнения. Этот бит разрешает формирование FIFO прерывания области сообщения n. Это прерывание генерируется, когда указатель на текущую область сообщения CUR достигает значения битового поля SEL регистра указателя FIFO/шлюза области сообщения n (MOFGPRn). 0 FIFO прерывание запрещено. 1 FIFO прерывание разрешено. Если область сообщения n является базовой областью FIFO приема, выходную линию прерываний (одну из 16) для данного типа прерывания выбирает битовое поле TXINP регистра MOIPRn. Если область сообщения n является базовой областью FIFO передачи, выходную линию прерываний (одну из 16) для данного типа прерывания выбирает битовое поле RXINP регистра MOIPRn. На все остальные типы областей сообщений флаг OVIE не оказывает влияния.
FRREN	20	Чтение Запись	Бит разрешения удаленного запроса. Этот бит определяет, устанавливается ли флаг TXRQ в регистре области сообщения n или в регистре другой области сообщения, на которую указывает указатель CUR: 0 При приеме корректного сообщения удаленного запроса устанавливается флаг TXRQ области сообщения n. 1 При приеме корректного сообщения удаленного запроса устанавливается флаг TXRQ области сообщения, указанной битовым полем CUR.

Окончание таблицы 23.33

1	2	3	4
RMM	21	Чтение Запись	Бит удаленного мониторинга: 0 Удаленный мониторинг выключен. Идентификатор, бит IDE и DLC области сообщения n остаются без изменений после приема корректного удаленного запроса. 1 Удаленный мониторинг включен. Идентификатор, бит IDE и DLC корректного удаленного запроса копируются в область для передачи n. Бит RMM оказывает влияние только в случае, если область сообщения n является областью для передачи.
SDT	22	Чтение Запись	Бит блокирования повторной пересылки данных: - если SDT = 1_B и область сообщения n не является базовой областью FIFO, тогда флаг MSGVAL сбрасывается, когда область сообщения n принимает участие в пересылке данных (прием или передача); - если SDT = 1_B, а область сообщения n является базовой областью FIFO, тогда флаг MSGVAL сбрасывается, когда указатель CUR на текущую область сообщения n достигает значения битового поля SEL регистра MOFGPRn; - если SDT = 0_B, на флаг MSGVAL не оказывается влияния.
STT	23	Чтение Запись	Бит однократной пересылки данных. Если этот бит установлен, то TXRQ сбрасывается с началом передачи содержимого области сообщения n. Таким образом, при неудачной пересылке, не будет осуществлена повторная передача сообщения.
DLC	27-24	Чтение Запись Аппаратное влияние	Код длины данных. Это битовое поле определяет количество байт данных, хранящихся в области сообщения n. Допустимые значения – от 0 до 8. Запись любого числа больше восьми в битовое поле DLC будет означать, что длина данных – 8 байт. Если получено сообщение с DLC > 8, полученное значение сохраняется в области сообщения.
0	7-4, 19, 31-28	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

[1] Область сообщения, используемая в качестве источника сообщения для пересылки через шлюз.

[2] Область сообщения, используемая в качестве приемника сообщения при пересылке через шлюз.

Стандартная область сообщения

Область сообщения определяется как стандартная область сообщения, если битовое поле MMC = 0000_B (регистр MOFCRn). Стандартная область сообщения может принимать и передавать сообщения, согласно правилам, описанным выше. Дополнительно имеются два режима, каждый из которых может быть выбран индивидуально:

- режим передачи данных с защитой от повторений;
- режим однократной пересылки данных.

Режим передачи данных с защитой от повторений

Выбирается установкой бита SDT регистра MOFCRn.

Режим передачи данных, в котором имеется механизм защиты от дублирования передачи информации.

Прием сообщений в режиме передачи данных с защитой от повторений

После приема сообщения данных, принятые данные сохраняются в выбранной области сообщения, в связи с чем, предыдущее содержимое этой области теряется. При дальнейшем приеме, очередные новые данные могут быть записаны в ту же область сообщения, что в свою очередь опять вызовет потерю хранившихся данных.

Если SDT = 1_B (режим включен), то после приема сообщения и сохранения его в выбранной области сообщения, флаг MSGVAL этой области автоматически аппаратно сбрасывается, что делает область некорректной, а, следовательно, недоступной для последующей записи.

После приема сообщения удаленного запроса, автоматического сброса флага MSGVAL не происходит.

Передача сообщений в режиме передачи данных с защитой от повторений

Если область сообщения получает серию удаленных запросов, она, в ответ на это, передает несколько сообщений данных. Если в промежутки времени между передачами содержимое области сообщения не изменялось, то на CAN шину будут несколько раз переданы одни и те же данные.

Если SDT = 1_B, то сразу после однократной успешной передачи данных флаг MSGVAL передающей области сообщения будет сброшен, и область станет недоступной для передачи.

После передачи сообщения удаленного запроса, автоматического сброса флага MSGVAL не происходит.

Режим однократной пересылки данных

Выбирается установкой бита STT регистра MOFCRn.

Если бит STT = 1_B, то бит TXRQ сбрасывается, когда содержимое области сообщения копируется в передающий буфер CAN узла. Таким образом, при дальнейшей неудачной (вследствие ошибок) пересылке сообщения по CAN шине, повторной передачи не будет.

FIFO структура области сообщения

Регистр указателя FIFO/шлюза области сообщения MOFGPRn содержит установки указателей на области сообщений, которые используются при операциях FIFO и шлюзовых операциях, см. рисунок 23.47, таблицу 23.34.

*) Адреса – в таблице 23.40.

Рисунок 23.47 – Формат регистра MOFGPRn

Таблица 23.34 – Функциональное назначение полей регистра MOFGPRn

Поле	Биты	Тип	Описание
BOT	7-0	Чтение Запись	Указатель начала. Это битовое поле хранит значение, которое указывает на первый элемент в структуре FIFO.
TOP	15-8	Чтение Запись	Указатель конца. Это битовое поле хранит значение, которое указывает на последний элемент в структуре FIFO.
CUR	23-16	Чтение Запись Аппаратное Влияние	Указатель текущей области. Это битовое поле хранит значение, которое указывает на выбранный в настоящий момент элемент в пределах списка структуры FIFO/шлюза. После каждой операции FIFO (также шлюзовой операции), битовое поле CUR изменяет свое значение – в него записывается номер следующей по списку области сообщения (указанной в битовом поле PNEXT регистра MOSTATn). Так происходит до тех пор, пока не будет достигнут последний элемент списка (битовое поле TOP), после чего в CUR заноситься номер первого элемента (битовое поле BOT).
SEL	31-24	Чтение Запись	Указатель выбора области. Это битовое поле является вторым (программным) указателем для дополнения аппаратного указателя CUR в структуре FIFO. Битовое поле SEL используется для мониторинга возникающих задач (генерирование прерываний FIFO).

В случае сильной загрузки ЦП, может быть затруднительным обработать серию сообщений модуля CAN в рамках необходимого времени. Такие ситуации могут возникать вследствие получения и/или передачи большого числа сообщений за малые промежутки времени.

Для таких случаев предусмотрена система буферов быстрого ввода-вывода, так называемая FIFO структура, которая позволяет избежать потери принимаемых сообщений и минимизировать время подготовки сообщений к отправке. Помимо этого, FIFO структура используется для автоматического приема/передачи серий сообщений и генерирования однократных прерываний по окончании операций с ними.

Допускается организация нескольких параллельных FIFO структур. Число структур и их составляющих зависит только от количества доступных областей сообщений. FIFO структура может быть создана, изменена и удалена в любой момент времени, даже во время операций модуля CAN.

На рисунке 23.48 представлена основная FIFO структура. Она состоит из одной базовой области и n-ого числа вспомогательных областей. Вспомогательные области объединяются последовательно в списки (подобно спискам областей сообщений). Базовая область может быть занесена в любой список. Хотя на рисунке базовая область не относится ни к одному из списков, она может быть вставлена в любую последовательность вспомогательных областей. Это означает, что базовая область одновременно является и вспомогательной областью (шлюзовые операции не возможны). Порядковые номера областей сообщений (0, 1, 2 и т. д.) не имеют никакого значения при FIFO операциях с областями.

Базовая область не нуждается в обязательном занесении ее в какой-либо список, в отличие от вспомогательных областей, которые должны быть определены в общий список (так как они последовательно связаны). С помощью указателей (битовые поля BOT, CUR и TOP регистра MOFGPRn) можно присоединять базовую область к вспомогательным областям, независимо от того принадлежат базовая и вспомогательные области одному списку или разным спискам.

Минимальная FIFO структура может состоять из одной области сообщения, которая будет одновременно являться и базовой и вспомогательной (фактически не используется). Максимальная FIFO структура может включать в себя все 32 области сообщений модуля CAN. В остальном, размеры FIFO структуры варьируются между этими границами.

В базовой области FIFO границы установлены. Битовое поле BOT регистра MOFGPRn базовой области указывает на самый младший элемент FIFO структуры. Битовое поле TOP регистра MOFGPRn базовой области указывает на самый старший элемент FIFO структуры. Битовое поле CUR регистра MOFGPRn базовой области указывает на вспомогательную область, которая в настоящий момент выбрана модулем CAN для передачи сообщения. Как только начинается передача, в CUR записывается номер следующей по списку вспомогательной области сообщения (CUR = PNEXT используемой области). Если значение битового поля CUR достигло номера старшего элемента списка (CUR = TOP), то следующее значение, которое будет записано в поле CUR – это BOT (реализация автоматического перехода в начало списка). Таким образом, реализуется замкнутая FIFO структура, в которой битовые поля TOP и BOT устанавливают связь между началом и концом списка.

Битовое поле SEL регистра MOFGPRn позволяет определить вспомогательную область в пределах списка, для которой генерируется прерывание всякий раз, когда указатель CUR достигает значения указателя SEL. Так же, битовое поле SEL позволяет отследить окончание запланированной передачи серии сообщений или выдать прерывание, предупреждающее о том, что FIFO структура становиться заполненной.

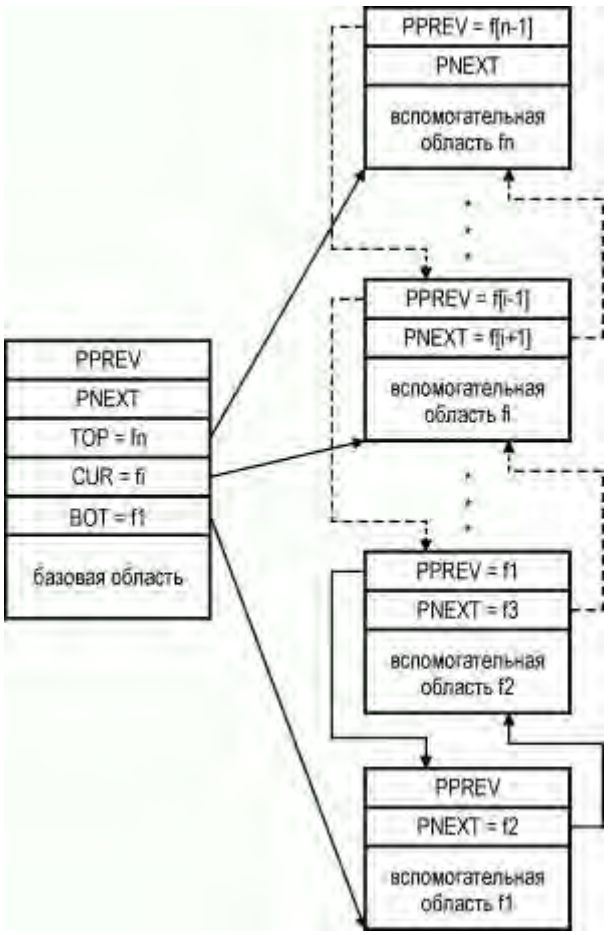


Рисунок 23.48 – FIFO структура с базовой областью и n вспомогательными областями

FIFO структура приема

FIFO структура приема используется для буферизации входящих (полученных) сообщений данных и удаленных запросов.

FIFO структура приема активируется записью значения 0001_B в битовое поле MMC регистра MOFCRn базовой области. Установка MMC = 0001_B автоматически определяет область сообщения как базовую область FIFO приема. Типы вспомогательных областей FIFO не имеют значения при операциях FIFO структуры приема.

Когда базовая область FIFO получает сообщение от CAN узла, которому она принадлежит, сообщение сохраняется не в этой базовой области, а в вспомогательной области сообщения, на которую указывает битовое поле CUR (регистра MOGPRn базовой области). При этом по умолчанию предполагается, что для вспомогательной области битовое поле MMC = 0000_B (действительное значение MMC игнорируется) и никаких операций фильтрации принимаемого сообщения не производится.

Одновременно с приемом сообщения, текущее значение указателя CUR базовой области меняется на номер следующей по списку вспомогательной области FIFO структуры. Эта вспомогательная область будет использована для приема следующего сообщения.

Если установлен флаг OVIE регистра MOFCRn базовой области, и значение указателя CUR становиться равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем TXINP базовой области сразу после сохранения полученного сообщения в вспомогательной области.

Прерывания передач генерируются, если это разрешено битом TXIE регистра MOFCRn.

Следует помнить, что сообщение сохраняется в базовой и вспомогательной области FIFO, только если MSGVAL = 1_B.

Во избежание непосредственного приема сообщения вспомогательной областью, как если бы она была независимой областью сообщения и не принадлежала FIFO структуре, флаги RXEN всех вспомогательных областей должны быть сброшены. Состояние флага RXEN неважно в случае, когда вспомогательная область занесена в список, не связанный с CAN узлом.

FIFO структура передачи

FIFO структура передачи используется для буферизации серий сообщений данных или удаленных запросов, которые должны быть отправлены. FIFO структура передачи состоит из базовой области и одной или более вспомогательных областей.

FIFO структура передачи активируется записью значения 0010_B в битовое поле MMC регистра MOFCRn базовой области. В отличие от FIFO структуры приема, в битовые поля MMC вспомогательных областей (FIFO структуры передачи) должно быть записано значение 0011_B. Указатели CUR всех вспомогательных областей должны указывать на базовую область FIFO передачи (чтобы инициализироваться программно).

Флаги TXEN1 всех вспомогательных областей сообщений, за исключением одной, на которую указывает CUR базовой области, должны быть программно сброшены. Флаг TXEN1 указанной области должен быть установлен. Указатель CUR базовой области может быть инициализирован для любой вспомогательной области.

При определении корректности областей сообщений FIFO структуры для начала FIFO операции базовая область должна быть первой определена, как корректная (MSGVAL = 1_B).

В случае удаления FIFO структуры, прежде чем начнется операция удаления, все вспомогательные области, принадлежащие этой FIFO структуре, должны быть определены как некорректные (MSGVAL = 0_B).

FIFO структура передачи использует флаги TXEN1 всех своих областей для выбора сообщения для передачи. В результате фильтрации право передавать сообщение получает та область, у которой выставлен флаг TXEN1. После передачи сообщения флаг TXEN1 аппаратно сбрасывается, а в указатель CUR записывается номер следующей области, требующей отправки сообщения, для которой уже выставлен (аппаратно) свой флаг TXEN1, и так далее для всей FIFO структуры.

Если установлен флаг OVIE регистра MOFCRn базовой области, и значение указателя CUR становится равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем RXINP базовой области после завершения операций получения сообщения.

Прерывания приема базовой области FIFO передачи генерируются, если это разрешено битом RXIE регистра MOFCRn.

Режим шлюза

Режим позволяет реализовывать автоматическую передачу информации (через шлюз) между двумя независимыми CAN шинами без участия ЦП.

Шлюз можно сформировать на уровне областей сообщений и осуществлять передачу информации между CAN узлами. Шлюз может быть сформирован между двумя любыми областями сообщений, принадлежащими разным CAN узлам. Количество шлюзов зависит только от количества областей сообщений, допускающих формирование структуры шлюзов.

Режим шлюза активируется записью значения 0100_B в битовое поле MMC регистра MOFCRn области сообщения и инициализирует ее как шлюзовую область-источник. Область сообщения, которая будет являться шлюзовой областью-приемником, выбирается указателем CUR области-источника. Для формирования шлюза достаточно, чтобы область-приемник была корректной (ее флаг MSGVAL = 1_B). Остальные параметры не влияют на возможность осуществления передачи между областями от источника к приемнику.

Шлюзовая область-источник, см. рисунок 23.49, функционирует как обычная область сообщения с тем отличием, что возможны дополнительные действия модуля CAN при приеме и сохранении сообщения в области-приемнике:

1 Если установлен флаг DLCC регистра MOFCRn области-источника, код длины данных MOFCRn.DLC копируется из шлюзовой области-источника в шлюзовую область-приемник.

2 Если установлен флаг IDC регистра MOFCRn области-источника, идентификатор MOARn.ID и расширение идентификатора MOARn.IDE копируются из шлюзовой области-источника в шлюзовую область-приемник.

3 Если установлен флаг DATC регистра MOFCRn области-источника, байты данных, хранящиеся в двух регистрах MODATALn и MODATAHn области-источника, копируются из шлюзовой области-источника в шлюзовую область-приемник. Копируются все 8 байт данных, вне зависимости от значения поля DLC.

4 Если флаг GDFS регистра MOFCRn области-источника установлен, то устанавливается бит запроса передачи TXRQ области-приемника.

5 Устанавливаются флаги RXPND и NEWDAT регистр MOSTATn области-приемника.

6 Если установлен флаг RXIE регистра MOSTATn области-приемника, то генерируется запрос на прерывание.

7 Указатель области CUR регистра MOFGPRn области-источника переводится на следующую область-приемник по правилам FIFO структуры. Сформировать шлюз между областью-источником и одной областью-приемником (значение указателя CUR будет оставаться неизменным) возможно установкой:

MOFGPRn.TOP = MOFGPRn.BOT = MOFGPRn.CUR = область-приемник.

Организация шлюзового соединения «область-источник – область-приемник» аналогична организации FIFO структуры «базовая область – вспомогательная область». Это означает, что имеется возможность формирования шлюза с интегрированным FIFO-приемником. На рисунке 23.49 область сообщения слева – шлюзовая область-источник, а область сообщения справа – шлюзовая область-приемник.

При получении сообщения данных (область-источник является областью для приема, DIR = 0_B) и при получении удаленного запроса (область-источник является областью для передачи) через шлюз используется один и тот же механизм.

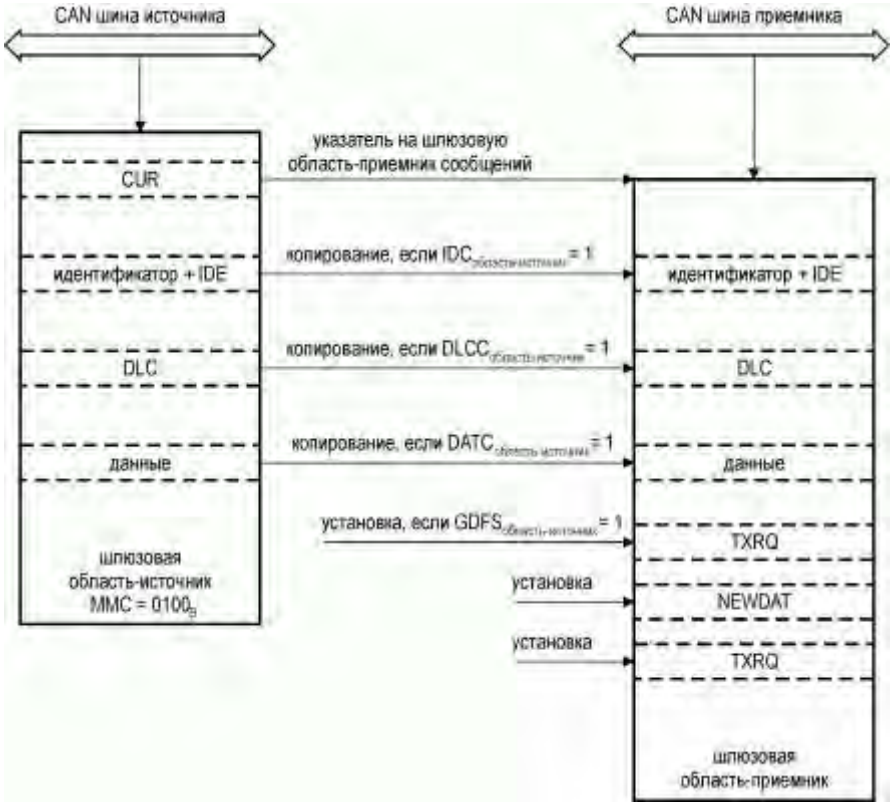


Рисунок 23.49 – Передача через шлюз от источника к приемнику

Удаленные запросы

После получения CAN узлом сообщения удаленного запроса и сохранения его в области сообщения, выставляется бит запроса передачи для ответа на удаленный запрос (отправка сообщения данных) или для автоматического повторения запроса. Если бит FRREN = 0_B регистра MOFCRn области сообщения, в которую было сохранено сообщение удаленного запроса, то выставляется флаг TXRQ регистра MOSTATn этой области.

Если бит FRREN = 1_B области сообщения, в которую было сохранено сообщение удаленного запроса, то выставляется флаг TXRQ той области сообщения, на которую указывает CUR. При этом указатель CUR (принадлежащий регистру MOFGPRn области сообщения, в которую было сохранено сообщение удаленного запроса) не меняет своего значения.

Несмотря на то, что механизм удаленных запросов работает независимо от типа области сообщения, он наиболее полезен при использовании шлюзов, для формирования удаленных запросов на шлюзовой шине источника после получения удаленного запроса на шлюзовой шине приемника. В зависимости от значения бита FRREN шлюзовой области-приемника, есть два варианта обработки удаленного запроса, возникшего с той стороны шлюза, где расположена область-приемник (при условии, что происходит передача из области-источника в область-приемник, т. е. $DIR_{область\text{-}источник} = 0_B$ и $DIR_{область\text{-}приемник} = 1_B$).

1 Обработка запроса, если шлюзовая область-приемник с FRREN = 0_B:

- сообщение удаленного запроса принимается шлюзовой областью-приемником;
- бит TXRQ шлюзовой области-приемника устанавливается автоматически;
- сообщение данных с текущей информацией, хранящейся в области-приемнике, передается на шину приемника.

2 Обработка запроса, если шлюзовая область-приемник с FRREN = 1_B:

- сообщение удаленного запроса принимается шлюзовой областью-приемником;
- бит TXRQ шлюзовой области-источника (должна быть указана в поле CUR регистра области-приемника) устанавливается автоматически;
- сообщение данных передается областью-источником на CAN шину источника;
- получатель удаленного запроса в ответ выдает сообщение данных на шину источника;
- сообщение данных сохраняется в области-источнике;
- сообщение данных копируется в область-приемник (через шлюз);
- выставляется бит TXRQ области-приемника (при условии, что $GDFS_{область\text{-}источник} = 1_B$);
- новые данные, сохраненные в области-приемнике, передаются на шину приемника, в ответ удаленный запрос на шине приемника.

Дополнительная информация по регистрам модуля CAN

Модуль CAN включает три группы регистров, см. рисунки 23.50 – 23.53, таблицы 23.35 – 23.41):

- регистры модуля;
- регистры узлов (для каждого узла своя группа регистров);
- регистры областей сообщений (для каждой области сообщения своя группа регистров).

Каждый регистр состоит из двух слов (старшего и младшего).

Допускается запись в регистры пословно и побайтно.

В зарезервированные биты (обозначены «0») всегда следует записывать «0».

23.9 Регистры модуля CAN

Регистр управления портом CAN узла NPCRx конфигурирует порты прима/передачи. Регистр NPCRx может быть доступен для записи, только если установлен бит CCE регистра NCRx. На рисунке 23.50 представлен формат регистра ID, функциональное назначение полей регистра ID – в таблице 23.35. На рисунке 23.51 представлен формат регистров NPCR0 и NPCR1, функциональное назначение полей регистров NPCR0 и NPCR1 представлено в таблице 23.36.

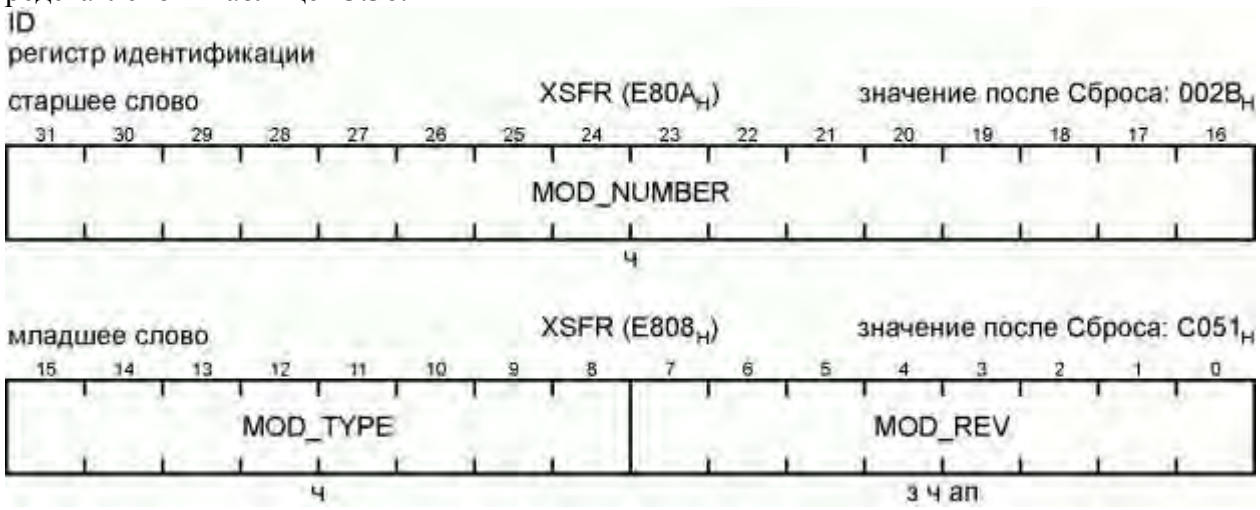


Рисунок 23.50 – Формат регистра ID

Таблица 23.35 – Функциональное назначение полей регистра ID

Поле	Биты	Тип	Описание
MOD_REV	7-0	Чтение	Число модификаций модуля CAN. Число модификаций модуля, начиная со значения 01_H (первая модификация).
MOD_TYPE	15-8	Чтение	Разрядность модуля CAN. По умолчанию, значение равно C0H. Модуль 32-разрядный.
MOD_NUMBER	31-16	Чтение	Номер модуля CAN. Идентификационный номер модуля CAN. По умолчанию – $002B_H$.

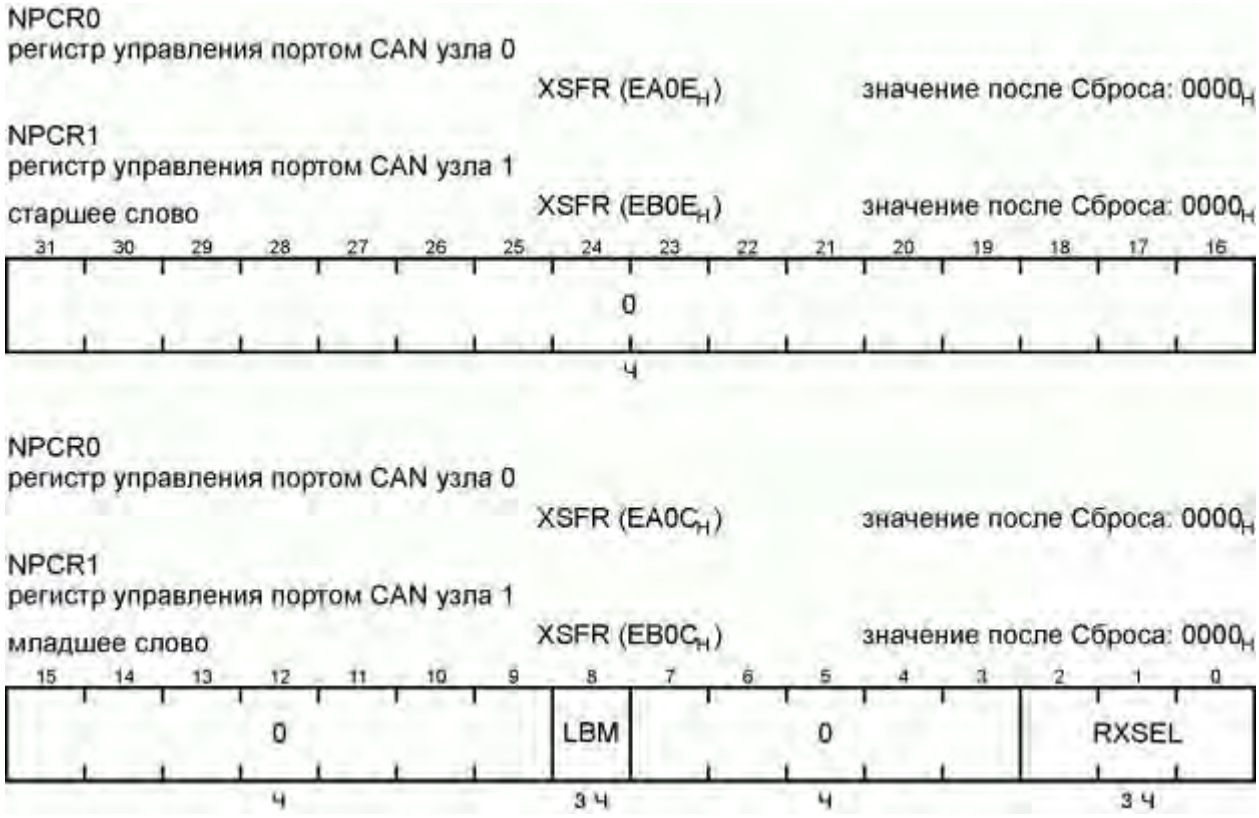


Рисунок 23.51 – Формат регистров NPCR0 и NPCR1

Таблица 23.36 – Функциональное назначение полей регистров NPCR0 и NPCR1

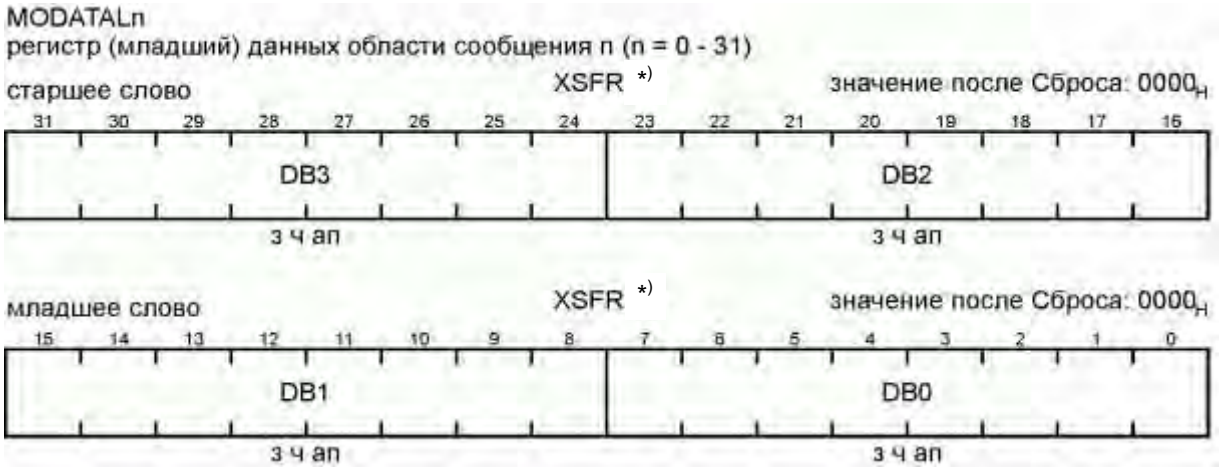
Поле	Биты	Тип	Описание
RXSEL	2-0	Чтение Запись	Выбор входа. Битовое поле выбирает один из выводов CAN узла, для приема сообщений. Кодировка в таблице 23.37.
LBM	8	Чтение Запись	Режим внутренней петли: 0 Режим внутренней петли выключен. 1 Режим внутренней петли включен. В этом режиме CAN узел подсоединяется к внутренней (виртуальной) CAN шине. CAN узлы, для которых включен режим внутренней петли, объединяются виртуальной CAN шиной и могут виртуально взаимодействовать друг с другом. В режиме внутренней петли на внешних выводах CAN узла (соединенных с внешней CAN шиной) поддерживается рецессивный уровень сигнала, т. е. узел неактивен.
0	7-3, 15-9, 31-16	Чтение	Зарезервировано. При чтении возвращает «0». При записи следует писать «0».

Младший регистр данных области сообщения MODATALn хранит 4 младших байта данных области сообщения n. Неиспользуемые байты данных заполняются нулями и не участвуют в операциях приема/передачи.

Коды выбора выводов для приема сообщений для CAN узла 0 и узла 1 представлены в таблице 23.37.

Таблица 23.37 – Коды выбора выводов для приема сообщений для CAN узла 0 и узла 1

Код	Вывод для приема
000_B	P4.5
001_B	P4.7
010_B	P7.6
011_B	P9.2
100_B	P4.4
101_B	P7.4
110_B	P9.0



*) Адреса – в таблице 23.40.

Рисунок 23.52 – Формат регистров MODATALn, где n = 0 – 31

Таблица 23.38 – Функциональное назначение полей регистров MODATALn, где n = 0 – 31

Поле	Биты	Тип	Описание
1	2	3	4
DB0	7-0	Чтение Запись Аппаратное влияние	Нулевой байт данных области сообщения
DB1	15-8	Чтение Запись Аппаратное влияние	Первый байт данных области сообщения
DB2	23-16	Чтение Запись Аппаратное влияние	Второй байт данных области сообщения
DB3	31-24	Чтение Запись Аппаратное влияние	Третий байт данных области сообщения

Старший регистр данных области сообщения MODATAHn, где n = 0 – 31, хранит четыре старших байта данных области сообщения n. Неиспользуемые байты данных заполняются нулями и не участвуют в операциях приема/передачи.

*) Адреса – в таблице 23.40.

Рисунок 23.53 – Формат регистров MODATAHn, где n = 0 – 31

Таблица 23.39 – Функциональное назначение полей регистров MODATAHn, где n = 0 – 31

Поле	Биты	Тип	Описание
DB4	7-0	Чтение Запись Аппаратное влияние	Четвертый байт данных области сообщения
DB5	15-8	Чтение Запись Аппаратное влияние	Пятый байт данных области сообщения
DB6	23-16	Чтение Запись Аппаратное влияние	Шестой байт данных области сообщения
DB7	31-24	Чтение Запись Аппаратное влияние	Седьмой байт данных области сообщения

Таблица 23.40 – Адреса регистров областей сообщений

Область сообщения	Регистры областей сообщений							
	MOFCR		MOFGPR		MOIPR		MOAMR	
	младшее слово	старшее слово	младшее слово	старшее слово	младшее слово	старшее слово	младшее слово	старшее слово
0	$EC00_H$	$EC02_H$	$EC04_H$	$EC06_H$	$EC08_H$	$EC0A_H$	$EC0C_H$	$EC0E_H$
1	$EC20_H$	$EC22_H$	$EC24_H$	$EC26_H$	$EC28_H$	$EC2A_H$	$EC2C_H$	$EC2E_H$
2	$EC40_H$	$EC42_H$	$EC44_H$	$EC46_H$	$EC48_H$	$EC4A_H$	$EC4C_H$	$EC4E_H$
3	$EC60_H$	$EC62_H$	$EC64_H$	$EC66_H$	$EC68_H$	$EC6A_H$	$EC6C_H$	$EC6E_H$
4	$EC80_H$	$EC82_H$	$EC84_H$	$EC86_H$	$EC88_H$	$EC8A_H$	$EC8C_H$	$EC8E_H$
5	$ECA0_H$	$ECA2_H$	$ECA4_H$	$ECA6_H$	$ECA8_H$	$ECAA_H$	$ECAC_H$	$ECAE_H$
6	$ECC0_H$	$ECC2_H$	$ECC4_H$	$ECC6_H$	$ECC8_H$	$ECCA_H$	$ECCC_H$	$ECCE_H$
7	$ECE0_H$	$ECE2_H$	$ECE4_H$	$ECE6_H$	$ECE8_H$	$ECEA_H$	$ECEC_H$	$ECEE_H$
8	$ED00_H$	$ED02_H$	$ED04_H$	$ED06_H$	$ED08_H$	$ED0A_H$	$ED0C_H$	$ED0E_H$
9	$ED20_H$	$ED22_H$	$ED24_H$	$ED26_H$	$ED28_H$	$ED2A_H$	$ED2C_H$	$ED2E_H$
10	$ED40_H$	$ED42_H$	$ED44_H$	$ED46_H$	$ED48_H$	$ED4A_H$	$ED4C_H$	$ED4E_H$
11	$ED60_H$	$ED62_H$	$ED64_H$	$ED66_H$	$ED68_H$	$ED6A_H$	$ED6C_H$	$ED6E_H$
12	$ED80_H$	$ED82_H$	$ED84_H$	$ED86_H$	$ED88_H$	$ED8A_H$	$ED8C_H$	$ED8E_H$
13	$EDA0_H$	$EDA2_H$	$EDA4_H$	$EDA6_H$	$EDA8_H$	$EDAA_H$	$EDAC_H$	$EDAE_H$
14	$EDC0_H$	$EDC2_H$	$EDC4_H$	$EDC6_H$	$EDC8_H$	$EDCA_H$	$EDCC_H$	$EDCE_H$
15	$EDE0_H$	$EDE2_H$	$EDE4_H$	$EDE6_H$	$EDE8_H$	$EDEA_H$	$EDEC_H$	$EDEE_H$
16	$EE00_H$	$EE02_H$	$EE04_H$	$EE06_H$	$EE08_H$	$EE0A_H$	$EE0C_H$	$EE0E_H$
17	$EE20_H$	$EE22_H$	$EE24_H$	$EE26_H$	$EE28_H$	$EE2A_H$	$EE2C_H$	$EE2E_H$
18	$EE40_H$	$EE42_H$	$EE44_H$	$EE46_H$	$EE48_H$	$EE4A_H$	$EE4C_H$	$EE4E_H$
19	$EE60_H$	$EE62_H$	$EE64_H$	$EE66_H$	$EE68_H$	$EE6A_H$	$EE6C_H$	$EE6E_H$
20	$EE80_H$	$EE82_H$	$EE84_H$	$EE86_H$	$EE88_H$	$EE8A_H$	$EE8C_H$	$EE8E_H$
21	$EEA0_H$	$EEA2_H$	$EEA4_H$	$EEA6_H$	$EEA8_H$	$EEAA_H$	$EEAC_H$	$EEAE_H$
22	$EEC0_H$	$EEC2_H$	$EEC4_H$	$EEC6_H$	$EEC8_H$	$EECA_H$	$EECC_H$	$EECE_H$
23	$EEE0_H$	$EEE2_H$	$EEE4_H$	$EEE6_H$	$EEE8_H$	$EEEA_H$	$EEEC_H$	$EEEE_H$
24	$EF00_H$	$EF02_H$	$EF04_H$	$EF06_H$	$EF08_H$	$EF0A_H$	$EF0C_H$	$EF0E_H$
25	$EF20_H$	$EF22_H$	$EF24_H$	$EF26_H$	$EF28_H$	$EF2A_H$	$EF2C_H$	$EF2E_H$
26	$EF40_H$	$EF42_H$	$EF44_H$	$EF46_H$	$EF48_H$	$EF4A_H$	$EF4C_H$	$EF4E_H$
27	$EF60_H$	$EF62_H$	$EF64_H$	$EF66_H$	$EF68_H$	$EF6A_H$	$EF6C_H$	$EF6E_H$
28	$EF80_H$	$EF82_H$	$EF84_H$	$EF86_H$	$EF88_H$	$EF8A_H$	$EF8C_H$	$EF8E_H$
29	$EFA0_H$	$EFA2_H$	$EFA4_H$	$EFA6_H$	$EFA8_H$	$EFAA_H$	$EFAC_H$	$EFAE_H$
30	$EFC0_H$	$EFC2_H$	$EFC4_H$	$EFC6_H$	$EFC8_H$	$EFCA_H$	$EFCC_H$	$EFCE_H$
31	$EFE0_H$	$EFE2_H$	$EFE4_H$	$EFE6_H$	$EFE8_H$	$EFEA_H$	$EFEC_H$	$EFEE_H$

Окончание таблицы 23.40

Область сообщения	Регистры областей сообщений							
	MODATAL		MODATAH		MOAR		MOSTAT, MOCTR	
	младшее слово	старшее слово	младшее слово	старшее слово	младшее слово	старшее слово	младшее слово	старшее слово
0	$EC10_H$	$EC12_H$	$EC14_H$	$EC16_H$	$EC18_H$	$EC1A_H$	$EC1C_H$	$EC1E_H$
1	$EC30_H$	$EC32_H$	$EC34_H$	$EC36_H$	$EC38_H$	$EC3A_H$	$EC3C_H$	$EC3E_H$
2	$EC50_H$	$EC52_H$	$EC54_H$	$EC56_H$	$EC58_H$	$EC5A_H$	$EC5C_H$	$EC5E_H$
3	$EC70_H$	$EC72_H$	$EC74_H$	$EC76_H$	$EC78_H$	$EC7A_H$	$EC7C_H$	$EC7E_H$
4	$EC90_H$	$EC92_H$	$EC94_H$	$EC96_H$	$EC98_H$	$EC9A_H$	$EC9C_H$	$EC9E_H$
5	$ECB0_H$	$ECB2_H$	$ECB4_H$	$ECB6_H$	$ECB8_H$	$ECBA_H$	$ECBC_H$	$ECBE_H$
6	$ECD0_H$	$ECD2_H$	$ECD4_H$	$ECD6_H$	$ECD8_H$	$ECDA_H$	$ECDC_H$	$ECDE_H$
7	$ECF0_H$	$ECF2_H$	$ECF4_H$	$ECF6_H$	$ECF8_H$	$ECFA_H$	$ECFC_H$	$ECFE_H$
8	$ED10_H$	$ED12_H$	$ED14_H$	$ED16_H$	$ED18_H$	$ED1A_H$	$ED1C_H$	$ED1E_H$
9	$ED30_H$	$ED32_H$	$ED34_H$	$ED36_H$	$ED38_H$	$ED3A_H$	$ED3C_H$	$ED3E_H$
10	$ED50_H$	$ED52_H$	$ED54_H$	$ED56_H$	$ED58_H$	$ED5A_H$	$ED5C_H$	$ED5E_H$
11	$ED70_H$	$ED72_H$	$ED74_H$	$ED76_H$	$ED78_H$	$ED7A_H$	$ED7C_H$	$ED7E_H$
12	$ED90_H$	$ED92_H$	$ED94_H$	$ED96_H$	$ED98_H$	$ED9A_H$	$ED9C_H$	$ED9E_H$
13	$EDB0_H$	$EDB2_H$	$EDB4_H$	$EDB6_H$	$EDB8_H$	$EDBA_H$	$EDBC_H$	$EDBE_H$
14	$EDD0_H$	$EDD2_H$	$EDD4_H$	$EDD6_H$	$EDD8_H$	$EDDA_H$	$EDDC_H$	$EDDE_H$
15	$EDF0_H$	$EDF2_H$	$EDF4_H$	$EDF6_H$	$EDF8_H$	$EDFA_H$	$EDFC_H$	$EDFE_H$
16	$EE10_H$	$EE12_H$	$EE14_H$	$EE16_H$	$EE18_H$	$EE1A_H$	$EE1C_H$	$EE1E_H$
17	$EE30_H$	$EE32_H$	$EE34_H$	$EE36_H$	$EE38_H$	$EE3A_H$	$EE3C_H$	$EE3E_H$
18	$EE50_H$	$EE52_H$	$EE54_H$	$EE56_H$	$EE58_H$	$EE5A_H$	$EE5C_H$	$EE5E_H$
19	$EE70_H$	$EE72_H$	$EE74_H$	$EE76_H$	$EE78_H$	$EE7A_H$	$EE7C_H$	$EE7E_H$
20	$EE90_H$	$EE92_H$	$EE94_H$	$EE96_H$	$EE98_H$	$EE9A_H$	$EE9C_H$	$EE9E_H$
21	$EEB0_H$	$EEB2_H$	$EEB4_H$	$EEB6_H$	$EEB8_H$	$EEBA_H$	$EEBC_H$	$EEBE_H$
22	$EED0_H$	$EED2_H$	$EED4_H$	$EED6_H$	$EED8_H$	$EEDA_H$	$EEDC_H$	$EEDE_H$
23	$EEF0_H$	$EEF2_H$	$EEF4_H$	$EEF6_H$	$EEF8_H$	$EEFA_H$	$EEFC_H$	$EEFE_H$
24	$EF10_H$	$EF12_H$	$EF14_H$	$EF16_H$	$EF18_H$	$EF1A_H$	$EF1C_H$	$EF1E_H$
25	$EF30_H$	$EF32_H$	$EF34_H$	$EF36_H$	$EF38_H$	$EF3A_H$	$EF3C_H$	$EF3E_H$
26	$EF50_H$	$EF52_H$	$EF54_H$	$EF56_H$	$EF58_H$	$EF5A_H$	$EF5C_H$	$EF5E_H$
27	$EF70_H$	$EF72_H$	$EF74_H$	$EF76_H$	$EF78_H$	$EF7A_H$	$EF7C_H$	$EF7E_H$
28	$EF90_H$	$EF92_H$	$EF94_H$	$EF96_H$	$EF98_H$	$EF9A_H$	$EF9C_H$	$EF9E_H$
29	$EFB0_H$	$EFB2_H$	$EFB4_H$	$EFB6_H$	$EFB8_H$	$EFBA_H$	$EFBC_H$	$EFBE_H$
30	$EFD0_H$	$EFD2_H$	$EFD4_H$	$EFD6_H$	$EFD8_H$	$EFDA_H$	$EFDC_H$	$EFDE_H$
31	$EFF0_H$	$EFF2_H$	$EFF4_H$	$EFF6_H$	$EFF8_H$	$EFFA_H$	$EFFC_H$	$EFFE_H$

Значения регистров MOCTRn и MOSTATn после сброса, где n = 0 – 31, представлены в таблице 23.41.

