МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1887ВЕ6Т

Руководство пользователя

2013

Содержание

Введение

В настоящем руководстве пользователя приведено описание архитектуры, функционального построения, системы команд и особенностей применения микросхемы 1887ВЕ6Т. Микросхема представляет собой СБИС 16-разрядного RISC микроконтроллера архитектуры С166 с повышенной стойкостью к СВВФ на базе КНИ-технологии. Это высокоинтегрированное устройство, включающее высоко-производительное процессорное ядро и большое количество современных интерфейсов (USART, CAN, SPI и др.), позволяющих не только организовывать каналы обмена между различными микроконтроллерами, но и управлять микросхемами внешних периферийных устройств (АЦП, EEPROM и т.д.) Поддержка сетевого протокола обмена CAN и DSP функций позволит использовать микросхему в легко масштабируемых, помехоустойчивых сетях обмена и обработки данных. Разработанную СБИС предполагается использовать в модернизированных и перспективных образцах ВВСТ космического назначения, а также в аппаратуре систем управления ракет-носителей.

Настоящее руководство пользователя может служить практическим руководством по применению микроконтроллера для разработчиков систем на основе ИС 1887ВЕ6Т.

1 Назначение, область применения и особенности микросхемы

Микроконтроллер 1887ВЕ6Т сочетает в себе высокую производительность центрального процессорного устройства с функциональными возможностями периферии.

Высокопроизводительное 16-разрядное ЦПУ с 4-ступенчатым конвейером:

- время умножения двух 16-разрядных чисел составляет 400 нс при частоте 25 МГц;
- время деления 32-разрядного числа на 16-разрядное число составляет 800 нс при частоте 25 МГц;
- многочисленные внутренние шины данных с высокой пропускной способностью;
- переключение банков регистров общего назначения (контекста) одной командой;
- 16 Мбайт адресного пространства для кода и данных (архитектура фон-Неймана);
- системный стек с аппаратным контролем переполнения/опустошения.

Высокоэффективная система команд:

- базовая, непосредственная, косвенная, непосредственно-косвенная адресация для работы с битами, байтами и словами;
- улучшенные логические операции для управления периферией и работы с флагами пользователя;
- аппаратный механизм обнаружения исключительных и ошибочных ситуаций;
- поддержка языка высокого уровня для управления операциями семафора и эффективного доступа к данным.

Интерфейс внешней шины:

- мультиплексная/демультиплексная конфигурируемая шина с возможностью переключения между режимами 8- и 16-разрядной шины данных;
- возможность сегментации и формирования сигнала выбора внешнего устройства;
- изменяемые временные параметры циклов шины для пяти программируемых адресных окон.

Прерывания:

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

16-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);
- компоновка каналов;
- возможность исключения излишних действий на сохранение и восстановление состояния системных регистров во время операций обслуживания прерываний.

Интегрированные периферийные модули:

- девять параллельных портов;
- многофункциональный таймерный модуль;
- часы реального времени;
- блоки захвата/сравнения и ШИМ;
- асинхронно/синхронные последовательные приемо-передатчики типа USART;
- поддержка интерфейсов SPI и I2C;
- поддержка интерфейса CAN;
- блок управления выходом из режимов пониженного энергопотребления;
- отладочная система OCDS/CERBERUS и JTAG.

Дополнительные возможности:

- отключение ядра и внутренней периферии;
- отключение только ядра при работе внутренней периферии.

2 Технические характеристики и параметры микросхемы

Микросхема 1887ВЕ6Т представляет собой 16-разрядный RISC-микроконтроллер с тактовой частотой 25 МГц со встроенным ОЗУ и расширенной периферией. Архитектура микроконтроллера оптимизирована для работы в режиме реального времени и обеспечения высокой производительности при малом времени реакции на внешние события. Мощная система команд, встроенные возможности цифровой обработки сигналов в комбинации с интегрированной интеллектуальной периферией со сниженной необходимостью вмешательства ЦПУ, обеспечивают высокую производительность, гибкость и универсальность.

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

Основные технические характеристики микросхемы:

- 16-разрядное ЦПУ с четырехуровневым конвейером команд
- тактовая частота, МГц до 25;
- объем адресуемой памяти, байт 16М;
- объем встроенного ОЗУ, байт 6К;
- объем области регистров специальных функций, байт 1К;
- количество источников прерываний 77;
- количество линий ввода-вывода 103;
- внешних линий вывода ШИМ-сигнала 6;
- каналов захвата/сравнения 32;
- количество асинхронно/синхронных последовательных приемо-передатчиков типа USART 2;
- количество высокоскоростных синхронных последовательных приемо-передатчиков с поддержкой интерфейса SPI 2;
- 16-разрядный многофункциональный таймерный модуль 2;
- программируемый сторожевой таймер 1;
- контроллер интерфейса I2C 1;
- контроллер сдвоенного интерфейса CAN 1;
- отладочный интерфейс JTAG 1;
- режимов пониженного энергопотребления 3;
- напряжение питания микросхемы, В 3,0 – 3,6.

Конструктивные характеристики микросхемы

Кристалл микросхем выполнен по КМОП технологии. Микросхема разработана в металлокерамическом корпусе 4247.144-1 (144CQFP) с четырехсторонним планарным расположением выводов.

2.1 Функциональные параметры

Условное графическое обозначение приведено на рисунке 2.1. Функциональное назначение выводов приведено в таблице 2.1.

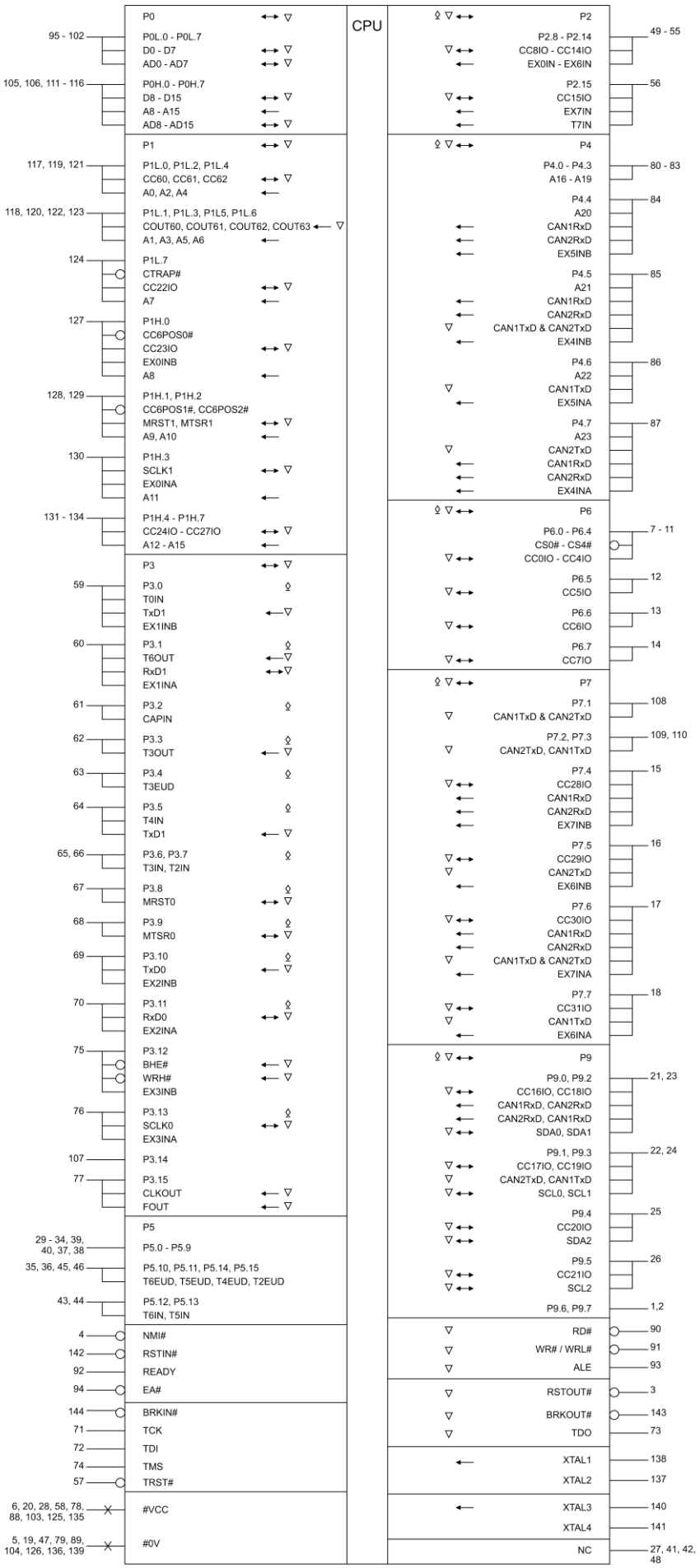


Рисунок 2.1 – Условное графическое изображение микросхемы 1887ВЕ6Т

Таблица 2.1 – Функциональное назначение выводов микроконтроллера

Обозначение		Номер вывода	Функциональное назначение	Тип вывода
вывода	альтернативной функции			
1	2	3	4	5
P0L.0		95	Вход/выход «порт 0, младший байт», 0 разряд	I/O/Z
	D0		Демультиплексированная шина 8/16 бит, вход/выход данных, 0 разряд	I/O/Z
	AD0		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 0 разряд	I/O/Z
P0L.1		96	Вход/выход «порт 0, младший байт», 1 разряд	I/O/Z
	D1		Демультиплексированная шина 8/16 бит, вход/выход данных, 1 разряд	I/O/Z
	AD1		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 1 разряд	I/O/Z
P0L.2		97	Вход/выход «порт 0, младший байт», 2 разряд	I/O/Z
	D2		Демультиплексированная шина 8/16 бит, вход/выход данных, 2 разряд	I/O/Z
	AD2		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 2 разряд	I/O/Z
P0L.3		98	Вход/выход «порт 0, младший байт», 3 разряд	I/O/Z
	D3		Демультиплексированная шина 8/16 бит, вход/выход данных, 3 разряд	I/O/Z
	AD3		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 3 разряд	I/O/Z
P0L.4		99	Вход/выход «порт 0, младший байт», 4 разряд	I/O/Z
	D4		Демультиплексированная шина 8/16 бит, вход/выход данных, 4 разряд	I/O/Z
	AD4		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 4 разряд	I/O/Z
P0L.5		100	Вход/выход «порт 0, младший байт», 5 разряд	I/O/Z
	D5		Демультиплексированная шина 8/16 бит, вход/выход данных, 5 разряд	I/O/Z
	AD5		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 5 разряд	I/O/Z
P0L.6		101	Вход/выход «порт 0, младший байт», 6 разряд	I/O/Z
	D6		Демультиплексированная шина 8/16 бит, вход/выход данных, 6 разряд	I/O/Z
	AD6		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 6 разряд	I/O/Z
P0L.7		102	Вход/выход «порт 0, младший байт», 7 разряд	I/O/Z
	D7		Демультиплексированная шина 8/16 бит, вход/выход данных, 7 разряд	I/O/Z
	AD7		Мультиплексированная шина 8/16 бит, вход/выход адреса/данных, 7 разряд	I/O/Z

Продолжение таблицы 2.1

1	2	3	4	5
P0H.0		105	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
P0H.1		106	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
P0H.2		111	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
P0H.3		112	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
P0H.4		113	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
P0H.5		114	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
P0H.6		115	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

Продолжение таблицы 2.1

1	2	3	4	5
P0H.7	 D15 A15 AD15	116	Вход/выход «порт 0, старший байт», 7 разряд Демультиплексированная шина 16 бит, вход/выход данных, 15 разряд Мультиплексированная шина 16 бит, выход адреса, 15 разряд Мультиплексированная шина 16 бит, вход/выход адреса/данных, 15 разряд	I/O/Z I/O/Z O I/O/Z
P1L.0	 CC60 A0	117	Вход/выход «порт 1, младший байт», 0 разряд Вход/выход модуля захвата/сравнения 6, канал 0 Демультиплексированная шина 8/16 бит, выход адреса, 0 разряд	I/O/Z I/O/Z O
P1L.1	 COUT60 A1	118	Вход/выход «порт 1, младший байт», 1 разряд Выход модуля захвата/сравнения 6, канал 0 Демультиплексированная шина 8/16 бит, выход адреса, 1 разряд	I/O/Z O/Z O
P1L.2	 CC61 A2	119	Вход/выход «порт 1, младший байт», 2 разряд Вход/выход модуля захвата/сравнения 6, канал 1 Демультиплексированная шина 8/16 бит, выход адреса, 2 разряд	I/O/Z I/O/Z O
P1L.3	 COUT61 A3	120	Вход/выход «порт 1, младший байт», 3 разряд Выход модуля захвата/сравнения 6, канал 1 Демультиплексированная шина 8/16 бит, выход адреса, 3 разряд	I/O/Z O/Z O
P1L.4	 CC62 A4	121	Вход/выход «порт 1, младший байт», 4 разряд Вход/выход модуля захвата/сравнения 6, канал 2 Демультиплексированная шина 8/16 бит, выход адреса, 4 разряд	I/O/Z I/O/Z O
P1L.5	 COUT62 A5	122	Вход/выход «порт 1, младший байт», 5 разряд Выход модуля захвата/сравнения 6, канал 2 Демультиплексированная шина 8/16 бит, выход адреса, 5 разряд	I/O/Z O/Z O
P1L.6	 COUT63 A6	123	Вход/выход «порт 1, младший байт», 6 разряд Выход модуля захвата/сравнения 6, канал 3 Демультиплексированная шина 8/16 бит, выход адреса, 6 разряд	I/O/Z O/Z O
P1L.7	 CTRAP# CC22IO A7	124	Вход/выход «порт 1, младший байт», 7 разряд Вход «системное прерывание» модуля захвата/сравнения и ШИМ Вход «захват»/выход «сравнение», 22 канал Демультиплексированная шина 8/16 бит, выход адреса, 7 разряд	I/O/Z I I/O/Z O
P1H.0	 CC6POS0# CC23IO EX0INB A8	127	Вход/выход «порт 1, старший байт», 0 разряд Вход «позиция 0» модуля захвата/сравнения 6 Вход «захват»/выход «сравнение», 23 канал Вход альтернативного сигнала B "прерывание 0" Демультиплексированная шина 8/16 бит, выход адреса, 8 разряд	I/O/Z I I/O/Z I O

Продолжение таблицы 2.1

1	2	3	4	5
P1H.1	 CC6POS1# MRST1 A9	128	Вход/выход «порт 1, старший байт», 1 разряд Вход «позиция 1» модуля захвата/сравнения 6 Вход/выход «M/S ввод-вывод данных синхронного порта 1» Демультиплексированная шина 8/16 бит, выход адреса, 9 разряд	I/O/Z I I/O/Z O
P1H.2	 CC6POS2# MTSR1 A10	129	Вход/выход «порт 1, старший байт», 2 разряд Вход «позиция 2» модуля захвата/сравнения 6 Вход/выход «M/S вывод-ввод данных синхронного порта 1» Демультиплексированная шина 8/16 бит, выход адреса, 10 разряд	I/O/Z I I/O/Z O
P1H.3	 SCLK1 EX0INA A11	130	Вход/выход «порт 1, старший байт», 3 разряд Вход/выход «M-вывода/S-ввода тактового сигнала синхронного порта 1» Вход альтернативного сигнала A «прерывание 0» Демультиплексированная шина 8/16 бит, выход адреса, 11 разряд	I/O/Z I/O/Z I O
P1H.4	 CC24IO A12	131	Вход/выход «порт 1, старший байт», 4 разряд Вход «захват»/выход «сравнение», 24 канал Демультиплексированная шина 8/16 бит, выход адреса, 12 разряд	I/O/Z I/O/Z O
P1H.5	 CC25IO A13	132	Вход/выход «порт 1, старший байт», 5 разряд Вход «захват»/выход «сравнение», 25 канал Демультиплексированная шина 8/16 бит, выход адреса, 13 разряд	I/O/Z I/O/Z O
P1H.6	 CC26IO A14	133	Вход/выход «порт 1, старший байт», 6 разряд Вход «захват»/выход «сравнение», 26 канал Демультиплексированная шина 8/16 бит, выход адреса, 14 разряд	I/O/Z I/O/Z O
P1H.7	 CC27IO A15	134	Вход/выход «порт 1, старший байт», 7 разряд Вход «захват»/выход «сравнение», 27 канал Демультиплексированная шина 8/16 бит, выход адреса, 15 разряд	I/O/Z I/O/Z O
P2.8	 CC8IO EX0IN	49	Вход/выход «порт 2», 8 разряд Вход «захват»/выход «сравнение», 8 канал Вход по умолчанию «прерывание 0»	I/O/Z/2 I/O/Z I
P2.9	 CC9IO EX1IN	50	Вход/выход «порт 2», 9 разряд Вход «захват»/выход «сравнение», 9 канал Вход по умолчанию «прерывание 1»	I/O/Z/2 I/O/Z I
P2.10	 CC10IO EX2IN	51	Вход/выход «порт 2», 10 разряд Вход «захват»/выход «сравнение», 10 канал Вход по умолчанию «прерывание 2»	I/O/Z/2 I/O/Z I
P2.11	 CC11IO EX3IN	52	Вход/выход «порт 2», 11 разряд Вход «захват»/выход «сравнение», 11 канал Вход по умолчанию «прерывание 3»	I/O/Z/2 I/O/Z I

Продолжение таблицы 2.1

1	2	3	4	5
P2.12		53	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
P2.13		54	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
P2.14		55	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
P2.15		56	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
P3.0		59	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
P3.1		60	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
P3.2		61	Вход/выход «порт 3», 2 разряд	I/O/Z/2
	CAPIN		Вход «загрузка»	I
P3.3		62	Вход/выход «порт 3», 3 разряд	I/O/Z/2
	T3OUT		Выход «триггер таймера 3»	O/Z
P3.4		63	Вход/выход «порт 3», 4 разряд	I/O/Z/2
	T3EUD		Вход «направление счета таймера 3»	I
P3.5		64	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
P3.6		65	Вход/выход «порт 3», 6 разряд	I/O/Z/2
	T3IN		Вход «режим таймера 3»	I
P3.7		66	Вход/выход «порт 3», 7 разряд	I/O/Z/2
	T2IN		Вход «режим таймера 2»	I
P3.8		67	Вход/выход «порт 3», 8 разряд	I/O/Z/2
	MRST0		Вход/выход «M/S ввод-вывод данных синхронного порта 0»	I/O/Z
P3.9		68	Вход/выход «порт 3», 9 разряд	I/O/Z/2
	MTSR0		Вход/выход «M/S вывод-ввод данных синхронного порта 0»	I/O/Z
P3.10		69	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

Продолжение таблицы 2.1

1	2	3	4	5
P3.11	 RxD0 EX2INA	70	Вход/выход «порт 3», 11 разряд Вход/выход «синхронный/асинхронный ввод/синхронный вывод данных УСАПП0» Вход альтернативного сигнала A «прерывание 2»	I/O/Z/2 I/O/Z I
P3.12	 BHE# WRH# EX3INB	75	Вход/выход «порт 3», 12 разряд Выход «выбор старшего байта» Выход «запись старшего байта» Вход альтернативного сигнала B «прерывание 3»	I/O/Z O/Z O/Z I
P3.13	 SCLK0 EX3INA	76	Вход/выход «порт 3», 13 разряд Вход/выход «M-вывод/S-ввод тактового сигнала синхронного порта 0» Вход альтернативного сигнала A «прерывание 3»	I/O/Z I/O/Z I
P3.14	–	107	Вход/выход «порт 3», 14 разряд	I/O/Z
P3.15	 CLKOUT FOUT	77	Вход/выход «порт 3», 15 разряд Выход «системный тактовый сигнал» Выход «программируемая частота»	I/O/Z O/Z O/Z
P4.0	 A16	80	Вход/выход «порт 4», 0 разряд Выход адреса, 16 разряд	I/O/Z/2 O
P4.1	 A17	81	Вход/выход «порт 4», 1 разряд Выход адреса, 17 разряд	I/O/Z/2 O
P4.2	 A18	82	Вход/выход «порт 4», 2 разряд Выход адреса, 18 разряд	I/O/Z/2 O
P4.3	 A19	83	Вход/выход «порт 4», 3 разряд Выход адреса, 19 разряд	I/O/Z/2 O
P4.4	 A20 CAN1RxD CAN2RxD EX5INB	84	Вход/выход «порт 4», 4 разряд Выход адреса, 20 разряд Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Вход альтернативного сигнала B «прерывание 5»	I/O/Z/2 O I I I
P4.5	 A21 CAN1RxD CAN2RxD CAN1TxD & CAN2TxD EX4INB	85	Вход/выход «порт 4», 5 разряд Выход адреса, 21 разряд Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Выход «логическое И выходов 1 и 2 данных» интерфейса CAN Вход альтернативного сигнала B «прерывание 4»	I/O/Z/2 O I I O/Z I
P4.6	 A22 CAN1TxD EX5INA	86	Вход/выход «порт 4», 6 разряд Выход адреса, 22 разряд Выход 1 данных интерфейса CAN Вход альтернативного сигнала A «прерывание 5»	I/O/Z/2 O O/Z I
P4.7	 A23 CAN2TxD CAN1RxD CAN2RxD EX4INA	87	Вход/выход «порт 4», 7 разряд Выход адреса, 23 разряд Выход 2 данных интерфейса CAN Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Вход альтернативного сигнала A «прерывание 4»	I/O/Z/2 O O/Z I I I

Продолжение таблицы 2.1

1	2	3	4	5
P5.0	–	29	Вход «порт 5», 0 разряд	I
P5.1	–	30	Вход «порт 5», 1 разряд	I
P5.2	–	31	Вход «порт 5», 2 разряд	I
P5.3	–	32	Вход «порт 5», 3 разряд	I
P5.4	–	33	Вход «порт 5», 4 разряд	I
P5.5	–	34	Вход «порт 5», 5 разряд	I
P5.6	–	39	Вход «порт 5», 6 разряд	I
P5.7	–	40	Вход «порт 5», 7 разряд	I
P5.8	–	37	Вход «порт 5», 8 разряд	I
P5.9	–	38	Вход «порт 5», 9 разряд	I
P5.10	T6EUD	35	Вход «порт 5», 10 разряд Вход «направление счета таймера 6»	I I
P5.11	T5EUD	36	Вход «порт 5», 11 разряд Вход «направление счета таймера 5»	I I
P5.12	T6IN	43	Вход «порт 5», 12 разряд Вход «счет таймера 6»	I I
P5.13	T5IN	44	Вход «порт 5», 13 разряд Вход «счет таймера 5»	I I
P5.14	T4EUD	45	Вход «порт 5», 14 разряд Вход «направление счета таймера 4»	I I
P5.15	T2EUD	46	Вход «порт 5», 15 разряд Вход «направление счета таймера 2»	I I
P6.0	CS0# CC0IO	7	Вход/выход «порт 6», 0 разряд Выход «выбор кристалла 0» Вход «захват»/выход «сравнение», 0 канал	I/O/Z/2 O I/O/Z
P6.1	CS1# CC1IO	8	Вход/выход «порт 6», 1 разряд Выход «выбор кристалла 1» Вход «захват»/выход «сравнение», 1 канал	I/O/Z/2 O I/O/Z
P6.2	CS2# CC2IO	9	Вход/выход «порт 6», 2 разряд Выход «выбор кристалла 2» Вход «захват»/выход «сравнение», 2 канал	I/O/Z/2 O I/O/Z
P6.3	CS3# CC3IO	10	Вход/выход «порт 6», 3 разряд Выход «выбор кристалла 3» Вход «захват»/выход «сравнение», 3 канал	I/O/Z/2 O I/O/Z
P6.4	CS4# CC4IO	11	Вход/выход «порт 6», 4 разряд Выход «выбор кристалла 4» Вход «захват»/выход «сравнение», 4 канал	I/O/Z/2 O I/O/Z
P6.5	CC5IO	12	Вход/выход «порт 6», 5 разряд Вход «захват»/выход «сравнение», 5 канал	I/O/Z/2 I/O/Z
P6.6	CC6IO	13	Вход/выход «порт 6», 6 разряд Вход «захват»/выход «сравнение», 6 канал	I/O/Z/2 I/O/Z
P6.7	CC7IO	14	Вход/выход «порт 6», 7 разряд Вход «захват»/выход «сравнение», 7 канал	I/O/Z/2 I/O/Z

Продолжение таблицы 2.1

1	2	3	4	5
P7.1	CAN1TxD & CAN2TxD	108	Вход/выход «порт 7», 1 разряд Выход 1, 2 данных интерфейса CAN	I/O/Z/2 O/Z
P7.2	CAN2TxD	109	Вход/выход «порт 7», 2 разряд Выход 2 данных интерфейса CAN	I/O/Z/2 O/Z
P7.3	CAN1TxD	110	Вход/выход «порт 7», 3 разряд Выход 1 данных интерфейса CAN	I/O/Z/2 O/Z
P7.4	CC28IO CAN1RxD CAN2RxD EX7INB	15	Вход/выход «порт 7», 4 разряд Вход «захват»/выход «сравнение», 28 канал Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Вход альтернативного сигнала B «прерывание 7»	I/O/Z/2 I/O/Z I I I
P7.5	CC29IO CAN2TxD EX6INB	16	Вход/выход «порт 7», 5 разряд Вход «захват»/выход «сравнение», 29 канал Выход 2 данных интерфейса CAN Вход альтернативного сигнала B «прерывание 6»	I/O/Z/2 I/O/Z O/Z I
P7.6	CC30IO CAN1RxD CAN2RxD CAN1TxD & CAN2TxD EX7INA	17	Вход/выход «порт 7», 6 разряд Вход «захват»/выход «сравнение», 30 канал Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Выход «логическое И выходов 1 и 2 данных» интерфейса CAN Вход альтернативного сигнала A «прерывание 7»	I/O/Z/2 I/O/Z I I O/Z I
P7.7	CC31IO CAN1TxD EX6INA	18	Вход/выход «порт 7», 7 разряд Вход «захват»/выход «сравнение», 31 канал Выход 1 данных интерфейса CAN Вход альтернативного сигнала A «прерывание 6»	I/O/Z/2 I/O/Z O/Z I
P9.0	CC16IO CAN1RxD CAN2RxD SDA0	21	Вход/выход «порт 9», 0 разряд Вход «захват»/выход «сравнение», 16 канал Вход 1 данных интерфейса CAN Вход 2 данных интерфейса CAN Вход/выход 0 «данные» модуля I2C	I/O/Z/2 I/O/Z I I I/O/Z
P9.1	CC17IO CAN2TxD SCL0	22	Вход/выход «порт 9», 1 разряд Вход «захват»/выход «сравнение», 17 канал Выход 2 данных интерфейса CAN Вход/выход 0 «синхронизация» модуля I2C	I/O/Z/2 I/O/Z O/Z I/O/Z
P9.2	CC18IO CAN2RxD CAN1RxD SDA1	23	Вход/выход «порт 9», 2 разряд Вход «захват»/выход «сравнение», 18 канал Вход 2 данных интерфейса CAN Вход 1 данных интерфейса CAN Вход/выход 1 «данные» модуля I2C	I/O/Z/2 I/O/Z I I I/O/Z
P9.3	CC19IO CAN1TxD SCL1	24	Вход/выход «порт 9», 3 разряд Вход «захват»/выход «сравнение», 19 канал Выход 1 данных интерфейса CAN Вход/выход 1 «синхронизация» модуля I2C	I/O/Z/2 I/O/Z O/Z I/O/Z

Продолжение таблицы 2.1

1	2	3	4	5
P9.4	CC20IO SDA2	25	Вход/выход «порт 9», 4 разряд Вход «захват»/выход «сравнение», 20 канал Вход/выход 2 «данные» модуля I2C	I/O/Z/2 I/O/Z I/O/Z
P9.5	CC21IO SCL2	26	Вход/выход «порт 9», 5 разряд Вход «захват»/выход «сравнение», 21 канал Вход/выход 2 «синхронизация» модуля I2C	I/O/Z/2 I/O/Z I/O/Z
P9.6	–	1	Вход/выход «порт 9», 6 разряд	I/O/Z/2
P9.7	–	2	Вход/выход «порт 9», 7 разряд	I/O/Z/2
NMI#	–	4	Вход «немаскируемое прерывание»	I
TRST#	–	57	Вход «сброс системы тестирования»	I
TCK	–	71	Вход «синхронизация JTAG» отладочной системы	I
TDI	–	72	Вход «данные JTAG» отладочной системы	I
TDO	–	73	Выход «данные JTAG» отладочной системы	O/Z
TMS	–	74	Вход «выбор тестового режима JTAG» отладочной системы	I
RD#	–	90	Выход «чтение команд/данных»	O/Z
WR#/WRL#	–	91	Выход «запись данных/запись младшего байта»	O/Z
READY	–	92	Вход «готовность»	I
ALE	–	93	Выход «строб адреса»	O/Z
EA#	–	94	Вход «внешний доступ»	I
RSTOUT#	–	3	Выход «индикация сброса»	O/Z
XTAL2		137	Выход усилителя задающего генератора Вывод для подключения кварцевого резонатора	O –
XTAL1		138	Вход внешнего тактового сигнала Вывод для подключения кварцевого резонатора	I –
XTAL3		140	Вход усилителя дополнительного (32 кГц) тактового генератора Вывод для подключения кварцевого резонатора	I –
XTAL4		141	Выход усилителя дополнительного (32 кГц) задающего генератора Вывод для подключения кварцевого резонатора	O –
RSTIN#		142	Вход «сброс»	I
BRKOUT#		143	Выход «останов» отладочной системы	O/Z
BRKIN#		144	Вход «останов» отладочной системы	I
NC	–	27, 41, 42, 48	Неподключенные выводы	–

Окончание таблицы 2.1

1	2	3	4	5
#VCC	–	6, 20, 28, 58, 78, 88, 103, 125, 135	Выводы питания микросхемы	–
#0V	–	5, 19, 47, 79, 89, 104, 126, 136, 139	Общие выводы микросхемы	–
Примечание – В графе "Тип вывода": I – вход; O – выход; Z – третье состояние; 2 – режим открытого стока.				

2.2 Электрические параметры

Полный перечень электрических параметров микросхемы при приёмке и поставке и предельно допустимые значения параметров приведены в таблицах 2.2 и 2.3 соответственно.

Таблица 2.2 – Электрические параметры микросхемы при приемке и поставке

Наименование параметра, единица измерения, режим измерения		Буквенное обозначение параметра	Норма параметра		Температура среды, ºС
			не менее	не более	
1		2	3	4	5
1 Выходное напряжение низкого уровня на выводах P0 – P4, P6, P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#, В, U_{CC} = 3,0 В, I_{OL} = 5 мА		U_{OL}	–	0,45	–60 ± 3 25 ± 10 85 ± 3
2 Выходное напряжение высокого уровня на выводах P0 – P4, P6, P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#[1], В, U_{CC} = 3,0 В, I_{OH} = –5 мА		U_{OH}	U_{CC}–0,5	–	
3 Ток утечки на входах P5.0 – P5.15[2], мкА, U_{CC} = 3,6 В	U_{IL} = 0 В	I_{ILL1}	–10	–	
	U_{IH} = 3,6 В	I_{ILH1}	–	10	
4 Ток утечки на входах NMI#, TRST#, TCK, TDI, TMS, RSTIN#, BRKIN#, EA#[2], мкА, U_{CC} = 3,6 В	U_{IL} = 0,45 В	I_{ILL2}	–10	–	
	U_{IH} = 3,6 В	I_{ILH2}	–	10	
5 Ток утечки в состоянии «Выключено» по выводам P0 – P4, P6, P7, P9, TDO[2], мкА, U_{CC} = 3,6 В	U_{IL} = 0,45 В	I_{OZL}	–10	–	
	U_{IH} = 3,6 В	I_{OZH}	–	10	
6 Входной ток по выводам P0, RD#, WR#/WRL#, READY[2, 3], мкА, U_{CC} = 3,6 В	U_{IL} = 0,8 В	I_{IL1}	–300	–	
	U_{IH} = 1,8 В	I_{IH1}	–	–10	
7 Входной ток по выводам XTAL1, XTAL3[2], мкА, U_{CC} = 3,6 В	U_{IL} = 0 В	I_{IL2}	–20	–	
	U_{IH} = 3,6 В	I_{IH2}	–	20	
8 Выходной ток по выводу ALE,[2, 3] мкА, U_{CC} = 3,6 В	U_{OL} = 0,8 В	I_{OL1}	10	–	
	U_{OH} = 1,8 В	I_{OH1}	–	270	
9 Выходной ток по выводам P6.0 – P6.4[2, 3], мкА, U_{CC} = 3,6 В	U_{OL} = 0,45 В	I_{OL2}	–200	–	
	U_{OH} = 2,25 В	I_{OH2}	–	–10	

Окончание таблицы 2.2

1	2	3	4	5
10 Динамический ток потребления[4], мА, U_{CC} = 3,6 В, $f_{CIXTAL1}$ = 25 МГц	I_{OCC}	–	160	–60 ± 3 25 ± 10 85 ± 3
11 Ток потребления в режиме холостого хода[4], мА, U_{CC} = 3,6 В, $f_{CIXTAL1}$ = 25 МГц	I_{CCI}	–	40	
12 Ток потребления в режиме хранения, мА, U_{CC} = 3,6 В, $f_{CIXTAL1}$ = 16 МГц	I_{CCS1}	–	5	
13 Ток потребления в режиме хранения, мА, U_{CC} = 3,6 В, $f_{CIXTAL3}$ = 32 кГц	I_{CCS2}	–	3	
14 Функциональный контроль U_{CC} = (3,0; 3,6) В, $f_{CIXTAL1}$ = (4,0; 25,0) МГц	ФК	–	–	

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

[2] 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.

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

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

Таблица 2.3 – 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 Напряжение питания, В	U_{CC}	3,0	3,6	–0,5	5,0
2 Входное напряжение низкого уровня по выводам XTAL1, XTAL3, В	U_{ILCI}	–0,5	$0,3U_{CC}$	–0,6	–
3 Входное напряжение высокого уровня по выводам XTAL1, XTAL3, В	U_{IHCI}	$0,7U_{CC}$	U_{CC}+0,5	–	U_{CC}+0,6
4 Входное напряжение низкого уровня по выводам P0 – P4, P6. P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#, В	U_{IL}	0	$0,2U_{CC}$–1	–0,6	–
5 Входное напряжение высокого уровня по выводам P0 – P4, P6. P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#, В	U_{IH}	$0,2U_{CC}$+1	U_{CC}+0,5	–	U_{CC}+0,6
6 Выходной ток низкого уровня[1], мА	I_{OL}	–	5	–	10
7 Выходной ток высокого уровня[1], мА	I_{OH}	–5	–	–10	–
8 Емкость нагрузки, пФ	C_L	–	50	–	100
9 Частота следования импульсов тактового сигнала, МГц	$f_{CIXTAL1}$	4,0	25,0	–	–
10 Длительность фронтов сигнала для входов XTAL1, XTAL3, нс	t_{LH1}, t_{HL1}	–	5	–	–
11 Длительность фронтов сигнала для остальных входов, нс	t_{LH2}, t_{HL2}	–	10	–	–

Примечание – Время работы в одном из предельных режимов должно быть не более 5 с (кроме п. 8).

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

3 Архитектура изделия

Архитектура микроконтроллера 1887ВЕ6Т объединяет в себе хорошо сбалансированные между собой преимущества архитектур RISC и CISC процессоров. Микроконтроллер не только имеет мощное процессорное ядро и набор периферийных модулей, но также имеет высокоэффективную систему взаимодействия между ними. Наравне с другими шинами, в микроконтроллере используется внутренняя шина X-периферии (шина XBUS), обеспечивающая стандартный способ интеграции в микроконтроллер специальных блоков для различных систем. На рисунке 3.1 показана структурная схема микроконтроллера.

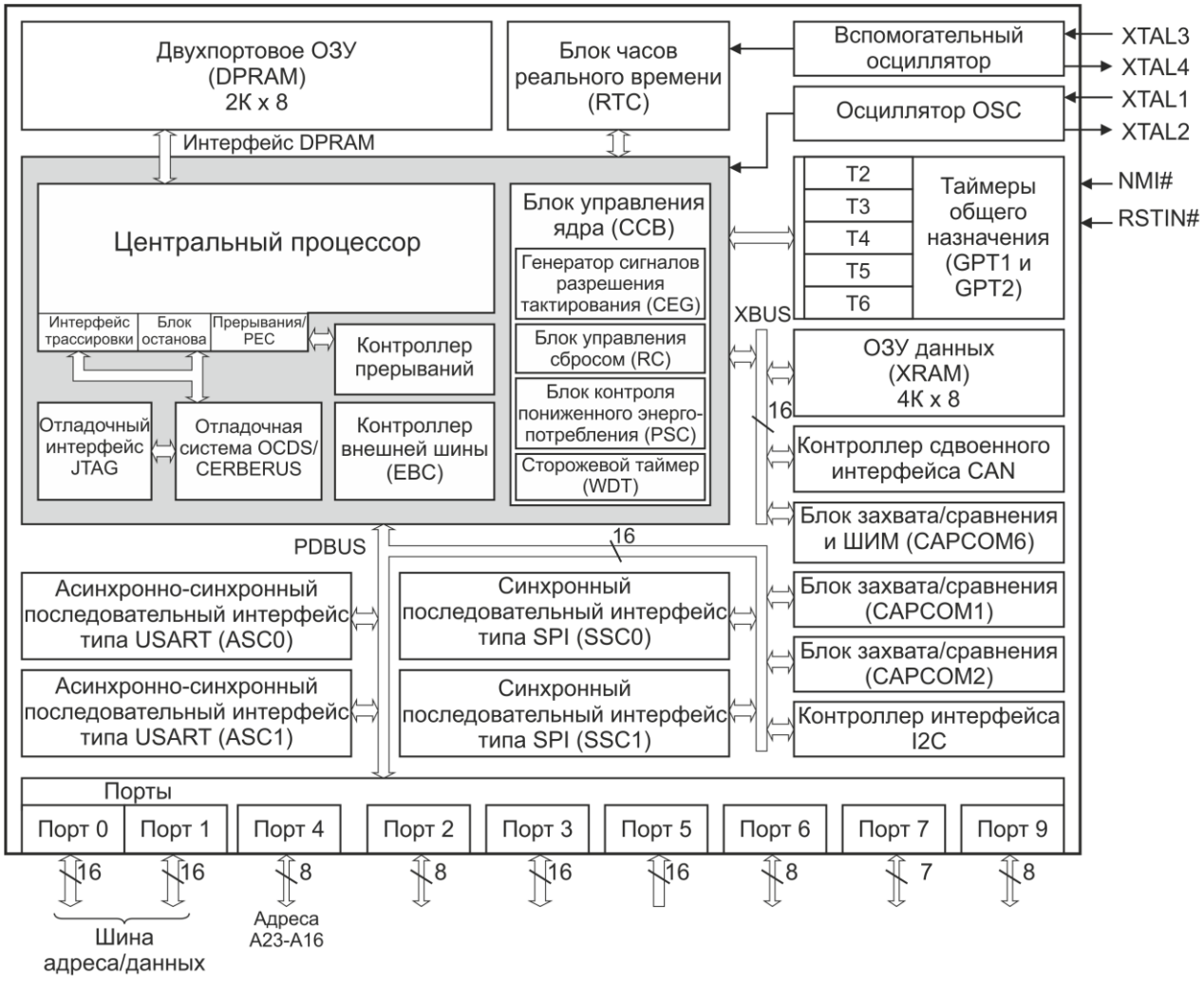


Рисунок 3.1 – Структурная схема микроконтроллера

3.1 Ядро центрального процессорного устройства

Ядро 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.

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:

1 Выборка. Команда считывается из внутренней или внешней памяти по адресу, определяемому значениями указателя команд IP и указателя сегмента кода CSP.

2 Декодирование. Выбранная команда декодируется, и считываются необходимые операнды.

3 Выполнение. Декодированная команда выполняется.

4 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 байт, 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». С помощью команд сравнения и инкремента или декремента, пользователь может сравнивать любые значения. Это позволяет счетчику таблицы охватывать любые пределы и 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:

- реализация длинных команд, которые необходимы для выполнения часто используемых команд и групп команд и совершения параллельных задач – таких, как сохранение режима работы процессора до начала входа в прерывание;
- исключение сложных схем декодирования путем расположения операндов в смежных областях для каждой команды, также исключение редко используемых режимов сложной адресации;
- обеспечение более частого использования команд длиною в слово, что позволяет расположить все команды (в том числе и двухсловные) в границах слов, упростить группировку команд и использовать больший набор команд относительных переходов.

Высокая производительность микроконтроллера 1887ВЕ6Т 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) байт 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 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.

Блок управления ядра (CCB)

Основной управляющий блок, выполняющий все основные задачи управления и специфические задачи и объединяющий в себе четыре блока (см. рисунок 3.1).

Блок управление сбросом (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.

Блок контроля пониженного энергопотребления (PSC) управляет режимами холостого хода Idle и пониженного энергопотребления PowerDown.

Генератор сигналов разрешения тактирования (CEG) формирует сигналы, которые используются различными блоками подсистемы. Основным и единственным сигналом тактирования подсистемы является сигнал Master Clock. На его основе 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. Все блоки подсистемы используют два (прямой и инверсный) тактовых сигнала ЦПУ, которые доступны только внутри подсистемы.

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.

3.2 Системные ресурсы ядра

Микроконтроллер 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 в байт, либо слово. Отдельная часть внутренней памяти может также иметь прямую битовую адресацию.

Внутреннее 16-разрядное ОЗУ (DPRAM) объемом 2 Кбайта обеспечивает быстрый доступ к регистрам общего назначения (регистры 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 CP 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 SP. 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.

3.3 Периферийные модули микроконтроллера

В микроконтроллере 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 (FE00h...FFFFh), либо в области дополнительных регистров области ESFR (F000h...FlFFh). Каждый периферийный модуль имеет набор флагов состояний. Для управления устройствами, подключенными к XBUS, имеется набор регистров области XSFR (E800h...EFFEh).

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 fpdbus 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 имеет 103 канала ввода-вывода, организованных в девять портов ввода-вывода. Все разряды портов являются бит-адресуемыми 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 с максимальной тактовой частотой работы fper/4. Блок GPT2 состоит из двух таймеров – T5, T6 с максимальной тактовой частотой работы fper/2 – и специального регистра для захвата/перезагрузки CAPREL. Каждый таймер может

работать независимо от остальных в различных режимах. Максимальное разрешение таймеров блока GPT1 составляет 800 нс при тактовой частоте 25 МГц, а таймеров блока GPT2 – 400 нс при тактовой частоте 25 МГц.

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

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 32-разрядным таймером и используется для отсчета текущей даты и времени, а также для измерения длительных интервалов времени.

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

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. 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 блок CAPCOM имеет в своем составе шестнадцать регистров захвата/сравнения. Каждый регистр может быть запрограммирован для работы с одним из двух таймеров блока. Помимо этого, с каждым регистром связан соответствующий канал порта, который используется как вход (в режиме захвата) или как выход (в режиме сравнения).

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. Таймер может работать как в режиме однонаправленного счетчика (счет только вверх), так и двунаправленного (счет вверх и вниз).

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

Асинхронные/синхронные 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. Далее по тексту номера 0 и 1 заменены символом «x».

Интерфейс ASCx поддерживает полнодуплексный асинхронный режим работы на скорости приема/передачи до 625 Кбод и полудуплексный синхронный режим на скорости приема/передачи до 2,5 Мбод с учетом того, что fper = 20 МГц.

Интерфейс SSCx работает в полнодуплексном и полудуплексном синхронном режиме на скорости приема/передачи до 10 Мбод в режиме мастера и до 5 Мбод в режиме ведомого с учетом того, что fper = 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 SSCx и канала ASCx.

Для обеспечения надежной связи используется механизм детектирования ошибок, позволяющий выявлять пакеты данных с пропущенными стоповыми битами, ошибки четности (для ASCx), а также ошибки скорости (для SSCx).

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, скоростной и высокоскоростной режимы работы как в качестве мастер шины, так и ведомого устройства.

Модуль CAN интерфейса

Микроконтроллер содержит в своем составе контроллер интерфейса CAN, который в свою очередь состоит из двух идентичных независимых узлов – CAN0 и CAN1 с общей памятью на 64 объекта сообщений. Узлы могут подключаться как к одной общей шине, так и к двум разным.

Интегрированные узлы CAN передают и принимают сообщения в соответствии со спецификацией версии 2.0B, т.е. обеспечивают возможность передачи и приема сообщений как со стандартным (11-разрядным), так и расширенным (29-разрядным) идентификаторами. Идентификаторы могут маскироваться.

Каждый объект сообщения программируется и функционирует независимо, имеет собственный идентификатор и может хранить данные размером до 8 байт.

Контроллер CAN стабильно функционирует в тяжелых условиях (по стандарту ISO11898) таких, как обрыв одного из трех проводов шины и закорачивание одного из проводов шины на питание/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.

Контроллер CAN реализует программируемую скорость обмена данными до 1 Мбит/с. Каждый узел CAN может принимать сообщения с любой из семи отведенных

под эти функции линий портов. Для передачи сообщений могут быть задействованы до восьми линий (по четыре линии на узел).

Блок управления выходом из режимов Idle, Sleep и PowerDown

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.

4 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 4.1.

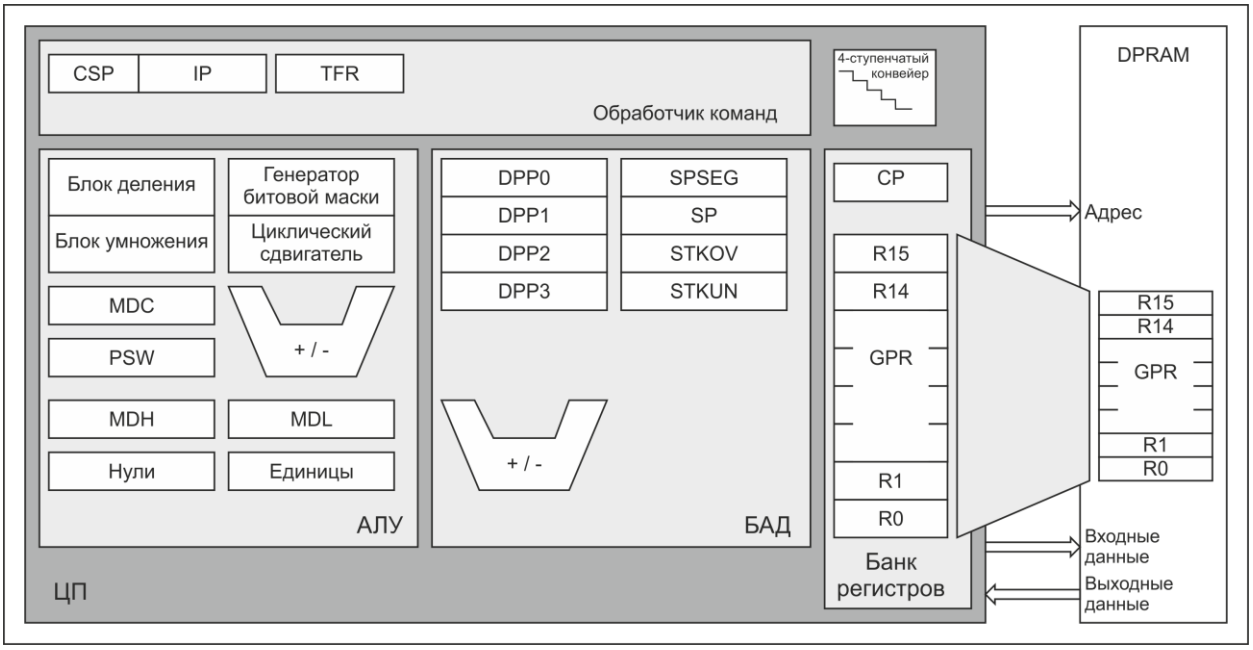


Рисунок 4.1 – Архитектура ЦПУ

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.

1 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.

2 Обработчик вставки/исключения команд выполняет:

- обработку запросов прерываний;
- обработку аппаратных сбоев и отказов.

Конвейер команд

Конвейер команд – 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.

4.1 Регистры специального назначения

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, разбиения памяти данных на страницы и доступа к регистрам основного назначения и системному стеку ядро ЦПУ использует регистры специального назначения – регистры SFR.

Регистры специального назначения расположены в двух 512-байтных областях. Первый блок регистра области SFR занимает 512 байт выше DPRAM (FFFFh – FE00h). Второй блок регистра области ESFR занимает 512 байт ниже DPRAM (F1FFh – F000h).

Все SFR могут управляться посредством любых команд, действительных для адресного пространства SFR, поэтому отпадает необходимость в создании набора специальных системных команд.

Примечание – Доступ к некоторым регистрам имеет особенности:

- недоступны регистры IP (указатель команд) и CSP (указатель сегмента кода) – их значения могут быть только косвенно изменены посредством команд перехода;

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

Регистры SFR доступны пословно или побайтно (некоторые – побитно). Чтение байта регистра – стандартная операция. В тоже время, любые операции программной записи одного байта в SFR очищают значение второго байта этого SFR. Неиспользуемые (зарезервированные) биты SFR не могут быть изменены и всегда выдают значение «0» при попытке чтения. Описание регистров находится в приложении А.

Адресация регистров SFR с помощью косвенного или длинного 16-битного режимов адресации. Слово и соответствующий ему младший байт могут быть адресованы с использованием 8-битного сдвига вместе с неявным базовым адресом. Обращение к старшему байту регистра SFR с использованием режима 8-битного сдвига невозможно.

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

Если возникает необходимость обращения к регистрам области ESFR, то при использовании 8-битной или прямой битовой адресации следует выполнить команду расширения EXTR. Это не требуется в случае 16-битной и косвенной адресации.

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 области ESFR в основном используются для инициализации и выбора режима при минимизации переключений банков SFR. Регистры, которые требуются для частого доступа, размещаются в стандартной области SFR.

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.

4.2 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 4.1.

Таблица 4.1 – Способы адресации при переходах

Мнемокод	Адрес перехода	Сегмент перехода	Допустимый диапазон адресов
caddr	(IP) = caddr	–	caddr = 0000h … FFFEh
rel	(IP) = (IP) + 2 * rel (IP) = (IP) + 2 * (rel# + 1)	–	rel = 00h…7Fh rel = 80h…FFh
[Rw]	(IP) = (Rw)	–	Rww = 0…15
seg	–	(CSP) = seg	seg = 0…255
#trap7	(IP) = 0000h + 4 * trap7	(CSP) = 0000h	trap7 = 00h…7Fh

Пояснения к таблице 4.1:

- 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, следующую за командой перехода. В зависимости от расположения начала диапазона адресов, возможны как переходы «вверх» (rel = 00h…7Fh), так и переходы «вниз» (rel = 80h…FFh). Команда перехода может выполняться повторно, если rel = –1 (FFh) для 16-разрядных команд перехода или если rel = –2 (FEh) для 32-разрядных команд перехода;

- [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. Микроконтроллер 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 00h до 7Fh может быть определен для доступа к любой 32-разрядной позиции кода в пределах диапазона адресов 0000h – 01FCh 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 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 1887ВЕ6Т 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 4.2. Здесь и далее на иллюстрациях приняты обозначения:

- 1 цикл – один машинный цикл;

- I_n – команда n;
- I_t – целевая команда t;
- (I_{inj}) – системная инжектированная команда;
- * – переход с сохранением в кэш;
- ** – инжектирование сохраненной в кэш команды.