Таблица 23.41 – Значения регистров MOCTRn и MOSTATn после сброса

Номер области сообщения n	Регистры MOCTRn и MOSTATn	
	младшее слово	старшее слово
0	0100_H	0000_H
1	0200_H	0000_H
2	0301_H	0000_H
3	0402_H	0000_H
4	0503_H	0000_H
5	0604_H	0000_H
6	0705_H	0000_H
7	0806_H	0000_H
8	0907_H	0000_H
9	$0A08_H$	0000_H
10	$0B09_H$	0000_H
11	$0C0A_H$	0000_H
12	$0D0B_H$	0000_H
13	$0E0C_H$	0000_H
14	$0F0D_H$	0000_H
15	$100E_H$	0000_H
16	$111F_H$	0000_H
17	1210_H	0000_H
18	1311_H	0000_H
19	1412_H	0000_H
20	1513_H	0000_H
21	1614_H	0000_H
22	1715_H	0000_H
23	1816_H	0000_H
24	1917_H	0000_H
25	$1A18_H$	0000_H
26	$1B19_H$	0000_H
27	$1C1A_H$	0000_H
28	$1D1B_H$	0000_H
29	$1E1C_H$	0000_H
30	$1F1D_H$	0000_H
31	$201E_H$	0000_H

В приложении Ж дана таблица Ж.1 регистров области XSFR, принадлежащих к блоку интерфейса CAN, с указанием физических адресов и значений после сброса (RESET).

23.10 Интерфейс модуля CAN

Внутренние соединения

16 выходных линий прерываний CAN_INT0, …, CAN_INT15 на рисунке 23.54 модуля CAN соединены с блоком управления прерываниями.

Внешние соединения

Выводы модуля CAN соединяются с портами ввода-вывода контроллера. Узлы модуля CAN могут принимать сигналы с любого, выбранного из семи выводов контроллера, для чего каждый узел имеет свой мультиплексор, управляемый регистром NPCR0/NPCR1.

Каждый узел может передавать информацию на несколько выводов контроллера, если это необходимо, см. рисунок 23.54.

Кроме этого, имеется механизм для выдачи на соответствующие выводы контроллера логического «И» выходных сигналов CAN узлов, управляемый регистрами альтернативных функций.

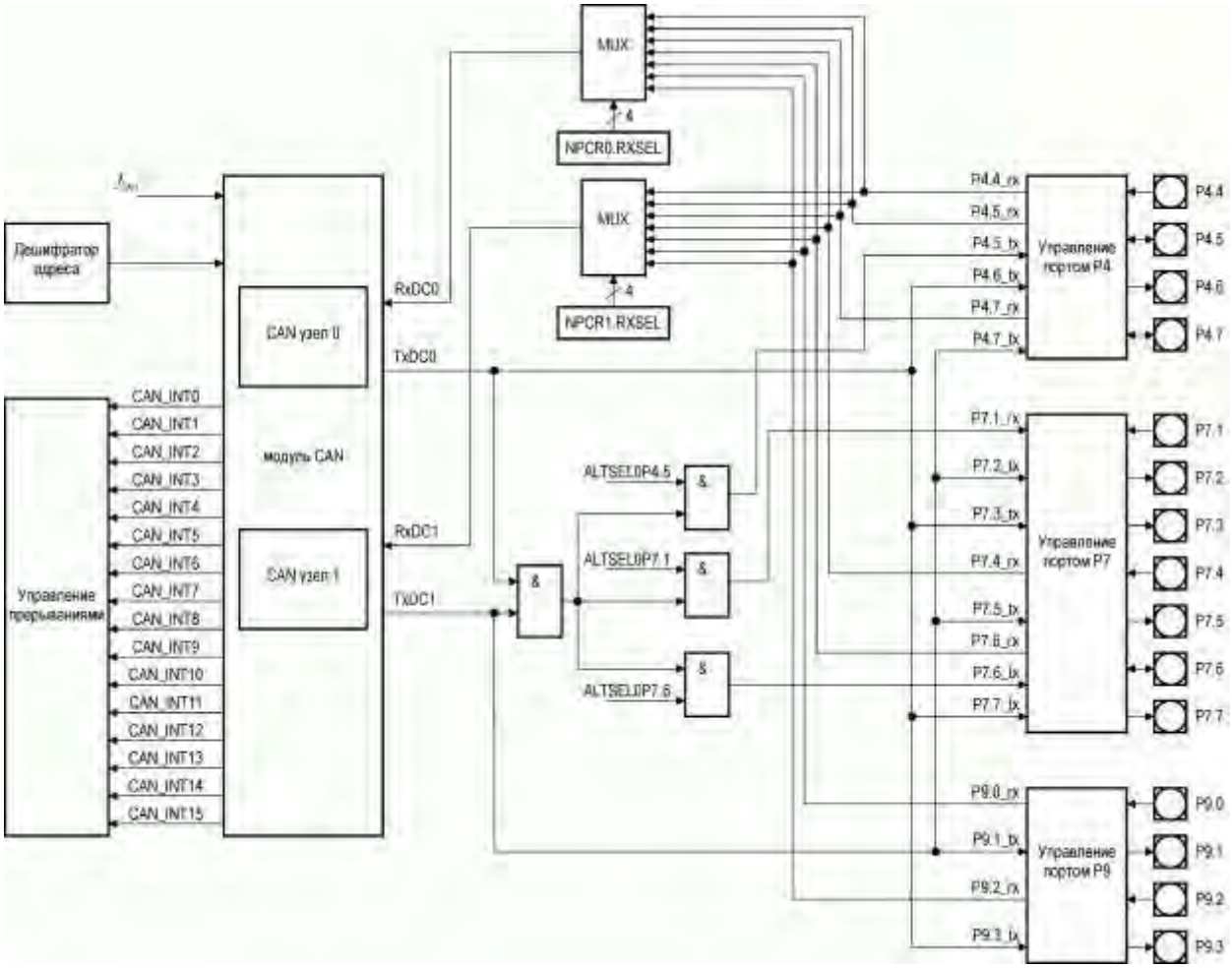


Рисунок 23.54 – Интерфейс модуля CAN

24 Заключение

В настоящем руководстве КФДЛ.431295.029 приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения микроконтроллера 1887ВЕ3Т, который представляет собой 16-разрядную однокристальную микро-ЭВМ с RISC-архитектурой и памятью типа Flash.

Микроконтроллер предназначен для применения в автомобильных и промышленных встраиваемых цифровых системах управления, везде, где требуются сбор, обработка и обмен данными между электронными системами управления, наблюдения или измерения, а также в системах связи. Все значения электрических параметров ИС приведены в техническом условии АЕЯР.431280.674 на изделие. Значения параметров, приведенные в настоящем руководстве, являются справочными.

Руководство КФДЛ.431295.029 может служить практическим пособием по применению микроконтроллеров 1887ВЕ3Т для разработчиков систем на основе ИС 1887ВЕ3Т и программистов.

Приложение А

(обязательное)

Таблица векторов прерываний, упорядоченных по номеру прерывания

Таблица А.1 – Векторы прерываний, упорядоченные по номеру прерывания

Номер прерывания, hex	Локализация вектора, hex	Источник прерывания	Регистр управления
1	2	3	4
00	0000	RESET	–
01	0004	–	–
02	0008	NMI (не маскируемое прерывание)	–
03	000C	–	–
04	0010	Переполнение стека	–
05	0014	–	–
06	0018	Опустошение стека	–
07	001C	–	–
08	0020	Программный останов	–
09	0024	–	–
0A	0028	Прерывание класса B	–
0B	002C	–	–
0C	0030	–	–
0D	0034	–	–
0E	0038	–	–
0F	003C	–	–
10	0040	Регистр сравнения 0 (CAPCOM1)	CC0IC
11	0044	Регистр сравнения 1 (CAPCOM1)	CC1IC
12	0048	Регистр сравнения 2 (CAPCOM1)	CC2IC
13	004C	Регистр сравнения 3 (CAPCOM1)	CC3IC
14	0050	Регистр сравнения 4 (CAPCOM1)	CC4IC
15	0054	Регистр сравнения 5 (CAPCOM1)	CC5IC
16	0058	Регистр сравнения 6 (CAPCOM1)	CC6IC
17	005C	Регистр сравнения 7 (CAPCOM1)	CC7IC
18	0060	Регистр сравнения 8 (CAPCOM1)	CC8IC
19	0064	Регистр сравнения 9 (CAPCOM1)	CC9IC
1A	0068	Регистр сравнения 10 CAPCOM1)	CC10IC
1B	006C	Регистр сравнения 11 (CAPCOM1)	CC11IC
1C	0070	Регистр сравнения 12 (CAPCOM1)	CC12IC
1D	0074	Регистр сравнения 13 (CAPCOM1)	CC13IC
1E	0078	Регистр сравнения 14 (CAPCOM1)	CC14IC
1F	007C	Регистр сравнения 15 (CAPCOM1)	CC15IC
20	0080	Таймер 0 (CAPCOM1)	T0IC
21	0084	Таймер 1 (CAPCOM1)	T1IC
22	0088	Таймер T2 (GPT1, GPT2)	T2IC
23	008C	Таймер T3 (GPT1, GPT2)	T3IC
24	0090	Таймер T4 (GPT1, GPT2)	T4IC
25	0094	Таймер T5 (GPT1, GPT2)	T5IC
26	0098	Таймер T6 (GPT1, GPT2)	T6IC
27	009C	Захват/перезагрузка (GPT1, GPT2)	CRIC
28	00A0	Завершение преобразования АЦП	ASC_CIC
29	00A4	Ошибка АЦП	ADC_EIC
2A	00A8	Передача (ASC0)	S0TIC

Продолжение таблицы А.1

1	2	3	4
2B	00AC	Прием (ASC0)	S0RIC
2C	00B0	Ошибка (ASC0)	S0EIC
2D	00B4	Передача (SSC0)	SSC0TIC
2E	00B8	Прием (SSC0)	SSC0RIC
2F	00BC	Ошибка (SSC0)	SSC0EIC
30	00C0	Регистр сравнения 16 (CAPCOM2)	CC16IC
31	00C4	Регистр сравнения 17 (CAPCOM2)	CC17IC
32	00C8	Регистр сравнения 18 (CAPCOM2)	CC18IC
33	00CC	Регистр сравнения 19 (CAPCOM2)	CC19IC
34	00D0	Регистр сравнения 20 (CAPCOM2)	CC20IC
35	00D4	Регистр сравнения 21 (CAPCOM2)	CC21IC
36	00D8	Регистр сравнения 22 (CAPCOM2)	CC22IC
37	00DC	Регистр сравнения 23 (CAPCOM2)	CC23IC
38	00E0	Регистр сравнения 24 (CAPCOM2)	CC24IC
39	00E4	Регистр сравнения 25 (CAPCOM2)	CC25IC
3A	00E8	Регистр сравнения 26 (CAPCOM2)	CC26IC
3B	00EC	Регистр сравнения 27 (CAPCOM2)	CC27IC
3C	00F0	Регистр сравнения 28 (CAPCOM2)	CC28IC
3D	00F4	Таймер 7 (CAPCOM2)	T7IC
3E	00F8	Таймер 8 (CAPCOM2)	T8IC
3F	00FC	Программный запрос 15	IRQ15
40	0100	I2C	I2C_DIC
41	0104	Завершение преобразования АЦП2	ADC2_CIC
42	0108	Ошибка АЦП2	ADC2_EIC
43	010C	Нахождение частоты PLL	PLLLOCK_IC
44	0110	Регистр сравнения 29 (CAPCOM2)	CC29IC
45	0114	Регистр сравнения 30 (CAPCOM2)	CC30IC
46	0118	Регистр сравнения 31 (CAPCOM2)	CC31IC
47	011C	Буфер передатчика ASC0	S0TBIC
48	0120	Передача ASC1	S1TIC
49	0124	Прием ASC1	S1RIC
4A	0128	Ошибка ASC1	S1EIC
4B	012C	Сторожевой таймер WDT	WDTIC
4C	0130	Окончание PEC передач	EOPIC
4D	0134	Таймер 12 (CAPCOM6)	CCU6_T12IC
4E	0138	Таймер 13 (CAPCOM6)	CCU6_T13IC
4F	013C	Датчик (CAPCOM6)	CCU6_EIC
50	0140	CAPCOM6	CCU6_IC
51	0144	Передача SSC1	SSC1TIC
52	0148	Прием SSC1	SSC1RIC
53	014C	Ошибка SSC1	SSC1EIC
54	0150	CAN0	CAN_0IC
55	0154	CAN1	CAN_1IC
56	0158	CAN2	CAN_2IC
57	015C	CAN3	CAN_3IC
58	0160	Программный запрос 72	IRQ 72
59	0164	CAN4	CAN_4IC
5A	0168	CAN5	CAN_5IC

Окончание таблицы А.1

1	2	3	4
5B	016C	CAN6	CAN_6IC
5C	0170	CAN7	CAN_7IC
5D	0174	RTC	RTC_IC
5E	0178	Буфер передатчика ASC1	S1TBIC
5F	017C	Программный запрос 79	IRQ 79
60	0180	CAN8	CAN_8IC
61	0184	CAN9	CAN_9IC
62	0188	CAN10	CAN_10IC
63	018C	CAN11	CAN_11IC
64	0190	CAN12	CAN_12IC
65	0194	CAN13	CAN_13IC
66	0198	CAN14	CAN_14IC
67	019C	CAN15	CAN_15IC
68	01A0	Регистр сравнения RTCCMP1	RTC_IC1
69	01A4	Регистр сравнения RTCCMP2	RTC_IC2
6A	01A8	Регистр сравнения RTCCMP3	RTC_IC3
6B	01AC	Потеря частоты PLL	PLLUNLOCK_IC
6C	01B0	Программный запрос 92	IRQ 92
6D	01B4	Программный запрос 93	IRQ 93
6E	01B8	Программный запрос 94	IRQ 94
6F	01BC	Программный запрос 95	IRQ 95
70	01C0	Программный запрос 96	IRQ 96
71	01C4	Программный запрос 97	IRQ 97
72	01C8	Программный запрос 98	IRQ 98
73	01CC	Программный запрос 99	IRQ 99
74	01D0	Программный запрос 100	IRQ 100
75	01D4	Программный запрос 101	IRQ 101
76	01D8	Программный запрос 102	IRQ 102
77	01DC	Программный запрос 103	IRQ 103
78	01E0	Программный запрос 104	IRQ 104
79	01E4	Программный запрос 105	IRQ 105
7A	01E8	Программный запрос 106	IRQ 106
7B	01EC	Программный запрос 107	IRQ 107
7C	01F0	Программный запрос 108	IRQ 108
7D	01F4	Программный запрос 109	IRQ 109
7E	01F8	Программный запрос 110	IRQ 110
7F	01FC	Программный запрос 111	IRQ 111

Приложение Б
(обязательное)
Таблица регистров областей SFR и ESFR, упорядоченных по физическим адресам

Таблица Б.1

Физический адрес (16 бит), hex	Название	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET)
1	2	3	4	5	6
F000	QX0	ESFR	00	Регистр смещения 0 для указателя адреса IDX0/1 (MAC)	0000
F002	QX1	ESFR	01	Регистр смещения 1 для указателя адреса IDX0/1 (MAC)	0000
F004	QR0	ESFR	02	Регистр смещения 0 для указателя адреса GPRi (MAC)	0000
F006	QR1	ESFR	03	Регистр смещения 1 для указателя адреса GPRi (MAC)	0000
F008	–	ESFR	04	Зарезервировано	–
F00A	–	ESFR	05	Зарезервировано	–
F00C	CPUID	ESFR	06	Регистр идентификации ЦПУ	0420
F00E	–	ESFR	07	Зарезервировано	–
F010	–	ESFR	08	Зарезервировано	–
F012	–	ESFR	09	Зарезервировано	–
F014	XADRS1	ESFR	0A	Регистр 1 выбора адреса (XBUS)	0000
F016	XADRS2	ESFR	0B	Регистр 2 выбора адреса (XBUS)	0000
F018	XADRS3	ESFR	0C	Регистр 3 выбора адреса (XBUS)	0000
F01A	XADRS4	ESFR	0D	Регистр 4 выбора адреса (XBUS)	0000
F01C	XADRS5	ESFR	0E	Регистр 5 выбора адреса (XBUS)	0000
F01E	XADRS6	ESFR	0F	Регистр 6 выбора адреса (XBUS)	0000
F020	–	ESFR	10	Зарезервировано	–
F022	–	ESFR	11	Зарезервировано	–
F024	XPERCON	ESFR	12	Регистр управления X-периферией (XBUS)	0000
F026	–	ESFR	13	Зарезервировано	–
F028	–	ESFR	14	Зарезервировано	–
F02A	–	ESFR	15	Зарезервировано	–
F02C	–	ESFR	16	Зарезервировано	–
F02E	–	ESFR	17	Зарезервировано	–
F030	RTCCST	ESFR	18	Регистр управления и статуса (RTC)	--00
F032	RTUDST	ESFR	19	Регистр состояния обновлений (RTC)	--00
F034	RTCEIST	ESFR	1A	Регистр состояния прерываний (RTC)	--00
F036	RTCIEN	ESFR	1B	Регистр разрешения прерываний (RTC)	--00
F038	RTPRD	ESFR	1C	Регистр прескалера (RTC)	0000
F03A	RTCRD_hi	ESFR	1D	Регистр реального времени, старшие разряды (RTC)	0000
F03C	RTCRD_lo	ESFR	1E	Регистр реального времени, младшие разряды (RTC)	0001

Продолжение таблицы Б.1

1	2	3	4	5	6
F03E	RTCLD_hi	ESFR	1F	Перезагружаемый регистр, старшие разряды (RTC)	0000
F040	RTCLD_lo	ESFR	20	Перезагружаемый регистр, младшие разряды (RTC)	0000
F042	RTCCMP1_hi	ESFR	21	Регистр сравнения 1, старшие разряды (RTC)	0000
F044	RTCCMP1_lo	ESFR	22	Регистр сравнения 1, младшие разряды (RTC)	0200
F046	RTCCMP2_hi	ESFR	23	Регистр сравнения 2, старшие разряды (RTC)	FFFF
F048	RTCCMP2_lo	ESFR	24	Регистр сравнения 2, младшие разряды (RTC)	FFFF
F04A	RTCCMP3_hi	ESFR	25	Регистр сравнения 3, старшие разряды (RTC)	FFFF
F04C	RTCCMP3_lo	ESFR	26	Регистр сравнения 3, младшие разряды (RTC)	FFFF
F04E	–	ESFR	27	Зарезервировано	–
F050	T7	ESFR	28	Регистр таймера T7 (CAPCOM2)	0000
F052	T8	ESFR	29	Регистр таймера T8 (CAPCOM2)	0000
F054	T7REL	ESFR	2A	Регистр загрузки таймера T7 (CAPCOM2)	0000
F056	T8REL	ESFR	2B	Регистр загрузки таймера T8 (CAPCOM2)	0000
F058	–	ESFR	2C	Зарезервировано	–
F05A	SSC1TB	ESFR	2D	Буферный регистр передатчика (SSC1)	0000
F05C	SSC1RB	ESFR	2E	Буферный регистр приемника (SSC1)	0000
F05E	SSC1BR	ESFR	2F	Регистр скорости пересылки (SSC1)	0000
F060	–	ESFR	30	Зарезервировано	–
F062	CC1_IOC	ESFR	31	Регистр управления вводом/выводом (CAPCOM1)	0000
F064	-	ESFR	32	Зарезервировано	–
F066	CC2_IOC	ESFR	33	Регистр управления вводом/выводом (CAPCOM2)	0000
F068	COMDATA	ESFR	34	Регистр режима коммуникации Cerberus (OCDS)	0000
F06A	RWDATA	ESFR	35	Регистр данных режима RW Cerberus (OCDS)	0000
F06C	IOSR	ESFR	36	Регистр статуса Cerberus (OCDS)	0000
F06E	–	ESFR	37	Зарезервировано	–
F070	–	ESFR	38	Зарезервировано	–
F072	–	ESFR	39	Зарезервировано	–
F074	–	ESFR	3A	Зарезервировано	–
F076	–	ESFR	3B	Зарезервировано	–
F078	–	ESFR	3C	Зарезервировано	–
F07A	–	ESFR	3D	Зарезервировано	–

Продолжение таблицы Б.1

1	2	3	4	5	6
F07C	–	ESFR	3E	Зарезервировано	–
F07E	–	ESFR	3F	Зарезервировано	–
F094	–	ESFR	4A	Зарезервировано	–
F096	–	ESFR	4B	Зарезервировано	–
F098	–	ESFR	4C	Зарезервировано	–
F09A	–	ESFR	4D	Зарезервировано	–
F09C	–	ESFR	4E	Зарезервировано	–
F09E	–	ESFR	4F	Зарезервировано	–
F0A0	ADC_DAT2	ESFR	50	Регистр результата АЦП инжектированного канала	0000
F0A2	ADC2_DAT2	ESFR	51	Регистр результата АЦП2 инжектированного канала	0000
F0A4	–	ESFR	52	Зарезервировано	–
F0A6	–	ESFR	53	Зарезервировано	–
F0A8	–	ESFR	54	Зарезервировано	–
F0AA	–	ESFR	55	Зарезервировано	–
F0AC	–	ESFR	56	Зарезервировано	–
F0AE	–	ESFR	57	Зарезервировано	–
F0B0	SSC0TB	ESFR	58	Буферный регистр передатчика (SSC0)	0000
F0B2	SSC0RB	ESFR	59	Буферный регистр приемника (SSC0)	0000
F0B4	SSC0BR	ESFR	5A	Регистр скорости пересылки (SSC0)	0000
F0B6	–	ESFR	5B	Зарезервировано	–
F0B8	–	ESFR	5C	Зарезервировано	–
F0BA	–	ESFR	5D	Зарезервировано	–
F0BC	–	ESFR	5E	Зарезервировано	–
F0BE	–	ESFR	5F	Зарезервировано	–
F0C0	–	ESFR	60	Зарезервировано	–
F0C2	–	ESFR	61	Зарезервировано	–
F0C4	–	ESFR	62	Зарезервировано	–
F0C6	–	ESFR	63	Зарезервировано	–
F0C8	–	ESFR	64	Зарезервировано	–
F0CA	–	ESFR	65	Зарезервировано	–
F0CC	–	ESFR	66	Зарезервировано	–
F0CE	–	ESFR	67	Зарезервировано	–
F0D0	–	ESFR	68	Зарезервировано	–
F0D2	–	ESFR	69	Зарезервировано	–
F0D4	–	ESFR	6A	Зарезервировано	–
F0D6	–	ESFR	6B	Зарезервировано	–
F0D8	DTIDR	ESFR	6C	Регистр идентификации задачи системы отладки (OCDS)	0000
F0DA	SMBSDA	ESFR	6D	Сдвиговый регистр данных (I2C)	--XX
F0DC	SMBST	ESFR	6E	Регистр статуса (I2C)	--00
F0DE	SMBCST	ESFR	6F	Регистр управления и статуса (I2C)	--00
F0E0	SMBCTL1	ESFR	70	Регистр управления 1 (I2C)	--00

Продолжение таблицы Б.1

1	2	3	4	5	6
F0E2	SMBADDR	ESFR	71	Регистр собственного адреса (I2C)	--00
F0E4	SMBCTL2	ESFR	72	Регистр управления 2 (I2C)	--00
F0E6	SMBTOPR	ESFR	73	Регистр делителя частоты времени ожидания (I2C)	--00
F0E8	SMBCTL3	ESFR	74	Регистр управления 3 (I2C)	--00
F0EA	CFG	ESFR	75	Регистр конфигурации (I2C)	--00
F0EC	DCMPSP	ESFR	76	Регистр выбора и программирования для DCMPx (OCDS)	0000
F0EE	DCMPDP	ESFR	77	Регистр данных программирования для DCMPx (OCDS)	0000
F0F0	DTREVT	ESFR	78	Регистр управления комбинациями аппаратных отладочных событий (OCDS)	0000
F0F2	DEXEVT	ESFR	79	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)	0000
F0F4	DSWEVT	ESFR	7A	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)	0000
F0F6	–	ESFR	7B	Зарезервировано	–
F0F8	DIP	ESFR	7C	Регистр указателя машинной команды	0000
F0FA	DIPX	ESFR	7D	Регистр расширенного указателя инструкций	3000
F0FC	DBGSR	ESFR	7E	Регистр состояния отладки (OCDS)	0000
F0FE	–	ESFR	7F	Зарезервировано	–
F100	DP0L	ESFR-b	80	Регистр выбора направления порта P0L (порт P0)	0000
F102	DP0H	ESFR-b	81	Регистр выбора направления порта P0H (порт P0)	0000
F104	DP1L	ESFR-b	82	Регистр выбора направления порта P1L (порт P1)	0000
F106	DP1H	ESFR-b	83	Регистр выбора направления порта P1H (порт P1)	0000
F108	RP0H	ESFR-b	84	Регистр начальной конфигурации внешней шины	--XX
F10A	–	ESFR-b	85	Зарезервировано	–
F10C	–	ESFR-b	86	Зарезервировано	–
F10E	–	ESFR-b	87	Зарезервировано	–
F110	–	ESFR-b	88	Зарезервировано	–
F112	–	ESFR-b	89	Зарезервировано	–
F114	XBCON1	ESFR-b	8A	Регистр 1 управления 1 (XBUS)	0000
F116	XBCON2	ESFR-b	8B	Регистр 2 управления 2 (XBUS)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
F118	XBCON3	ESFR-b	8C	Регистр 3 управления 3 (XBUS)	0000
F11A	XBCON4	ESFR-b	8D	Регистр 4 управления 4 (XBUS)	0000
F11C	XBCON5	ESFR-b	8E	Регистр 5 управления 5 (XBUS)	0000
F11E	XBCON6	ESFR-b	8F	Регистр 6 управления 6 (XBUS)	0000
F120	CCU6_IC	ESFR-b	90	Регистр управления прерываниями (CAPCOM6)	--00
F122	SSC1TIC	ESFR-b	91	Регистр контроля прерывания по передаче SSC1	0000
F124	SSC1RIC	ESFR-b	92	Регистр контроля прерывания по приему SSC1	0000
F126	SSC1EIC	ESFR-b	93	Регистр контроля прерывания по ошибке SSC1	0000
F128	–	ESFR-b	94	Зарезервировано	–
F12A	–	ESFR-b	95	Зарезервировано	–
F12C	–	ESFR-b	96	Зарезервировано	–
F12E	–	ESFR-b	97	Зарезервировано	–
F130	–	ESFR-b	98	Зарезервировано	–
F132	–	ESFR-b	99	Зарезервировано	–
F134	–	ESFR-b	9A	Зарезервировано	–
F136	–	ESFR-b	9B	Зарезервировано	–
F138	–	ESFR-b	9C	Зарезервировано	–
F13A	RTC_IC	ESFR-b	9D	Регистр контроля прерываний RTC	0000
F13C	S1TBIC	ESFR-b	9E	Регистр управления прерыванием по опустошению буфера передатчика ASC1	0000
F13E	–	ESFR-b	9F	Зарезервировано	–
F140	–	ESFR-b	A0	Зарезервировано	–
F142	–	ESFR-b	A1	Зарезервировано	–
F144	–	ESFR-b	A2	Зарезервировано	–
F146	–	ESFR-b	A3	Зарезервировано	–
F148	–	ESFR-b	A4	Зарезервировано	–
F14A	–	ESFR-b	A5	Зарезервировано	–
F14C	–	ESFR-b	A6	Зарезервировано	–
F14E	–	ESFR-b	A7	Зарезервировано	–
F150	RTC_IC1	ESFR-b	A8	Регистр контроля прерывания 1 RTC	0000
F152	RTC_IC2	ESFR-b	A9	Регистр контроля прерывания 2 RTC	0000
F154	RTC_IC3	ESFR-b	AA	Регистр контроля прерывания 3 RTC	0000
F156	PLLUNLOCK_IC	ESFR-b	AB	Регистр управления прерыванием по потере частоты PLL	0000
F158	–	ESFR-b	AC	Зарезервировано	–
F15A	–	ESFR-b	AD	Зарезервировано	–
F15C	–	ESFR-b	AE	Зарезервировано	–
F15E	–	ESFR-b	AF	Зарезервировано	–

Продолжение таблицы Б.1

1	2	3	4	5	6
F160	CC16IC	ESFR-b	B0	Регистр управления прерываниями 16 (CAPCOM2)	0000
F162	CC17IC	ESFR-b	B1	Регистр управления прерываниями 17 (CAPCOM2)	0000
F164	CC18IC	ESFR-b	B2	Регистр управления прерываниями 18 (CAPCOM2)	0000
F166	CC19IC	ESFR-b	B3	Регистр управления прерываниями 19 (CAPCOM2)	0000
F168	CC20IC	ESFR-b	B4	Регистр управления прерываниями 20 (CAPCOM2)	0000
F16A	CC21IC	ESFR-b	B5	Регистр управления прерываниями 21 (CAPCOM2)	0000
F16C	CC22IC	ESFR-b	B6	Регистр управления прерываниями 22 (CAPCOM2)	0000
F16E	CC23IC	ESFR-b	B7	Регистр управления прерываниями 23 (CAPCOM2)	0000
F170	CC24IC	ESFR-b	B8	Регистр управления прерываниями 24 (CAPCOM2)	0000
F172	CC25IC	ESFR-b	B9	Регистр управления прерываниями 25 (CAPCOM2)	0000
F174	CC26IC	ESFR-b	BA	Регистр управления прерываниями 26 (CAPCOM2)	0000
F176	CC27IC	ESFR-b	BB	Регистр управления прерываниями 27 (CAPCOM2)	0000
F178	CC28IC	ESFR-b	BC	Регистр управления прерываниями 28 (CAPCOM2)	0000
F17A	T7IC	ESFR-b	BD	Регистр управления прерываниями таймера T7 (CAPCOM2)	--00
F17C	T8IC	ESFR-b	BE	Регистр управления прерываниями таймера T8 (CAPCOM2)	--00
F17E	–	ESFR-b	BF	Зарезервировано	–
F180	EOPIC	ESFR-b	C0	Регистр контроля за прерыванием по завершению PEC передачи	0000
F182	S1TIC	ESFR-b	C1	Регистр управления прерыванием по передаче (ASC1)	0000
F184	CC29IC	ESFR-b	C2	Регистр управления прерываниями 29 (CAPCOM2)	0000
F186	I2C_DIC	ESFR-b	C3	Регистр управления прерыванием (I2C)	--00
F188	CCU6_EIC	ESFR-b	C4	Регистр управления прерываниями по ошибке (CAPCOM6)	--00
F18A	S1RIC	ESFR-b	C5	Регистр управления прерыванием по приему (ASC1)	0000
F18C	CC30IC	ESFR-b	C6	Регистр управления прерываниями 30 (CAPCOM2)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
F18E	ADC2_CIC	ESFR-b	C7	Регистр контроля прерывания по завершению преобразования (АЦП2)	0000
F190	CCU6_T12IC	ESFR-b	C8	Регистр управления прерываниями таймера T12 (CAPCOM6)	--00
F192	S1EIC	ESFR-b	C9	Регистр управления прерыванием по ошибке (ASC1)	0000
F194	CC31IC	ESFR-b	CA	Регистр управления прерываниями 31 (CAPCOM2)	0000
F196	ADC2_EIC	ESFR-b	CB	Регистр контроля прерывания по ошибке (АЦП2)	0000
F198	CCU6_T13IC	ESFR-b	CC	Регистр управления прерываниями таймера T13 (CAPCOM6)	--00
F19A	WDTIC	ESFR-b	CD	Регистр управления прерыванием сторожевого таймера (WDT)	0000
F19C	S0TBIC	ESFR-b	CE	Регистр контроля прерывания по опустошению буфера передачи (ASC0)	0000
F19E	PLLLOCK_IC	ESFR-b	CF	Регистр управления прерыванием по нахождению частоты PLL	0000
F1A0	ADC2_CON	ESFR-b	D0	Регистр управления АЦП2	0000
F1A2	–	ESFR-b	D1	Зарезервировано	–
F1A4	–	ESFR-b	D2	Зарезервировано	–
F1A6	ADC2_CON1	ESFR-b	D3	Регистр управления АЦП2	0000
F1A8	–	ESFR-b	D4	Зарезервировано	–
F1AA	–	ESFR-b	D5	Зарезервировано	–
F1AC	–	ESFR-b	D6	Зарезервировано	–
F1AE	–	ESFR-b	D7	Зарезервировано	–
F1B0	–	ESFR-b	D8	Зарезервировано	–
F1B2	–	ESFR-b	D9	Зарезервировано	–
F1B4	–	ESFR-b	DA	Зарезервировано	–
F1B6	–	ESFR-b	DB	Зарезервировано	–
F1B8	S1CON	ESFR-b	DC	Регистр управления (ASC1)	0000
F1BA	–	ESFR-b	DD	Зарезервировано	–
F1BC	–	ESFR-b	DE	Зарезервировано	–
F1BE	–	ESFR-b	DF	Зарезервировано	–
F1C0	EXICON	ESFR-b	E0	Регистр контроля внешних прерываний	0000
F1C2	ODP2	ESFR-b	E1	Регистр управления режимом с открытым стоком порта P2	0000
F1C6	ODP3	ESFR-b	E3		0000
F1C8	–	ESFR-b	E4	Зарезервировано	–
F1CA	ODP4	ESFR-b	E5	Регистр управления режимом с открытым стоком порта P4	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
F1CC	GENCON	ESFR-b	E6	Регистр управления вспомогательным осциллятором PLL	0000
F1CE	ODP6	ESFR-b	E7	Регистр управления режимом с открытым стоком порта P6	0000
F1D0	PLLCON	ESFR-b	E8	Регистр управления PLL	XXXX
F1D2	ODP7	ESFR-b	E9	Регистр управления режимом с открытым стоком порта P7	0000
F1D4	–	ESFR-b	EA	Зарезервировано	
F1D6	ODP9	ESFR-b	EB	Регистр управления режимом с открытым стоком порта P9	0000
F1D8	EXISEL1	ESFR-b	EC	Регистр 1 выбора источника внешних прерываний	0000
F1DA	EXISEL0	ESFR-b	ED	Регистр 0 выбора источника внешних прерываний	0000
F1DC	–	ESFR-b	EE	Зарезервировано	–
F1DE	–	ESFR-b	EF	Зарезервировано	–
F1E0	–	ESFR	F0	Зарезервировано	–
F1E2	–	ESFR	F1	Зарезервировано	–
F1E4	–	ESFR	F2	Зарезервировано	–
F1E6	–	ESFR	F3	Зарезервировано	–
F1E8	–	ESFR	F4	Зарезервировано	–
F1EA	ALTSEL0P1L	ESFR	F5	Регистр управления альтернативными функциями порта P1L (порт P1)	0000
F1EC	ALTSEL0P1H	ESFR	F6	Регистр управления альтернативными функциями порта P1H (порт P1)	0000
F1EE	ALTSEL0P2	ESFR	F7	Регистр управления альтернативными функциями порта P2	0000
F1F0	ALTSEL0P3	ESFR	F8	Регистр 0 управления альтернативными функциями порта P3	0000
F1F2	ALTSEL1P3	ESFR	F9	Регистр 1 управления альтернативными функциями порта P3	0000
F1F4	ALTSEL0P4	ESFR	FA	Регистр управления альтернативными функциями порта P4	0000
F1F6	ALTSEL0P6	ESFR	FB	Регистр управления альтернативными функциями порта P6	0000
F1F8	ALTSEL0P7	ESFR	FC	Регистр 0 управления альтернативными функциями порта P7	0000
F1FA	ALTSEL1P7	ESFR	FD	Регистр 1 управления альтернативными функциями порта P7	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
F1FC	ALTSEL0P9	ESFR	FE	Регистр 0 управления альтернативными функциями порта P9	0000
F1FE	ALTSEL1P9	ESFR	FF	Регистр 1 управления альтернативными функциями порта P9	0000
FE00	DPP0	SFR	00	Регистр указателя нулевой страницы данных (10 битов) ЦПУ	0000
FE02	DPP1	SFR	01	Регистр указателя первой страницы данных (10 битов) ЦПУ	0001
FE04	DPP2	SFR	02	Регистр указателя страницы второй данных (10 битов) ЦПУ	0002
FE06	DPP3	SFR	03	Регистр указателя страницы третьей данных (10 битов) ЦПУ	0003
FE08	CSP	SFR	04	Регистр указателя сегмента кода (8 битов) ЦПУ	0000
FE0A	–	SFR	05	Зарезервировано	–
FE0C	MDH	SFR	06	Старший регистр умножения/деления ЦПУ	0000
FE0E	MDL	SFR	07	Младший регистр умножения/деления ЦПУ	0000
FE10	CP	SFR	08	Регистр контекстного указателя ЦПУ	FC00
FE12	SP	SFR	09	Регистр указателя стека ЦПУ	FC00
FE14	STKOV	SFR	0A	Регистр указателя переполнения стека ЦПУ	FA00
FE16	STKUN	SFR	0B	Регистр опустошения стека ЦПУ	FC00
FE18	ADDRSEL1	SFR	0C	Регистр выбора адреса 1	0000
FE1A	ADDRSEL2	SFR	0D	Регистр выбора адреса 2	0000
FE1C	ADDRSEL3	SFR	0E	Регистр выбора адреса 3	0000
FE1E	ADDRSEL4	SFR	0F	Регистр выбора адреса 4	0000
FE20	–	SFR	10	Зарезервировано	–
FE22	–	SFR	11	Зарезервировано	–
FE24	–	SFR	12	Зарезервировано	–
FE26	–	SFR	13	Зарезервировано	–
FE28	CC2_SEM	SFR	14	Регистр режима однократного срабатывания (CAPCOM2)	0000
FE2A	CC2_SEE	SFR	15	Регистр однократного срабатывания (CAPCOM2)	0000
FE2C	CC1_SEM	SFR	16	Регистр режима однократного срабатывания (CAPCOM1)	0000
FE2E	CC1_SEE	SFR	17	Регистр однократного срабатывания (CAPCOM1)	0000
FE30	–	SFR	18	Зарезервировано	–
FE32	–	SFR	19	Зарезервировано	–
FE34	–	SFR	1A	Зарезервировано	–
FE36	–	SFR	1B	Зарезервировано	–
FE38	–	SFR	1C	Зарезервировано	–

Продолжение таблицы Б.1

1	2	3	4	5	6
FE3A	–	SFR	1D	Зарезервировано	–
FE3C	–	SFR	1E	Зарезервировано	–
FE3E	–	SFR	1F	Зарезервировано	–
FE40	T2	SFR	20	Регистр таймера 2 (GPT1, GPT2)	0000
FE42	T3	SFR	21	Регистр таймера 3 (GPT1, GPT2)	0000
FE44	T4	SFR	22	Регистр таймера 4 (GPT1, GPT2)	0000
FE46	T5	SFR	23	Регистр таймера 5 (GPT1, GPT2)	0000
FE48	T6	SFR	24	Регистр таймера 6 (GPT1, GPT2)	0000
FE4A	CAPREL	SFR	25	Регистр захвата/перезагрузки (GPT1, GPT2)	0000
FE4C	–	SFR	26	Зарезервировано	–
FE4E	–	SFR	27	Зарезервировано	–
FE50	T0	SFR	28	Регистр таймера T0 (CAPCOM1)	0000
FE52	T1	SFR	29	Регистр таймера T1 (CAPCOM1)	0000
FE54	T0REL	SFR	2A	Регистр загрузки таймера 0 (CAPCOM1)	0000
FE56	T1REL	SFR	2B	Регистр загрузки таймера 1 (CAPCOM1)	0000
FE58	–	SFR	2C	Зарезервировано	–
FE5A	–	SFR	2D	Зарезервировано	–
FE5C	MAL	SFR	2E	Регистр младшего байта аккумулятора (MAC)	0000
FE5E	MAH	SFR	2F	Регистр старшего байта аккумулятора (MAC)	0000
FE60	CC16	SFR	30	Регистр захвата/сравнения 16 (CAPCOM2)	0000
FE62	CC17	SFR	31	Регистр захвата/сравнения 17 (CAPCOM2)	0000
FE64	CC18	SFR	32	Регистр захвата/сравнения 18 (CAPCOM2)	0000
FE66	CC19	SFR	33	Регистр захвата/сравнения 19 (CAPCOM2)	0000
FE68	CC20	SFR	34	Регистр захвата/сравнения 20 (CAPCOM2)	0000
FE6A	CC21	SFR	35	Регистр захвата/сравнения 21 (CAPCOM2)	0000
FE6C	CC22	SFR	36	Регистр захвата/сравнения 22 (CAPCOM2)	0000
FE6E	CC23	SFR	37	Регистр захвата/сравнения 23 (CAPCOM2)	0000
FE70	CC24	SFR	38	Регистр захвата/сравнения 24 (CAPCOM2)	0000
FE72	CC25	SFR	39	Регистр захвата/сравнения 25 (CAPCOM2)	0000
FE74	CC26	SFR	3A	Регистр захвата/сравнения 26 (CAPCOM2)	0000
FE76	CC27	SFR	3B	Регистр захвата/сравнения 27 (CAPCOM2)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FE78	CC28	SFR	3C	Регистр захвата/сравнения 28 (CAPCOM2)	0000
FE7A	CC29	SFR	3D	Регистр захвата/сравнения 29 (CAPCOM2)	0000
FE7C	CC30	SFR	3E	Регистр захвата/сравнения 30 (CAPCOM2)	0000
FE7E	CC31	SFR	3F	Регистр захвата/сравнения 31 (CAPCOM2)	0000
FE80	CC0	SFR	40	Регистр захвата/сравнения 0 (CAPCOM1)	0000
FE82	CC1	SFR	41	Регистр захвата/сравнения 1 (CAPCOM1)	0000
FE84	CC2	SFR	42	Регистр захвата/сравнения 2 (CAPCOM1)	0000
FE86	CC3	SFR	43	Регистр захвата/сравнения 3 (CAPCOM1)	0000
FE88	CC4	SFR	44	Регистр захвата/сравнения 4 (CAPCOM1)	0000
FE8A	CC5	SFR	45	Регистр захвата/сравнения 5 (CAPCOM1)	0000
FE8C	CC6	SFR	46	Регистр захвата/сравнения 6 (CAPCOM1)	0000
FE8E	CC7	SFR	47	Регистр захвата/сравнения 7 (CAPCOM1)	0000
FE90	CC8	SFR	48	Регистр захвата/сравнения 8 (CAPCOM1)	0000
FE92	CC9	SFR	49	Регистр захвата/сравнения 9 (CAPCOM1)	0000
FE94	CC10	SFR	4A	Регистр захвата/сравнения 10 (CAPCOM1)	0000
FE96	CC11	SFR	4B	Регистр захвата/сравнения 11 (CAPCOM1)	0000
FE98	CC12	SFR	4C	Регистр захвата/сравнения 12 (CAPCOM1)	0000
FE9A	CC13	SFR	4D	Регистр захвата/сравнения 13 (CAPCOM1)	0000
FE9C	CC14	SFR	4E	Регистр захвата/сравнения 14 (CAPCOM1)	0000
FE9E	CC15	SFR	4F	Регистр захвата/сравнения 15 (CAPCOM1)	0000
FEA0	ADC_DAT	SFR	50	Регистр результата АЦП	0000
FEA2	ADC2_DAT	SFR	51	Регистр результата АЦП2	0000
FEA4	ADC_SWITCH	SFR	52	Регистр выбора режима работы АЦП	0000
FEA6	S1TBUF	SFR	53	Буферный регистр передатчика (ASC1)	0000
FEA8	S1RBUF	SFR	54	Буферный регистр приемника (ASC1)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FEAA	S1BG	SFR	55	Регистр скорости пересылки (ASC1)	0000
FEAC	S1FDV	SFR	56	Регистр дробного делителя (ASC1)	0000
FEAE	WDT	SFR	57	Регистр сторожевого таймера (WDT)	0000
FEB0	S0TBUF	SFR	58	Буферный регистр передатчика (ASC0)	0000
FEB2	S0RBUF	SFR	59	Буферный регистр приемника (ASC0)	0000
FEB4	S0BG	SFR	5A	Регистр скорости пересылки (ASC0)	0000
FEB6	S0FDV	SFR	5B	Регистр дробного делителя (ASC0)	0000
FEB8	PECSN12	SFR	5C	Регистр указателя сегмента 12 (PEC)	0000
FEBA	PECSN13	SFR	5D	Регистр указателя сегмента 13 (PEC)	0000
FEBC	PECSN14	SFR	5E	Регистр указателя сегмента 14 (PEC)	0000
FEBE	PECSN15	SFR	5F	Регистр указателя сегмента 15 (PEC)	0000
FEC0	PECC0	SFR	60	Регистр управления PEC-каналом 0	0000
FEC2	PECC1	SFR	61	Регистр управления PEC-каналом 1	0000
FEC4	PECC2	SFR	62	Регистр управления PEC-каналом 2	0000
FEC6	PECC3	SFR	63	Регистр управления PEC-каналом 3	0000
FEC8	PECC4	SFR	64	Регистр управления PEC-каналом 4	0000
FECA	PECC5	SFR	65	Регистр управления PEC-каналом 5	0000
FECC	PECC6	SFR	66	Регистр управления PEC-каналом 6	0000
FECE	PECC7	SFR	67	Регистр управления PEC-каналом 7	0000
FED0	PECSN0	SFR	68	Регистр указателя сегмента 0 (PEC)	0000
FED2	PECSN1	SFR	69	Регистр указателя сегмента 1 (PEC)	0000
FED4	PECSN2	SFR	6A	Регистр указателя сегмента 2 (PEC)	0000
FED6	PECSN3	SFR	6B	Регистр указателя сегмента 3 (PEC)	0000
FED8	PECSN4	SFR	6C	Регистр указателя сегмента 4 (PEC)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FEDA	PECSN5	SFR	6D	Регистр указателя сегмента 5 (PEC)	0000
FEDC	PECSN6	SFR	6E	Регистр указателя сегмента 6 (PEC)	0000
FEDE	PECSN7	SFR	6F	Регистр указателя сегмента 7 (PEC)	0000
FEE0	PECSN8	SFR	70	Регистр указателя сегмента 8 (PEC)	0000
FEE2	PECSN9	SFR	71	Регистр указателя сегмента 9 (PEC)	0000
FEE4	PECSN10	SFR	72	Регистр указателя сегмента 10 (PEC)	0000
FEE6	PECSN11	SFR	73	Регистр указателя сегмента 11 (PEC)	0000
FEE8	PECC8	SFR	74	Регистр управления PEC-каналом 8	0000
FEEA	PECC9	SFR	75	Регистр управления PEC-каналом 9	0000
FEEC	PECC10	SFR	76	Регистр управления PEC-каналом 10	0000
FEEE	PECC11	SFR	77	Регистр управления PEC-каналом 11	0000
FEF0	PECXC0	SFR	78	Регистр расширенного управления каналом 0 (PEC)	0000
FEF2	PECXC2	SFR	79	Регистр расширенного управления каналом 2 (PEC)	0000
FEF4	–	SFR	7A	Зарезервировано	–
FEF6	–	SFR	7B	Зарезервировано	–
FEF8	PECC12	SFR	7C	Регистр управления PEC-каналом 12	0000
FEFA	PECC13	SFR	7D	Регистр управления PEC-каналом 13	0000
FEFC	PECC14	SFR	7E	Регистр управления PEC-каналом 14	0000
FEFE	PECC15	SFR	7F	Регистр управления PEC-каналом 15	0000
FF00	P0L	SFR-b	80	Младший регистр порта P0	00
FF02	P0H	SFR-b	81	Старший регистр порта P0	00
FF04	P1L	SFR-b	82	Младший регистр порта P1	00
FF06	P1H	SFR-b	83	Старший регистр порта P1	00
FF08	IDX0	SFR-b	84	Регистр 0 указателя адреса MAC	0000
FF0A	IDX1	SFR-b	85	Регистр 1 указателя адреса MAC	0000
FF0C	BUSCON0	SFR-b	86	Регистр конфигурации шины 0	0000
FF0E	MDC	SFR-b	87	Регистр управления умножением/делением ЦПУ	0000
FF10	PSW	SFR-b	88	Регистр слова состояния процессора ЦПУ	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FF12	SYSCON	SFR-b	89	Регистр конфигурации системы (ЦПУ)	XXXX
FF14	BUSCON1	SFR-b	8A	Регистр конфигурации шины 1	0000
FF16	BUSCON2	SFR-b	8B	Регистр конфигурации шины 2	0000
FF18	BUSCON3	SFR-b	8C	Регистр конфигурации шины 3	0000
FF1A	BUSCON4	SFR-b	8D	Регистр конфигурации шины 4	0000
FF1C	ZEROS	SFR-b	8E	Регистр постоянного значения 0	0000
FF1E	ONES	SFR-b	8F	Регистр постоянного значения 1	FFFF
FF20	CC2_T78CON	SFR-b	90	Регистр управления таймерами T7 и T8 (CAPCOM2)	0000
FF22	CCM4	SFR-b	91	Регистр управления режимом захвата/сравнения 4 (CAPCOM1)	0000
FF24	CCM5	SFR-b	92	Регистр управления режимом захвата/сравнения 5 (CAPCOM1)	0000
FF26	CCM6	SFR-b	93	Регистр управления режимом захвата/сравнения 6 (CAPCOM1)	0000
FF28	CCM7	SFR-b	94	Регистр управления режимом захвата/сравнения 7 (CAPCOM1)	0000
FF2A	CC2_DRM	SFR-b	95	Регистр двухрегистрового режима сравнения (CAPCOM2)	0000
FF2C	CC2_OUT	SFR-b	96	Регистр выводов сравнения (CAPCOM2)	0000
FF2E	-	SFR-b	97	Зарезервировано	
FF30	-	SFR-b	98	Зарезервировано	
FF32	-	SFR-b	99	Зарезервировано	
FF34	-	SFR-b	9A	Зарезервировано	
FF36	-	SFR-b	9B	Зарезервировано	
FF38	-	SFR-b	9C	Зарезервировано	
FF3A	-	SFR-b	9D	Зарезервировано	
FF3C	-	SFR-b	9E	Зарезервировано	
FF3E	-	SFR-b	9F	Зарезервировано	
FF40	T2CON	SFR-b	A0	Регистр управления таймера 2 (GPT1, GPT2)	0000
FF42	T3CON	SFR-b	A1	Регистр управления таймера 3 (GPT1, GPT2)	0000
FF44	T4CON	SFR-b	A2	Регистр управления таймера 4 (GPT1, GPT2)	0000
FF46	T5CON	SFR-b	A3	Регистр управления таймера 5 (GPT1, GPT2)	0000
FF48	T6CON	SFR-b	A4	Регистр управления таймера 6 (GPT1, GPT2)	0000
FF4A	-	SFR-b	A5	Зарезервировано	
FF4C	-	SFR-b	A6	Зарезервировано	
FF4E	-	SFR-b	A7	Зарезервировано	
FF50	CC1_T01CON	SFR-b	A8	Регистр управления таймерами T0 и T1 (CAPCOM1)	0000
FF52	CCM0	SFR-b	A9	Регистр управления режимом захвата/сравнения 0 (CAPCOM1)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FF54	CCM1	SFR-b	AA	Регистр управления режимом захвата/сравнения 1 (CAPCOM1)	0000
FF56	CCM2	SFR-b	AB	Регистр управления режимом захвата/сравнения 2 (CAPCOM1)	0000
FF58	CCM3	SFR-b	AC	Регистр управления режимом захвата/сравнения 3 (CAPCOM1)	0000
FF5A	CC1_DRM	SFR-b	AD	Регистр двухрегистрового режима сравнения (CAPCOM1)	0000
FF5C	CC1_OUT	SFR-b	AE	Регистр выводов сравнения (CAPCOM1)	0000
FF5E	SSC1CON	SFR-b	AF	Регистр управления (SSC1)	0000
FF60	T2IC	SFR-b	B0	Регистр управления прерываниями таймера 2 (GPT1, GPT2)	0000
FF62	T3IC	SFR-b	B1	Регистр управления прерываниями таймера 3 (GPT1, GPT2)	0000
FF64	T4IC	SFR-b	B2	Регистр управления прерываниями таймера 4 (GPT1, GPT2)	0000
FF66	T5IC	SFR-b	B3	Регистр управления прерываниями таймерами 5 (GPT1, GPT2)	0000
FF68	T6IC	SFR-b	B4	Регистр управления прерываниями таймера 6 (GPT1, GPT2)	0000
FF6A	CRIC	SFR-b	B5	Регистр управления прерыванием захвата/перезагрузки (GPT1, GPT2)	0000
FF6C	S0TIC	SFR-b	B6	Регистр контроля прерывания по передаче (ASC0)	0000
FF6E	S0RIC	SFR-b	B7	Регистр контроля прерывания по приему (ASC0)	0000
FF70	S0EIC	SFR-b	B8	Регистр контроля прерывания по ошибке (ASC0)	0000
FF72	SSC0TIC	SFR-b	B9	Регистр контроля прерывания по передаче (SSC0)	0000
FF74	SSC0RIC	SFR-b	BA	Регистр контроля прерывания по приему (SSC0)	0000
FF76	SSC0EIC	SFR-b	BB	Регистр контроля прерывания по ошибке (SSC0)	0000
FF78	CC0IC	SFR-b	BC	Регистр управления прерываниями 0 (CAPCOM1)	0000
FF7A	CC1IC	SFR-b	BD	Регистр управления прерываниями 1 (CAPCOM1)	0000
FF7C	CC2IC	SFR-b	BE	Регистр управления прерываниями 2 (CAPCOM1)	0000

Продолжение таблицы Б.1

1	2	3	4	5	6
FF7E	CC3IC	SFR-b	BF	Регистр управления прерываниями 3 (CAPCOM1)	0000
FF80	CC4IC	SFR-b	C0	Регистр управления прерываниями 4 (CAPCOM1)	0000
FF82	CC5IC	SFR-b	C1	Регистр управления прерываниями 5 (CAPCOM1)	0000
FF84	CC6IC	SFR-b	C2	Регистр управления прерываниями 6 (CAPCOM1)	0000
FF86	CC7IC	SFR-b	C3	Регистр управления прерываниями 7 (CAPCOM1)	0000
FF88	CC8IC	SFR-b	C4	Регистр управления прерываниями 8 (CAPCOM1)	0000
FF8A	CC9IC	SFR-b	C5	Регистр управления прерываниями 9 (CAPCOM1)	0000
FF8C	CC10IC	SFR-b	C6	Регистр управления прерываниями 10 (CAPCOM1)	0000
FF8E	CC11IC	SFR-b	C7	Регистр управления прерываниями 11 (CAPCOM1)	0000
FF90	CC12IC	SFR-b	C8	Регистр управления прерываниями 12 (CAPCOM1)	0000
FF92	CC13IC	SFR-b	C9	Регистр управления прерываниями 13 (CAPCOM1)	0000
FF94	CC14IC	SFR-b	CA	Регистр управления прерываниями 14 (CAPCOM1)	0000
FF96	CC15IC	SFR-b	CB	Регистр управления прерываниями 15 (CAPCOM1)	0000
FF98	ADC_CIC	SFR-b	CC	Регистр контроля прерывания по завершению преобразования (АЦП1)	0000
FF9A	ADC_EIC	SFR-b	CD	Регистр контроля прерывания по ошибке (АЦП1)	0000
FF9C	T0IC	SFR-b	CE	Регистр управления прерываниями таймера 0 (CAPCOM1)	--00
FF9E	T1IC	SFR-b	CF	Регистр управления прерываниями таймера 1 (CAPCOM1)	--00
FFA0	ADC_CON	SFR-b	D0	Регистр управления АЦП	0000
FFA2	P5	SFR-b	D1	Регистр данных порта P5	0000
FFA4	–	SFR-b	D2	Зарезервировано	–
FFA6	ADC_CON1	SFR-b	D3	Регистр управления АЦП	0000
FFA8	PECISNC	SFR-b	D4	Регистр управления узлом прерываний PEC	0000
FFAA	FOCON	SFR-b	D5	Регистр управления выходным тактовым сигналом	0000
FFAC	TFR	SFR-b	D6	Регистр флагов ловушек	0000
FFAE	WDTCON	SFR-b	D7	Регистр управления сторожевым таймером WDT	008X

Окончание таблицы Б.1

1	2	3	4	5	6
FFB0	S0CON	SFR-b	D8	Регистр управления ASC0	0000
FFB2	SSC0CON	SFR-b	D9	Регистр управления SSC0	0000
FFB4	–	SFR-b	DA	Зарезервировано	–
FFB6	–	SFR-b	DB	Зарезервировано	–
FFB8	–	SFR-b	DC	Зарезервировано	–
FFBA	–	SFR-b	DD	Зарезервировано	–
FFBC	–	SFR-b	DE	Зарезервировано	–
FFBE	–	SFR-b	DF	Зарезервировано	–
FFC0	P2	SFR-b	E0	Регистр данных порта P2	0000
FFC2	DP2	SFR-b	E1	Регистр выбора направления порта P2	0000
FFC4	P3	SFR-b	E2	Регистр данных порта P3	0000
FFC6	DP3	SFR-b	E3	Регистр выбора направления порта P3	0000
FFC8	P4	SFR-b	E4	Регистр данных порта P4	0000
FFCA	DP4	SFR-b	E5	Регистр выбора направления порта P4	0000
FFCC	P6	SFR-b	E6	Регистр данных порта P6	0000
FFCE	DP6	SFR-b	E7	Регистр выбора направления порта P6	0000
FFD0	P7	SFR-b	E8	Регистр данных порта P7	0000
FFD2	DP7	SFR-b	E9	Регистр выбора направления порта P7	0000
FFD4	P9	SFR-b	EA	Регистр данных порта P9	0000
FFD6	DP9	SFR-b	EB	Регистр выбора направления порта P9	0000
FFD8	–	SFR-b	EC	Зарезервировано	–
FFDA	MRW	SFR-b	ED	Регистр повторения блока MAC	0000
FFDC	MCW	SFR-b	EE	Регистр управления блока MAC	0000
FFDE	MSW	SFR-b	EF	Статусный регистр блока MAC	0200
FFE0	–	SFR-b	F0	Зарезервировано	–
FFE2	ASC0ID	SFR-b	F1	Регистр идентификации ASC01	44XX
FFE4	SSC0ID	SFR-b	F2	Регистр идентификации SSC0	45XX
FFE6	GPTID	SFR-b	F3	Регистр идентификации GPT1 и GPT2	58XX
FFE8	–	SFR-b	F4	Зарезервировано	–
FFEA	–	SFR-b	F5	Зарезервировано	–
FFEC	–	SFR-b	F6	Зарезервировано	–
FFEE	–	SFR-b	F7	Зарезервировано	–
FFF0	–	SFR-b	F8	Зарезервировано	–
FFF2	–	SFR-b	F9	Зарезервировано	–
FFF4	–	SFR-b	FA	Зарезервировано	–
FFF6	–	SFR-b	FB	Зарезервировано	–
FFF8	–	SFR-b	FC	Зарезервировано	–
FFFA	–	SFR-b	FD	Зарезервировано	–
FFFC	–	SFR-b	FE	Зарезервировано	–
FFFE	–	SFR-b	FF	Зарезервировано	–

Приложение В
(обязательное)

Таблица регистров областей SFR и ESFR, упорядоченных по именам регистров

Таблица В.1 – Таблица регистров областей (E)SFR, упорядоченных по именам

Название	Физи-ческий адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET)
1	2	3	4	5	6
ADC_CIC	FF98	SFR-b	CC	Регистр контроля прерывания по завершению преобразования АЦП1	0000
ADC_CON	FFA0	SFR-b	D0	Регистр управления АЦП	0000
ADC_CON1	FFA6	SFR-b	D3	Регистр управления АЦП	0000
ADC_DAT	FEA0	SFR	50	Регистр результата АЦП	0000
ADC_DAT2	F0A0	ESFR	50	Регистр результата АЦП инжектированного канала	0000
ADC_EIC	FF9A	SFR-b	CD	Регистр контроля прерывания по ошибке АЦП1	0000
ADC_SWITCH	FEA4	SFR	52	Регистр выбора режима модуля АЦП	0000
ADC2_CIC	F18E	ESFR-b	C7	Регистр контроля прерывания по завершению преобразования АЦП2	0000
ADC2_CON	F1A0	ESFR-b	D0	Регистр управления АЦП2	0000
ADC2_CON1	F1A6	ESFR-b	D3	Регистр управления АЦП2	0000
ADC2_DAT	FEA2	SFR	51	Регистр результата АЦП2	0000
ADC2_DAT2	F0A2	ESFR	51	Регистр результата АЦП2 преобразования инжектированного канала	0000
ADC2_EIC	F196	ESFR-b	CB	Регистр контроля прерывания по ошибке АЦП2	0000
ADDRSEL1	FE18	SFR	0C	Регистр выбора адреса 1	0000
ADDRSEL2	FE1A	SFR	0D	Регистр выбора адреса 2	0000
ADDRSEL3	FE1C	SFR	0E	регистр выбора адреса 3	0000
ADDRSEL4	FE1E	SFR	0F	Регистр выбора адреса 4	0000
ALTSEL0P1H	F1EC	ESFR	F6	Регистр управления альтернативными функциями порта P1H	0000
ALTSEL0P1L	F1EA	ESFR	F5	Регистр управления альтернативными функциями порта P1L	0000
ALTSEL0P2	F1EE	ESFR	F7	Регистр управления альтернатив-ными функциями порта P2	0000
ALTSEL0P3	F1F0	ESFR	F8	Регистр 0 управления альтернатив-ными функциями порта P3	0000
ALTSEL0P4	F1F4	ESFR	FA	Регистр управления альтернатив-ными функциями порта P4	0000

Продолжение таблицы В.1

1	2	3	4	5	6
ALTSEL0P6	F1F6	ESFR	FB	Регистр управления альтернативными функциями порта P6	0000
ALTSEL0P7	F1F8	ESFR	FC	Регистр 0 управления альтернативными функциями порта P7	0000
ALTSEL0P9	F1FC	ESFR	FE	Регистр 0 управления альтернативными функциями порта P9	0000
ALTSEL1P3	F1F2	ESFR	F9	Регистр 1 управления альтернативными функциями порта P3	0000
ALTSEL1P7	F1FA	ESFR	FD	Регистр 1 управления альтернативными функциями порта P7	0000
ALTSEL1P9	F1FE	ESFR	FF	Регистр 1 управления альтернативными функциями порта P9	0000
ASC0ID	FFE2	SFR-b	F1	Регистр идентификации ASC0	44XX
BUSCON0	FF0C	SFR-b	86	Регистр конфигурации шины 0	0000
BUSCON1	FF14	SFR-b	8A	Регистр конфигурации шины 1	0000
BUSCON2	FF16	SFR-b	8B	Регистр конфигурации шины 2	0000
BUSCON3	FF18	SFR-b	8C	Регистр конфигурации шины 3	0000
BUSCON4	FF1A	SFR-b	8D	Регистр конфигурации шины 4	0000
CAPREL	FE4A	SFR	25	Регистр захвата/перезагрузки (GPT1, GPT2)	0000
CC0IC	FF78	SFR-b	BC	Регистр управления прерываниями 0 (CAPCOM1)	0000
CC1_DRM	FF5A	SFR-b	AD	Регистр двухрегистрового режима сравнения (CAPCOM1)	0000
CC1_IOC	F062	ESFR	31	Регистр управления вводом/выводом (CAPCOM1)	0000
CC1_OUT	FF5C	SFR-b	AE	Регистр выводов сравнения (CAPCOM1)	0000
CC1_SEE	FE2E	SFR	17	Регистр однократного срабатывания (CAPCOM1)	0000
CC1_SEM	FE2C	SFR	16	Регистр режима однократного срабатывания (CAPCOM1)	0000
CC10	FE94	SFR	4A	Регистр захвата/сравнения 10 (CAPCOM1)	0000
CC10IC	FF8C	SFR-b	C6	Регистр управления прерываниями 10 (CAPCOM1)	0000
CC11	FE96	SFR	4B	Регистр захвата/сравнения 11 (CAPCOM1)	0000
CC11IC	FF8E	SFR-b	C7	Регистр управления прерываниями 11 (CAPCOM1)	0000
C12	FE98	SFR	4C	Регистр захвата/сравнения 12 (CAPCOM1)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
CC12IC	FF90	SFR-b	C8	Регистр управления прерываниями 12 (CAPCOM1)	0000
CC13	FE9A	SFR	4D	Регистр захвата/сравнения 13 (CAPCOM1)	0000
CC13IC	FF92	SFR-b	C9	Регистр управления прерываниями 13 (CAPCOM1)	0000
CC14	FE9C	SFR	4E	Регистр захвата/сравнения 14 (CAPCOM1)	0000
CC14IC	FF94	SFR-b	CA	Регистр управления прерываниями 14 (CAPCOM1)	0000
CC15	FE9E	SFR	4F	Регистр захвата/сравнения 15 (CAPCOM1)	0000
CC15IC	FF96	SFR-b	CB	Регистр управления прерываниями 15 (CAPCOM1)	0000
CC16	FE60	SFR	30	Регистр захвата/сравнения 16 (CAPCOM2)	0000
CC16IC	F160	ESFR-b	B0	Регистр управления прерываниями 16 (CAPCOM2)	0000
CC17	FE62	SFR	31	Регистр захвата/сравнения 17 (CAPCOM2)	0000
CC17IC	F162	ESFR-b	B1	Регистр управления прерываниями 17 (CAPCOM2)	0000
CC18	FE64	SFR	32	Регистр захвата/сравнения 18 (CAPCOM2)	0000
CC18IC	F164	ESFR-b	B2	Регистр управления прерываниями 18 (CAPCOM2)	0000
CC19	FE66	SFR	33	Регистр захвата/сравнения 19 (CAPCOM2)	0000
CC19IC	F166	ESFR-b	B3	Регистр управления прерываниями 19 (CAPCOM2)	0000
CC1IC	FF7A	SFR-b	BD	Регистр управления прерываниями 1 (CAPCOM1)	0000
CC2_DRM	FF2A	SFR-b	95	Регистр двухрегистрового режима сравнения (CAPCOM2)	0000
CC2_IOC	F066	ESFR	33	Регистр управления вводом/выводом (CAPCOM2)	0000
CC2_OUT	FF2C	SFR-b	96	Регистр выводов сравнения (CAPCOM2)	0000
CC2_SEE	FE2A	SFR	15	Регистр однократного срабатывания (CAPCOM2)	0000
CC2_SEM	FE28	SFR	14	Регистр режима однократного срабатывания (CAPCOM2)	0000
CC20	FE68	SFR	34	Регистр захвата/сравнения 20 (CAPCOM2)	0000
CC20IC	F168	ESFR-b	B4	Регистр управления прерываниями 20 (CAPCOM2)	0000
CC21	FE6A	SFR	35	Регистр захвата/сравнения 21 (CAPCOM2)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
CC21IC	F16A	ESFR-b	B5	Регистр управления прерываниями 21 (CAPCOM2)	0000
CC22	FE6C	SFR	36	Регистр захвата/сравнения 22 (CAPCOM2)	0000
CC22IC	F16C	ESFR-b	B6	Регистр управления прерываниями 22 (CAPCOM2)	0000
CC23	FE6E	SFR	37	Регистр захвата/сравнения 23 (CAPCOM2)	0000
CC23IC	F16E	ESFR-b	B7	Регистр управления прерываниями 23 (CAPCOM2)	0000
CC24	FE70	SFR	38	Регистр захвата/сравнения 24 (CAPCOM2)	0000
CC24IC	F170	ESFR-b	B8	Регистр управления прерываниями 24 (CAPCOM2)	0000
CC25	FE72	SFR	39	Регистр захвата/сравнения 25 (CAPCOM2)	0000
CC25IC	F172	ESFR-b	B9	Регистр управления прерываниями 25 (CAPCOM2)	0000
CC26	FE74	SFR	3A	Регистр захвата/сравнения 26 (CAPCOM2)	0000
CC26IC	F174	ESFR-b	BA	Регистр управления прерываниями 26 (CAPCOM2)	0000
CC27	FE76	SFR	3B	Регистр захвата/сравнения 27 (CAPCOM2)	0000
CC27IC	F176	ESFR-b	BB	Регистр управления прерываниями 27 (CAPCOM2)	0000
CC28	FE78	SFR	3C	Регистр захвата/сравнения 28 (CAPCOM2)	0000
CC28IC	F178	ESFR-b	BC	Регистр управления прерываниями 28 (CAPCOM2)	0000
CC29	FE7A	SFR	3D	Регистр захвата/сравнения 29 (CAPCOM2)	0000
CC29IC	F184	ESFR-b	C2	Регистр управления прерываниями 29 (CAPCOM2)	0000
CC2IC	FF7C	SFR-b	BE	Регистр управления прерываниями 2 (CAPCOM1)	0000
CC3	FE86	SFR	43	Регистр захвата/сравнения 3 (CAPCOM1)	
CC30	FE7C	SFR	3E	Регистр захвата/сравнения 30 (CAPCOM2)	0000
CC30IC	F18C	ESFR-b	C6	Регистр управления прерываниями 30 (CAPCOM2)	0000
CC31	FE7E	SFR	3F	Регистр захвата/сравнения 31 (CAPCOM2)	0000
CC31IC	F194	ESFR-b	CA	Регистр управления прерываниями 31 (CAPCOM2)	0000
CC3IC	FF7E	SFR-b	BF	Регистр управления прерываниями 3 (CAPCOM1)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
CC4	FE88	SFR	44	Регистр захвата/сравнения 4 (CAPCOM1)	0000
CC4IC	FF80	SFR-b	C0	Регистр управления прерываниями 4 (CAPCOM1)	0000
CC5	FE8A	SFR	45	Регистр захвата/сравнения 5 (CAPCOM1)	0000
CC5IC	FF82	SFR-b	C1	Регистр управления прерываниями 5 (CAPCOM1)	0000
CC6	FE8C	SFR	46	Регистр захвата/сравнения 6 (CAPCOM1)	0000
CC6IC	FF84	SFR-b	C2	Регистр управления прерываниями 6 (CAPCOM1)	0000
CC7	FE8E	SFR	47	Регистр захвата/сравнения 7 (CAPCOM1)	0000
CC7IC	FF86	SFR-b	C3	Регистр управления прерываниями 7 (CAPCOM1)	0000
CC8	FE90	SFR	48	Регистр захвата/сравнения 8 (CAPCOM1)	0000
CC8IC	FF88	SFR-b	C4	Регистр управления прерываниями 8 (CAPCOM1)	0000
CC9	FE92	SFR	49	Регистр захвата/сравнения 9 (CAPCOM1)	0000
CC9IC	FF8A	SFR-b	C5	Регистр управления прерываниями 9 (CAPCOM1)	0000
CC1_M0	FF52	SFR-b	A9	Регистр управления режимом захвата/сравнения 0 (CAPCOM1)	0000
CC1_M1	FF54	SFR-b	AA	Регистр управления режимом захвата/сравнения 1 (CAPCOM1)	0000
CC1_M2	FF56	SFR-b	AB	Регистр управления режимом захвата/сравнения 2 (CAPCOM1)	0000
CC1_M3	FF58	SFR-b	AC	Регистр управления режимом захвата/сравнения 3 (CAPCOM1)	0000
CC2_M4	FF22	SFR-b	91	Регистр управления режимом захвата/сравнения 4 (CAPCOM2)	0000
CC2_M5	FF24	SFR-b	92	Регистр управления режимом захвата/сравнения 5 (CAPCOM2)	0000
CC2_M6	FF26	SFR-b	93	Регистр управления режимом захвата/сравнения 6 (CAPCOM2)	0000
CC2_M7	FF28	SFR-b	94	Регистр управления режимом захвата/сравнения 7 (CAPCOM2)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
CCU6_EIC	F188	ESFR-b	C4	Регистр управления прерываниями по ошибке (CAPCOM6)	--00
CCU6_IC	F120	ESFR-b	90	Регистр управления прерываниями (CAPCOM6)	--00
CCU6_T12IC	F190	ESFR-b	C8	Регистр управления прерываниями таймера T12 (CAPCOM6)	--00
CCU6_T13IC	F198	ESFR-b	CC	Регистр управления прерываниями таймера T13 (CAPCOM6)	--00
CFG	F0EA	ESFR	75	Регистр конфигурации I2C	--00
COMDATA	F068	ESFR	34	Регистр режима коммуникации Cerberus (OCDS)	0000
CP	FE10	SFR	08	Регистр контекстного указателя (ЦПУ)	FC00
CPUID	F00C	ESFR	06	Регистр идентификации ЦПУ	0420
CRIC	FF6A	SFR-b	B5	Регистр управления прерыванием захвата/перезагрузки (GPT1, GPT2)	0000
DBGSR	F0FC	ESFR	7E	Регистр состояния отладки (OCDS)	0000
DCMPDP	F0EE	ESFR	77	Регистр данных программирования для DCMPx (OCDS)	0000
DCMPSP	F0EC	ESFR	76	Регистр выбора и программирования для DCMPx (OCDS)	0000
DEXEVT	F0F2	ESFR	79	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)	0000
DIP	F0F8	ESFR	7C	Регистр указателя машинной команды	0000
DIPX	F0FA	ESFR	7D	Регистр расширенного указателя инструкций	3000
DP0H	F102	ESFR-b	81	Регистр выбора направления порта P0H	0000
DP0L	F100	ESFR-b	80	Регистр выбора направления порта P0L	0000
DP1H	F106	ESFR-b	83	Регистр выбора направления порта P1H	0000
DP1L	F104	ESFR-b	82	Регистр выбора направления порта P1L	0000
DP2	FFC2	SFR-b	E1	Регистр выбора направления порта P2	0000

Продолжение таблицы В.1

1	2	3	4	5	6
DP3	FFC6	SFR-b	E3	Регистр выбора направления порта P3	0000
DP4	FFCA	SFR-b	E5	Регистр выбора направления порта P4	0000
DP6	FFCE	SFR-b	E7	Регистр выбора направления порта P6	0000
DP7	FFD2	SFR-b	E9	Регистр выбора направления порта P7	0000
DP9	FFD6	SFR-b	EB	Регистр выбора направления порта P9	0000
DPP0	FE00	SFR	00	Регистр указателя нулевой страницы данных (10 битов) ЦПУ	0000
DPP1	FE02	SFR	01	Регистр указателя первой страницы данных (10 битов) ЦПУ	0001
DPP2	FE04	SFR	02	Регистр указателя второй страницы данных (10 битов) ЦПУ	0002
DPP3	FE06	SFR	03	Регистр указателя третьей страницы данных (10 битов) ЦПУ	0003
DSWEVT	F0F4	ESFR	7A	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)	0000
DTIDR	F0D8	ESFR	6C	Регистр идентификации задачи системы отладки OCDS	0000
DTREVT	F0F0	ESFR	78	Регистр управления комбинациями аппаратных отладочных событий OCDS	0000
EOPIC	F180	ESFR-b	C0	Регистр управления прерываниями	0000
EXICON	F1C0	ESFR-b	E0	Регистр контроля внешних прерываний	0000
EXISEL0	F1DA	ESFR-b	ED	Регистр 0 выбора источника внешних прерываний	0000
EXISEL1	F1D8	ESFR-b	EC	Регистр 1 выбора источника внешних прерываний	0000
F0CON	FFAA	SFR-b	D5	Регистр управления выходным тактовым сигналом	0000
GENCON	F1CC	ESFR-b	E6	Регистр управления вспомогательным осциллятором	0000
GPTID	FFE6	SFR-b	F3	Регистр идентификации (GPT1, GPT2)	58XX
I2C_DIC	F186	ESFR-b	C3	Регистр управления прерыванием (I2C)	--00
IDX0	FF08	SFR-b	84	Регистр 0 указателя адреса (MAC)	0000
IDX1	FF0A	SFR-b	85	Регистр 1 указателя адреса (MAC)	0000
IOSR	F06C	ESFR	36	Регистр статуса Cerberus (OCDS)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
MAH	FE5E	SFR	2F	Регистр старшего байта аккумулятора (MAC)	0000
MAL	FE5C	SFR	2E	Регистр младшего байта аккумулятора (MAC)	0000
MCW	FFDC	SFR-b	EE	Статусный регистр блока MAC	0000
MDC	FF0E	SFR-b	87	Регистр управления умножением/делением ЦПУ	0000
MDH	FE0C	SFR	06	Старший регистр умножения/деления ЦПУ	0000
MDL	FE0E	SFR	07	Младший регистр умножения/деления ЦПУ	0000
MRW	FFDA	SFR-b	ED	Регистр повторения блока MAC	0000
MSW	FFDE	SFR-b	EF	Статусный регистр блока MAC	0200
ODP2	F1C2	ESFR-b	E1	Регистр управления режимом с открытым стоком порта P2	0000
ODP3	F1C6	ESFR-b	E3	Регистр управления режимом с открытым стоком порта P3	0000
ODP4	F1CA	ESFR-b	E5	Регистр управления режимом с открытым стоком порта P4	0000
ODP6	F1CE	ESFR-b	E7	Порт 6 регистр управления режимом с открытым стоком	0000
ODP7	F1D2	ESFR-b	E9	Регистр управления режимом с открытым стоком порта P6	0000
ODP9	F1D6	ESFR-b	EB	Регистр управления режимом с открытым стоком порта P9	0000
ONES	FF1E	SFR-b	8F	Регистр постоянного значения 1	FFFF
P0H	FF02	SFR-b	81	Старший регистр порта P0	--00
P0L	FF00	SFR-b	80	Младший регистр порта P0	--00
P1H	FF06	SFR-b	83	Старший регистр порта P1	--00
P1L	FF04	SFR-b	82	Младший регистр порта P1	--00
P2	FFC0	SFR-b	E0	Регистр данных порта P2	0000
P3	FFC4	SFR-b	E2	Регистр данных порта P3	0000
P4	FFC8	SFR-b	E4	Регистр данных порта P4	0000
P5	FFA2	SFR-b	D1	Регистр данных порта P5	0000
P6	FFCC	SFR-b	E6	Регистр данных порта P6	0000
P7	FFD0	SFR-b	E8	Регистр данных порта P7	0000
P9	FFD4	SFR-b	EA	Регистр данных порта P9	0000
PECC0	FEC0	SFR	60	Регистр управления PEC-каналом 0	0000
PECC1	FEC2	SFR	61	Регистр управления PEC-каналом 1	0000
PECC10	FEEC	SFR	76	Регистр управления PEC-каналом 10	0000
PECC11	FEEE	SFR	77	Регистр управления PEC-каналом 11	0000
PECC12	FEF8	SFR	7C	Регистр управления PEC-каналом 12	0000
PECC13	FEFA	SFR	7D	Регистр управления PEC-каналом 13	0000

Продолжение таблицы В.1

1	2	3	4	5	6
PECC14	FEFC	SFR	7E	Регистр управления PEC-каналом 14	0000
PECC15	FEFE	SFR	7F	Регистр управления PEC-каналом 15	0000
PECC2	FEC4	SFR	62	Регистр управления PEC-каналом 2	0000
PECC3	FEC6	SFR	63	Регистр управления PEC-каналом 3	0000
PECC4	FEC8	SFR	64	Регистр управления PEC-каналом 4	0000
PECC5	FECA	SFR	65	Регистр управления PEC-каналом 5	0000
PECC6	FECC	SFR	66	Регистр управления PEC-каналом 6	0000
PECC7	FECE	SFR	67	Регистр управления PEC-каналом 7	0000
PECC8	FEE8	SFR	74	Регистр управления PEC-каналом 8	0000
PECC9	FEEA	SFR	75	Регистр управления PEC-каналом 9	0000
PECISNC	FFA8	SFR-b	D4	Регистр управления узлом прерываний (PEC)	0000
PECSN0	FED0	SFR	68	Регистр указателя сегмента 0 (PEC)	0000
PECSN1	FED2	SFR	69	Регистр указателя сегмента 1 (PEC)	0000
PECSN10	FEE4	SFR	72	Регистр указателя сегмента 10 (PEC)	0000
PECSN11	FEE6	SFR	73	Регистр указателя сегмента 11 (PEC)	0000
PECSN12	FEB8	SFR	5C	Регистр указателя сегмента 12 (PEC)	0000
PECSN13	FEBA	SFR	5D	Регистр указателя сегмента 13 (PEC)	0000
PECSN14	FEBC	SFR	5E	Регистр указателя сегмента 14 (PEC)	0000
PECSN15	FEBE	SFR	5F	Регистр указателя сегмента 15 (PEC)	0000
PECSN2	FED4	SFR	6A	Регистр указателя сегмента 2 (PEC)	0000
PECSN3	FED6	SFR	6B	Регистр указателя сегмента 3 (PEC)	0000
PECSN4	FED8	SFR	6C	Регистр указателя сегмента 4 (PEC)	0000
PECSN5	FEDA	SFR	6D	Регистр указателя сегмента 5 (PEC)	0000
PECSN6	FEDC	SFR	6E	Регистр указателя сегмента 6 (PEC)	0000

Продолжение таблицы В.1

1	2	3	4	5	6
PECSN7	FEDE	SFR	6F	Регистр указателя сегмента 7 (PEC)	0000
PECSN8	FEE0	SFR	70	Регистр указателя сегмента 8 (PEC)	0000
PECSN9	FEE2	SFR	71	Регистр указателя сегмента 9 (PEC)	0000
PECXC0	FEF0	SFR	78	Регистр расширенного управления каналом 0 (PEC)	0000
PECXC2	FEF2	SFR	79	Регистр расширенного управления каналом 2 (PEC)	0000
PLLCON	F1D0	ESFR-b	E8	Регистр управления PLL	XXXX
PLLLOCK_IC	F19E	ESFR-b	CF	Регистр управления прерыванием по нахождению частоты PLL	0000
PLLUNLOCK_IC	F156	ESFR-b	AB	Регистр управления прерыванием по потере частоты PLL	0000
PSW	FF10	SFR-b	88	Регистр слова состояния процессора ЦПУ	0000
QR0	F004	ESFR	02	Регистр смещения для указателей адреса GPR	0000
QR1	F006	ESFR	03	Регистр смещения для указателей адреса GPR	0000
QX0	F000	ESFR	00	Регистр смещения 0 для указателя адреса IDX0/1	0000
QX1	F002	ESFR	01	Регистр смещения 1 для указателя адреса IDX0/1	0000
RP0H	F108	ESFR-b	84	Регистр начальной конфигурации внешней шины	--XX
RTC_IC	F13A	ESFR-b	9D	Регистр контроля прерываний RTC	0000
RTC_IC1	F150	ESFR-b	A8	Регистр контроля прерывания 1 RTC	0000
RTC_IC2	F152	ESFR-b	A9	Регистр контроля прерывания 2 RTC	0000
RTC_IC3	F154	ESFR-b	AA	Регистр контроля прерывания 3 RTC	0000
RTCCMP1_hi	F042	ESFR	21	Регистр сравнения 1, старшие разряды RTC	0000
RTCCMP1_lo	F044	ESFR	22	Регистр сравнения 1, младшие разряды RTC	0200
RTCCMP2_hi	F046	ESFR	23	Регистр сравнения 2, старшие разряды RTC	FFFF
RTCCMP2_lo	F048	ESFR	24	Регистр сравнения 2, младшие разряды RTC	FFFF
RTCCMP3_hi	F04A	ESFR	25	Регистр сравнения 3, старшие разряды RTC	FFFF
RTCCMP3_lo	F04C	ESFR	26	Регистр сравнения 3, младшие разряды RTC	FFFF

Продолжение таблицы В.1

1	2	3	4	5	6
RTCCST	F030	ESFR	18	Регистр управления и статуса RTC	--00
RTCEIST	F034	ESFR	1A	Регистр состояния прерываний RTC	--00
RTCIEN	F036	ESFR	1B	Регистр разрешения прерываний RTC	--00
RTCLD_hi	F03E	ESFR	1F	Перезагружаемый регистр, старшие разряды RTC	0000
RTCLD_lo	F040	ESFR	20	Перезагружаемый регистр, младшие разряды RTC	0000
RTCRD_hi	F03A	ESFR	1D	Регистр реального времени, старшие разряды RTC	0000
RTCRD_lo	F03C	ESFR	1E	Регистр реального времени, младшие разряды RTC	0001
RTPRD	F038	ESFR	1C	Регистр прескалера RTC	0000
RTUDST	F032	ESFR	19	Регистр состояния обновлений (RTC)	--00
RWDATA	F06A	ESFR	35	Регистр данных режима RW Cerberus OCDS	0000
S0BG	FEB4	SFR	5A	Регистр таймера генератора скорости передачи ASC0	0000
S0CON	FFB0	SFR-b	D8	Регистр управления ASC0	0000
S0EIC	FF70	SFR-b	B8	Регистр контроля прерывания по ошибке ASC0	0000
S0FDV	FEB6	SFR	5B	Регистр дробного делителя ASC0	0000
S0RBUF	FEB2	SFR	59	Буферный регистр приемника ASC0	0000
S0RIC	FF6E	SFR-b	B7	Регистр контроля прерывания по приему ASC0	0000
S0TBIC	F19C	ESFR-b	CE	Регистр контроля прерывания по опустошению буфера передачи ASC0	0000
S0TBUF	FEB0	SFR	58	Буферный регистр передатчика ASC0	0000
S0TIC	FF6C	SFR-b	B6	Регистр контроля прерывания по передаче ASC0	0000
S1BG	FEAA	SFR	55	Регистр таймера генератора скорости передачи ASC1	0000
S1CON	F1B8	ESFR-b	DC	Регистр управления ASC1	0000
S1EIC	F192	ESFR-b	C9	Регистр контроля прерывания по ошибке ASC1	0000
S1FDV	FEAC	SFR	56	Регистр дробного делителя ASC1	0000
S1RBUF	FEA8	SFR	54	Буферный регистр приемника ASC1	0000
S1RIC	F18A	ESFR-b	C5	Регистр контроля прерывания по приему ASC1	0000