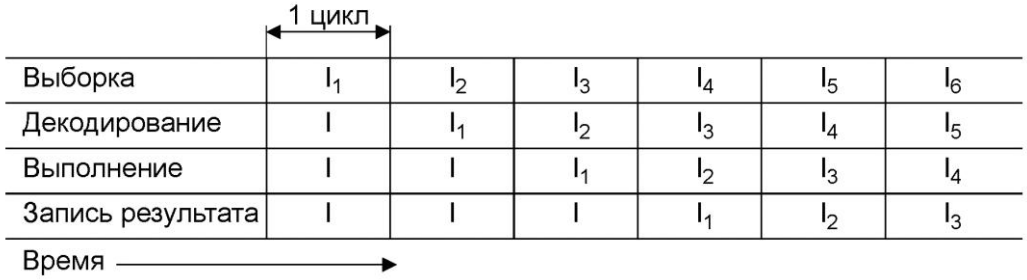


Рисунок 4.2 – Последовательное выполнение команд на конвейере

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 4.3.

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.

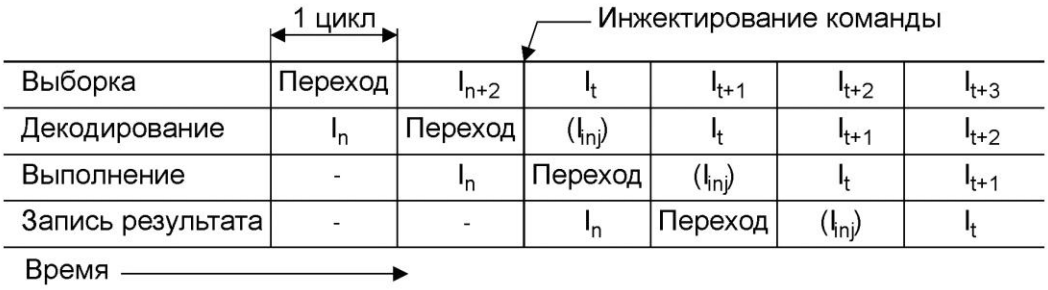


Рисунок 4.3 – Выполнение команд перехода

Кэш для инструкций перехода

Микроконтроллер 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 4.4.

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

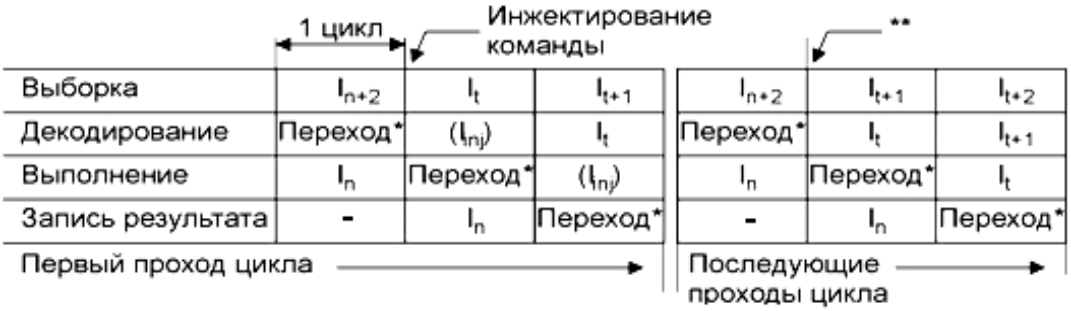


Рисунок 4.4 – Выполнение команд перехода с сохранением в кэш

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 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 4.5.

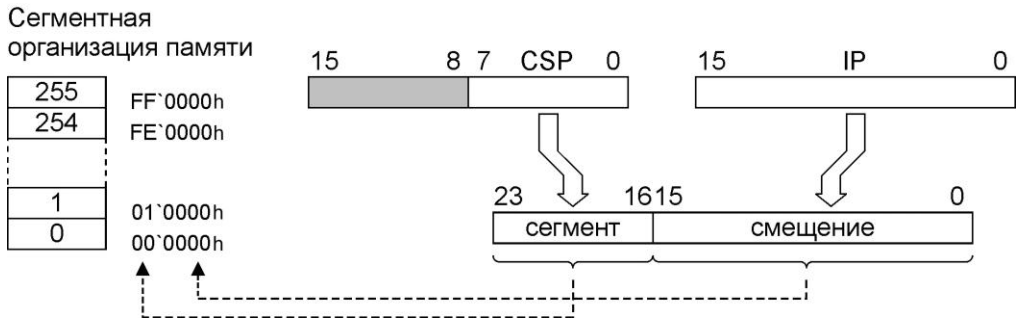


Рисунок 4.5 – Адресация посредством CSP и IP 00h

Указатель команд IP

Регистр IP определяет внутренний 16-битный сегментный адрес текущей команды, при этом сегмент данных выбирается при помощи регистра CSP. Регистр IP размещен вне адресного пространства микроконтроллера 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.

Указатель сегмента кода CSP

CSP – побитно не адресуемый регистр, отвечающий за выбор сегмента кода для доступа к командам. Младшие 8 бит регистра CSP выбирают один из 256 сегментов.

Адрес ячейки кода создается путем прямого расширения 16-битного содержимого IP-регистра содержимым регистра CSP.

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.

В режиме 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.

В режиме 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 и управления им используется побитно адресуемый регистр SYSCON. Состояние регистра после сброса зависит от состояния конфигурационных входов во время сброса.

4.3 Регистры общего назначения

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

Содержимое регистра контекстного указателя CP определяет базовый адрес банка регистров, активного в настоящий момент (см. рисунок 4.6). 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.

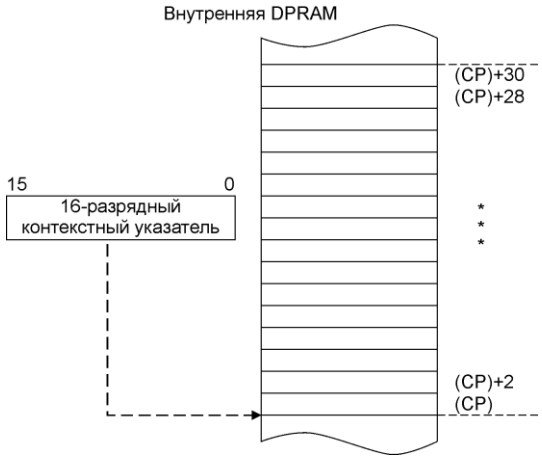


Рисунок 4.6 – Выбор банка регистров посредством регистра CP

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 4.2.

Таблица 4.2 – Карта адресов регистров области GPR

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

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 четыре значащих бита в диапазоне от F0h до FFh, 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 4.7.

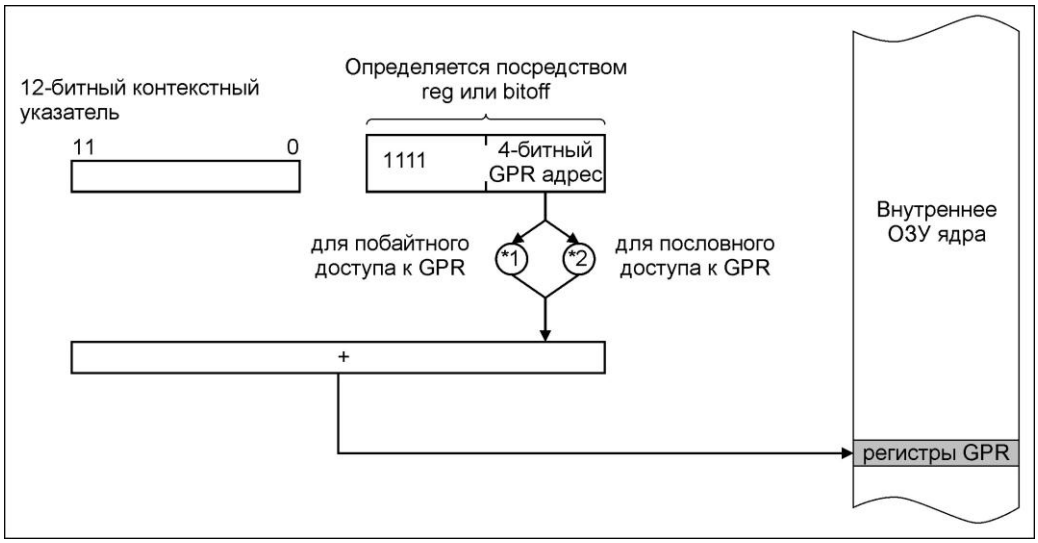


Рисунок 4.7 – Неявное использование регистра CP логикой режима короткой адресации

24-битный режим адресации

24-разрядные адреса памяти в пределах диапазона CP от 0 до 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 4.3. Первые восемь регистров R0, …, R7 из GPR доступны побайтно. При этом запись в адресуемый байт регистра не оказывает влияние на содержимое второго байта того же регистра.

Таблица 4.3 – Режимы адресации для доступа к GPR-словам

Имя	Физический адрес	8-битный адрес	4-битный адрес	Описание	Значение после сброса
R0	(CP) + 0	F0h	0h	Регистр R0 (слово)	Не определено
R1	(CP) + 2	F1h	1h	Регистр R1 (слово)	
R2	(CP) + 4	F2h	2h	Регистр R2 (слово)	
R3	(CP) + 6	F3h	3h	Регистр R3 (слово)	
R4	(CP) + 8	F4h	4h	Регистр R4 (слово)	
R5	(CP) + 10	F5h	5h	Регистр R5 (слово)	
R6	(CP) + 12	F6h	6h	Регистр R6 (слово)	
R7	(CP) + 14	F7h	7h	Регистр R7 (слово)	
R8	(CP) + 16	F8h	8h	Регистр R8 (слово)	
R9	(CP) + 18	F9h	9h	Регистр R9 (слово)	
R10	(CP) + 20	FAh	Ah	Регистр R10 (слово)	
R11	(CP) + 22	FBh	Bh	Регистр R11 (слово)	
R12	(CP) + 24	FCh	Ch	Регистр R12 (слово)	
R13	(CP) + 26	FDh	Dh	Регистр R13 (слово)	
R14	(CP) + 28	FEh	Eh	Регистр R14 (слово)	
R15	(CP) + 30	FFh	Fh	Регистр R15 (слово)	

Каждая половина регистра-байта (8 бит) имеет свое уникальное имя, см. таблицу 4.4.

Таблица 4.4 – Режимы адресации для доступа к GPR-байтам

Имя	Физический адрес	8-битный адрес	4-битный адрес	Описание	Значение после сброса
RL0	(CP) + 0	F0h	0h	Регистр RL0 (байт)	Не определено
RH0	(CP) + 1	F1h	1h	Регистр RL1 (байт)	
RL1	(CP) + 2	F2h	2h	Регистр RL2 (байт)	
RH1	(CP) + 3	F3h	3h	Регистр RL3 (байт)	
RL2	(CP) +4	F4h	4h	Регистр RL4 (байт)	
RH2	(CP) + 5	F5h	5h	Регистр RL5 (байт)	
RL3	(CP) + 6	F6h	6h	Регистр RL6 (байт)	
RH3	(CP) + 7	F7h	7h	Регистр RL7 (байт)	
RL4	(CP) + 8	F8h	8h	Регистр RL8 (байт)	
RH4	(CP) + 9	F9h	9h	Регистр RL9 (байт)	
RL5	(CP) + 10	FAh	Ah	Регистр RL10 (байт)	
RH5	(CP) + 11	FBh	Bh	Регистр RL11 (байт)	
RL6	(CP) + 12	FCh	Ch	Регистр RL12 (байт)	
RH6	(CP) + 13	FDh	Dh	Регистр RL13 (байт)	
RL7	(CP) + 14	FEh	Eh	Регистр RL14 (байт)	
RH7	(CP) + 15	FFh	Fh	Регистр RL15 (байт)	

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

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.

4.4 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 (см. таблицу 4.5).

Возможно выполнение следующих задач:

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

Таблица 4.5 – 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’FE00h + 2*reg 00’F000h + 2*reg (CP) + 2*(reg^0Fh) (CP) + 1*(reg^0Fh)	reg = 00h...EFh reg = 00h...EFh reg = F0h...FFh reg = F0h...FFh	Область SFR (слово, младший байт) Область ESFR (слово, младший байт) Область GPR (слово, байт)
bitoff	00’FD00h + 2*bitoff 00’FF00h + 2*(bitoff^7Fh) 00’F100h + 2*(bitoff^7Fh) (CP) + 2*(bitoff^0Fh)	bitoff= 00h...7Fh bitoff= 80h...EFh bitoff= 80h...EFh bitoff= F0h...FFh	Бит в смещении слова из областей DPRAM, SFR, ESFR, GPR
bitaddr	Смещение как в bitoff. Непосредственная позиция бита (bitpos)	bitoff = 00h...FFh bitpos = 0...15	Любой бит

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.

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

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

Пояснения к таблице 4.5:

- 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 00h до EFh всегда определяют (E)SFR. В этом случае базовый адрес будет FE00h для стандартной области SFR или F000h 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 F0h до FFh 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 00h до 7Fh используют в качестве базового адреса значение FD00h для определения 128 верхних слов DPRAM в диапазоне от FD00h до FDFEh. Короткая адресация bitoff в диапазоне от 80h к EFh использует базовый адрес FF00h, чтобы определить внутреннее положение слова SFR в диапазоне FF00h до FFDEh или базового адреса F100h, чтобы определить внутреннее положение слова ESFR в диапазоне от F100h до F1DEh. 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 F0h до FFh 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 4.8.

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

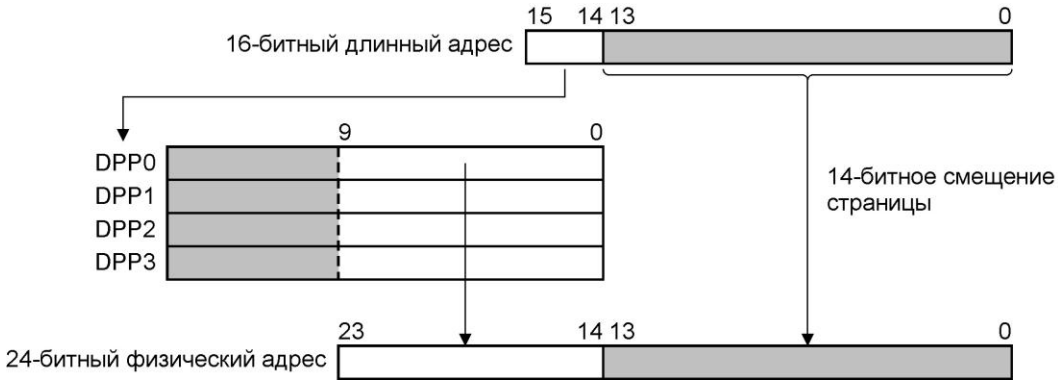


Рисунок 4.8 – Интерпретация 16-битного длинного адреса

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

Адресация с использованием указателя страницы данных DPP

Четыре не адресуемых побитно регистра DPP выбирают одну из четырех страниц данных. Младшие 10 битов каждого регистра DPP выбирают одну из 1 024 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, что показано на рисунке 4.9.

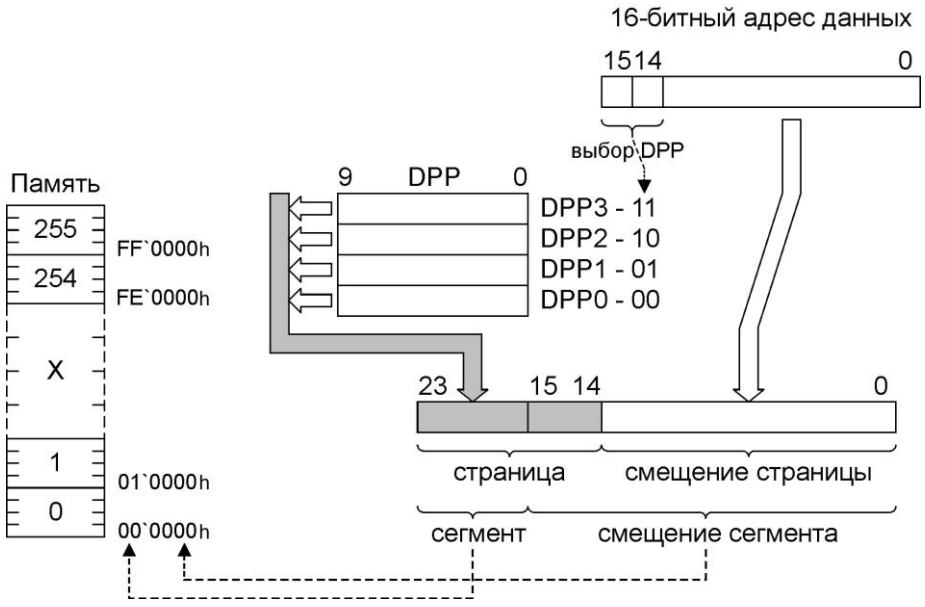


Рисунок 4.9 – Адресация посредством регистров DPP

После сброса содержимое регистров DPP указывает на третью, вторую, первую и нулевую страницы данных нулевого сегмента.

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 использованием 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), что проиллюстрировано на рисунке 4.10.

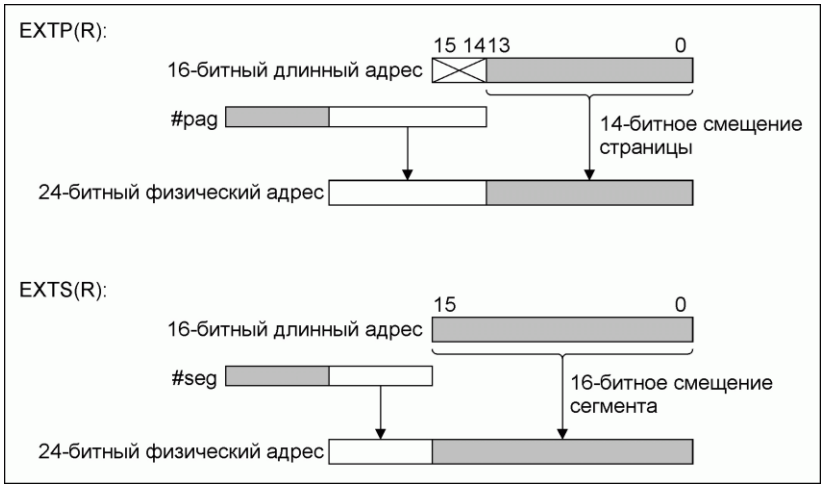


Рисунок 4.10 – Механизм замещения регистра DPP

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 4.6.

Режим длинной адресации обозначается как mem.

Таблица 4.6 – Режимы длинной адресации

Мнемоника	Физический адрес	Область доступа
mem	(DPP0)|| mem^3FFFh (DPP1)|| mem^3FFFh (DPP2)|| mem^3FFFh (DPP3)|| mem^3FFFh	Любое слово или байт
mem	pag|| mem^3FFFh	
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:

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

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:

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

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 4.7.

Таблица 4.7 – 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

4.5 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 F000h до FFFEh. 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.

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

Переполнение и опустошение стека

Направлением перемещения данных из стека/в стек управляют два регистра STKOV (указатель переполнения стека) и STKUN (указатель опустошения стека). Специальные системные 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 F000h до FFFEh.

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 F000h до FFFEh.

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.

4.6 Линейный стек

В микроконтроллере реализована функция линейного стека (STKSZ = 111b), в которой вся область DPRAM может использоваться как системный стек. Это позволяет организовать более объемный стек без необходимости реализации обработки перемещения данных для циркулярного стека. В тоже 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 F000h до FFFEh, системный (физический) стек должен быть расположен в

пределах области DPRAM и только в диапазоне от F600h до FDFEh. 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 FE00h до FFFEh (область SFR), могут привести к непредсказуемым результатам.

4.7 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 F000h до FFFEh). 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 4.8.

Таблица 4.8 – Трансформация адресов циркулярного стека регистра SYSCON

STKSZ	Размер стека, количество слов	Адрес DPRAM (по словам) физического стека	Значимые биты указателя стека SP
000b	256	FBFEh – FA00h по умолчанию после сброса	SP.8 – SP.1
001b	128	FBFEh – FB00h	SP.7 – SP.1
010b	64	FBFEh – FB80h	SP.6 – SP.1
011b	32	FBFEh – FBC0h	SP.5 – SP.1
100b	512	FBFEh – F800h Не использовать, если DPRAM – объемом 1 Кбайт	SP.9 – SP.1
101b	Зарезервировано. Не использовать		
110b			
111b	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 FBFEh. Эта трансформация осуществляется аппаратно.

Значения регистров после сброса STKOV = FA00h, STKUN = FC00h, SP = FC00h, STKSZ = 000b таким образом распределяют область стека, что создается возможность использования системного стека без каких-либо изменений, не выходя за границы 256 слов. Формирование физического адреса стека представлено на рисунке 4.11.

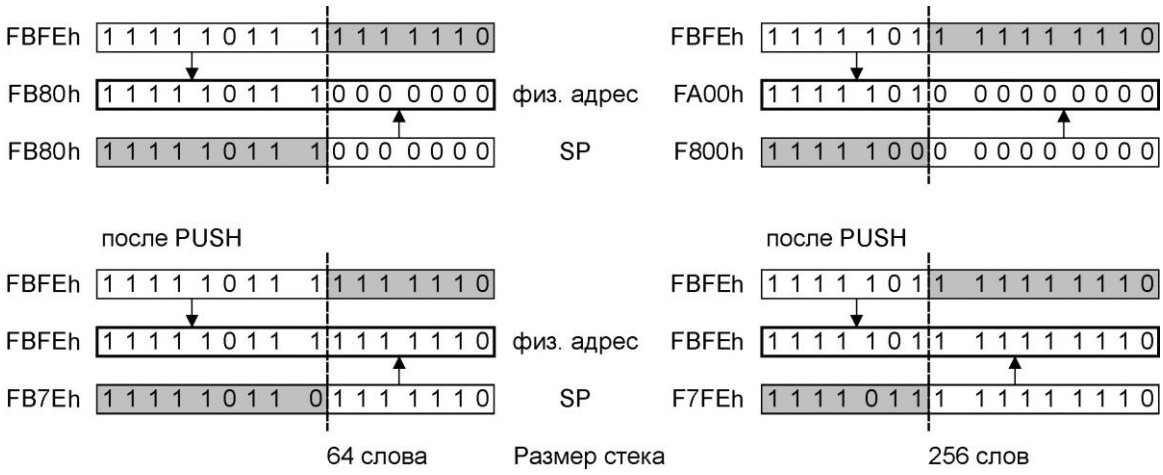


Рисунок 4.11 – Формирование физического адреса стека

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 000b до 100h. Механизм не работает при STKSZ= = 111b, 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 1/10 до 1/4 размеров внутреннего стека. После завершения перемещения указатели границ обновляются для отражения распределенного заново 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.

4.8 Обработка данных

Все стандартные арифметические, сдвиговые и логические операции производятся в 16-разрядном арифметико-логическом устройстве АЛУ. В дополнение к стандартному АЛУ микроконтроллер 1887ВЕ6Т 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 поддерживает операции с битами, битовыми строками, символами и целочисленными типами данных. Большинство команд работает с определенными типами данных, см. таблицу 4.9. Форматы данных поддерживают все типы данных ANSI C. Кроме того, некоторые Си-компиляторы поддерживают новые типы, которые позволяют эффективнее использовать команды. Микроконтроллер непосредственно поддерживает форматы данных ЦПУ, которые приведены в таблице 4.9.

Таблица 4.9 – Типы данных ANSI C

Тип данных ANSI C	Размер	Диапазон	Формат данных ЦПУ
1	2	3	4
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 –128 до 127	BYTE
unsigned char	8 бит	от 0 до 255	BYTE
sfr	8 бит	от 0 до 65 535	WORD
esfr	8 бит	от 0 до 65 535	WORD
signed short	16 бит	от –32 768 до 32 767	WORD
unsigned short	16 бит	от 0 до 65 535	WORD
bitword	16 бит	от 0 до 65 535	WORD или BIT
signed int	16 бит	от –32 768 до 32 767	WORD

Окончание таблицы 4.9

1	2	3	4
unsigned int	16 бит	от 0 до 65 535	WORD
signed long	32 бит	от –2 147 483 648 до 2 147 483 647	Не поддерживает
unsigned long	32 бит	от 0 до 4 294 967 295	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

Таблица 4.10 – Форматы данных центрального процессорного устройства

Формат данных ЦПУ	Размер	Диапазон
BIT	1 бит	0 или 1
BYTE	8 бит	от 0 до 255 (без знака) или от –128 до 127
WORD	16 бит	от 0 до 65 535 (без знака) или от –32 768 до 32 767

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 4.11.

Таблица 4.11 – Формат констант

Мнемоника	Операция со словом	Операция с байтом
#data3	0000h + data3	00h + data3
#data4	0000h + data4	00h + data4
#data8	0000h + data8	Data8
#data16	data16	Data16 ^ FFh
#mask	0000h + 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 переноса C (регистр PSW) 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 байт, 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

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.

Младший регистр умножения/деления MDL

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.

Как только программно изменяется значение регистра, устанавливается флаг MDRIU регистра MDC. В случае программного чтения регистра MDL, флаг MDRIU сбрасывается.

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.

Управляющий регистр умножения/деления MDC

Адресуемый побитно 16-битный регистр. Используется ЦПУ во время совершения умножения или деления для сохранения требуемой информации для управления операциями умножения и деления. Значение регистра MDC изменяется аппаратно, в течение каждого такта команды умножения и деления.

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 возобновить 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 (регистр PSW) 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 с установкой флага 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 также 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

Побитно адресуемый регистр. Отражает 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.

Флаги состояния (N, C, V, Z, E, MULIP) показывают состояние АЛУ после выполнения последней операции. 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 регистра PSW, флаги состояния не могут быть распознаны, как описано, так как явное чтение состояния регистра сменит значения флагов состояния, потому что они генерируются ЦПУ. Явное чтение регистра PSW выдаст результат в виде значения, которое показывает состояние регистра после выполнения непосредственно предшествовавшей операции.

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

4.9 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 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 1:

```
scxt CP, #0FC00h  ; Сохранение и задание нового значения базового адреса контекста

                  ; Должна быть любая инструкция, не использующая GPR
…
mov r0, #data     ; Запись в GPR нового значения
```

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 2:

```
mov DPP0, #4  ; Выбор четвертой страницы данных через DPP0.
              ; 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
mov 0000h, r1 ; 01'0000h (четвертая страница).
```

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 3:

```
mov SP, #0FA40h   ; Загрузка нового значения указателя вершины стека.
                  ; 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 четыре 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 регистр 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 регистра PSW, необходимо выполнить минимум одну команду, которая (или которые) не критична к обработкам запросов.

Пример 4:

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

Инициализация портов

Настройка портов микроконтроллера на ввод или вывод осуществляется путем обнуления или установки в единичное значение соответствующего бита в регистре DPx (x – номер порта). Изменение отдельных битов использует внутреннюю последовательность «чтение-изменение-запись», 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 5:

```
                ; НЕПРАВИЛЬНО:
bset DP3.13     ; Установка канала P3.13 на вывод.
                ; 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
  bset P3.5     ; выходной триггер P3.13.

                ; ПРАВИЛЬНО:
bset DP3.13     ; Установка канала P3.13 на вывод.
                ; 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.
bset P3.5       ; Прочитанное значение запишется обратно, не изменив состояния на
                ; выходе P3.13.
```

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

Оба примера, приведенные ниже, являются корректными.

Пример 6:

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

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 7:

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

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 (регистр 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.

Изменение регистров управления внешней шиной

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.

4.10 Специализированные регистры CSFR

Регистр постоянного нуля ZEROS

Побитно адресуемый регистр, доступный только для чтения. Все биты аппаратно зафиксированы в нуле. 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.

Регистр постоянной единицы ONES

Побитно адресуемый регистр, доступный только для чтения. Все биты аппаратно зафиксированы в единице. 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 адресоваться к регистру области SFR.

Регистр идентификации CPUID

Регистр содержит номер и версию микроконтроллера 1887ВЕ6Т.

5 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 5.1.

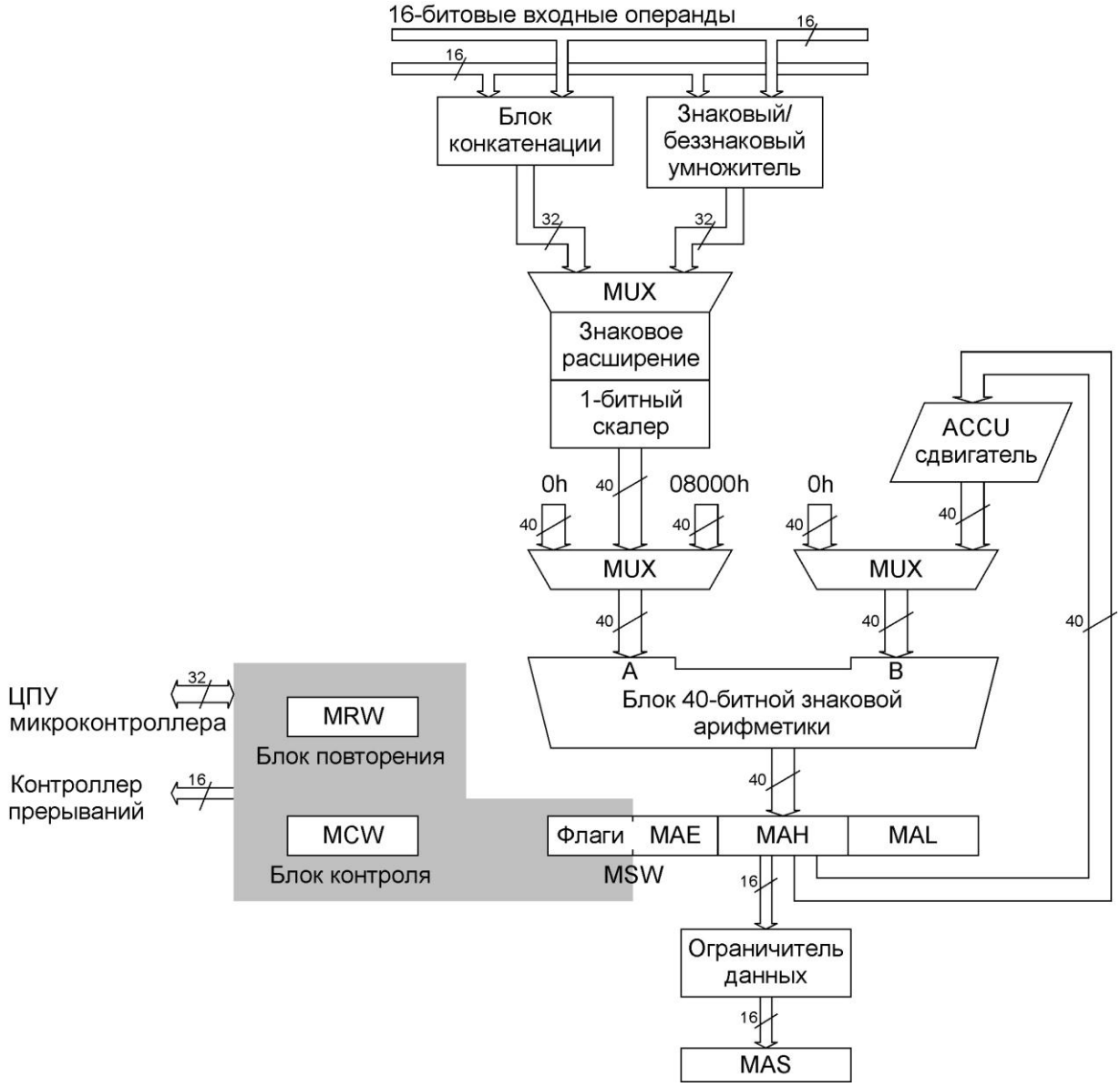


Рисунок 5.1 – Архитектура блока MAC

Блок MAC имеет особенности:

- 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);

- выполнение всех операций за один машинный цикл ЦПУ;

- возможность повтора некоторых команд до 8 192 раз с возможностью прерывания цикла;

- генерация прерываний (аппаратные ловушки класса B) при установке соответствующих флагов.

5.1 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 5.1. Символы [$Rw_n\otimes$] и [$IDX_i\otimes$] относятся к данным режимам адресации.

Таблица 5.1 – Указатели адреса с последующим изменением. Комбинации для [$IDX_i\otimes$] и [$Rw_n\otimes$]

Символ	Мнемоника	Операция указателя адреса
[$IDX_i\otimes$]	[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)
[$Rw_n\otimes$]	[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 5.2.

Таблица 5.2 – 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 5.2.

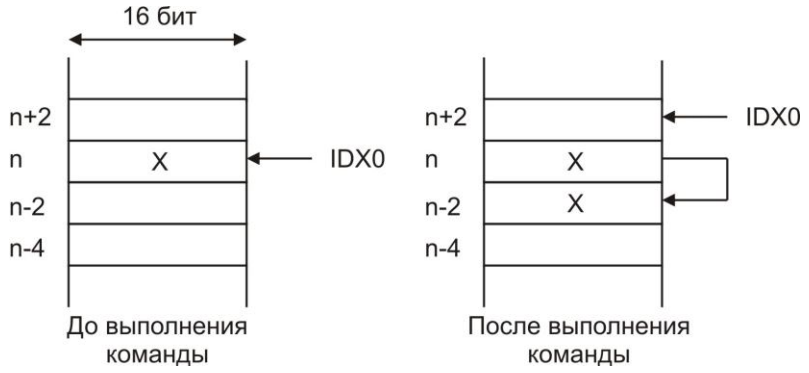


Рисунок 5.2 – Пример параллельной пересылки данных

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 5.3.

Таблица 5.3 – 5-битный режим адресации CoReg

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

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).

5.2 Компоненты блока MAC

Основные компоненты блока MAC показаны на рисунке 5.1.

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 0000000000h, 0000008000h (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 0000000000h, 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 0000008000h 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 007FFFFFFh (положительное) или FF80000000h (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 007FFF0000h.

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 007FFFFFFh или FF80000000h.

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 5.4.

Таблица 5.4 – Выходные значения ограничителя данных

Бит E	Бит N	Выход ограничителя MAS
0	X	Равно значению регистра MAH
1	0	7FFFh
1	1	8000h

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} (8 192) 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 за вычетом единицы.

```
MOV MRW, #00FFh                     ; 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.

5.3 Прерывания блока MAC

Блок MAC может генерировать прерывания в соответствии со значениями флагов состояния 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 Ah, 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.

5.4 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.

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

40-битный аккумулятор состоит из регистров: MAH, MAL и младшего байта регистра MSW. Регистры MAH и MAL не являются бит-адресуемыми SFR. Регистр MAL автоматически обнуляется, если происходит запись в регистр MAH с помощью стандартных команд ЦПУ.

Бит-адресуемый статусный регистр MSW отображает текущее состояние блока MAC. Данный регистр состоит из 8-битного расширения аккумулятора MAE и семи флагов. Флаги состояния изменяются (при необходимости) при выполнении команды. Они не изменяются при выполнении стандартных команд ЦПУ.

Бит-адресуемый регистр MCW управляет работой блока MAC и определяет функциональность прерываний.

Регистр повторений MRW содержит число повторений, которые должна выполнить команда.

5.5 Система команд MAC

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 5.5.

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 и режимы адресации блока MAC представлены в таблице 5.5.

Таблица 5.5 – Команды и режимы адресации

Мнемокод	Режимы адресации	Повтор
1	2	3
CoMul	Rwn, Rwm Rwn, [Rwm⊗] [IDXi⊗], [Rwm⊗]	Нет
CoMulu		
CoMulus		
CoMulsu		
CoMul-		
CoMulu-		
CoMulus-		
CoMulsu-		
CoMul + rnd		
CoMulu + rnd		
CoMulus + rnd		
CoMulsu + rnd		
CoMAC	Rwn, Rwm Rwn, [Rwm⊗] [IDXi⊗], [Rwm⊗]	Нет Да Да
CoMACu		
CoMACus		
CoMACsu		
CoMAC-		
CoMACu-		
CoMACus-		
CoMACsu-		
CoMAC + rnd		
CoMACu + rnd		

Продолжение таблицы 5.5

1	2	3
CoMACus + rnd	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i⊗], [Rw_m⊗]	Нет Да Да
CoMACsu + rnd		
CoMACR		
CoMACRu		
CoMACRus		
CoMACRsu		
CoMACR + rnd		
CoMACRu + rnd		
CoMACRus + rnd		
CoMACRsu + rnd		
CoNOP	[Rw_n ⊗] [IDX_i ⊗] [IDX_i ⊗], [Rw_m ⊗]	Да
CoNEG	–	Нет
CoNEG + rnd		
CoRND		
CoSTORE	Rw_n, CoReg [Rw_n ⊗], CoReg	Нет Да
CoMov	[IDXi ⊗], [Rwm ⊗]	Да
CoMACM		
CoMACMu		
CoMACMus		
CoMACMsu		
CoMACM-		
CoMACMu-		
CoMACMus-		
CoMACMsu-		
CoMACM + rnd		
CoMACMu + rnd		
CoMACMus + rnd		
CoMACMsu + rnd		
CoMACMRu		
CoMACMRus		
CoMACMRsu		
CoMACMR + rnd		
CoMACMRu + rnd		
CoMACMRus + rnd		
CoMACMRsu + rnd		
CoADD	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i ⊗], [Rw_m ⊗]	Нет Да Да
CoADD2		
CoSUB		
CoSUB2		
CoSUBR		
CoSUB2R		
CoMAX		
CoMIN		

Окончание таблицы 5.5

1	2	3
CoLOAD	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i], [Rw_m⊗]	Нет Нет Нет
CoLOAD-		
CoLOAD2		
CoLOAD2-		
CoCMP		
CoSHL	#data4 Rw_m [Rw_m⊗]	Нет Да Да
CoSHR		
CoASHR		
CoASHR + rnd		
CoABS	Rw_n, Rw_m Rw_n, [Rw_m⊗] [IDX_i⊗], [Rw_m⊗]	Нет

6 Организация памяти

Пространство памяти ИС 1887ВЕ6Т 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 DPRAM, внутренние регистры специального назначения SFR, расширенные регистры специального назначения ESFR, память XRAM и внешняя память расположены в одном общем адресном пространстве.

Микросхема 1887ВЕ6Т 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 6.1.

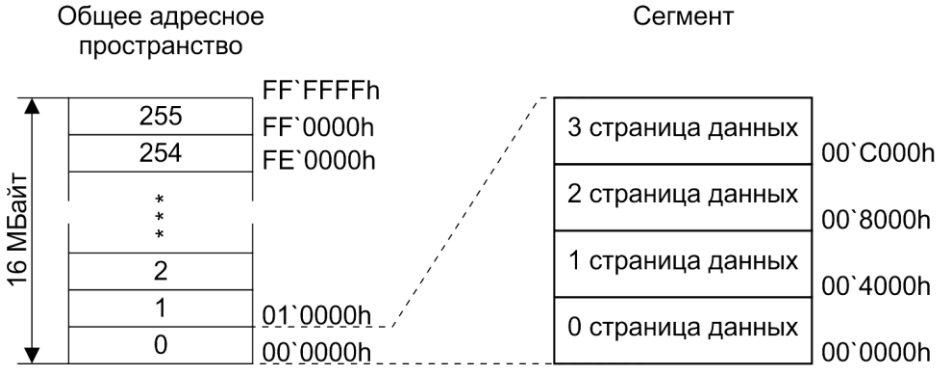


Рисунок 6.1 – Область памяти и адресное пространство

Большая часть внутренней памяти отражена в нулевом (системном) сегменте. Верхние 4 Кбайт нулевого сегмента, адреса 00’F000h – 00’FFFFh, отведены под SFR, ESFR и DPRAM. Младшие 54 Кбайт нулевого сегмента, адреса 00’0000h – 00’D7FFh, отведены под первую часть области внешней памяти, вторая часть которой занимает адреса больше 01’0000h. Область памяти в 2 Кбайт, отведенная под XSFR порта CAN, занимает адреса 00’E800h – 00’EFFFh. Область памяти, отведенная под XSFR блока CAPCOM6, занимает адреса 01’E800h – 01’E8FFh. Оперативная память данных XRAM объемом 4 Кбайт занимает адреса 00’D800h – 00’E7FFh. Область памяти XRAM и XSFR может быть назначена для доступа к внешней памяти.

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 6.1.

Таблица 6.1 – Распределение адресов памяти

Адресная область	Начальный адрес	Конечный адрес	Объем памяти
SFR	00’FE00h	00’FFFFh	512 байт
DPRAM	00’F600h	00’FDFFh	2 Кбайт
ESFR	00’F000h	00’F1FFh	512 байт
XRAM	00’D800h	00’E7FFh	4 Кбайт
XSFR (CAN)	00’E800h	00’EFFFh	2 Кбайт
XSFR (CAPCOM6)	01’E800h	01’E8FFh	256 байт

6.1 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 (неприсоединенных). Память и регистры хранят данные и команды в прямом порядке передачи байт (самые младшие байты в младших адресах). Последовательность байт отражена на рисунке 6.2. Нулевая позиция бита имеет наименьшее значение в байте с четным адресом, и позиция бита 15 имеет наибольшее значение в байте со следующим нечетным адресом. Битовая адресация поддерживается для части регистров SFR, части DPRAM и для регистров общего назначения GPR.

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

Рисунок 6.2 – Хранение слов, байт и битов памяти, организованных по байтовому принципу

Внутренняя область локальной памяти LM

Микроконтроллер 1887BE6T не имеет внутренней локальной памяти. 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. Всегда следует записывать «0» в этот бит.

Области памяти DPRAM, SFR, ESFR, XSFR, XRAM

В микроконтроллере 1887BE6T память разграничена на внутреннюю память данных DPRAM и внутреннюю память области периферии. Области DPRAM и SFR расположены на третьей странице данных и обеспечивают быстрый доступ с помощью указателя страницы данных DPP, см. рисунок 6.3.

Область DPRAM является оперативным запоминающим устройством для хранения:

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

Под DPRAM резервируется 2 Кбайт области памяти. Старшие 256 байт DPRAM (00'FD00h – 00'FDFFh) и текущий банк регистров области GPR предусматривают хранение единичных битов и являются побитно адресуемыми, см. рисунок 6.3. Любые слова и данные в DPRAM могут быть доступны с помощью косвенного или длинного 16-битного режима адресации, если выбранный регистр DPPx указывает на страницу данных 3. При обращении к коду всегда производится доступ к четным адресам байт. Для команд, состоящих из одного слова, наибольший возможный адрес обращения – FDFEh, для двухсловных команд – FDFCh. 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.

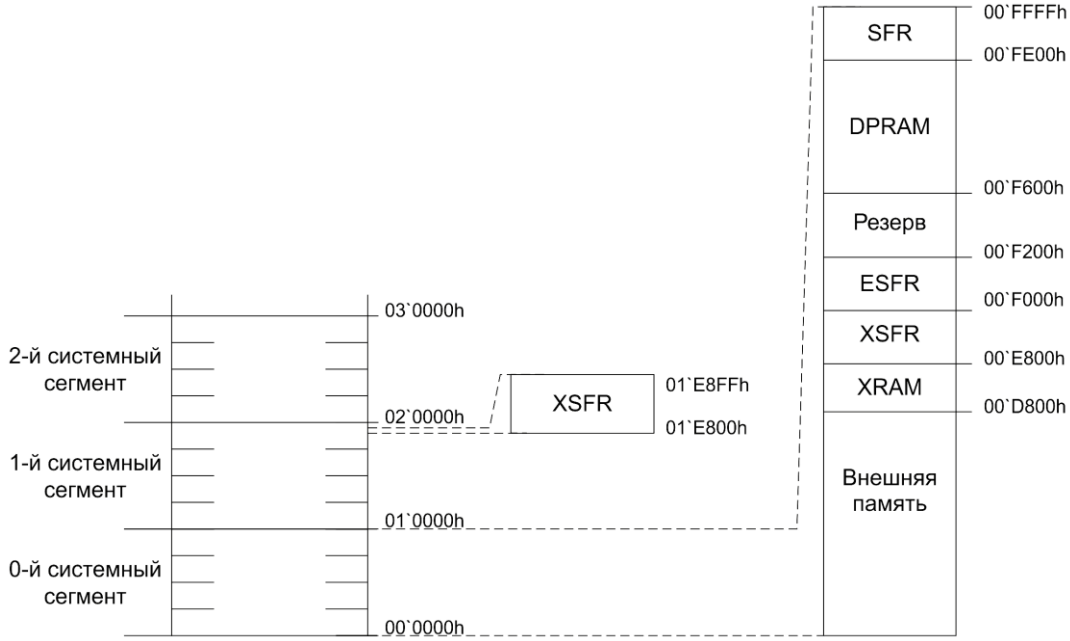


Рисунок 6.3 – Карта памяти микроконтроллера

Области SFR и ESFR (по 512 байт каждая) отведены под регистры специального назначения, которые управляют функциями ЦПУ, шинным интерфейсом, портами ввода-вывода и расположенной на кристалле периферии.

Область XSFR объемом 2 Кбайта (00'E800h – 00'EFFFh) и 256 байт (01'E800h – 01'E8FFh выделена под регистры специального назначения управления устройствами, подключенными к шине XBUS.

ОЗУ данных XRAM (объемом 4 Кбайт) предназначено для хранения данных, которое также может быть использовано для организации системного стека. Память XRAM связана с процессором 16-разрядной шиной данных.

Указатели адресов назначения и источников PEC

16 (24/32) слов, использующие восемь (12/16) каналов PEC в DPRAM от FCE0h (FCD0h/FCC0h) до FCFEh (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 байт или слов данных.

Более детальное описание об использовании SRCPx и DSTPx для передачи данных по каналам PEC приведено в разделе 7 «Система прерываний и ловушек».

6.2 Пространство внешней памяти

Доступно 16 Мбайт адресного пространства. Часть этого адресного пространства занимают внутренняя DPRAM и область 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 EBC осуществляется связь между ЦПУ микросхемы, внешней XBUS и внешним интерфейсом. Внешний шинный интерфейс осуществляет доступ к внешним периферийным устройствам и дополнительной энергозависимой или энергонезависимой памяти и может ограничивать количество адресуемой внешней памяти.

К внешнему слову данных можно обратиться только через косвенный или длинный 16-битный режим адресации, используя один из четырех DPPx регистров. Любой доступ к данным осуществляется по четному адресу. Внешняя память не адресуется побитно.

6.3 Пересечение границ памяти

Адресное пространство условно разделено на блоки одинакового размера, но разных форматов, и логические участки. Пересечение границ блоков (коды или данные) или участков требуют особого внимания, чтобы гарантировать, что контроллер выполняет желаемые операции.

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 1 024 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.

7 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 16 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 72 канала 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 канал 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 77 источников. Запросы могут быть вызваны периферийными устройствами или сигналами по внешним выводам. Прерывание «окончание PEC» обеспечивает расширение функциональных возможностей PEC и связано 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 IR. Запрос на прерывание также может быть вызван программно, если программа устанавливает соответствующий бит запроса на прерывание.

Если бит запроса установлен, и этот запрос на прерывание разрешен (установлен бит IE регистра xxIC), 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 7.1.

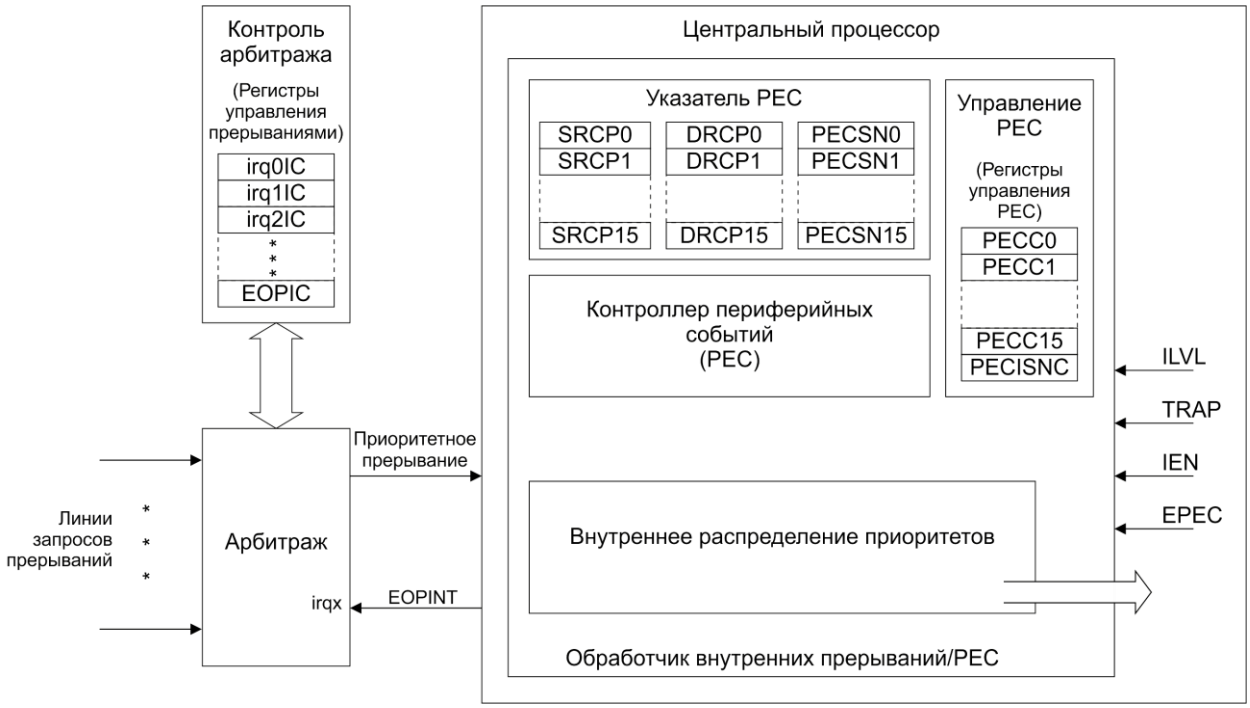


Рисунок 7.1 – Арбитраж прерываний

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 GP регистра xxIC.

Примечание – У всех источников прерываний, 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 (поле 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.

EOPINT соединяется с одной из линий запросов прерываний, таким образом для обработки доступны только 111 линий.