Продолжение таблицы В.1

1	2	3	4	5	6
S1TBIC	F13C	ESFR-b	9E	Регистр контроля прерывания по опустошению буфера передачи ASC1	0000
S1TBUF	FEA6	SFR	53	Буферный регистр передатчика ASC1	0000
S1TIC	F182	ESFR-b	C1	Регистр контроля прерывания по передаче ASC1	0000
SMBADDR	F0E2	ESFR	71	I2C регистр собственного адреса	--00
SMBCST	F0DE	ESFR	6F	I2C регистр управления и статуса	--00
SMBCTL1	F0E0	ESFR	70	I2C регистр управления 1	--00
SMBCTL2	F0E4	ESFR	72	I2C регистр управления 2	--00
SMBCTL3	F0E8	ESFR	74	I2C регистр управления 3	--00
SMBSDA	F0DA	ESFR	6D	I2C сдвиговый регистр данных	--XX
SMBST	F0DC	ESFR	6E	I2C регистр статуса	--00
SMBTOPR	F0E6	ESFR	73	Регистр прескалера времени ожидания SCL (I2C)	--00
SP	FE12	SFR	09	Регистр указателя стека ЦПУ	FC00
SSC0BR	F0B4	ESFR	5A	Регистр скорости пересылки SSC0	0000
SSC0CON	FFB2	SFR-b	D9	Регистр управления SSC0	0000
SSC0EIC	FF76	SFR-b	BB	Регистр контроля прерывания по ошибке SSC0	0000
SSC0ID	FFE4	SFR-b	F2	Регистр идентификации SSC0	45XX
SSC0RB	F0B2	ESFR	59	Буферный регистр приемника SSC0	0000
SSC0RIC	FF74	SFR-b	BA	Регистр контроля прерывания по приему SSC0	0000
SSC0TB	F0B0	ESFR	58	Буферный регистр передатчика SSC0	0000
SSC0TIC	FF72	SFR-b	B9	Регистр контроля прерывания по передаче SSC0	0000
SSC1BR	F05E	ESFR	2F	Регистр скорости пересылки SSC1	0000
SSC1CON	FF5E	SFR-b	AF	Регистр управления SSC1	0000
SSC1EIC	F126	ESFR-b	93	Регистр контроля прерывания по ошибке SSC1	0000
SSC1RB	F05C	ESFR	2E	Буферный регистр приемника SSC1	0000
SSC1RIC	F124	ESFR-b	92	Регистр контроля прерывания по приему SSC1	0000
SSC1TB	F05A	ESFR	2D	Буферный регистр передатчика SSC1	0000
SSC1TIC	F122	ESFR-b	91	Регистр контроля прерывания по передаче SSC1	0000
STKOV	FE14	SFR	0A	Регистр указателя переполнения стека ЦПУ	FA00

Продолжение таблицы В.1

1	2	3	4	5	6
STKUN	FE16	SFR	0B	Регистр указателя опустошения стека ЦПУ	FC00
SYSCON	FF12	SFR-b	89	Регистр конфигурации системы ЦПУ	XXXX
T0	FE50	SFR	28	Регистр таймера T0 (CAPCOM1)	0000
CC1_T01CON	FF50	SFR-b	A8	Регистр управления таймерами T0 и T1 (CAPCOM1)	0000
T0IC	FF9C	SFR-b	CE	Регистр управления прерываниями таймера 0 (CAPCOM1)	--00
T0REL	FE54	SFR	2A	Регистр загрузки таймера 0 (CAPCOM1)	0000
T1	FE52	SFR	29	Регистр таймера T1 (CAPCOM1)	0000
T1IC	FF9E	SFR-b	CF	Регистр управления прерываниями таймера 1 (CAPCOM1)	--00
T1REL	FE56	SFR	2B	Регистр загрузки таймера 1 (CAPCOM1)	0000
T2	FE40	SFR	20	Регистр таймера 2 (GPT1, GPT2)	0000
T2CON	FF40	SFR-b	A0	Регистр управления таймера 2 (GPT1, GPT2)	0000
T2IC	FF60	SFR-b	B0	Регистр управления прерыванием таймера 2 (GPT1, GPT2)	0000
T3	FE42	SFR	21	Регистр таймера 3 (GPT1, GPT2)	0000
T3CON	FF42	SFR-b	A1	Регистр управления таймера 3 (GPT1, GPT2)	0000
T3IC	FF62	SFR-b	B1	Регистр управления прерыванием таймера 3 (GPT1, GPT2)	0000
T4	FE44	SFR	22	Регистр таймера 4 (GPT1, GPT2)	0000
T4CON	FF44	SFR-b	A2	Регистр управления таймера 4 (GPT1, GPT2)	0000
T4IC	FF64	SFR-b	B2	Регистр управления прерыванием таймера 4 (GPT1, GPT2)	0000
T5	FE46	SFR	23	Регистр таймера 5 (GPT1, GPT2)	0000
T5CON	FF46	SFR-b	A3	Регистр управления таймера 5 (GPT1, GPT2)	0000
T5IC	FF66	SFR-b	B3	Регистр управления прерыванием таймера 5 (GPT1, GPT2)	0000
T6	FE48	SFR	24	Регистр таймера 6 (GPT1, GPT2)	0000
T6CON	FF48	SFR-b	A4	Регистр управления таймера 6 (GPT1, GPT2)	0000

Окончание таблицы В.1

1	2	3	4	5	6
T6IC	FF68	SFR-b	B4	Регистр управления прерыванием таймера 6 (GPT1, GPT2)	0000
T7	F050	ESFR	28	Регистр таймера 7 (CAPCOM2)	0000
CC2_T78CON	FF20	SFR-b	90	Регистр управления таймерами 7 и 8 (CAPCOM2)	0000
T7IC	F17A	ESFR-b	BD	Регистр управления прерываниями таймера T7 (CAPCOM2)	--00
T7REL	F054	ESFR	2A	Регистр загрузки таймера T7 (CAPCOM2)	0000
T8	F052	ESFR	29	Регистр таймера T8 (CAPCOM2)	0000
T8IC	F17C	ESFR-b	BE	Регистр управления прерываниями таймера T8 (CAPCOM2)	--00
T8REL	F056	ESFR	2B	Регистр загрузки таймера T8 (CAPCOM2)	0000
TFR	FFAC	SFR-b	D6	Регистр флагов ловушек	0000
WDT	FEAE	SFR	57	Регистр сторожевого таймера	0000
WDTCON	FFAE	SFR-b	D7	Регистр управления сторожевым таймером	008X
WDTIC	F19A	ESFR-b	CD	Регистр управления прерыванием сторожевого таймера	0000
XADRS1	F014	ESFR	0A	Регистр 1 выбора адреса XBUS	0000
XADRS2	F016	ESFR	0B	Регистр 2 выбора адреса XBUS	0000
XADRS3	F018	ESFR	0C	Регистр 3 выбора адреса XBUS	0000
XADRS4	F01A	ESFR	0D	Регистр 4 выбора адреса XBUS	0000
XADRS5	F01C	ESFR	0E	Регистр 5 выбора адреса XBUS	0000
XADRS6	F01E	ESFR	0F	Регистр 6 выбора адреса XBUS	0000
XBCON1	F114	ESFR-b	8A	Регистр 1 управления XBUS	0000
XBCON2	F116	ESFR-b	8B	Регистр 2 управления XBUS	0000
XBCON3	F118	ESFR-b	8C	Регистр 3 управления XBUS	0000
XBCON4	F11A	ESFR-b	8D	Регистр 4 управления XBUS	0000
XBCON5	F11C	ESFR-b	8E	Регистр 5 управления XBUS	0000
XBCON6	F11E	ESFR-b	8F	Регистр 6 управления XBUS	0000
XPERCON	F024	ESFR	12	Регистр управления X-периферии	0000
ZEROS	FF1C	SFR-b	8E	Регистр постоянного значения 0	0000
CSP	FE08	SFR	04	Регистр указателя сегмента кода (8 битов) ЦПУ	0000

Приложение Г
(обязательное)

Таблицы регистров областей SFR и ESFR, упорядоченных в зависимости от отношения к блоку микроконтроллера

В настоящем приложении регистры областей (E)SFR и XSFR микроконтроллера сгруппированы по принадлежности к тому или иному блоку.

Таблица Г.1 – Регистры конфигурации, выбора адреса и указателей

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
ADDRSEL1	FE18	SFR	0C	Регистр выбора адреса 1	0000
ADDRSEL2	FE1A	SFR	0D	Регистр выбора адреса 2	0000
ADDRSEL3	FE1C	SFR	0E	регистр выбора адреса 3	0000
ADDRSEL4	FE1E	SFR	0F	Регистр выбора адреса 4	0000
RP0H	F108	ESFR-b	84	Регистр начальной конфигурации внешней шины	--XX
BUSCON0	FF0C	SFR-b	86	Регистр конфигурации шины 0	0000
BUSCON1	FF14	SFR-b	8A	Регистр конфигурации шины 1	0000
BUSCON2	FF16	SFR-b	8B	Регистр конфигурации шины 2	0000
BUSCON3	FF18	SFR-b	8C	Регистр конфигурации шины 3	0000
BUSCON4	FF1A	SFR-b	8D	Регистр конфигурации шины 4	0000
ZEROS	FF1C	SFR-b	8E	Регистр постоянного значения 0	0000
ONES	FF1E	SFR-b	8F	Регистр постоянного значения 1	FFFF
DIP	F0F8	ESFR	7C	Регистр указателя машинной команды	0000
DIPX	F0FA	ESFR	7D	Регистр расширенного указателя инструкций	3000
TFR	FFAC	SFR-b	D6	Регистр флагов ловушек	0000

Таблица Г.2 – Регистры ЦПУ

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
CPUID	F00C	ESFR	06	Регистр идентификации ЦПУ	0420
SYSCON	FF12	SFR-b	89	Регистр конфигурации системы	XXXX
SP	FE12	SFR	09	Регистр указателя стека	FC00
CSP	FE08	SFR	04	Регистр указателя сегмента кода (8 битов)	0000
CP	FE10	SFR	08	Регистр контекстного указателя	FC00
STKUN	FE16	SFR	0B	Регистр указателя опустошения стека	FC00
STKOV	FE14	SFR	0A	Регистр указателя переполнения стека	FA00
DPP0	FE00	SFR	00	Регистр указателя нулевой страницы данных (10 битов)	0000
DPP1	FE02	SFR	01	Регистр указателя первой страницы данных (10 битов)	0001
DPP2	FE04	SFR	02	Регистр указателя второй страницы данных (10 битов)	0002
DPP3	FE06	SFR	03	Регистр указателя третьей страницы данных (10 битов)	0003
MDL	FE0E	SFR	07	Младший регистр умножения/деления	0000
MDH	FE0C	SFR	06	Старший регистр умножения/деления	0000
MDC	FF0E	SFR-b	87	Регистр управления умножением/делением	0000
PSW	FF10	SFR-b	88	Регистр слова состояния процессора	0000

Таблица Г.3 – Регистры контроллера периферийных событий PEC

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
1	2	3	4	5	6
PECSN0	FED0	SFR	68	Регистр указателя сегмента 0	0000
PECSN1	FED2	SFR	69	Регистр указателя сегмента 1	0000
PECSN10	FEE4	SFR	72	Регистр указателя сегмента 10	0000
PECSN11	FEE6	SFR	73	Регистр указателя сегмента 11	0000
PECSN12	FEB8	SFR	5C	Регистр указателя сегмента 12	0000
PECSN13	FEBA	SFR	5D	Регистр указателя сегмента 13	0000
PECSN14	FEBC	SFR	5E	Регистр указателя сегмента 14	0000
PECSN15	FEBE	SFR	5F	Регистр указателя сегмента 15	0000
PECSN2	FED4	SFR	6A	Регистр указателя сегмента 2	0000
PECSN3	FED6	SFR	6B	Регистр указателя сегмента 3	0000
PECSN4	FED8	SFR	6C	Регистр указателя сегмента 4	0000
PECSN5	FEDA	SFR	6D	Регистр указателя сегмента 5	0000
PECSN6	FEDC	SFR	6E	Регистр указателя сегмента 6	0000
PECSN7	FEDE	SFR	6F	Регистр указателя сегмента 7	0000
PECSN8	FEE0	SFR	70	Регистр указателя сегмента 8	0000
PECSN9	FEE2	SFR	71	Регистр указателя сегмента 9	0000
PECXC0	FEF0	SFR	78	Регистр расширенного управления каналом 0	0000
PECXC2	FEF2	SFR	79	Регистр расширенного управления каналом 2	0000
PECISNC	FFA8	SFR-b	D4	Регистр управления узлом прерываний	0000
PECC0	FEC0	SFR	60	Регистр управления PEC-каналом 0	0000
PECC1	FEC2	SFR	61	Регистр управления PEC-каналом 1	0000
PECC2	FEC4	SFR	62	Регистр управления PEC-каналом 2	0000
PECC3	FEC6	SFR	63	Регистр управления PEC-каналом 3	0000
PECC4	FEC8	SFR	64	Регистр управления PEC-каналом 4	0000
PECC5	FECA	SFR	65	Регистр управления PEC-каналом 5	0000
PECC6	FECC	SFR	66	Регистр управления PEC-каналом 6	0000
PECC7	FECE	SFR	67	Регистр управления PEC-каналом 7	0000
PECC8	FEE8	SFR	74	Регистр управления PEC-каналом 8	0000
PECC9	FEEA	SFR	75	Регистр управления PEC-каналом 9	0000
PECC10	FEEC	SFR	76	Регистр управления PEC-каналом 10	0000

Окончание таблицы Г.3

1	2	3	4	5	6
PECC11	FEEE	SFR	77	Регистр управления PEC-каналом 11	0000
PECC12	FEF8	SFR	7C	Регистр управления PEC-каналом 12	0000
PECC13	FEFA	SFR	7D	Регистр управления PEC-каналом 13	0000
PECC14	FEFC	SFR	7E	Регистр управления PEC-каналом 14	0000
PECC15	FEFE	SFR	7F	Регистр управления PEC-каналом 15	0000

Таблица Г.4 – Регистры блока умножения-накопления MAC

Название	Физи-ческий адрес (16 бит), hex	Тип	Адрес (8бит), hex	Описание	Значение после сброса (RESET), hex
MAL	FE5C	SFR	2E	Регистр младшего байта аккумулятора	0000
MAH	FE5E	SFR	2F	Регистр старшего байта аккумулятора	0000
QR0	F004	ESFR	02	Регистр смещения для указателей адреса GPR	0000
QX0	F000	ESFR	00	Регистр смещения 0 для указателя адреса IDX0/1	0000
QR1	F006	ESFR	03	Регистр смещения для указателей адреса GPR	0000
QX1	F002	ESFR	01	Регистр смещения 1 для указателя адреса IDX0/1	0000
MSW	FFDE	SFR-b	EF	Статусный регистр блока MAC	0200
MRW	FFDA	SFR-b	ED	Регистр повторения блока MAC	0000
IDX0	FF08	SFR-b	84	Регистр 0 указателя адреса	0000
IDX1	FF0A	SFR-b	85	Регистр 1 указателя адреса	0000
MCW	FFDC	SFR-b	EE	Регистр управления блока MAC	0000

Таблица Г.5 – Регистры портов

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
1	2	3	4	5	6
ALTSEL0P1H	F1EC	ESFR	F6	Регистр управления альтернативными функциями порта P1H	0000
ALTSEL0P1L	F1EA	ESFR	F5	Регистр управления альтернативными функциями порта P1L	0000
ALTSEL0P2	F1EE	ESFR	F7	Регистр управления альтернативными функциями порта P2	0000
ALTSEL0P3	F1F0	ESFR	F8	Регистр 0 управления альтернативными функциями порта P3	0000
ALTSEL0P4	F1F4	ESFR	FA	Регистр управления альтернативными функциями порта P4	0000
ALTSEL0P6	F1F6	ESFR	FB	Регистр управления альтернативными функциями порта P6	0000
ALTSEL0P7	F1F8	ESFR	FC	Регистр 0 управления альтернативными функциями порта P7	0000
ALTSEL0P9	F1FC	ESFR	FE	Регистр 0 управления альтернативными функциями порта P9	0000
ALTSEL1P3	F1F2	ESFR	F9	Регистр 1 управления альтернативными функциями порта P3	0000
ALTSEL1P7	F1FA	ESFR	FD	Регистр 1 управления альтернативными функциями порта P7	0000
ALTSEL1P9	F1FE	ESFR	FF	Регистр 1 управления альтернативными функциями порта P9	0000
DP0H	F102	ESFR-b	81	Регистр выбора направления порта P0H	0000
DP0L	F100	ESFR-b	80	Регистр выбора направления порта P0L	0000
DP1H	F106	ESFR-b	83	Регистр выбора направления порта P1H	0000
DP1L	F104	ESFR-b	82	Регистр выбора направления порта P1L	0000
DP2	FFC2	SFR-b	E1	Регистр выбора направления порта P2	0000
DP3	FFC6	SFR-b	E3	Регистр выбора направления порта P3	0000

Окончание таблицы Г.5

1	2	3	4	5	6
DP4	FFCA	SFR-b	E5	Регистр выбора направления порта P4	0000
DP6	FFCE	SFR-b	E7	Регистр выбора направления порта P6	0000
DP7	FFD2	SFR-b	E9	Регистр выбора направления порта P7	0000
DP9	FFD6	SFR-b	EB	Регистр выбора направления порта P9	0000
ODP2	F1C2	ESFR-b	E1	Регистр управления режимом с открытым стоком порта P2	0000
ODP3	F1C6	ESFR-b	E3	Регистр управления режимом с открытым стоком порта P3	0000
ODP4	F1CA	ESFR-b	E5	Регистр управления режимом с открытым стоком порта P4	0000
ODP6	F1CE	ESFR-b	E7	Регистр управления режимом с открытым стоком порта P6	0000
ODP7	F1D2	ESFR-b	E9	Регистр управления режимом с открытым стоком порта P7	0000
ODP9	F1D6	ESFR-b	EB	Регистр управления режимом с открытым стоком порта P9	0000
P0H	FF02	SFR-b	81	Старший регистр порта P0	00
P0L	FF00	SFR-b	80	Младший регистр порта P0	00
P1H	FF06	SFR-b	83	Старший регистр порта P1	00
P1L	FF04	SFR-b	82	Младший регистр порта P1	00
P2	FFC0	SFR-b	E0	Регистр данных порта P2	0000
P3	FFC4	SFR-b	E2	Регистр данных порта P3	0000
P4	FFC8	SFR-b	E4	Регистр данных порта P4	0000
P5	FFA2	SFR-b	D1	Регистр данных порта P5	0000
P6	FFCC	SFR-b	E6	Регистр данных порта P6	0000
P7	FFD0	SFR-b	E8	Регистр данных порта P7	0000
P9	FFD4	SFR-b	EA	Регистр данных порта P9	0000

Таблица Г.6 – Регистры шины XBUS

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
XADRS1	F014	ESFR	0A	Регистр 1 выбора адреса XBUS	0000
XADRS2	F016	ESFR	0B	Регистр 2 выбора адреса XBUS	0000
XADRS3	F018	ESFR	0C	Регистр 3 выбора адреса XBUS	0000
XADRS4	F01A	ESFR	0D	Регистр 4 выбора адреса XBUS	0000
XADRS5	F01C	ESFR	0E	Регистр 5 выбора адреса XBUS	0000
XADRS6	F01E	ESFR	0F	Регистр 6 выбора адреса XBUS	0000
XBCON1	F114	ESFR-b	8A	Регистр 1 управления XBUS	0000
XBCON2	F116	ESFR-b	8B	Регистр 2 управления XBUS	0000
XBCON3	F118	ESFR-b	8C	Регистр 3 управления XBUS	0000
XBCON4	F11A	ESFR-b	8D	Регистр 4 управления XBUS	0000
XBCON5	F11C	ESFR-b	8E	Регистр 5 управления XBUS	0000
XBCON6	F11E	ESFR-b	8F	Регистр 6 управления XBUS	0000
XPERCON	F024	ESFR	12	Регистр управления X-периферии	0000

Таблица Г.7 – Регистры генератора тактовых сигналов

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
FOCON	FFAA	SFR-b	5D	Регистр управления выходным тактовым сигналом	0000
GENCON	F1CC	ESFR-b	E6	Регистр управления вспомогательным осциллятором	0000
PLLCON	F1D0	ESFR-b	E8	Регистр управления PLL	XXXX

Таблица Г.8 – Регистры блоков таймеров GPT1 и GPT2

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
CAPREL	FE4A	SFR	25	Регистр захвата/перезагрузки	0000
GPTID	FFE6	SFR-b	F3	Регистр идентификации	58XX
T2	FE40	SFR	20	Регистр таймера 2	0000
T2CON	FF40	SFR-b	A0	Регистр управления таймера 2	0000
T3	FE42	SFR	21	Регистр таймера 3	0000
T3CON	FF42	SFR-b	A1	Регистр управления таймера 3	0000
T4	FE44	SFR	22	Регистр таймера 4	0000
T4CON	FF44	SFR-b	A2	Регистр управления таймера 4	0000
T5	FE46	SFR	23	Регистр таймера 5	0000
T5CON	FF46	SFR-b	A3	Регистр управления таймера 5	0000
T6	FE48	SFR	24	Регистр таймера 6	0000
T6CON	FF48	SFR-b	A4	Регистр управления таймера 6	0000

Таблица Г.9 – Регистры блока часов реального времени RTC

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
RTCCMP1_hi	F042	ESFR	21	Регистр сравнения 1, старшие разряды	0000
RTCCMP1_lo	F044	ESFR	22	Регистр сравнения 1, младшие разряды	0200
RTCCMP2_hi	F046	ESFR	23	Регистр сравнения 2, старшие разряды	FFFF
RTCCMP2_lo	F048	ESFR	24	Регистр сравнения 2, младшие разряды	FFFF
RTCCMP3_hi	F04A	ESFR	25	Регистр сравнения 3, старшие разряды	FFFF
RTCCMP3_lo	F04C	ESFR	26	Регистр сравнения 3, младшие разряды	FFFF
RTCCST	F030	ESFR	18	Регистр управления и статуса	--00
RTCEIST	F034	ESFR	1A	Регистр состояния прерываний	--00
RTCIEN	F036	ESFR	1B	Регистр разрешения прерываний	--00
RTCLD_hi	F03E	ESFR	1F	Перезагружаемый регистр, старшие разряды	0000
RTCLD_lo	F040	ESFR	20	Перезагружаемый регистр, младшие разряды	0000
RTCRD_hi	F03A	ESFR	1D	Регистр реального времени, старшие разряды	0000
RTCRD_lo	F03C	ESFR	1E	Регистр реального времени, младшие разряды	0001
RTPRD	F038	ESFR	1C	Регистр прескалера	0000
RTUDST	F032	ESFR	19	Регистр состояния обновлений	--00

Таблица Г.10 – Регистры блока АЦП (ADC)

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
ADC_CON	FFA0	SFR-b	D0	Регистр управления АЦП	0000
ADC_CON1	FFA6	SFR-b	D3	Регистр управления АЦП	0000
ADC_DAT	FEA0	SFR	50	Регистр результата АЦП	0000
ADC_DAT2	F0A0	ESFR	50	Регистр результата АЦП инжектированного канала	0000
ADC_SWITCH	FEA4	SFR	52	Регистр выбора режима модуля АЦП	0000
ADC2_CON	F1A0	ESFR-b	D0	Регистр управления АЦП2	0000
ADC2_CON1	F1A6	ESFR-b	D3	Регистр управления АЦП2	0000
ADC2_DAT	FEA2	SFR	51	Регистр результата АЦП2	0000
ADC2_DAT2	F0A2	ESFR	51	Регистр результата АЦП2 инжектированного канала	0000

Таблица Г.11 – Регистры блока захвата/сравнения CAPCOM1

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
CC1_T01CON	FF50	SFR-b	A8	Регистр управления таймерами 0 и 1	0000
CC1_DRM	FF5A	SFR-b	AD	Регистр двухрегистрового режима сравнения	0000
CC1_IOC	F062	ESFR	31	Регистр управления вводом-выводом	0000
CC1_OUT	FF5C	SFR-b	AE	Регистр выводов сравнения	0000
CC1_SEE	FE2E	SFR	17	Регистр однократного срабатывания	0000
CC1_SEM	FE2C	SFR	16	Регистр режима однократного срабатывания	0000
CC1_M0	FF52	SFR-b	A9	Регистр управления режимом захвата/сравнения 0	0000
CC1_M1	FF54	SFR-b	AA	Регистр управления режимом захвата/сравнения 1	0000
CC1_M2	FF56	SFR-b	AB	Регистр управления режимом захвата/сравнения 2	0000
CC1_M3	FF58	SFR-b	AC	Регистр управления режимом захвата/сравнения 3	0000
T0	FE50	SFR	28	Регистр таймера T0	0000
T0REL	FE54	SFR	2A	Регистр загрузки таймера 0	0000
T1	FE52	SFR	29	Регистр таймера T1	0000
T1REL	FE56	SFR	2B	Регистр загрузки таймера 1	0000
CC0	FE80	SFR	40	Регистр захвата/сравнения 0	0000
CC1	FE82	SFR	41	Регистр захвата/сравнения 1	0000
CC2	FE84	SFR	42	Регистр захвата/сравнения 2	0000
CC3	FE86	SFR	43	Регистр захвата/сравнения 3	0000
CC4	FE88	SFR	44	Регистр захвата/сравнения 4	0000
CC5	FE8A	SFR	45	Регистр захвата/сравнения 5	0000
CC6	FE8C	SFR	46	Регистр захвата/сравнения 6	0000
CC7	FE8E	SFR	47	Регистр захвата/сравнения 7	0000
CC8	FE90	SFR	48	Регистр захвата/сравнения 8	0000
CC9	FE92	SFR	49	Регистр захвата/сравнения 9	0000
CC10	FE94	SFR	4A	Регистр захвата/сравнения 10	0000
CC11	FE96	SFR	4B	Регистр захвата/сравнения 11	0000
CC12	FE98	SFR	4C	Регистр захвата/сравнения 12	0000
CC13	FE9A	SFR	4D	Регистр захвата/сравнения 13	0000
CC14	FE9C	SFR	4E	Регистр захвата/сравнения 14	0000
CC15	FE9E	SFR	4F	Регистр захвата/сравнения 15	0000

Таблица Г.12 – Регистры блока захват/сравнения CAPCOM2

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
CC16	FE60	SFR	30	Регистр захвата/сравнения 16	0000
CC17	FE62	SFR	31	Регистр захвата/сравнения 17	0000
CC18	FE64	SFR	32	Регистр захвата/сравнения 18	0000
CC19	FE66	SFR	33	Регистр захвата/сравнения 19	0000
CC2_DRM	FF2A	SFR-b	95	Регистр двухрегистрового режима сравнения	0000
CC2_IOC	F066	ESFR	33	Регистр управления вводом/выводом	0000
CC2_OUT	FF2C	SFR-b	96	Регистр выводов сравнения	0000
CC2_SEE	FE2A	SFR	15	Регистр однократного срабатывания	0000
CC2_SEM	FE28	SFR	14	Регистр режима однократного срабатывания	0000
CC2_M4	FF22	SFR-b	91	Регистр управления режимом захвата/сравнения 4	0000
CC2_M5	FF24	SFR-b	92	Регистр управления режимом захвата/сравнения 5	0000
CC2_M6	FF26	SFR-b	93	Регистр управления режимом захвата/сравнения 6	0000
CC2_M7	FF28	SFR-b	94	Регистр управления режимом захвата/сравнения 7	0000
CC20	FE68	SFR	34	Регистр захвата/сравнения 20	0000
CC21	FE6A	SFR	35	Регистр захвата/сравнения 21	0000
CC22	FE6C	SFR	36	Регистр захвата/сравнения 22	0000
CC23	FE6E	SFR	37	Регистр захвата/сравнения 23	0000
CC24	FE70	SFR	38	Регистр захвата/сравнения 24	0000
CC25	FE72	SFR	39	Регистр захвата/сравнения 25	0000
CC26	FE74	SFR	3A	Регистр захвата/сравнения 26	0000
CC27	FE76	SFR	3B	Регистр захвата/сравнения 27	0000
CC28	FE78	SFR	3C	Регистр захвата/сравнения 28	0000
CC29	FE7A	SFR	3D	Регистр захвата/сравнения 29	0000
CC30	FE7C	SFR	3E	Регистр захвата/сравнения 30	0000
CC31	FE7E	SFR	3F	Регистр захвата/сравнения 31	0000
T7	F050	ESFR	28	Регистр таймера T7	0000
CC2_T78CON	FF20	SFR-b	90	Регистр управления таймерами 7 и 8	0000
T7REL	F054	ESFR	2A	Регистр загрузки таймера T7	0000
T8	F052	ESFR	29	Регистр таймера T8	0000
T8REL	F056	ESFR	2B	Регистр загрузки таймера T8	0000

Таблица Г.13 – Регистры сторожевого таймера WDT

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
WDT	FEAE	SFR	57	Регистр сторожевого таймера	0000
WDTCON	FFAE	SFR-b	D7	Регистр управления сторожевым таймером	008X

Таблица Г.14 – Регистры асинхронно/синхронного последовательного интерфейса ASC0

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
ASC0ID	FFE2	SFR-b	F1	Регистр идентификации ASC0	44XX
S0BG	FEB4	SFR	5A	Регистр таймера генератора скорости передачи ASC0	0000
S0CON	FFB0	SFR-b	D8	Регистр управления ASC0	0000
S0FDV	FEB6	SFR	5B	Регистр дробного делителя ASC0	0000
S0RBUF	FEB2	SFR	59	Буферный регистр приемника ASC0	0000
S0TBUF	FEB0	SFR	58	Буферный регистр передатчика ASC0	0000

Таблица Г.15 – Регистры асинхронно/синхронного последовательного интерфейса ASC1

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
S1BG	FEAA	SFR	55	Регистр таймера генератора скорости передачи ASC1	0000
S1CON	F1B8	ESFR-b	DC	Регистр управления ASC1	0000
S1FDV	FEAC	SFR	56	Регистр дробного делителя ASC1	0000
S1RBUF	FEA8	SFR	54	Буферный регистр приемника ASC1	0000
S1TBUF	FEA6	SFR	53	Буферный регистр передатчика ASC1	0000

Таблица Г.16 – Регистры высокоскоростного синхронного последовательного интерфейса SSC0

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
SSC0BR	F0B4	ESFR	5A	Регистр скорости пересылки	0000
SSC0CON	FFB2	SFR-b	D9	Регистр управления	0000
SSC0ID	FFE4	SFR-b	F2	Регистр идентификации	45XX
SSC0RB	F0B2	ESFR	59	Буферный регистр приемника	0000
SSC0TB	F0B0	ESFR	58	Буферный регистр передатчика	0000

Таблица Г.17 – Регистры высокоскоростного синхронного последовательного интерфейса SSC1

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
SSC1BR	F05E	ESFR	2F	Регистр скорости пересылки	0000
SSC1CON	FF5E	SFR-b	AF	Регистр управления	0000
SSC1RB	F05C	ESFR	2E	Буферный регистр приемника	0000
SSC1TB	F05A	ESFR	2D	Буферный регистр передатчика	0000

Таблица Г.18 – Регистры модуля I2C

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
CFG	F0EA	ESFR	75	Регистр конфигурации I2C	--00
SMBADDR	F0E2	ESFR	71	Регистр собственного адреса I2C	--00
SMBCST	F0DE	ESFR	6F	Регистр управления и статуса I2C	--00
SMBCTL1	F0E0	ESFR	70	Регистр управления 1 I2C	--00
SMBCTL2	F0E4	ESFR	72	Регистр управления 2 I2C	--00
SMBCTL3	F0E8	ESFR	74	Регистр управления 3 I2C	--00
SMBSDA	F0DA	ESFR	6D	Сдвиговый регистр данных I2C	--XX
SMBST	F0DC	ESFR	6E	Регистр статуса I2C	--00
SMBTOPR	F0E6	ESFR	73	Регистр прескалера времени ожидания SCL	--00

Таблица Г.19 – Регистры прерываний микроконтроллера

Название	Физический адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
1	2	3	4	5	6
S0RIC	FF6E	SFR-b	B7	Регистр контроля прерывания по приему (ASC0)	0000
S0TBIC	F19C	ESFR-b	CE	Регистр контроля прерывания по опустошению буфера передачи (ASC0)	0000
S0TIC	FF6C	SFR-b	B6	Регистр контроля прерывания по передаче (ASC0)	0000
S0EIC	FF70	SFR-b	B8	Регистр контроля прерывания по ошибке (ASC0)	0000
S1RIC	F18A	ESFR-b	C5	Регистр управления прерыванием по приему (ASC1)	0000
S1TBIC	F13C	ESFR-b	9E	Регистр управления прерыванием по опустошению буфера передатчика (ASC1)	0000
S1TIC	F182	ESFR-b	C1	Регистр управления прерыванием по передаче (ASC1)	0000
S1EIC	F192	ESFR-b	C9	Регистр управления прерыванием по ошибке (ASC1)	0000
SSC0RIC	FF74	SFR-b	BA	Регистр контроля прерывания по приему (SSC0)	0000
SSC0TIC	FF72	SFR-b	B9	Регистр контроля прерывания по передаче (SSC0)	0000
SSC0EIC	FF76	SFR-b	BB	Регистр контроля прерывания по ошибке (SSC0)	0000
SSC1RIC	F124	ESFR-b	92	Регистр контроля прерывания по приему (SSC1)	0000
SSC1TIC	F122	ESFR-b	91	Регистр контроля прерывания по передаче (SSC1)	0000
SSC1EIC	F126	ESFR-b	93	Регистр контроля прерывания по ошибке (SSC1)	0000
CC0IC	FF78	SFR-b	BC	Регистр управления прерываниями 0 (CAPCOM1)	0000
CC1IC	FF7A	SFR-b	BD	Регистр управления прерываниями 1 (CAPCOM1)	0000
CC2IC	FF7C	SFR-b	BE	Регистр управления прерываниями 2 (CAPCOM1)	0000
CC3IC	FF7E	SFR-b	BF	Регистр управления прерываниями 3 (CAPCOM1)	0000
CC4IC	FF80	SFR-b	C0	Регистр управления прерываниями 4 (CAPCOM1)	0000
CC5IC	FF82	SFR-b	C1	Регистр управления прерываниями 5 (CAPCOM1)	0000

Продолжение таблицы Г.19

1	2	3	4	5	6
CC6IC	FF84	SFR-b	C2	Регистр управления прерываниями 6 (CAPCOM1)	0000
CC7IC	FF86	SFR-b	C3	Регистр управления прерываниями 7 (CAPCOM1)	0000
CC8IC	FF88	SFR-b	C4	Регистр управления прерываниями 8 (CAPCOM1)	0000
CC9IC	FF8A	SFR-b	C5	Регистр управления прерываниями 9 (CAPCOM1)	0000
CC10IC	FF8C	SFR-b	C6	Регистр управления прерываниями 10 (CAPCOM1)	0000
CC11IC	FF8E	SFR-b	C7	Регистр управления прерываниями 11 (CAPCOM1)	0000
CC12IC	FF90	SFR-b	C8	Регистр управления прерываниями 12 (CAPCOM1)	0000
CC13IC	FF92	SFR-b	C9	Регистр управления прерываниями 13 (CAPCOM1)	0000
CC14IC	FF94	SFR-b	CA	Регистр управления прерываниями 14 (CAPCOM1)	0000
CC15IC	FF96	SFR-b	CB	Регистр управления прерываниями 15 (CAPCOM1)	0000
CC16IC	F160	ESFR-b	B0	Регистр управления прерываниями 16 (CAPCOM2)	0000
CC17IC	F162	ESFR-b	B1	Регистр управления прерываниями 17 (CAPCOM2)	0000
CC18IC	F164	ESFR-b	B2	Регистр управления прерываниями 18 (CAPCOM2)	0000
CC19IC	F166	ESFR-b	B3	Регистр управления прерываниями 19 (CAPCOM2)	0000
CC20IC	F168	ESFR-b	B4	Регистр управления прерываниями 20 (CAPCOM2)	0000
CC21IC	F16A	ESFR-b	B5	Регистр управления прерываниями 21 (CAPCOM2)	0000
CC22IC	F16C	ESFR-b	B6	Регистр управления прерываниями 22 (CAPCOM2)	0000
CC23IC	F16E	ESFR-b	B7	Регистр управления прерываниями 23 (CAPCOM2)	0000
CC24IC	F170	ESFR-b	B8	Регистр управления прерываниями 24 (CAPCOM2)	0000
CC25IC	F172	ESFR-b	B9	Регистр управления прерываниями 25 (CAPCOM2)	0000
CC26IC	F174	ESFR-b	BA	Регистр управления прерываниями 26 (CAPCOM2)	0000
CC27IC	F176	ESFR-b	BB	Регистр управления прерываниями 27 (CAPCOM2)	0000
CC28IC	F178	ESFR-b	BC	Регистр управления прерываниями 28 (CAPCOM2)	0000
CC29IC	F184	ESFR-b	C2	Регистр управления прерываниями 29 (CAPCOM2)	0000

Продолжение таблицы Г.19

1	2	3	4	5	6
CC30IC	F18C	ESFR-b	C6	Регистр управления прерываниями 30 (CAPCOM2)	0000
CC31IC	F194	ESFR-b	CA	Регистр управления прерываниями 31 (CAPCOM2)	0000
CCU6_EIC	F188	ESFR-b	C4	Регистр управления прерываниями по ошибке (CAPCOM6)	--00
CCU6_IC	F120	ESFR-b	90	Регистр управления прерываниями (CAPCOM6)	--00
CCU6_T12IC	F190	ESFR-b	C8	Регистр управления прерываниями таймера T12 (CAPCOM6)	--00
CCU6_T13IC	F198	ESFR-b	CC	Регистр управления прерываниями таймера T13 (CAPCOM6)	--00
CRIC	FF6A	SFR-b	B5	Регистр управления прерыванием захвата/перезагрузки (GPT1, GPT2)	0000
T0IC	FF9C	SFR-b	CE	Регистр управления прерываниями таймера 0 (CAPCOM1)	--00
T1IC	FF9E	SFR-b	CF	Регистр управления прерываниями таймера 1 (CAPCOM1)	--00
T2IC	FF60	SFR-b	B0	Регистр управления прерываниями таймера 2 (GPT1, GPT2)	0000
T3IC	FF62	SFR-b	B1	Регистр управления прерываниями таймера 3 (GPT1, GPT2)	0000
T4IC	FF64	SFR-b	B2	Регистр управления прерываниями таймера 4 (GPT1, GPT2)	0000
T5IC	FF66	SFR-b	B3	Регистр управления прерываниями таймерами 5 (GPT1, GPT2)	0000
T6IC	FF68	SFR-b	B4	Регистр управления прерываниями таймера 6 (GPT1, GPT2)	0000
T7IC	F17A	ESFR-b	BD	Регистр управления прерываниями таймера T7 (CAPCOM2)	--00
T8IC	F17C	ESFR-b	BE	Регистр управления прерываниями таймера T8 (CAPCOM2)	--00
RTC_IC	F13A	ESFR-b	9D	Регистр контроля прерываний (RTC)	0000
RTC_IC1	F150	ESFR-b	A8	Регистр контроля прерывания 1 (RTC)	0000

Окончание таблицы Г.19

1	2	3	4	5	6
RTC_IC2	F152	ESFR-b	A9	Регистр контроля прерывания 2 RTC	0000
RTC_IC3	F154	ESFR-b	AA	Регистр контроля прерывания 3 RTC	0000
I2C_DIC	F186	ESFR-b	C3	Регистр управления прерыванием (I2C)	--00
ADC_CIC	FF98	SFR-b	CC	Регистр контроля прерывания по завершению преобразования АЦП1	0000
ADC_EIC	FF9A	SFR-b	CD	Регистр контроля прерывания по ошибке АЦП1	0000
ADC2_CIC	F18E	ESFR-b	C7	Регистр контроля прерывания по завершению преобразования АЦП2	0000
ADC2_EIC	F196	ESFR-b	CB	Регистр контроля прерывания по ошибке АЦП2	0000
EXICON	F1C0	ESFR-b	E0	Регистр контроля внешних прерываний	0000
EXISEL0	F1D	ESFR-b	ED	Регистр 0 выбора источника внешних прерываний	0000
EXISEL1	F1D8	ESFR-b	EC	Регистр 1 выбора источника внешних прерываний	0000
EOPIC	F180	ESFR-b	C0	Регистр управления прерываниями	0000
PLLLOCK_IC	F19E	ESFR-b	CF	Регистр управления прерыванием по нахождению частоты PLL	0000
PLLUNLOCK_IC	F156	ESFR-b	AB	Регистр управления прерыванием по потере частоты PLL	0000
WDTIC	F19A	ESFR-b	CD	Регистр управления прерыванием сторожевого таймера WDT	0000

Таблица Г.20 – Регистры отладочной системы OCDS

Название	Физи-ческий адрес (16 бит), hex	Тип	Адрес (8 бит), hex	Описание	Значение после сброса (RESET), hex
DTREVT	F0F0	ESFR	78	Регистр управления комбинациями аппаратных отладочных событий	0000
DSWEVT	F0F4	ESFR	7A	Регистр управления отладочными событиями вывода BREAK и программного обеспечения	0000
DEXEVT	F0F2	ESFR	79	Регистр управления отладочными событиями вывода BREAK и программного обеспечения	0000
COMDATA	F068	ESFR	34	Регистр режима коммуникации Cerberus	
DBGSR	F0FC	ESFR	7E	Регистр состояния отладки	
IOSR	F06C	ESFR	36	Регистр статуса Cerberus	
DCMPSP	F0EC	ESFR	76	Регистр выбора и программирования для DCMPx	0000
DCMPDP	F0EE	ESFR	77	Регистр данных программирования для DCMPx	0000
RWDATA	F06A	ESFR	35	Регистр данных режима RW Cerberus	0000
DTIDR	F0D8	ESFR	6C	Регистр идентификации задачи системы отладки	0000

Приложение Д
(обязательное)

Подробное описание команд блока умножения-накопления MAC

CoABS **Абсолютная величина**

Группа Арифметические команды

Синтаксис CoABS

Операнд(ы) источника ACC → 40-битная величина со знаком

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← Abs(ACC)

Описание Вычисление абсолютной величины содержимого 40-битного аккумулятора ACC

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	0	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если содержимое аккумулятора ACC равно 8000000000_H. В противном случае не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Примечание – Поведение флага SV изменилось, чтобы гарантировать арифметическую корректность.