Уровень приоритета «0000» является значением по умолчанию центрального процессора. Поэтому запрос с уровнем «0000» не будет принят ЦПУ. Однако, каждый разрешенный запрос на прерывание, включая все запросы с уровнем «0000», вызовет выход ЦПУ из режима простоя Idle, независимо от состояния бита глобального разрешения прерываний IEN регистра PSW.

Все регистры управления прерываниями организованы идентично. Младшие восемь разрядов регистра управляют прерыванием и содержат информацию о состоянии источника прерывания, требующимся во время определения приоритета (арбитражный цикл). Старшие восемь разрядов регистра управления зарезервированы. Все регистры управления прерываниями являются бит-адресуемыми, 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. 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 GP, если задействовано меньше 64 узлов прерываний и меньше восьми 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 четыре и восемь слов.

Функции управления прерываниями регистра PSW

Регистр «слово состояния процессора PSW» функционально делится на две части. Младший байт PSW представляет арифметическое состояние ЦПУ, старший байт управляет системой прерываний контроллера и механизмом управления интерфейсом внешней шины.

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), 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 7.2. 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.

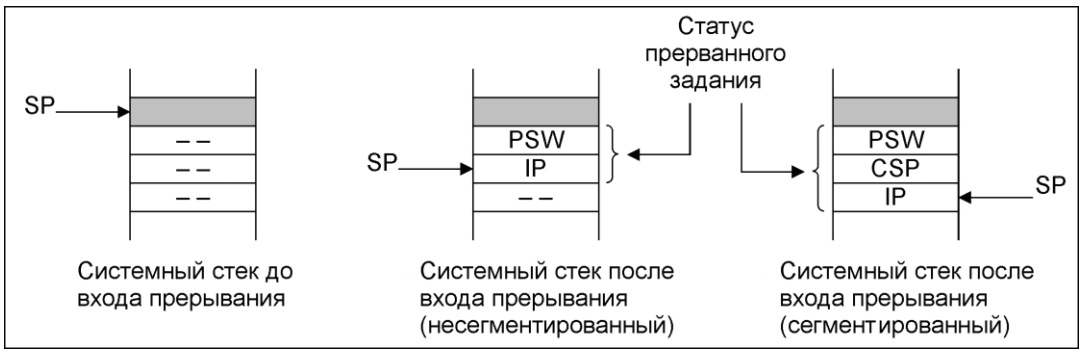


Рисунок 7.2 – Сохранение состояния в системном стеке

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 «Другой банк» отсылает содержимое контекстного указателя CP в системный стек и загружает в CP значение базового адреса «Другого банка». 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.

7.1 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 не маскируемы 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.

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.

Таблица 7.1 – Приоритет ловушек

Ловушка	Параметры ловушки			
	Флаг	Вектор	Номер	Приоритет
Функции RESET:				
Аппаратный сброс	–	RESET	00h	IV
Программный сброс	–	RESET	00h	IV
Переполнение сторожевого таймера	–	RESET	00h	IV
Ловушка отладки	DEBUG	DEBTAP	08h	III
Класс A аппаратных ловушек:				
Немаскируемое прерывание	NMI	NMITRAP	02h	II.3
Переполнение стека	STKOF	STOTRAP	04h	II.2
Очистка стека	STKUF	STUTRAP	06h	II.1
Программный разрыв	SOFTBRK	SDRKTRAP	08h	II.0
Класс B аппаратных ловушек:				
Неопределенный программный код	UNDOPC	BTRAP	0Ah	I
Ошибка защиты	PRTFLT	BTRAP	0Ah	I
Неверный доступ к операнду слова	ILLOPA	BTRAP	0Ah	I
Неверный командный доступ	ILLINA	BTRAP	0Ah	I
Неверный доступ к внешней шине	ILLBUS	BTRAP	0Ah	I

Ловушки класса A

Ловушка класса A активируется системными событиями NMI или специальными событиями ЦПУ, такими, как программный разрыв, переполнение стека или очистка стека. Ловушки класса A не используются для указания отказов аппаратных средств. Каждой ловушке класса A соответствует собственный вектор в таблице векторов. В случае одновременной активации нескольких ловушек класса A, 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 должно 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.

7.2 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 восемью 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 один 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 FFFFh в случае передачи байта BWT = 1b или FFFEh в случае передачи слова BWT = 0b, то следующее наращивание значения приведет к переполнению указателя адреса.

Примечание – Если для определенного канала PEC выбрана передача слова данных BWT = 0b, 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.

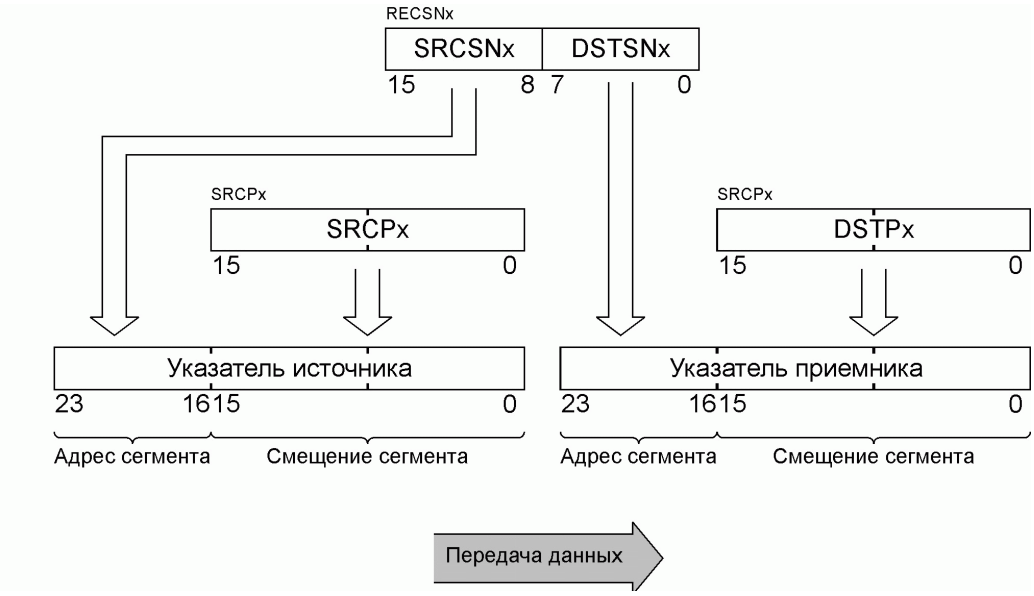


Рисунок 7.3 – Работа указателя адреса PEC (x – номер канала)

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.

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 = 00b), 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 = 0b, 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:

1 Если значение счетчика передач PEC установлено в «00h», PEC передачи запрещены, происходит нормальная обработка запросов на прерывание.

2 Если значение счетчика передач PEC установлено в «FFh», разрешено неограниченное число передач соответствующего канала.

3 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 00h, вырабатывается запрос на прерывание по окончанию PEC передач, имеющий тот же уровень приоритета, что и передача (если EOPINT = 0b) или другой уровень приоритета (EOPINT = 1b). В момент перехода содержимого счетчика COUNT передач из 01h в 00h, после того как передача данных будет завершена, флаг запроса очищается, если EOPINT будет установлен. Если EOPINT = 0b, то флаг запроса не очищается, и следующий запрос на прерывание будет обслужен с тем же уровнем приоритета. Если COUNT имеет значение «00h», 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 = 1b в регистре управления PECCx), то действиями PEC канала управляет поле COUNT2 в регистре PECXCx.

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 00h.

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 0000h, 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 = 0b), или с другим уровнем приоритета (EOPINT = 1b). Когда счетчик COUNT2 переходит из значения 0001H в 0000h, после окончания передачи, флаг запроса будет очищен, если EOPINT = 1b. Если EOPINT = 0b, то флаг запроса не очищается, а другой запрос на прерывание будет сформирован с тем же самым уровнем приоритета. Если значение счетчика COUNT2 равно 0000h, 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 = 0b или COUNT2 = 0b, в зависимости от режима) активного канала также равно нулю, то передача данных окончена. В этом случае выставляется запрос об окончании PEC передач.

В таблице 7.2 приведены данные о возможных соединениях каналов в пары и очередность запуска каналов внутри каждой пары.

Таблица 7.2 – 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 7.3 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 7.3 перечислены все возможные комбинации.

Таблица 7.3 – Уровни приоритета и значения соответствующих битов PLEV в регистре управления PECCx

Уровень приоритета		Выбранный PEC канал (x)			
Уровень приоритета ILVL	Уровень группового приоритета GP, GLVL	PLEV = 00b	PLEV = 01b	PLEV = 10b	PLEV = 11b
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.

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.

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, управляемые регистрами прерываний IRQx (см. приложение B), 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) могут стать сигналы с соответствующих выводов порта (стандартный вывод EXnIN или два альтернативных источника).

Выбор источников производится с помощью регистров EXISEL0 и EXISEL1. Помимо выбора одного из трех возможных источников, два или три из них могут быть логически объединены.

В следующей таблице показаны соответствия полей регистра EXISEL (т.е. сигналов прерываний) входам.

Таблица 7.4 – 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

8 Генератор тактовых сигналов

Всеми действиями аппаратных средств микроконтроллера и периферийных устройств управляют тактовые сигналы, генерируемые модулем генератора тактовых сигналов. Модулем формируется четыре тактовых сигнала:

- Fcpu – тактовый сигнал ядра контроллера, он же – Fper – тактовый сигнал периферийных модулей, подключенных к шинам PDBUS и 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 указанных выше тактовых сигналов (см. рисунок 8.1).

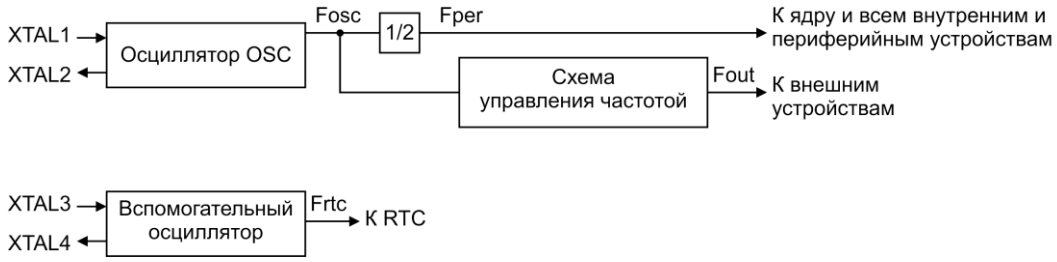


Рисунок 8.1 – Функциональная схема модуля генератора тактовых сигналов

8.1 Осцилляторы

Основной осциллятор

Осциллятор OSC 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 (см. рисунок 8.1). Осциллятор оптимизирован на работу для диапазона входных частот от 10 до 25 МГц.

При работе от внешнего генератора, тактовый сигнал подается на вход XTAL1. В этом случае частота входного тактового сигнала может находиться в диапазоне от 0 до 25 МГц. Максимальная частота входного сигнала ограничена максимальной тактовой частотой контроллера.

Осциллятор автоматически выключается при переходе контроллера в режим PowerDown. При этом модуль RTC продолжает функционировать.

Осциллятор OSC запускается во время системного сброса.

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 32,768 кГц. Такая узкая рабочая полоса частот позволила существенно снизить потребляемую мощность вспомогательного осциллятора.

Примечание – Значения емкостей внешних конденсаторов C1 и C2 обычно выбираются в диапазоне от 10 до 30 пФ, а номинал резистора R2 – от 3,5 до 5 МОм.

Вспомогательный осциллятор может работать от внешнего тактового сигнала частотой 32,768 кГц, который в этом случае подается на вход XTAL3.

Схемы подключения внешних источников синхросигналов показаны на рисунке 8.2.

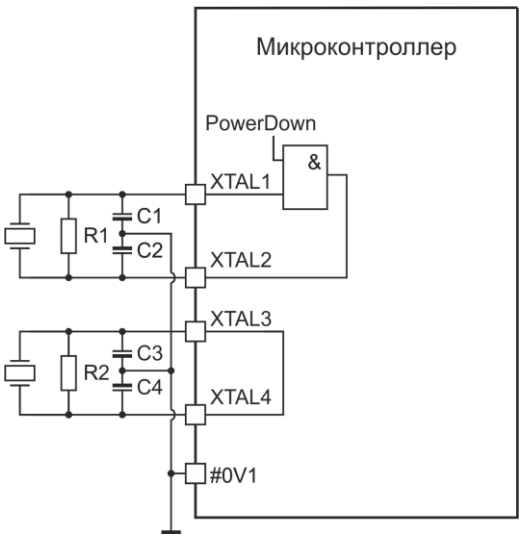


Рисунок 8.2 – Схема внешних источников синхросигналов

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 8.1 представлены домены тактовых сигналов и их состояние в различных режимах работы контроллера.

Таблица 8.1 – Домены тактовых сигналов

Домен тактового сигнала	Название тактового сигнала	Активный режим	Режим IDLE	Режим Power_Down	Подключенные модули
CPU	Fcpu	Включен	Выключен	Выключен	CPU, DPRAM
XBUS	Fper	Включено	Включено	Включено	CAN, CAPCOM6, XRAM
PDBUS	Fper	Включено	Включено	Включено	ASC0, ASC1, CAPCOM1, I2C, GPT1, GPT2, CAPCOM2, SSC0, SSC1, порты, RTC, WDT
RTC	Frtc	Включен	Включен	Включен/ выключен	RTC
Примечание – Модуль RTC относится к двум доменам тактовых сигналов – запись/чтение регистров производится по периферийному тактовому сигналу Fper, а счетчик реального времени и прескалер функционируют в домене медленного тактового сигнала Frtc.					

Генерация внешнего тактового сигнала

Для тактирования внешних устройств можно использовать тактовый сигнал, выводимый наружу через вывод P3.15. Могут быть выбраны два типа сигнала (см. альтернативные функции порта 3):

- CLKOUT, в качестве которого используется тактовый сигнал Fper;
- 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 8.3 представлена схема формирования выходного тактового сигнала.

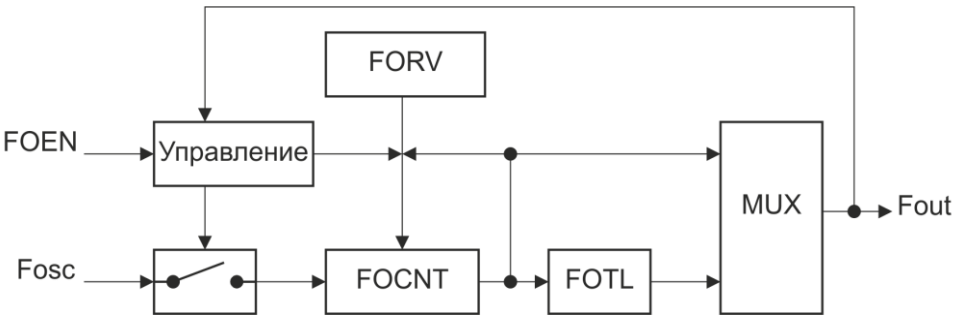


Рисунок 8.3 – Схема формирования выходного тактового сигнала

Контроль над формированием тактового сигнала Fout осуществляется с помощью регистра управления FOCON.

Генератор выходного сигнала Fout всегда обеспечивает формирование законченного периода тактового сигнала:

- когда Fout стартует (установка бит FOEN), поле FOCNT загружается значением FORV;
- когда Fout прекращается (сброс бита FOEN), FOCNT останавливается в момент перехода Fout в «0», см. рисунок 8.4.

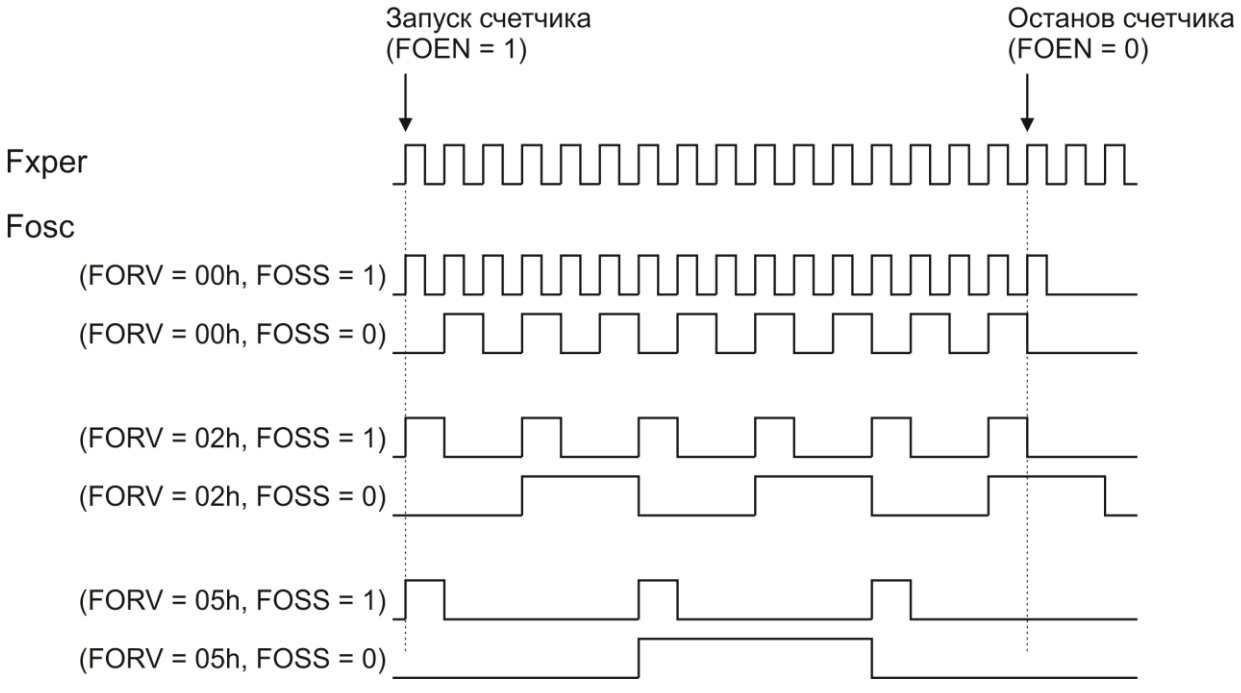


Рисунок 8.4 – Временная диаграмма формирования выходных сигналов

За вывод сигнала CLKOUT отвечают биты CLKEN (регистр SYSCON) и CLKOUT (регистр ALTSEL0P3). Для вывода сигнала Fout необходимо сбросить указанные биты и установить бит FOEN (регистр FOCON) и CLKOUT (регистр ALTSEL0P3).

Вычисление выходной частоты

Значение выходной частоты может быть вычислено по формуле:

$$fout = \frac{fosc}{(FORV + 1) \times 2^{(1 - FOSS)}} \quad (8.1)$$

В таблице 8.2 представлен список возможных выходных частот сигнала Fout.

Таблица 8.2 – Список возможных выходных частот сигнала Fout

fosc, МГц	fout, кГц, (для FORV = XXh, FOSS = 1/0)					FORV для fout = 1 МГц	
	00h	01h	02h	3Eh	3Fh	FOSS = 0b	FOSS = 1b
4	4 000 2 000	2 000 1 000	1 333,33 666,67	63,492 31,746	62,500 31,250	01h	03h
10	10 000 5 000	5 000 2 500	3 333,33 1 666,67	158,730 79,365	156,250 78,125	04h	09h
12	12 000 6 000	6 000 3 000	4 000,00 2 000,00	190,476 95,238	187,500 93,750	05h	0Bh
16	16 000 8 000	8 000 4 000	5 333,33 2 666,67	253,968 126,984	250,000 125,000	07h	0Fh
20	20 000 10 000	10 000 5 000	6 666,67 333,33	317,460 158,730	312,500 156,250	09h	13h
25	25 000 12 500	12 500 6 250	8 333,33 4 166,67	396,825 198,413	390,265 195,313	0Bh (1,04167) 0Ch 0,96154	18h

9 Блок сторожевого таймера

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.

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.

Текущее значение счета сторожевого таймера хранится в регистре таймера (не адресуемый 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 fper шины PDBUS, деленной на выбранный регистром WDTCON делитель.

Функционирование сторожевого таймера

После любого сброса сторожевой таймер начинает считать от значения 0000h с частотой fwdt, равной по умолчанию fper/256. Значение частоты может быть изменено программно.

Режим WDT-сброса

Если сторожевой таймер не блокируется командой DISWDT, он продолжает счет, даже во время режима Idle. Если таймер достигает значения FFFFh (и нет обращения к таймеру посредством команды SRVWDT), он переполняется, что приводит к генерированию WDT-сброса, установке флага WDTR и генерированию 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 = 0b (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) и перезагрузкой счетчика значением WDTREL регистра WDTCON.

Период счета Pwdt от перезагрузки счетчика до переполнения может быть вычислен:

$$Pwdt = \frac{2^{(1 + <WDTPRE> + <WDTIN> \times 6)} \times (2^{16} - <WDTREL> \times 2^8)}{fper} \qquad (9.1)$$

Примечание – Желательно перезаписывать содержимое WDTCON каждый раз перед началом обращения к сторожевому таймеру.

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.

10 Внутренняя и внешняя шины

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. XBUS является внутренним представлением интерфейса внешней шины, позволяющим интегрировать периферийные устройства без изменения внутренней архитектуры микроконтроллера.

10.1 Внутренняя шина XBUS

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 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.

Микропроцессор использует биты с 5 по 0. Может быть подключено шесть устройств периферии. Если в бите «1», устройство разрешено, иначе «0» – запрещено. Каждый бит PERx включает сигнал выбора периферийных устройств XCSx.

В микроконтроллере используются четыре внутренних сигнала выбора устройств:

- XCS1 по биту PER0 для интерфейса CAN;
- XCS2 по биту PER1 для первых 4 Кбайт XRAM;
- XCS3 по биту PER2 для вторых 4 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.

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.

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 = 1b. 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 = 1b), то такое обращение обладает самым низшим приоритетом – приоритетом 4. При этом текущий адрес (полный или часть) выводится на внешнюю шину, конфигурация которой определяется регистром BUSCON0, и формируется сигнал выборки CS0#.

10.2 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 порта 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).

Режим работы без внешних устройств

Микроконтроллер 1887BE6T не может выполнять свои функции в этом режиме, так как не имеет встроенной памяти типа Flash.

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.

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 01b 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 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.

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) 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 10.1.

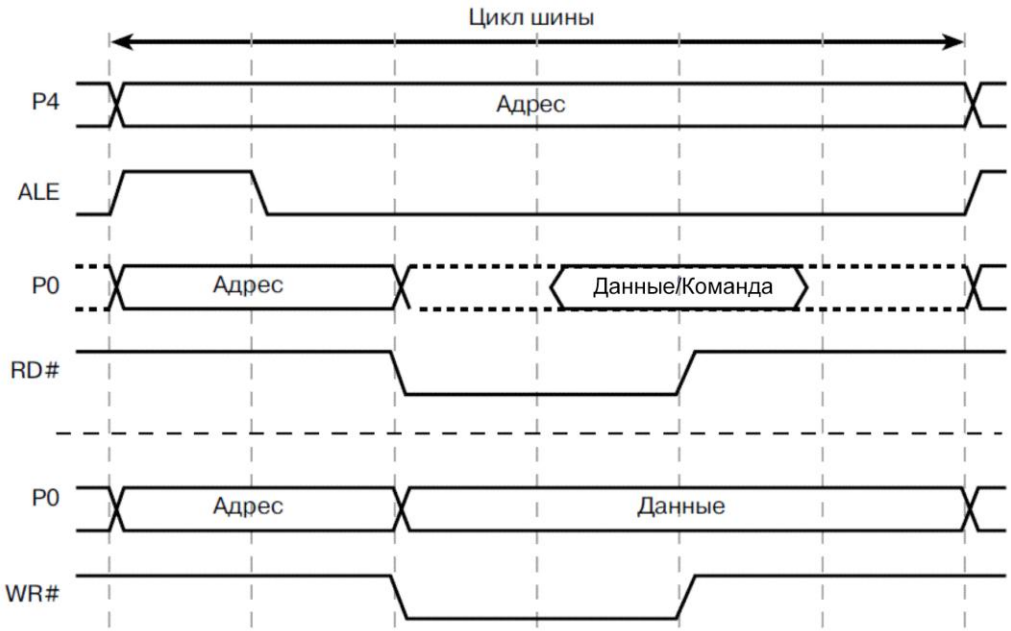


Рисунок 10.1 – Цикл мультиплексной шины

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 выдаются на внешнюю шину через порт P4, см. рисунок 10.2.

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.

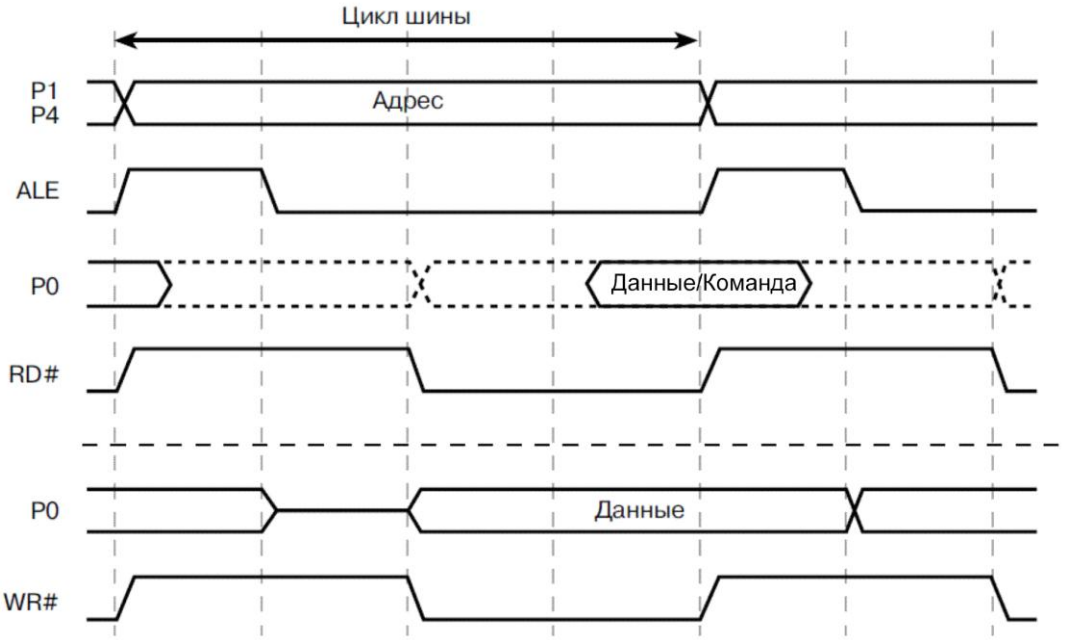


Рисунок 10.2 – Цикл демультиплексной шины

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.

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 10.3. 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 10.3. Вставка дополнительного машинного цикла контролируется битом BSWCx в регистре BUSCONx для адресного окна, из которого производится переключение в другое адресное окно. Разрешение вставки BSWCx = 1b не влияет на быстродействие, если производится последовательное обращение к одному и тому же адресному окну.

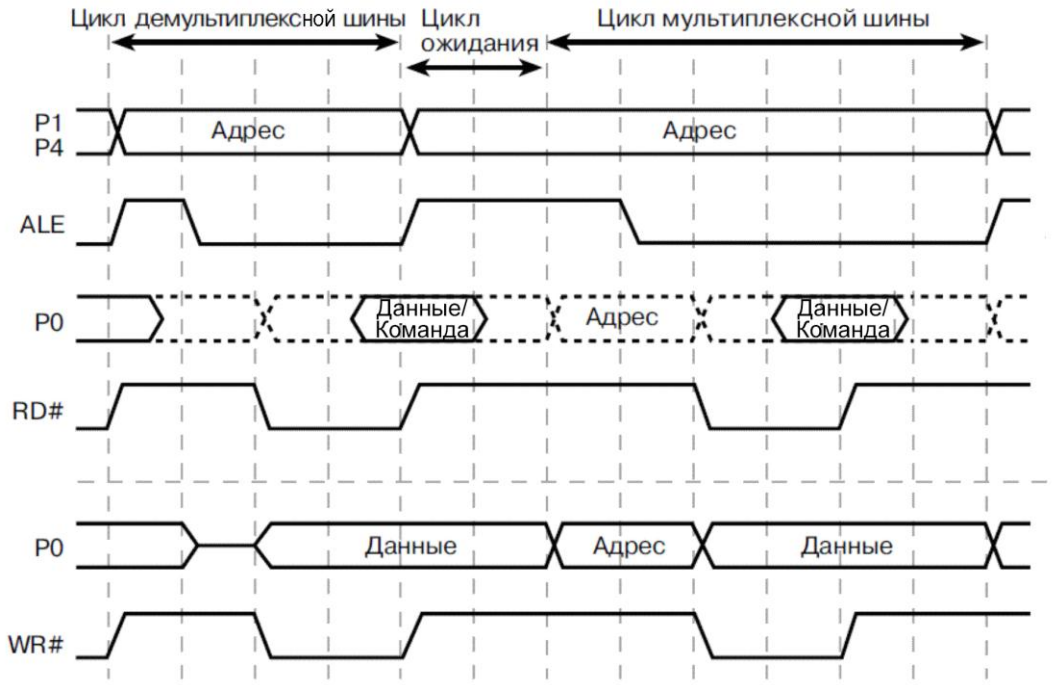


Рисунок 10.3 – Переключение от демультиплексной шины к мультиплексной

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 байт. 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 10.1 приведен коэффициент увеличения времени доступа по отношению ко времени обращения по 16-разрядной демультиплексной шине к 16-разрядным данным.

Таблица 10.1 – Коэффициент увеличения 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	–

Использование сигнала BHE#

Сигнал BHE# автоматически разрешается (BYTDIS = 0b) 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 единице, 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), если выбрана 16-битная шина данных. Иначе – запрещена (BYTDIS = 1). 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 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.

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).

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. Количество используемых каналов CSx# устанавливается во время сброса и задается битовым полем CSSEL регистра RP0h.

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.

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 CSCFG регистра SYSCON. При CSCFG = 0 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.

Сигнал CSx# не изменяется, если адрес находится в пределах собственного окна или во внутренней памяти (исключая устройства, подключенные к XBUS). При CSCFG = 1 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.

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, 256 Кбайт, 1 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.

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:

1 Длительность сигнала ALE и время удержания адреса на шине после отрицательного фронта ALE.

2 Длительность циклов памяти (с расширением от 1 до 15 TCL) определяет необходимое время доступа.

3 Длительность времени высокоимпедансного состояния на шине (с расширением на 1 TCL).

4 Задержка сигналов чтения и записи после отрицательного фронта ALE.

5 Использование сигнала готовности внешнего устройства READY#.

Программируемые интервалы цикла внешней шины приведены на рисунке 10.4.

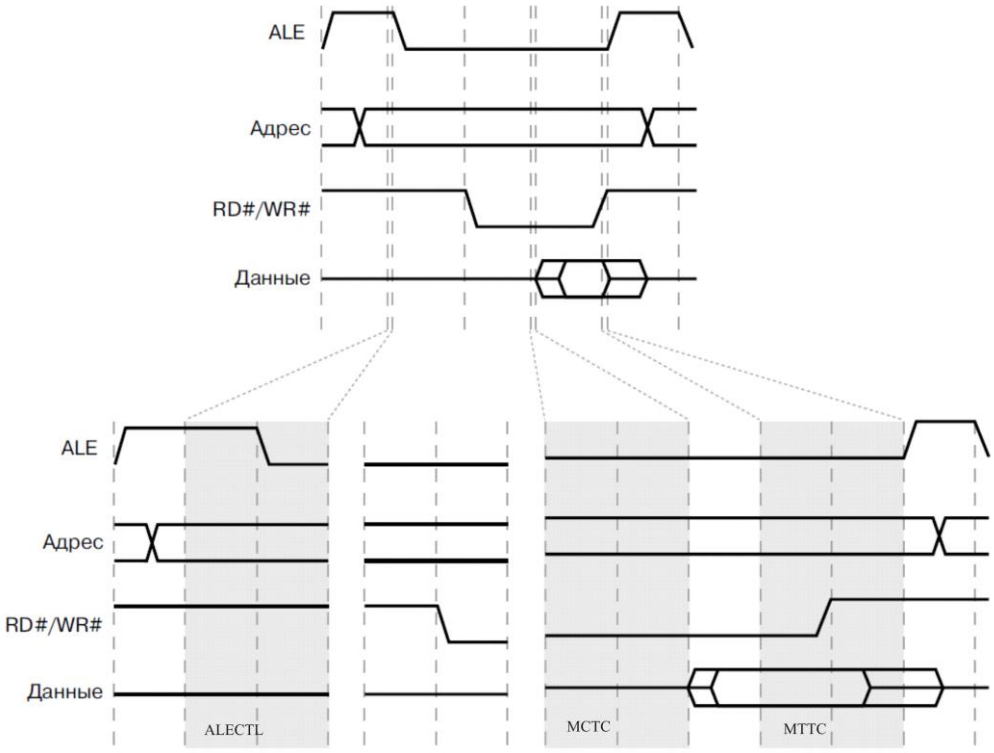


Рисунок 10.4 – Программируемые временные параметры циклов шины

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 ALECTLx установлен в 1, длительность сигнала ALE увеличивается на половину периода сигнала тактового CPU (1TCL). Время удержания адреса для мультиплексной шины при ALECTLx = 1b 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 10.5. 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 10.5.

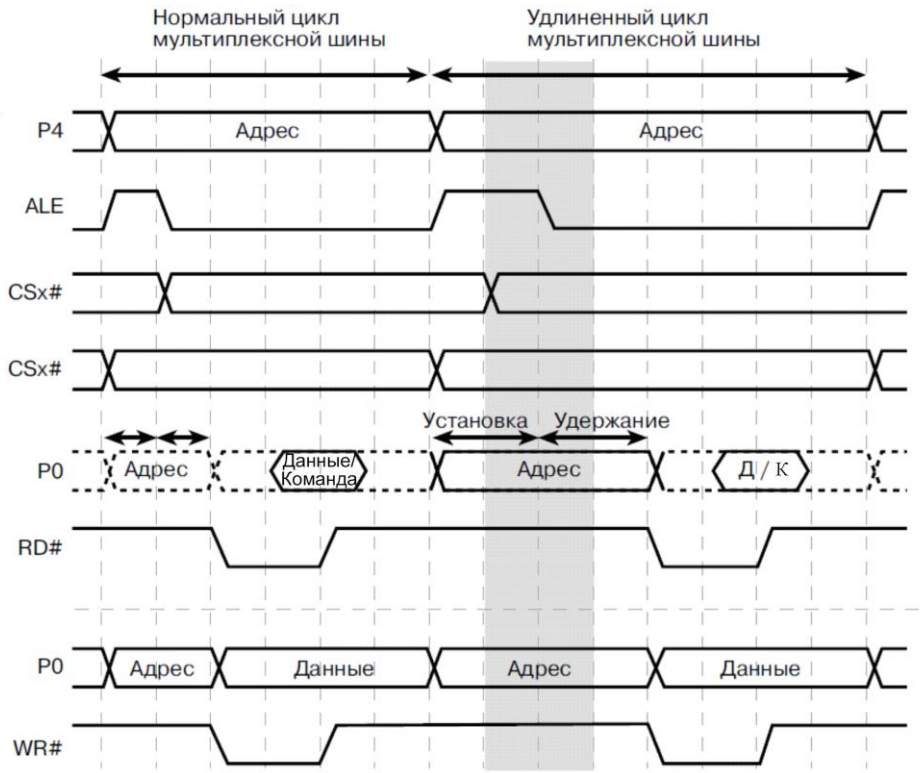


Рисунок 10.5 – Цикл сигнала ALE

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 10.6. 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 0h, соответствует максимальной задержке в 15 тактов. Если MCTC равно Fh, 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.

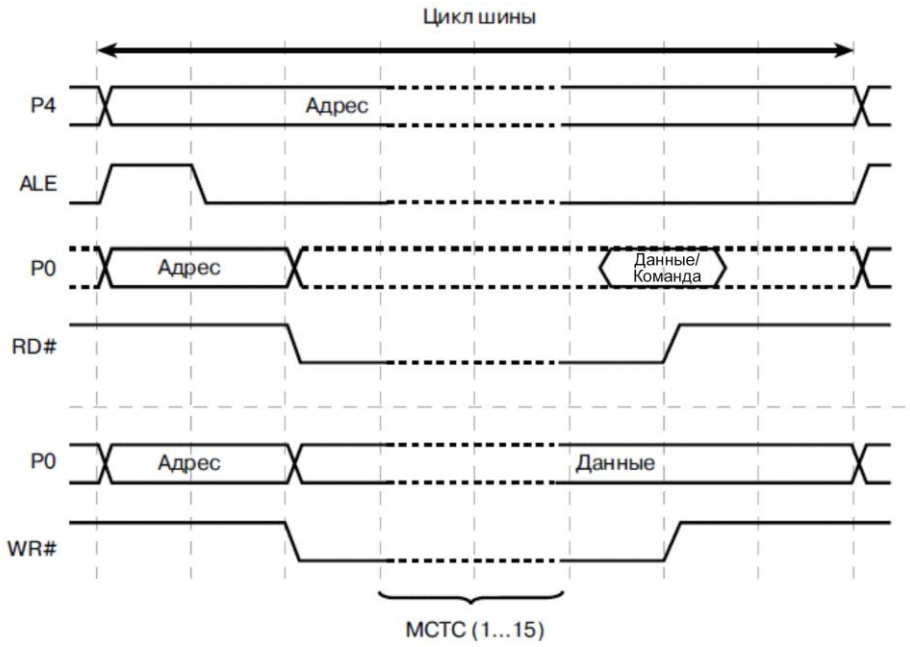


Рисунок 10.6 – Длительность циклов чтения и записи

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 MTTC регистра BUSCONx.

Как показано на рисунке 10.7, циклы чтения и записи будут завершаться с дополнительной задержкой, если MTTC = 0 – значение по умолчанию после сброса микроконтроллера.

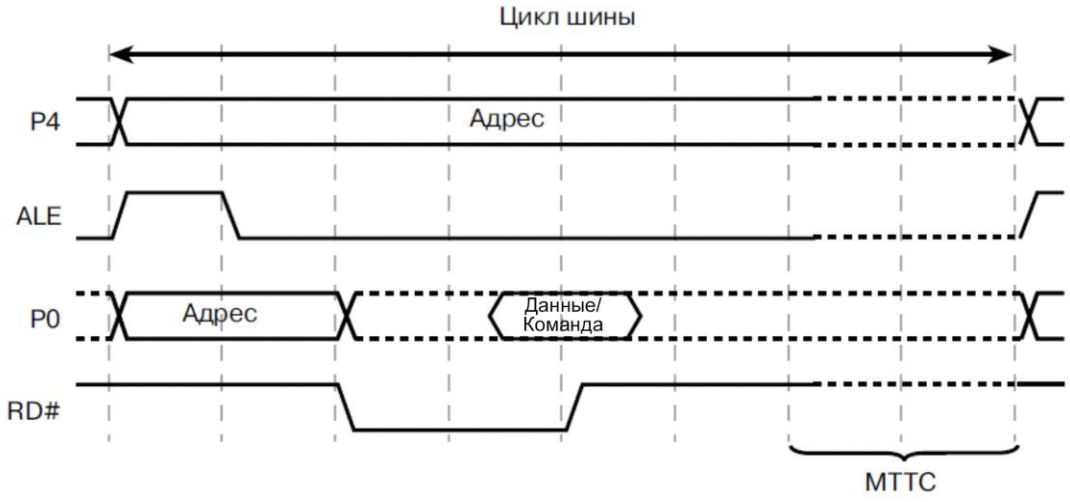


Рисунок 10.7 – Время задержки после чтения

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 10.8.

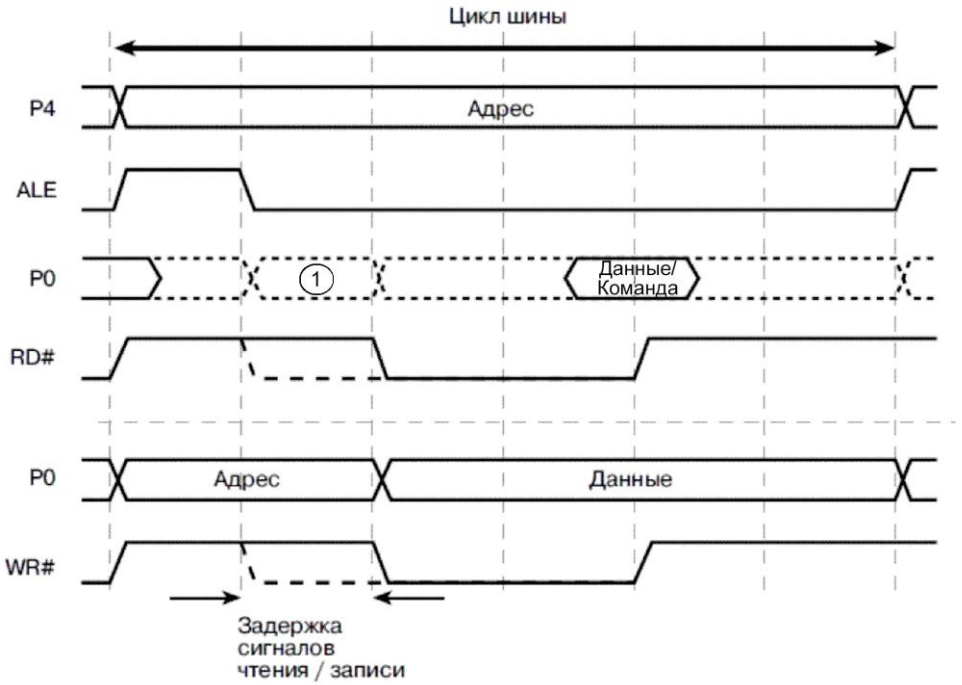


Рисунок 10.8 – Задержка сигналов чтения /записи

На рисунке 10.8 в промежутке времени (1) шинные буферы внешнего устройства должны отключаться от шины до начала активного уровня сигнала чтения RD#.

Настройки контроллера внешней шины EBC позволяют задать параметры диаграмм обращения чтения/записи с учетом временных характеристик внешней периферии, см. рисунок 10.8. 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 = 1b 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, что показано на стандартной временной диаграммой, см. рисунок 10.9. 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 EWEN регистров BUSCON. Сигнал WR# будет укорочен, если EWEN = 1. После старта микроконтроллера EWEN = 0 и сигнал WR# работает в стандартном режиме.

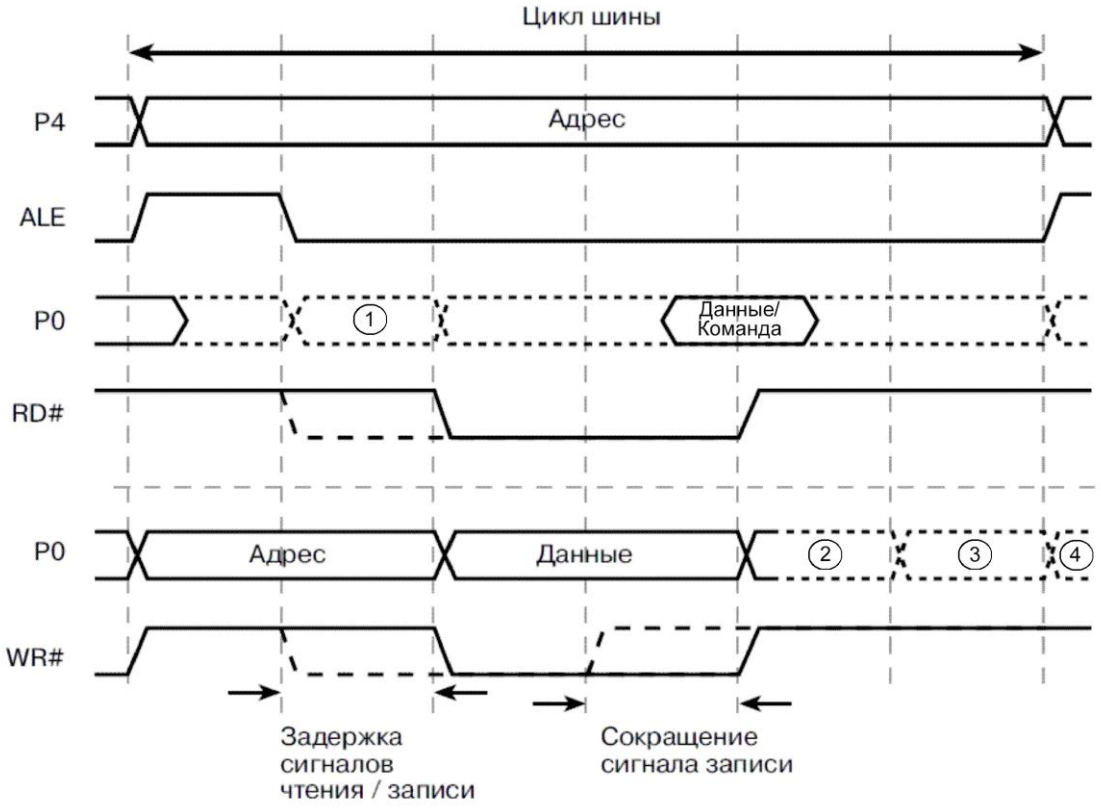


Рисунок 10.9 – Завершение формирования сигнала записи

На рисунке 10.9 для временных интервалов (1), (2), (3) и (4) предусмотрены действия:

(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.

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, RDYEN = 1, только соответствующие младшие три бита MCTC определяют число вставленных циклов ожидания от 0 до 7 TCL. Как показано на рисунке 10.10, 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 10.10, а для доступа к периферии переход в активное состояние может быть задержан – цикл шины с поздним READY#, см. рисунок 10.10. Обычно внешняя память является более быстрым устройством, чем периферия, поэтому использование сигнала READY# не должно оказывать влияния на скорость выполнения программы.

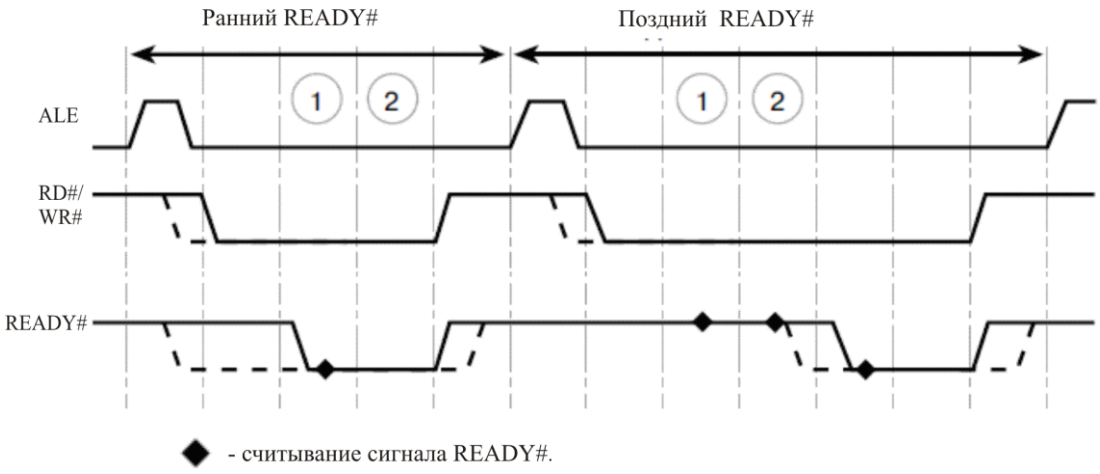


Рисунок 10.10 – Циклы шины с использованием сигнала READY#

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 (отмечено цифрами 1 и 2), смотри рисунок 10.10.

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 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 BUSACT, 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.

Регистр ADDRSEL0 отсутствует, так как регистр BUSCON0 управляет всеми внешними доступами во всем адресном пространстве, кроме области, занимаемой четырьмя адресными окнами для BUSCON1, …, BUSCON4.

11 Параллельные порты ввода-вывода

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 (P0, P1, P3), один 16-разрядный входной порт (P5), четыре 8-разрядных порта ввода-вывода (P2, P4, P6, P9) и один 7-разрядный порт ввода-вывода (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.

Если вывод сконфигурирован как выход, то выходной триггер и вывод микросхемы соединены (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.

Направление работы порта задается соответствующим регистром DPx.

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.

Режим с открытым стоком

В микроконтроллере 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 11.1 показана реализация этой функции. Использование этой возможности позволит объединять по несколько портов вместе для обеспечения конфигурации монтажное И, т. е. без дополнительного программного кода можно осуществлять операции разрешения/запрещения выходного сигнала.

Эта функция управляется с помощью специальных регистров управления режимом с открытым стоком ODPx.

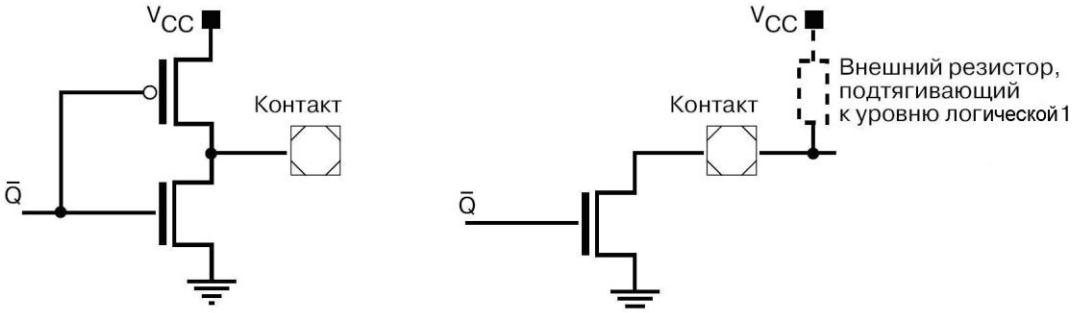


Рисунок 11.1 – Выходные буферы в режиме push/pull и в режиме с открытым стоком соответственно (сигнал $\overline{Q}$ – сигнал, приходящий с выхода триггера порта)

Альтернативные функции портов

При использовании альтернативной функции вывода порта для вывода данных, порт должен быть сконфигурирован как выход (посредством регистра DPx), 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 записывать единицу, потому что он объединяется с данными альтернативного вывода по функции логического И.

Если используется альтернативная функция ввода, то вывод следует сконфигурировать как вход. 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 конфигурацию отвечает программа пользователя. Для некоторых линий порта направление устанавливается автоматически. 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 аппаратно.

Программное управление альтернативными функциями порта обеспечивается специальными регистрами – основным ALTSEL0Px и дополнительным ALTSEL1Px (x – номер порта).

Примечание – Порт P0 не имеет регистров альтернативных функций. Порты P1, P2, P4, P6 имеют только по одному – основному – регистру альтернативных функций. Порты P3, P7, P9 имеют по два (основной и дополнительный) регистра альтернативных функций.