Формат инструкции

Мнемоника	Формат	Повтор
CoABS	A3 00 1A 00	Нет

CoABS **Абсолютная величина**

Группа Арифметические команды

Синтаксис CoABS op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← Abs ((op2) || (op1))

Описание Вычисление абсолютной величины 40-битного операнда источника и запись результата в 40-битный аккумулятор ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoABS	Rw_n, Rw_m	A3 nm CA 00	Нет
CoABS	Rw_n, [Rw_m⊗]	83 nm CA 0:0qqq	Нет
CoABS	[IDX_i⊗], [Rw_m⊗]	93 Xm CA 0:0qqq	Нет

CoADD **Суммирование**

Группа Арифметические команды

Синтаксис CoADD op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← (ACC) + (tmp)

Описание Суммирование 40-битного операнда и 40-битного содержимого аккумулятора ACC с сохранением результата в регистре ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. При выполнении команды могут использоваться режимы косвенной адресации, допускается два параллельных чтения памяти. Команда является повторяемой.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoADD	Rw_n, Rw_m	A3	nm	02	00	Нет
CoADD	Rw_n, [Rw_m⊗]	83	nm	02	rrrr:rqqq	Да
CoADD	[IDX_i⊗], [Rw_m⊗]	93	Xm	02	rrrr:rqqq	Да

CoADD2 **Суммирование**

Группа Арифметические команды

Синтаксис CoADD2 op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← 2 * ((op2) || (op1))
(ACC) ← (ACC) + (tmp)

Описание 40-битный операнд умножается на 2, затем произведение добавляется к 40-битному регистру ACC, полученный результат сохраняется в аккумуляторе ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. При выполнении команды могут использоваться режимы косвенной адресации, допускается два параллельных чтения памяти. Команда является повторяемой.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoADD2	Rw_n, Rw_m	A3 nm 42 00	Нет
CoADD2	Rw_n, [Rw_m⊗]	83 nm 42 rrrr:rqqq	Да
CoADD2	[IDX_i⊗], [Rw_m⊗]	93 Xm 42 rrrr:rqqq	Да

CoASHR **Арифметический сдвиг вправо с округлением**

Группа Команды сдвига

Синтаксис CoASHR op1, rnd

Операнд(ы) источника op1 → счетчик сдвига

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0
  (ACC [n]) ← (ACC [n + 1]) (n = 0 – 38)
  (count) ← (count) - 1
END WHILE
(ACC) ← (ACC) + 00008000_H
(MAL) ← 0

Описание Арифметический сдвиг регистра ACC вправо на количество битов, определяемых операндом op1. Полученный результат в дополнительном коде округляется перед записью в регистр ACC. Для сохранения знака ACC в значащий бит записывается 0 (если первоначально значащий бит был равен 0) или 1 (если первоначально значащий бит был равен 1). Величина сдвига может быть выбрана от 0 до 8 (включительно). Операнд op1 может быть представлен как 4-битной константой без знака, так и четырьмя младшими битами (без знака) прямо или косвенно адресованного операнда.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если при округлении произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoASHR	#data4, rnd	A3 00 B2 0sss:s000	Нет
CoASHR	Rw_n, rnd	A3 nn BA rrrr:r000	Да
CoASHR	[Rw_m⊗], rnd	83 mm BA rrrr:rqqq	Да

CoASHR **Арифметический сдвиг вправо**

Группа Команды сдвига

Синтаксис CoASHR op1

Операнд(ы) источника op1 → счетчик сдвига

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0
  (ACC [n]) ← (ACC [n + 1]) (n = 0 – 38)
  (count) ← (count) - 1
END WHILE

Описание Арифметический сдвиг регистра ACC вправо на количество битов, определяемых операндом op1. Для сохранения знака ACC в значащий бит записывается 0 (если первоначально значащий бит был равен 0) или 1 (если первоначально значащий бит был равен 1). Величина сдвига может быть выбрана от 0 до 8 (включительно), op1 может быть представлен как 4-битной константой без знака, так и четырьмя младшими битами (без знака) прямо или косвенно адресованного операнда. Установка бита MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	*	-	0	*	*	нет

SL Не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoASHR	#data4	A3 00 A2 0sss:s000	Нет
CoASHR	Rw_n	A3 nn AA rrrr:r000	Да
CoASHR	[$Rw_m\otimes$]	83 mm AA rrrr:rqqq	Да

CoCMP **Сравнение**

Группа Команды сравнения

Синтаксис CoCMP op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника Нет

Действие tmp ← (op2) || (op1)
(ACC) ⇔ (tmp)

Описание Вычитание из регистра ACC 40-битного операнда со знаком, обновление флагов N, Z и C регистра MSW не изменяет содержимое регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Установка бита MS регистра MCW не влияет на результат. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	-	-	*	*	*	нет

SL Не изменяется.

E Не изменяется.

SV Не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoCMP	Rw_n, Rw_m	A3 nm C2 00	Нет
CoCMP	Rw_n, [Rw_m⊗]	83 nm C2 0:0qqq	Нет
CoCMP	[IDX_i⊗], [Rw_m⊗]	93 Xm C2 0:0qqq	Нет

CoLOAD **Запись в аккумулятор**

Группа Арифметические команды

Синтаксис CoLOAD op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
tmp ← (op2) || (op1)
(ACC) ← 0 + (tmp)

Описание Запись 40-битного операнда источника в 40-битный регистр ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Дополнение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoLOAD	Rw_n, Rw_m	A3 nm 22 00	Нет
CoLOAD	Rw_n, [Rw_m⊗]	83 nm 22 0:0qqq	Нет
CoLOAD	[IDX_i⊗], [Rw_m⊗]	93 Xm 22 0:0qqq	Нет

CoLOAD- **Запись в аккумулятор**

Группа Арифметические команды

Синтаксис CoLOAD- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
tmp ← (op2) || (op1)
(ACC) ← 0 - (tmp)

Описание Запись 40-битного операнда источника в 40-битный регистр ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. 40-битный операнд записывается в аккумулятор ACC с противоположным знаком. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoLOAD-	Rw_n, Rw_m	A3 nm 2A 00	Нет
CoLOAD-	Rw_n, [Rw_m⊗]	83 nm 2A 0:0qqq	Нет
CoLOAD-	[IDX_i⊗], [Rw_m⊗]	93 Xm 2A 0:0qqq	Нет

CoLOAD2 **Запись в аккумулятор**

Группа Арифметические команды

Синтаксис CoLOAD2 op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
tmp ← 2 * ((op2) || (op1))
(ACC) ← 0 + (tmp)

Описание Запись 40-битного операнда источника в 40-битный регистр ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. 40-битный операнд умножается на 2 перед записью в аккумулятор. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoLOAD2	Rw_n, Rw_m	A3 nm 62 00	Нет
CoLOAD2	Rw_n, [Rw_m⊗]	83 nm 62 0:0qqq	Нет
CoLOAD2	[IDX_i⊗], [Rw_m⊗]	93 Xm 62 0:0qqq	Нет

CoLOAD2- **Запись в аккумулятор**

Группа Арифметические команды

Синтаксис CoLOAD2- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
tmp ← 2 * ((op2) || (op1))
(ACC) ← 0 - (tmp)

Описание Запись 40-битного операнда источника в 40-битный регистр ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. 40-битный операнд умножается на 2 и с противоположным знаком записывается в аккумулятор. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoLOAD2-	Rw_n, Rw_m	A3 nm 6A 00	Нет
CoLOAD2-	Rw_n, [Rw_m⊗]	83 nm 6A 0:0qqq	Нет
CoLOAD2-	[IDX_i⊗], [Rw_m⊗]	93 Xm 6A 0:0qqq	Нет

CoMAC **Умножение - накопление с округлением**

Группа Команды умножения/умножения-накопления

Синтаксис CoMAC op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
  (tmp) ← ((op1) * (op2)) << 1
  (ACC) ← (ACC) + (tmp) + 0000008000_H
ELSE
  (tmp) ← (op1) * (op2)
  (ACC) ← (ACC) + (tmp) + 0000008000_H
END IF
(MAL) ← 0

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее суммируется с 40-битным содержимым аккумулятора ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMAC	Rw_n, Rw_m, rnd	A3	nm	D1	00	Нет
CoMAC	Rw_n, [Rw_m⊗], rnd	83	nm	D1	rrrr:rqqq	Да
CoMAC	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	D1	rrrr:rqqq	Да

CoMAC **Умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMAC op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (tmp) ← ((op1) * (op2)) << 1
  (ACC) ← (ACC) + (tmp)
ELSE
  (tmp) ← (op1) * (op2)
  (ACC) ← (ACC) + (tmp)
END IF
```

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее суммируется с 40-битным содержимым аккумулятора ACC. Затем полученный результат записывается в регистр ACC. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMAC	Rw_n, Rw_m	A3	nm	D0	00	Нет
CoMAC	Rw_n, [Rw_m⊗]	83	nm	D0	rrrr:rqqq	Да
CoMAC	[IDX_i⊗], [Rw_m⊗]	93	Xm	D0	rrrr:rqqq	Да

CoMAC- **Умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMAC- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (tmp) ← ((op1) * (op2)) << 1
  (ACC) ← (ACC) - (tmp)
ELSE
  (tmp) ← (op1) * (op2)
  (ACC) ← (ACC) - (tmp)
END IF
```

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее вычитается из 40-битного содержимого аккумулятора ACC. Полученный результат записывается в регистр ACC. При выполнении команды допускается два параллельных чтения памяти.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMAC-	Rw_n, Rw_m	A3	nm	E0	00	Нет
CoMAC-	Rw_n, [Rw_m⊗]	83	nm	E0	rrrr:rqqq	Да
CoMAC-	[IDX_i⊗], [Rw_m⊗]	93	Xm	E0	rrrr:rqqq	Да

CoMACM **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACM op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← (((op1)) * ((op2))) << 1
(ACC) ← (ACC) + (tmp) + 0000008000_H
ELSE
(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
END IF
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее суммируется с 40-битным содержимым аккумулятора ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACM	[IDXi⊗], [Rwm⊗], rnd	93	Xm	D9	rrrr:rqqq	Да

CoMACM **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACM op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
   (tmp) ← ((op1) * (op2)) << 1
   (ACC) ← (ACC) + (tmp)
ELSE
   (tmp) ← (op1) * (op2)
   (ACC) ← (ACC) + (tmp)
END IF
((IDXi (- ⊗))) ← ((IDXi))
```

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее суммируется с 40-битным содержимым аккумулятора ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACM	[IDX_i⊗], [Rw_m⊗]	93 Xm D8 rrrr:rqqq	Да

CoMACM- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACM- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← ((op1) * (op2)) << 1
(ACC) ← (ACC) - (tmp)
ELSE
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)
END IF
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее вычитается из 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACM-	[IDX_i⊗], [Rw_m⊗]	93 Xm E8 rrrr:rqqq	Да

CoMACMR **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMR op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← (((op1)) * ((op2))) << 1
(ACC) ← (tmp) - (ACC) + 0000008000_H
ELSE
(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
END IF
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее из него вычитается с 40-битный аккумулятор ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMR	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	F9	rrrr:rqqq	Да

CoMACMR **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← (((op1)) * ((op2))) << 1
(ACC) ← (tmp) - (ACC)
ELSE
(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
END IF
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMR	[IDX_i⊗], [Rw_m⊗]	93	Xm	F8	rrrr:rqqq	Да

CoMACMRsu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRsu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 79 rrrr:rqqq	Да

CoMACMRsu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 78 rrrr:rqqq	Да

CoMACMRu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 39 rrrr:rqqq	Да

CoMACMRu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRu	[IDX_i⊗], [Rw_m⊗]	93 Xm 38 rrrr:rqqq	Да

CoMACMRus **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm B9 rrrr:rqqq	Да

CoMACMRus **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMRus	[IDX_i⊗], [Rw_m⊗]	93	Xm	B8	rrrr:rqqq	Да

CoMACMsu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	59	rrrr:rqqq	Да

CoMACMsu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMsu	[IDX_i⊗], [Rw_m⊗]	93	Xm	58	rrrr:rqqq	Да

CoMACMsu- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается. ACC → 40-битная величина со знаком

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 68 rrrr:rqqq	Да

CoMACMu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	19	rrrr:rqqq	Да

CoMACMu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX$_i$ (- ⊗))) ← ((IDX$_i$))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu	[IDX$_i$⊗], [Rw$_m$⊗]	93	Xm	18	rrrr:rqqq	Да

CoMACMu- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu-	[IDX_i⊗], [Rw_m⊗]	93	Xm	28	rrrr:rqqq	Да

CoMACMus **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 99 rrrr:rqqq	Да

CoMACMus **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX$_i$ (- ⊗))) ← ((IDX$_i$))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMus	[IDX$_i$⊗], [Rw$_m$⊗]	93	Xm	98	rrrr:rqqq	Да

CoMACMus- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMus-	[IDX_i⊗], [Rw_m⊗]	93 Xm A8 rrrr:rqqq	Да

CoMACR **Умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
  (tmp) ← ((op1) * (op2)) << 1
  (ACC) ← (tmp) - (ACC) + 0000008000_H
ELSE
  (tmp) ← (op1) * (op2)
  (ACC) ← (tmp) - (ACC) + 0000008000_H
END IF
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее, если установлен флаг MP, полученное произведение сдвигается на 1 бит влево, а затем из него вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACR	Rw_n, Rw_m, rnd	A3	nm	F1	00	Нет
CoMACR	Rw_n, [Rw_m⊗], rnd	83	nm	F1	rrrr:rqqq	Да
CoMACR	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	F1	rrrr:rqqq	Да

CoMACR **Умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
   (tmp) ← ((op1) * (op2)) << 1
   (ACC) ← (tmp) - (ACC)
ELSE
   (tmp) ← (op1) * (op2)
   (ACC) ← (tmp) - (ACC)
END IF
```

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее, если установлен флаг MP, полученное произведение сдвигается на 1 бит влево, а затем из него вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACR	Rw_n, Rw_m	A3	nm	F0	00	Нет
CoMACR	Rw_n, [Rw_m⊗]	83	nm	F0	rrrr:rqqq	Да
CoMACR	[IDX_i⊗], [Rw_m⊗]	93	Xm	F0	rrrr:rqqq	Да

CoMACRsu **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRsu	Rw_n, Rw_m, rnd	A3	nm	71	00	Нет
CoMACRsu	Rw_n, [Rw_m⊗], rnd	83	nm	71	rrrr:rqqq	Да
CoMACRsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	71	rrrr:rqqq	Да

CoMACRsu **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACRsu	Rw_n, Rw_m	A3 nm 70 00	Нет
CoMACRsu	Rw_n, [Rw_m⊗]	83 nm 70 rrrr:rqqq	Да
CoMACRsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 70 rrrr:rqqq	Да

CoMACRu **Умножение-накопление без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRu	Rw_n, Rw_m, rnd	A3	nm	31	00	Нет
CoMACRu	Rw_n, [Rw_m⊗], rnd	83	nm	31	rrrr:rqqq	Да
CoMACRu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	31	rrrr:rqqq	Да

CoMACRu **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACRu	Rw_n, Rw_m	A3 nm 30 00	Нет
CoMACRu	Rw_n, [Rw_m⊗]	83 nm 30 rrrr:rqqq	Да
CoMACRu	[IDX_i⊗], [Rw_m⊗]	93 Xm 30 rrrr:rqqq	Да

CoMACRus **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRus	Rw_n, Rw_m, rnd	A3	nm	B1	00	Нет
CoMACRus	Rw_n, [Rw_m⊗], rnd	83	nm	B1	rrrr:rqqq	Да
CoMACRus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	B1	rrrr:rqqq	Да

CoMACRus **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Дополнение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRus	Rw_n, Rw_m, rnd	A3	nm	B0	00	Нет
CoMACRus	Rw_n, [Rw_m⊗], rnd	83	nm	B0	rrrr:rqqq	Да
CoMACRus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	B0	rrrr:rqqq	Да

CoMACsu **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACsu	Rw_n, Rw_m, rnd	A3	nm	51	00	Нет
CoMACsu	Rw_n, [Rw_m⊗], rnd	83	nm	51	rrrr:rqqq	Да
CoMACsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	51	rrrr:rqqq	Да

CoMACsu **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACsu	Rw_n, Rw_m	A3 nm 50 00	Нет
CoMACsu	Rw_n, [Rw_m⊗]	83 nm 50 rrrr:rqqq	Да
CoMACsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 50 rrrr:rqqq	Да

CoMACsu- **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACsu-	Rw_n, Rw_m	A3 nm 60 00	Нет
CoMACsu-	Rw_n, [Rw_m⊗]	83 nm 60 rrrr:rqqq	Да
CoMACsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 60 rrrr:rqqq	Да

CoMACu **Умножение-накопление без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACu	Rw_n, Rw_m, rnd	A3	nm	11	00	Нет
CoMACu	Rw_n, [Rw_m⊗], rnd	83	nm	11	rrrr:rqqq	Да
CoMACu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	11	rrrr:rqqq	Да

CoMACu **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACu	Rw_n, Rw_m, rnd	A3	nm	10	00	Нет
CoMACu	Rw_n, [Rw_m⊗], rnd	83	nm	10	rrrr:rqqq	Да
CoMACu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	10	rrrr:rqqq	Да

CoMACu- **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение есть
*	*	*	*	*	*	

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACu-	Rw_n, Rw_m, rnd	A3	nm	20	00	Нет
CoMACu-	Rw_n, [Rw_m⊗], rnd	83	nm	20	rrrr:rqqq	Да
CoMACu-	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	20	rrrr:rqqq	Да

CoMACus **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus	Rw_n, Rw_m, rnd	A3 nm 91 00	Нет
CoMACus	Rw_n, [Rw_m⊗], rnd	83 nm 91 rrrr:rqqq	Да
CoMACus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 91 rrrr:rqqq	Да

CoMACus **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus	Rw_n, Rw_m	A3 nm 90 00	Нет
CoMACus	Rw_n, [Rw_m⊗]	83 nm 90 rrrr:rqqq	Да
CoMACus	[IDX_i⊗], [Rw_m⊗]	93 Xm 90 rrrr:rqqq	Да

CoMACus- **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus-	Rw_n, Rw_m	A3 nm A0 00	Нет
CoMACus-	Rw_n, [Rw_m⊗]	83 nm A0 rrrr:rqqq	Да
CoMACus-	[IDX_i⊗], [Rw_m⊗]	93 Xm A0 rrrr:rqqq	Да

CoMAX **Максимальное значение**

Группа Команды сравнения

Синтаксис CoMAX op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← max ((ACC) , (tmp))

Описание Сравнение 40-битного операнда со знаком со значением регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Если содержимое регистра ACC меньше, чем 40-битный операнд, то в регистр ACC записывается значение 40-битного операнда. В противном случае содержимое регистра ACC не изменяется. Установка бита MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	нет

SL Устанавливается, если содержимое аккумулятора ACC изменилось. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMAX	Rw_n, Rw_m	A3	nm	3A	00	Нет
CoMAX	Rw_n, [Rw_m⊗]	83	nm	3A	rrrr:rqqq	Да
CoMAX	[IDX_i⊗], [Rw_m⊗]	93	Xm	3A	rrrr:rqqq	Да

CoMIN **Минимальное значение**

Группа Команды сравнения

Синтаксис CoMIN op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← min ((ACC) , (tmp))

Описание Сравнение 40-битного операнда со знаком со значением регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Если содержимое регистра ACC больше, чем 40-битный операнд, то в регистр ACC записывается значение 40-битного операнда. В противном случае содержимое регистра ACC не изменяется. Установка бита MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	нет

SL Устанавливается, если содержимое аккумулятора ACC изменилось. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMIN	Rw_n, Rw_m	A3 nm 7A 00	Нет
CoMIN	Rw_n, [Rw_m⊗]	83 nm 7A rrrr:rqqq	Да
CoMIN	[IDX_i⊗], [Rw_m⊗]	93 Xm 7A rrrr:rqqq	Да

CoMOV **Пересылка из области памяти в область памяти**

Группа Команды пересылки данных

Синтаксис CoMOV op1, op2

Операнд(ы) источника op2 → WORD

Операнд(ы) приемника op1 → WORD

Действие (op1) ← (op2)

Описание Пересылка содержимого из области памяти, определяемой операндом источника op2, в область памяти, определяемой операндом приемника op1. Примечательно то, что в отличие от других команд, IDXi может адресовать всю область памяти. Данная команда не влияет на флаги блока MAC, но устанавливает флаги ЦПУ, как любая другая команда пересылки MOV.

Флаги состояния ЦПУ

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд операнда источника op2. В противном случае сбрасывается.

Примечание – CoMOV – единственная команда блока MAC, которая изменяет флаги ЦПУ. Флаги MAC не изменяются.

Формат инструкции

Мнемоника		Формат				Повтор
CoMOV	[IDX_i⊗], [Rw_m⊗]	D3	Xm	00	rrrr:rqqq	Да

CoMUL **Умножение со знаком, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(ACC) ← ((op1) * (op2)) << + 0000008000_H
ELSE
(ACC) ← (op1) * (op2) + 0000008000_H
END IF
(MAL) ← 0

Описание Умножение двух 16- битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Не изменяется, когда биты MP или MS равны нулю. В противном случае устанавливается в случае умножения 8000_H на 8000_H.

E Устанавливается, если бит MP равен единице, а MS равен нулю и в случае умножения 8000_H на 8000_H. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMUL	Rw_n, Rw_m, rnd	A3	nm	C1	00	Нет
CoMUL	Rw_n, [Rw_m⊗], rnd	83	nm	C1	0:0qqq	Нет
CoMUL	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	C1	0:0qqq	Нет

CoMUL **Умножение со знаком**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (ACC) ← ((op1) * (op2))
ELSE
  (ACC) ← (op1) * (op2)
END IF
```

Описание Умножение двух 16- битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат записывается в регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Не изменяется, когда биты MP или MS равны нулю. В противном случае устанавливается в случае умножения 8000_H на 8000_H.

E Устанавливается, если бит MP равен единице, а MS равен нулю и в случае умножения 8000_H на 8000_H. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMUL	Rw_n, Rw_m	A3	nm	C0	00	Нет
CoMUL	Rw_n, [Rw_m⊗]	83	nm	C0	0:0qqq	Нет
CoMUL	[IDX_i⊗], [Rw_m⊗]	93	Xm	C0	0:0qqq	Нет

CoMUL- **Умножение со знаком**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (ACC) ← - ((op1) * (op2))
ELSE
  (ACC) ← - ((op1) * (op2))
END IF
```

Описание Умножение двух 16- битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат с противоположным знаком записывается в регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMUL-	Rw_n, Rw_m	A3	nm	C8	00	Нет
CoMUL-	Rw_n, [Rw_m⊗]	83	nm	C8	0:0qqq	Нет
CoMUL-	[IDX_i⊗], [Rw_m⊗]	93	Xm	C8	0:0qqq	Нет

CoMULsu **Смешанное умножение, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16- битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMULsu	Rw_n, Rw_m, rnd	A3	nm	41	00	Нет
CoMULsu	Rw_n, [Rw_m⊗], rnd	83	nm	41	0:0qqq	Нет
CoMULsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	41	0:0qqq	Нет

CoMULsu **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2)

Описание Умножение двух 16- битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMULsu	Rw_n, Rw_m	A3 nm 40 00	Нет
CoMULsu	Rw_n, [Rw_m⊗]	83 nm 40 0:0qqq	Нет
CoMULsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 40 0:0qqq	Нет

CoMULsu- **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← - ((op1) * (op2))

Описание Умножение двух 16- битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат с противоположным знаком записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMULsu-	Rw_n, Rw_m,	A3 nm 48 00	Нет
CoMULsu-	Rw_n, [Rw_m⊗],	83 nm 48 0:0qqq	Нет
CoMULsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 48 0:0qqq	Нет

CoMULu **Умножение без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16- битных операндов источника без знака op1 и op2. Сначала происходит дополнение нулями получившегося 32-битного числа без знака. Затем полученный результат округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	0	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Всегда сбрасывается.

Примечание – Поведение флага E и флага SV изменилось, чтобы гарантировать арифметическую корректность. Если умножаются два больших 16-битных числа без знака, то результат не может быть представлен в 32-битном формате со знаком. В этом случае следует использовать как расширение ACC (автоматическое ограничение запрещено, MS = 0), так и ограничение до 32-битного значения со знаком (автоматическое ограничение разрешено, MS = 1).

Формат инструкции

Мнемоника		Формат	Повтор
CoMULu	Rw_n, Rw_m, rnd	A3 nm 01 00	Нет
CoMULu	Rw_n, [Rw_m⊗], rnd	83 nm 01 0:0qqq	Нет
CoMULu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 01 0:0qqq	Нет

CoMULu **Умножение без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2)

Описание Умножение двух 16- битных операндов источника без знака op1 и op2. Сначала происходит дополнение нулями получившегося 32-битного числа без знака. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	0	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Всегда сбрасывается.

Примечание – Поведение флага E и флага SV изменилось, чтобы гарантировать арифметическую корректность. Если умножаются два больших 16-битных числа без знака, то результат не может быть представлен в 32-битном формате со знаком. В этом случае следует использовать как расширение ACC (автоматическое ограничение запрещено, MS = 0), так и ограничение до 32-битного значения со знаком (автоматическое ограничение разрешено, MS = 1).

Формат инструкции

Мнемоника		Формат	Повтор
CoMULu	Rw_n, Rw_m	A3 nm 00 00	Нет
CoMULu	Rw_n, [Rw_m⊗]	83 nm 00 0:0qqq	Нет
CoMULu	[IDX_i⊗], [Rw_m⊗]	93 Xm 00 0:0qqq	Нет

CoMULu- **Умножение без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← - ((op1) * (op2))

Описание Умножение двух 16- битных операндов источника без знака op1 и op2. Сначала происходит дополнение нулями получившегося 32-битного числа без знака. Затем полученный результат записывается с противоположным знаком в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Всегда сбрасывается.

Примечание – Поведение флага E и флага SV изменилось, чтобы гарантировать арифметическую корректность. Если умножаются два больших 16-битных числа без знака, то результат не может быть представлен в 32-битном формате со знаком. В этом случае следует использовать как расширение ACC (автоматическое ограничение запрещено, MS = 0), так и ограничение до 32-битного значения со знаком (автоматическое ограничение разрешено, MS = 1).

Формат инструкции

Мнемоника		Формат	Повтор
CoMULu-	Rw_n, Rw_m	A3 nm 08 00	Нет
CoMULu-	Rw_n, [Rw_m⊗]	83 nm 08 0:0qqq	Нет
CoMULu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 08 0:0qqq	Нет

CoMULus **Смешанное умножение, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16- битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMULus	Rw_n, Rw_m, rnd	A3	nm	81	00	Нет
CoMULus	Rw_n, [Rw_m⊗], rnd	83	nm	81	0:0qqq	Нет
CoMULus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	81	0:0qqq	Нет

CoMULus **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2)

Описание Умножение двух 16- битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMULus	Rw_n, Rw_m, rnd	A3 nm 80 00	Нет
CoMULus	Rw_n, [Rw_m⊗], rnd	83 nm 80 0:0qqq	Нет
CoMULus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 80 0:0qqq	Нет

CoMULus- **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULus- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← - ((op1) * (op2))

Описание Умножение двух 16- битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат с противоположным знаком записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMULus	Rw_n, Rw_m, rnd	A3	nm	88	00	Нет
CoMULus	Rw_n, [Rw_m⊗], rnd	83	nm	88	0:0qqq	Нет
CoMULus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	88	0:0qqq	Нет

CoNEG **Изменение знака аккумулятора**

Группа Арифметические команды

Синтаксис CoNEG

Операнд(ы) источника ACC → 40-битная величина со знаком

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← 0 - (ACC)

Описание Вычитание из нуля содержимого регистра ACC, запись получившегося значения в регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника	Формат	Повтор
CoNEG	A3 00 32 00	Нет

CoNEG **Изменение знака аккумулятора, округление**

Группа Арифметические команды

Синтаксис CoNEG rnd

Операнд(ы) источника ACC → 40-битная величина со знаком

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← 0 - (ACC) + 0000008000_H
(MAL) ← 0

Описание Вычитание из нуля содержимого регистра ACC, далее округление получившегося значения и запись его в регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение есть
*	*	*	*	*	*	

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoNEG	rnd	A3 00 72 00	Нет

CoNOP **Нет операции**

Группа Арифметические команды

Синтаксис CoNOP

Операнд(ы) источника Нет

Операнд(ы) приемника Нет

Действие Нет операции

Описание Изменение указателя адреса.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	-	-	-	-	-	нет

SL Не изменяется.

E Не изменяется.

SV Не изменяется.

C Не изменяется.

Z Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат				Повтор
CoNOP	[IDX_i⊗], [Rw_m⊗]	93	Xm	5A	rrrr:rqqq	Да
CoNOP	[IDX_i⊗]	93	X0	5A	rrrr:r000	Да
CoNOP	[Rw_m⊗]	93	0m	5A	rrrr:rqqq	Да

CoRND **Округление значения аккумулятора**

Группа Команды сдвига

Синтаксис CoRND

Операнд(ы) источника ACC → 40-битная величина со знаком

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (ACC) + 0000008000_H
(MAL) ← 0

Описание Округление содержимого регистра ACC с помощью добавления к нему числа 0000008000_H, запись получившегося значения в регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Примечание – Команда CoRND является частным случаем команды CoASHR (CoASHR #0, rnd).

Формат инструкции

Мнемоника	Формат	Повтор
CoRND	A3 00 B2 00	Нет

CoSHL **Логический сдвиг влево**

Группа Команды сдвига

Синтаксис CoSHL op1

Операнд(ы) источника op1 → счетчик сдвига

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
(count) ← (op1)
(C) ← (ACC [39])
DO WHILE ((count) ≠ 0)
   (C) ← (ACC [39])
   (ACC [n] ← (ACC [n - 1])      (n = 39 – 1)
   (ACC) [0] ← 0
   (count) ← (count - 1)
END WHILE
```

Описание Сдвиг влево содержимого 40-битного аккумулятора ACC на число битов, определяемое операндом op1. Младший бит результата соответственно обнуляется. Допускается сдвиг на величину от 0 до 8 (включительно). op1 может быть представлен как 4-битной константой без знака, так и четырьмя младшими битами (без знака) прямо или косвенно адресованного операнда.
При установке бита MS регистра MCW и при возникновении 32-битного переполнения или опустошения полученный результат становится равен $007FFFFFFF_H$ или $FF80000000_H$ соответственно.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoSHL	#data4	A3 00 82 0sss:s000	Нет
CoSHL	Rw_n	A3 nn 8A rrrr:r000	Да
CoSHL	[Rw_m⊗]	83 mm 8A rrrr:rqqq	Да

CoSHR **Логический сдвиг вправо**

Группа Команды сдвига

Синтаксис CoSHR op1

Операнд(ы) источника op1 → счетчик сдвига

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0)
  (ACC [n] ← (ACC [n + 1]) (n = 0 – 38)
  (ACC) [39] ← 0
  (count) ← (count - 1)
END WHILE

Описание Сдвиг вправо содержимого 40-битного аккумулятора ACC на число битов, определяемое операндом op1. Младший бит результата соответственно обнуляется. Допускается сдвиг на величину от 0 до 8 (включительно). op1 может быть представлен как 4-битной константой без знака, так и четырьмя младшими битами (без знака) прямо или косвенно адресованного операнда. Бит MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	*	-	0	*	*	нет

SL Не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoSHR	#data4	A3 00 92 0sss:s000	Нет
CoSHR	Rw_n	A3 nn 9A rrrr:r000	Да
CoSHR	$[Rw_m\otimes]$	83 mm 9A rrrr:rqqq	Да

CoSTORE **Запись в регистры блока MAC**

Группа Команды пересылки данных

Синтаксис CoSTORE op1, op2

Операнд(ы) источника op2 → WORD

Операнд(ы) приемника op1 → WORD

Действие (op1) ← (op2)

Описание Пересылка содержимого регистра блока MAC, определяемого операндом источника op2, в область памяти, определяемую операндом приемника op1.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	-	-	-	-	-	нет

SL Не изменяется.

E Не изменяется.

SV Не изменяется.

C Не изменяется.

Z Не изменяется.

N Не изменяется.

Примечание – В соответствие с действиями конвейера, команда CoSTORE не может следовать сразу за командой MOV, также использующей регистры блока MAC MSW, MAH, MAL, MAS, MRW или MCW. В таких случаях между CoSTORE и MOV должна быть вставлена команда NOP.

Формат инструкции

Мнемоника		Формат				Повтор
CoSTORE	Rw_n, CoReg	C3	nn	wwww:w000	00	Нет
CoSTORE	[$Rw_n\otimes$], CoReg	B3	nn	wwww:w000	rrrr:rqqq	Да

CoSUB **Вычитание**

Группа Арифметические команды

Синтаксис CoSUB op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← (ACC) - (tmp)

Описание Вычитание 40-битного операнда из 40-битного регистра ACC, сохранение полученного значения в регистре ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoSUB	Rw_n, [Rw_m⊗]	A3	nm	0A	00	Нет
CoSUB	Rw_n, [Rw_m⊗]	83	nm	0A	rrrr:rqqq	Да
CoSUB	[IDX_i⊗], [Rw_m⊗]	93	Xm	0A	rrrr:rqqq	Да

CoSUB2 **Вычитание**

Группа Арифметические команды

Синтаксис CoSUB2 op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← 2 * (op2) || (op1)
(ACC) ← (ACC) - (tmp)

Описание Вычитание 40-битного операнда из 40-битного регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Затем данный 40-битный операнд умножается на 2, а потом вычитается из регистра ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoSUB2	Rw_n, [Rw_m⊗]	A3	nm	4A	00	Нет
CoSUB2	Rw_n, [Rw_m⊗]	83	nm	4A	rrrr:rqqq	Да
CoSUB2	[IDX_i⊗], [Rw_m⊗]	93	Xm	4A	rrrr:rqqq	Да

CoSUB2R **Вычитание**

Группа Арифметические команды

Синтаксис CoSUB2R op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← 2 * (op2) || (op1)
(ACC) ← (tmp) - (ACC)

Описание Вычитание из 40-битного операнда 40-битного регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Затем данный 40-битный операнд умножается на 2, а потом из него вычитается регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoSUB2R	Rw_n, [Rw_m⊗]	A3	nm	52	00	Нет
CoSUB2R	Rw_n, [Rw_m⊗]	83	nm	52	rrrr:rqqq	Да
CoSUB2R	[IDX_i⊗], [Rw_m⊗]	93	Xm	52	rrrr:rqqq	Да

CoSUBR **Вычитание**

Группа Арифметические команды

Синтаксис CoSUBR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← (tmp) - (ACC)

Описание Вычитание из 40-битного операнда 40-битного регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoSUBR	Rw_n, [Rw_m⊗]	A3	nm	12	00	Нет
CoSUBR	Rw_n, [Rw_m⊗]	83	nm	12	rrrr:rqqq	Да
CoSUBR	[IDX_i⊗], [Rw_m⊗]	93	Xm	12	rrrr:rqqq	Да

Приложение Е
(обязательное)

Подробное описание команд микроконтроллера 1887ВЕ3Т

ADD **Целочисленное сложение**

Синтаксис ADD op1, op2

Действие (opl) ← (opl) + (op2)

Тип данных WORD

Описание Выполняется сложение содержимого двух операндов, представленных в дополнительном коде – источника op2 и приемника op1. Сумма сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда для указанного типа данных. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADD	Rw_n , Rw_m	00 nm	2
ADD	Rw_n, [Rw_i]	08 n:1 0 i i	2
ADD	Rw_n, [Rw_i +]	08 n:l l i i	2
ADD	Rw_n, #data3	08 n:0 ###	2
ADD	reg, #data16	06 RR ####	4
ADD	reg, mem	02 RR MM MM	4
ADD	mem, reg	04 RR MM MM	4

ADDB **Целочисленное сложение**

Синтаксис ADDB op1, op2

Действие (opl) ← (opl) + (op2)

Тип данных BYTE

Описание Выполняется сложение содержимого двух операндов, представленных в дополнительном коде – источника op2 и приемника op1. Сумма сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда для указанного типа данных. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDB	Rb_n , Rb_m	01 nm	2
ADDB	Rb_n, [Rb_i]	09 n:1 0 i i	2
ADDB	Rb_n, [Rb_i +]	09 n:l l i i	2
ADDB	Rb_n, #data3	09 n:0 ###	2
ADDB	reg, #data16	07 RR ## xx	4
ADDB	reg, mem	03 RR MM MM	4
ADDB	mem, reg	05 RR MM MM	4

ADDC **Целочисленное сложение с переносом**

Синтаксис ADDC op1, op2

Действие (opl) ← (opl) + (op2) + (C)

Тип данных WORD

Описание Выполняется сложение содержимого трех операндов – источника op2, приемника op1 и бита переноса C, где операнды op1 и op2 – числа в дополнительном коде. Сумма сохраняется в op1. Эта инструкция может использоваться для вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю и флаг Z был установлен до выполнения инструкции. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDC	Rw_n , Rw_m	10 nm	2
ADDC	Rw_n, [Rw_i]	18 n:10 i i	2
ADDC	Rw_n, [Rw_i +]	18 n:l l i i	2
ADDC	Rw_n, #data3	18 n:0 # # #	2
ADDC	reg, #data16	16 RR ## ##	4
ADDC	Reg, mem	12 RR MM MM	4
ADDC	mem, reg	14 RR MM MM	4

ADDCB **Целочисленное сложение с переносом**

Синтаксис ADDCB op1, op2

Действие (opl) ← (opl) + (op2) + (C)

Тип данных BYTE

Описание Выполняется сложение содержимого трех операндов – источника op2, приемника op1 и бита переноса C, где операнды op1 и op2 – числа в дополнительном коде. Сумма сохраняется в op1. Эта инструкция может использоваться для вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю и флаг Z был установлен до выполнения инструкции. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDCB	Rb_n , Rb_m	11 nm	2
ADDCB	Rb_n, [Rw_i]	19 n:10 i i	2
ADDCB	Rb_n, [Rw_i +]	19 n:l l i i	2
ADDCB	Rb_n, #data3	19 n:0 # # #	2
ADDCB	reg, #data16	17 RR # # xx	4
ADDCB	reg, mem	13 RR MM MM	4
ADDCB	mem, reg	15 RR MM MM	4

AND **Логическое «И»**

Синтаксис AND op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных WORD

Описание Выполняется побитовая операция логического «И» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
AND	Rw_n , Rw_m	60 nm	2
AND	Rw_n, [Rw_i]	68 n:10 i i	2
AND	Rw_n, [Rw_i +]	68 n:l l i i	2
AND	Rw_n, #data3	68 n:0 # # #	2
AND	reg, #data16	66 RR ## ##	4
AND	reg, mem	62 RR MM MM	4
AND	mem, reg	64 RR MM MM	4

ANDB **Логическое «И»**

Синтаксис ANDB op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных BYTE

Описание Выполняется побитовая операция логического «И» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ANDB	Rb_n , Rb_m	61 nm	2
ANDB	Rb_n, [Rw_i]	69 n:10 i i	2
ANDB	Rb_n, [Rw_i +]	69 n:11 i i	2
ANDB	Rb_n, #data3	69 n:0 # # #	2
ANDB	reg, #data8	67 RR ## xx	4
ANDB	reg, mem	63 RR MM MM	4
ANDB	mem, reg	65 RR MM MM	4

ASHR **Арифметический сдвиг в сторону младших адресов**

Синтаксис ASHR op1, op2

Действие
(count) ← (op1)
(V) ← 0, (C) ← 0
DO WHILE ((count) ≠ 0)
(V) ← (C) v (V), (C) ← ($op1_0$)
($op1_n$) ← ($op1_{n+1}$) [n = 0, …, 14]
(count) ← (count) – 1
END WHILE

Тип данных WORD

Описание Арифметически сдвигается содержимое приемника op1 на количество разрядов, определяемое источником op2. Сдвиг производится в сторону младших разрядов. Для сохранения знака операнда op1 старшие разряды результата заполняются нулями, если первоначально старший разряд был 0, или единицами, если старший разряд был 1. Флаг переполнения V используется как флаг округления. Младший значащий разряд сдвигается во флаг переноса C. Допустимые значения сдвига – от 0 до 15 включительно. Если op2 является регистром общего назначения, то используются только 4 младших бита.

Флаги состояния

E	Z	V	C	N
0	*	S	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если в любом цикле сдвига происходит переполнение (теряется 1 из флага C). Сбрасывается, если значение содержимого op2 равно 0.

C Принимает значение бита, сдвигаемого из младшего разряда op1. Сбрасывается, если значение содержимого op2 равно 0.

N Устанавливается, если установлен старший разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ASHR	Rw_n , Rw_m	AC nm	2
ASHR	Rw_n, #data4	BC #n	2

ATOMIC **Временное запрещение обработки запросов на прерывание**

Синтаксис ATOMIC op1

Действие

```
(count) ← opl   [1 ≤ opl ≤ 4]
IRQ–processing = Disable      // запрещение обработки
                              // запросов на
                              // прерывание
DO WHILE ( ( count ) ≠ 0 AND Class_B_IRQ ≠ TRUE)
      ...  // выполнение инструкции
      (count) ← (count) - 1
END WHILE
(count) = 0
IRQ_processing = Enable       //разрешение обработки
                              // запросов на
                              // прерывание
```

Описание Запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC на время выполнения указанного количества инструкций. Действие ATOMIC распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие ATOMIC, определяется операндом op1 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие ATOMIC прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
ATOMIC	#irang2	D1: 00 ## - 0	2

BAND **Битовое логическое «И»**

Синтаксис BAND op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных BIT

Описание Выполняется операция логического «И» над содержимым источника op2 и содержимым приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕЕ ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BAND	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	6A QQ ZZ qz	4

BCLR **Очистка бита**

Синтаксис BCLR op1

Действие (opl) ← 0

Тип данных BIT

Описание Обнуляется значение битового операнда op1. Эта инструкция обычно используется для управления системой и периферийными устройствами.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее значение указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
BCLR	$bitaddr_{Q.q}$	qE QQ	2

BCMP **Сравнение битов**

Синтаксис BCMP op1, op2

Действие (op1) ⇔ (op2)

Тип данных BIT

Описание Выполняется сравнение содержимого операнда op1 с содержимым операнда op2. Результат не сохраняется. Изменяются только флаги состояния.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BCMP	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	2A QQ ZZ qz	4

BFLDH **Битовое поле старшего байта**

Синтаксис BFLDH op1, op2, op3

Действие (tmp) ← (op1)
(high byte (tmp)) ← ((high byte (tmp) ^ ¬op2) v op3)
(op1) ← (tmp)

Тип данных WORD

Описание Производится побитная замена старшего байта содержимого приемника op1 значениями источника op3. Маской замены являются значения битов операнда op2. Если значение бита маски равно 1, то производится замена в соответствующем бите приемника, если значение бита маски равно 0, то замена не производится.

Флаги состояния

E	Z	V	C	N
0	*	0	0	*

E Всегда сбрасывается.

Z Устанавливается, если старший и младший байты результата равны нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд старшего байта результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
BFLDH	$bitoff_Q$, #mask8, #data8	1A QQ ## @@	4

BFLDL **Битовое поле младшего байта**

Синтаксис BFLDL op1, op2, op3

Действие (tmp) ← (op1)
(low byte (tmp) ← ((low byte (tmp) ^ ¬op2) v op3)
(op1) ← (tmp)

Тип данных WORD

Описание Производится побитная замена младшего байта содержимого приемника op1 значениями источника op3. Маской замены являются значения битов операнда op2. Если значение бита маски равно 1, то производится замена в соответствующем бите приемника, если значение бита маски равно 0, то замена не производится.

Флаги состояния

E	Z	V	C	N
0	*	0	0	*

E Всегда сбрасывается.

Z Устанавливается, если старший и младший байты результата op1 равны нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд старшего байта результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
BFLDL	$bitoff_Q$, #mask8, #data8	0A QQ @@ ##	4

BMOV **Битовая пересылка**

Синтаксис BMOV op1, op2

Действие (opl) ← (op2)

Тип данных BIT

Описание Содержимое источника op2 пересылается в приемник op1. Флаги устанавливаются соответственно биту источника.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния источника.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние источника.

Формат инструкции

Мнемоника		Формат	Размер
BMOV	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	4A QQ ZZ qz	4

BMOVN **Битовая пересылка с инверсией**

Синтаксис BMOVN op1, op2

Действие (opl) ← ¬ (op2)

Тип данных BIT

Описание Инвертированное значение источника op2 пересылается в приемник op1. Флаги устанавливаются соответственно биту источника.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Не изменяется.

Z Содержит инверсное значение предыдущего состояния источника.

V Не изменяется.

C Не изменяется.

N Содержит предыдущее состояние источника.

Формат инструкции

Мнемоника		Формат	Размер
BMOVN	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	3A QQ ZZ qz	4

BOR **Битовое «ИЛИ»**

Синтаксис BOR op1, op2

Действие (opl) ← (opl) v (op2)

Тип данных BIT

Описание Выполняется операция логического «ИЛИ» над содержимым источника op2 и приемника op1. Результат записывается в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BOR	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	5A QQ ZZ qz	4

BSET **Установка бита**

Синтаксис BSET op1

Действие (opl) ← 1

Тип данных BIT

Описание Содержимое операнда op1 устанавливается в 1. Эта инструкция обычно используется для управления системой и периферийными устройствами.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
BSET	$bitaddr_{Q.q}$	qF QQ	2

BXOR **Битовое «ИСКЛЮЧАЮЩЕЕ ИЛИ»**

Синтаксис BXOR op1, op2

Действие (opl) ← (opl) $\oplus$ (op2)

Тип данных BIT

Описание Выполняется операция «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BXOR	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	7A QQ ZZ qz	4

CALLA **Абсолютный вызов подпрограммы**

Синтаксис CALLA op1, op2

Действие