Выводы портов, альтернативные функции которых не включены, могут использоваться как выводы общего назначения. При этом для включения выходных драйверов необходимо записать инициализирующие значения в соответствующие регистры портов, во избежание нежелательных передач на выводы.

```
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.

11.1 Порт P0

Два 8-разрядных порта P0H и P0L 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 DP0H и DP0L.

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 необходимы два цикла для доступа к слову, сначала для доступа к младшему байту и потом для доступа к старшему байту слова.

Порт P0 также используется для выбора конфигурации системы при старте, см. приложение Ж. Во время сброса порт P0 настроен на вход, и через внутренний «pullup» на каждой линии выставлен высокий уровень. Каждая линия может быть индивидуально подключена к нулевому потенциалу, с помощью внешнего «pulldown». Выставляя необходимые линии на низкий уровень напряжения, можно изменять установочные значения.

Альтернативные функции представлены в таблице 11.1.

Таблица 11.1 – Выводы порта P0 и их альтернативные функции

Вывод порта	Альтернативные функции выводов			
	Демультиплексированная внешняя шина		Мультиплексированная внешняя шина	
	8-разрядная	16-разрядная	8-разрядная	16-разрядная
P0H.7	Отсутствует	D15	A15	AD15
P0H.6		D14	A14	AD14
P0H.5		D13	A13	AD13
P0H.4		D12	A12	AD12
P0H.3		D11	A11	AD11
P0H.2		D10	A10	AD10
P0H.1		D9	A9	AD9
P0H.0		D8	A8	AD8
P0L.7	D7	D7	AD7	AD7
P0L.6	D6	D6	AD6	AD6
P0L.5	D5	D5	AD5	AD5
P0L.4	D4	D4	AD4	AD4
P0L.3	D3	D3	AD3	AD3
P0L.2	D2	D2	AD2	AD2
P0L.1	D1	D1	AD1	AD1
P0L.0	D0	D0	AD0	AD0

Внутренний «pullup» разработан 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 альтернативного вывода данных, с помощью мультиплексора. Альтернативные данные могут быть или 16-разрядным внутрисегментным адресом, или 8/16-разрядными данными. Входные данные порта P0 читаются на линии альтернативных входных данных. В то время, когда включен режим внешней шины, не должна производиться запись в выходной триггер порта программой пользователя, поскольку это может привести к непредсказуемому результату. Когда внешняя шина отключена, вступает в силу последнее записанное пользователем содержимое регистра направления. На рисунке 11.2 показана структура вывода порта P0.

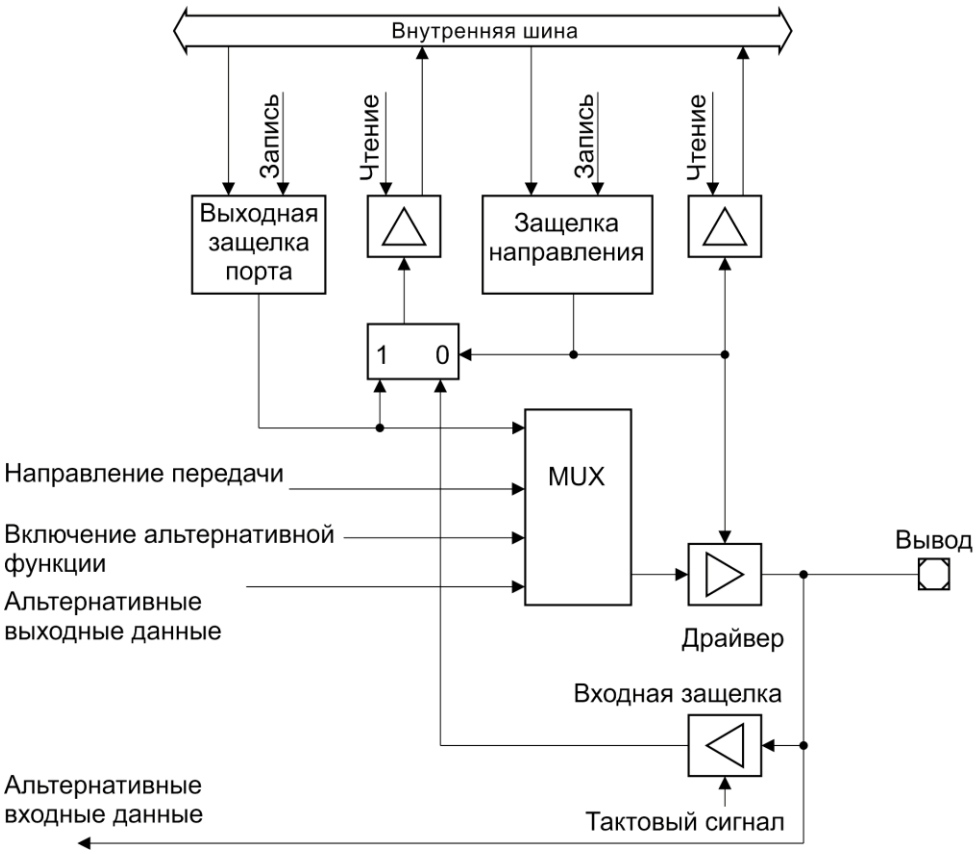


Рисунок 11.2 – Структурная схема вывода порта P0

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 импеданса. 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 на выводе P0L.1 следует удерживать высокий уровень сигнала. По умолчанию режим Adapt отключен.

11.2 Порт P1

Два 8-разрядных порта P1H и P1L 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 общего назначения, направление каждой линии может быть сконфигурировано с помощью регистров направления DP1H и DP1L.

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 как выводы ввода-вывода общего назначения.

Выводы порта P1 и их альтернативные функции приведены в таблицах 11.2 и 11.3 структурная схема вывода порта P1 представлена на рисунке 11.3. Для управления альтернативными функциями используются регистры ALTSEL0P1H и ALTSEL0P1L.

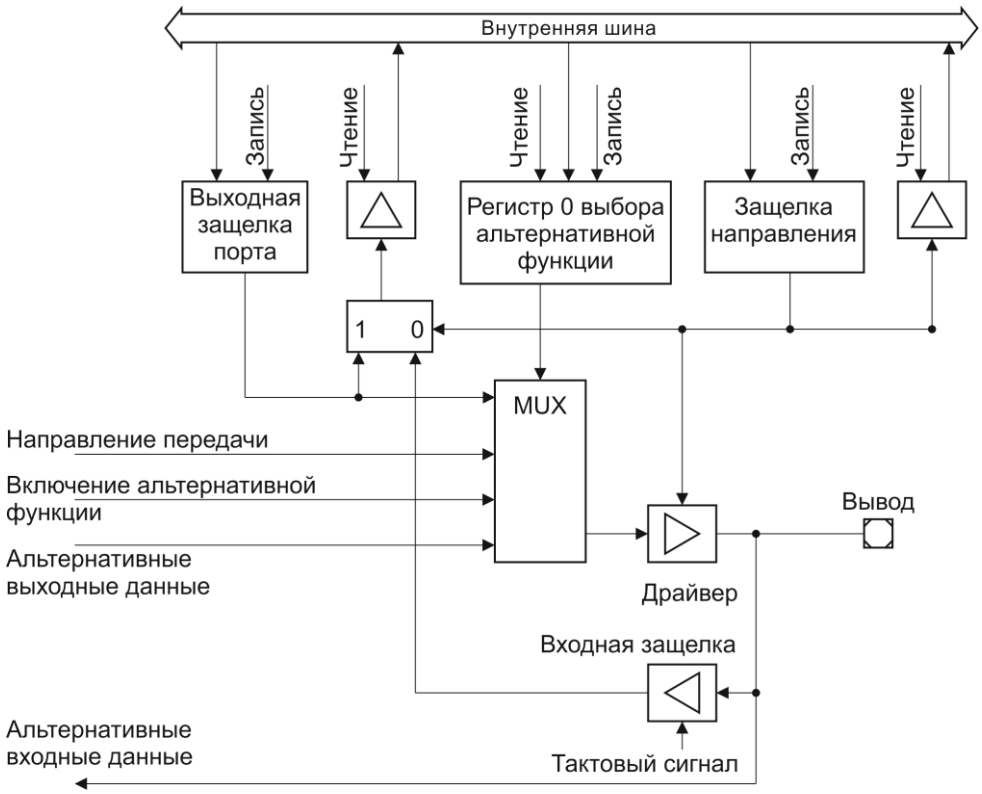


Рисунок 11.3 – Структурная схема вывода порта P1

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.

Таблица 11.2 – Выводы старшего слова порта P1 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние EBC	Состояние соответствующих битов регистров управления	
			ALTSEL0P1H	DP1H
P1H.7 – P1H.4			P7 = 0, P6 = 0, P5 = 0, P4 = 0	P7 = 0/1, P6 = 0/1, P5 = 0/1, P4 = 0/1
	A15 – A12	Активен		
	CC27IO – CC24IO (входы)			P7 = 0, P6 = 0, P5 = 0, P4 = 0
	CC27IO – CC24IO (выходы)		P7 = 1, P6 = 1, P5 = 1, P4 = 1	P7 = 1, P6 = 1, P5 = 1, P4 = 1
P1H.3			P3 = 0	P3 = 0/1
	A11	Активен		
	SCLK1 (вход)			P3 = 0
	SCLK1 (выход)		P3 = 1	P3 = 1
	EX0INA			P3 = 0
P1H.2			P2 = 0	P2 = 0/1
	A10	Активен		
	MTSR1 (вход)			P2 = 0
	MTSR1 (выход)		P2 = 1	P2 = 1
	CC6POS2#			P2 = 0
P1H.1			P1 = 0	P1 = 0/1
	A9	Активен		
	MRST1 (вход)			P1 = 0
	MRST1 (выход)		P1 = 1	P1 = 1
	CC6POS1#			P1 = 0
P1H.0			P0 = 0	P0 = 0/1
	A8	Активен		
	CC23IO (вход)			P0 = 0
	CC23IO (выход)		P0 = 1	P0 = 1
	CC6POS0#			P0 = 0
	EX0INB			P0 = 0

Таблица 11.3 – Выводы младшего слова порта P1 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние EBC	Состояние соответствующих битов регистров управления	
			ALTSEL0P1L	DP1L
1	2	3	4	5
P1L.7			P7 = 0	P7 = 0/1
	A7	Активен		
	CC22IO (вход)			P7 = 0
	CC22IO (выход)		P7 = 1	P7 = 1
	CTRAP#			P7 = 0
P1L.6			P6 = 0	P6 = 0/1
	A6	Активен		
	COUT63		P6 = 1	P6 = 1
P1L.5			P5 = 0	P5 = 0/1
	A5	Активен		
	COUT62		P5 = 1	P5 = 1

Окончание таблицы 11.3

1	2	3	4	5
P1L.4			P4 = 0	P4 = 0/1
	A4	Активен		
	CC62 (вход)			P4 = 0
	CC62 (выход)		P4 = 1	P4 = 1
P1L.3			P3 = 0	P3 = 0/1
	A3	Активен		
	COUT61		P3 = 1	P3 = 1
P1L.2			P2 = 0	P2 = 0/1
	A2	Активен		
	CC61 (вход)			P2 = 0
	CC61 (выход)		P2 = 1	P2 = 1
P1L.1			P1 = 0	P1 = 0/1
	A1	Активен		
	COUT60		P0 = 1	P0 = 1
P1L.0			P0 = 0	P0 = 0/1
	A0	Активен		
	CC60 (вход)			P0 = 0
	CC60 (выход)		P0 = 1	P0 = 1

11.3 Порт P2

8-разрядный порт. Используется для ввода и вывода общего назначения и выполнения альтернативных функций. Направление работы может быть сконфигурировано с помощью регистра направления DP2. Каждая линия порта P2 может быть переключена в режим «push/pull» или в режим с открытым стоком, с помощью регистра управления открытым стоком ODP2.

Альтернативные функции порта P2

Альтернативные функции порта P2 управляются регистром ALTSEL0P2 и представлены в таблице 11.4. Структурная схема вывода порта P2 представлена на рисунке 11.4.

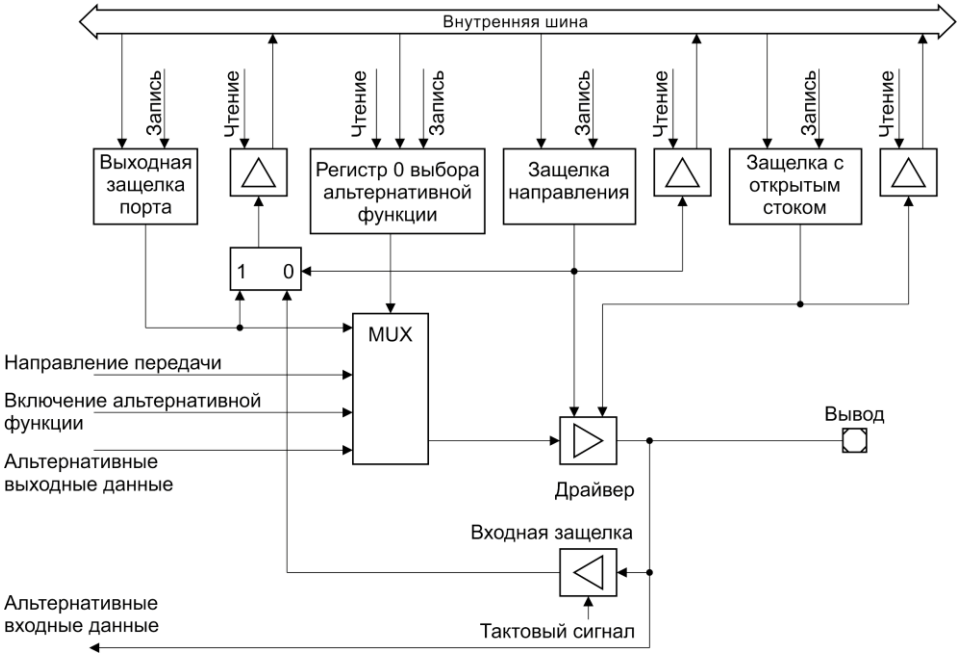


Рисунок 11.4 – Структурная схема вывода порта P2

Таблица 11.4 – Выводы порта P2 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние соответствующих битов регистров управления	
		ALTSEL0P2	DP2
P2.15		P15 = 0	P15 = 0/1
	CC15IO (вход)		P15 = 0
	CC15IO (выход)	P15 = 1	P15 = 1
	EX7IN		P15 = 0
	T7IN		P15 = 0
P2.14 – P2.8		P14 = 0, P13 = 0, P12 = 0, P11 = 0, P10 = 0, P9 = 0, P8 = 0	P14 = 0/1, P13 = 0/1, P12 = 0/1, P11 = 0/1, P10 = 0/1, P9 = 0/1, P8 = 0/1
	CC14IO – CC8IO (входы)		P14 = 0, P13 = 0, P12 = 0, P11 = 0, P10 = 0, P9 = 0, P8 = 0
	CC14IO – CC8IO (выходы)	P14 = 1, P13 = 1, P12 = 1, P11 = 1, P10 = 1, P9 = 1, P8 = 1	P14 = 1, P13 = 1, P12 = 1, P11 = 1, P10 = 1, P9 = 1, P8 = 1
	EX6IN – EX0IN		P14 = 0, P13 = 0, P12 = 0, P11 = 0, P10 = 0, P9 = 0, P8 = 0

Когда линии порта P2 используются в качестве входов захвата, входные триггеры напрямую подключены к блоку CAPCOM1. При этом выводы порта должны быть сконфигурированы как входы.

Если порт используется для вывода сигналов, то будет считываться состояние выходных триггеров порта, что может использоваться для программного управления переключением выводов. Когда выводы порта используются в качестве выходов сигналов сравнения блока CAPCOM1, то сравниваемое событие напрямую влияет на состояние выходного триггера порта.

Следует помнить, что к выводам порта, сконфигурированным как выходы, не должно быть подключено никаких внешних устройств, которые могут подавать сигналы, чтобы избежать конфликта сигналов.

Дополнительно все выводы порта являются входами внешних прерываний.

11.4 Порт P3

16-разрядный порт. Используется для ввода и вывода общего назначения и выполнения альтернативных функций. Направление работы каждого вывода может быть задано с помощью регистра направления DP3. Большинство линий порта может быть переключено в режим «push/pull» или режим с открытым стоком с помощью регистра управления открытым стоком ODP3. Выводы P3.12, P3.14 и P3.15 не поддерживают режим с открытым стоком!

Альтернативные функции порта P3

Выводы порта P3 выполняют различные функций, в том числе передачу сигналов BHE#/WRH# и CLKOUT/Fout. Альтернативные функции порта управляются регистрами ALTSEL0P3 и ALTSEL1P3 и приведены в таблице 11.5, структурная схема вывода порта показана на рисунке 11.5.

Если вывод сконфигурирован как вход, то состояние вывода может быть прочитано из триггера порта. Если вывод сконфигурирован как выход и включена альтернативная

функция, то следует помнить, что выходной альтернативный сигнал и выход триггера порта объединены логически по И.

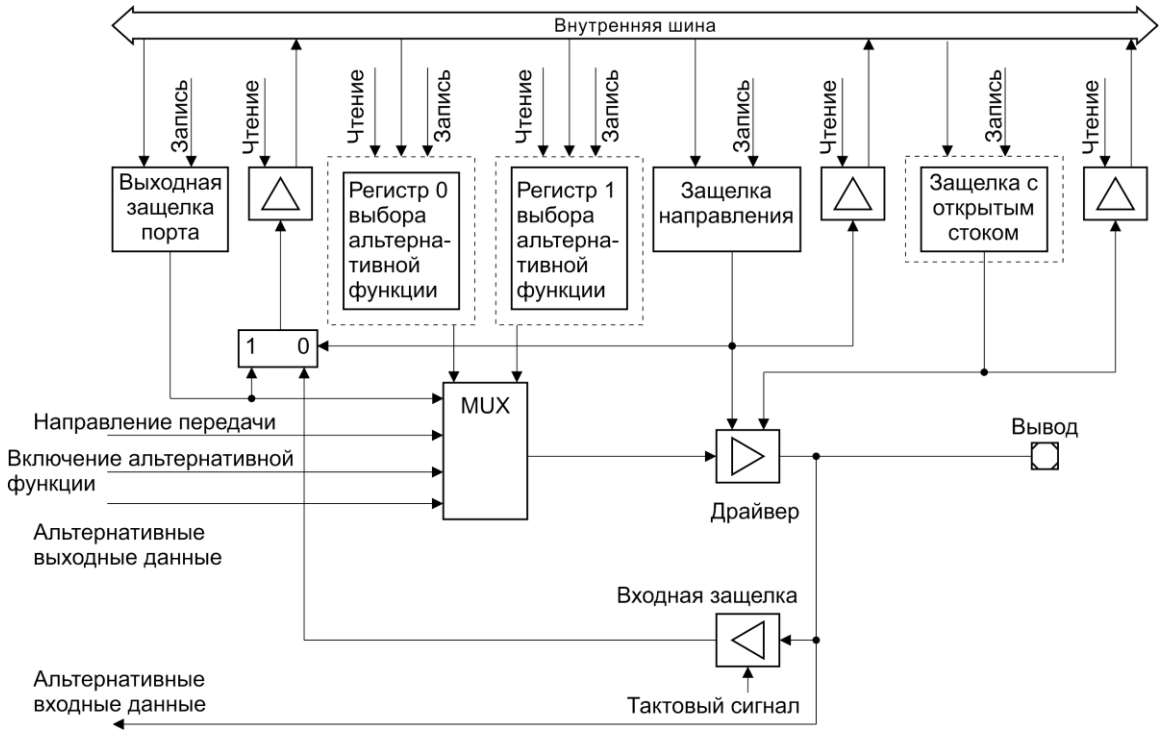


Рисунок 11.5 – Структурная схема вывода порта P3

Разрешение функции CLKOUT/Fout автоматически включает выходной драйвер для вывода P3.15, поэтому запись единицы в бит 15 регистра DP3 не требуется.

Таблица 11.5 – Выводы порта P3 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние соответствующих битов регистров управления		
		ALTSEL0P3	ALTSEL1P3	DP3
1	2	3	4	5
P3.15		P15 = 0		P15 = 0/1
	CLKOUT	P15 = 1	P15 = 0	
	FOUT	P15 = 0	P15 = 1	
P3.14				P14 = 0/1
P3.13		P13 = 0		P13 = 0/1
	SCLK0 (вход)			P13 = 0
	SCLK0 (выход)	P13 = 1		P13 = 1
	EX3INA			P13 = 0
P3.12				P12 = 0/1
	BHE#			
	WRH#			
	EX3INB			
P3.11		P11 = 0		P11= 0/1
	RxD0 (вход)			P11 = 0
	RxD0 (выход)	P11 = 1		P11 = 1
	EX2INA			P11 = 0

Окончание таблицы 11.5

1	2	3	4	5
P3.10		P10 = 0		P10 = 0/1
	TxD0	P10 = 1		P10 = 1
	EX2INB			P10 = 0
P3.9		P9 = 0		P9 = 0/1
	MTSR0 (вход)			P9 = 0
	MTSR0 (выход)	P9 = 1		P9 = 1
P3.8		P8 = 0		P8 = 0/1
	MRST0 (вход)			P8 = 0
	MRST0 (выход)	P8 = 1		P8 = 1
P3.7		P7 = 0		P7 = 0/1
	T2IN			P7 = 0
P3.6				P6 = 0/1
	T3IN			P6 = 0
P3.5		P5 = 0		P5 = 0/1
	T4IN			P5 = 0
	TxD1		P5 = 1	
P3.4				P4 = 0/1
	T3EUD			P4 = 0
P3.3		P3 = 0		P3 = 0/1
	T3OUT	P3 = 1		P3 = 1
P3.2				P2 = 0/1
	CAPIN			P2 = 0
P3.1		P1 = 0	P1 = 0	P1 = 0/1
	T6OUT	P1 = 0	P1 = 1	P1 = 1
	RxD1 (вход)			P1 = 0
	RxD1 (выход)	P1 = 1		P1 = 1
	EX1INA			P1 = 0
P3.0		P0 = 0		P0 = 0/1
	T0IN			P0 = 1
	TxD1	P0 = 1		P0 = 0
	EX1INB			P0 = 0

11.5 Порт P4

8-разрядный порт. Используется для ввода и вывода общего назначения и выполнения альтернативных функций. Направление работы каждого вывода может быть задано с помощью регистра направления DP4. Если порт P4 используется как внешняя шина, то в этом случае (и только в этом) его функциями управляет контроллер EBC.

Альтернативные функции порта P4

При использовании режима сегментированной памяти, во время циклов работы внешней шины, выводы 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 микроконтроллера.

Альтернативные функции порта управляются регистром ALTSEL0P4 и приведены в таблице 11.6, структурная схема вывода порта показана на рисунке 11.6.

Таблица 11.6 – Выводы порта P4 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние EBC	Состояние соответствующих битов регистров управления	
			ALTSEL0P4	DP4
P4.7			P7 = 0	P7 = 0/1
	A23	Активен		
	CAN2TxD		P7 = 1	P7 = 1
	CAN1RxD			P7 = 0
	CAN2RxD			P7 = 0
	EX4INA			P7 = 0
P4.6			P6 = 0	P6 = 0/1
	A22	Активен		
	CAN1TxD		P6 = 1	P6 = 1
	EX5INA			P6 = 0
P4.5			P5 = 0	P5 = 0/1
	A21	Активен		
	CAN1RxD		P5 = 0	P5 = 0
	CAN2RxD		P5 = 0	P5 = 0
	CAN1TxD & CAN2TXD		P5 = 1	P5 = 1
	EX4INB			
P4.4			P4 = 0	P4 = 0/1
	A20	Активен		
	CAN1RxD			P4 = 0
	CAN2RxD			P4 = 0
	EX5INB			P4 = 0
P4.3 – P4.0			P3 = 0, P2 = 0, P1 = 0, P0 = 0	P3 = 0/1, P2 = 0/1, P1 = 0/1, P0 = 0/1
	A19 – A16	Активен		

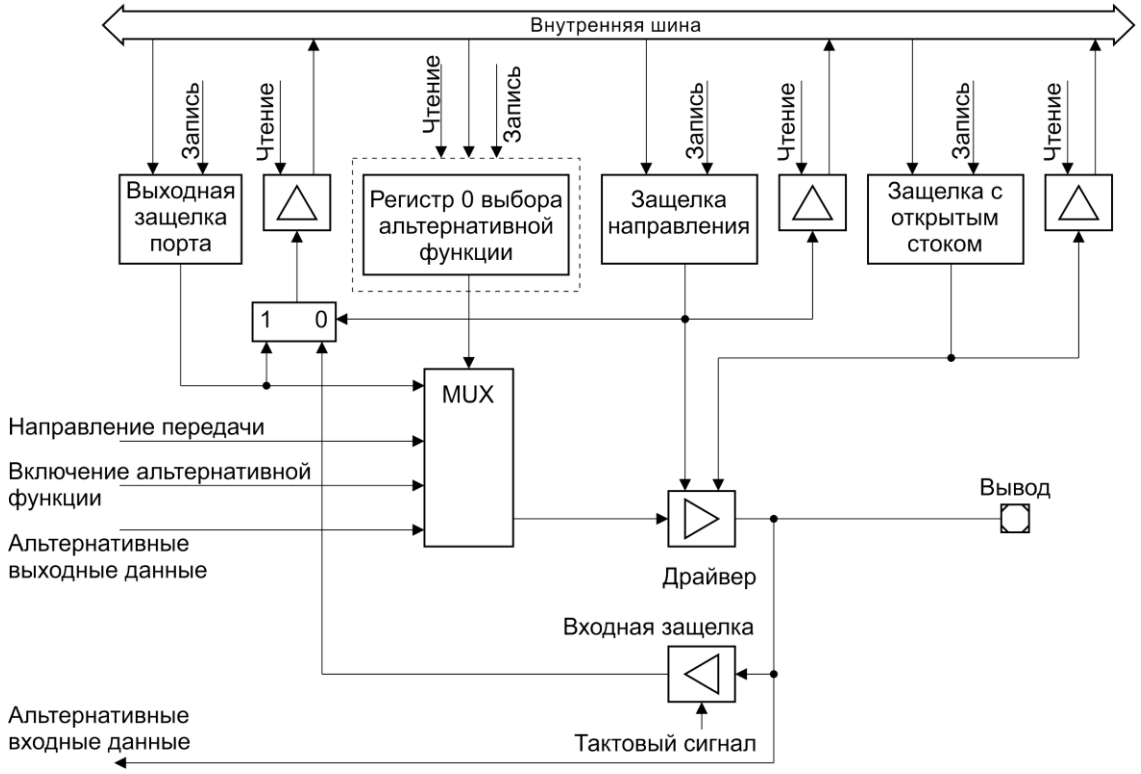


Рисунок 11.6 – Структурная схема вывода порта P4

11.6 Порт P5

16-разрядный входной порт общего назначения. Для этого порта нет выходного триггера и регистра направления. Регистр порта предназначен только для чтения.

Альтернативные функции порта P5

Порт не имеет регистра управления альтернативными функциями. Старшие шесть входов порта P5 дополнительно могут работать в качестве линий внешнего управления таймерами модулей GPT1 и GPT2. Альтернативные функции порта приведены в таблице 11.7, структурная схема вывода порта показана на рисунке 11.7.

Таблица 11.7 – Выводы порта P5 и их альтернативные функции

Вывод порта	Альтернативная функция
P5.15	
	T2EUD
P5.14	
	T4EUD
P5.13	
	T5IN
P5.12	
	T6IN
P5.11	
	T5EUD
P5.10	
	T6EUD
P5.9 – P5.0	Отсутствует

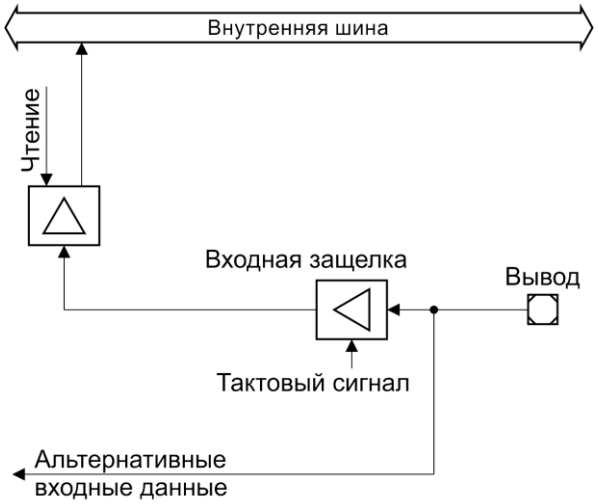


Рисунок 11.7 – Структурная схема вывода порта P5

11.7 Порт P6

8-разрядный порт. Используется для ввода и вывода общего назначения и выполнения альтернативных функций. Направление работы каждого вывода может быть сконфигурировано с помощью регистра направления DP6. Каждая линия порта может быть переключена или в режим «push/pull», или в режим с открытым стоком, с помощью регистра управления открытым стоком ODP6.

Альтернативные функции порта P6

Сигналы выбора кристалла CS4#, CS3#, CS2#, CS1# и CS0# могут быть заданы на пяти выводах порта. Число задействованных сигналов устанавливается во время

системного сброса в соответствии с конфигурацией кристалла. Линии выбора кристалла имеют слабый внутренний «pullup», аппаратно включающийся 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.

Альтернативные функции порта управляются регистром ALTSEL0P6 и приведены в таблице 11.8, структурная схема вывода порта показана на рисунке 11.8.

Таблица 11.8 – Выводы порта P6 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние соответствующих битов регистров управления	
		ALTSEL0P6	DP6
P6.7		P7 = 0 и выбор кристалла запрещен	P7 = 0/1
	CC7IO (вход)		P7 = 0
	CC7IO (выход)	P7 = 1	P7 = 1
P6.6		P6 = 0 и выбор кристалла запрещен	P6 = 0/1
	CC6IO (вход)		P6 = 0
	CC6IO (выход)	P6 = 1	P6 = 1
P6.5		P5 = 0 и выбор кристалла запрещен	P5 = 0/1
	CC5IO (вход)		P5 = 0
	CC5IO (выход)	P5 = 1 и выбор кристалла разрешен	P5 = 1
P6.4 – P6.0		P4, P3, …, P0 = 0 и выбор кристалла запрещен	P4, P3, …, P0 = 0/1
	CS4# – CS0#		
	CC4IO – CC0IO (входы)		P4, P3, …, P0 = 0
	CC4IO – CC0IO (выходы)	P4, P3, …, P0 = 1 и выбор кристалла запрещен	P4, P3, …, P0 = 1

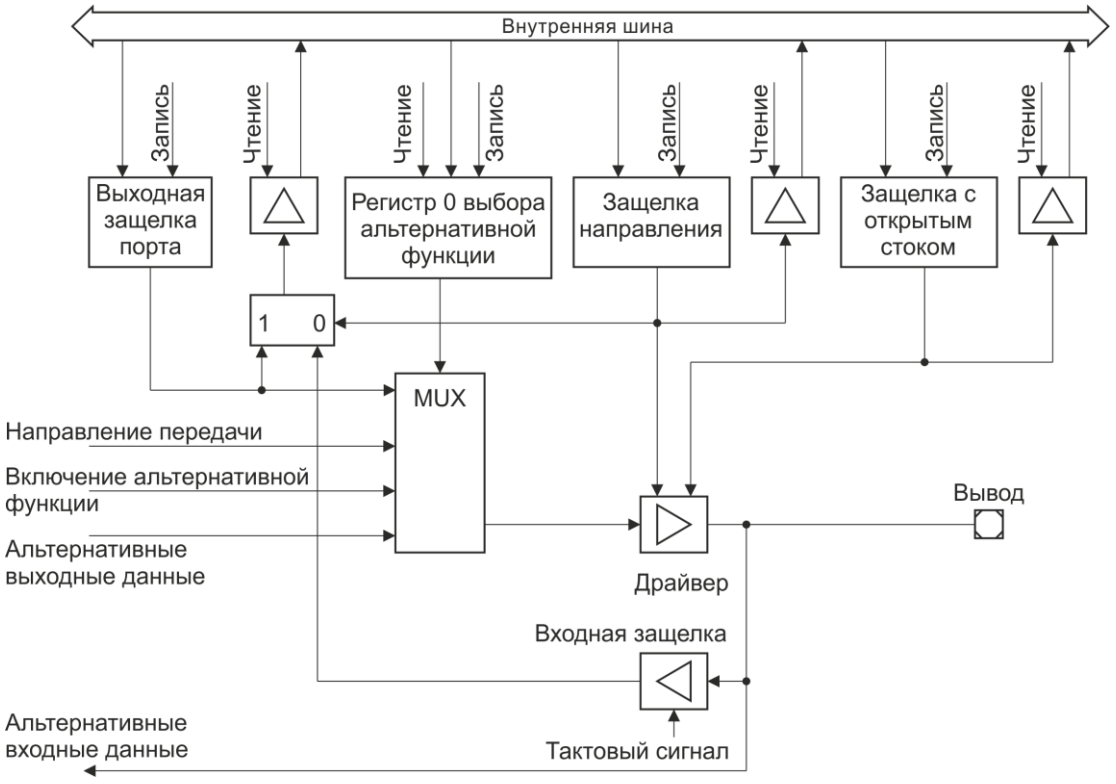


Рисунок 11.8 – Структурная схема вывода порта P6

11.8 Порт P7

7-разрядный порт. Используется для ввода и вывода общего назначения. Направление работы каждого вывода может быть сконфигурировано с помощью регистра направления DP7. Каждая линия порта может быть переключена в режим «push/pull» или в режим с открытым стоком с помощью регистра ODP7.

Альтернативные функции порта P7

Альтернативные функции порта управляются регистрами ALTSEL0P7 и ALTSEL1P7 и приведены в таблице 11.9, структурная схема вывода порта показана на рисунке 11.9.

Таблица 11.9 – Выводы порта P7 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние соответствующих битов регистров управления		
		ALTSEL0P7	ALTSEL1P7	DP7
P7.7		P7 = 0	P7 = 0	P7 = 0/1
	CC31IO (вход)			P7 = 0
	CC31IO (выход)	P7 = 0	P7 = 1	P7 = 1
	CAN1TxD	P7 = 1	P7 = 0	P7 = 1
	EX6INA			P7 = 0
P7.6		P6 = 0	P6 = 0	P6 = 0/1
	CC30IO (вход)			P6 = 0
	CC30IO (выход)	P6 = 0	P6 = 1	P6 = 1
	CAN1RxD			P6 = 0
	CAN2RxD			P6 = 0
	CAN1TxD & CAN2TXD	P6 = 1	P6 = 0	P6 = 1
	EX7INA			
P7.5		P5 = 0	P5 = 0	P5 = 0/1
	CC29IO (вход)			P5 = 0
	CC29IO (выход)	P5 = 0	P5 = 1	P5 = 1
	CAN2TxD			
	EX6INB			P5 = 0
P7.4			P4 = 0	P4 = 0/1
	CC28IO (вход)			P4 = 0
	CC28IO (выход)		P4 = 1	P4 = 1
	CAN1RxD			P4 = 0
	CAN2RxD			P4 = 0
	EX7INB			P4 = 0
P7.3		P3 = 0		P3 = 0/1
	CAN1TxD	P3 = 1		P3 = 1
P7.2		P2 = 0		P2 = 0/1
	CAN2TxD	P2 = 1		P2 = 1
P7.1		P1= 0		P1 = 0/1
	CAN1TxD & CAN2TxD	P1= 1		P1 = 1
P7.0	Не используется			

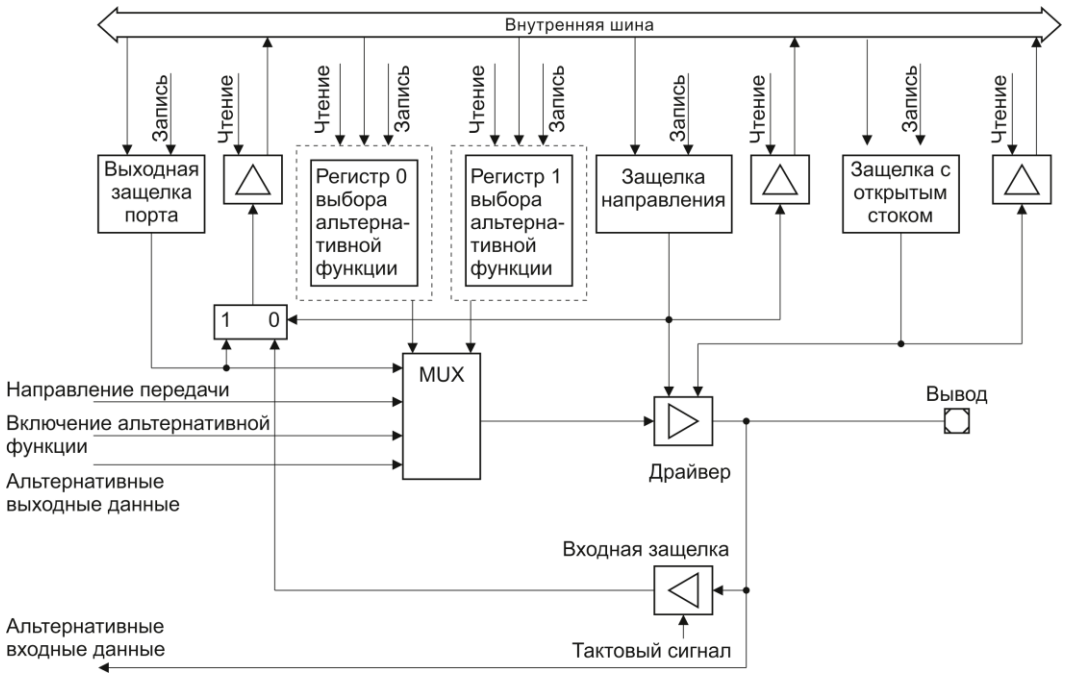


Рисунок 11.9 – Структурная схема вывода порта P7

11.9 Порт P9

8-разрядный порт. Используется для ввода и вывода общего назначения. Направление работы каждого вывода может быть сконфигурировано с помощью регистра направления DP9. Каждая линия порта может быть переключена в режим «push/pull» или в режим с открытым стоком с помощью регистра ODP9.

Альтернативные функции порта управляются регистрами ALTSEL0P9 и ALTSEL1P9 и приведены в таблице 11.10, структурная схема вывода порта показана на рисунке 11.10.

Таблица 11.10 – Выводы порта P9 и их альтернативные функции

Вывод порта	Альтернативная функция	Состояние соответствующих битов регистров управления		
		ALTSEL0P9	ALTSEL1P9	DP9
1	2	3	4	5
P9.7	Отсутствует	P7 = 0	P7 = 0	P7 = 0/1
P9.6	Отсутствует	P6 = 0	P6 = 0	P6 = 0/1
P9.5		P5 = 0	P5 = 0	P5 = 0/1
	CC21IO (вход)			P5 = 0
	CC21IO (выход)	P5 = 0	P5 = 1	P5 = 1
	SCL2 (вход)			P5 = 0
	SCL2 (выход)	P5 = 1	P5 = 0/1	P5 = 1
P9.4		P4 = 0	P4 = 0	P4 = 0/1
	CC20IO (вход)			P4 = 0
	CC20IO (выход)	P4 = 0	P4 = 1	P4 = 1
	SDA2 (вход)			P4 = 0
	SDA2 (выход)	P4 = 1	P4 = 0/1	P4 = 1
P9.3		P3 = 0	P3 = 0	P3 = 0/1
	CC19IO (вход)			P3 = 0
	CC19IO (выход)	P3 = 0	P3 = 1	P3 = 1
	CAN1TxD	P3 = 1	P3 = 1	P3 = 1
	SCL1 (вход)			P3 = 0
	SCL1 (выход)			P3 = 1

Окончание таблицы 11.10

1	2	3	4	5
P9.2		P2 = 0	P2 = 0	P2 = 0/1
	CC18IO (вход)			P2 = 0
	CC18IO (выход)	P2 = 0	P2 = 1	P2 = 1
	CAN2RxD			P2 = 1
	CAN1RxD			P2 = 1
	SDA1 (вход)			P2 = 0
	SDA1 (выход)			P2 = 1
P9.1		P1 = 0	P1 = 0	P1 = 0/1
	CC17IO (вход)			P1 = 0
	CC17IO (выход)	P1 = 0	P1 = 1	P1 = 1
	CAN2TxD	P1 = 1	P1 = 1	P1 = 1
	SCL0 (вход)			P1 = 0
	SCL0 (выход)			P1 = 1
P9.0		P2 = 0	P2 = 0	P2 = 0/1
	CC16IO (вход)			P2 = 0
	CC16IO (выход)	P2 = 0	P2 = 1	P2 = 1
	CAN1RxD			P2 = 0
	CAN2RxD			P2 = 0
	SDA0 (вход)			P2 = 0
	SDA0 (выход)			P2 = 1

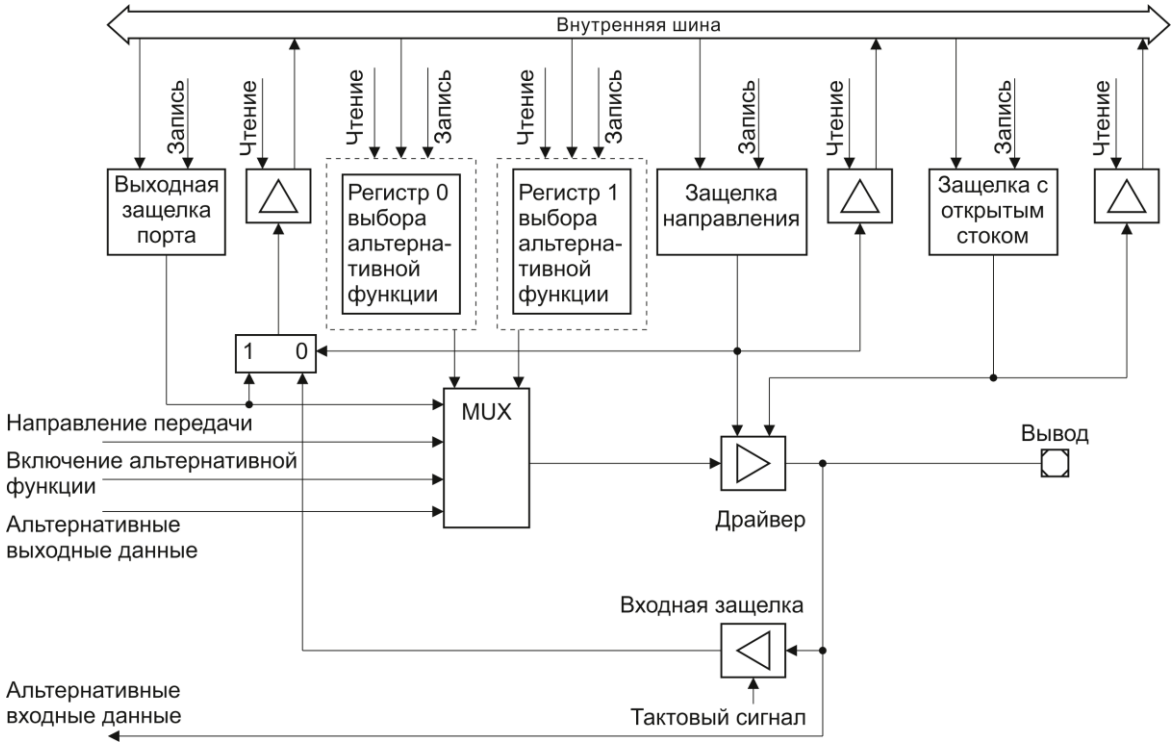


Рисунок 11.10 – Структурная схема вывода порта P9

11.10 Специальные выводы

Некоторые сигналы используют отдельные выводы микроконтроллера. Список специальных выводов микроконтроллера приведен в таблице 11.11.

Таблица 11.11 – Специальные выводы

Вывод	Функция
ALE	Включение защелки адреса
RD#	Стробирующий сигнал чтения внешней памяти
WR#/WRL#	Стробирующий сигнал записи для внешней памяти – запись данных/запись младшего байта
READY	Вход готовности
EA#	Включение внешнего доступа
NMI#	Вход немаскируемого прерывания
XTAL1, XTAL2	Вход и выход основного осциллятора
XTAL3, XTAL4	Вход и выход вспомогательного осциллятора
RSTIN#	Вход сброса
RSTOUT#	Выход сброса
TRST#	Вход сброса для системы отладки OCDS
TMS, TCK, TDI, TDO	Интерфейс JTAG (используется системой OCDS)
BRKIN#, BRKOUT#	Интерфейс останова для системы OCDS
#VCC, #0V	Выводы питания и общие выводы 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 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 #VCC, #0V предназначены для подключения питания микроконтроллера. Разделительные конденсаторы рекомендуется подключать как можно ближе к выводам питания.

Сигнал ALE осуществляет управление 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 и внутренним регистрам SFR. Во время сброса внутренняя схема обеспечивает низкий уровень сигнала ALE. Уровень сигнала на выводе ALE во время сброса задает режим работы от осциллятора – ALE = 0 и RD# = 0.

Сигнал RD# предназначен для управления выходными буферами внешней памяти или периферии при работе с внешними устройствами. Во время доступа к X-периферии остается в неактивном высоком состоянии. 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 должен быть низким.

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 байт, либо сигнал WRL# – для записи только младшего байта слова, во время доступа к X-периферии – остается неактивным. 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. Поскольку микроконтроллер не имеет внутреннего ПЗУ, то вывод EA# во время сигнала сброса должен находиться в состоянии логического нуля, чтобы после сброса микроконтроллера выборка команд началась из внешней памяти.

Установка EA# в состояние логической единицы не допускается, поскольку приведет к неконтролируемой работе микроконтроллера.

Вход RSTIN# внешнего сигнала сброса обеспечивает аппаратный сброс микроконтроллера при включении питания в случае отказа аппаратных средств или при ручном сбросе. Сигнал внешнего сброса на выводе RSTIN# должен удерживаться не менее 1 мкс.

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.

12 Часы реального времени (RTC)

Часы реального времени (далее модуль RTC) представляют собой счетчик, основным назначением которого является непрерывный отсчет реального времени (от начала запуска счетчика) и предоставление информации о времени центральному процессору. Период модуля RTC составляет 136 лет при тактировании сигналом с частотой 32,768 кГц (формируется внешним источником на выводе XTAL3 микроконтроллера). Модуль имеет механизм подстройки частоты внутреннего синхросигнала для компенсации отклонений частоты опорного сигнала, не прерывает работу при пониженном энергопотреблении и может быть остановлен только при выполнении Power-On-Reset. Функциональная схема модуля RTC представлена на рисунке 12.1.

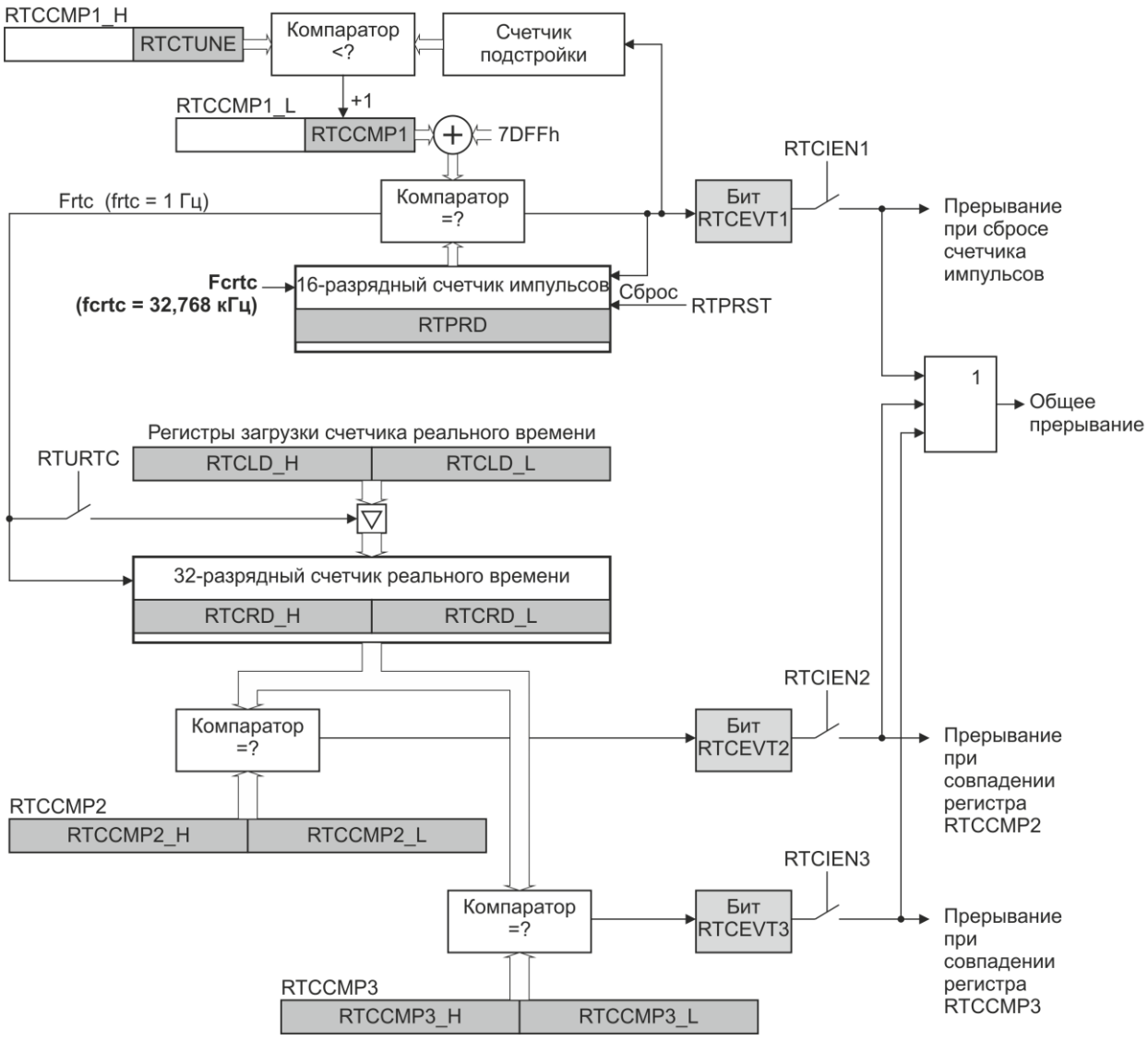


Рисунок 12.1 – Схема модуля RTC

Функционирование модуля RTC

Вся работа модуля тактируется входным синхросигналом Fcrtc, рекомендуемая частота которого составляет 32,768 кГц. Входной синхросигнал Fcrtc с входа XTAL3 микроконтроллера поступает на 16-разрядный счетчик импульсов (регистр RTPRD), который инкрементируется каждый такт синхросигнала. Одновременно с переключением значение счетчика сравнивается посредством компаратора со значением, которое

получается сложением битового поля RTCCMP1 (регистр RTCCMP_L) и константы 7DFFh. Как только возникает совпадение, счетчик импульсов сбрасывается. Одновременно с этим, в регистре RTCEIST устанавливается флаг RTCEVT1 и, если установлен бит RTCIEN1 (регистр RTCIEN), формируется запрос на прерывание.

После сброса счетчик импульсов продолжает инкрементироваться и при переключении из 0000h в 0001h формируется импульс сигнала Frtc. Этот сигнал является тактирующим сигналом 32-разрядного счетчика реального времени. При точном отсчете времени частота сигнала frtc должна равняться 1 Гц.

При поддержании на входе счетчика импульсов рекомендованной частоты синхросигнала, счетчик переключается 32 768 раз в секунду (время одного переключения составляет 0,03 мс). Отсюда следует, что для формирования сигнала с частотой 1 Гц суммарное значение поля RTCCMP1 и константы 7DFFh должно быть 8000h (т.е. 32 768 в десятичном формате). И, значит, для грубой настройки модуля RTC достаточно записать в поле RTCCMP1 значение 200h. На рисунке 12.2 показано влияние значения RTCCMP1 на период сигнала Frtc.

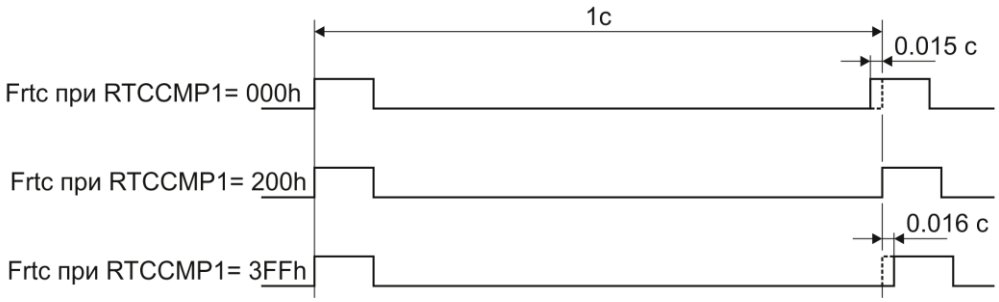


Рисунок 12.2 – Изменение периода сигнала Frtc в зависимости от состояния поля RTCCMP1

Как видно, использование механизма с программируемой компенсацией отклонения частоты опорного синхросигнала (до ±1,5 %) позволяет быстро настроить часы реального времени.

Для более точной подстройки выходной частоты модуля в RTC имеется 4-разрядный счетчик подстройки. Значение подстройки содержится в поле RTCTUNE регистра RTCCMP1_H. Схема подстройки осуществляет коррекцию на протяжении 16 периодов основного счетчика. После сброса счетчика импульсов и выработки сигнала Frtc, сигнал Frtc будет задержан на один период тактового сигнала Fcrtc (32,768 кГц). Подобным образом, в зависимости от значения RTCTUNE, может быть добавлено от одного до 15 периодов тактового сигнала на протяжении 16 тактов основного счетчика времени. Шаг подстройки составляет 0,03 мс или 2 ppm, диапазон увеличения периода сигнала Frtc составляет от 0,03 до 0,45 мс (см. рисунок 12.3).

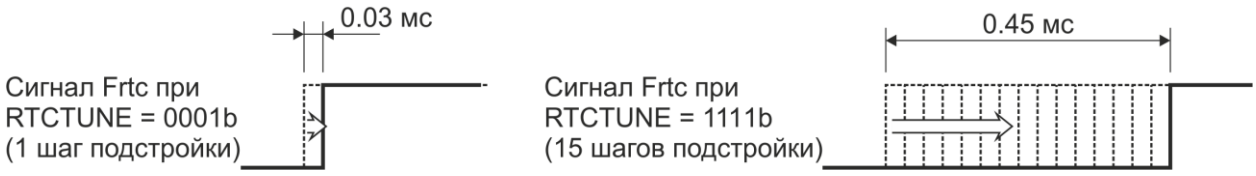


Рисунок 12.3 – Увеличение периода сигнала Frtc

Сформированный сигнал Frtc тактирует основной 32-разрядный счетчик реального времени (регистры RTCRD_H и RTCRD_L), период которого составляет 136 лет (при входной частоте 1 Гц). При каждом инкрементировании значение счетчика реального времени сравнивается посредством двух компараторов с двумя независимыми

32-разрядными регистрами сравнения RTCCMP2 и RTCCMP3. При совпадении значений устанавливается соответствующий флаг в регистре RTCEIST и, если разрешено, формируется прерывание.

Инициализация, загрузка и сброс счетчиков и регистров

16-разрядный счетчик импульсов может быть в любой момент перезапущен установкой бита RTPRST регистра RTCCST. Регистр счетчика не доступен для записи, но всегда доступен для чтения посредством регистра RTPRD.

Обратный счетчик подстройки может быть перезапущен установкой бита RTPRST, т.е. одновременно с перезапуском счетчика импульсов. После запуска состояние регистра всегда равно 1111b.

32-разрядный счетчик реального времени имеет бит запуска RTSTRT. После запуска счетчик не может быть остановлен программно, а только выполнением Power-On-Reset. Счетчик доступен как для чтения, так и для записи. Состояние счетчика может быть прочитано посредством регистров RTCRD_H и RTCRD_L. Для загрузки счетчика используется регистр RTCLD (состоит из регистров RTCLD_H и RTCLD_L). Значение, которое нужно загрузить в счетчик реального времени, записывается в регистры RTCLD_H и RTCLD_L.

Примечание – Между записью регистров (последовательность не важна) следует делать промежуток, равный времени выполнения не менее 150 команд NOP.

Так как основные регистры RTC функционируют в домене быстрого (системного) тактового сигнала и медленного (Fcrtc), то для их синхронизации предусмотрена особая схема записи/считывания. После поступления команды записи регистров, работающих в домене медленного тактового сигнала, аппаратно устанавливается бит ожидания изменения значения соответствующего регистра в регистре RTUDST. Если бит ожидания установлен, то соответствующий регистр будет изменен при поступлении следующего импульса Frtc, а бит ожидания в регистре RTUDST будет сброшен.

Так, при появлении очередного импульса сигнала Frtc содержимое регистра RTCLD загружается в счетчик реального времени, а бит RTURTC сбрасывается. Аналогичный способ считывания и загрузки имеют регистры RTC_CMP1, RTC_CMP2 и RTC_CMP3. При записи данных устанавливается соответствующий бит ожидания в регистре RTUDST. Загрузка новых данных в регистр RTC_CMP1 синхронизируется аппаратно с переключением счетчика импульсов. Загрузка новых данных в регистры RTC_CMP2 и RTC_CMP3 синхронизируется аппаратно с переключением счетчика реального времени.

Системный сброс микроконтроллера оказывает влияние не на все регистры модуля RTC. Большинство регистров могут быть сброшены только выполнением Power-On-Reset. В таблице 12.1 указано каким образом могут быть проинициализированы регистры и их начальные состояния после инициализации.

Таблица 12.1– Разделение регистров на две группы в зависимости от способа их инициализации

Регистры, которые сбрасываются при системном сбросе	Регистры, которые сбрасываются только при выполнении Power-On-Reset
RTCCST (00h)	RTPRD (0000h)
RTUDST (00h)	RTCRD_H (0000h)
RTCEIST (00h)	RTCRD_L (0000h)
RTCIEN (00h)	RTC_CMP1_H (0000h)
RTCLD (0000h)	RTC_CMP1_L (0200h)
	RTC_CMP2_H (FFFFh)
	RTC_CMP2_L (FFFFh)
	RTC_CMP3_H (FFFFh)