```
IF  (opl)  THEN
      (SP) ← (SP) - 2
      ( (SP) ) ← (IP)
      (IP) ← (op2)
ELSE
  ...  // выполнение  следующей инструкции
END  IF
```

Описание Если выполняется условие, указанное в op1 (см. коды условий перехода), то осуществляется переход внутри сегмента по адресу, указанному в op2. Значение указателя стека SP уменьшается на 2 и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Если условие не выполняется, никаких действий не производится, и следующая инструкция выполняется как обычно.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLA	cc, caddr	CA c0 MM MM	4

CALLI **Косвенный вызов подпрограммы**

Синтаксис CALLI op1, op2

Действие

```
IF (opl) THEN
      (SP) ← (SP) - 2
      ( (SP) ) ← (IP)
      (IP) ← (op2)
ELSE
  ...  // выполнение  следующей инструкции
END  IF
```

Описание Если выполняется условие, указанное в op1 (см. коды условий перехода), то осуществляется переход внутри сегмента по адресу, равному содержимому op2. Значение указателя стека SP уменьшается на 2 и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Если условие не выполняется, никаких действий не производится, и следующая инструкция выполняется как обычно.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLI	cc, [Rw_n]	AB cn	2

CALLR **Относительный вызов подпрограммы**

Синтаксис CALLR op1

Действие (SP) ← (SP) - 2
((SP)) ← (IP)
(IP) ← (IP) + opl * 2

Описание Осуществляется безусловный переход внутри сегмента по адресу, определяемому суммой указателя инструкций IP и удвоенного смещения op1. Смещение воспринимается как число в дополнительном коде. Значение указателя стека SP уменьшается на 2, и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Значение IR, используемое для вычисления адреса перехода, является адресом инструкции, следующей за CALLR.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLR	rel	BB rr	2

CALLS **Межсегментный вызов подпрограммы**

Синтаксис CALLS op1, op2

Действие
(SP) ← (SP) - 2
((SP)) ← (CSP)
(SP) ← (SP) - 2
((SP)) ← (IP)
(CSP) ← opl
(IP) ← op2

Описание Осуществляется безусловный переход по полному 24-битовому адресу. Номер сегмента определяется операндом op1, а внутрисегментное смещение – операндом op2. Содержимое указателя команд IP и указателя сегмента CSP помещаются на вершину системного стека. Значение указателя стека SP перед каждым занесением на стек уменьшается на 2. Поскольку IP всегда указывает на инструкцию, следующую переходом, значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLS	seg, caddr	DA SS MM MM	4

CMP **Целочисленное сравнение**

Синтаксис CMP op1, op2

Действие (opl) ⇔ (op2)

Тип данных WORD

Описание Сравнивается содержимое операнда op1 с содержимым операнда op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. Флаги устанавливаются по правилам вычитания. Операнды остаются неизменными.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMP	Rw_n , Rw_n	40 nm	2
CMP	Rw_n, [Rw_i]	48 n:10 i i	2
CMP	Rw_n, [Rw_i +]	48 n:l l i i	2
CMP	Rw_n, #data3	48 n:0 # # #	2
CMP	reg, #data16	46 RR ## ##	4
CMP	reg, mem	42 RR MM MM	4

CMPB **Целочисленное сравнение**

Синтаксис CMPB op1, op2

Действие (opl) ⇔ (op2)

Тип данных BYTE

Описание Сравнивается содержимое операнда op1 и операнда op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. Флаги устанавливаются по правилам вычитания. Операнды остаются неизменными.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPB	Rb_n , Rb_n	41 nm	2
CMPB	Rb_n, [Rw_i]	49 n:10 i i	2
CMPB	Rb_n, [Rw_i +]	49 n:ll i i	2
CMPB	Rb_n, #data3	49 n:0 # # #	2
CMPB	reg, #data16	47 RR ## xx	4
CMPB	reg, mem	43 RR MM MM	4

CMPD1 **Целочисленное сравнение с уменьшением на единицу**

Синтаксис CMPD1 op1, op2

Действие (opl) ⇔ (op2)
(opl) ← (opl) - 1

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 уменьшается на 1. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий бит результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPD1	Rw_n, #data4	A0 #n	2
CMPD1	Rw_n, #data16	A6 Fn ## ##	4
CMPD1	Rw_n, mem	A2 RR MM MM	4

CMPD2 **Целочисленное сравнение с уменьшением на 2**

Синтаксис CMPD2 op1, op2

Действие (opl) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) - 2

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 уменьшается на 2. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPD2	Rw_n, #data4	B0 #n	2
CMPD2	Rw_n, #data16	B6 Fn ## ##	4
CMPD2	Rw_n, mem	B2 RR MM MM	4

CMPI1 **Целочисленное сравнение с увеличением на единицу**

Синтаксис CMPI1 op1, op2

Действие (opl) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) + 1

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 увеличивается на 1. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т.е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPI1	Rw_n, #data4	80 #n	2
CMPI1	Rw_n, #data16	86 Fn ## ##	4
CMPI1	Rw_n, mem	82 Fn MM MM	4

CMPI2 **Целочисленное сравнение с увеличением на 2**

Синтаксис CMPI2 op1, op2

Действие (opl) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) + 2

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 увеличивается на 2. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPI2	Rw_n, #data4	90 #n	2
CMPI2	Rw_n, #data16	96 Fn ## ##	4
CMPI2	Rw_n, mem	92 Fn MM MM	4

CPL **Поразрядная инверсия**

Синтаксис CPL op1

Действие (opl) ← ¬ (opl)

Тип данных WORD

Описание Вычисляется поразрядная инверсия содержимого операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CPL	Rw_n	91 n0	2

CPLB **Поразрядная инверсия**

Синтаксис CPLB op1

Действие (opl) ← ¬ (opl)

Тип данных BYTE

Описание Вычисляется поразрядная инверсия содержимого операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CPLB	Rb_n,	B1 n0	2

DISWDT **Запрещение работы сторожевого таймера WDT**

Синтаксис DISWDT

Действие Запрещение работы сторожевого таймера

Описание Запрещается работа сторожевого таймера WDT. WDT начинает работать после сброса. После сброса эту инструкцию следует выполнить до выполнения инструкции перезапуска SRVWDT сторожевого таймера или инструкции конца инициализации EINIT. После выполнения одной из этих инструкций, DISWDT перестает действовать. Чтобы избежать случайного выполнения, инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
DISWDT	A5 5A A5 A5	4

DIV **Деление со знаком (16 разрядов/16 разрядов)**

Синтаксис DIV op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MDL) mod (opl)

Тип данных WORD

Описание Выполняется деление содержимого 16-разрядного приемника MDL (младшее слово 32-разрядного регистра MD) на содержимое 16-разрядного источника op1. Оба операнда рассматриваются как числа в дополнительном коде. Частное сохраняется в MDL, остаток сохраняется в MDH (старшем слове регистра MD).
Инструкция DIV выполняется 20 машинных циклов. DIV является прерываемой инструкцией. Если во время выполнения DIV произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю.
В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIV	Rw_n	4B nn	2

DIVL **Деление со знаком (32 разряда/16 разрядов)**

Синтаксис DIVL op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MD) mod (opl)

Тип данных WORD, DOUBLE WORD

Описание Выполняется деление содержимого 32-разрядного приемника MD на содержание 16-разрядного источника op1. Оба операнда рассматриваются как числа в дополнительном коде. Частное со знаком сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD). Инструкция DIVL выполняется 20 машинных циклов. DIVL является прерываемой инструкцией. Если во время выполнения DIVL произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVL	Rw_n	6B nn	2

DIVLU **Беззнаковое деление (32 разряда/16 разрядов)**

Синтаксис DIVLU op1

Действие (MDL) ← (MD) / (opl)
(MDH) ← (MD) mod (opl)

Тип данных WORD, DOUBLE WORD

Описание Выполняется беззнаковое деление содержимого 32-разрядного приемника MD на содержимое 16-разрядного источника op1. Частное сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD). Инструкция DIVLU выполняется 20 машинных циклов. DIVLU является прерываемой инструкцией. Если во время выполнения DIVLU произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVLU	Rw_n	7B nn	2

DIVU **Беззнаковое деление (16 разрядов/16разрядов)**

Синтаксис DIVU op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MDL) mod (opl)

Тип данных WORD

Описание Выполняется беззнаковое деление содержимого 16-разрядного приемника (младшее слово регистра MD) на содержимое 16-разрядного источника op1. Частное сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD).
Инструкция DIVU выполняется 20 машинных циклов. DIVU является прерываемой инструкцией. Если во время выполнения DIVU произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVU	Rw_n	5B nn	2

EINIT **Конец инициализации**

Синтаксис EINIT

Действие Конец инициализации.

Описание После сброса на вывод микроконтроллера RSTOUT# выдается уровень логического 0, который сохраняется до выполнения EINIT, после чего выдается уровень логической 1. Это позволяет программе посылать внешним цепям микроконтроллера сигнал об успешной инициализации. После выполнения EINIT, инструкция запрещения сторожевого таймера DISWDT игнорируется. Инструкция EINIT используется для завершения инициализационной части программы. Чтобы избежать случайного выполнения, инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
EINIT	B5 4A B5 B5	4

EXTR **Переадресация регистров**

Синтаксис EXTR op1

Действие

```
(count) ← opl [ 1 ≤ opl ≤ 4 ]
IRQ_processing = Disable        // запрещение обработки
                                // запросов на прерывание
SFR_range = Extended  // перенаправление 8-битовой
                      // адресации в область ESFR
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
    ... // выполнение следующей инструкции
 (count) ← (count) - 1
END WHILE
(count) = 0
SFR_range = Standard  // отмена перенаправления
IRQ_processing = Enable       // разрешение обработки
                              // запросов на прерывание
```

Описание При использовании адресации "reg", "bitoff", "bitaddr" производится обращение к расширенной области регистров специального назначения. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTR распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOP. Количество инструкций, на которые распространяется действие EXTR, определяется op1 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTR прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется

Z Не изменяется

V Не изменяется

C Не изменяется

N Не изменяется

Формат инструкции

Мнемоника		Формат	Размер
EXTR	#irang2	D1 : 10 ## -0	4

EXTP **Страничная переадресация**

Синтаксис EXTP op1, op2

Действие
```
(count) ← op2 [ 1 ≤ opl ≤ 4 ]
IRQ_processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Page = (opl)
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) - 1
END WHILE
(count) = 0
Data_Page = (DPPx)
IRQ_Processing = Enable      // разрешение обработки
                             // запросов на прерывание
```

Описание Заменяется стандартная схема адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 10-битовый номер страницы (разряды адреса A23, ..., A14) определяется не содержимым соответствующего регистра DPPx, а значением op1.
14-битовое внутристраничное смещение (разряды адреса A13, ..., A0) определяется младшими разрядами адреса, указанного в инструкциях.
На время выполнения указанного количества инструкцией запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTP распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTP, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTP прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
EXTP	Rw_m, #irang2	DC : 01 ## -m	2
EXTP	$\#pag_{10}$, #irang2	D7 : 01 ## -0 PP 0:00PP	4

EXTPR **Страничная переадресация и переадресация регистров**

Синтаксис EXTPR op1, op2

Действие

```
(count) ← op2 [ 1 ≤ op2 ≤ 4 ]
IRQ_Processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Page = (opl)
SFR_range = Extended      // перенаправление 8-битовой
                          // адресации в область ESFR
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) - 1
END WHILE
(count) = 0
Data_Page = (DPPx)
SFR_range = Standard   // отмена перенаправления
IRQ_processing = Enable    // разрешение обработки
                           // запросов на прерывание
```

Описание При использовании адресации "reg", "bitoff", "bitaddr" производится обращение к расширенной области регистров специального назначения. Кроме того, заменяется стандартная схема адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 10-битовый номер страницы (разряды адреса A23, ..., A14) определяется не содержимым соответствующего регистра DPPx, а значением op1. 14-битовое внутристраничное смещение (разряды адреса A13, ..., A0) определяется младшими разрядами адреса, указанного в инструкциях. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTPR распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTPR, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTPR прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
EXTPR	Rw_m, #irang2	DC : 11 ## -m	2
EXTPR	#pag10, #irang2	D7 : 11 ## -0 PP 0:00PP	4

EXTS **Сегментная переадресация**

Синтаксис EXTS op1, op2

Действие

```
(count) ← op2 [ 1 ≤ op ≤ 4 ]
IRQ_Processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Segment = (opl)
DO WHILE ( (count) ≠ 0 AND Class_B_Trap_Condition ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) – 1
END WHILE
(count) = 0
Data_Page = (DPPx)
IRQ_processing = Enable     // разрешение обработки
                            // запросов на прерывание
```

Описание Заменяет стандартную схему адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 8-битовый номер сегмента (разряды адреса A23, ..., A16) определяется значением op1. 16-битовое внутрисегментное смещение (разряды адреса A15, ..., A0) определяется адресом, указанным в инструкциях. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTS распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTS, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTS прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется

Z Не изменяется

V Не изменяется

C Не изменяется

N Не изменяется

Формат инструкции

Мнемоника		Формат	Размер
EXTS	Rw_m, #irang2	DC : 00 ## -m	2
EXTS	#seg, #irang2	D7 : 00 ## -0 SS 00	4

EXTSR **Сегментная переадресация и переадресация регистров**

Синтаксис EXTSR op1, op2

Действие

```
(count) ← op2 [ 1 ≤ op2 ≤ 4 ]
IRQ_Processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Segment = (opl)
SFR_range = Extended      // перенаправление 8-битовой
                          // адресации в область ESFR
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) – 1
END WHILE
(count) = 0
Data_Page = (DPPx)
SFR_range = Standard   // отмена перенаправления
IRQ_processing = Enable    // разрешение обработки
                           // запросов на прерывание
```

Описание При использовании адресации "reg", "bitoff", "bitaddr" производится обращение к расширенной области регистров специального назначения. Кроме того, заменяется стандартная схема адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 8-битовый номер сегмента (разряды адреса A23, ..., A16) определяется значением op1. 16-битовое внутрисегментное смещение (разряды адреса A15, ..., A0) определяется адресом, указанным в инструкциях. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTSR распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTSR, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTSR прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
EXTSR	Rw_m, #irang2	DC : 10 ## -m	2
EXTSR	#seg, #irang2	D7 : 10 ## -0 SS	4

IDLE **Режим ожидания**

Синтаксис IDLE

Действие Переход в режим ожидания.

Описание Осуществляет переход в режим ожидания. В этом режиме ЦПУ останавливается, в то время как периферийные устройства продолжают работу. Выход из режима осуществляется по прерыванию от внутрикристальных периферийных устройств или по внешнему прерыванию. Чтобы избежать случайного выполнения, инструкция реализована защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
IDLE	87 78 87 87	4

JB **Относительный переход, если бит установлен**

Синтаксис JB op1, op2

Действие

```
IF (opl) = 1 THEN
      (IP) ← (IP) + op2 * 2
ELSE
  ... // выполнение следующей инструкции
END IF
```

Тип данных BIT

Описание Если содержимое op1 равно 1, то осуществляется переход внутри сегмента по адресу, определяемому суммой содержимого указателя инструкций IP и удвоенного смещения op2. Смещение воспринимается как число в дополнительном коде. Для вычисления адреса перехода используется значение IR равное адресу инструкции, следующей за JB. Если указанный бит равен 0, то выполняется инструкция, следующая за JB.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JB	$bitaddr_{Q.q}$, rel	8A QQ rr qQ	4

JBC **Относительный переход с очисткой бита, если бит установлен**

Синтаксис JBC op1, op2

Действие

```
IF (opl) = 1 THEN
      (opl) = 0
      (IP) ← (IP) + op2 * 2
ELSE
  ... // выполнение следующей инструкции
END IF
```

Тип данных BIT

Описание Если содержимое op1 равно 1, то op1 обнуляется, и осуществляется переход внутри сегмента по адресу, определяемому суммой содержимого указателя инструкций IP и удвоенного смещения op2. Смещение воспринимается как число в дополнительном коде. Для вычисления адреса перехода используется значение IP, равное адресу инструкции, следующей за JBC. Если указанный бит равен 0, выполняется инструкция, следующая за JBC.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
JBC	$bitaddr_{Q.q}$, rel	AA QQ rr qQ	4

JMPA **Абсолютный условный переход**

Синтаксис JMPA op1, op2

Действие

```
IF (opl) = 1 THEN
      (IP) ← op2
ELSE
... // выполнение следующей инструкции
END IF
```

Описание Если выполняется условие, указанное в op1, осуществляется переход внутри сегмента по адресу, указанному в op2. Если условие не выполняется, никаких действий не производится, и выполняется инструкция, следующая за JMPA.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JMPA	cc, caddr	EA c0 MM MM	4

JMPI **Косвенный условный переход**

Синтаксис JMPI op1, op2

Действие

```
IF (opl) = 1 THEN
      (IP) ← op2
ELSE
... // выполнение следующей инструкции
END IF
```

Описание Если выполняется условие, указанное в op1, осуществляется переход внутри сегмента по адресу, равному содержимому op2. Если условие не выполняется, никаких действий не производится, и выполняется инструкция, следующая за JMPI.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JMPI	cc, [Rw_n]	9C cn	2

JMPR **Относительный условный переход**

Синтаксис JMPR op1, op2

Действие

```
IF (opl) = 1 THEN
      (IP) ← (IP) + op2 * 2
ELSE
... // выполнение следующей инструкции
END IF
```

Описание Если выполняется условие, указанное в op1, то осуществляется переход внутри сегмента по адресу, определяемому суммой указателя инструкций IP и удвоенного смещения, указанного в op2. Смещение воспринимается как число в дополнительном коде. Для вычисления адреса перехода используется значение IP, равное адресу инструкции, следующей за JMPR. Если условие не выполняется, никаких действий не производится, и выполняется инструкция, следующая за JMPR.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JMPR	cc, rel	cD rr	2

JMPS **Абсолютный межсегментный переход**

Синтаксис JMPS op1, op2

Действие (CSP) ← op1
(IP) ← op2

Описание Осуществляется безусловный переход по полному 24-битовому адресу. Номер сегмента определяется операндом op1, а внутрисегментное смещение – операндом op2.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JMPS	seg, caddr	FA SS MM MM	4

JNB **Относительный переход, если бит сброшен**

Синтаксис JNB op1, op2

Действие
```
IF (opl) = 1 THEN
(IP) ← (IP) + op2 * 2
ELSE
... // выполнение следующей инструкции
END IF
```

Тип данных BIT

Описание Если содержимое op1 равно 0, то осуществляется переход внутри сегмента по адресу, определяемому суммой содержимого указателя инструкций IP и удвоенного смещения op2. Смещение воспринимается как число в дополнительном коде. Для вычисления адреса перехода используется значение IR равное адресу инструкции, следующей за JNB. Если указанный бит равен 1, выполняется инструкция, следующая за JNB.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
JNB	$bitaddr_{Q.q}$, rel	9A QQ rr qQ	4

JNBS **Относительный переход с установкой бита, если бит сброшен**

Синтаксис JNBS op1, op2

Действие

```
IF (opl) = 0 THEN
      (op1) = 1
      (IP1) ← (IP) + op2 * 2
ELSE
... // выполнение следующей инструкции
END IF
```

Тип данных BIT

Описание Если содержимое op1 равно 0, то осуществляется переход внутри сегмента по адресу, определяемому суммой содержимого указателя инструкций IP и удвоенного смещения op2. Смещение воспринимается как число в дополнительном коде. Бит, указанный в op1, перед переходом устанавливается в 1. Для вычисления адреса перехода используется значение IP, равное адресу инструкции, следующей за JNBS. Если указанный бит равен 1, выполняется инструкция, следующая за JNBS.

Флаги состояния

E	Z	V	C	N
-	B#	-	-	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
JNBS	$bitaddr_{Q.q}$, rel	BA QQ rr qQ	4

MOV **Пересылка данных**

Синтаксис MOV op1, op2

Действие (opl) ← (op2)

Тип данных WORD

Описание Содержимое источника op2 пересылается в приемник op1. Пересылаемые данные анализируются, и выставляются флаги состояния.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд содержимого op2. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MOV	Rw_n, Rw_m	F0 nm	2
MOV	Rw_n, #data4	E0 #n	2
MOV	reg, #data16	E6 RR ## ##	4
MOV	Rw_n, [Rw_m]	A8 nm	2
MOV	Rw_n, [Rw_m +]	98 nm	2
MOV	[Rw_m], Rw_n	B8 nm	2
MOV	[-Rw_m], Rw_n	88 nm	2
MOV	[Rw_n], [Rw_m]	C8 nm	2
MOV	[Rw_n+], [Rw_m]	D8 nm	2
MOV	[Rw_n], [Rw_m +]	E8 nm	2
MOV	Rw_n, [Rw_m + #data16]	D4 nm ## ##	4
MOV	[Rw_m + #data16], Rw_n	C4 nm ## ##	4
MOV	[Rw_n], mem	84 0n MM MM	4
MOV	mem, [Rw_n]	94 0n MM MM	4
MOV	reg, mem	F2 RR MM MM	4
MOV	mem, reg	F6 RR MM MM	4

MOVB **Пересылка данных**

Синтаксис MOVB op1, op2

Действие (opl) ← (op2)

Тип данных BYTE

Описание Пересылается содержимое источника op2 в приемник op1. Пересылаемые данные анализируются и выставляются флаги состояния.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд содержимого op2. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MOVB	Rb_n, Rb_m	F1 nm	2
MOVB	Rb_n, #data4	E1 #n	2
MOVB	reg, #data8	E7 RR ## xx	4
MOVB	Rb_n, [Rw_m]	A9 nm	2
MOVB	Rb_n, [Rw_m +]	99 nm	2
MOVB	[Rw_m], Rb_n	B9 nm	2
MOVB	[-Rw_m], Rb_n	89 nm	2
MOVB	[Rw_n], [Rw_m]	C9 nm	2
MOVB	[Rw_n+], [Rw_m]	D9 nm	2
MOVB	[Rw_n], [Rw_m +]	E9 nm	2
MOVB	Rb_n, [Rw_m + #data16]	F4 nm ## ##	4
MOVB	[Rw_m + #data16], Rb_n	E4 nm ## ##	4
MOVB	[Rw_n], mem	A4 0n MM MM	4
MOVB	mem, [Rw_n]	B4 0n MM MM	4
MOVB	reg, mem	F3 RR MM MM	4
MOVB	mem, reg	F7 RR MM MM	4

MOVBS **Пересылка байта с расширением знака**

Синтаксис MOVBS op1, op2

Действие
(low byte opl) ← (op2)
IF ($op2_7$) = 1 THEN
(high byte (opl)) ← FFh
ELSE
(high byte (opl)) ← 00h
END IF

Тип данных WORD, BYTE

Описание Пересылается байт содержимого источника op2 в младший байт приемника op1. Пересылаемые данные анализируются, и выставляются флаги состояния. Старший байт приемника op1 заполняется знаковым (старшим) битом источника op2.

Флаги состояния

E	Z	V	C	N
0	*	-	-	0

E Всегда сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд содержимого op2. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MOVBS	Rw_n, Rb_m	D0 nm	2
MOVBS	reg, mem	D2 RR MM MM	4
MOVBS	mem, reg	D5 RR MM MM	4

MOVBSZ **Пересылка байта с очисткой старшего байта**

Синтаксис MOVBSZ op1, op2

Действие (low byte opl) ← (op2)
(high byte (opl)) ← 00h

Тип данных WORD, BYTE

Описание Пересылается байт содержимого источника op2 в младший байт приемника op1. Пересылаемые данные анализируются, и выставляются флаги состояния. Старший байт приемника op1 заполняется нулями.

Флаги состояния

E	Z	V	C	N
0	*	-	-	0

E Всегда сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Всегда сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MOVBSZ	Rw_n, Rb_m	C0 nm	2
MOVBSZ	reg, mem	C2 RR MM MM	4
MOVBSZ	mem, reg	C5 RR MM MM	4

MUL **Умножение со знаком**

Синтаксис MUL op1, op2

Действие (MD) ← (opl) * (op2)

Тип данных WORD

Описание Выполняется умножение содержимого двух 16-разрядных операндов op1 и op2. Оба операнда рассматриваются как числа в дополнительном коде. Результат (32 разряда) сохраняется в регистре MD. Инструкция MUL выполняется 10 машинных циклов. MUL является прерываемой инструкцией. Если во время выполнения MUL произошло прерывание, то устанавливается бит MULIP в регистре PSW, и умножение откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	0

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если результат не может быть представлен 16-разрядным числом. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MUL	Rw_n, Rw_m	0B nm	2

MULU **Беззнаковое умножение**

Синтаксис MULU op1, op2

Действие (MD) ← (opl) * (op2)

Тип данных WORD

Описание Выполняется беззнаковое умножение содержимого двух 16-разрядных операндов op1 и op2. Результат (32 разряда) сохраняется в регистре MD. Инструкция MULU выполняется 10 машинных циклов. MULU является прерываемой инструкцией. Если во время выполнения MULU произошло прерывание, то устанавливается бит MULIP в регистре PSW, и умножение откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если результат не может быть представлен 16-разрядным числом. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
MULU	Rw_n, Rw_m	1B nm	2

NEG **Инверсия знака**

Синтаксис NEG op1

Действие (opl) ← 0 - (opl)

Тип данных WORD

Описание Выполняется изменение знака операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий бит результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
NEG	Rw_n	81 n0	2

NEGB **Инверсия знака**

Синтаксис NEGB op1

Действие (opl) ← 0 - (opl)

Тип данных BYTE

Описание Выполняется изменение знака операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
NEGB	Rb_n	A1 n0	2

NOP **Нет операции**

Синтаксис NOP

Действие Пустая команда.

Описание Эта инструкция не вызывает никаких действий и не влияет на флаги.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
NOP	CC 00	2

OR **Логическое «ИЛИ»**

Синтаксис OR op1, op2

Действие (opl) ← (opl) v (op2)

Тип данных WORD

Описание Выполняется побитовая операция логического «ИЛИ» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
OR	Rw_n , Rw_m	70 nm	2
OR	Rw_n, [Rw_i]	78 n:10 i i	2
OR	Rw_n, [Rw_i +]	78 n:l l i i	2
OR	Rw_n, #data3	78 n:0 # # #	2
OR	reg, #data16	76 RR ## ##	4
OR	reg, mem	72 RR MM MM	4
OR	mem, reg	74 RR MM MM	4

ORB **Логическое «ИЛИ»**

Синтаксис ORB op1, op2

Действие (opl) ← (opl) v (op2)

Тип данных BYTE

Описание Выполняется побитовая операция логического «ИЛИ» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ORB	Rbw_n , Rb_m	71 nm	2
ORB	Rb_n, $[Rw_i]$	79 n:10 i i	2
ORB	Rb_n, $[Rw_i +]$	79 n:ll i i	2
ORB	Rb_n, #data3	79 n:0 # # #	2
ORB	reg, #data8	77 RR ## xx	4
ORB	reg, mem	73 RR MM MM	4
ORB	mem, reg	75 RR MM MM	4

PCALL **Абсолютный вызов подпрограмм с сохранением слова на стеке**

Синтаксис PCALL op1, op2

Действие
(tmp) ← (opl)
(SP) ← (SP) - 2
((SP)) ← (tmp)
(SP) ← (SP) - 2
((SP)) ← (IP)
(IP) ← op2

Тип данных WORD

Описание На вершину системного стека помещается содержимое операнда op1 и содержимое указателя инструкций IP, и осуществляется переход по адресу, определяемому операндом op2. Значение указателя стека SP перед каждым занесением на стек уменьшается на 2. Поскольку IP всегда указывает на инструкцию, следующую за PCALL, значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если содержимое операнда op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если содержимое операнда op1 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд содержимого op1. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
PCALL	reg, caddr	E2 RR MM MM	4

POP **Извлечь слово из системного стека**

Синтаксис POP op1

Действие (tmp) ← ((SP))
(SP) ← (SP) + 2
(opl) ← (tmp)

Тип данных WORD

Описание С вершины системного стека извлекается значение и помещается в приемник в op1. Затем указатель стека SP увеличивается на 2.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если извлекаемое значение равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если значение извлекаемого слова равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд извлекаемого значения. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
POP	reg	FC RR	2

PRIOR **Счетчик нормализации**

Синтаксис PRIOR op1, op2

Действие

(tmp) ← (op2)
(count) ← 0
DO WHILE ((tmp$_{15}$) ≠ 1 AND (count) ≠ 15 AND (op2) ≠0)
    (tmp$_n$) ← (tmp$_{n-1}$)
    (count) ← (count) + 1
END WHILE
    (op1) ← (count)

Тип данных WORD

Описание Вычисляется значение счетчика op1, показывающее количество битовых сдвигов в сторону старших разрядов, требуемых для нормализации содержимого операнда op2, (чтобы его старший значащий бит после сдвига был равен 1). Если содержимое op2 равно нулю, то содержимое op1 обнуляется, и устанавливается флаг Z, в противном случае флаг Z сбрасывается. Эта инструкция может использоваться для реализации вычислений с плавающей точкой.

Флаги состояния

E	Z	V	C	N
0	*	0	0	0

E Всегда сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Всегда сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
PRIOR	Rw$_n$, Rw$_m$	2B nm	2

PUSH **Поместить слово на стек**

Синтаксис PUSH op1

Действие (tmp) ← (opl)
(SP) ← (SP) - 2
((SP)) ← (tmp)

Тип данных WORD

Описание Помещает содержимое op1 на вершину системного стека после уменьшения указателя стека SP на 2.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если помещаемое значение равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если помещаемое слово равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд помещаемого значения. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
PUSH	reg	EC RR	2

PWRDN **Режим пониженного потребления**

Синтаксис PWRDN

Действие Микроконтроллер входит в режим пониженного потребления.

Описание Осуществляется переход всей внутренней периферии и ЦПУ в режим пониженного энергопотребления до прихода сигнала внешнего сброса микроконтроллера. Для защиты от неправильного выполнения эта инструкция является защищенной.
Выполнение команды PWRDN разрешено, когда к контакту немаскируемого запроса на прерывание NMI приложено напряжение уровня логического 0, в противном случае команда не выполняется.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
PWRDN	97 68 97 97	4

RET **Возврат из подпрограммы**

Синтаксис RET

Действие (IP) ← ((SP))
(SP) ← (SP) + 2

Описание Осуществляется возврат из подпрограммы. С вершины системного стека извлекается значение и помещается в указатель инструкций IP для адресации следующей исполняемой инструкции. После извлечения указатель стека увеличивается на 2. Возврат из подпрограммы, вызванной с помощью CALLI, CALLR или CALLA, осуществляется на инструкцию, следующую за вызовом.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
RET	CB 00	2

RETI **Возврат из подпрограммы обработки прерывания**

Синтаксис RETI

Действие
```
(IP) ← (SP))
(SP) ← (SP) + 2
IF (SYSCON.SGDTDIS = 0) THEN
      (CSP) ← ( (SP) )
      (SP) ← (SP) + 2
END IF
(PSW) ← ( (SP) )
(SP) ← (SP) + 2
```

Описание Осуществляется возврат из подпрограммы обработки прерывания. С вершины системного стека извлекаются слова и помещаются в IR CSP и PSW. После каждого извлечения указатель стека SP увеличивается на 2. Выполнение продолжается с инструкции, следующей за той, во время выполнения которой пришел запрос на прерывание.
Предыдущее состояние ЦПУ восстанавливается после извлечения PSW. Значение CSP извлекается только, если разрешена сегментация.

Флаги состояния

E	Z	V	C	N
S	S	S	S	S

E Восстанавливается при извлечении PSW из стека.

Z Восстанавливается при извлечении PSW из стека.

V Восстанавливается при извлечении PSW из стека.

C Восстанавливается при извлечении PSW из стека.

N Восстанавливается при извлечении PSW из стека.

Формат инструкции

Мнемоника	Формат	Размер
RETI	FB 88	2

RETP **Возврат из подпрограммы и извлечение слова**

Синтаксис RETP op1

Действие
(IP) ← ((SP))
(SP) ← (SP) + 2
(tmp) ← ((SP))
(SP) ← (SP) + 2
(opl) ← (tmp)

Тип данных WORD

Описание Осуществляется возврат из подпрограммы. С вершины системного стека извлекается слово и помещается в указатель инструкций IP для адресации следующей исполняемой инструкции. После извлечения указатель стека увеличивается на 2. Затем с измененной вершины системного стека извлекается значение и записывается в приемник op1, указатель стека еще раз увеличивается на 2. Возврат из подпрограммы, вызванной с помощью PCALL, осуществляется на инструкцию, следующую за вызовом.

Флаги состояния

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если извлекаемое в op1 значение равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если значение извлекаемого слова равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд извлекаемого в op1 значения. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
RETP	reg	EB RR	2

RETS **Межсегментный возврат из подпрограммы**

Синтаксис RETS

Действие
(IP) ← ((SP))
(SP) ← (SP) + 2
(CSP) ← ((SP))
(SP) ← (SP) + 2

Описание Осуществляется межсегментный возврат из подпрограммы. С вершины системного стека извлекаются два значения и помещаются в указатель инструкций IP и указатель номера сегмента CSP для адресации следующей исполняемой инструкции. После каждого извлечения указатель стека увеличивается на 2. Возврат из подпрограммы, вызванной с помощью CALLS, осуществляется на инструкцию, следующую за вызовом.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
RETS	DB 00	2

ROL **Циклический сдвиг в сторону старших разрядов**

Синтаксис ROL op1, op2

Действие

(count) ← (op2)
(C) ← 0
DO WHILE ((count) * 0)
    (C) ← (opl_{15})
    (opl_n) ← (opl_{n-1}) [n = 1, … , 15]
    (opl_0) ← (C)
    (count) ← (count) - 1
END WHILE

Тип данных WORD

Описание Осуществляется циклический сдвиг содержимого op1 в сторону старших разрядов на количество позиций, определяемое операндом op2. Разряд 15 циклически сдвигается в разряд 0 и во флаг переноса C. Допустимые значения количества сдвигов – от 0 до 15 включительно. Если количество сдвигов определяется содержимым РОН, то используется только четыре младших разряда.

Флаги состояния

E	Z	V	C	N
0	*	0	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Устанавливается в значение последнего выдвинутого из op1 старшего значащего разряда. Сбрасывается при нулевом количестве позиций.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ROL	Rw_n, Rw_m	0C nm	2
ROL	Rw_n, #data4	1C #n	2

ROR **Циклический сдвиг в сторону младших разрядов**

Синтаксис ROR op1, op2

Действие

(count) ← (op2)
(C) ← 0
(V) ← 0
DO WHILE ((count) ≠ 0)
    (V) ← (V) v (C)
    (C) ← (opl_0)
    (opl_n) ← (opl_{n+1}) [n = 0, …, 14]
    (opl_{15}) ← (C)
    (count) ← (count) - 1
END WHILE

Тип данных WORD

Описание Осуществляется циклический сдвиг содержимого op1 в сторону младших разрядов на количество позиций, определяемое операндом op2. Разряд 0 циклически сдвигается в разряд 15 и во флаг переноса C. Допустимые значения количества сдвигов – от 0 до 15 включительно. Если количество сдвигов определяется содержимым РОН, то используется только четыре младших разряда.

Флаги состояния

E	Z	V	C	N
0	*	S	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если на любом этапе сдвига значение 1 выдвигается из флага переноса C. Сбрасывается при нулевом количестве позиций.

C Устанавливается в значение последнего выдвинутого из op1 старшего значащего разряда. Сбрасывается при нулевом количестве позиций.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ROR	Rw_n, Rw_m	2C nm	2
ROR	Rw_n, #data4	3C #n	2

SCXT **Переключение контекста**

Синтаксис SCXT opl, op2

Действие
(tmpl) ← (opl)
(tmp2) ← (op2)
(SP) ← (SP) - 2
((SP)) ← (tmpl)
(opl) ← (tmp2)

Описание Осуществляется сохранение на вершине стека содержимого регистра и загрузка регистра новым значением. После сохранения, перед загрузкой, указатель стека уменьшается на 2. Эта инструкция используется в основном для смены банка РОН и сохранения MDC обработчиком прерывания.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
SCXT	reg, #data16	C6 RR ## ##	4
SCXT	reg, mem	D6 RR MM MM	4

SHL **Сдвиг в сторону старших адресов**

Синтаксис SHL op1, op2

Действие

(count) ← (op2)
(C) ← 0
DO WHILE ((count) ≠ 0)
    (C) ← (opl_{15})
    (opl_n) ← (opl_{n-1}) [n = 1, …, 15]
    (opl_0) ← 0
    (count) ← (count) – 1
END WHILE

Тип данных WORD

Описание Осуществляется сдвиг содержимого op1 в сторону старших разрядов на количество позиций, определяемое операндом op2. Младшие разряды op1 заполняются нулями. Старший значащий разряд выдвигается во флаг переноса C. Допустимые значения количества сдвигов – от 0 до 15 включительно. Если количество сдвигов определяется содержимым РОН, то используется только четыре младших разряда.

Флаги состояния

E	Z	V	C	N
0	*	0	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Устанавливается в значение последнего выдвинутого из op1 старшего значащего разряда. Сбрасывается при нулевом количестве позиций.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SHL	Rw_n, Rw_m	4C nm	2
SHL	Rw_n, #data4	5C #n	2

SHR **Сдвиг в сторону младших адресов**

Синтаксис SHR op1, op2

Действие
(count) ←(op2)
(C) ← 0, (V) ← 0
DO WHILE ((count) ≠ 0)
(V) ← (C) v (V)
(C) ← (opl_0), (opl_n) ← ($op1_{n+1}$) [n = 0, ..., 14]
(opl_{15}) ← 0
(count) ← (count) – 1
END WHILE

Тип данных WORD

Описание Осуществляется сдвиг содержимого op1 в сторону младших разрядов на количество позиций, определяемое операндом op2. Старшие разряды op1 заполняются нулями. Поскольку выдвигаемые биты представляют остаток, флаг переполнения V используется как флаг округления. Этот флаг совместно с флагом переноса C помогает пользователю определить, были ли потерянные биты остатка больше, меньше или равны половине младшего значащего бита. Допустимые значения количества сдвигов – от 0 до 15 включительно. Если количество сдвигов определяется содержимым РОН, то используется только четыре младших разряда.

Флаги состояния

E	Z	V	C	N
0	*	S	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен 0. В противном случае сбрасывается.

V Устанавливается, если в любом цикле операции сдвига из флага переноса выдвигается 1. Сбрасывается при нулевом количестве сдвигов.

C Устанавливается в значение последнего выдвинутого из op1 старшего значащего бита. Сбрасывается при нулевом количестве сдвигов.

N Устанавливается, если установлен старший значащий бит результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SHR	Rw_n, Rw_m	6C nm	2
SHR	Rw_n, #data4	7C #n	2

SRST **Программный сброс**

Синтаксис SRST

Действие Программный сброс.

Описание Выполняется программный сброс микроконтроллера. Программный сброс аналогичен внешнему аппаратному сбросу. Во избежание случайного выполнения эта инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
0	0	0	0	0

E Всегда сбрасывается.

Z Всегда сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Всегда сбрасывается.

Формат инструкции

Мнемоника	Формат	Размер
SRST	B7 48 B7 B7	4

SRVWDT **Перезапуск сторожевого таймера**

Синтаксис SRVWDT

Действие Перезапуск сторожевого таймера.

Описание Перезагружается старший байт счетчика сторожевого таймера значением из регистра WDTCON, и очищается младший байт. После выполнения данной инструкции невозможно запретить работу сторожевого таймера до следующего сброса микроконтроллера. Во избежание случайного выполнения, инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется

Z Не изменяется

V Не изменяется

C Не изменяется

N Не изменяется

Формат инструкции

Мнемоника	Формат	Размер
SRVWDT	A7 58 A7 A7	4

SUB **Целочисленное вычитание**

Синтаксис SUB op1, op2

Действие (opl) ← (opl) - (op2)

Тип данных WORD

Описание Выполняется двоичное вычитание в дополнительном коде содержимого источника op2 из содержимого приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SUB	Rw_n, Rw_m	20 nm	2
SUB	Rw_n, [Rw_i]	28 n: 10 i i	2
SUB	Rw_n, [Rw_i +]	28 n: 11 i i	2
SUB	Rw_n, #data3	28 n: 0# ##	2
SUB	reg, #data16	26 RR ## ##	4
SUB	reg, mem	22 RR MM MM	4
SUB	mem, reg	24 RR MM MM	4

SUBB **Целочисленное вычитание**

Синтаксис SUBB op1, op2

Действие (opl) ← (opl) - (op2)

Тип данных BYTE

Описание Выполняется двоичное вычитание в дополнительном коде содержимого источника op2 из содержимого приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SUBB	Rb_n, Rb_m	21 nm	2
SUBB	Rb_n, [Rw_i]	29 n:10 i i	2
SUBB	Rb_n, [Rw_i +]	29 n:11 i i	2
SUBB	Rb_n, #data3	29 n:0# ##	2
SUBB	reg, #data8	27 RR ## xx	4
SUBB	reg, mem	23 RR MM MM	4
SUBB	mem, reg	25 RR MM MM	4

SUBC **Целочисленное вычитание с переносом**

Синтаксис SUBC op1, op2

Действие (opl) ← (opl) - (C)

Тип данных WORD

Описание Выполняется вычитание содержимого источника op2 и бита переноса C из содержимого приемника op1. Результат сохраняется в op1. Эта инструкция используется для реализации вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю, и перед выполнением команды был установлен флаг Z. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий бит результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SUBC	Rw_n, Rw_m	30 nm	2
SUBC	Rw_n, [Rw_i]	38 n: 10 i i	2
SUBC	Rw_n, [Rw_i +]	38 n: 11 i i	2
SUBC	Rw_n, #data3	38 n: 0# ##	2
SUBC	reg, #data16	36 RR ## ##	4
SUBC	reg, mem	32 RR MM MM	4
SUBC	mem, reg	34 RR MM MM	4

SUBCB **Целочисленное вычитание с переносом**

Синтаксис SUBCB op1, op2

Действие (opl) ← (opl) – (op2) – (C)

Тип данных BYTE

Описание Выполняется вычитание содержимого источника op2 и бита переноса C из содержимого приемника op1. Результат сохраняется в op1. Эта инструкция используется для реализации вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю, и перед выполнением команды был установлен флаг Z. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий бит результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
SUBCB	Rb_n, Rb_m	31 nm	2
SUBCB	Rb_n, [Rw_i]	39 n: 10 i i	2
SUBCB	Rb_n, [Rw_i +]	39 n: 11 i i	2
SUBCB	Rb_n, #data3	39 n: 0# ##	2
SUBCB	reg, #data8	37 RR ## xx	4
SUBCB	reg, mem	33 RR MM MM	4
SUBCB	mem, reg	35 RR MM MM	4

TRAP **Программное прерывание**

Синтаксис TRAP op1

Действие