	RTC_CMP3_L (FFFFh)

Прерывания

Управление прерываниями в пределах модуля RTC осуществляется посредством регистра RTCIEN.

После установки флага RTCEVT1, при сбросе счетчика генерируется прерывание, если установлен бит RTCIEN1.

После установки флага RTCEVT2, при совпадении значения регистра RTCCMP2 со значением счетчика реального времени генерируется прерывание, если установлен бит RTCIEN2.

После установки флага RTCEVT3, при совпадении значения регистра RTCCMP3 со значением счетчика реального времени генерируется прерывание, если установлен бит RTCIEN3.

Вне модуля RTC, управление прерываниями осуществляется регистрами специального назначения RTC_IC1, RTC_IC2 и RTC_IC3.

Дополнительное четвертое прерывание от модуля RTC формируется, если устанавливается хотя бы один из трех флагов. Для управления прерыванием используется только регистр специального назначения RTC_IC.

13 Таймеры общего назначения (GPT)

Блоки таймеров общего 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 (GPT1) блока T2, T3, T4 имеют максимальное разрешение fosc/4. Вспомогательные таймеры первого блока могут быть настроены как регистры перезагрузки или захвата для основного таймера.

Таймеры второго (GPT2) блока T5 и T6 имеют максимальное разрешение fosc/2. Дополнительный CAPREL регистр второго блока поддерживает операции захвата и перезагрузки.

Все регистры блока GPT1 расположены в областях памяти SFR. Все три таймера блока GPT1 могут работать в четырех основных режимах: режим таймера, режим внешнего управления таймером, режим счетчика, режим внешнего инкрементирования по двум входам. Все таймеры могут считать как в положительном, так и отрицательном направлении. Каждый таймер управляется регистром TxCON (x – номер таймера).

Каждый таймер имеет входную линию TxIN (альтернативная функция 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 (вход 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 может быть прочитано и изменено с помощью ЦП посредством регистров T2, T3, T4 (расположены в не адресуемом побитно пространстве SFR). 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.

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.

Структурные схемы блоков GPT1 и GPT2 показаны на рисунках 13.1 и 13.2.

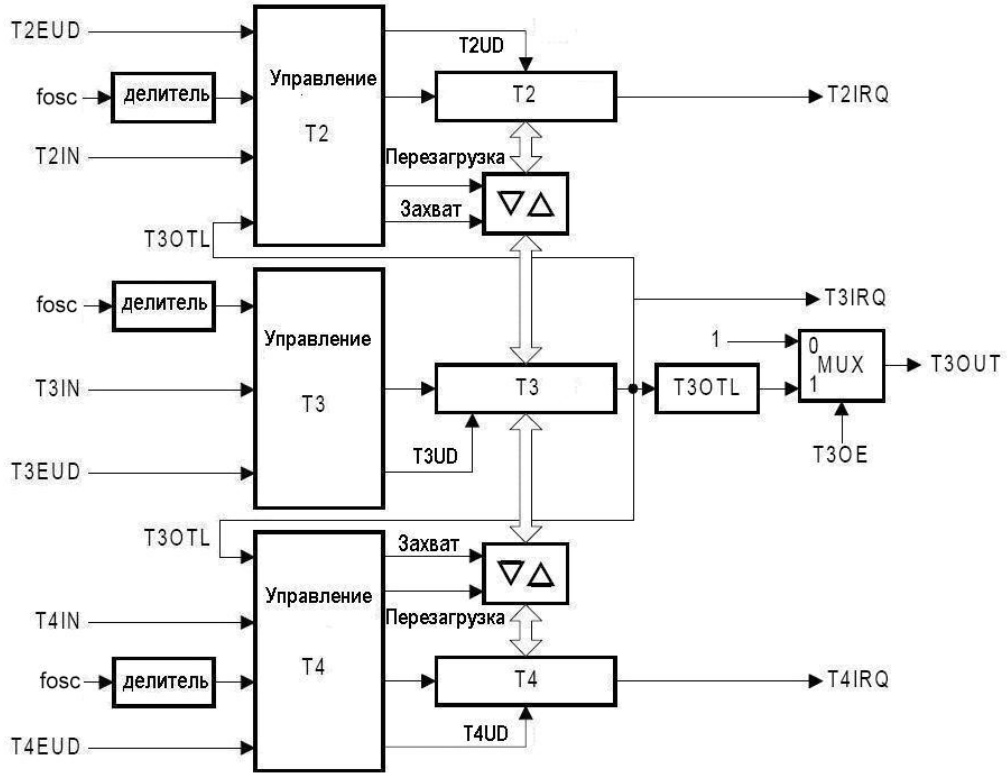


Рисунок 13.1 – Структурная схема блока GPT1

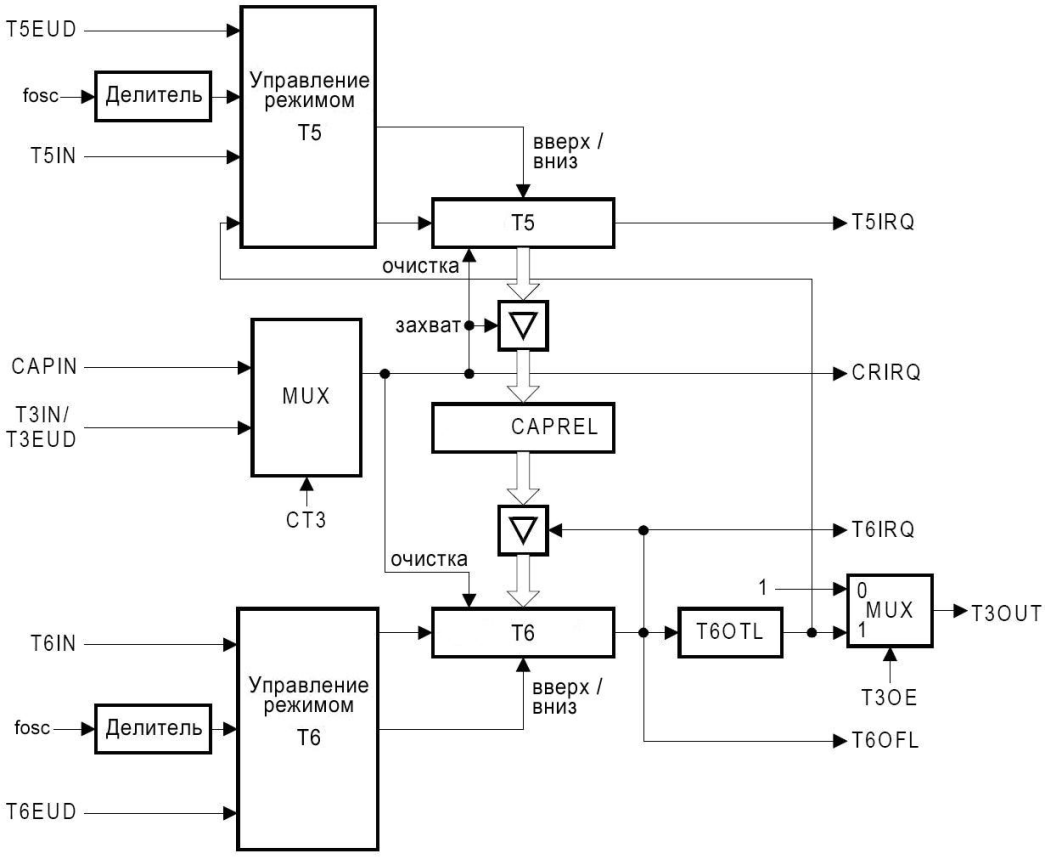


Рисунок 13.2 – Структурная схема блока GPT2

13.1 Таймер T3

Таймер T3 является основным таймером блока GPT1. Настраивается и управляется посредством побитно адресуемого регистра T3CON. Регистр T3 таймера доступен только для записи.

Управление работой таймера T3

Таймер T3 запускается/останавливается программно посредством бита T3R. В режиме внешнего управления таймер будет работать, только если бит T3R установлен 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 таймера может контролироваться программно (бит T3UD) или посредством вывода микроконтроллера T3EUD, в зависимости от состояния бита T3UDE. Направление счета можно изменять вне зависимости от того, работает таймер или нет.

Когда вывод T3UED (P3.4) используется, он должен быть сконфигурирован как вход.

Переполнение/опустошение таймера T3

Сигнал переполнения/опустошения с выхода таймера T3 подключен к схеме выходного триггера. Переполнение/опустошение таймера T3 изменяет значение бита T3OTL. Бит T3OTL также может быть установлен/сброшен программно. Бит T3OE разрешает вывод сигнала на внешнюю линию T3OUT. Если эта линия связана с внешним выводом, то T3OUT можно использовать для управления внешним устройством.

Переключение бита T3OTL может использоваться в качестве тактового сигнала счетчика, или сигнала инициализации перезагрузки вспомогательных таймеров T2 и T4. В этом случае состояние T3OTL не может быть выведено через T3OUT.

Таймер T3 в режиме таймера

Выбирается при записи значения 000b в поле T3M. Структурная схема работы T3 в режиме таймера показана на рисунке 13.3.

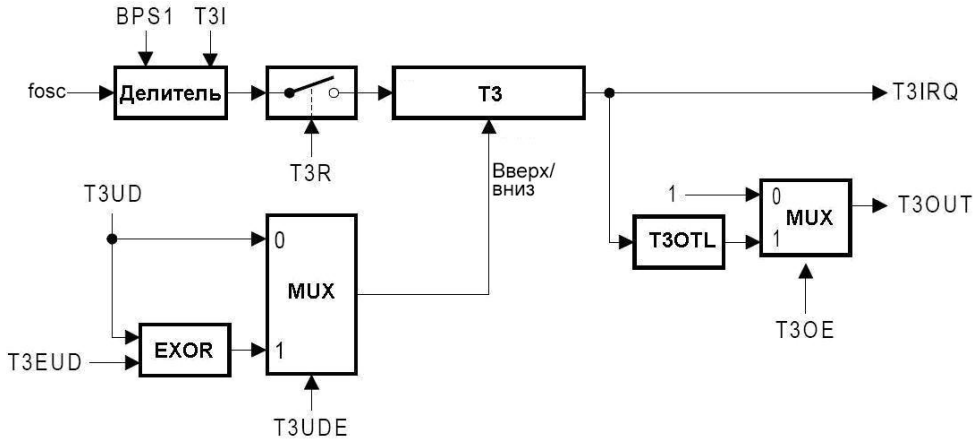


Рисунок 13.3 – Структурная схема работы таймера T3 в режиме таймера

В этом режиме тактовый сигнал fclk (в МГц) приходит на программируемый блок делителя, которым управляют битовые поля T3I и BPS1. Входная частота таймера ft3 (в МГц) и длительность такта rt3 (в мс) рассчитываются по формулам:

$$ft3 = \frac{fosc}{<BPS1> \times 2^{<T3I>}}, \qquad (13.1)$$

$$rt3 = \frac{<BPS1> \times 2^{<T3I>}}{fosc} \qquad (13.2)$$

Формулы (13.1) и (13.2) применимы к таймеру Т3 в режиме внешнего управления и к вспомогательным таймерам Т2 и Т4 в режимах таймера и внешнего управления.

Т3 в режиме внешнего управления таймером

Выбирается при записи значений 010b или 011b в поле T3M. Структурная схема работы Т3 в режиме внешнего управления таймером показана на рисунке 13.4.

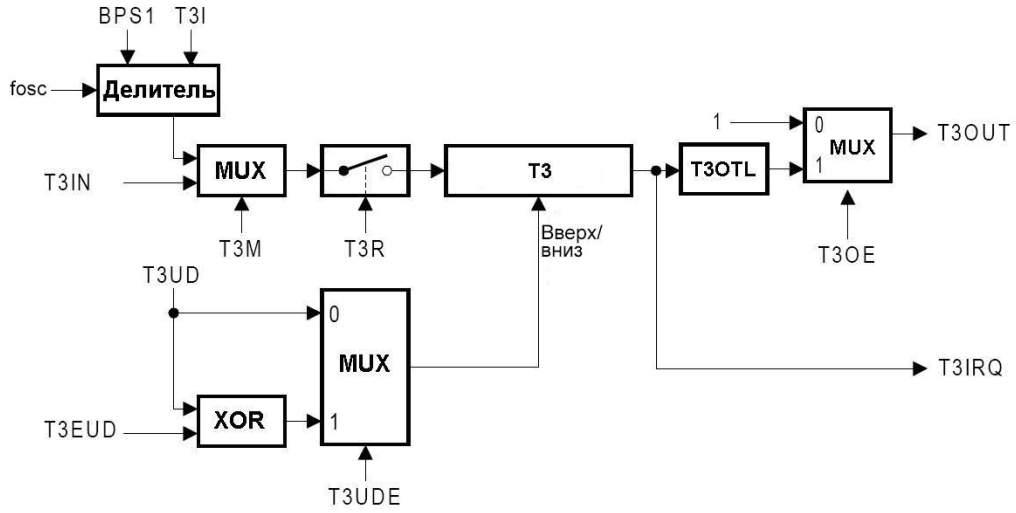


Рисунок 13.4 – Структурная схема работы таймера Т3 в режиме внешнего управления таймером

Значения младшего бита поля T3M указывают 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 Т3 отключается после подачи сигнала на вход T3IN, являющийся альтернативной функцией вывода P3.6. Для работы в этом режиме необходимо сконфигурировать P3.6 как вход.

Если T3M = 010b, то таймер будет работать, пока на T3IN удерживается ноль.

Если T3M = 011b, то таймер работает, когда на T3IN удерживается единица.

Таймер будет находиться в рабочем режиме только в том случае, когда выполняются сразу два условия – установлен бит T3R и на входе T3IN активный уровень.

Примечание – Подача сигнала внешнего управления на вход T3IN не генерирует запрос на прерывание.

Таймер Т3 в режиме счетчика

Выбирается при записи значения 001b в поле T3M.

Структурная схема работы Т3 в режиме счетчика показана на рисунке 13.5.

В этом режиме отсчеты таймера производятся по фронту сигнала на входе T3IN, являющегося альтернативной функцией P3.6. Фронт сигнала, приводящий к увеличению или уменьшению значения в счетчике, может быть как положительный, так и отрицательный. Фронт задается полем T3I.

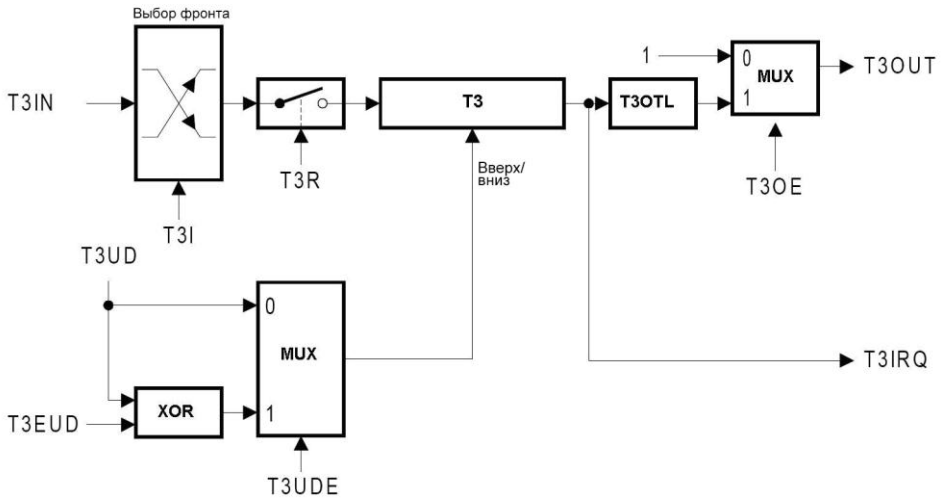


Рисунок 13.5 – Структурная схема работы таймера T3 в режиме счетчика

Максимальная входная частота, допустимая в режиме счетчика, составляет fosc/8 (BPS1 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум четыре такта fclk.

Таймер T3 в режиме внешнего инкрементирования

Выбирается при записи значений 110b или 111b в поле T3M. Структурная схема работы T3 в режиме внешнего управления таймером показана на рисунке 13.6.

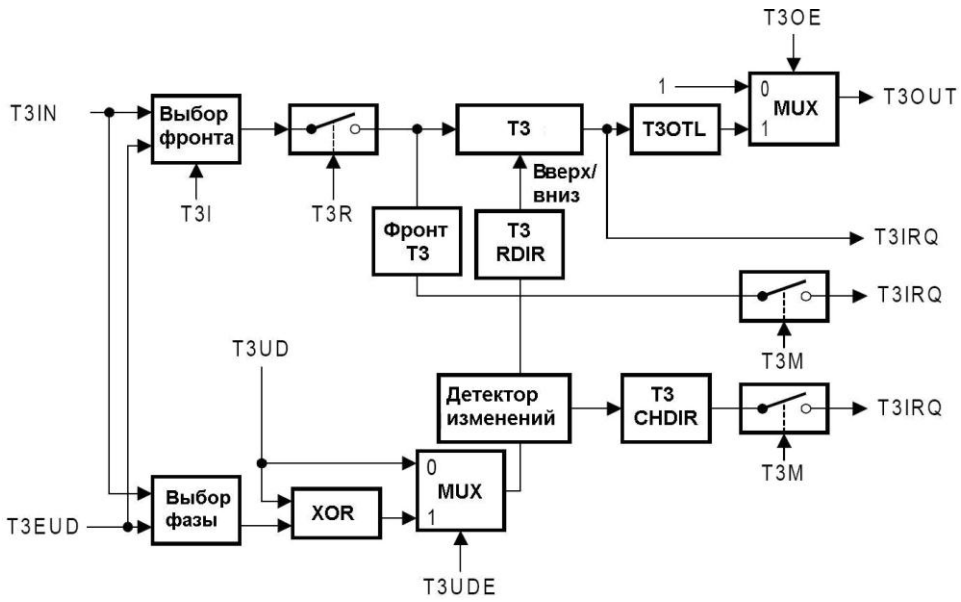


Рисунок 13.6 – Структурная схема работы таймера T3 внешнего инкрементирования

В этом режиме два входа, связанные с таймером T3 (T3IN, T3EUD), используются для соединения с внешним кодером. Таймер T3 тактируется каждым фронтом одной или двух внешних линий, который дает 2-кратное или 4-кратное разрешение входов кодера. Битовое поле T3I выбирает фронт для запуска.

Последовательность импульсов двух входных сигналов оценивается, затем генерируются счетные импульсы для таймера, и выбирается направление счета. В зависимости от выбранного режима определения периода (T3M = 110b) или определения фронта (T3M = 111b), формируется запрос на прерывание T3IRQ. В режиме определения периода прерывание генерируется каждый раз при изменении направления счета таймера

Т3. В режиме определения фронта прерывание генерируется каждый раз при счете таймера Т3. Направление счета, изменение направления счета и запросы на счет контролируются с помощью битов T3RDIR, T3CHDIR, T3EDGE.

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 13.7.

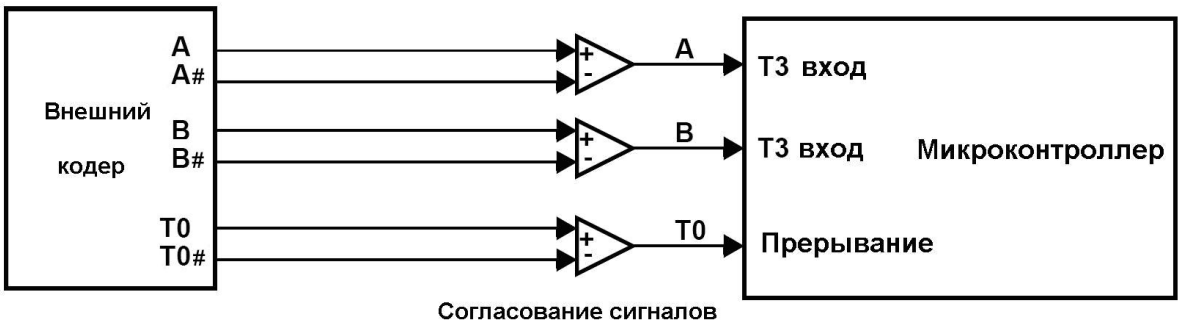


Рисунок 13.7 – Интерфейс кодера и микроконтроллера

Примечание – Третий выход T0 кодера, который указывает начальное положение, может быть подключен к внешнему прерыванию, чтобы вызвать сброс таймера Т3.

Для работы в режиме внешнего инкрементирования необходимо соблюдать следующие условия:

- выводы микроконтроллера, связанные с линиями T3IN и T3EUD, должны быть сконфигурированы как входы;
- бит T3UDE должен быть установлен для автоматической конфигурации выводов.

Максимальная частота в режиме внешнего инкрементирования составляет fosc/8 (BPS1 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум четыре такта fclk.

В режиме внешнего инкрементирования направление счета автоматически берется из последовательности, в которой изменение входных сигналов соответствует направлению вращения подключенного датчика. В таблице 13.1 объединены возможные варианты.

Таблица 13.1 – Таймер Т3 – 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 13.8 и 13.9 показаны примеры работы таймера Т3, отражающие генерирование счетных сигналов и управление направлением. На примерах видно, как компенсируется входной джиттер, который возникает в случае отсутствия сигнала датчика в моменты переключения таймера.

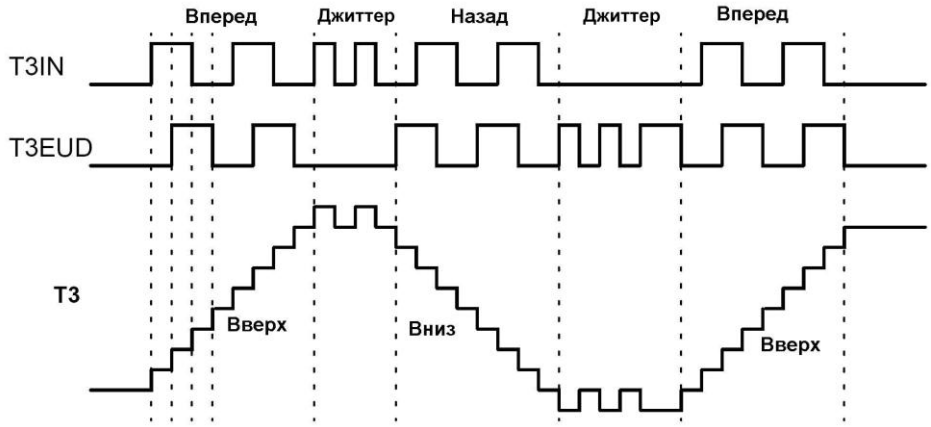


Рисунок 13.8 – Пример работы таймера T3 в режиме внешнего инкрементирования (T3I = 011b)

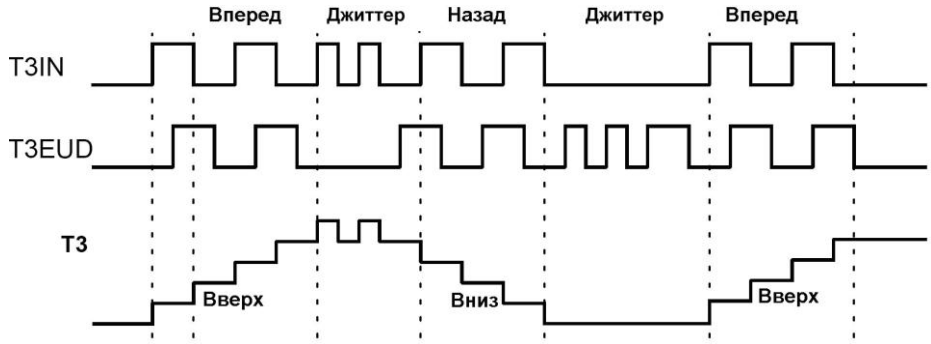


Рисунок 13.9 – Пример работы таймера T3 в режиме внешнего инкрементирования (T3I = 001b)

Таймер T3, работающий в режиме внешнего инкрементирования, автоматически хранит информацию о текущем состоянии датчика. Информация о скорости, ускорении, замедлении может быть получена при измерении периода входного сигнала.

13.2 Вспомогательные таймеры T2 и T4

Таймеры T2 и T4 являются вспомогательными таймерами блока GPT1. Настраиваются и управляются посредством побитно адресуемых регистров T2CON и T4CON. Регистры T2 и T4 таймеров доступны только для записи.

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.

Управление работой вспомогательных таймеров T2 и T4 осуществляется посредством битов T2R и T4R, а также удаленно (при установленных битах T2RC и T4RC) битом T3R.

Таймеры T2 и T4 в режиме таймера и 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 к таймерам T2 и T4. Имеются только два отличия:

- для таймеров T2 и T4 отсутствуют выходные сигналы T2OUT и T4OUT;
- отсутствует контроль переполнения/опустошения (нет битов T2OTL и T4OTL).

Таймеры T2 и T4 в режиме счетчика

В режиме счетчика таймер T2/T4 может тактироваться сигналом, приходящим на один из двух входов T2IN/T4IN или T3OTL.

Структурная схема работы T2/T4 в режиме счетчика показана на рисунке 13.10.

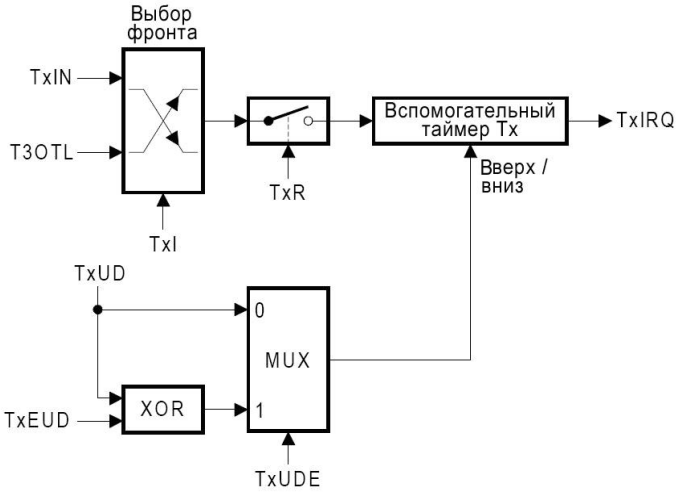


Рисунок 13.10 – Схема вспомогательного таймера в режиме счетчика

Примечание – Только переполнение/опустошение таймера T3 будет влиять на счетчик таймера T2/T4 (вход T3OTL). Программное изменение значения T3OTL не оказывает влияния.

Максимальная частота в режиме счетчика составляет fosc/8 (BPS1 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум четыре такта fclk.

Объединение таймеров

Использование T3OTL в качестве источника сигнала тактирования для вспомогательного таймера T2/T4 в режиме счетчика, объединяет этот таймер с основным таймером T3. В зависимости от того, какой фронт входного сигнала (на линии T3OTL) является активным для вспомогательного таймера, объединенные таймеры могут сформировать 32- или 33-разрядный таймер/счетчик (см. рисунок 13.11).

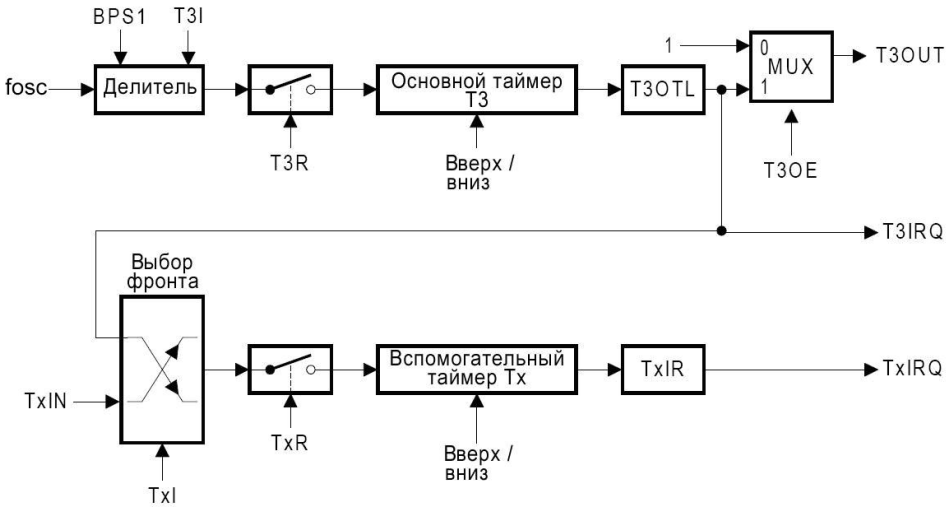


Рисунок 13.11 – Объединение основного таймера T3 и вспомогательного таймера Tx

Если для тактирования вспомогательного таймера выбраны оба фронта входного сигнала (линия T3OTL), тогда этот таймер будет переключаться при каждом

переполнении/опустошении основного таймера Т3. Таким образом, будет сформирован 32-разрядный таймер.

Если для тактирования вспомогательного таймера выбран один из фронтов (положительный или отрицательный) входного сигнала (линия Т3OTL), тогда этот таймер будет переключаться при каждом втором переполнении/опустошении основного таймера Т3. И, таким образом, будет сформирован 33-разрядный таймер (16-разрядный основной таймер + Т3OTL + 16-разрядный вспомогательный таймер).

Направления счета основного и вспомогательного таймеров в случае их объединения могут не совпадать. При этом таймер Т3 может функционировать в режимах таймера, внешнего управления и счетчика.

Примечание – На изменение уровня сигнала на линии Т3IRQ (на рисунке 13.11) оказывает влияние только переполнение/опустошение таймера Т3. Программное изменение состояния бита Т3OTL не влияет на состояние линии Т3IRQ.

Таймеры Т2 и Т4 в режиме перезагрузки

В режиме перезагрузки происходит загрузка содержимого регистра вспомогательного таймера в регистр таймера Т3, инициируемая одним из двух различных сигналов. Этот сигнал выбирается также, как сигнал тактирования в режиме счетчика. В режиме перезагрузки вспомогательный таймер Т2/Т4 может быть остановлен программно, вне зависимости от состояния его флага Т2R/Т4R.

Структурная схема работы вспомогательного таймера в режиме счетчика показана на рисунке 13.12.

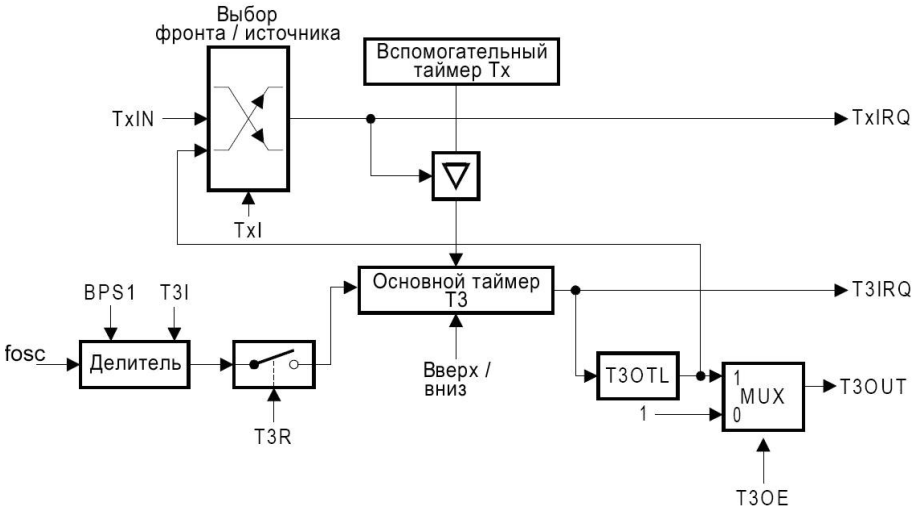


Рисунок 13.12 – Вспомогательный таймер в режиме перезагрузки

Примечание – На изменение уровня сигнала на линии Т3IRQ (на рисунке 13.12) оказывает влияние только переполнение/опустошение таймера Т3. Программное изменение состояния бита Т3OTL не влияет на состояние линии Т3IRQ.

С появлением сигнала тактирования в таймер Т3 загружается содержимое регистра таймера Т2/Т4 и Т2IRQ/Т4IRQ переходит в состояние единицы.

Режим перезагрузки с тактированием по входу Т3OTL может использоваться в различных конфигурациях. В зависимости от выбранного типа активного фронта тактового сигнала, возможны следующие функции:

1 Стандартный режим перезагрузки. Используются оба фронта сигнала тактирования по входу Т3OTL. При переполнении/опустошении основного таймера, в него загружается содержимое вспомогательного таймера.

2 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.

3 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 фронтом входного сигнала, а другой – задним, то основной таймер будет поочередно загружаться значениями вспомогательных таймеров.

На рисунке 13.13 показана схема конфигурации блока GPT1 для генерирования ШИМ-сигнала с использованием механизма поочередной загрузки основного таймера.

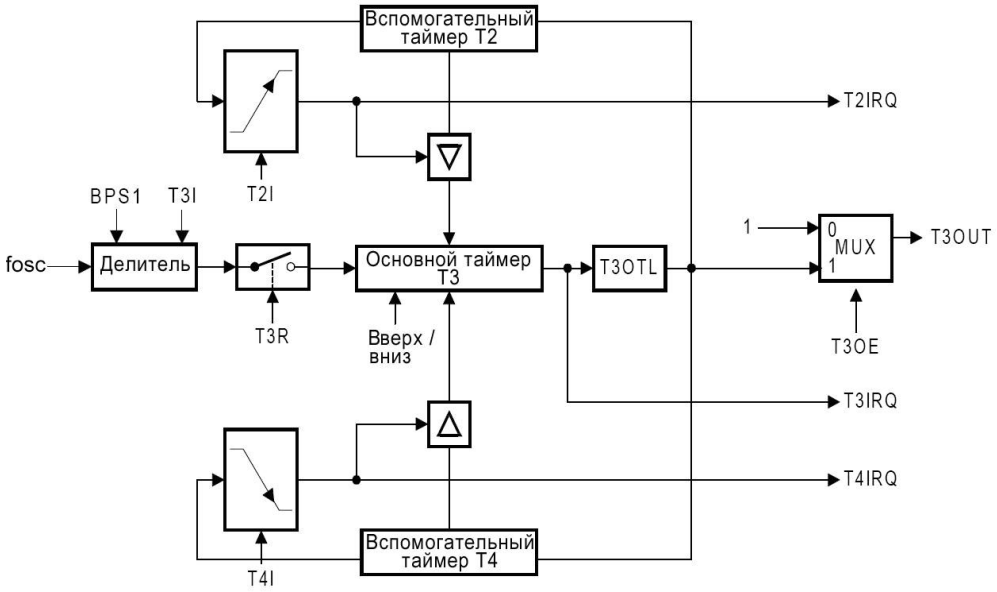


Рисунок 13.13 – Конфигурация блока GPT1 для формирования ШИМ-сигнала

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 установлен бит T3OE. 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.

Таймеры T2 и T4 в режиме захвата

В режиме захвата выбранный фронт сигнала тактирования, приходящий на вход TxIN вспомогательного таймера Tx, инициирует загрузку содержимого основного таймера T3 в регистр вспомогательного таймера. Структурная схема работы вспомогательного таймера в режиме захвата показана на рисунке 13.14.

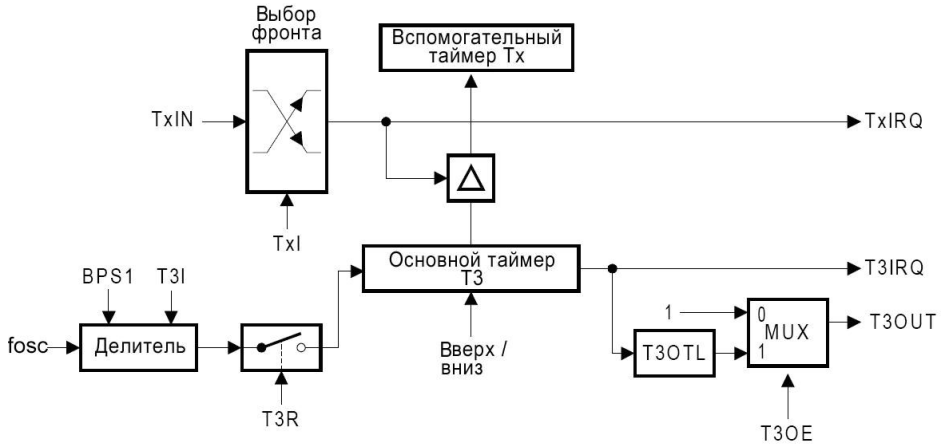


Рисунок 13.14 – Вспомогательный таймер Tx в режиме захвата

В режиме захвата два младших бита битового поля TxI используются для выбора фронта сигнала тактирования. Состояние старшего бита не важно (рекомендуемое значение для записи – ноль).

Примечание – Вспомогательный таймер T2/T4 может быть остановлен программно, независимо от состояния его бита T2R/T4R.

Одновременно с захватом значения таймера T3, линия T2IRQ/T4IRQ переходит в состояние единицы.

Максимальная частота входного сигнала составляет fosc/8 (BPS1 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум четыре такта fclk.

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 два исключения:

- для таймеров T2 и T4 отсутствует выходные сигналы T2OUT и T4OUT;
- отсутствует контроль переполнения/опустошения (нет битов T2OTL и T4OTL).

13.3 Таймер T6

Таймер T6 является основным таймером блока GPT2. Настраивается и управляется посредством побитно адресуемого регистра T6CON. Регистр T6 таймера доступен только для записи.

Управление работой таймера T6

Таймер T6 запускается/останавливается программно посредством бита T6R. В режиме внешнего управления таймер будет работать, только если бит T6R установлен и выбран активный уровень («0» или «1», в зависимости от запрограммированного значения).

Примечание – Когда бит T5RC установлен в регистре управления T5CON, бит T6R будет также управлять вспомогательным таймером T5.

Управление направлением счета

Направление счета таймера может контролироваться программно (бит T6UD) или посредством вывода микроконтроллера T6EUD, в зависимости от состояния бита T6UDE. Направление счета можно изменять вне зависимости от того, работает таймер или нет.

Когда вывод T6UED (P5.10) используется, он должен быть сконфигурирован как вход.

Переполнение/опустошение таймера T6

Сигнал переполнения/опустошения с выхода таймера T6 подключен к схеме выходного триггера. Переполнение/опустошение таймера T6 изменяет значение бита T6OTL. Бит T6OTL также может быть установлен/сброшен программно. Бит T6OE

разрешает вывод сигнала на внешнюю линию T6OUT. Если эта линия связана с внешним выводом, то T3OUT можно использовать для управления внешним устройством.

Переключение бита T3OTL может использоваться в качестве тактового сигнала счетчика или сигнала инициализации перезагрузки вспомогательного таймера T5. В этом случае состояние T6OTL не может быть выведено через T6OUT. Для тактирования других таймеров используется выход T6OFL

Таймер T6 в режиме таймера

Структурная схема работы T6 в режиме таймера показана на рисунке 13.15

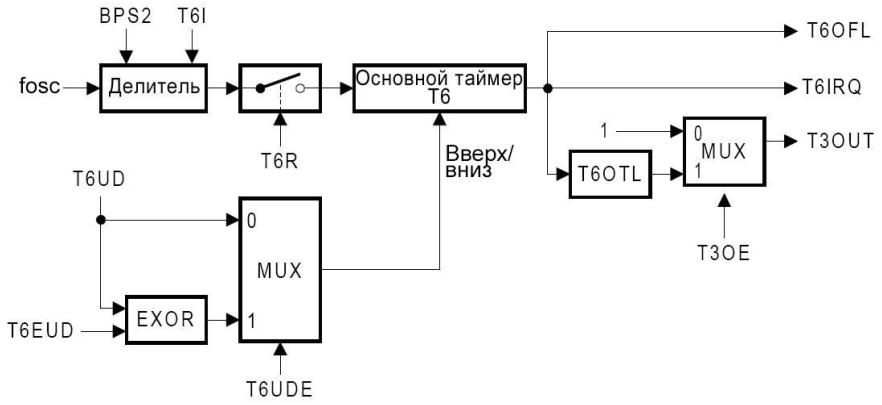


Рисунок 13.15 – Структурная схема работы таймера T6 в режиме таймера

В этом режиме тактовый сигнал fclk (в МГц) приходит на программируемый блок делителя, которым управляют битовые поля T6I и BPS2. Входная частота таймера ft6 (в МГц) и длительность такта rt6 (в мс) рассчитываются по формулам:

$$ft6 = \frac{fosc}{<BPS2> \times 2^{<T6I>}}, \quad (13.3)$$

$$rt6 = \frac{<BPS2> \times 2^{<T6I>}}{fosc} \quad (13.4)$$

T6 в режиме внешнего управления таймером

Структурная схема работы T6 в режиме внешнего управления таймером показана на рисунке 13.16.

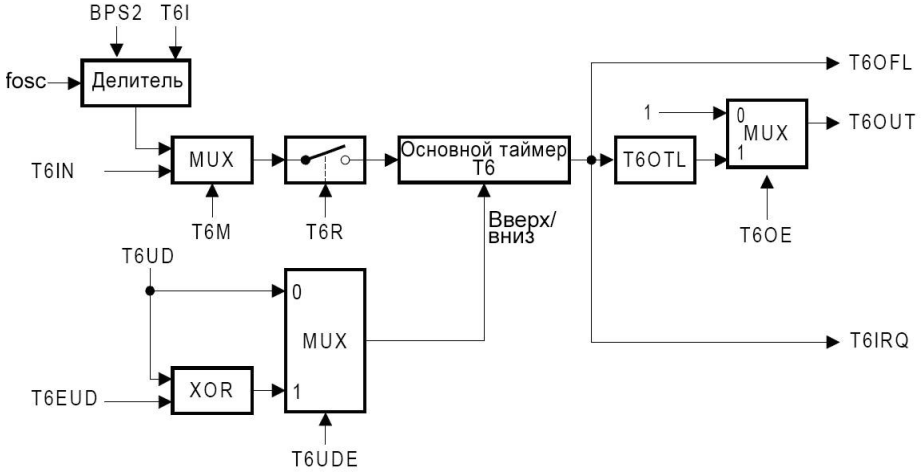


Рисунок 13.16 – Структурная схема работы таймера T6 в режиме внешнего управления таймером

Значения младшего бита поля T6M указывают 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 T6 отключается после подачи сигнала на вход T6IN, являющейся альтернативной функцией вывода P5.12. Для работы в этом режиме необходимо сконфигурировать P5.12 как вход.

Если T6M = 010b, то таймер будет работать, пока на T6IN удерживается ноль.

Если T6M = 011b, то таймер работает, когда на T6IN удерживается единица.

Таймер будет находиться в рабочем режиме только в том случае, когда выполняются сразу два условия – установлен бит T6R и на входе T6IN активный уровень.

Примечание – Подача сигнала внешнего управления на вход T6IN не генерирует запрос на прерывание.

Таймер T6 в режиме счетчика

Структурная схема работы T6 в режиме счетчика показана на рисунке 13.17.

В этом режиме отсчеты таймера производятся по фронту сигнала на входе T6IN, являющегося альтернативной функцией P5.12. Фронт сигнала, приводящего к увеличению или уменьшению значения в счетчике, может быть как положительный, так и отрицательный. Фронт задается полем T6I.

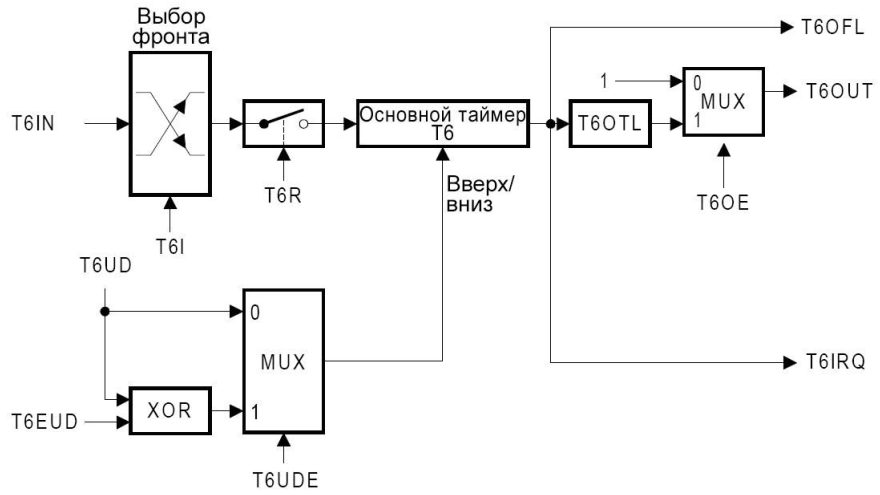


Рисунок 13.17 – Структурная схема работы таймера T6 в режиме счетчика

Максимальная входная частота, допустимая в режиме счетчика, составляет fosc/4 (BPS2 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум два такта fclk.

13.4 Вспомогательный таймер T5

Таймер T5 является вспомогательным таймером блока GPT2. Настраивается и управляется посредством побитно адресуемого регистра T5CON. Регистр T5 таймера доступен только для записи.

Таймер T5 может быть сконфигурирован для работы в режиме таймера, в режиме внешнего управления таймером, в режиме счетчика с такими же настройками тактовой частоты и входного сигнала, как для основного таймера T6. В дополнение к этим трем режимам, вспомогательный таймер может быть объединен с основным таймером.

Управление работой вспомогательного таймера T5 осуществляется посредством бита T5R, а также удаленно (при установленном бите T5RC) битом T6R. Управление направлением счета вспомогательного таймера осуществляется аналогично управлению счетом таймера T6.

Таймер T5 в режиме таймера и режиме внешнего управления таймером

Работа таймера T5 в режиме таймера или режиме внешнего управления таймером аналогична работе основного таймера T6. Описание, рисунки и таблицы, относящиеся к таймеру T6, применимы к таймеру T5. Имеются только три отличия:

- отсутствует выходной сигнал T5OUT;
- отсутствует вход T5OFL;
- отсутствует контроль переполнения/опустошения (нет бита T5OTL).

Таймер T5 в режиме счетчика

В режиме счетчика таймер T5 может тактироваться сигналом, приходящим на один из двух входов T5IN или T6OTL.

Структурная схема работы T5 в режиме счетчика показана на рисунке 13.18.

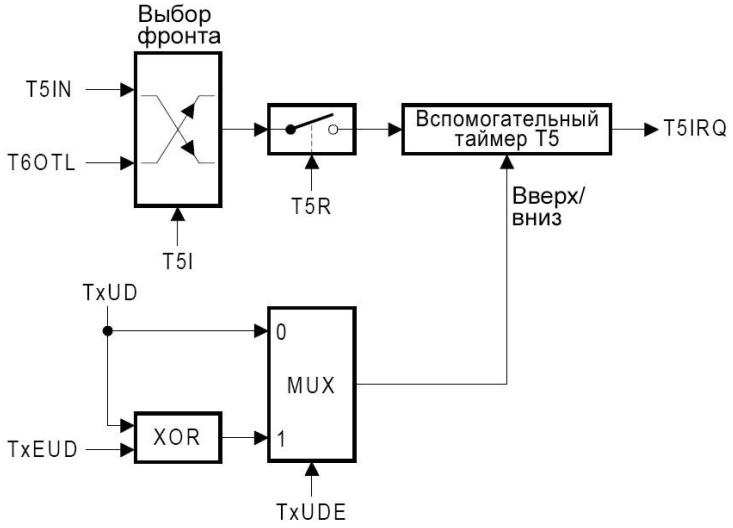


Рисунок 13.18 – Блок-схема вспомогательного таймера в режиме счетчика

Примечание – Только переполнение/опустошение таймера T6 будет влиять на счетчик таймера T5 (вход T6OTL). Программное изменение значения T6OTL не оказывает влияния.

Максимальная частота в режиме счетчика составляет fosc/4 (BPS2 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум два такта fclk.

Объединение таймеров

Использование T6OTL в качестве источника сигнала тактирования для вспомогательного таймера T5 в режиме счетчика, объединяет этот таймер с основным таймером T6. В зависимости от того, какой фронт входного сигнала (на линии T6OTL) является активным для вспомогательного таймера, объединенные таймеры могут сформировать 32- или 33-разрядный таймер/счетчик (см. рисунок 13.19).

Если для тактирования вспомогательного таймера 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.

Если для тактирования вспомогательного таймера выбран один из фронтов (положительный или отрицательный) входного сигнала (линия T6OTL), тогда этот таймер будет переключаться при каждом втором переполнении/опустошении основного таймера T6. И таким образом будет сформирован 33-разрядный таймер (16-разрядный основной таймер + T6OTL + 16-разрядный 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.

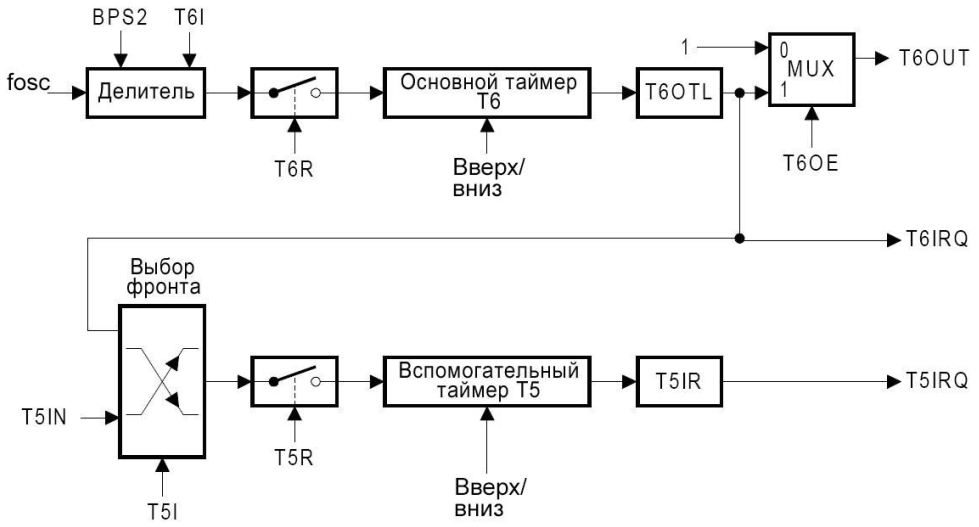


Рисунок 13.19 – Объединение основного таймера T6 и вспомогательного таймера T5

Примечание – На изменение уровня сигнала на линии T6IRQ (на рисунке 13.19) оказывает влияние только переполнение/опустошение таймера T6. Программное изменение состояния бита T6OTL не влияет на состояние линии T6IRQ.

13.5 Регистр захвата/перезагрузки CAPREL

Регистр захвата/перезагрузки CAPREL доступен только для записи

Регистр CAPREL в режиме захвата

Режим выбирается установкой бита T5SC в регистре T5CON. Структурная схема работы регистра в режиме захвата показана на рисунке 13.20.

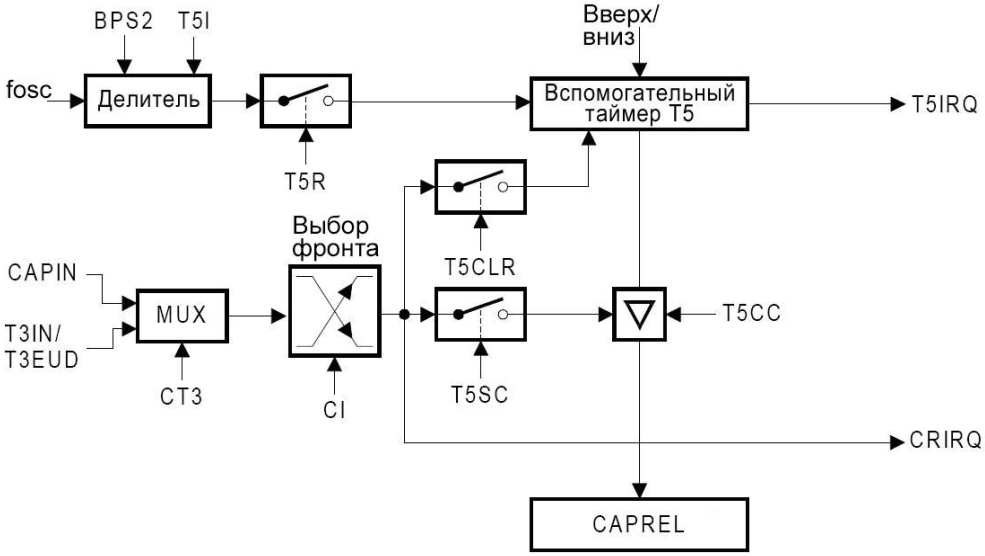


Рисунок 13.20 – Регистр CAPREL блока в режиме захвата таймера T5

Захват содержимого таймера T5 происходит по запрограммированному фронту сигнала, приходящему на выбранную битом CT3 входную линию (вход CAPIN, либо один из входов T3IN и T3EUD). За выбор источника сигнала захвата, по которому происходит захват в регистр CAPREL, отвечает битовое поле CI регистра T5CON.

Максимально допустимая частота входного сигнала равна fclk/2 (BPS2 = 01b). Время удержания уровня сигнала до появления очередного фронта – минимум один такт fclk.

Каждый раз с приходом ожидаемого фронта сигнала на выбранный вход регистр CAPREL захватывает содержимое таймера T5. Эти значения могут использоваться для измерения входного сигнала таймера T5. При захвате содержимого таймера T5 регистром CAPREL на линии прерывания CRIRQ появляется единица. Одновременно с этим, таймер T5 может быть сброшен в 0000h, если установлен бит T5CLR.

Примечание – В случае, если бит T5SC сброшен, захвата содержимого таймера не происходит, но все остальные операции могут выполняться.

Регистр CAPREL в режиме перезагрузки

Режим выбирается установкой бита T6SR в регистре T6CON. Событием, вызывающим перезагрузку, является переполнение/опустошение таймера T6 (см. рисунок 13.21).

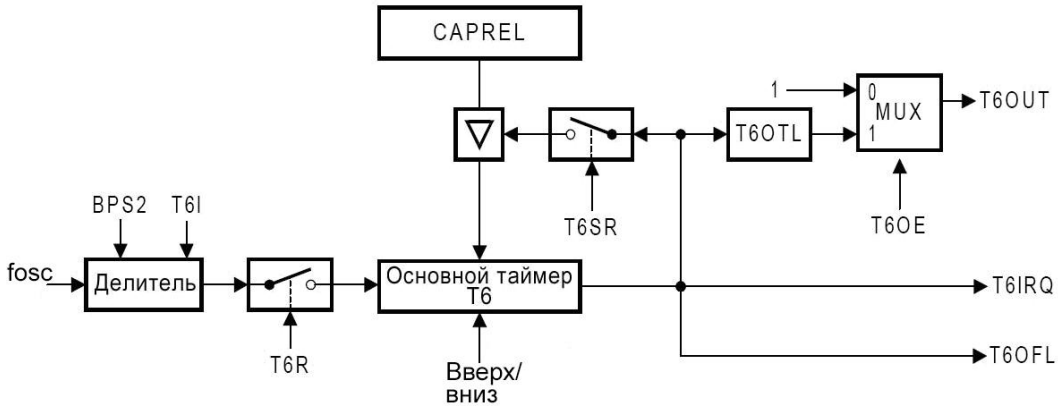


Рисунок 13.21 – Регистр CAPREL блока в режиме перезагрузки таймера T6

Когда таймер T6 переполняется при счете «вверх» и сбрасывается из состояния FFFFh в 0000h или опустошается при счете «вниз» и переключается из 0000h в FFFFh, 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 выставляется единица, которая сигнализирует о переполнении/опустошении таймера T6.

Регистр CAPREL в комбинированном режиме

Поскольку каждый из двух режимов – захвата и перезагрузки – может быть включен независимо, 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 13.22 представлена схема функционирования регистра CAPREL в комбинированном режиме.

Например, таймер T5 работает в режиме таймера и считает «вверх» с частотой fosc/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.

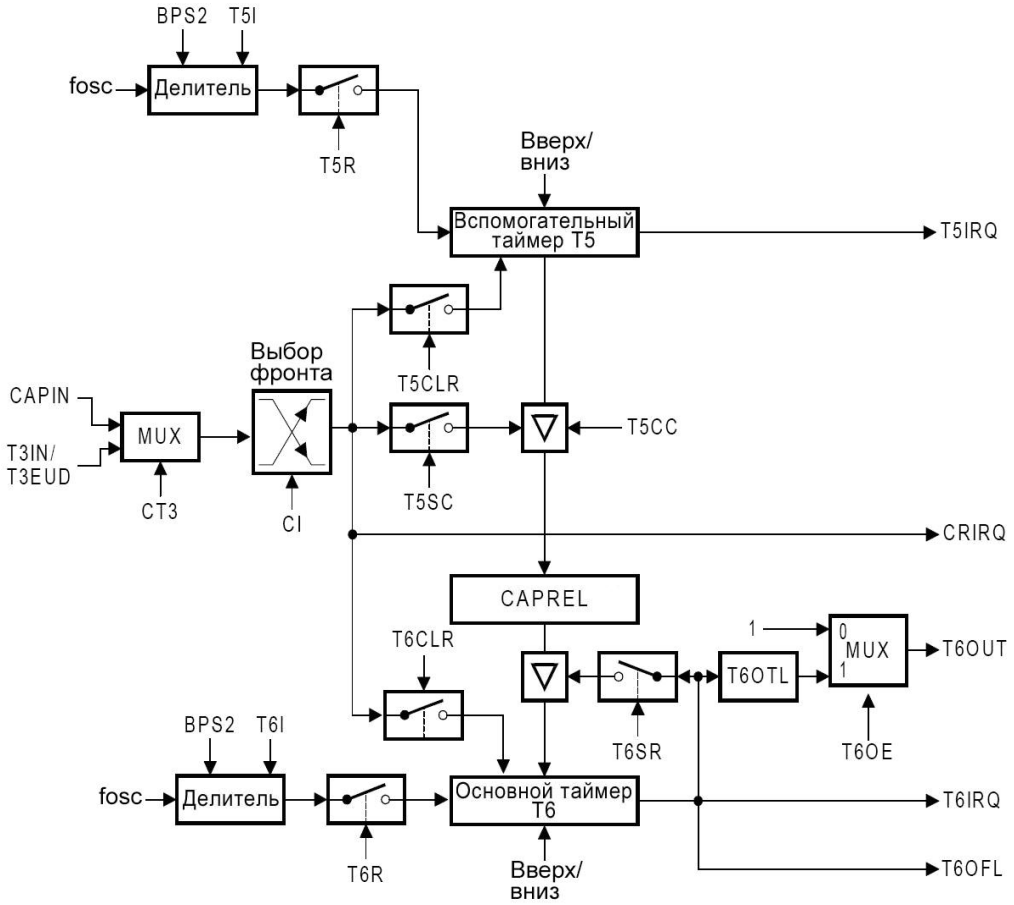


Рисунок 13.22 – Схема работы регистра CAPREL в комбинированном режиме

Таймер T6, который работает в режиме счетчика и считает «вниз» с частотой fosc/4, использует значение 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 64h/100d 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 64h, 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 бит T5CC установлен, содержимое таймера T5 декрементируется на единицу перед захватом. Такой способ позволяет устранить отклонение выходной частоты. После корректировки таймер T5 может захватить 63h/99d, и выходная частота будет равна 80 кГц.

Примечание – Помимо прочего, сигнал опустошения таймера T6 может использоваться для тактирования таймеров других блоков таймеров посредством сигнала T6OFL.

14 Асинхронно/синхронные последовательные интерфейсы ASC0 и ASC1

Асинхронно/синхронный последовательный интерфейс типа USART обеспечивает передачу данных между микроконтроллером и другими микроконтроллерами, микропроцессорами и периферией. Микроконтроллер 1887ВЕ6Т содержит два модуля последовательного интерфейса типа USART – ASC0 и ASC1. Поскольку модули полностью идентичны (за исключением адресов регистров), то в дальнейшем номер модуля будет заменен символом «x». В названиях регистров номера также буду заменены символом «x».

Особенности модуля ASCx:

- полнодуплексные асинхронные режимы (8/9-битные фреймы данных, передача младшим битом вперед, генерация/проверка бита четности, один/два стоповых бита);
- скорость пересылки данных до 390 Кбод (частота fper = 12,5 МГц);
- многопроцессорный режим для автоматического определения байта адреса/данных;
- полудуплексный 8-битный синхронный режим со скоростью пересылки данных до 1,5 Мбод (частота fper = 12,5 МГц);
- двойная буферизация при передаче/приеме;
- генерация прерываний;
- определение ошибок четности, фрейма, переполнения.

На рисунках 14.1 и 14.2 показаны блок-схемы интерфейса ASCx для асинхронного и синхронного режимов работы.

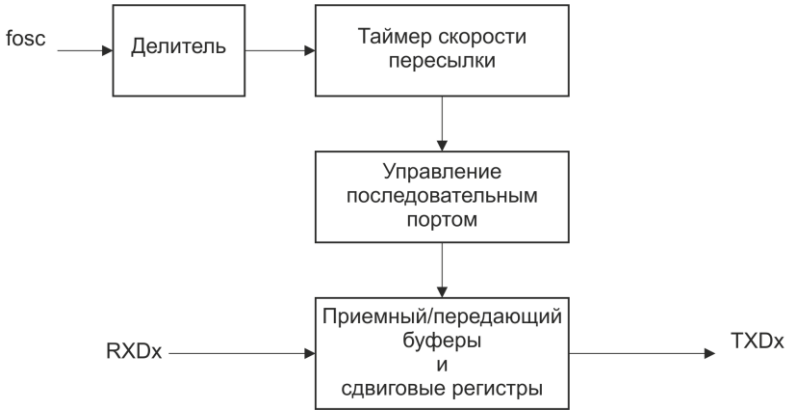


Рисунок 14.1 – Блок-схема асинхронного интерфейса ASCx

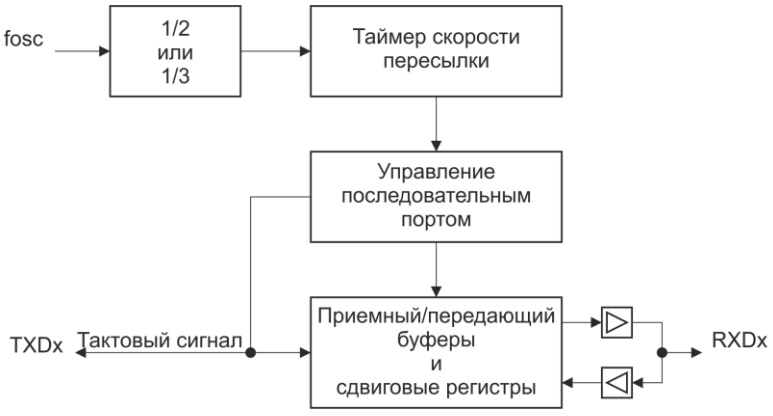


Рисунок 14.2 – Блок-схема синхронного интерфейса ASCx

14.1 Общие операции

ASCx поддерживает полнодуплексную асинхронную передачу до 390 Кбод и полудуплексную синхронную передачу до 1,5 Мбод (частота тактового генератора fper = 12,5 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 байт данных. 13-битовый таймер скорости пересылки данных с универсальной входной схемой делителя тактового генератора обеспечивает последовательный тактовый сигнал.

Конфигурация модуля ASCx задается посредством регистров SxCON, SxFDV и SxBG. Передача начинается после записи данных в передающий буферный регистр SxTBUF. Выбранный 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 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.

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 петлевом режиме 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 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.

Режимом последовательного интерфейса управляет поле M в регистре SxCON. Этот регистр содержит служебные биты для разрешения режимов определения ошибок и биты флагов состояния ошибок.

14.2 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 TXDx и принимаются на линии RXDx. На рисунке 14.3 представлена схема работы интерфейса ASCx в асинхронном режиме.

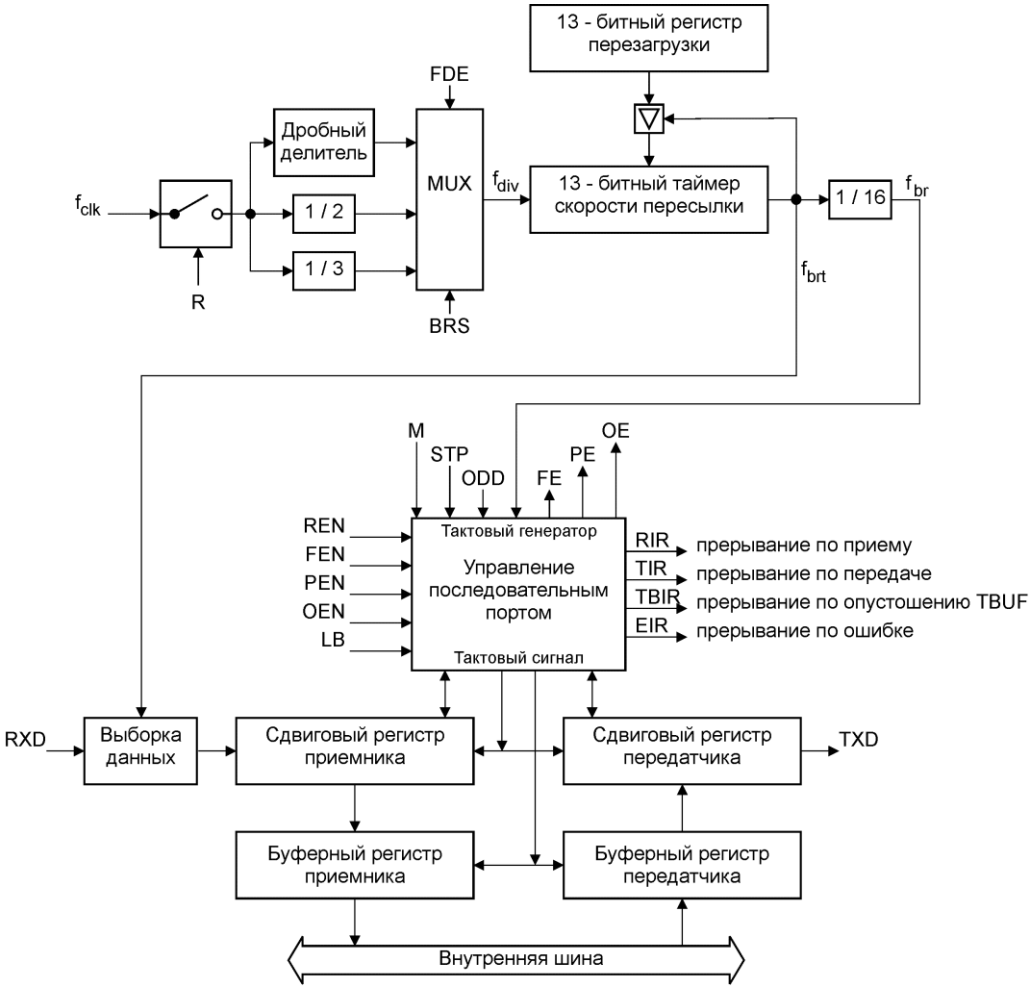


Рисунок 14.3 – Асинхронный режим работы интерфейса ASC

Асинхронные 8-разрядные фреймы данных

8-разрядные фреймы данных состоят из любых восьми информационных бит данных D7, …, D0 (M = 001b) или из семи бит данных D6, …, D0 плюс автоматически генерируемого бита четности (M = 011b) (см. рисунок 14.4).

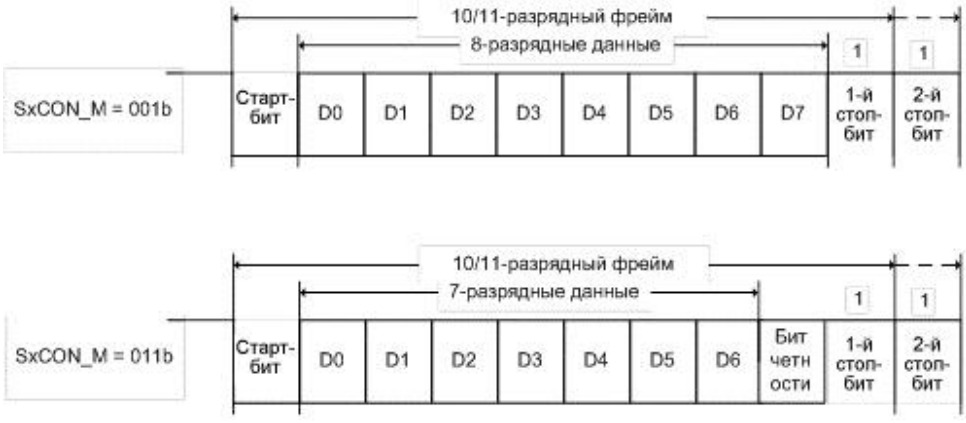


Рисунок 14.4 – Асинхронный 8-битный фрейм (младшим битом вперед)

Бит четности может быть нечетным или четным, в зависимости от бита 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 PEN (проверка всегда выключена в 8-битовом режиме данных). Флаг ошибки четности PE будет установлен наряду с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит самой четности будет сохранен в седьмом разряде регистра SxRBUF.

Асинхронные 9-разрядные фреймы данных

9-разрядные фреймы данных (см. рисунок 14.5) состоят из любых девяти информационных бит D8, …, D0 (M = 100b) или из восьми бит данных плюс автоматически генерируемого бита четности M = 101b.

Рисунок 14.5 – Асинхронный 9-битный фрейм (младшим битом вперед)

Четность может быть четной или нечетной, в зависимости от бита 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 PEN (проверка всегда выключена в 9-битовом режиме данных и режиме пробуждения). Флаг ошибки четности PE будет установлен наряду с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит самой четности будет сохранен в седьмом разряде регистра SxRBUF.

В режиме пробуждения принимаемый фрейм передается в приемный буферный регистр, если только бит 9 (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 младшего бита поля M). 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 fbr, если бит R установлен и данные были загружены в SxTBUF. 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.

Примечание – Для передачи данных вывод TXDx должен быть сконфигурирован как альтернативный выход порта.

Асинхронный прием

Асинхронный прием инициализируется падающим фронтом на линии RXDx, при условии, что биты R и REN установлены. Измерение значения на входе данных RXDx 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. 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 RXDx должен быть сконфигурирован как вход.

Асинхронный прием прекращается очисткой бита 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.

14.3 Синхронные операции

Синхронный режим поддерживает полудуплексную связь, в основном, для простого расширения ввода-вывода через сдвиговые регистры. Данные передаются и принимаются через вывод RXDx, при этом тактовый сигнал выводится через TXDx. Эти сигналы являются альтернативными функциями порта. Для работы в синхронном режиме необходимо установить 000b в битовом поле M. Схема функционирования в синхронном режиме представлена на рисунке 14.6.

8 бит данных передаются и получаются синхронно, тактовыми импульсами, генерируемыми внутренним генератором скорости пересылки. Тактовый генератор выдает тактовый сигнал только в случае передачи или приема данных.

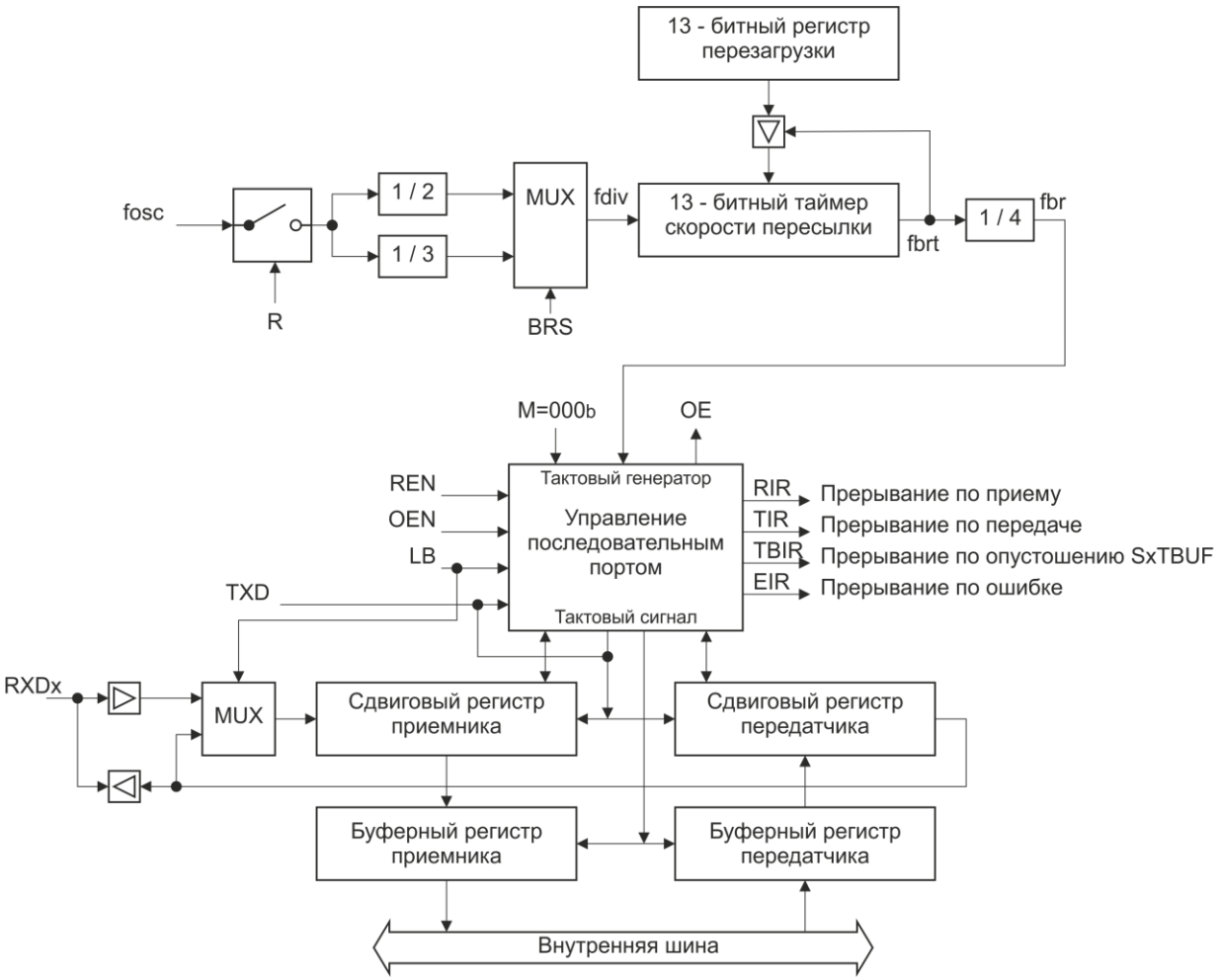


Рисунок 14.6 – Синхронный режим работы интерфейса ASC

Синхронная передача

Синхронная передача начинается в пределах четырех тактов после того, как данные были загружены в буферный регистр передачи SxTBUF, при условии, что R установлен и REN очищен (полудуплексный режим, нет приема).

Примечание – В петлевом режиме (бит LB установлен) бит 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 TBIR запроса на прерывание по очистке передающего буферного регистра. 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 TXDx. После завершения передачи восьмого бита данных выставляется высокий уровень напряжения на выводах RXDx и TXDx, а также, устанавливается флаг TIR запроса на прерывание по передаче и прекращается передача данных.

Примечание – Для правильной работы в этом режиме необходимо правильно сконфигурировать выводы – вывод TXDx должен быть настроен для дополнительного вывода данных, чтобы обеспечить вывод тактового сигнала, а RXDx – должен быть сконфигурирован как вывод во время передачи данных.

Синхронный прием

Синхронный прием инициализируется установкой бита REN. Если бит R установлен, данные принимаются на RXDx и поступают в сдвиговый регистр по импульсам тактового генератора, выводимым на TXDx. После получения последнего бита данные из сдвигового регистра передаются в буферный приемный регистр SxRBUF. Вместе с этим будет установлен флаг RIR запроса на прерывание по приему, после этого бит REN очищается.

Примечание – Вывод TXDx должен быть сконфигурирован как выход.

Синхронный прием данных прекращается после очистки бита 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 EIR запроса на прерывание, а также установлен флаг OE, если установлен бит OEN.

Синхронизация

На рисунке 14.7 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 RXDx по падающему фронту тактового сигнала. В режиме синхронного приема данные принимаются на линии RXDx по возрастающему фронту тактового сигнала. Между двумя последовательными передачами или приемами данных вставлен один тактовый цикл задержки.

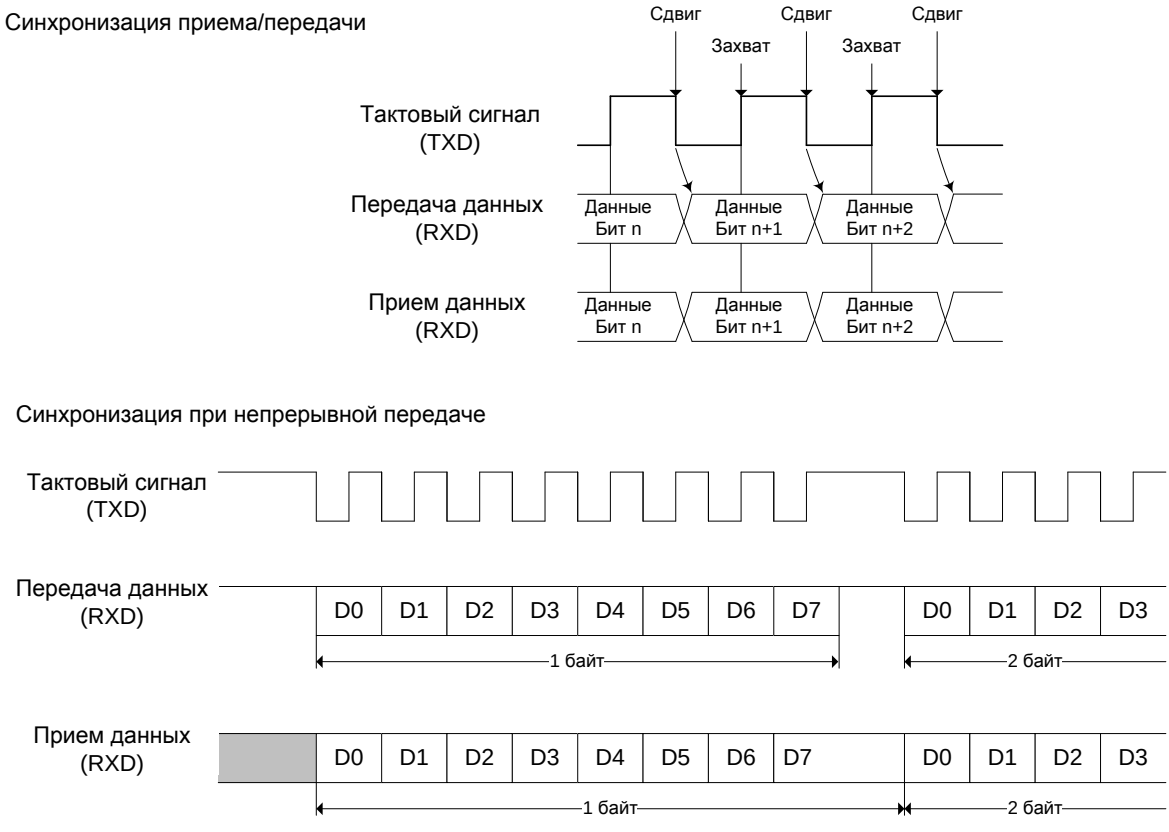


Рисунок 14.7 – Временная диаграмма работы интерфейса ASCx в синхронном режиме приема/передачи

14.4 Генератор скорости пересылки данных

Последовательный интерфейс ASCx 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 Fosc. Счет таймера скорости пересылки данных происходит в обратном направлении и может быть начат или остановлен через бит управления 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 fbr 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 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 BRS и режимом (асинхронный или синхронный).

Регистр SxBG 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 BG установлен. Однако, если R очищен во время операции записи в BG, значение таймера не будет перезагружено до тех пор, пока не будет разрешена работа тактового генератора. Для правильной установки скорости передачи данные в регистр SxBG необходимо записывать только при сброшенном бите R. Запись данных в регистр SxBG при установленном бите R может привести к непредсказуемому поведению ASCx.

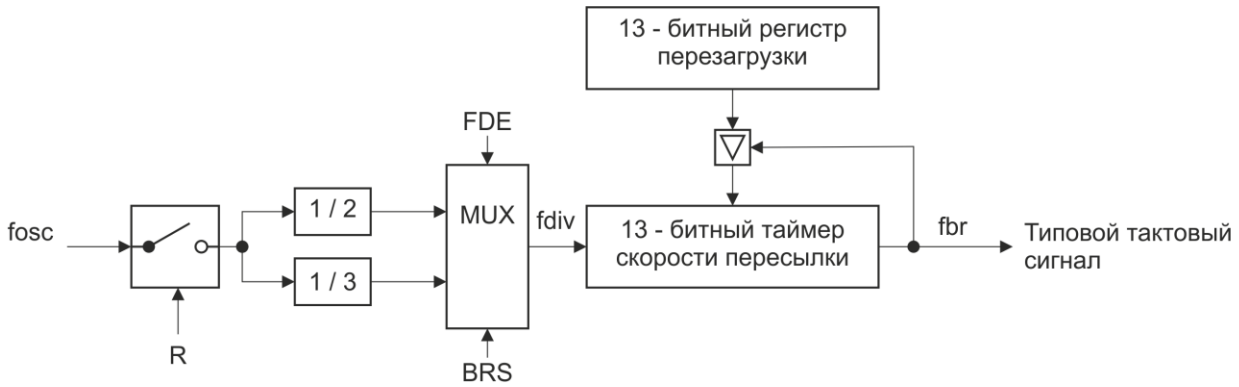
Скорость пересылки данных в асинхронном режиме

Во время асинхронного приема данных измерение значения на входе данных RXDx 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 (см. рисунок 14.8).

Скорость пересылки данных зависит от следующих сигналов, битов и значений регистров:

- входного тактового сигнала Fosc;
- битов FDE и BRS, определяющих частоту тактового сигнала fdiv, формируемого из входного тактового сигнала;
- значения регистра дробного делителя SxFDV, если бит FDE установлен (дробный делитель включен);
- значения 13-битового перезагружаемого регистра SxBG.

Тактовый сигнал скорости пересылки данных fbr получен из тактового сигнала генератора Fdiv путем деления на 16. Регистр дробного делителя SxFDV содержит 9-битовое значение (дробный делитель используется только в асинхронных режимах).



BRS	Выбранный делитель
0	1 / 2
1	1 / 3

Рисунок 14.8 – Генератор скорости передачи в асинхронном режиме

Скорость пересылки для асинхронного режима, когда используется фиксированный тактовый сигнал, SxCON_FDE = 0b, и необходимые значения перезагрузки для данной скорости определяется по формулам из таблицы 14.1.

Таблица 14.1 – Формулы для определения скорости пересылки данных при использовании фиксированного тактового сигнала

FDE	BRS	BG	Формула
0	0	0, …, 8191	$baudrate = \frac{fosc}{32 \times (BG + 1)}$ (14.1) $BG = \frac{fosc}{32 \times baudrate} - 1$ (14.2)
	1		$baudrate = \frac{fosc}{48 \times (BG + 1)}$ (14.3) $BG = \frac{fosc}{48 \times baudrate} - 1$ (14.4)

Максимальная скорость пересылки данных, которая может быть достигнута для асинхронных режимов, используя два фиксированных делителя и входной тактовый сигнал модуля частотой 12,5 МГц, должна быть 390 Кбод.

Использование дробного делителя

Для включения дробного делителя необходимо установить бит SxCON_FDE. Тактовый сигнал F_{DIV} формируется из Fosc. При этом Fosc делится на дробь n/512 для любого значения n от 0 до 511. Если n = 0, то отношение делителя 1, т. е. F_{DIV} = Fosc. Таким образом, дробный делитель позволяет программировать скорость передачи данных с намного большей точностью, чем двумя основными делителями частоты.

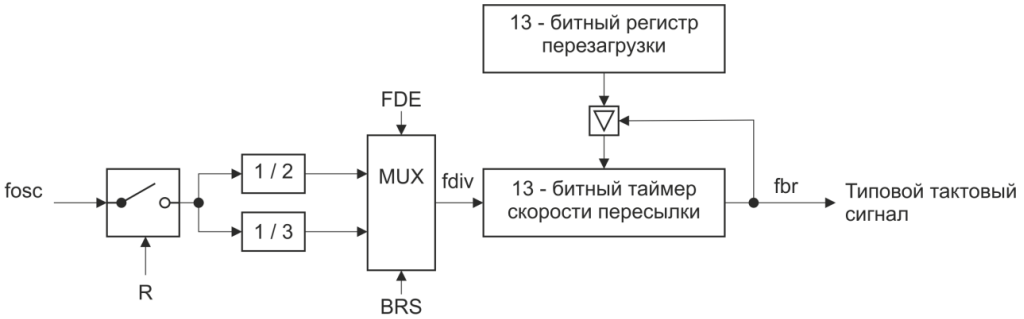
В таблице 14.2 представлены формулы для расчета скорости пересылки данных в асинхронном режиме при использовании дробного делителя.

Таблица 14.2 – Формулы расчета скорости пересылки данных при использовании дробного делителя

FDE	BRS	BG	FDV	Формула
1	–	1, …, 8191	1, …, 511	$\text{baudrate} = \frac{\text{FDV}}{512} \times \frac{\text{fosc}}{16 \times (\text{BG} + 1)}$ (14.5)
			0	$\text{baudrate} = \frac{\text{fosc}}{16 \times (\text{BG} + 1)}$ (14.6)

Скорость пересылки данных в синхронном режиме

В синхронном режиме скорость пересылки данных получается из сигнала генератора скорости пересылки делением на четыре.



BRS	Выбранный делитель
0	1 / 2
1	1 / 3

Рисунок 14.9 – Схема формирования скорости передачи в синхронном режиме

Скорость пересылки данных для синхронного режима может быть определена по формулам в таблице 14.3.

Таблица 14.3 – Формулы для определения скорости передачи данных (baudrate) в синхронном режиме

BRS	BG	Формула
0	1, …, 8191	$\text{baudrate} = \frac{\text{fosc}}{8 \times (\text{BG} + 1)}$ (14.7) $\text{baudrate} = \frac{\text{fosc}}{8 \times (\text{BG} + 1)} - 1$ (14.8)
1		$\text{baudrate} = \frac{f_{CLK}}{12 \times (\text{BG} + 1)}$ (14.9) $\text{baudrate} = \frac{f_{CLK}}{12 \times (\text{BG} + 1)} - 1$ (14.10)

Максимальная скорость пересылки данных, которая может быть достигнута в синхронном режиме, используя входной тактовый сигнал 12,5 МГц, должна быть 1,5 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 FEN не обнаружена единица в любом из ожидаемых стоповых битов. После этого будет установлен флаг FE, указывающий на ошибку формата передаваемых данных (только для асинхронного режима);
- при включенном режиме проверки четности (PEN = 1b) в случае обнаружения неправильного результата. Также будет установлен флаг PE, индицируя ошибку четности (только для асинхронного режима);
- при разрешенном обнаружении ошибки переполнения (OEN = 1b), если полученные данные не считались из приемного буфера до окончания следующего цикла получения данных. При этом устанавливается флаг OE, указывающий на ошибку переполнения (асинхронный и синхронный режимы).

14.5 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 Fosc.

Причина запроса прерывания по ошибке (ошибка синхронизации кадров, ошибка четности и ошибка переполнения) может быть определена по статусным флагам FE, PE или OE.

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.

Генерация прерываний ASC представлена на рисунке 14.10.

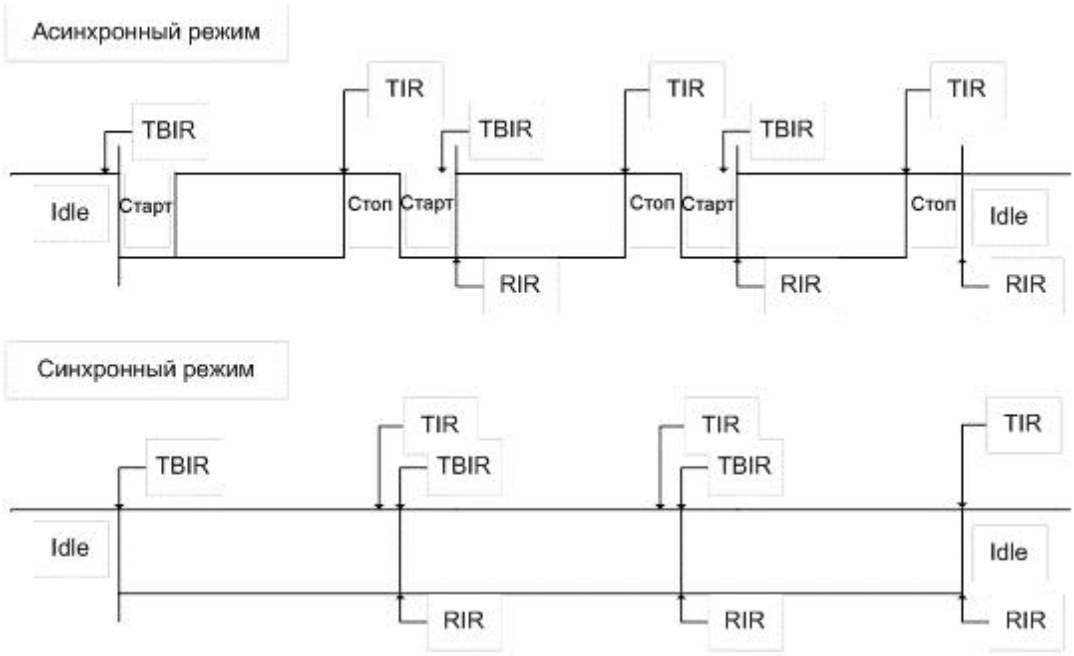


Рисунок 14.10 – Генерация прерываний ASC

Как видно из рисунка 14.10, 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.

15 Синхронные последовательные интерфейсы SSC0 и SSC1

Высокоскоростные последовательные интерфейсы типа SPI (SSC0 и SSC1) поддерживают полнодуплексный и полудуплексный последовательный синхронный обмен данными на скоростях до 6,25 Мбод при частоте входного тактового сигнала блока 12,5 МГц (25 МГц внешняя). Последовательный тактовый сигнал может быть сформирован непосредственно модулем SSC (в режиме мастера) или может быть принят от внешнего источника (в режиме ведомого). 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.

Поскольку модули полностью идентичны (за исключением адресов регистров), то в дальнейшем номер модуля будет заменен символом «x». В названиях регистров номера также буду заменены символом «x».

Особенности интерфейса SSCx:

- функционирование в режиме мастера или ведомого;
- полнодуплексный или полудуплексный обмен данными;
- гибкий формат данных:
- программирование числа бит данных: от 2 до 16 бит;
- программирование передачи байта – младшим или старшим битом вперед;
- программирование полярности тактового сигнала – низкий или высокий уровень на линии при отсутствии тактового сигнала;
- программирование фазы – данные сдвигаются/захватываются по положительному или отрицательному фронту тактового сигнала.
- скорость передачи данных до 6,25 Мбод в режиме мастера и до 3,125 Мбод в режиме ведомого (частота fper = 12,5 МГц);
- генерирование прерываний после передачи/приема данных и при обнаружении ошибок (приема, фазы, скорости, передачи).

Блок-схема высокоскоростного синхронного последовательного интерфейса SSC представлена на рисунке 15.1.

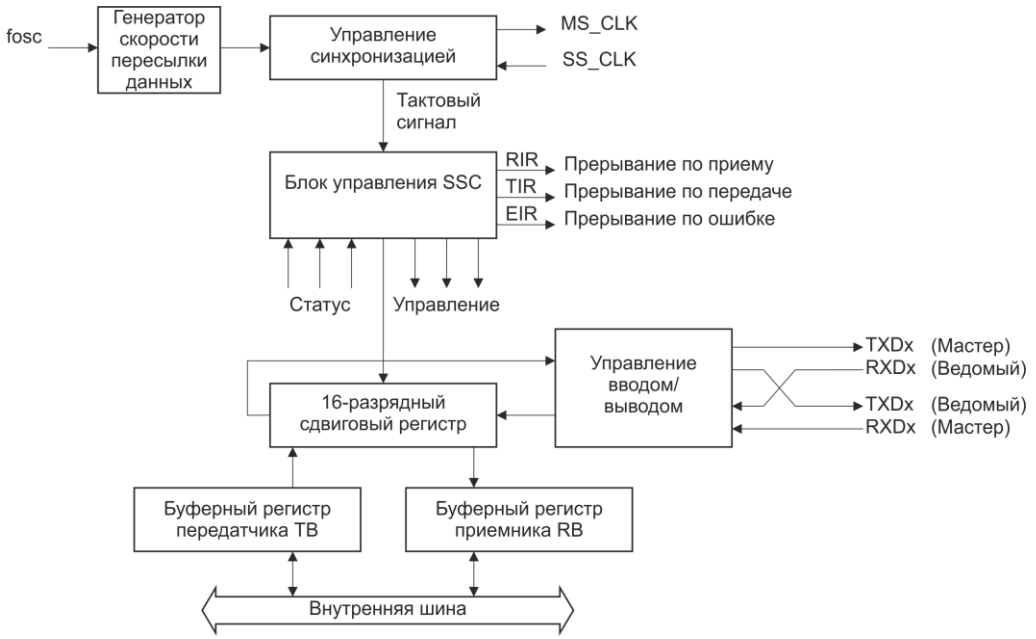


Рисунок 15.1 – Блок-схема высокоскоростного синхронного последовательного интерфейса SSC

Данные передаются и принимаются по линиям TXDx и RXDx, обычно подключаемым к выводам MTSR (мастер передает/ведомый принимает) и MRST (мастер принимает/ведомый передает). Сигнал тактового генератора выводится на линию MS_CLK или вводится через линию SS_CLK. Обе эти линии подключаются к SCLK и являются альтернативными функциями выводов порта.

Режимом работы последовательного интерфейса управляет регистр управления SSCxCON. Этот регистр имеет две функции:

- обеспечение доступа к служебным битам во время программирования модуля SSCx (бит EN сброшен);
- обеспечение доступа к флагам состояния во время работы модуля SSCx (бит EN установлен).

При записи в SSCxCON в зарезервированные биты необходимо записывать нули.

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 SSCxTB и сразу же перемещаются в сдвиговый регистр, если он пуст. Мастер немедленно начинает передачу. Ведомый ждет поступления тактового сигнала.

Когда передача начата, устанавливается флаг занятости BSY и флаг запроса на прерывание по передаче TIR, указывающий на то, что регистр SSCxTB может быть вновь загружен. После передачи запрограммированного числа битов (от двух до 16) содержимое сдвигового регистра перемещается в приемный буферный SSCxRB, и выставляется флаг RIR запроса на прерывание по приему. При отсутствии очередных данных для передачи (регистр SSCxTB пуст) бит BSY сбрасывается. Флаг BSY устанавливается и сбрасывается только аппаратно.

Буферный регистр передатчика – SSCxTB – содержит данные, которые необходимо передать. Буферный регистр приемника – SSCxRB – содержит принятые данные.

Примечание – При использовании нескольких модулей последовательного интерфейса только один модуль SSCx может быть мастером.

Настройку последовательной передачи данных можно производить в широком диапазоне:

- данные могут содержать от 2 до 16 бит;
- передача может идти как старшим, так и младшим битом вперед;
- на линии тактового сигнала в режиме ожидания может быть как низкий, так и высокий уровень сигнала;
- данные могут передаваться по переднему или по заднему фронту тактового сигнала;
- скорость передачи данных может быть установлена от 152,53 бод до 10 Мбод (при частоте входного тактового сигнала fosc = 20 МГц);
- тактовый сигнал может генерироваться модулем SSCx (в режиме мастера) или приниматься от внешнего источника (в режиме ведомого).

Вне зависимости от направления передачи данных (старшим или младшим битом вперед), передаваемые данные всегда выстраиваются в регистрах SSCxTB и SSCxRB в правильном порядке. С помощью логики сдвигового регистра биты данных перераспределяются для передачи. Значения в невыбранных битах регистра SSCxTB игнорируются, а значения в невыбранных битах регистра 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 – для захвата данных (управляющий бит – PH). Бит PO задает уровень сигнала на тактовом выходе в режиме ожидания, см. рисунок 15.2.

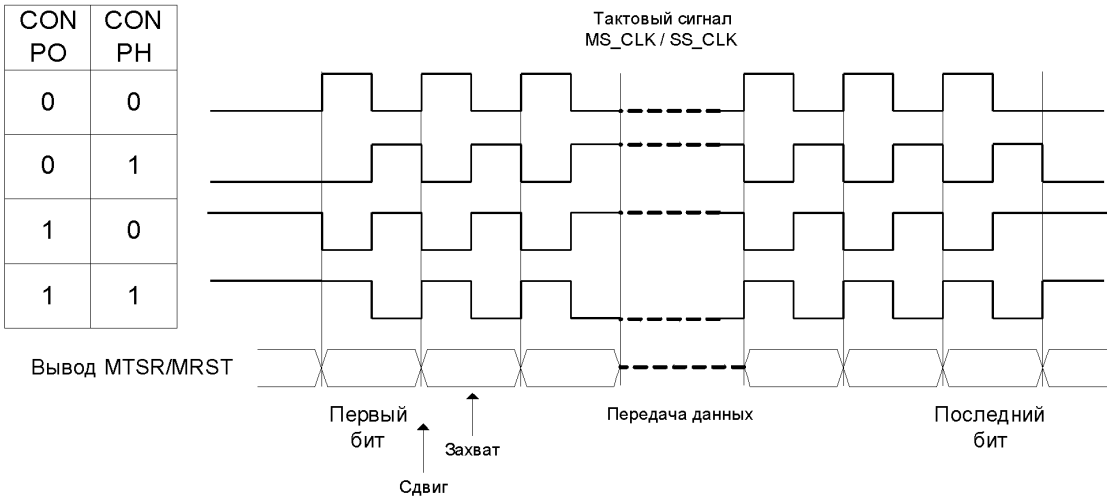


Рисунок 15.2 – Полярность и фаза тактового сигнала

15.1 Дуплексный режим

Обмен информацией и тактирование передач осуществляется посредством трех выводов микроконтроллера, через которые он подключается к линиям шины. Режимы работы линий всегда задает мастер. 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 мастер. Все остальные устройства работают в режиме ведомых и могут только принимать тактовый сигнал (см. рисунок 15.3).

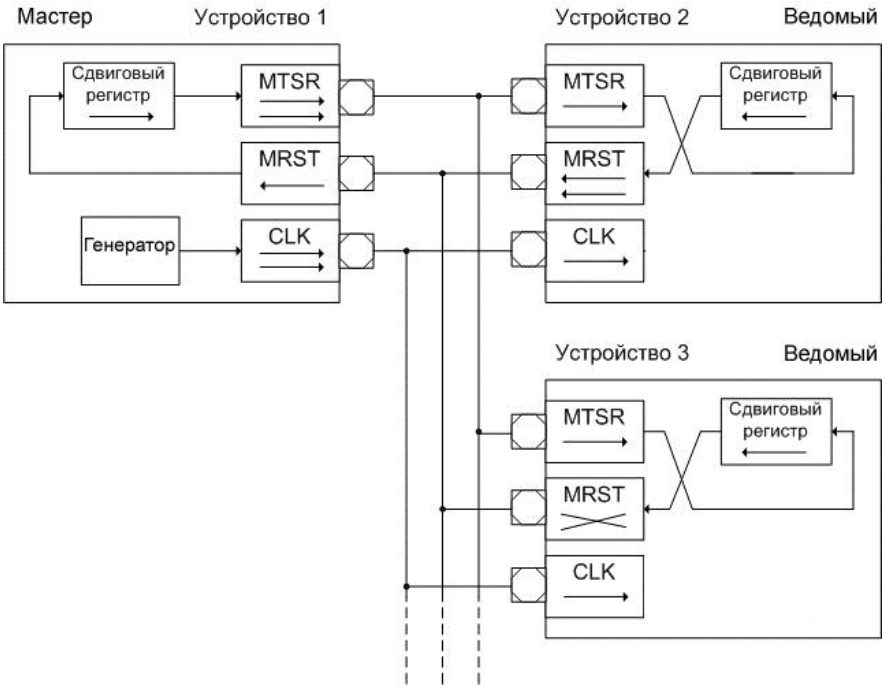


Рисунок 15.3 – Модуль SSCx в полнодуплексном режиме работы

Все выходные линии MRST всех ведомых должны быть объединены и подключены к линии приема данных. Конфликты на шине разрешаются двумя способами:

- только одно ведомое устройство включает драйвер передачи для линии MRST. Для всех других устройств вывод MRST программируется на вход. Таким образом, только

одно ведомое устройство может выставлять данные на линию. Мастер выбирает ведомого, от которого ожидает поступление данных либо с помощью независимых линий выбора, либо с помощью специальных команд, посланных ведомому. После этого выбранный ведомый переключает линию MRST на выход и находится в таком состоянии до тех пор, пока не получит сигнала или команды отключения;

- выходы всех ведомых устройств объединяются по принципу монтажного И с внешней подтяжкой до единицы. Все невыбранные ведомые удерживают свои выходы в единицах. Мастер выбирает ведомого либо с помощью специальной команды, либо с помощью независимых линий выбора.

После инициализации модуль SSCx может быть включен.

Мастер может начать первую передачу данных после записи данных в регистр SSCxTB. Это значение скопируется в регистр сдвига (если он пустой), и выбранный первый бит данных будет перемещен на линию TXDx по следующему импульсу тактового сигнала контроллера скорости передачи (передача начинается, только если установлен бит EN). В зависимости от выбранной фазы импульс тактового сигнала также будет сгенерирован на линии MS_CLK. По противоположному фронту синхроимпульса мастер захватит данные, обнаруженные на входной линии RXDx. Таким образом, происходит обмен передаваемых данных с принимаемыми данными. 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 ведомого, первый бит сразу поступает на выход RXDx, не дожидаясь первого тактового импульса от мастера. Бит BSY не будет установлен до тех пор, пока не появится передний фронт синхроимпульса.

Примечание – Прием и передача данных всегда происходят одновременно, независимо от правильности передаваемых или принимаемых данных. В этом заключается одно из отличий модуля SSCx от модуля ASCx.

Полудуплексный режим

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.

Мастер управляет передачей данных при помощи собственного тактового сигнала. Ведомый 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;

- подключение выходов всех устройств по принципу монтажного И.

Так как входы и выходы данных соединены вместе, то передаваемые устройством данные подаются на его собственный вход MRST в режиме мастера и вход MTSR – в режиме ведомого. Поэтому имеется возможность детектирования искажения данных на общей линии передачи.

Рисунок 15.4 иллюстрирует работу модуля SSCx в полудуплексном режиме работы.

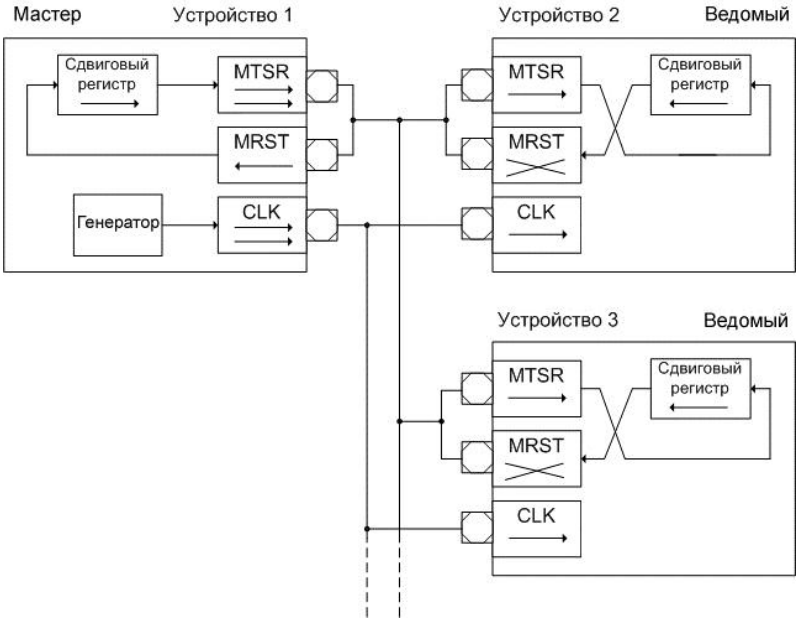


Рисунок 15.4 – Интерфейс SSC в полудуплексном режиме работы

15.2 Непрерывные передачи

Когда флаг запроса прерывания по передаче установлен, это указывает на то, что буфер передачи SSCxTB 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 байт 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.

15.3 Управление скоростью передачи

Модуль SSCx имеет свой собственный 16-разрядный генератор скорости передачи с возможностью перезагрузки содержимым 16-разрядного регистра, позволяющий формировать синхросигнал с необходимой частотой независимо от таймеров (см. рисунок 15.5).

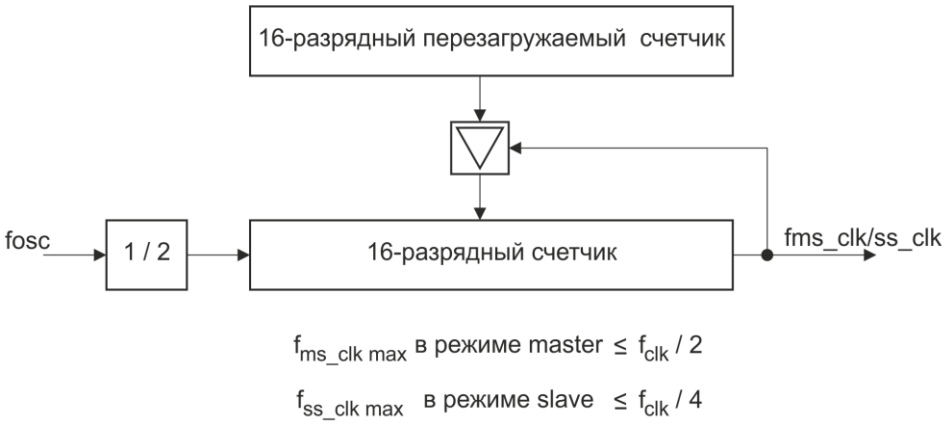


Рисунок 15.5 – Генератор скорости передачи данных

Генератор скорости передачи тактируется внешним тактовым сигналом Fosc. Отсчет таймера ведется в обратном направлении. Регистр SSCxBR одновременно является регистром скорости пересылки и регистром перезагрузки. Чтение SSCxBR во время, когда модуль SSCx выключен возвращает значение перезагрузки. Новое значение перезагрузки может быть записано в регистр в режиме программирования модуля. Не следует производить запись в регистр SSCxBR во время работы модуля.

Перезагружаемый регистр генератора скорости пересылки

Формулы (15.1), (15.2) позволяют вычислить скорость пересылки данных для запрограммированного значения перезагрузки или необходимое значение перезагрузки для желаемой скорости пересылки.

$$\text{baudrate} = \frac{\text{fosc}}{2 \times (<\text{BR}> + 1)}, \qquad (15.1)$$

$$\text{BR} = \frac{\text{fosc}}{2 \times \text{baudrate}} - 1 \qquad (15.2)$$

«BR» представляет собой содержимое перезагружаемого регистра, взятое как 16-разрядное целое число без знака, а скорость пересылки, представленная на рисунке 15.2, равна fmc_clf/ss_clk.

Максимальная скорость пересылки данных, которая может быть запрограммирована при fosc = 12,5 МГц и BR = 0000h:

- в режиме мастера – 6 Мбод
- в режиме ведомого – 3,125 Мбод.

15.4 Механизм определения ошибок

Логика модуля SSCx может обнаружить четыре различных типа ошибок (см. рисунок 15.6).

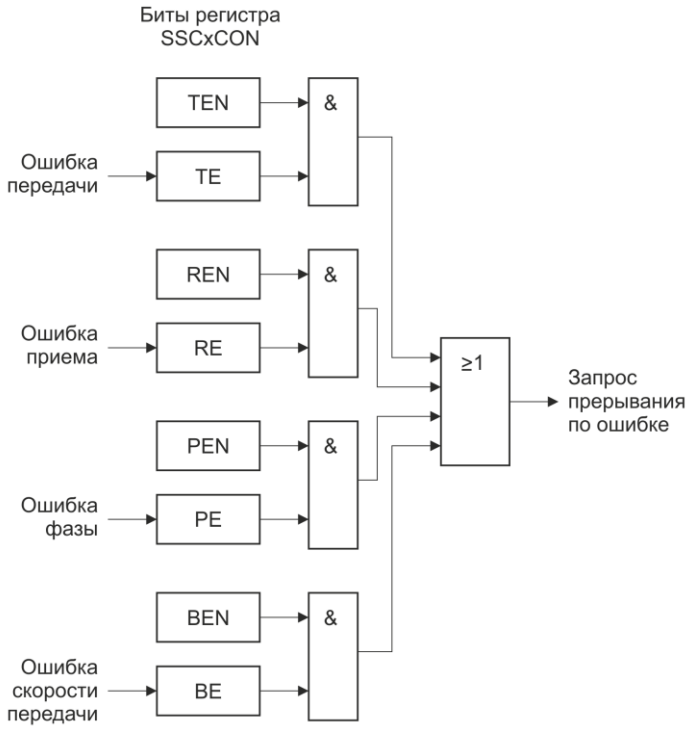


Рисунок 15.6 – Управление прерываниями

Ошибка приема и ошибка фазы определяются во всех режимах работы. Ошибка передачи и ошибка скорости передачи определяются только в режиме ведомого. 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 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. Кроме того, подпрограмма обработки прерываний должна очищать сопоставленный флаг ошибки для предотвращения генерирования повторного запроса прерывания.

Ошибка приема

Детектируется, если предыдущие данные не были считаны из буфера приема SSCxRB до завершения приема новых данных. В этом случае устанавливается флаг RE. Запрос прерывания по ошибке вырабатывается в том случае, если установлен бит REN. После этого в приемный буфер SSCxRB будет записано новое значение данных, и старое значение будет потеряно.

Ошибка фазы

Детектируется, когда сигнал на входе MRST (в режиме мастера) или входе MTSR (в режиме ведомого) 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 PE.

Ошибка скорости передачи

Детектируется в режиме ведомого 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 BE и, если установлен бит BEN, выставляется запрос на прерывание по ошибке EIR. Использование этой функции позволяет определять дополнительные ложные или пропущенные такты на линии передачи тактового сигнала.

Примечание – Если на момент возникновения ошибки бит REN установлен, то автоматически производится сброс модуля SSCx. Это необходимо для повторной инициализации модуля SSCx в том случае, когда обнаруживается слишком много или слишком мало тактовых импульсов.

Ошибка передачи

Детектируется режиме ведомого в том случае, когда мастер инициализировал передачу данных (послал тактовый сигнал), а значение в буфере передачи SSCxTB ведомого не было изменено с последней передачи. При этом устанавливается флаг TE и, если установлен бит TEN, генерируется запрос на прерывание по ошибке EIR. Если передача начинается без записи нового значения в буфер передачи, ведомый будет передавать 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 ведомый не выбран для передачи.

Примечание – Все ведомые, не выбранные для передачи, должны содержать в своих буферах передачи значение FFFFh.

Прерывание может быть сгенерировано установкой флага в регистре SSCxCON.

Примечание – В отличие от бита EIR, флаги TE, RE и BE не сбрасываются аппаратно, и поэтому должны очищаться программно перед выходом из подпрограммы обслуживания прерываний.

16 Контроллер интерфейса I2C

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 протоколами SMBus 1.1 и SMBus 2.0, ACCESS.Bus, I2C 2.1;
- поддержка скоростного/стандартного (FS) и высокоскоростного (HS) режимов;
- программирование действий мастера/ведомого;
- возможность подключения к шине нескольких ведущих устройств, т.е. поддержка режима мультимастер (MM);
- один программно задаваемый адрес;
- 7- или 10-битная адресация ведомого;
- поддержка адреса общего вызова.

Особые возможности SMBus:

- отслеживание времени простоя линии SCL;
- наличие функции отслеживания ошибок в пакетах данных (PEC) с использованием метода расчета контрольной суммы (CRC);
- поддержка адреса отклика мастера;
- поддержка полинга и контроля прерываний.

16.1 Протокол шины