```
(SP) ← (SP) - 2
( (SP) ) ← (PSW)
IF (SYSCON.SGTDIS = 0) THEN
      (SP) ← (SP) - 2
      ( (SP) ) ← (CSP)
      (CSP) ← 0
END IF
(SP) ← (SP) – 2
( (SP) ) ← (IP)
(IP) ← opl * 4
```

Описание Производится обработка программного прерывания: на вершине системного стека сохраняется содержимое регистров PSW, CSP и IP, и производится переход по адресу вектора прерывания, определяемому операндом op1. Перед каждым сохранением значение указателя стека SP уменьшается на 2. Содержимое CSP сохраняется, если разрешена сегментация (бит SGTDIS в регистре SYSCON равен нулю). Переход на обработчик по адресу вектора прерывания осуществляется аппаратно после прихода запроса или программно с помощью инструкции TRAP. Обработчику не передается никакой дополнительной информации о том, какой тип перехода произошел. Сохранение PSW, CSP и IP производится аналогично аппаратному переходу. В отличие от аппаратного прерывания, уровень приоритета ЦПУ не изменяется. Для возврата из обработчика следует использовать инструкцию RETI.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
TRAP	#trap7	9B t: t t t 0	2

XOR **«ИСКЛЮЧАЮЩЕЕ ИЛИ»**

Синтаксис XOR op1, op2

Действие (opl) ← (opl) $\oplus$ (op2)

Тип данных WORD

Описание Выполняется операция побитового «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операнда op2 и содержимым приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
XOR	Rw_n, Rw_m	50 nm	2
XOR	Rw_n, [Rw_i]	58 n:10 i i	2
XOR	Rw_n, [Rw_i +]	58 n:ll i i	2
XOR	Rw_n, #data3	58 n:0 # # #	2
XOR	reg, #data16	56 RR ## ##	4
XOR	reg, mem	52 RR MM MM	4
XOR	mem, reg	54 RR MM MM	4

XORB **«ИСКЛЮЧАЮЩЕЕ ИЛИ»**

Синтаксис XORB op1, op2

Действие (opl) ← (opl) $\oplus$ (op2)

Тип данных BYTE

Описание Выполняется операция побитового «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операнда op2 и содержимым приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
XORB	Rb_n, Rb_m	51 nm	2
XORB	Rb_n, [Rw_i]	59 n:10 i i	2
XORB	Rb_n, [Rw_i +]	59 n:1 1 i i	2
XORB	Rb_n, #data3	59 n:0 # # #	2
XORB	reg, #data8	57 RR ## xx	4
XORB	reg, mem	53 RR MM MM	4
XORB	mem, reg	55 RR MM MM	4

Приложение Ж
(обязательное)

Таблицы регистров области XSFR, разделенных по принадлежности к блоку интерфейса CAN или блоку захвата/сравнения CAPCOM6

В настоящем приложении регистры области XSFR микроконтроллера 1887ВЕ3Т сгруппированы по принадлежности к блоку интерфейса CAN или блоку захвата/ сравнения CAPCOM6

Таблица Ж.1 – Регистры модуля CAN

Название	Физический адрес (16 бит), hex	Тип	Описание	Значение после сброса (RESET), hex
1	2	3	4	5
CLCL	E800	XSFR	Регистр управления частотой (младшее слово)	0003
CLCH	E802	XSFR	Регистр управления частотой (старшее слово)	0000
IDL	E808	XSFR	Регистр идентификации (младшее слово)	C051
IDH	E80A	XSFR	Регистр идентификации (старшее слово)	002B
FDRL	E80C	XSFR	Регистр делителя (младшее слово)	0000
FDRH	E80E	XSFR	Регистр делителя (старшее слово)	0000
LIST0L	E900	XSFR	Регистр списка №0 не распределенных областей сообщений (младшее слово)	7F00
LIST0H	E902	XSFR	Регистр списка №0 не распределенных областей сообщений (старшее слово)	007F
LIST1L	E904	XSFR	Регистр списка №1 CAN узла 0 (младшее слово)	0000
LIST1H	E906	XSFR	Регистр списка №1 CAN узла 0 (старшее слово)	1000
LIST2L	E908	XSFR	Регистр списка №2 CAN узла 1 (младшее слово)	0000
LIST2H	E90A	XSFR	Регистр списка №2 CAN узла 1 (старшее слово)	1000
LIST3L	E90C	XSFR	Регистр свободного списка 3 (младшее слово)	0000
LIST3H	E90E	XSFR	Регистр свободного списка 3 (старшее слово)	1000
LIST4L	E910	XSFR	Регистр свободного списка 4 (младшее слово)	0000
LIST4H	E912	XSFR	Регистр свободного списка 4 (старшее слово)	1000
LIST5L	E914	XSFR	Регистр свободного списка 5 (младшее слово)	0000

Продолжение таблицы Ж.1

1	2	3	4	5
LIST5H	E916	XSFR	Регистр свободного списка 5 (старшее слово)	1000
LIST6L	E918	XSFR	Регистр свободного списка 6 (младшее слово)	0000
LIST6H	E91A	XSFR	Регистр свободного списка 6 (старшее слово)	1000
LIST7L	E91C	XSFR	Регистр свободного списка 7 (младшее слово)	0000
LIST7H	E91E	XSFR	Регистр свободного списка 7 (старшее слово)	1000
MSPND0L	E940	XSFR	Регистр 0 ждущих прерываний (младшее слово)	0000
MSPND0H	E942	XSFR	Регистр 0 ждущих прерываний (старшее слово)	0000
MSPND1L	E944	XSFR	Регистр 1 ждущих прерываний (младшее слово)	0000
MSPND1H	E946	XSFR	Регистр 1 ждущих прерываний (старшее слово)	0000
MSID0L	E980	XSFR	Регистр индекса сообщения CAN узла 0 (младшее слово)	0020
MSID0H	E982	XSFR	Регистр индекса сообщения CAN узла 0 (старшее слово)	0000
MSID1L	E984	XSFR	Регистр индекса сообщения CAN узла 1 (младшее слово)	0020
MSID1H	E986	XSFR	Регистр индекса сообщения CAN узла 1 (старшее слово)	0000
MSIMASKL	E9C0	XSFR	Регистр маски индекса сообщения (младшее слово)	0000
MSIMASKH	E9C2	XSFR	Регистр маски индекса сообщения (старшее слово)	0000
PANCTRL	E9C4	XSFR	Регистр панели команд (младшее слово)	0301
PANCTRH	E9C6	XSFR	Регистр панели команд (старшее слово)	0000
MCRL	E9C8	XSFR	Регистр управления (младшее слово)	0000
MCRH	E9CA	XSFR	Регистр управления (старшее слово)	0000
MITRL	E9CC	XSFR	Регистр прерываний (младшее слово)	0000
MITRH	E9CE	XSFR	Регистр прерываний (старшее слово)	0000
NCR0L	EA00	XSFR	Регистр управления CAN узла 0 (младшее слово)	0001
NCR0H	EA02	XSFR	Регистр управления CAN узла 0 (старшее слово)	0000
NSR0L	EA04	XSFR	Регистр состояния CAN узла 0 (младшее слово)	0000
NSR0H	EA06	XSFR	Регистр состояния CAN узла 0 (старшее слово)	0000
NIPR0L	EA08	XSFR	Регистр указателя прерываний CAN узла 0 (младшее слово)	0000
NIPR0H	EA0A	XSFR	Регистр указателя прерываний CAN узла 0 (старшее слово)	0000

Продолжение таблицы Ж.1

1	2	3	4	5
NPCR0L	EA0C	XSFR	Регистр управления портом CAN узла 0 (младшее слово)	0000
NPCR0H	EA0E	XSFR	Регистр управления портом CAN узла 0 (старшее слово)	0000
NBTR0L	EA10	XSFR	Регистр синхронизации битов CAN узла 0 (младшее слово)	0000
NBTR0H	EA12	XSFR	Регистр синхронизации битов CAN узла 0 (старшее слово)	0000
NECNT0L	EA14	XSFR	Регистр счетчика ошибок CAN узла 0 (младшее слово)	0000
NECNT0H	EA16	XSFR	Регистр счетчика ошибок CAN узла 0 (старшее слово)	0060
NFCR0L	EA18	XSFR	Регистр счетчика сообщений CAN узла 0 (младшее слово)	0000
NFCR0H	EA1A	XSFR	Регистр счетчика сообщений CAN узла 0 (старшее слово)	0000
NCR1L	EB00	XSFR	Регистр управления CAN узла 1 (младшее слово)	0001
NCR1H	EB02	XSFR	Регистр управления CAN узла 1 (старшее слово)	0000
NSR1L	EB04	XSFR	Регистр состояния CAN узла 1 (младшее слово)	0000
NSR1H	EB06	XSFR	Регистр состояния CAN узла 1 (старшее слово)	0000
NIPR1L	EB08	XSFR	Регистр указателя прерываний CAN узла 1 (младшее слово)	0000
NIPR1H	EB0A	XSFR	Регистр указателя прерываний CAN узла 1 (старшее слово)	0000
NPCR1L	EB0C	XSFR	Регистр управления портом CAN узла 1 (младшее слово)	0000
NPCR1H	EB0E	XSFR	Регистр управления портом CAN узла 1 (старшее слово)	0000
NBTR1L	EB10	XSFR	Регистр синхронизации битов CAN узла 1 (младшее слово)	0000
NBTR1H	EB12	XSFR	Регистр синхронизации битов CAN узла 1 (младшее слово)	0000
NECNT1L	EB14	XSFR	Регистр счетчиков ошибок CAN узла 1 (младшее слово)	0000
NECNT1H	EB16	XSFR	Регистр счетчиков ошибок CAN узла 1 (старшее слово)	0060
NFCR1L	EB18	XSFR	Регистр счетчика сообщений CAN узла 1 (младшее слово)	0000
NFCR1H	EB1A	XSFR	Регистр счетчика сообщений CAN узла 1 (старшее слово)	0000

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR0L	EC00	XSFR	Регистр управления функционированием области сообщения 0 (младшее слово)	0000
MOFCR0H	EC02	XSFR	Регистр управления функционированием области сообщения 0 (старшее слово)	0000
MOFGPR0L	EC04	XSFR	Регистр указателя FIFO/шлюза области сообщения 0 (младшее слово)	0000
MOFGPR0H	EC06	XSFR	Регистр указателя FIFO/шлюза области сообщения 0 (старшее слово)	0000
MOIPR0L	EC08	XSFR	Регистр указателя прерываний области сообщения 0 (младшее слово)	0000
MOIPR0H	EC0A	XSFR	Регистр указателя прерываний области сообщения 0 (старшее слово)	0000
MOAMR0L	EC0C	XSFR	Регистр маски области сообщения 0 (младшее слово)	FFFF
MOAMR0H	EC0E	XSFR	Регистр маски области сообщения 0 (старшее слово)	3FFF
MODATAL0L	EC10	XSFR	Регистр (младший) данных области сообщения 0 (младшее слово)	0000
MODATAL0H	EC12	XSFR	Регистр (младший) данных области сообщения 0 (старшее слово)	0000
MODATAH0L	EC14	XSFR	Регистр (старший) данных области сообщения 0 (младшее слово)	0000
MODATAH0H	EC16	XSFR	Регистр (старший) данных области сообщения 0 (старшее слово)	0000
MOAR0L	EC18	XSFR	Регистр арбитража области сообщения 0 (младшее слово)	0000
MOAR0H	EC1A	XSFR	Регистр арбитража области сообщения 0 (старшее слово)	0000
MOSTAT0L MOCTR0L	EC1C	XSFR	Регистр состояния области сообщения 0 (младшее слово) (только чтение) Регистр состояния области сообщения 0 (младшее слово) (только запись)	0000
MOSTAT0H MOCTR0H	EC1E	XSFR	Регистр состояния области сообщения 0 (старшее слово) (только чтение) Регистр состояния области сообщения 0 (старшее слово) (только запись)	0100

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR1L	EC20	XSFR	Регистр управления функционированием области сообщения 1 (младшее слово)	0000
MOFCR1H	EC22	XSFR	Регистр управления функционированием области сообщения 1 (старшее слово)	0000
MOFGPR1L	EC24	XSFR	Регистр указателя FIFO/шлюза области сообщения 1 (младшее слово)	0000
MOFGPR1H	EC26	XSFR	Регистр указателя FIFO/шлюза области сообщения 1 (старшее слово)	0000
MOIPR1L	EC28	XSFR	Регистр указателя прерываний области сообщения 1 (младшее слово)	0000
MOIPR1H	EC2A	XSFR	Регистр указателя прерываний области сообщения 1 (старшее слово)	0000
MOAMR1L	EC2C	XSFR	Регистр маски области сообщения 1 (младшее слово)	FFFF
MOAMR1H	EC2E	XSFR	Регистр маски области сообщения 1 (старшее слово)	3FFF
MODATAL1L	EC30	XSFR	Регистр (младший) данных области сообщения 1 (младшее слово)	0000
MODATAL1H	EC32	XSFR	Регистр (младший) данных области сообщения 1 (старшее слово)	0000
MODATAH1L	EC34	XSFR	Регистр (старший) данных области сообщения 1 (младшее слово)	0000
MODATAH1H	EC36	XSFR	Регистр (старший) данных области сообщения 1 (старшее слово)	0000
MOAR1L	EC38	XSFR	Регистр арбитража области сообщения 1 (младшее слово)	0000
MOAR1H	EC3A	XSFR	Регистр арбитража области сообщения 1 (старшее слово)	0000
MOSTAT1L MOCTR1L	EC3C	XSFR	Регистр состояния области сообщения 1 (младшее слово) (только чтение) Регистр состояния области сообщения 1 (младшее слово) (только запись)	0000
MOSTAT1H MOCTR1H	EC3E	XSFR	Регистр состояния области сообщения 1 (старшее слово) (только чтение) Регистр состояния области сообщения 1 (старшее слово) (только запись)	0200

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR2L	EC40	XSFR	Регистр управления функционированием области сообщения 2 (младшее слово)	0000
MOFCR2H	EC42	XSFR	Регистр управления функционированием области сообщения 2 (старшее слово)	0000
MOFGPR2L	EC44	XSFR	Регистр указателя FIFO/шлюза области сообщения 2 (младшее слово)	0000
MOFGPR2H	EC46	XSFR	Регистр указателя FIFO/шлюза области сообщения 2 (старшее слово)	0000
MOIPRL	EC48	XSFR	Регистр указателя прерываний области сообщения 2 (младшее слово)	0000
MOIPR2H	EC4A	XSFR	Регистр указателя прерываний области сообщения 2 (старшее слово)	0000
MOAMR2L	EC4C	XSFR	Регистр маски области сообщения 2 (младшее слово)	FFFF
MOAMR2H	EC4E	XSFR	Регистр маски области сообщения 2 (старшее слово)	3FFF
MODATAL2L	EC50	XSFR	Регистр (младший) данных области сообщения 2 (младшее слово)	0000
MODATAL2H	EC52	XSFR	Регистр (младший) данных области сообщения 2 (старшее слово)	0000
MODATAH2L	EC54	XSFR	Регистр (старший) данных области сообщения 2 (младшее слово)	0000
MODATAH2H	EC56	XSFR	Регистр (старший) данных области сообщения 2 (старшее слово)	0000
MOAR2L	EC58	XSFR	Регистр арбитража области сообщения 2 (младшее слово)	0000
MOAR2H	EC5A	XSFR	Регистр арбитража области сообщения 2 (старшее слово)	0000
MOSTAT2L MOCTR2L	EC5C	XSFR	Регистр состояния области сообщения 2 (младшее слово) (только чтение) Регистр состояния области сообщения 2 (младшее слово) (только запись)	0000
MOSTAT2H MOCTR2H	EC5E	XSFR	Регистр состояния области сообщения 2 (старшее слово) (только чтение) Регистр состояния области сообщения 2 (старшее слово) (только запись)	0301

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR3L	EC60	XSFR	Регистр управления функционированием области сообщения 3 (младшее слово)	0000
MOFCR3H	EC62	XSFR	Регистр управления функционированием области сообщения 3 (старшее слово)	0000
MOFGPR3L	EC64		Регистр указателя FIFO/шлюза области сообщения 3 (младшее слово)	0000
MOFGPR3H	EC66	XSFR	Регистр указателя FIFO/шлюза области сообщения 3 (старшее слово)	0000
MOIPR3L	EC68	XSFR	Регистр указателя прерываний области сообщения 3 (младшее слово)	0000
MOIPR3H	EC6A	XSFR	Регистр указателя прерываний области сообщения 3 (старшее слово)	0000
MOAMR3L	EC6C	XSFR	Регистр маски области сообщения 3 (младшее слово)	FFFF
MOAMR3H	EC6E	XSFR	Регистр маски области сообщения 3 (старшее слово)	3FFF
MODATAL3L	EC70	XSFR	Регистр (младший) данных области сообщения 3 (младшее слово)	0000
MODATAL3H	EC72	XSFR	Регистр (младший) данных области сообщения 3 (старшее слово)	0000
MODATAH3L	EC74	XSFR	Регистр (старший) данных области сообщения 3 (младшее слово)	0000
MODATAH3H	EC76	XSFR	Регистр (старший) данных области сообщения 3 (старшее слово)	0000
MOAR3L	EC78	XSFR	Регистр арбитража области сообщения 3 (младшее слово)	0000
MOAR3H	EC7A	XSFR	Регистр арбитража области сообщения 3 (старшее слово)	0000
MOSTAT3L MOCTR3L	EC7C	XSFR	Регистр состояния области сообщения 3 (младшее слово) (только чтение) Регистр состояния области сообщения 3 (младшее слово) (только запись)	0000
MOSTAT3H MOCTR3H	EC7E	XSFR	Регистр состояния области сообщения 3 (старшее слово) (только чтение) Регистр состояния области сообщения 3 (старшее слово) (только запись)	0402

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR4L	EC80	XSFR	Регистр управления функционированием области сообщения 4 (младшее слово)	0000
MOFCR4H	EC82	XSFR	Регистр управления функционированием области сообщения 4 (старшее слово)	0000
MOFGPR4L	EC84	XSFR	Регистр указателя FIFO/шлюза области сообщения 4 (младшее слово)	0000
MOFGPR4H	EC86	XSFR	Регистр указателя FIFO/шлюза области сообщения 4 (старшее слово)	0000
MOIPR4L	EC88	XSFR	Регистр указателя прерываний области сообщения 4 (младшее слово)	0000
MOIPR4H	EC8A	XSFR	Регистр указателя прерываний области сообщения 4 (старшее слово)	0000
MOAMR4L	EC8C	XSFR	Регистр маски области сообщения 4 (младшее слово)	FFFF
MOAMR4H	EC8E	XSFR	Регистр маски области сообщения 4 (старшее слово)	3FFF
MODATAL4L	EC90	XSFR	Регистр (младший) данных области сообщения 4 (младшее слово)	0000
MODATAL4H	EC92	XSFR	Регистр (младший) данных области сообщения 4 (старшее слово)	0000
MODATAH4L	EC94	XSFR	Регистр (старший) данных области сообщения 4 (младшее слово)	0000
MODATAH4H	EC96	XSFR	Регистр (старший) данных области сообщения 4 (старшее слово)	0000
MOAR4L	EC98	XSFR	Регистр арбитража области сообщения 4 (младшее слово)	0000
MOAR4H	EC9A	XSFR	Регистр арбитража области сообщения 4 (старшее слово)	0000
MOSTAT4L MOCTR4L	EC9C	XSFR	Регистр состояния области сообщения 4 (младшее слово) (только чтение) Регистр состояния области сообщения 4 (младшее слово) (только запись)	0000
MOSTAT4H MOCTR4H	EC9E	XSFR	Регистр состояния области сообщения 4 (старшее слово) (только чтение) Регистр состояния области сообщения 4 (старшее слово) (только запись)	0503

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR5L	ECA0	XSFR	Регистр управления функционированием области сообщения 5 (младшее слово)	0000
MOFCR5H	ECA2	XSFR	Регистр управления функционированием области сообщения 5 (старшее слово)	0000
MOFGPR5L	ECA4	XSFR	Регистр указателя FIFO/шлюза области сообщения 5 (младшее слово)	0000
MOFGPR5H	ECA6	XSFR	Регистр указателя FIFO/шлюза области сообщения 5 (старшее слово)	0000
MOIPR5L	ECA8	XSFR	Регистр указателя прерываний области сообщения 5 (младшее слово)	0000
MOIPR5H	ECAA	XSFR	Регистр указателя прерываний области сообщения 5 (старшее слово)	0000
MOAMR5L	ECAC	XSFR	Регистр маски области сообщения 5 (младшее слово)	FFFF
MOAMR5H	ECAE	XSFR	Регистр маски области сообщения 5 (старшее слово)	3FFF
MODATAL5L	ECB0	XSFR	Регистр (младший) данных области сообщения 5 (младшее слово)	0000
MODATAL5H	ECB2	XSFR	Регистр (младший) данных области сообщения 5 (старшее слово)	0000
MODATAH5L	ECB4	XSFR	Регистр (старший) данных области сообщения 5 (младшее слово)	0000
MODATAH5H	ECB6	XSFR	Регистр (старший) данных области сообщения 5 (старшее слово)	0000
MOAR5L	ECB8	XSFR	Регистр арбитража области сообщения 5 (младшее слово)	0000
MOAR5H	ECBA	XSFR	Регистр арбитража области сообщения 5 (старшее слово)	0000
MOSTAT5L MOCTR5L	ECBC	XSFR	Регистр состояния области сообщения 5 (младшее слово) (только чтение) Регистр состояния области сообщения 5 (младшее слово) (только запись)	0000
MOSTAT5H MOCTR5H	ECBE	XSFR	Регистр состояния области сообщения 5 (старшее слово) (только чтение) Регистр состояния области сообщения 5 (старшее слово) (только запись)	0604

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR6L	ECC0	XSFR	Регистр управления функционированием области сообщения 6 (младшее слово)	0000
MOFCR6H	ECC2	XSFR	Регистр управления функционированием области сообщения 6 (старшее слово)	0000
MOFGPR6L	ECC4	XSFR	Регистр указателя FIFO/шлюза области сообщения 6 (младшее слово)	0000
MOFGPR6H	ECC6	XSFR	Регистр указателя FIFO/шлюза области сообщения 6 (старшее слово)	0000
MOIPR6L	ECC8	XSFR	Регистр указателя прерываний области сообщения 6 (младшее слово)	0000
MOIPR6H	ECCA	XSFR	Регистр указателя прерываний области сообщения 6 (старшее слово)	0000
MOAMR6L	ECCC	XSFR	Регистр маски области сообщения 6 (младшее слово)	FFFF
MOAMR6H	ECCE	XSFR	Регистр маски области сообщения 6 (старшее слово)	3FFF
MODATAL6L	ECD0	XSFR	Регистр (младший) данных области сообщения 6 (младшее слово)	0000
MODATAL6H	ECD2	XSFR	Регистр (младший) данных области сообщения 6 (старшее слово)	0000
MODATAH6L	ECD4	XSFR	Регистр (старший) данных области сообщения 6 (младшее слово)	0000
MODATAH6H	ECD6	XSFR	Регистр (старший) данных области сообщения 6 (старшее слово)	0000
MOAR6L	ECD8	XSFR	Регистр арбитража области сообщения 6 (младшее слово)	0000
MOAR6H	ECDA	XSFR	Регистр арбитража области сообщения 6 (старшее слово)	0000
MOSTAT6L MOCTR6L	ECDC	XSFR	Регистр состояния области сообщения 6 (младшее слово) (только чтение) Регистр состояния области сообщения 6 (младшее слово) (только запись)	0000
MOSTAT6H MOCTR6H	ECDE	XSFR	Регистр состояния области сообщения 6 (старшее слово) (только чтение) Регистр состояния области сообщения 6 (старшее слово) (только запись)	0705

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR7L	ECE0	XSFR	Регистр управления функционированием области сообщения 7 (младшее слово)	0000
MOFCR7H	ECE2	XSFR	Регистр управления функционированием области сообщения 7 (старшее слово)	0000
MOFGPR7L	ECE4	XSFR	Регистр указателя FIFO/шлюза области сообщения 7 (младшее слово)	0000
MOFGPR7H	ECE6	XSFR	Регистр указателя FIFO/шлюза области сообщения 7 (старшее слово)	0000
MOIPR7L	ECE8	XSFR	Регистр указателя прерываний области сообщения 7 (младшее слово)	0000
MOIPR7H	ECEA	XSFR	Регистр указателя прерываний области сообщения 7 (старшее слово)	0000
MOAMR7L	ECEC	XSFR	Регистр маски области сообщения 7 (младшее слово)	FFFF
MOAMR7H	ECEE	XSFR	Регистр маски области сообщения 7 (старшее слово)	3FFF
MODATAL7L	ECF0	XSFR	Регистр (младший) данных области сообщения 7 (младшее слово)	0000
MODATAL7H	ECF2	XSFR	Регистр (младший) данных области сообщения 7 (старшее слово)	0000
MODATAH7L	ECF4	XSFR	Регистр (старший) данных области сообщения 7 (младшее слово)	0000
MODATAH7H	ECF6	XSFR	Регистр (старший) данных области сообщения 7 (старшее слово)	0000
MOAR7L	ECF8	XSFR	Регистр арбитража области сообщения 7 (младшее слово)	0000
MOAR7H	ECFA	XSFR	Регистр арбитража области сообщения 7 (старшее слово)	0000
MOSTAT7L MOCTR7L	ECFC	XSFR	Регистр состояния области сообщения 7 (младшее слово) (только чтение) Регистр состояния области сообщения 7 (младшее слово) (только запись)	0000
MOSTAT7H MOCTR7H	ECFE	XSFR	Регистр состояния области сообщения 7 (старшее слово) (только чтение) Регистр состояния области сообщения 7 (старшее слово) (только запись)	0806

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR8L	ED00	XSFR	Регистр управления функционированием области сообщения 8 (младшее слово)	0000
MOFCR8H	ED02	XSFR	Регистр управления функционированием области сообщения 8 (старшее слово)	0000
MOFGPR8L	ED04	XSFR	Регистр указателя FIFO/шлюза области сообщения 8 (младшее слово)	0000
MOFGPR8H	ED06	XSFR	Регистр указателя FIFO/шлюза области сообщения 8 (старшее слово)	0000
MOIPR8L	ED08	XSFR	Регистр указателя прерываний области сообщения 8 (младшее слово)	0000
MOIPR8H	ED0A	XSFR	Регистр указателя прерываний области сообщения 8 (старшее слово)	0000
MOAMR8L	ED0C	XSFR	Регистр маски области сообщения 8 (младшее слово)	FFFF
MOAMR8H	ED0E	XSFR	Регистр маски области сообщения 8 (старшее слово)	3FFF
MODATAL8L	ED10	XSFR	Регистр (младший) данных области сообщения 8 (младшее слово)	0000
MODATAL8H	ED12	XSFR	Регистр (младший) данных области сообщения 8 (старшее слово)	0000
MODATAH8L	ED14	XSFR	Регистр (старший) данных области сообщения 8 (младшее слово)	0000
MODATAH8H	ED16	XSFR	Регистр (старший) данных области сообщения 8 (старшее слово)	0000
MOAR8L	ED18	XSFR	Регистр арбитража области сообщения 8 (младшее слово)	0000
MOAR8H	ED1A	XSFR	Регистр арбитража области сообщения 8 (старшее слово)	0000
MOSTAT8L MOCTR8L	ED1C	XSFR	Регистр состояния области сообщения 8 (младшее слово) (только чтение) Регистр состояния области сообщения 8 (младшее слово) (только запись)	0000
MOSTAT8H MOCTR8H	ED1E	XSFR	Регистр состояния области сообщения 8 (старшее слово) (только чтение) Регистр состояния области сообщения 8 (старшее слово) (только запись)	0907

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR9L	ED20	XSFR	Регистр управления функционированием области сообщения 9 (младшее слово)	0000
MOFCR9H	ED22	XSFR	Регистр управления функционированием области сообщения 9 (старшее слово)	0000
MOFGPR9L	ED24	XSFR	Регистр указателя FIFO/шлюза области сообщения 9 (младшее слово)	0000
MOFGPR9H	ED26	XSFR	Регистр указателя FIFO/шлюза области сообщения 9 (старшее слово)	0000
MOIPR9L	ED28	XSFR	Регистр указателя прерываний области сообщения 9 (младшее слово)	0000
MOIPR9H	ED2A	XSFR	Регистр указателя прерываний области сообщения 9 (старшее слово)	0000
MOAMR9L	ED2C	XSFR	Регистр маски области сообщения 9 (младшее слово)	FFFF
MOAMR9H	ED2E	XSFR	Регистр маски области сообщения 9 (старшее слово)	3FFF
MODATAL9L	ED30	XSFR	Регистр (младший) данных области сообщения 9 (младшее слово)	0000
MODATAL9H	ED32	XSFR	Регистр (младший) данных области сообщения 9 (старшее слово)	0000
MODATAH9L	ED34	XSFR	Регистр (старший) данных области сообщения 9 (младшее слово)	0000
MODATAH9H	ED36	XSFR	Регистр (старший) данных области сообщения 9 (старшее слово)	0000
MOAR9L	ED38	XSFR	Регистр арбитража области сообщения 9 (младшее слово)	0000
MOAR9H	ED3A	XSFR	Регистр арбитража области сообщения 9 (старшее слово)	0000
MOSTAT9L MOCTR9L	ED3C	XSFR	Регистр состояния области сообщения 9 (младшее слово) (только чтение) Регистр состояния области сообщения 9 (младшее слово) (только запись)	0000
MOSTAT9H MOCTR9H	ED3E	XSFR	Регистр состояния области сообщения 9 (старшее слово) (только чтение) Регистр состояния области сообщения 9 (старшее слово) (только запись)	0A08

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR10L	ED40	XSFR	Регистр управления функционированием области сообщения 10 (младшее слово)	0000
MOFCR10H	ED42	XSFR	Регистр управления функционированием области сообщения 10 (старшее слово)	0000
MOFGPR10L	ED44	XSFR	Регистр указателя FIFO/шлюза области сообщения 10 (младшее слово)	0000
MOFGPR10H	ED46	XSFR	Регистр указателя FIFO/шлюза области сообщения 10 (старшее слово)	0000
MOIPR10L	ED48	XSFR	Регистр указателя прерываний области сообщения 10 (младшее слово)	0000
MOIPR10H	ED4A	XSFR	Регистр указателя прерываний области сообщения 10 (старшее слово)	0000
MOAMR10L	ED4C	XSFR	Регистр маски области сообщения 10 (младшее слово)	FFFF
MOAMR10H	ED4E	XSFR	Регистр маски области сообщения 10 (старшее слово)	3FFF
MODATAL10L	ED50	XSFR	Регистр (младший) данных области сообщения 10 (младшее слово)	0000
MODATAL10H	ED52	XSFR	Регистр (младший) данных области сообщения 10 (старшее слово)	0000
MODATAH10L	ED54	XSFR	Регистр (старший) данных области сообщения 10 (младшее слово)	0000
MODATAH10H	ED56	XSFR	Регистр (старший) данных области сообщения 10 (старшее слово)	0000
MOAR10L	ED58	XSFR	Регистр арбитража области сообщения 10 (младшее слово)	0000
MOAR10H	ED5A	XSFR	Регистр арбитража области сообщения 10 (старшее слово)	0000
MOSTAT10L MOCTR10L	ED5C	XSFR	Регистр состояния области сообщения 10 (младшее слово) (только чтение) Регистр состояния области сообщения 10 (младшее слово) (только запись)	0000 ----
MOSTAT10H MOCTR10H	ED5E	XSFR	Регистр состояния области сообщения 10 (старшее слово) (только чтение) Регистр состояния области сообщения 10 (старшее слово) (только запись)	0B09

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR11L	ED60	XSFR	Регистр управления функционированием области сообщения 11 (младшее слово)	0000
MOFCR11H	ED62	XSFR	Регистр управления функционированием области сообщения 11 (старшее слово)	0000
MOFGPR11L	ED64	XSFR	Регистр указателя FIFO/шлюза области сообщения 11 (младшее слово)	0000
MOFGPR11H	ED66	XSFR	Регистр указателя FIFO/шлюза области сообщения 11 (старшее слово)	0000
MOIPR11L	ED68	XSFR	Регистр указателя прерываний области сообщения 11 (младшее слово)	0000
MOIPR11H	ED6A	XSFR	Регистр указателя прерываний области сообщения 11 (старшее слово)	0000
MOAMR11L	ED6C	XSFR	Регистр маски области сообщения 11 (младшее слово)	FFFF
MOAMR11H	ED6E	XSFR	Регистр маски области сообщения 11 (старшее слово)	3FFF
MODATAL11L	ED70	XSFR	Регистр (младший) данных области сообщения 11 (младшее слово)	0000
MODATAL11H	ED72	XSFR	Регистр (младший) данных области сообщения 11 (старшее слово)	0000
MODATAH11L	ED74	XSFR	Регистр (старший) данных области сообщения 11 (младшее слово)	0000
MODATAH11H	ED76	XSFR	Регистр (старший) данных области сообщения 11 (старшее слово)	0000
MOAR11L	ED78	XSFR	Регистр арбитража области сообщения 11 (младшее слово)	0000
MOAR11H	ED7A	XSFR	Регистр арбитража области сообщения 11 (старшее слово)	0000
MOSTAT11L MOCTR11L	ED7C	XSFR	Регистр состояния области сообщения 11 (младшее слово) (только чтение) Регистр состояния области сообщения 11 (младшее слово) (только запись)	0000
MOSTAT11H MOCTR11H	ED7E	XSFR	Регистр состояния области сообщения 11 (старшее слово) (только чтение) Регистр состояния области сообщения 11 (старшее слово) (только запись)	0C0A

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR12L	ED80	XSFR	Регистр управления функционированием области сообщения 12 (младшее слово)	0000
MOFCR12H	ED82	XSFR	Регистр управления функционированием области сообщения 12 (старшее слово)	0000
MOFGPR12L	ED84	XSFR	Регистр указателя FIFO/шлюза области сообщения 12 (младшее слово)	0000
MOFGPR12H	ED86	XSFR	Регистр указателя FIFO/шлюза области сообщения 12 (старшее слово)	0000
MOIPR12L	ED88	XSFR	Регистр указателя прерываний области сообщения 12 (младшее слово)	0000
MOIPR12H	ED8A	XSFR	Регистр указателя прерываний области сообщения 12 (старшее слово)	0000
MOAMR12L	ED8C	XSFR	Регистр маски области сообщения 12 (младшее слово)	FFFF
MOAMR12H	ED8E	XSFR	Регистр маски области сообщения 12 (старшее слово)	3FFF
MODATAL12L	ED90	XSFR	Регистр (младший) данных области сообщения 12 (младшее слово)	0000
MODATAL12H	ED92	XSFR	Регистр (младший) данных области сообщения 12 (старшее слово)	0000
MODATAH12L	ED94	XSFR	Регистр (старший) данных области сообщения 12 (младшее слово)	0000
MODATAH12H	ED96	XSFR	Регистр (старший) данных области сообщения 12 (старшее слово)	0000
MOAR12L	ED98	XSFR	Регистр арбитража области сообщения 12 (младшее слово)	0000
MOAR12H	ED9A	XSFR	Регистр арбитража области сообщения 12 (старшее слово)	0000
MOSTAT12L MOCTR12L	ED9C	XSFR	Регистр состояния области сообщения 12 (младшее слово) (только чтение) Регистр состояния области сообщения 12 (младшее слово) (только запись)	0000
MOSTAT12H MOCTR12H	ED9E	XSFR	Регистр состояния области сообщения 12 (старшее слово) (только чтение) Регистр состояния области сообщения 12 (старшее слово) (только запись)	0000 0D0B

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR13L	EDA0	XSFR	Регистр управления функционированием области сообщения 13 (младшее слово)	0000
MOFCR13H	EDA2	XSFR	Регистр управления функционированием области сообщения 13 (старшее слово)	0000
MOFGPR13L	EDA4	XSFR	Регистр указателя FIFO/шлюза области сообщения 13 (младшее слово)	0000
MOFGPR13H	EDA6	XSFR	Регистр указателя FIFO/шлюза области сообщения 13 (старшее слово)	0000
MOIPR13L	EDA8	XSFR	Регистр указателя прерываний области сообщения 13 (младшее слово)	0000
MOIPR13H	EDAA	XSFR	Регистр указателя прерываний области сообщения 13 (старшее слово)	0000
MOAMR13L	EDAC	XSFR	Регистр маски области сообщения 13 (младшее слово)	FFFF
MOAMR13H	EDAE	XSFR	Регистр маски области сообщения 13 (старшее слово)	3FFF
MODATAL13L	EDB0	XSFR	Регистр (младший) данных области сообщения 13 (младшее слово)	0000
MODATAL13H	EDB2	XSFR	Регистр (младший) данных области сообщения 13 (старшее слово)	0000
MODATAH13L	EDB4	XSFR	Регистр (старший) данных области сообщения 13 (младшее слово)	0000
MODATAH13H	EDB6	XSFR	Регистр (старший) данных области сообщения 13 (старшее слово)	0000
MOAR13L	EDB8	XSFR	Регистр арбитража области сообщения 13 (младшее слово)	0000
MOAR13H	EDBA	XSFR	Регистр арбитража области сообщения 13 (старшее слово)	0000
MOSTAT13L MOCTR13L	EDBC	XSFR	Регистр состояния области сообщения 13 (младшее слово) (только чтение) Регистр состояния области сообщения 13 (младшее слово) (только запись)	0000
MOSTAT13H MOCTR13H	EDBE	XSFR	Регистр состояния области сообщения 13 (старшее слово) (только чтение) Регистр состояния области сообщения 13 (старшее слово) (только запись)	0E0C

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR14L	EDC0	XSFR	Регистр управления функционированием области сообщения 14 (младшее слово)	0000
MOFCR14H	EDC2	XSFR	Регистр управления функционированием области сообщения 14 (старшее слово)	0000
MOFGPR14L	EDC4	XSFR	Регистр указателя FIFO/шлюза области сообщения 14 (младшее слово)	0000
MOFGPR14H	EDC6	XSFR	Регистр указателя FIFO/шлюза области сообщения 14 (старшее слово)	0000
MOIPR14L	EDC8	XSFR	Регистр указателя прерываний области сообщения 14 (младшее слово)	0000
MOIPR14H	EDCA	XSFR	Регистр указателя прерываний области сообщения 14 (старшее слово)	0000
MOAMR14L	EDCC	XSFR	Регистр маски области сообщения 14 (младшее слово)	FFFF
MOAMR14H	EDCE	XSFR	Регистр маски области сообщения 14 (старшее слово)	3FFF
MODATAL14L	EDD0	XSFR	Регистр (младший) данных области сообщения 14 (младшее слово)	0000
MODATAL14H	EDD2	XSFR	Регистр (младший) данных области сообщения 14 (старшее слово)	0000
MODATAH14L	EDD4	XSFR	Регистр (старший) данных области сообщения 14 (младшее слово)	0000
MODATAH14H	EDD6	XSFR	Регистр (старший) данных области сообщения 14 (старшее слово)	0000
MOAR14L	EDD8	XSFR	Регистр арбитража области сообщения 14 (младшее слово)	0000
MOAR14H	EDDA	XSFR	Регистр арбитража области сообщения 14 (старшее слово)	0000
MOSTAT14L MOCTR14L	EDDC	XSFR	Регистр состояния области сообщения 14 (младшее слово) (только чтение) Регистр состояния области сообщения 14 (младшее слово) (только запись)	0000
MOSTAT14H MOCTR14H	EDDE	XSFR	Регистр состояния области сообщения 14 (старшее слово) (только чтение) Регистр состояния области сообщения 14 (старшее слово) (только запись)	0F0D

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR15L	EDE0	XSFR	Регистр управления функционированием области сообщения 15 (младшее слово)	0000
MOFCR15H	EDE2	XSFR	Регистр управления функционированием области сообщения 15 (старшее слово)	0000
MOFGPR15L	EDE4	XSFR	Регистр указателя FIFO/шлюза области сообщения 15 (младшее слово)	0000
MOFGPR15H	EDE6	XSFR	Регистр указателя FIFO/шлюза области сообщения 15 (старшее слово)	0000
MOIPR15L	EDE8	XSFR	Регистр указателя прерываний области сообщения 15 (младшее слово)	0000
MOIPR15H	EDEA	XSFR	Регистр указателя прерываний области сообщения 15 (старшее слово)	0000
MOAMR15L	EDEC	XSFR	Регистр маски области сообщения 15 (младшее слово)	FFFF
MOAMR15H	EDEE	XSFR	Регистр маски области сообщения 15 (старшее слово)	3FFF
MODATAL15L	EDF0	XSFR	Регистр (младший) данных области сообщения 15 (младшее слово)	0000
MODATAL15H	EDF2	XSFR	Регистр (младший) данных области сообщения 15 (старшее слово)	0000
MODATAH15L	EDF4	XSFR	Регистр (старший) данных области сообщения 15 (младшее слово)	0000
MODATAH15H	EDF6	XSFR	Регистр (старший) данных области сообщения 15 (старшее слово)	0000
MOAR15L	EDF8	XSFR	Регистр арбитража области сообщения 15 (младшее слово)	0000
MOAR15H	EDFA	XSFR	Регистр арбитража области сообщения 15 (старшее слово)	0000
MOSTAT15L MOCTR15L	EDFC	XSFR	Регистр состояния области сообщения 15 (младшее слово) (только чтение) Регистр состояния области сообщения 15 (младшее слово) (только запись)	0000
MOSTAT15H MOCTR15H	EDFE	XSFR	Регистр состояния области сообщения 15 (старшее слово) (только чтение) Регистр состояния области сообщения 15 (старшее слово) (только запись)	100E

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR16L	EE00	XSFR	Регистр управления функционированием области сообщения 16 (младшее слово)	0000
MOFCR16H	EE02	XSFR	Регистр управления функционированием области сообщения 16 (старшее слово)	0000
MOFGPR16L	EE04	XSFR	Регистр указателя FIFO/шлюза области сообщения 16 (младшее слово)	0000
MOFGPR16H	EE06	XSFR	Регистр указателя FIFO/шлюза области сообщения 16 (старшее слово)	0000
MOIPR16L	EE08	XSFR	Регистр указателя прерываний области сообщения 16 (младшее слово)	0000