Протокол I2C использует двухпроводной интерфейс для двусторонней связи между устройствами, подключенными к шине. Двунаправленная шина состоит из двух линий: данных SDA и тактового сигнала 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 мультимастер, в котором шина может контролироваться одним или несколькими устройствами из подключенных к шине. Каждое устройство, подключенное к шине, имеет свой адрес и может быть как приемником, так и передатчиком (некоторые только приемниками).

Операции с данными

Устройство, которое начинает передачу данных, становится мастером. Мастер генерирует тактовый сигнал SCL, а также инициирует и завершает передачу данных по шине. За один такт сигнала SCL передается один бит данных по линии SDA (рисунок 16.1).

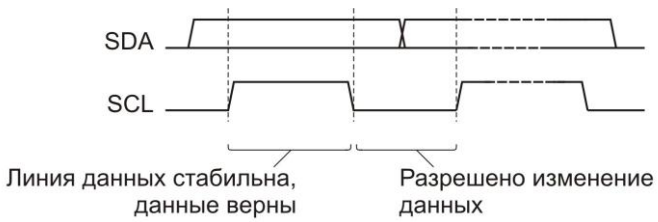


Рисунок 16.1 – Передача бита данных

Данные валидны (верны), пока уровень сигнала на линии SCL высокий. Когда на линии SCL низкий уровень сигнала, данные могут меняться.

Старт и стоп

Состояние старта формируется тогда, когда на линии SCL держится высокий уровень сигнала, а на линии SDA возникает перепад уровня сигнала из высокого в низкий.

Состояние стопа (останова) формируется тогда, когда на линии SCL держится высокий уровень сигнала, а на линии SDA возникает перепад уровня сигнала из низкого в высокий (см. рисунок 16.2).

Состояния старта и стопа формирует только мастер.

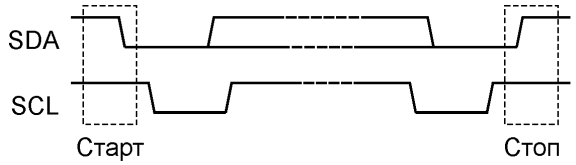


Рисунок 16.2 – Состояния старта и стопа

После того, как сформировано состояние старта, шина считается занятой и другие устройства не должны пытаться управлять ей. Шина считается занятой до тех пор, пока не будет сформировано состояние стопа. В середине передачи может быть сформировано состояние повторного старта, если мастеру нужно обратиться к другому ведомому или если требуется изменение направления передачи данных без потери контроля над шиной (см. рисунок 16.3).

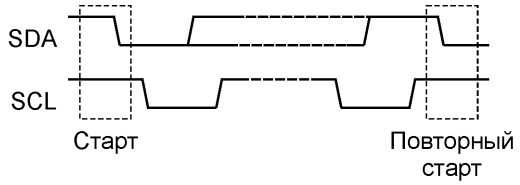


Рисунок 16.3 – Состояние повторного старта

Арбитраж

Арбитраж выполняется в момент времени, когда на линии SCL находится «1». Два устройства могут сгенерировать стартовое состояние в одно и то же время. Далее арбитраж будет продолжаться до тех пор, пока одно из устройств сформирует «0», а другое – «1» на линии SDA. Устройство, которое установило «1» на линии SDA, проигрывает арбитраж. На рисунке 16.4 приведен пример арбитража.

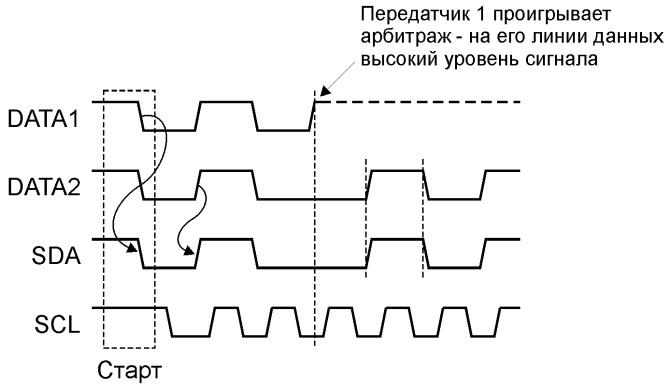


Рисунок 16.4 – Арбитраж на линии SDA

Два устройства передают свои данные DATA1 и DATA2 на линию SDA. В момент времени, когда очередной бит данных DATA1 равен «1», а бит данных DATA2 равен «0»,

второе устройство выигрывает арбитраж и продолжает передачу своих данных, а первое устройство прекращает передачу.

Если устройство проигрывает арбитраж во время передачи первого байта после старта (во время передачи адреса ведомого), оно становится ведомым приемником и мониторит передаваемый адрес на случай совпадения. Арбитраж также может быть проигран в режиме мастера приемника во время квитирования или в режиме ведомого передатчика во время ответа на адрес отклика на сигнал предупреждения.

В случае проигрывания арбитража в битовом поле MODE регистра SMBST устанавливается соответствующий код и генерируется прерывание.

Синхронизация

Синхронизация тактовых сигналов разных устройств, подключенных к шине I2C, реализуется в случаях, когда несколько устройств являются мастерами, и выполняется с использованием той особенности, что линия SCL реализована как монтажное «И» линий тактовых сигналов этих устройств. Для примера рассмотрим синхронизацию двух мастеров с линиями тактовых сигналов CLK1 и CLK2 (см. рисунок 16.5).

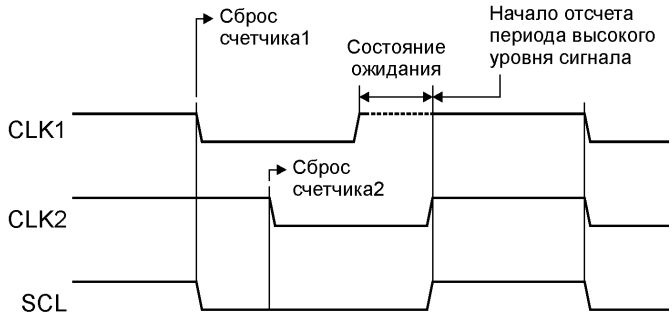


Рисунок 16.5 – Синхронизация

Линия SCL переводится в состояние «0» сразу, как только один из мастеров выставляет на своей линии тактового сигнала низкий уровень сигнала (CLK1 на рисунке 16.5). При этом его внутренний счетчик длительности низкого уровня сигнала сбрасывается и начинает отсчет. Второй мастер выставляет низкий уровень позже, и его счетчик также сбрасывается (CLK2).

Как только внутренний счетчик первого мастера переполнится, мастер выставит на линии CLK1 высокий уровень сигнала. Тем не менее, линия SCL будет по-прежнему оставаться в состоянии «0», удерживаемая вторым мастером. В связи с этим, первый мастер перейдет в состояние ожидания (см. рисунок 16.5). Когда переполнится счетчик второго мастера, он выставит на линии CLK2 высокий уровень сигнала, и в этот момент линия SCL перейдет в состояние «1». С этого момента внутренние счетчики длительности высокого уровня сигнала обоих мастеров начнут синхронный отсчет.

Каждая передача данных состоит из начального состояния «старт», состояний передач битов и состояния «стоп». Данные передаются старшим битом (MSB) вперед. Передача каждого байта завершается квитированием, т.е. приемник подтверждает окончание приема сигналом подтверждения (ACK). Ведомое устройство 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/байта (см. рисунок 16.6).

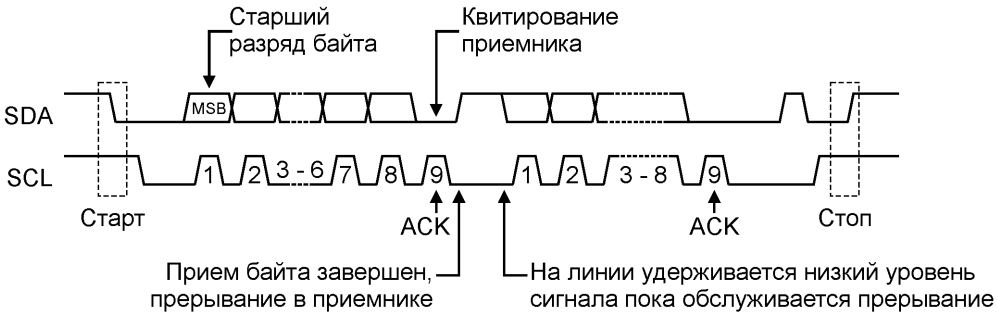


Рисунок 16.6 – Передача данных

Квитирование

Каждый байт посылки должен быть завершен квитированием, т.е. ответом на прием сигнала запроса подтверждения приема (ACK). На рисунках 16.6 и 16.7 показано положение момента квитирования в пределах посылки.

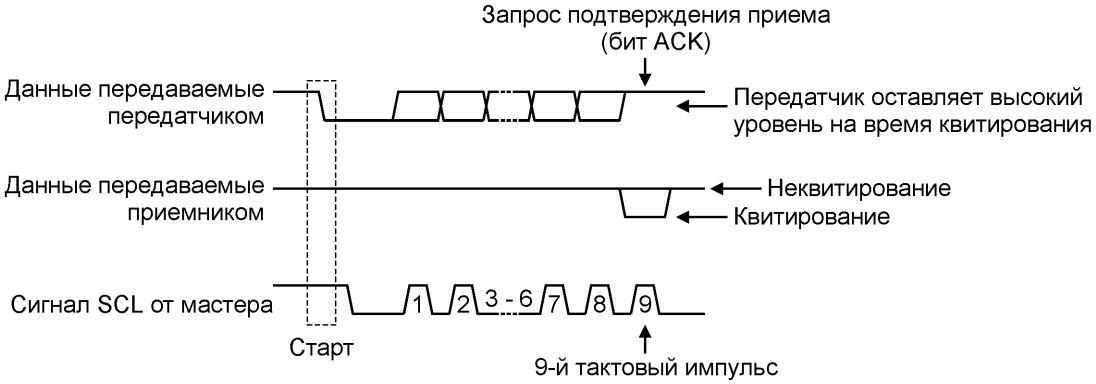


Рисунок 16.7 – Квитирование и неквитирование бита подтверждения ACK

Бит запроса подтверждения приема генерируется мастером. Передатчик (мастер или ведомый) в момент девятого такта синхросигнала оставляет линию SDA в состоянии «1» (бит ACK). В свою очередь, приемник должен сбросить линию SDA в «0» в течение времени, пока на линии SCL удерживается высокий уровень девятого импульса тактового сигнала, т.е. квитировать прием (см. рисунок 16.7). Если приемник не отвечает на запрос подтверждения и не подтверждает прием байта, то он оставляет линию SDA без изменений в состоянии «1», т.е. не квитирует прием.

Примечание – Все устройства, подсоединенные к шине I2C, в обязательном порядке должны квитировать бит ACK при получении байта с их собственным адресом. Этот механизм используется для отслеживания наличия отключившихся (самостоятельно или по каким-то причинам) от шины устройств.

Ведомое устройство имеет право не квитировать бит ACK в следующих случаях:

- если ведомый не может принять данные или он занят. Мастер, обнаружив неквитирование байта, должен сгенерировать состояние стопа и прервать передачу. Как альтернатива, ведомый может затянуть период низкого уровня сигнала тактирования на линии SCL для завершения своих операций и продолжить передачу;
- если ведомый обнаружил некорректную команду или некорректные данные. В этом случае, ведомый должен не квитировать принятый байт. Мастер, обнаружив неквитирование байта, должен сгенерировать состояние стопа и повторить передачу;
- если мастер функционирует как приемник, то, приняв байт, он должен сообщить ведомому об окончании данных неквитированием бита ACK, посланного ведомым. После этого ведомый передатчик должен освободить линию SDA для того, чтобы мастер смог сгенерировать состояние завершения передачи (состояние стопа).

Формат передачи данных с 7-битной адресацией

На рисунке 16.8 показана передача адреса и двух байт данных. Каждому устройству, подключенному к шине, присваивается уникальный 7-битный адрес. Первые семь бит, передаваемые после старта, представляют собой адрес ведомого, восьмой бит (R/W#) определяет направление передачи – от ведомого (чтение, если R/W# = «1») или к ведомому (запись, если R/W# = «0»).

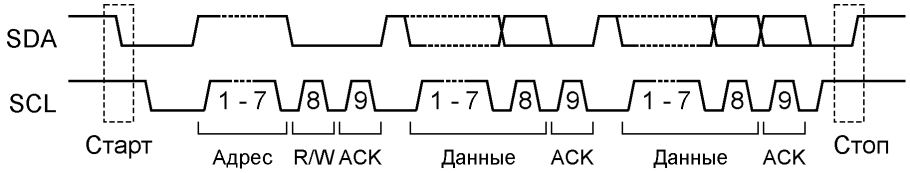


Рисунок 16.8 – Передача данных с 7-битной адресацией

Каждый ведомый, получивший байт адреса, сравнивает его со своим собственным адресом. Если адрес распознается как «свой», ведомый квитирует прием и далее, в зависимости от состояния бита R/W#, становится передатчиком или приемником.

Протокол SMBus/I2C позволяет генерировать адрес общего вызова для одновременного обращения ко всем устройствам, подключенным к шине. Первым передается адрес общего вызова (00h), затем следует байт назначения общего вызова. Ведомые, которые ожидают данные, квитируют этот байт и становятся приемниками, остальные игнорируют общий вызов.

Протокол SMBus/I2C поддерживает уникальную функцию – распознавание адреса отклика на сигнал предупреждения (Alert Response Address – ARA). В системах с несколькими ведомыми каждое устройство может послать мастеру сигнал предупреждения. Для этого используется дополнительная третья линия ALERT#, физически идентичная линиям SDA и SCL, реализованная по принципу монтажное «И». К этой линии также подключаются все устройства. Когда какому-то ведомому (или нескольким ведомым) необходимо обратиться к мастеру, он (или они) выставляет на линии ALERT# низкий уровень сигнала – это сигнал предупреждения (см. рисунок 16.9).

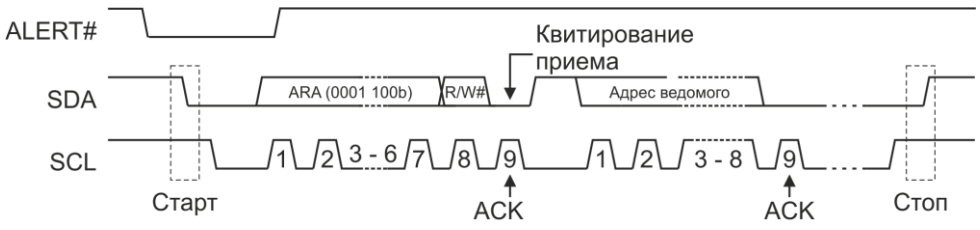


Рисунок 16.9 – Передача адреса отклика на сигнал предупреждения

Мастер, обнаружив «0» на линии ALERT#, обращается ко всем ведомым, посылая адрес отклика на сигнал предупреждения (ARA). Адрес состоит из семи битов (0001_100b) и бита R/W# = «1» (чтение). Ведомый, который отправил сигнал предупреждения, получив ARA, квитирует его и затем отправляет свой 7-битный адрес (восьмой бит может быть как «0», так и «1»), сообщая таким образом ведомому, какое именно устройство посылало сигнал предупреждения. Кроме этого, ведомый, который выставлял «0» на линии ALERT#, должен перестать удерживать линию, чтобы на ней установился высокий уровень сигнала. В том случае, если несколько устройств посылали сигнал предупреждения, то после получения ARA, свой адрес передает то устройство, которое захватывает шину по стандартным правилам арбитража. Если после обслуживания ведомого мастер все еще обнаруживает на линии ALERT# низкий уровень сигнала, он понимает это как то, что сигнал предупреждения посылался несколькими

ведомыми. Мастер снова отправляет ARA и затем общается со следующим ведомым. Появление на линии ALERT# высокого уровня сигнала означает, что все ведомые, которые требовали обращения, обслужены.

Примечание – Описываемый в настоящем РП модуль I2C не имеет выделенной линии ALERT#. При необходимости, пользователь может задействовать свободный вывод микроконтроллера и программно реализовать возможность передачи сигнала предупреждения от ведомого к мастеру. В свою очередь, функция распознавания адреса отклика (ARA) и последующей отправки собственного адреса реализована полностью. Включить функцию можно установкой бита SMBARE в регистре SMBCTRL1.

Формат передачи данных с 10-битной адресацией

10-битная адресация позволяет адресовать до 1 024 ведомых устройств, с использованием резервной комбинации 1111_0xxb, которая передается по линии SDA сразу после старта. 10-битный формат полностью совместим с 7-битным форматом и может использоваться одновременно с ним, что позволяет соединять по шине I2C устройства с разной адресацией.

Основной идеей формата является передача 10-битного адреса в двух первых байтах, следующих сразу после старта. В первом байте передается значение 1111_0xxb, где «xx» – это два старших бита адреса и бит R/W# (на рисунке 16.10 обозначен символом «W» – запись), который должен быть равен «0», чтобы ведомый понял, что в следующем байте будут переданы остальные 8 бит адреса. Во втором байте передаются 8 бит адреса (см. рисунок 16.10).

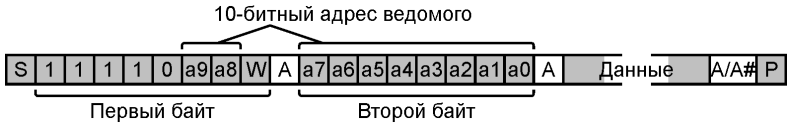


Рисунок 16.10 – Передача данных ведомому с 10-битным адресом (для расшифровки обозначений, применяемых на рисунке, следует обратиться к таблице 16.1)

Чтобы осуществить чтение ведомого, которого адресует мастер, после второго байта адреса следует отправить бит повторного старта и затем комбинацию 1111_0xxb и бит R/W# (обозначен символом «R» – чтение), который на этот раз равен «1» (см. рисунок 16.11).

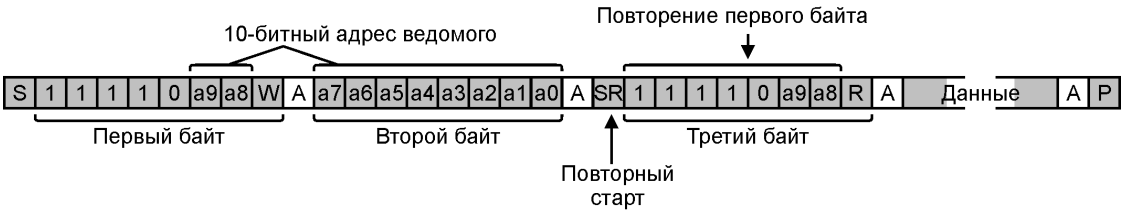


Рисунок 16.11 – Получение данных от ведомого с 10-битным адресом

На рисунках 16.10 и 16.11 биты посылки условно обозначены буквами S, W и др., или состояния битов указаны непосредственно «0» или «1». В дальнейшем на подобных рисунках, поясняющих содержимое посылки при передаче или приеме данных, будут применяться такие же и другие обозначения. Все обозначения, которые будут использоваться, указаны в таблице 16.1 с подробными пояснениями.

Таблица 16.1 – Условные обозначения, принятые на рисунках, показывающих содержимое посылок данных или адресов на линии SDA

Обозначение	Расшифровка обозначения
S	Состояние старта. Символом «S» обозначается стартовый бит посылки
SR	Состояние повторного старта
R/W	Бит указания направления передачи. В тексте настоящего описания он упоминается как R/W#. Наличие этого обозначения в бите посылки указывает на то, что этот бит может быть равен как «0», так и «1»
R	Частный случай обозначения бита направления передачи R/W#. Если в обозначении бита стоит символ «R», то это указывает на то, что в данной посылке бит R/W# должен быть равен «1», т.е. направление передачи данных происходит от ведомого к мастеру (чтение)
W	Частный случай обозначения бита направления передачи R/W#. Если в обозначении бита стоит символ «W», то это указывает на то, что в данной посылке бит R/W# должен быть равен «0», т.е. направление передачи данных происходит от мастера к ведомому (запись)
A/A#	Бит квитированого/неквитированого приема, посылаемый приемником в ответ на запрос передатчика подтвердить прием. Наличие этого обозначения в бите посылки указывает на то, что этот бит может быть равен как «0», так и «1»
A	Частный случай обозначения бита A/A#. Если в обозначении бита стоит символ «A», то это указывает на то, что в данной посылке в ответ на запрос подтверждения приема байта произошло квитирование, т.е. приемник установил линию SDA в «0». В тексте настоящего описания квитированный бит запроса подтверждения приема обозначается как ACK
A#	Частный случай обозначения бита A/A#. Если в обозначении бита стоит символ «A#», то это указывает на то, что в данной посылке в ответ на запрос подтверждения приема байта произошло неквитирование, т.е. приемник не изменил линию SDA и оставил ее в состоянии «1». В тексте настоящего описания, неквитированный бит запроса подтверждения приема обозначается как NACK
P	Состояние окончания передачи. Символом «P» обозначается стоповый бит посылки
Код мастера	8-битный код мастера. Значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств
Адрес	7-битный адрес ведомого, передаваемый мастером
Адрес ведомого	Адрес ведомого, передаваемый во втором байте посылки. В режиме HS это 7-битный адрес, на что указывает идущий следом бит R/W#, в остальных случаях это восемь младших бит 10-битного адреса
Данные	Байт или несколько байт данных
GC	Байт адреса общего вызова (0000_0000b)
AR	Адрес отклика (0001_100b)
	Изображение числа в овале с линией, прикрепляющей его к изображению передачи битов, обозначает код операции и указывает момент, в который этот код записывается в поле MODE регистра SMBST
Цветное поле	Серым цветом обозначены биты, передаваемые от мастера к ведомому
Белое поле	Белым цветом обозначены биты, передаваемые от ведомого к мастеру

16.2 Функциональное описание

Структурная схема модуля I2C представлена на рисунке 16.12. Далее приводится краткое описание назначения блоков модуля.

Входные и выходные каскады линий SDA и SCL

Для обеих линий используются входные шумовые фильтры. В режиме FS эти фильтры подавляют любые импульсы входного сигнала, длительность которых не превышает один такт системного синхросигнала. Выходные каскады включают в себя понижающие (до уровня «0») устройства с открытым стоком. Функционирование входных и выходных каскадов зависит от состояния модуля I2C, т.е. включен или выключен.

Управление режимом работы и опрос состояния

Управление модулем осуществляют блоки управления режимом работы и скоростью передачи, регистров управления и состояния. В состав этих блоков входят следующие регистры:

- SMBST. Содержит биты, отражающие текущую конфигурацию модуля I2C (мастер или ведомый, передатчик или приемник) и бит флага прерывания;
- SMBCST. Является одновременно регистром управления шиной и регистром состояния шины;
- SMBCTRL1. Управляет генерированием состояний старта, повторного старта и останова, а также квитированием;
- SMBCTRL2 и SMBCTRL3. Устанавливают параметры тактового сигнала в режиме мастера и контролируют режим 10-битной адресации;
- SMBSDA. Осуществляет последовательный прием/передачу данных модуля интерфейса I2C;
- SMBADDR. Содержит собственный адрес модуля интерфейса I2C.

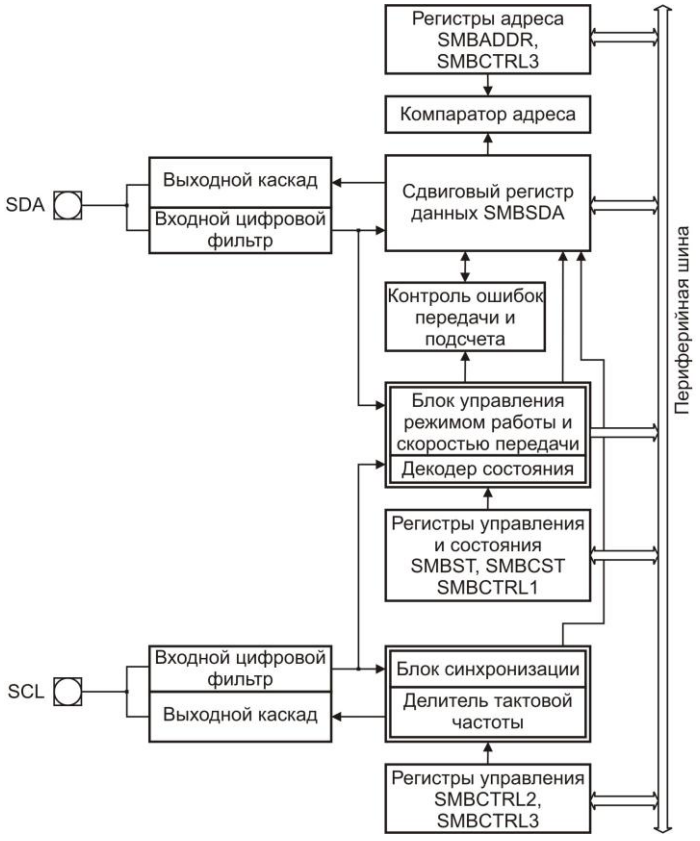


Рисунок 16.12 – Структурная схема модуля I2C

Регистры адреса и компаратор адреса

В регистр адреса SMBADDR может быть записан 7-битный адрес, который является адресом устройства при работе его в режиме ведомого. Распознавание адреса включается установкой бита SAEN.

Компаратор адреса сравнивает принятый 7-битный адрес со значением, хранящимся в поле ADDR. Если разрешено распознавание адреса общего вызова (установлен бит GCMEN регистра SMBCTRL1), то компаратор сравнивает принятый адрес со значением 0000_000b. Если разрешено распознавание адреса отклика на сигнал предупреждения (установлен бит SMBARE регистра SMBCTRL1), то компаратор сравнивает принятый адрес со значением 0001_100b.

Если включен режим 10-битной адресации (одновременно установлены биты SAEN и S10EN регистров SMBADDR и SMBCTRL3 соответственно), компаратор сравнивает старшие пять битов первого полученного байта со значением 1111_0b, а следующие два бита со значением второго и первого битов поля S10ADR регистра SMBCTRL3. Старший бит второго полученного байта сравнивается со значением нулевого бита поля S10ADR, а оставшиеся семь битов – со значением битового поля ADDR регистра SMBADDR (см. рисунок 16.13).

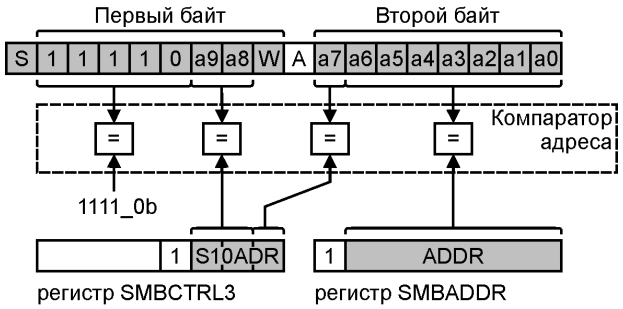


Рисунок 16.13 – Компаратор адреса в режиме 10-битной адресации

Сдвиговый регистр данных

Регистр SMBSDA представляет собой сдвиговый регистр, используемый для приема и передачи данных. Старший бит регистра передается/принимается первым, младший бит – последним. Запись в регистр SMBSDA возможна только, если установлен бит INT регистра SMBST. Регистр может быть прочитан в любой момент времени, но прочитанные данные будут гарантированно достоверными только при установленном бите INT. Регистр SMBSDA не очищается при сбросе и хранит случайные данные до тех пор, пока не будет перезаписан программно или аппаратно после приема байта.

Генерация тактового сигнала и синхронизация

Последовательный тактовый сигнал (выходной сигнал модуля I2C в режиме мастера) формируется генератором на базе системного тактового сигнала (с частотой fosc).

Модуль I2C может функционировать в двух глобальных режимах – стандартном/скоростном (FS) и высокоскоростном (HS).

В режиме FS используется 7-битный предделитель. Значение старших 6 бит определяется значением битового поле SCLFRQ регистра SMBCTRL2, а младший бит всегда равен нулю (деление производится только на четное число). Минимальный и максимальный коэффициенты деления – 8 и 128 соответственно. Блок синхронизации тактового сигнала производит синхронизацию генератора тактового сигнала и выходного сигнала SCL с тактовым сигналом других устройств, подключенных к шине.

В режиме HS используется 4-битный предделитель. Значение старших бит определяется значением битового поля HSDIV регистра SMBCTRL3, младший бит всегда равен нулю.

Определяемое спецификацией протокола SMBus наименьшее время ожидания на линии SCL составляет 25 мс. Если пауза между двумя тактовыми импульсами превысила 25 мс, то устройство должно прервать текущую передачу. Мастер должен сформировать состояние старта в процессе передачи или после ее окончания. Ведомый должен освободить шину. Устройства, обнаружившие данное состояние, должны восстановить свои соединения и ожидать формирования состояния старта в пределах 10 мс.

Для отслеживания периодов ожиданий на шине в модуле I2C имеется счетчик времени ожидания. Функциональная схема счетчика показана на рисунке 16.14.

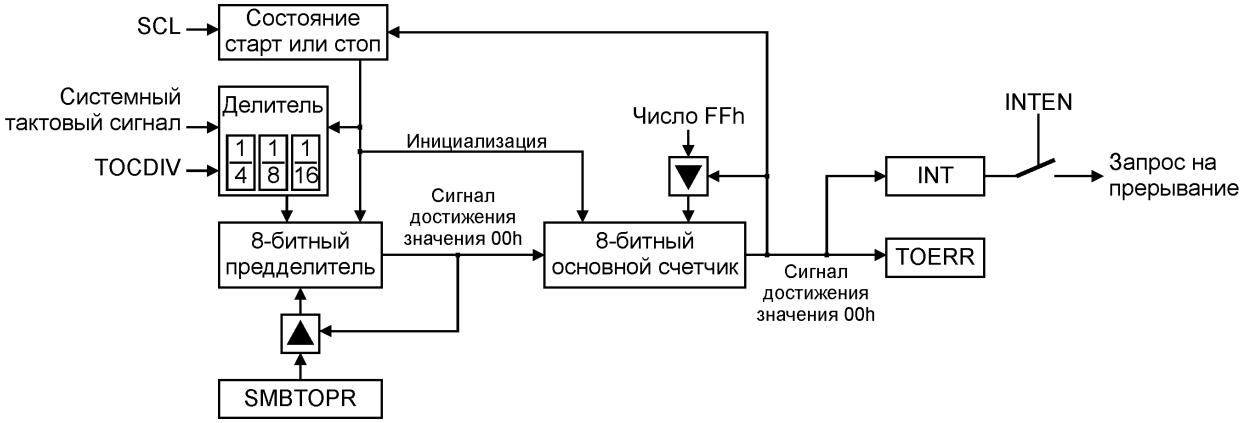


Рисунок 16.14 – Функциональная схема счетчика времени ожидания

Счетчик времени ожидания состоит из делителя, 8-битного программируемого предделителя и 8-битного основного счетчика. Все элементы счетчика времени ожидания начинают работу по отрицательному фронту сигнала на линии SCL (если работа счетчика разрешена). Положительный фронт сигнала на линии SCL сбрасывает значения делителя, предделителя и основного счетчика. Предделитель считает вниз, начиная со значения, записанного в регистр SMBTOPR. После достижения нуля счетчиком предделителя, он загружается значением из регистра SMBTOPR. Основной счетчик считает вниз от значения FFh. Каждое достижение нуля предделителем декрементирует значение основного счетчика. Обнуление основного счетчика и загрузка его значением FFh вызывает остановку основного счетчика, предделителя и делителя и установку флага TOERR в регистре SMBCST. Дополнительно устанавливается флаг INT и, если разрешено, генерируется прерывание.

Период времени ожидания определяется следующим выражением:

$$Тожид = Tosc \times TOCDIV \times (SMBTOPR + 1) \times 256, \quad (16.1)$$

где Tosc – период системного тактового сигнала с частотой fosc.

Арбитраж и обнаружение ошибок на шине

Арбитраж в режиме мастера передатчика может быть потерян в случае, когда два мастера одновременно формируют состояние старта и начинают передачу данных. Потеря арбитража может происходить как во время передачи адреса, так и во время передачи данных.

В случае потери приоритета при передаче байта адреса, мастер переходит в режим ведомого приемника и начинает принимать адрес. Если принятый адрес оказался «своим», модуль I2C далее функционирует в режиме ведомого. Если принятый адрес не оказался «своим», то модуль I2C переходит в режим безадресного ведомого.

В случае потери приоритета при передаче байта данных модуль I2C сразу переходит в режим безадресного ведомого.

Обнаружение и исправление ошибок на шине

Состояние ошибки на шине возникает в том случае, если во время передачи адреса/данных или во время квитирования на шине обнаруживаются состояния старта или стопа. При обнаружении ошибки на шине выполняются действия:

- в поле MODE регистра SMBST записывается код ошибки 1Fh;
- генерируется прерывание (если разрешено);
- модуль I2C переходит в режим безадресного ведомого;
- линии SDA и SCL освобождаются.

Обнаружение ошибки на шине может вызвать у простой шины некорректное формирование состояния старта и отключение модуля I2C. Поэтому для возврата к нормальной работе следует выполнить действия:

- выключить и снова включить модуль I2C (бит ENABLE в регистре SMBCTRL2);
- в течение времени простоя проверить, не подключен ли другой активный мастер к шине (бит BB регистра SMBCST должен быть обнулен);
- в режиме мастера шины сформировать состояние старта, передать адрес и затем сформировать состояние останова, таким образом, проведя синхронизацию всех ведомых устройств (в том числе и тех, которые не обнаружили ошибку на шине).

Режим IDLE

Переход в режим IDLE происходит при отключении внешнего сигнала тактирования модуля I2C записью нуля в бит I2CCLKEN регистра CLKREG. Переход в режим IDLE подобен программному выключению модуля I2C (очистка бита ENABLE в регистре SMBCTRL2). Регистры SMBCTRL1, SMBST и SMBCST очищаются, чтобы гарантировать нормальный старт после возобновления функционирования модуля.

Выход из режима IDLE осуществляется записью единицы в бит I2CCLKEN и включением модуля битом ENABLE.

16.3 Инициализация и функционирование

В целом, модуль I2C поддерживает два базовых режима – режим FS и режим HS.

Стандартный/скоростной режим или режим FS – стандартный режим работы, в котором модуль функционирует по умолчанию. Диапазон частот сигнала на линии SCL – от 65 кГц до 2,35 МГц (при XTAL1 = 33 МГц).

Высокоскоростной режим или режим HS – режим работы, который включается программно. Режим HS значительно превосходит режим FS по скорости – диапазон частот сигнала на линии SCL от 710 кГц до 7,3 МГц (при XTAL1 = 33 МГц).

Все операции режима HS начинаются в режиме FS в следующем порядке:

- стартовое состояние;
- 8-битный код мастера (значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств);
- неквитирование.

Арбитраж на шине происходит в момент передачи несколькими мастерами своих уникальных кодов. Выигравший арбитраж мастер захватывает шину. В связи с такой организацией режима HS, дальнейший арбитраж и синхронизация на шине не реализуются.

После выполнения вышеуказанных шагов устройства, поддерживающих режим HS, переключаются в этот режим. Мастер генерирует состояние повторного старта (SR), а затем передает адрес ведомого и бит направления передачи R/W# (см. рисунок 16.15. Расшифровка обозначений, принятых на рисунке, приведена в таблице 16.1).

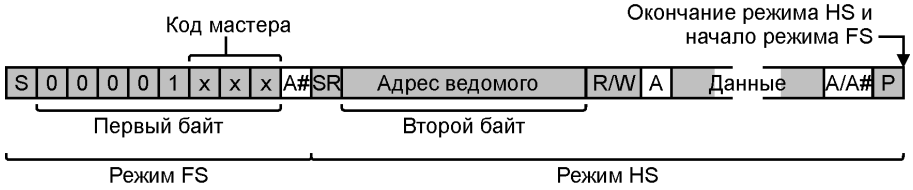


Рисунок 16.15 – Переход в режим HS и обратно в режим FS

Все передачи в режиме HS по формату идентичны передачам режима FS, что делает эти два режима полностью совместимыми.

Выход из режима HS происходит генерированием состояния окончания передачи (P), после которого все устройства переключаются обратно в режим FS.

В каждом из двух базовых режимов – FS и HS – модуль I2C может функционировать как мастер или ведомый, получать или передавать информацию.

Далее все режимы работы модуля I2C будут рассмотрены подробно.

Инициализация

Для начала работы следует произвести инициализацию:

1 Включить модуль I2C установкой бита ENABLE в регистре SMBCTRL2.

2 Если активен режим мастера, записать нужный коэффициент деления в битовое поле SCLFRQ в регистре SMBCTRL2 для выбора периода тактового сигнала SCL (для режима HS записать коэффициент деления в поле HSDIV регистра SMBCTRL3).

3 Если активен режим ведомого, необходимо:

- записать «собственный» адрес ведомого в битовое поле ADDR и установить бит SAEN регистра SMBADDR;
- для реализации 10-битной адресации записать старшие биты адреса в битовое поле S10AD и установить бит S10EN регистра SMBCTRL3;
- для включения функции распознавания адреса общего вызова установить бит GCMEN в регистре SMBCTRL1;
- для включения функции распознавания адреса отклика установить бит SMBARE в регистре SMBCTRL1.

4 При необходимости отслеживания периодов ожидания на шине записать желаемые значения в регистр SMBTOPR и в битовое поле TOCDIV (регистр SMBCST) для отсчета времени ожидания на линии SCL. Для автоматического отслеживания времени ожидания записать ненулевое значение в битовое поле TOCDIV регистра SMBCST.