MOIPR16H	EE0A	XSFR	Регистр указателя прерываний области сообщения 16 (старшее слово)	0000
MOAMR16L	EE0C	XSFR	Регистр маски области сообщения 16 (младшее слово)	FFFF
MOAMR16H	EE0E	XSFR	Регистр маски области сообщения 16 (старшее слово)	3FFF
MODATAL16L	EE10	XSFR	Регистр (младший) данных области сообщения 16 (младшее слово)	0000
MODATAL16H	EE12	XSFR	Регистр (младший) данных области сообщения 16 (старшее слово)	0000
MODATAH16L	EE14	XSFR	Регистр (старший) данных области сообщения 16 (младшее слово)	0000
MODATAH16H	EE16	XSFR	Регистр (старший) данных области сообщения 16 (старшее слово)	0000
MOAR16L	EE18	XSFR	Регистр арбитража области сообщения 16 (младшее слово)	0000
MOAR16H	EE1A	XSFR	Регистр арбитража области сообщения 16 (старшее слово)	0000
MOSTAT16L MOCTR16L	EE1C	XSFR	Регистр состояния области сообщения 16 (младшее слово) (только чтение) Регистр состояния области сообщения 16 (младшее слово) (только запись)	0000
MOSTAT16H MOCTR16H	EE1E	XSFR	Регистр состояния области сообщения 16 (старшее слово) (только чтение) Регистр состояния области сообщения 16 (старшее слово) (только запись)	111F

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR17L	EE20	XSFR	Регистр управления функционированием области сообщения 17 (младшее слово)	0000
MOFCR17H	EE22	XSFR	Регистр управления функционированием области сообщения 17 (старшее слово)	0000
MOFGPR17L	EE24	XSFR	Регистр указателя FIFO/шлюза области сообщения 17 (младшее слово)	0000
MOFGPR17H	EE26	XSFR	Регистр указателя FIFO/шлюза области сообщения 17 (старшее слово)	0000
MOIPR17L	EE28	XSFR	Регистр указателя прерываний области сообщения 17 (младшее слово)	0000
MOIPR17H	EE2A	XSFR	Регистр указателя прерываний области сообщения 17 (старшее слово)	0000
MOAMR17L	EE2C	XSFR	Регистр маски области сообщения 17 (младшее слово)	FFFF
MOAMR17H	EE2E	XSFR	Регистр маски области сообщения 17 (старшее слово)	3FFF
MODATAL17L	EE30	XSFR	Регистр (младший) данных области сообщения 17 (младшее слово)	0000
MODATAL17H	EE32	XSFR	Регистр (младший) данных области сообщения 17 (старшее слово)	0000
MODATAH17L	EE34	XSFR	Регистр (старший) данных области сообщения 17 (младшее слово)	0000
MODATAH17H	EE36	XSFR	Регистр (старший) данных области сообщения 17 (старшее слово)	0000
MOAR17L	EE38	XSFR	Регистр арбитража области сообщения 17 (младшее слово)	0000
MOAR17H	EE3A	XSFR	Регистр арбитража области сообщения 17 (старшее слово)	0000
MOSTAT17L MOCTR17L	EE3C	XSFR	Регистр состояния области сообщения 17 (младшее слово) (только чтение) Регистр состояния области сообщения 17 (младшее слово) (только запись)	0000
MOSTAT17H MOCTR17H	EE3E	XSFR	Регистр состояния области сообщения 17 (старшее слово) (только чтение) Регистр состояния области сообщения 17 (старшее слово) (только запись)	1210

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR18L	EE40	XSFR	Регистр управления функционированием области сообщения 18 (младшее слово)	0000
MOFCR18H	EE42	XSFR	Регистр управления функционированием области сообщения 18 (старшее слово)	0000
MOFGPR18L	EE44	XSFR	Регистр указателя FIFO/шлюза области сообщения 18 (младшее слово)	0000
MOFGPR18H	EE46	XSFR	Регистр указателя FIFO/шлюза области сообщения 18 (старшее слово)	0000
MOIPR18L	EE48	XSFR	Регистр указателя прерываний области сообщения 18 (младшее слово)	0000
MOIPR18H	EE4A	XSFR	Регистр указателя прерываний области сообщения 18 (старшее слово)	0000
MOAMR18L	EE4C	XSFR	Регистр маски области сообщения 18 (младшее слово)	FFFF
MOAMR18H	EE4E	XSFR	Регистр маски области сообщения 18 (старшее слово)	3FFF
MODATAL18L	EE50	XSFR	Регистр (младший) данных области сообщения 18 (младшее слово)	0000
MODATAL18H	EE52	XSFR	Регистр (младший) данных области сообщения 18 (старшее слово)	0000
MODATAH18L	EE54	XSFR	Регистр (старший) данных области сообщения 18 (младшее слово)	0000
MODATAH18H	EE56	XSFR	Регистр (старший) данных области сообщения 18 (старшее слово)	0000
MOAR18L	EE58	XSFR	Регистр арбитража области сообщения 18 (младшее слово)	0000
MOAR18H	EE5A	XSFR	Регистр арбитража области сообщения 18 (старшее слово)	0000
MOSTAT18L MOCTR18L	EE5C	XSFR	Регистр состояния области сообщения 18 (младшее слово) (только чтение) Регистр состояния области сообщения 18 (младшее слово) (только запись)	0000
MOSTAT18H MOCTR18H	EE5E	XSFR	Регистр состояния области сообщения 18 (старшее слово) (только чтение) Регистр состояния области сообщения 18 (старшее слово) (только запись)	1311

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR19L	EE60	XSFR	Регистр управления функционированием области сообщения 19 (младшее слово)	0000
MOFCR19H	EE62	XSFR	Регистр управления функционированием области сообщения 19 (старшее слово)	0000
MOFGPR19L	EE64	XSFR	Регистр указателя FIFO/шлюза области сообщения 19 (младшее слово)	0000
MOFGPR19H	EE66	XSFR	Регистр указателя FIFO/шлюза области сообщения 19 (старшее слово)	0000
MOIPR19L	EE68	XSFR	Регистр указателя прерываний области сообщения 19 (младшее слово)	0000
MOIPR19H	EE6A	XSFR	Регистр указателя прерываний области сообщения 19 (старшее слово)	0000
MOAMR19L	EE6C	XSFR	Регистр маски области сообщения 19 (младшее слово)	FFFF
MOAMR19H	EE6E	XSFR	Регистр маски области сообщения 19 (старшее слово)	3FFF
MODATAL19L	EE70	XSFR	Регистр (младший) данных области сообщения 19 (младшее слово)	0000
MODATAL19H	EE72	XSFR	Регистр (младший) данных области сообщения 19 (старшее слово)	0000
MODATAH19L	EE74	XSFR	Регистр (старший) данных области сообщения 19 (младшее слово)	0000
MODATAH19H	EE76	XSFR	Регистр (старший) данных области сообщения 19 (старшее слово)	0000
MOAR19L	EE78	XSFR	Регистр арбитража области сообщения 19 (младшее слово)	0000
MOAR19H	EE7A	XSFR	Регистр арбитража области сообщения 19 (старшее слово)	0000
MOSTAT19L MOCTR19L	EE7C	XSFR	Регистр состояния области сообщения 19 (младшее слово) (только чтение) Регистр состояния области сообщения 19 (младшее слово) (только запись)	0000
MOSTAT19H MOCTR19H	EE7E	XSFR	Регистр состояния области сообщения 19 (старшее слово) (только чтение) Регистр состояния области сообщения 19 (старшее слово) (только запись)	1412

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR20L	EE80	XSFR	Регистр управления функционированием области сообщения 20 (младшее слово)	0000
MOFCR20H	EE82	XSFR	Регистр управления функционированием области сообщения 20 (старшее слово)	0000
MOFGPR20L	EE84	XSFR	Регистр указателя FIFO/шлюза области сообщения 20 (младшее слово)	0000
MOFGPR20H	EE86	XSFR	Регистр указателя FIFO/шлюза области сообщения 20 (старшее слово)	0000
MOIPR20L	EE88	XSFR	Регистр указателя прерываний области сообщения 20 (младшее слово)	0000
MOIPR20H	EE8A	XSFR	Регистр указателя прерываний области сообщения 20 (старшее слово)	0000
MOAMR20L	EE8C	XSFR	Регистр маски области сообщения 20 (младшее слово)	FFFF
MOAMR20H	EE8E	XSFR	Регистр маски области сообщения 20 (старшее слово)	3FFF
MODATAL20L	EE90	XSFR	Регистр (младший) данных области сообщения 20 (младшее слово)	0000
MODATAL20H	EE92	XSFR	Регистр (младший) данных области сообщения 20 (старшее слово)	0000
MODATAH20L	EE94	XSFR	Регистр (старший) данных области сообщения 20 (младшее слово)	0000
MODATAH20H	EE96	XSFR	Регистр (старший) данных области сообщения 20 (старшее слово)	0000
MOAR20L	EE98	XSFR	Регистр арбитража области сообщения 20 (младшее слово)	0000
MOAR20H	EE9A	XSFR	Регистр арбитража области сообщения 20 (старшее слово)	0000
MOSTAT20L MOCTR20L	EE9C	XSFR	Регистр состояния области сообщения 20 (младшее слово) (только чтение) Регистр состояния области сообщения 20 (младшее слово) (только запись)	0000 –
MOSTAT20H MOCTR20H	EE9E	XSFR	Регистр состояния области сообщения 20 (старшее слово) (только чтение) Регистр состояния области сообщения 20 (старшее слово) (только запись)	1513

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR21L	EEA0	XSFR	Регистр управления функционированием области сообщения 21 (младшее слово)	0000
MOFCR21H	EEA2	XSFR	Регистр управления функционированием области сообщения 21 (старшее слово)	0000
MOFGPR21L	EEA4	XSFR	Регистр указателя FIFO/шлюза области сообщения 21 (младшее слово)	0000
MOFGPR21H	EEA6	XSFR	Регистр указателя FIFO/шлюза области сообщения 21 (старшее слово)	0000
MOIPR21L	EEA8	XSFR	Регистр указателя прерываний области сообщения 21 (младшее слово)	0000
MOIPR21H	EEAA	XSFR	Регистр указателя прерываний области сообщения 21 (старшее слово)	0000
MOAMR21L	EEAC	XSFR	Регистр маски области сообщения 21 (младшее слово)	FFFF
MOAMR21H	EEAE	XSFR	Регистр маски области сообщения 21 (старшее слово)	3FFF
MODATAL21L	EEB0	XSFR	Регистр (младший) данных области сообщения 21 (младшее слово)	0000
MODATAL21H	EEB2	XSFR	Регистр (младший) данных области сообщения 21 (старшее слово)	0000
MODATAH21L	EEB4	XSFR	Регистр (старший) данных области сообщения 21 (младшее слово)	0000
MODATAH21H	EEB6	XSFR	Регистр (старший) данных области сообщения 21 (старшее слово)	0000
MOAR21L	EEB8	XSFR	Регистр арбитража области сообщения 21 (младшее слово)	0000
MOAR21H	EEBA	XSFR	Регистр арбитража области сообщения 21 (старшее слово)	0000
MOSTAT21L MOCTR21L	EEBC	XSFR	Регистр состояния области сообщения 21 (младшее слово) (только чтение) Регистр состояния области сообщения 21 (младшее слово) (только запись)	0000
MOSTAT21H MOCTR21H	EEBE	XSFR	Регистр состояния области сообщения 21 (старшее слово) (только чтение) Регистр состояния области сообщения 21 (старшее слово) (только запись)	1614

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR22L	EEC0	XSFR	Регистр управления функционированием области сообщения 22 (младшее слово)	0000
MOFCR22H	EEC2	XSFR	Регистр управления функционированием области сообщения 22 (старшее слово)	0000
MOFGPR22L	EEC4	XSFR	Регистр указателя FIFO/шлюза области сообщения 22 (младшее слово)	0000
MOFGPR22H	EEC6	XSFR	Регистр указателя FIFO/шлюза области сообщения 22 (старшее слово)	0000
MOIPR22L	EEC8	XSFR	Регистр указателя прерываний области сообщения 22 (младшее слово)	0000
MOIPR22H	EECA	XSFR	Регистр указателя прерываний области сообщения 22 (старшее слово)	0000
MOAMR22L	EECC	XSFR	Регистр маски области сообщения 22 (младшее слово)	FFFF
MOAMR22H	EECE	XSFR	Регистр маски области сообщения 22 (старшее слово)	3FFF
MODATAL22L	EED0	XSFR	Регистр (младший) данных области сообщения 22 (младшее слово)	0000
MODATAL22H	EED2	XSFR	Регистр (младший) данных области сообщения 22 (старшее слово)	0000
MODATAH22L	EED4	XSFR	Регистр (старший) данных области сообщения 22 (младшее слово)	0000
MODATAH22H	EED6	XSFR	Регистр (старший) данных области сообщения 22 (старшее слово)	0000
MOAR22L	EED8	XSFR	Регистр арбитража области сообщения 22 (младшее слово)	0000
MOAR22H	EEDA	XSFR	Регистр арбитража области сообщения 22 (старшее слово)	0000
MOSTAT22L MOCTR22L	EEDC	XSFR	Регистр состояния области сообщения 22 (младшее слово) (только чтение) Регистр состояния области сообщения 22 (младшее слово) (только запись)	0000
MOSTAT22H MOCTR22H	EEDE	XSFR	Регистр состояния области сообщения 22 (старшее слово) (только чтение) Регистр состояния области сообщения 22 (старшее слово) (только запись)	1715

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR23L	EEE0	XSFR	Регистр управления функционированием области сообщения 23 (младшее слово)	0000
MOFCR23H	EEE2	XSFR	Регистр управления функционированием области сообщения 23 (старшее слово)	0000
MOFGPR23L	EEE4	XSFR	Регистр указателя FIFO/шлюза области сообщения 23 (младшее слово)	0000
MOFGPR23H	EEE6	XSFR	Регистр указателя FIFO/шлюза области сообщения 23 (старшее слово)	0000
MOIPR23L	EEE8	XSFR	Регистр указателя прерываний области сообщения 23 (младшее слово)	0000
MOIPR23H	EEEA	XSFR	Регистр указателя прерываний области сообщения 23 (старшее слово)	0000
MOAMR23L	EEEC	XSFR	Регистр маски области сообщения 23 (младшее слово)	FFFF
MOAMR23H	EEEE	XSFR	Регистр маски области сообщения 23 (старшее слово)	3FFF
MODATAL23L	EEF0	XSFR	Регистр (младший) данных области сообщения 23 (младшее слово)	0000
MODATAL23H	EEF2	XSFR	Регистр (младший) данных области сообщения 23 (старшее слово)	0000
MODATAH23L	EEF4	XSFR	Регистр (старший) данных области сообщения 23 (младшее слово)	0000
MODATAH23H	EEF6	XSFR	Регистр (старший) данных области сообщения 23 (старшее слово)	0000
MOAR23L	EEF8	XSFR	Регистр арбитража области сообщения 23 (младшее слово)	0000
MOAR23H	EEFA	XSFR	Регистр арбитража области сообщения 23 (старшее слово)	0000
MOSTAT23L MOCTR23L	EEFC	XSFR	Регистр состояния области сообщения 23 (младшее слово) (только чтение) Регистр состояния области сообщения 23 (младшее слово) (только запись)	0000
MOSTAT23H MOCTR23H	EEFE	XSFR	Регистр состояния области сообщения 23 (старшее слово) (только чтение) Регистр состояния области сообщения 23 (старшее слово) (только запись)	1816

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR24L	EF00	XSFR	Регистр управления функционированием области сообщения 24 (младшее слово)	0000
MOFCR24H	EF02	XSFR	Регистр управления функционированием области сообщения 24 (старшее слово)	0000
MOFGPR24L	EF04	XSFR	Регистр указателя FIFO/шлюза области сообщения 24 (младшее слово)	0000
MOFGPR24H	EF06	XSFR	Регистр указателя FIFO/шлюза области сообщения 24 (старшее слово)	0000
MOIPR24L	EF08	XSFR	Регистр указателя прерываний области сообщения 24 (младшее слово)	0000
MOIPR24H	EF0A	XSFR	Регистр указателя прерываний области сообщения 24 (старшее слово)	0000
MOAMR24L	EF0C	XSFR	Регистр маски области сообщения 24 (младшее слово)	FFFF
MOAMR24H	EF0E	XSFR	Регистр маски области сообщения 24 (старшее слово)	3FFF
MODATAL24L	EF10	XSFR	Регистр (младший) данных области сообщения 24 (младшее слово)	0000
MODATAL24H	EF12	XSFR	Регистр (младший) данных области сообщения 24 (старшее слово)	0000
MODATAH24L	EF14	XSFR	Регистр (старший) данных области сообщения 24 (младшее слово)	0000
MODATAH24H	EF16	XSFR	Регистр (старший) данных области сообщения 24 (старшее слово)	0000
MOAR24L	EF18	XSFR	Регистр арбитража области сообщения 24 (младшее слово)	0000
MOAR24H	EF1A	XSFR	Регистр арбитража области сообщения 24 (старшее слово)	0000
MOSTAT24L MOCTR24L	EF1C	XSFR	Регистр состояния области сообщения 24 (младшее слово) (только чтение) Регистр состояния области сообщения 24 (младшее слово) (только запись)	0000
MOSTAT24H MOCTR24H	EF1E	XSFR	Регистр состояния области сообщения 24 (старшее слово) (только чтение) Регистр состояния области сообщения 24 (старшее слово) (только запись)	1917

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR25L	EF20	XSFR	Регистр управления функционированием области сообщения 25 (младшее слово)	0000
MOFCR25H	EF22	XSFR	Регистр управления функционированием области сообщения 25 (старшее слово)	0000
MOFGPR25L	EF24	XSFR	Регистр указателя FIFO/шлюза области сообщения 25 (младшее слово)	0000
MOFGPR25H	EF26	XSFR	Регистр указателя FIFO/шлюза области сообщения 25 (старшее слово)	0000
MOIPR25L	EF28	XSFR	Регистр указателя прерываний области сообщения 25 (младшее слово)	0000
MOIPR25H	EF2A	XSFR	Регистр указателя прерываний области сообщения 25 (старшее слово)	0000
MOAMR25L	EF2C	XSFR	Регистр маски области сообщения 25 (младшее слово)	FFFF
MOAMR25H	EF2E	XSFR	Регистр маски области сообщения 25 (старшее слово)	3FFF
MODATAL25L	EF30	XSFR	Регистр (младший) данных области сообщения 25 (младшее слово)	0000
MODATAL25H	EF32	XSFR	Регистр (младший) данных области сообщения 25 (старшее слово)	0000
MODATAH25L	EF34	XSFR	Регистр (старший) данных области сообщения 25 (младшее слово)	0000
MODATAH25H	EF36	XSFR	Регистр (старший) данных области сообщения 25 (старшее слово)	0000
MOAR25L	EF38	XSFR	Регистр арбитража области сообщения 25 (младшее слово)	0000
MOAR25H	EF3A	XSFR	Регистр арбитража области сообщения 25 (старшее слово)	0000
MOSTAT25L MOCTR25L	EF3C	XSFR	Регистр состояния области сообщения 25 (младшее слово) (только чтение) Регистр состояния области сообщения 25 (младшее слово) (только запись)	0000
MOSTAT25H MOCTR25H	EF3E	XSFR	Регистр состояния области сообщения 25 (старшее слово) (только чтение) Регистр состояния области сообщения 25 (старшее слово) (только запись)	1A18

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR26L	EF40	XSFR	Регистр управления функционированием области сообщения 26 (младшее слово)	0000
MOFCR26H	EF42	XSFR	Регистр управления функционированием области сообщения 26 (старшее слово)	0000
MOFGPR26L	EF44	XSFR	Регистр указателя FIFO/шлюза области сообщения 26 (младшее слово)	0000
MOFGPR26H	EF46	XSFR	Регистр указателя FIFO/шлюза области сообщения 26 (старшее слово)	0000
MOIPR26L	EF48	XSFR	Регистр указателя прерываний области сообщения 26 (младшее слово)	0000
MOIPR26H	EF4A	XSFR	Регистр указателя прерываний области сообщения 26 (старшее слово)	0000
MOAMR26L	EF4C	XSFR	Регистр маски области сообщения 26 (младшее слово)	FFFF
MOAMR26H	EF4E	XSFR	Регистр маски области сообщения 26 (старшее слово)	3FFF
MODATAL26L	EF50	XSFR	Регистр (младший) данных области сообщения 26 (младшее слово)	0000
MODATAL26H	EF52	XSFR	Регистр (младший) данных области сообщения 26 (старшее слово)	0000
MODATAH26L	EF54	XSFR	Регистр (старший) данных области сообщения 26 (младшее слово)	0000
MODATAH26H	EF56	XSFR	Регистр (старший) данных области сообщения 26 (старшее слово)	0000
MOAR26L	EF58	XSFR	Регистр арбитража области сообщения 26 (младшее слово)	0000
MOAR26H	EF5A	XSFR	Регистр арбитража области сообщения 26 (старшее слово)	0000
MOSTAT26L MOCTR26L	EF5C	XSFR	Регистр состояния области сообщения 26 (младшее слово) (только чтение) Регистр состояния области сообщения 26 (младшее слово) (только запись)	0000
MOSTAT26H MOCTR26H	EF5E	XSFR	Регистр состояния области сообщения 26 (старшее слово) (только чтение) Регистр состояния области сообщения 26 (старшее слово) (только запись)	1B19

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR27L	EF60	XSFR	Регистр управления функционированием области сообщения 27 (младшее слово)	0000
MOFCR27H	EF62	XSFR	Регистр управления функционированием области сообщения 27 (старшее слово)	0000
MOFGPR27L	EF64	XSFR	Регистр указателя FIFO/шлюза области сообщения 27 (младшее слово)	0000
MOFGPR27H	EF66	XSFR	Регистр указателя FIFO/шлюза области сообщения 27 (старшее слово)	0000
MOIPR27L	EF68	XSFR	Регистр указателя прерываний области сообщения 27 (младшее слово)	0000
MOIPR27H	EF6A	XSFR	Регистр указателя прерываний области сообщения 27 (старшее слово)	0000
MOAMR27L	EF6C	XSFR	Регистр маски области сообщения 27 (младшее слово)	FFFF
MOAMR27H	EF6E	XSFR	Регистр маски области сообщения 27 (старшее слово)	3FFF
MODATAL27L	EF70	XSFR	Регистр (младший) данных области сообщения 27 (младшее слово)	0000
MODATAL27H	EF72	XSFR	Регистр (младший) данных области сообщения 27 (старшее слово)	0000
MODATAH27L	EF74	XSFR	Регистр (старший) данных области сообщения 27 (младшее слово)	0000
MODATAH27H	EF76	XSFR	Регистр (старший) данных области сообщения 27 (старшее слово)	0000
MOAR27L	EF78	XSFR	Регистр арбитража области сообщения 27 (младшее слово)	0000
MOAR27H	EF7A	XSFR	Регистр арбитража области сообщения 27 (старшее слово)	0000
MOSTAT27L MOCTR27L	EF7C	XSFR	Регистр состояния области сообщения 27 (младшее слово) (только чтение) Регистр состояния области сообщения 27 (младшее слово) (только запись)	0000
MOSTAT27H MOCTR27H	EF7E	XSFR	Регистр состояния области сообщения 27 (старшее слово) (только чтение) Регистр состояния области сообщения 27 (старшее слово) (только запись)	1C1A

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR28L	EF80	XSFR	Регистр управления функционированием области сообщения 28 (младшее слово)	0000
MOFCR28H	EF82	XSFR	Регистр управления функционированием области сообщения 28 (старшее слово)	0000
MOFGPR28L	EF84	XSFR	Регистр указателя FIFO/шлюза области сообщения 28 (младшее слово)	0000
MOFGPR28H	EF86	XSFR	Регистр указателя FIFO/шлюза области сообщения 28 (старшее слово)	0000
MOIPR28L	EF88	XSFR	Регистр указателя прерываний области сообщения 28 (младшее слово)	0000
MOIPR28H	EF8A	XSFR	Регистр указателя прерываний области сообщения 28 (старшее слово)	0000
MOAMR28L	EF8C	XSFR	Регистр маски области сообщения 28 (младшее слово)	FFFF
MOAMR28H	EF8E	XSFR	Регистр маски области сообщения 28 (старшее слово)	3FFF
MODATAL28L	EF90	XSFR	Регистр (младший) данных области сообщения 28 (младшее слово)	0000
MODATAL28H	EF92	XSFR	Регистр (младший) данных области сообщения 28 (старшее слово)	0000
MODATAH28L	EF94	XSFR	Регистр (старший) данных области сообщения 28 (младшее слово)	0000
MODATAH28H	EF96	XSFR	Регистр (старший) данных области сообщения 28 (старшее слово)	0000
MOAR28L	EF98	XSFR	Регистр арбитража области сообщения 28 (младшее слово)	0000
MOAR28H	EF9A	XSFR	Регистр арбитража области сообщения 28 (старшее слово)	0000
MOSTAT28L MOCTR28L	EF9C	XSFR	Регистр состояния области сообщения 28 (младшее слово) (только чтение) Регистр состояния области сообщения 28 (младшее слово) (только запись)	0000
MOSTAT28H MOCTR28H	EF9E	XSFR	Регистр состояния области сообщения 28 (старшее слово) (только чтение) Регистр состояния области сообщения 28 (старшее слово) (только запись)	1D1B

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR29L	EFA0	XSFR	Регистр управления функционированием области сообщения 29 (младшее слово)	0000
MOFCR29H	EFA2	XSFR	Регистр управления функционированием области сообщения 29 (старшее слово)	0000
MOFGPR29L	EFA4	XSFR	Регистр указателя FIFO/шлюза области сообщения 29 (младшее слово)	0000
MOFGPR29H	EFA6	XSFR	Регистр указателя FIFO/шлюза области сообщения 29 (старшее слово)	0000
MOIPR29L	EFA8	XSFR	Регистр указателя прерываний области сообщения 29 (младшее слово)	0000
MOIPR29H	EFAA	XSFR	Регистр указателя прерываний области сообщения 29 (старшее слово)	0000
MOAMR29L	EFAC	XSFR	Регистр маски области сообщения 29 (младшее слово)	FFFF
MOAMR29H	EFAE	XSFR	Регистр маски области сообщения 29 (старшее слово)	3FFF
MODATAL29L	EFB0	XSFR	Регистр (младший) данных области сообщения 29 (младшее слово)	0000
MODATAL29H	EFB2	XSFR	Регистр (младший) данных области сообщения 29 (старшее слово)	0000
MODATAH29L	EFB4	XSFR	Регистр (старший) данных области сообщения 29 (младшее слово)	0000
MODATAH29H	EFB6	XSFR	Регистр (старший) данных области сообщения 29 (старшее слово)	0000
MOAR29L	EFB8	XSFR	Регистр арбитража области сообщения 29 (младшее слово)	0000
MOAR29H	EFBA	XSFR	Регистр арбитража области сообщения 29 (старшее слово)	0000
MOSTAT29L MOCTR29L	EFBC	XSFR	Регистр состояния области сообщения 29 (младшее слово) (только чтение) Регистр состояния области сообщения 29 (младшее слово) (только запись)	0000
MOSTAT29H MOCTR29H	EFBE	XSFR	Регистр состояния области сообщения 29 (старшее слово) (только чтение) Регистр состояния области сообщения 29 (старшее слово) (только запись)	1E1C

Продолжение таблицы Ж.1

1	2	3	4	5
MOFCR30L	EFC0	XSFR	Регистр управления функционированием области сообщения 30 (младшее слово)	0000
MOFCR30H	EFC2	XSFR	Регистр управления функционированием области сообщения 30 (старшее слово)	0000
MOFGPR30L	EFC4	XSFR	Регистр указателя FIFO/шлюза области сообщения 30 (младшее слово)	0000
MOFGPR30H	EFC6	XSFR	Регистр указателя FIFO/шлюза области сообщения 30 (старшее слово)	0000
MOIPR30L	EFC8	XSFR	Регистр указателя прерываний области сообщения 30 (младшее слово)	0000
MOIPR30H	EFCA	XSFR	Регистр указателя прерываний области сообщения 30 (старшее слово)	0000
MOAMR30L	EFCC	XSFR	Регистр маски области сообщения 30 (младшее слово)	FFFF
MOAMR30H	EFCE	XSFR	Регистр маски области сообщения 30 (старшее слово)	3FFF
MODATAL30L	EFD0	XSFR	Регистр (младший) данных области сообщения 30 (младшее слово)	0000
MODATAL30H	EFD2	XSFR	Регистр (младший) данных области сообщения 30 (старшее слово)	0000
MODATAH30L	EFD4	XSFR	Регистр (старший) данных области сообщения 30 (младшее слово)	0000
MODATAH30H	EFD6	XSFR	Регистр (старший) данных области сообщения 30 (старшее слово)	0000
MOAR30L	EFD8	XSFR	Регистр арбитража области сообщения 30 (младшее слово)	0000
MOAR30H	EFDA	XSFR	Регистр арбитража области сообщения 30 (старшее слово)	0000
MOSTAT30L MOCTR30L	EFDC	XSFR	Регистр состояния области сообщения 30 (младшее слово) (только чтение) Регистр состояния области сообщения 30 (младшее слово) (только запись)	0000
MOSTAT30H MOCTR30H	EFDE	XSFR	Регистр состояния области сообщения 30 (старшее слово) (только чтение) Регистр состояния области сообщения 30 (старшее слово) (только запись)	1F1D

Окончание таблицы Ж.1

1	2	3	4	5
MOFCR31L	EFE0	XSFR	Регистр управления функционированием области сообщения 31 (младшее слово)	0000
MOFCR31H	EFE2	XSFR	Регистр управления функционированием области сообщения 31 (старшее слово)	0000
MOFGPR31L	EFE4	XSFR	Регистр указателя FIFO/шлюза области сообщения 31 (младшее слово)	0000
MOFGPR31H	EFE6	XSFR	Регистр указателя FIFO/шлюза области сообщения 31 (старшее слово)	0000
MOIPR31L	EFE8	XSFR	Регистр указателя прерываний области сообщения 31 (младшее слово)	0000
MOIPR31H	EFEA	XSFR	Регистр указателя прерываний области сообщения 31 (старшее слово)	0000
MOAMR31L	EFEC	XSFR	Регистр маски области сообщения 31 (младшее слово)	FFFF
MOAMR31H	EFEE	XSFR	Регистр маски области сообщения 31 (старшее слово)	3FFF
MODATAL31L	EFF0	XSFR	Регистр (младший) данных области сообщения 31 (младшее слово)	0000
MODATAL31H	EFF2	XSFR	Регистр (младший) данных области сообщения 31 (старшее слово)	0000
MODATAH31L	EFF4	XSFR	Регистр (старший) данных области сообщения 31 (младшее слово)	0000
MODATAH31H	EFF6	XSFR	Регистр (старший) данных области сообщения 31 (старшее слово)	0000
MOAR31L	EFF8	XSFR	Регистр арбитража области сообщения 31 (младшее слово)	0000
MOAR31H	EFFA	XSFR	Регистр арбитража области сообщения 31 (старшее слово)	0000
MOSTAT31L MOCTR31L	EFFC	XSFR	Регистр состояния области сообщения 31 (младшее слово) (только чтение) Регистр состояния области сообщения 31 (младшее слово) (только запись)	0000
MOSTAT31H MOCTR31H	EFFE	XSFR	Регистр состояния области сообщения 31 (старшее слово) (только чтение) Регистр состояния области сообщения 31 (старшее слово) (только запись)	201E

Таблица Ж.2 – Регистры блока захвата/сравнения CAPCOM6

Название	Физи-ческий адрес (16 бит), hex	Тип	Описание	Значение после сброса (RESET), hex
CCU6_T12	E890	XSFR	Регистр счета таймера T12	0000
CCU6_T12PR	E892	XSFR	Регистр периода таймера T12	0000
CCU6_T12DTC	E894	XSFR	Регистр контроля задержки «dead-time» T12	0000
CCU6_CC60R	E898	XSFR	Регистр захвата/сравнения канала 0	0000
CCU6_CC61R	E89A	XSFR	Регистр захвата/сравнения канала 1	0000
CCU6_CC62R	E89C	XSFR	Регистр захвата/сравнения канала 2	0000
CCU6_CC60SR	E8A0	XSFR	Теневой регистр захвата/сравнения канала 0	0000
CCU6_ CC61SR	E8A2	XSFR	Теневой регистр захвата/сравнения канала 1	0000
CCU6_ CC62SR	E8A4	XSFR	Теневой регистр захвата/сравнения канала 2	0000
CCU6_ TCTR4	E8A6	XSFR	Регистр управления таймерами 4	0000
CCU6_CMPSTAT	E8A8	XSFR	Регистр состояния захвата/сравнения	0000
CCU6_CMPMODIF	E8AA	XSFR	Регистр изменения состояния захвата/сравнения	0000
CCU6_TCTR0	E8AC	XSFR	Регистр управления таймерами 0	0000
CCU6_TCTR2	E8AE	XSFR	Регистр управления таймерами 2	0000
CCU6_T13	E8B0	XSFR	Регистр счета таймера T13	0000
CCU6_T13PR	E8B2	XSFR	Регистр периода таймера T13	0000
CCU6_CC63R	E8B4	XSFR	Регистр захвата/сравнения канала 3	0000
CCU6_CC63SR	E8B6	XSFR	Теневой регистр захвата/сравнения 3	0000
CCU6_MODCTR	E8C0	XSFR	Регистр управления модуляцией	0000
CCU6_TRPCTR	E8C2	XSFR	Регистр управления ловушками	0000
CCU6_PSLR	E8C4	XSFR	Регистр уровня пассивного состояния	0000
CCU6_T12MSEL	E8C6	XSFR	Регистр выбора режима захвата/сравнения T12	0000
CCU6_MCMOUTS	E8CA	XSFR	Теневой регистр управления выходами в мультиканальном режиме	0000
CCU6_MCMOUT	E8CC	XSFR	Регистр управления выходами в мультиканальном режиме	0000
CCU6_MCMCTR	E8CE	XSFR	Регистр управления мультиканальным режимом	0000
CCU6_IS	E8D0	XSFR	Регистр состояния прерываний	0000
CCU6_ISS	E8D2	XSFR	Регистр установки состояния прерываний	0000
CCU6_ISR	E8D4	XSFR	Регистр сброса состояния прерываний	0000
CCU6_INP	E8D6	XSFR	Регистр указателя узла прерываний	3940
CCU6_IEN	E8D8	XSFR	Регистр разрешения прерываний	0000

Приложение И
(обязательное)

Стартовая конфигурация ИС 1887ВЕ3Т для внешнего RESET

Для конфигурирования микроконтроллера с внешним RESET (EA# = 0) используются выводы порта P0 и выводы ALE, RD# и EA#, состояние которых должно быть установлено во время активного уровня сигнала сброса RSTIN#. После появления положительного фронта сигнала RSTIN#, необходимо удерживать комбинацию сигналов на выводах порта P0 не менее 20 периодов XTAL1, чтобы конфигурация прошла успешно.

На рисунке И.1 представлены выводы порта P0 и регистры, состояние которых они изменяют.

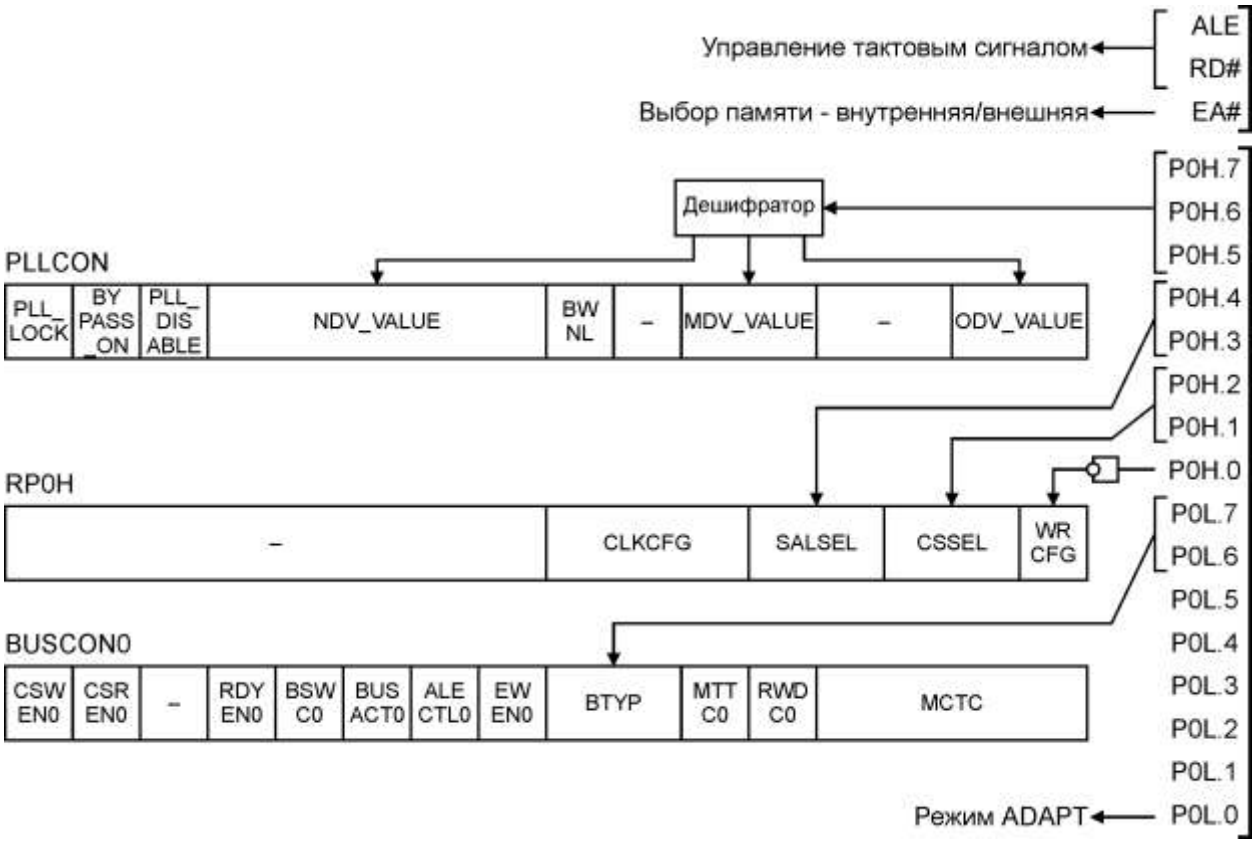


Рисунок И.1 – Порт P0 и его взаимодействие с регистрами микроконтроллера при конфигурировании

В таблице И.1 указано назначение выводов порта P0 и выводов ALE, RD# и EA# при конфигурировании микроконтроллера.

Таблица И.1 – Функциональное назначение выводов микроконтроллера при конфигурировании

<table>
<tr><th>Обозначение вывода</th><th>Функциональное назначение выводов при конфигурации микроконтроллера</th></tr>
<tr><td>1</td><td>2</td></tr>
<tr><td>P0H.7,
P0H.6,
P0H.5</td><td>Устанавливают коэффициент деления/умножения входной частоты.
В зависимости от комбинации сигналов на выводах P0H.7 – P0H.5 в соответствующие битовые поля регистра PLLCON записывается коэффициент деления/умножения входной частоты.
<table>
<tr><th colspan="3">Состояние выводов порта P0</th><th colspan="3">Коэффициенты</th><th rowspan="2">Частота на выходе</th></tr>
<tr><th>P0H.7</th><th>P0H.6</th><th>P0H.5</th><th>ODV_ VALUE</th><th>MDV_ VALUE</th><th>NDV_ VALUE</th></tr>
<tr><td>1</td><td>1</td><td>1</td><td>01_B</td><td>00_B</td><td>01011_B</td><td>$f_{OSC} \times 3$</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>01_B</td><td>00_B</td><td>10001_B</td><td>$f_{OSC} \times 4,5$</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>01_B</td><td>01_B</td><td>01111_B</td><td>$f_{OSC} \times 2$</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>01_B</td><td>00_B</td><td>10011_B</td><td>$f_{OSC} \times 5$</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>01_B</td><td>01_B</td><td>00111_B</td><td>$f_{OSC} \times 1$</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>01_B</td><td>01_B</td><td>10011_B</td><td>$f_{OSC} \times 2,5$</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>00_B</td><td>10_B</td><td>10011_B</td><td>$f_{OSC} \times 2,5$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>01_B</td><td>10_B</td><td>00111_B</td><td>$f_{OSC} / 2$</td></tr>
</table></td></tr>
<tr><td>P0H.4,
P0H.3</td><td>Устанавливают количество разрядов старшей части адреса, используемых для адресации по внешней шине.
С выводов P0H.4, P0H.3 считывается код, определяющий разрядность.
Код записывается в битовое поле SALSEL регистра RP0H.
<table>
<tr><th colspan="2">SALSEL</th><th>Количество разрядов</th><th>Сегментная часть адреса</th><th>Адресное пространство</th></tr>
<tr><td>0</td><td>0</td><td>4</td><td>A19, A18, A17, A16</td><td>1 Мбайт</td></tr>
<tr><td>0</td><td>1</td><td></td><td></td><td>64 Кбайт</td></tr>
<tr><td>1</td><td>0</td><td>8</td><td>A23, A22, A21, A20, A19, A18, A17, A16</td><td>16 Мбайт</td></tr>
<tr><td>1</td><td>1</td><td>2</td><td>A17, A16</td><td>256 Кбайт</td></tr>
</table></td></tr>
<tr><td>P0H.2,
P0H.1</td><td>Устанавливают количество используемых каналов CS#.
С выводов P0H.2, P0H.1 считывается код, определяющий количество каналов. Код записывается в битовое поле CSSEL регистра RP0H.
<table>
<tr><th colspan="2">CSSEL</th><th>Количество каналов</th><th>Действующие каналы порта P6</th><th>Свободные каналы порта P6</th></tr>
<tr><td>0</td><td>0</td><td>3</td><td>CS0#, CS1#, CS2#</td><td>3, 4, 5, 6, 7</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>CS0#, CS1#</td><td>2, 3, 4, 5, 6, 7</td></tr>
<tr><td>1</td><td>0</td><td>–</td><td>–</td><td>Все</td></tr>
<tr><td>1</td><td>1</td><td>5</td><td>CS0#, CS1#, CS2#, CS3#, CS4#</td><td>5, 6, 7
(без pull-down резисторов)</td></tr>
</table></td></tr>
</table>

Окончание таблицы И.1

<table>
<tr><th>1</th><th>2</th></tr>
<tr><td>P0H.0</td><td>Конфигурирует работу сигнала записи.
Значение, считанное с вывода P0H.0, записывается в инверсном виде в бит WRCFG регистра RP0H.</td></tr>
<tr><td>P0L.7,
P0L.6</td><td>Конфигурируют внешнюю шину.
С выводов P0L.7, P0L.6 считывается код, определяющий разрядность и тип шины. Код записывается в битовое поле BUSTYP регистра BUSCON0.
<table>
<tr><th colspan="2">BUSTYP</th><th>Разрядность и тип шины</th></tr>
<tr><td>0</td><td>0</td><td>8-разрядная немультиплексная шина</td></tr>
<tr><td>0</td><td>1</td><td>8-разрядная мультиплексная шина</td></tr>
<tr><td>1</td><td>0</td><td>16-разрядная немультиплексная шина</td></tr>
<tr><td>1</td><td>1</td><td>Не используется</td></tr>
</table></td></tr>
<tr><td>P0L.5 – P0L.1</td><td>Не участвуют в конфигурации.</td></tr>
<tr><td>P0L.0</td><td>Выбирает режим электрической изоляции ADAPT.
Считывание «0» с вывода P0L.0 включает режим ADAPT.
Считывание «1» с вывода P0L.0 выключает режим ADAPT.</td></tr>
<tr><td>ALE,
RD#</td><td>Управляют тактовым сигналом микроконтроллера
<table>
<tr><th>ALE</th><th>RD#</th><th>Режим работы микроконтроллера</th></tr>
<tr><td>0</td><td>0</td><td>без PLL</td></tr>
<tr><td>1</td><td>1</td><td>с PLL</td></tr>
</table></td></tr>
<tr><td>EA#</td><td>Выбирает режим работы микроконтроллера с памятью.
Считывание «0» с вывода EA# включает режим работы с внешней памятью.
Считывание «1» с вывода EA# включает режим работы с внутренней памятью.</td></tr>
</table>

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Подп.	Дата
	изме-ненных	замененных	новых	анну-лиро-ванных				
-	-	-	все	-	835			28.03.11
1	-	44, 125, 172, 211, 212, 253		-	-			10.11.11
2		750	-	-	-			23.11.11
3	-	16,76,77, 234,235,236 721,722	-	-	-			20.12.11
4		6, 7, 581	-	-	-			19.02.13
5	-	220, 221	-	-	-			20.01.15
6	-	7	-	-	-			30.06.15
7	-	20, 78, 213, 225, 229-236	232а	-	836			21.07.15