5 Для разрешения формирования запроса на прерывание установить бит INTEN в регистре SMBCTRL1.

Функционирование

Модуль I2C может работать в режиме мастера или ведомого. Также он может функционировать как передатчик или приемник. Итого, модуль I2C поддерживает девять режимов:

- безадресный ведомый;
- мастер передатчик в режиме FS;
- мастер передатчик в режиме HS;
- мастер приемник в режиме FS;
- мастер приемник в режиме HS;
- ведомый передатчик в режиме FS;
- ведомый передатчик в режиме HS;
- ведомый приемник в режиме FS;
- ведомый приемник в режиме HS.

Передача информации по шине состоит из последовательности различных действий (начало передачи, прием данных и др.). Каждое действие называется состоянием (состояние старта, состояние останова и др.). После того, как то или иное состояние сформировано, его код аппаратно записывается в регистр SMBST в битовое поле MODE и может быть прочитано программно. В таблицах 16.2 и 16.3 приводятся все возможные состояния, их мнемонические обозначения и коды. На квитирование или неквитирование приема указывает запись «ACK» или «NACK» соответственно. Так, например, если мастер отправил байт адреса ведомому, который после получения квитировал прием, то на это будет указывать «ACK», а в поле MODE регистра SMBST будет записан код 04h, соответствующий состоянию с мнемоническим обозначением «MTADPA». Более подробно каждый режим работы модуля I2C будет рассмотрен далее. На рисунках 16.16 –

16.23, поясняющих работу модуля в том или ином режиме, приняты обозначения, расшифровка которых приводится в таблице 16.1. Для получения дополнительной информации и понимания работы модуля I2C можно воспользоваться приложением Е.

Таблица 16.2 – Коды функционирования модуля I2C в режиме FS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра SMBST	ACK/ NACK
Общий		00h	IDLE	IDLE, нет доступной валидной информации о статусе	–
Мастер в режиме FS	–	01h	STDONE	Сформировано состояние старта	–
		02h	RSDONE	Сформировано состояние повторного старта	–
		03h	IDLARL	Потеря арбитража, переход в режим безадресного ведомого	–
	Передача	04h	MTADPA	Отправлен адрес ведомого	ACK
		05h	MTADNA	Отправлен адрес ведомого	NACK
		06h	MTDAPA	Отправлен байт данных	ACK
		07h	MTDANA	Отправлен байт данных	NACK
	Прием	08h	MRADPA	Отправлен адрес ведомого	ACK
		09h	MRADNA	Отправлен адрес ведомого	NACK
		0Ah	MRDAPA	Принят байт данных	ACK
		0Bh	MRDANA	Принят байт данных	NACK
	–	0Ch	MTMCER	Отправлен код мастера, обнаружена ошибка	ACK
Ведомый в режиме FS	Прием	10h	SRADPA	Принят адрес	ACK
		11h	SRAAPA	Принят адрес после потери арбитража	ACK
		12h	SRDAPA	Принят байт данных	ACK
		13h	SRDANA	Принят байт данных	NACK
	Передача	14h	STADPA	Принят адрес	ACK
		15h	STAAPA	Принят адрес после потери арбитража	ACK
		16h	STDAPA	Отправлен байт данных	ACK
		17h	STDANA	Отправлен байт данных	NACK
	Передача адреса отклика	18h	SATADP	Принят адрес отклика на предупреждение	ACK
		19h	SATAAP	Принят адрес отклика на предупреждение после потери арбитража	ACK
		1Ah	SATDAP	Отправлены данные в ответ на получение адреса отклика	ACK
		1Bh	SATDAN	Отправлены данные в ответ на получение адреса отклика	NACK
	–	1Ch	SSTOP	Обнаружено состояние останова ведомого	–
		1Dh	SGADPA	Принят адрес общего вызова	ACK
		1Eh	SDAAPA	Принят адрес общего вызова после потери арбитража	ACK
Общий		1Fh	BERROR	Обнаружена ошибка на шине (некорректное состояние старта или останова)	–

Примечание – Диапазон значений кодов 0Dh–0Fh зарезервирован и не доступен для использования. Дополнительная информация находится в приложении Е.

Таблица 16.3 – Коды функционирования модуля I2C в режиме HS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра SMBST	ACK/ NACK
Мастер в режиме HS	–	21h	HMTMCOK	Код мастера передан успешно, переход в режим HS	–
		22h	HRSDONE	Сформировано состояние повторного старта	–
		23h	HIDLARL	Потеря арбитража, переход в режим HS безадресного ведомого	–
	Передача	24h	HMTADPA	Отправлен адрес ведомого	ACK
		25h	HMTADNA	Отправлен адрес ведомого	NACK
		26h	HMTDAPA	Отправлен байт данных	ACK
		27h	HMTDANA	Отправлен байт данных	NACK
	Прием	28h	HMRADPA	Отправлен адрес ведомого	ACK
		29h	HMRADNA	Отправлен адрес ведомого	NACK
		2Ah	HMRDAPA	Принят байт данных	ACK
		2Bh	HMRDANA	Принят байт данных	NACK
Ведомый в режиме HS	Прием	30h	HSRADPA	Принят адрес	ACK
		32h	HSRDAPA	Принят байт данных	ACK
		33h	HSRDANA	Принят байт данных	NACK
	Передача	34h	HSTADPA	Принят адрес	ACK
		36h	HSTDAPA	Отправлен байт данных	ACK
		37h	HSTDANA	Отправлен байт данных	NACK
Примечание – Диапазоны значений кодов 2Ch–2Fh и 38h–3Fh, а также коды 20h, 31h и 35h зарезервированы и не доступны для использования. Дополнительная информация находится в приложении Е.					

Режим безадресного ведомого

Режим работы по умолчанию (MODE = 00h). После включения модуль I2C начинает функционировать в режиме безадресного ведомого и непрерывно мониторит шину. При обнаружении состояния старта или повторного старта переходит в режим ведомого приемника. Для перехода в режим мастера передатчика нужно сформировать корректное состояние старта.

Переключение в режим безадресного ведомого происходит в случаях:

- стартовое состояние не было успешно сформировано, так как другое устройство удерживало на линии SCL низкий уровень сигнала;
- произошла потеря арбитража во время передачи байта данных в режиме мастера передатчика или во время передачи бита R/W# в режиме мастера приемника;
- произошла потеря арбитража во время ответа на полученный адрес отклика;
- неквитирование принятого адреса в режиме ведомого приемника (адрес не совпал со «своим» или запрещен);
- неквитирование в конце переданного байта в режиме ведомого передатчика;
- обнаружено состояние останова;
- обнаружена ошибка на шине;
- модуль I2C был сброшен;
- модуль I2C был выключен.

Режим FS мастера передатчика

Включение режима:

1 Переход в режим мастера передатчика происходит после успешного формирования состояния старта. Первый байт, передаваемый мастером сразу после старта, состоит из адреса ведомого и бита направления.

2 В зависимости от состояния бита направления (R/W#), модуль I2C далее функционирует как мастер передатчик (если R/W# = «0») или как мастер приемник (если R/W# = «1»). Для перехода в режим HS мастер может передать код мастера (0000_1xxxb) вместо первого байта адреса.

3 Переход в режим мастера произойдет после установки бита START в регистре SMBCTRL1. Если бит BB в регистре SMBCST сброшен, т.е. шина свободна, будет сгенерировано состояние старта. Если бит BB = 1b, то бит START останется установленным, а состояние старта будет сгенерировано по истечении времени, равного одному такту сигнала тактирования на линии SCL, после освобождения шины.

4 Как только стартовое состояние будет сгенерировано успешно, бит START сбросится, модуль I2C перейдет в состояние STDONE (в поле MODE запишется значение 01h), установится флаг INT, и линия SCL будет удерживаться в «0» до тех пор, пока флаг INT не будет сброшен. Если разрешено битом INTEN (регистр SMBCTRL1), сгенерируется прерывание.

Передача адреса и данных:

1 Пока удерживается флаг INT, программа записывает адрес ведомого и бит направления передачи в регистр данных SMBSDA (адрес записывается в биты с седьмого по первый).

2 После записи в регистр SMBSDA флаг INT сбрасывается программно установкой бита CLRST в регистре SMBCTRL1.

3 После сброса флага INT и по истечении времени, требуемого для установки данных, на линии SCL появляется тактовый сигнал и данные, хранящиеся в регистре SMBSDA, начинают передаваться по линии SDA.

4 После завершения передачи байта и получения ответа на запрос подтверждения передачи (ACK), т.е. после девятого такта сигнала тактирования на линии SCL, аппаратная часть анализирует квитирование/неквитирование передачи и устанавливает соответствующий код в поле MODE.

5 Во время передачи линии SCL и SDA постоянно мониторятся с целью выявления возможных конфликтов с другими устройствами, подключенными к шине. В случае обнаружения конфликта передача прерывается, и в поле MODE записывается код 11b (состояние SRAAPA – переход в режим ведомого приемника после потери арбитража) или код 03h (состояние IDLARL – переход в режим безадресного ведомого после потери арбитража).

6 Если бит направления равен единице и не обнаружено ошибок на шине, модуль I2C переходит в режим мастера приемника.

7 Если бит направления равен нулю и передача адреса ведомого завершена успешно (значение кода в поле MODE не равно 05h/1Fh), устанавливается флаг INT, указывая на то, что ожидается запись первого байта данных в регистр SMBSDA для дальнейшей передачи, и, если разрешено, генерируется прерывание. Пока флаг INT будет оставаться установленным, линия SCL будет удерживаться в «0».

8 Байт данных записывается программно в регистр SMBSDA, и передача продолжается.

9 Если ведомый приемник не квитирует отправленный ему байт данных, в поле MODE записывается код 0Bh (состояние MRDANA). На линии SCL будет установлен низкий уровень сигнала и, если разрешено, сгенерировано прерывание.

Для отслеживания ошибок в пакетах данных применяется механизм вычисления контрольной суммы (CRC) для нескольких байт данных. В режиме мастера передатчика установка бита PECNEXT в регистре SMBCST вызовет перенос содержимого регистра ошибок (не доступен программно) в регистр SMBSDA и инициирует передачу байта CRC (байт контрольной суммы) ведомому. Передача байта CRC должна выполняться после передачи последнего байта данных и перед формированием состояния останова или повторного старта.

Мастер передатчик контролирует шину и может адресовать любое ведомое устройство и изменять направление передачи без потери контроля над шиной, используя возможность формирования состояния повторного старта. Для формирования состояния следует:

1 Установить бит START.

2 В режиме мастера приемника прочитать последний полученный байт из регистра SMBSDA.

3 Сбросить флаг прерывания INT.

После этих действий будет освобождена линия SCL, сгенерировано состояние повторного старта и сгенерировано прерывание. В поле MODE будет записан код 02h (состояние RSDONE).

Модуль I2C может быть выведен из режима мастера передатчика генерированием состояния останова. Для этого необходимо:

1 Установить бит STOP в регистре SMBCTRL1.

2 В режиме мастера приемника прочитать последний полученный байт из регистра SMBSDA.

3 Сбросить флаг INT.

Вышеуказанные действия приведут к незамедлительному формированию состояния останова и очистке бита STOP.

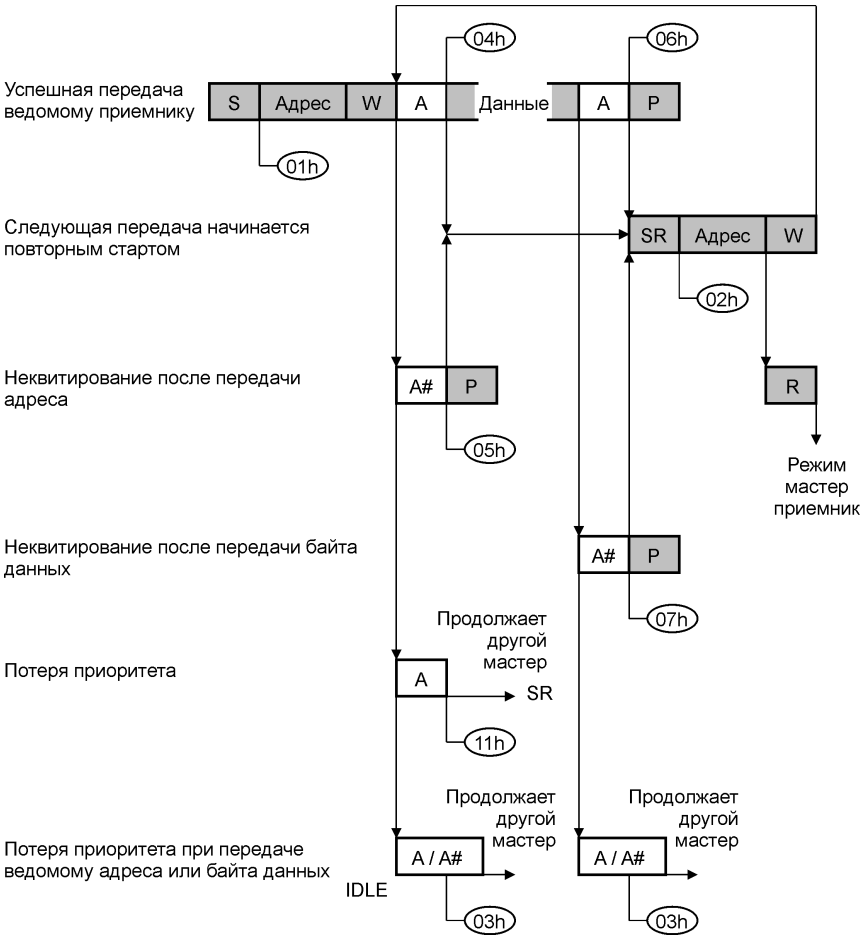


Рисунок 16.16 – Режим FS мастера передатчика

Состояние останова может быть сформировано только, если модуль I2C функционирует как мастер и контролирует шину (в поле MODE находится любое значение кода из диапазона 01h – 0Bh).

Дополнительно можно обратиться к приложению Е.

На рисунке 16.16 представлено графическое пояснение к описанию режима.

Режим HS мастера передатчика

Переход в режим HS мастера передатчика происходит в том случае, если после состояния старта мастер передает код мастера (0000_1xxxb) вместо адреса ведомого. По окончании передачи кода мастера устанавливается флаг INT и, если разрешено, генерируется прерывание. Вслед за успешной передачей кода мастера в поле MODE записывается код 21h (состояние HMTMCOK), и мастер переходит в режим HS.

Далее необходимо сформировать состояние повторного старта, записав единицу в бит START и сбросить флаг INT записью единицей в бит CLRST.

После сгенерированного состояния повторного старта устанавливается флаг INT, и в поле MODE записывается код 22h (состояние HRSDONE). Дальнейший порядок действий по передачи адреса и данных аналогичен описанному режиму FS мастера передатчика.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.17 представлено графическое пояснение к описанию режима.

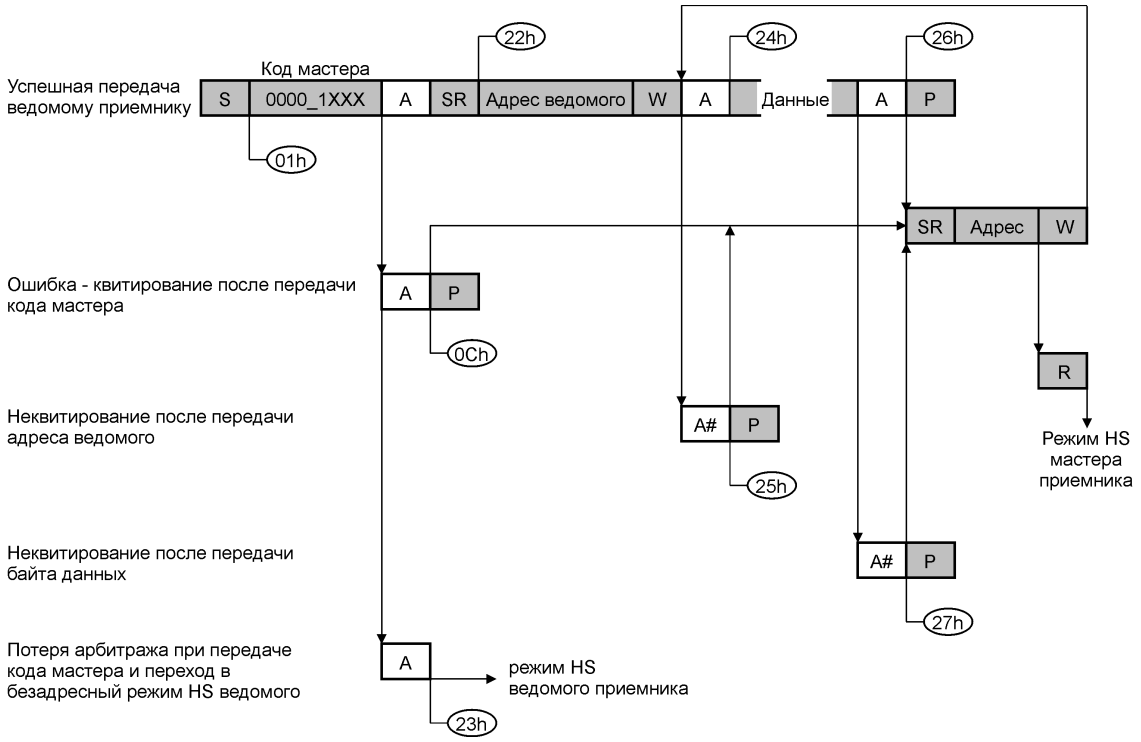


Рисунок 16.17 – Режим HS мастера передатчика

Режим FS мастера приемника

Переход в режим мастера приемника происходит после успешной передачи адреса ведомого с единичным битом направления (R/W# = «1»). В режиме мастера приемника модуль I2C получает данные от ведомого устройства, поэтому теряет контроль над шиной SDA. В тоже время мастер продолжает тактировать передачу и должен отвечать на бит ACK каждого принятого байта.

После каждого принятого байта устанавливается флаг INT, и пользовательская программа читает полученные данные из регистра SMBSDA. Линия SCL удерживается в «0», пока установлен флаг INT. После сброса флага INT может стартовать прием следующего байта. После этого (согласно протоколу SMBus) состояния повторного старта или стопа не должны генерироваться мастером, поскольку мастер теперь не является единственным контролером линии SDA. В конце приема каждого байта мастер не квитирует прием, сообщая таким образом ведомому об успешном приеме.

После приема предпоследнего байта перед сбросом флага INT следует записать ноль в бит ACK регистра SMBCTRL1. В тоже время, если требуется отправка байта CRC,

следует установить бит PECNEXT в регистре SMBCST. После сброса флага INT будет принят последний байт данных и не квитирован. По окончании приема мастер возвращается в режим передатчика и теперь может сгенерировать состояние повторного старта или останова.

Если механизм отслеживания ошибок включен, то последний переданный от ведомого байт будет байтом CRC. В случае, если результат вычисления контрольной суммы не нулевой, то установится флаг ошибки PECFAULT в регистре SMBCST.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.18 представлено графическое пояснение к описанию режима.

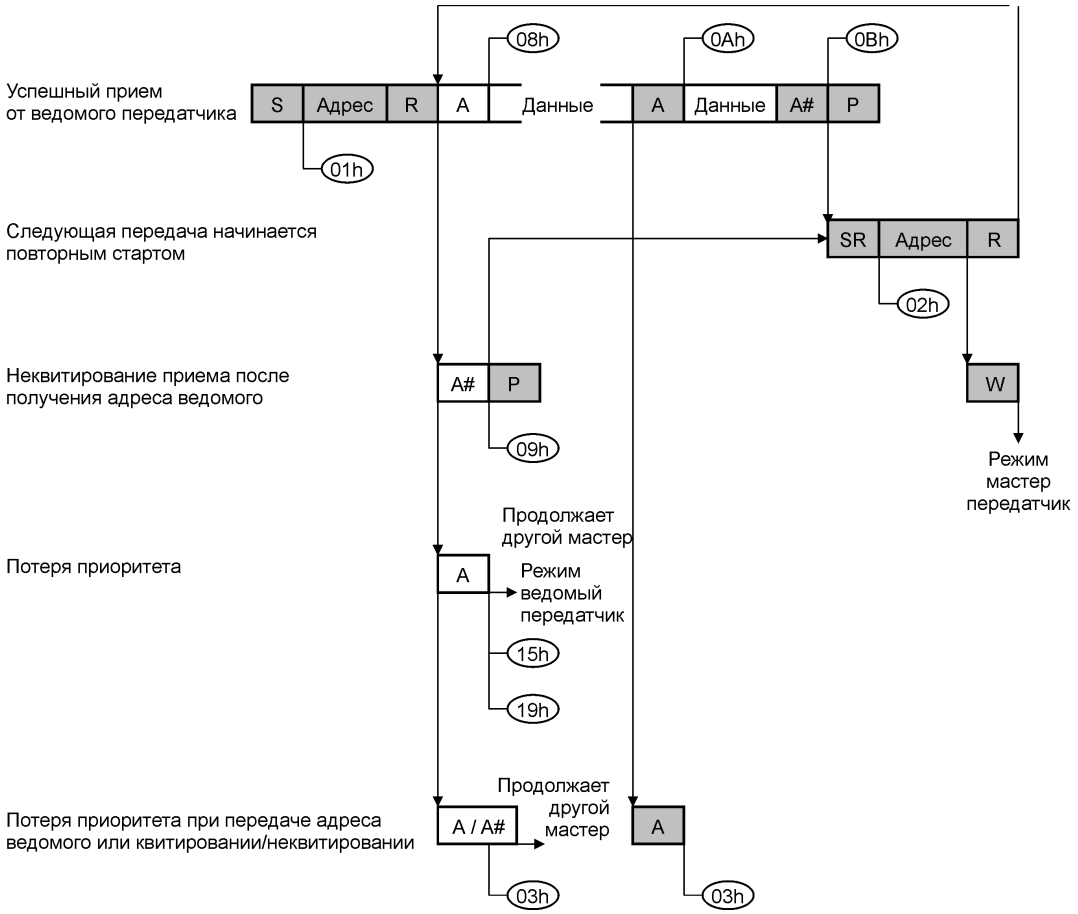


Рисунок 16.18 – Режим FS мастера приемника

Режим HS мастера приемника

Переход в режим HS мастера приемника происходит, если после переданного кода мастера и последовавшего за ним состоянием повторного старта производится передача адреса ведомого с битом направления R/W# = «1». Модуль I2C переходит в режим HS мастера приемника, устанавливается флаг INT, а в поле MODE записывается соответствующий код из диапазона 28h – 2Bh.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.19 представлено графическое пояснение к описанию режима.

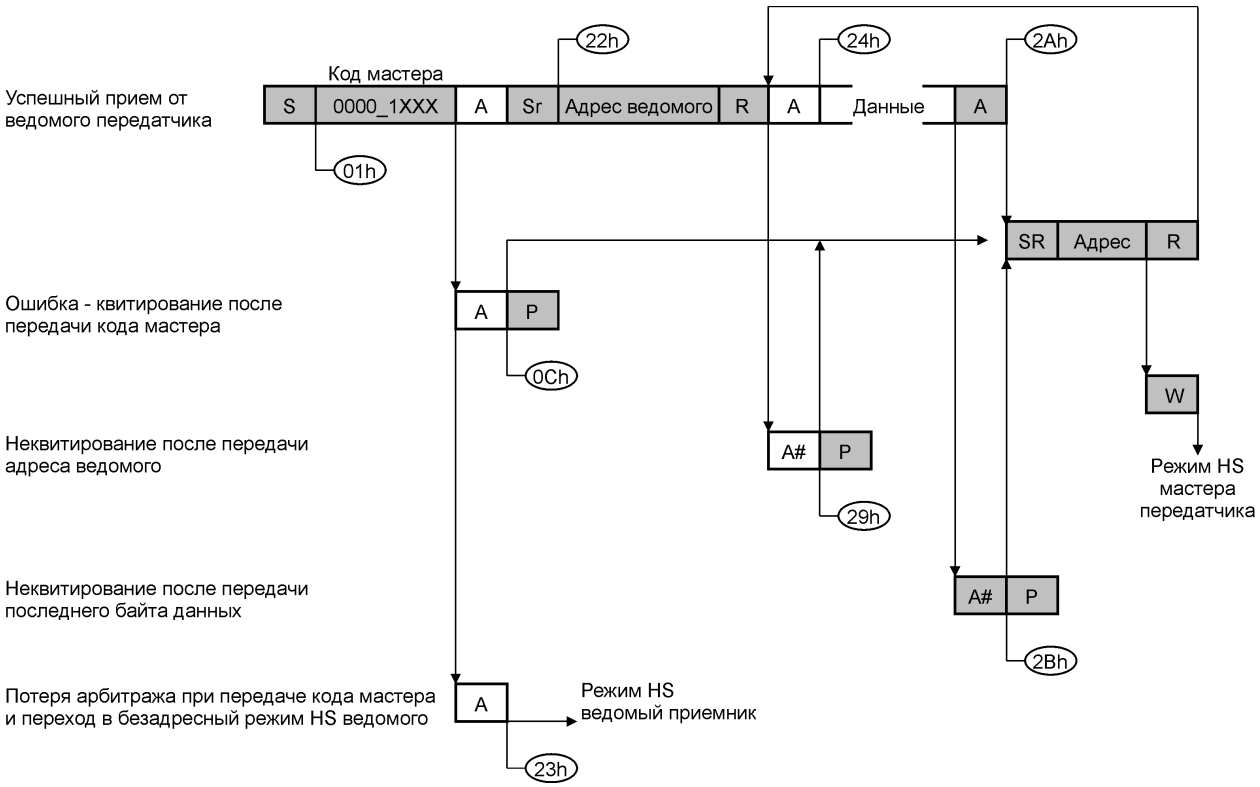


Рисунок 16.19 – Режим HS мастера приемника

Режим FS ведомого приемника

В этом режиме данные принимаются от мастера передатчика. Ведомый квитирует или не квитирует прием каждого байта.

После включения модуль I2C мониторит шину. При обнаружении состояния старта, модуль I2C переключается в режим ведомого приемника и начинает принимать семь бит адреса и бит направления передачи от мастера. Мастер-передатчик может переключиться в режим ведомого приемника вследствие потери арбитража при передаче адреса.

После получения байта адреса ведомый сравнивает полученный адрес с:

- полем ADDR регистра SMBADDR, если установлен бит SAEN;
- со значением 0000_000b (адрес общего вызова), если установлен бит GCMEN;
- со значением 0001_100b (адрес отклика), если установлен бит SMBARE.

Квитирование приема производится, если принятый адрес совпал с «собственным» (запрограммированным пользователем) адресом общего вызова или адресом отклика. После обнаружения совпадения адреса и квитирования в поле MODE записывается соответствующий код, и устанавливается флаг INT. Также, если разрешено битом INTEN, генерируется прерывание. Принятый байт (адрес и бит направления) переписывается в регистр SMBSDA.

В зависимости от состояния бита направления, модуль I2C переходит в режим ведомого передатчика (если R/W# = «1») или остается в режиме ведомого приемника (R/W# = «0»).

После каждого принятого байта устанавливается флаг INT, указывающий на то, что необходимо прочитать данные из регистра SMBSDA, а линия SCL удерживается в «0». После программного чтения регистра SMBSDA флаг INT сбрасывается (записью единицы в бит CLRST), и линия SCL освобождается.

Установка битов SAEN и S10EN включает режим 10-битной адресации ведомого приемника. После обнаружения состояния старта ведомый последовательно принимает два байта, в которых содержится адрес.

После корректного приема ведомым двух байтов и совпадении принятого адреса с собственным, байты сохраняются в регистре SMBSDA и сдвиговом регистре, прием квитируется, устанавливается флаг INT, а в поле MODE записывается соответствующий код состояния – 10h или 11h.

Если включен механизм обнаружения ошибок, последний байт, принятый от мастера передатчика, будет байтом CRC. Если результат вычисления контрольной суммы не нулевой, устанавливается флаг ошибки PECFAULT и передача не квитируется. Программа пользователя должна «знать» о количестве передаваемых мастером байт и устанавливать бит PECNEXT перед чтением предпоследнего байта из регистра SMBSDA и уже потом сбрасывать флаг INT. В результате будет аппаратно рассчитана контрольная сумма, и результат отправлен мастеру в момент передачи бита ACK. Если ошибок нет, будет выполнено квитирование (отправлен «0» в ответ на запрос ACK), если ошибки есть – неквитирование (отправлен «1» в ответ на запрос ACK).

Если ведомому приемнику нужно сообщить мастеру, что он не может более принимать данные, следует сначала установить бит ACK, а затем – бит CLRST (для сброса флага INT). Далее будет принят последний байт данных, который не будет квитирован (бит ACK = 1b), и установится флаг INT. После этого программа может прочитать последний полученный байт из регистра SMBSDA и сбросить флаг INT, после чего модуль I2C освободит шину.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.20 представлено графическое пояснение к описанию режима.

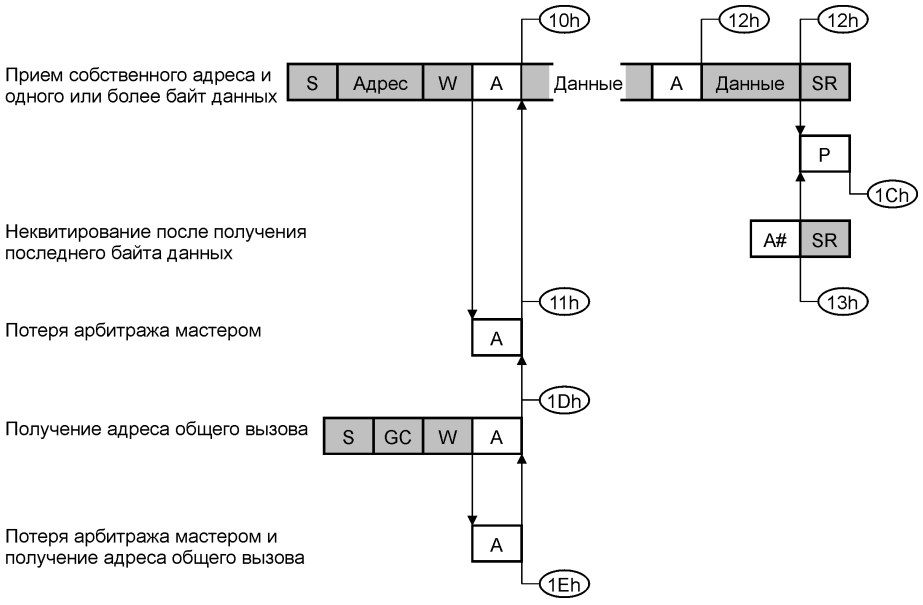


Рисунок 16.20 – Режим FS ведомого приемника

Режим HS ведомого приемника

Включение режима происходит после получения валидного кода мастера (0000_1xxxb). После передачи кода мастера формируется состояние повторного старта, а затем передается адрес ведомого с нулевым битом направления (R/W# = «0»). После получения байта адреса ведомый проверяет его на совпадение (см. ранее «Режим FS ведомого приемника»).

Дополнительно можно обратиться к приложению Е.

На рисунке 16.21 представлено графическое пояснение к описанию режима.

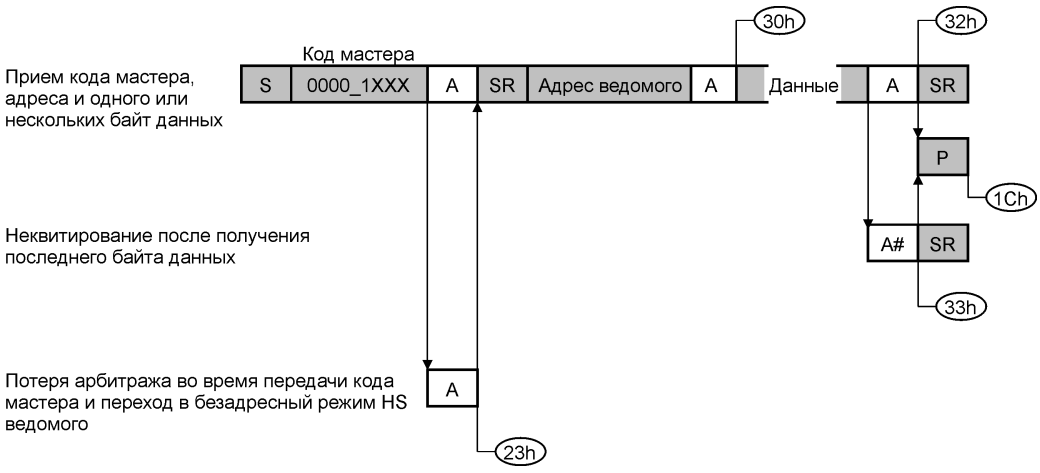


Рисунок 16.21 – Режим HS ведомого приемника

Режим FS ведомого передатчика

В этом режиме данные передаются от ведомого передатчика к мастеру приемнику. Ведомый проверяет ответ мастера на бит ACK.

Переход в режим передатчика происходит из режима ведомого приемника. После получения собственного адреса и бита направления, равного единице (R/W# = «1»), ведомый становится передатчиком. Флаг INT устанавливается, указывая на то, что в регистр SMBSDA следует записать данные. Пока установлен флаг INT, линия SCL удерживается в «0». После записи данных в регистр SMBSDA следует сбросить флаг INT. После этого, по истечении времени, необходимого для установки данных на линии SDA, линия SCL освобождается, и данные начинают передаваться.

Передача данных аналогична передаче в режиме мастера передатчика. После каждого успешного приема байта устанавливается флаг INT, а в поле MODE записывается соответствующий код. Линия SCL удерживается в состоянии «0» до тех пор, пока флаг INT остается установленным. Флаг INT должен сбрасываться только после записи данных в регистр SMBSDA. Каждый последующий байт должен записываться в регистр SMBSDA до тех пор, пока в поле MODE не появится код 17h (состояние STDANA), указывающий на то, что мастер «не желает» далее принимать данные.

Вывод ведомого из режима передатчика осуществляется только мастером приемником. Мастер приемник должен не квитировать последний (согласно запланированному количеству) полученный байт данных. При обнаружении неквитирования переданных данных, модуль I2C переходит в режим безадресного ведомого и в поле MODE записывается код 00h (состояние IDLE). Далее ведомый мониторит шину в ожидании состояния старта или повторного старта.

Для работы в режиме с 10-битной адресацией следует осуществить действия, аналогичные описанным для режима FS ведомого приемника.

Сначала модуль I2C переходит в режим ведомого приемника и получает 10-битный адрес. Если программно не требуется никаких действий, то флаг INT не устанавливается, линия SCL не удерживается в «0» и поле MODE содержит соответствующую информацию о состоянии. Далее (см. ранее «Формат передачи данных с 10-битной адресацией»), вслед за вторым байтом адреса может последовать состояние повторного старта и затем повторная передача первого байта адреса с той лишь разницей, что бит направления содержит единицу (R/W# = «1»). Таким образом, после приема трех байт, если принятый 10-битный адрес окажется «своим», установится флаг INT и ведомый переключится в режим передатчика. В поле MODE запишется один из двух кодов – 14h или 15h.

Если включен механизм распознавания ошибок, то последний отправленный ведомым передатчиком байт будет байтом CRC. Программа должна «знать» количество байт, посылаемых в пакете данных, и после отправки всех байт устанавливать бит

PECNEXT (вместо записи очередных данных в регистр SMBSDA) для того, чтобы в регистр SMBSDA записался байт контрольной суммы.

В модуле I2C поддерживается функция распознавания адреса отклика, который передается мастером шины ко всем ведомым. Ведомое устройство, получившее адрес отклика (0001_100b), переключается в режим передатчика и начинает передавать свой собственный адрес.

Для включения функции распознавания адреса отклика следует установить бит SMBARE в регистре SMBCTRL1.

Модуль I2C реагирует на адрес отклика только при работе в режиме ведомого. В ответ на получение адреса отклика начать передачу адресов могут несколько ведомых. Ведомый, выигравший арбитраж, продолжает передачу, остальные – освобождают шину.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.22 представлено графическое пояснение к описанию режима.

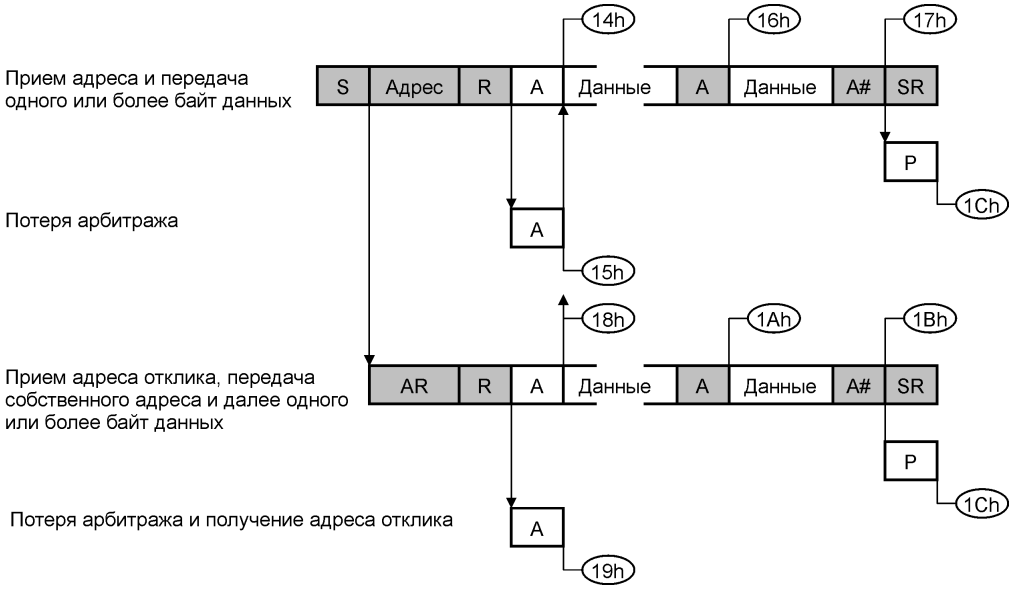


Рисунок 16.22 – Режим FS ведомого передатчика

Режим HS ведомого передатчика

Модуль I2C переходит в режим HS ведомого после получения валидного кода мастера (0000_1xxxb). Далее следует состояние повторного старта и передача адреса ведомого с единичным битом направления (R/W# = «1»). После этого ведомый переключается в режим HS ведомого передатчика. Функционирование в этом режиме в целом идентично режиму FS ведомого передатчика, с теми отличиями, что поддерживается более высокая скорость передачи, а значения кодов состояний (поле MODE) находятся в диапазоне 34h – 37h.

Дополнительно можно обратиться к приложению Е.

На рисунке 16.23 представлено графическое пояснение к описанию режима.

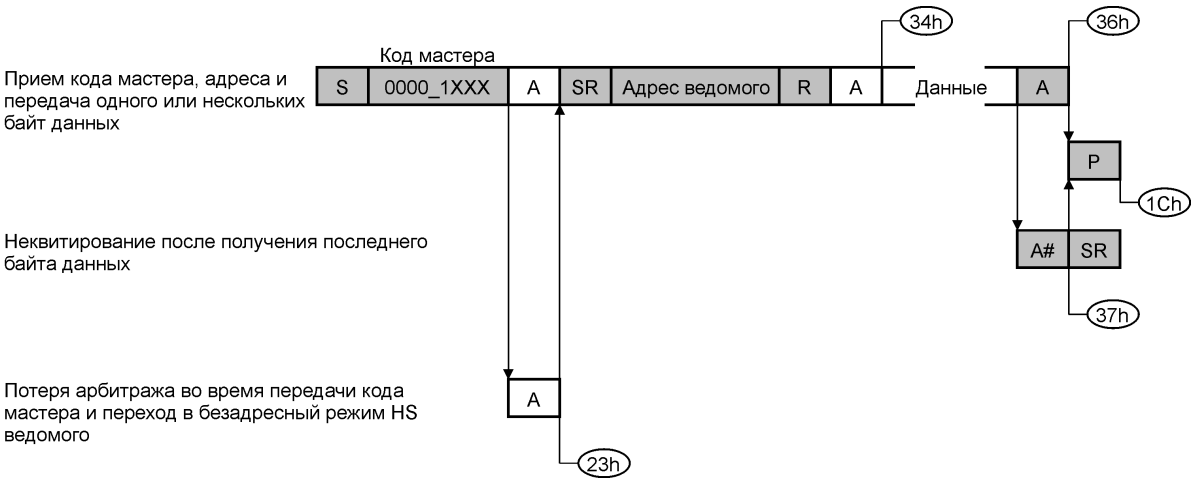


Рисунок 16.23 – Режим HS ведомого передатчика

Дополнительная информация о работе модуля

1 Когда модуль I2C выключен, бит BB регистра SMBCST очищен. Включения модуля в системе с более, чем одним мастером, может произойти в момент времени, когда по шине идет передача. Бит BB не сможет это показать. Во избежание создания ошибок на шине, модуль I2C должен синхронизироваться с сигналами на шине прежде, чем сделать попытку стать мастером. Для этого следует дождаться момента, когда на шине не будет выявлена активность, т.е. периодически проверять бит BB через периоды времени, равные периоду ожидания на шине.

2 Бит BB позволяет мониторить шину и не допускать формирования ошибочных состояний старта в процессе передачи между другими устройствами на шине.

3 В некоторых случаях шина может «зависать» при активных (с нулевым уровнем) сигналах на линиях SDA и/или SCL. Источниками таких состояний могут быть необнаруженные ошибочные стартовые или стоповые состояния, сформировавшиеся в течение приема ведомых данных. Если считать, что причиной зависания явился модуль I2C, то возможны следующие два варианта развития событий:

а) если зависла линия SCL, ничего не будет происходить, а мастер, захвативший шину, должен освободить ее;

б) если зависла линия SDA, мастер должен освободить шину. Следует помнить, что в нормальном состоянии удерживать линию SCL может только текущий мастер шины. Последовательность действий для выхода из зависания следующая (при условии, что на шине только один мастер):

- выключить и включить модуль I2C для перевода его в режим безадресного ведомого;

- установить бит START для создания состояния старта;

- проверить, удерживается ли линия SDA в «0» (активное состояние) чтением бита TSDA регистра SMBCST. Если линия активна, отправить одиночный импульс по линии SCL, установив бит TGSCL в регистре SMBCST;

- проверить, что в поле MODE записан код 01b (состояние STDONE), который укажет на то, что состояние старта сформировано. Если нет, то повторять предыдущий и этот шаги до тех пор, пока линия SDA не освободится.

17 Контроллер интерфейса CAN

Последовательный интерфейс CAN (Controller Area Network) – интерфейс связи, эффективно поддерживающий распределенное управление в реальном масштабе времени с высокой помехозащищенностью. Протокол связи определен в спецификации CAN 2.0B.

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. Неадресная 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), хорошая 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 (см. рисунок 17.1).

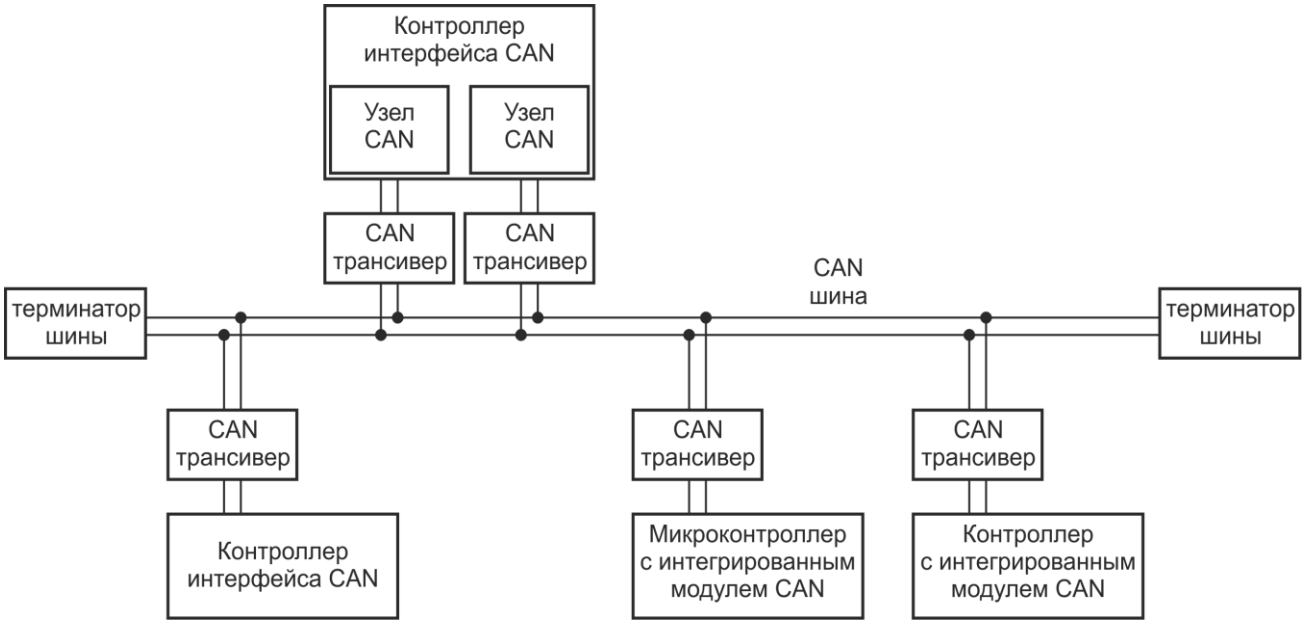


Рисунок 17.1 – Общая структура CAN сети

Логика шины работает по механизму монтажного И, в котором рецессивный бит соответствует логической единице, а доминантный – логическому нулю. 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 линии связи 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 1000 м скорость шины должна быть не более 40 Кбит/с). Из-за дифференциального характера линии связи шина CAN малочувствительна к электромагнитным помехам. Экранирование шины значительно снизит воздействие внешнего электромагнитного поля, что особенно важно для высокоскоростных режимов работы.

Двоичная информация кодируется. Доминантным является низкий уровень, рецессивным – высокий. Для гарантированной синхронизации данных всеми узлами шины используется принцип «бит-стаффинга». Это означает, что 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.) и степень приоритета 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 передаче сообщения узел становится управляющим шины, все остальные узлы переходят в режим приема. После приема сообщения (подтвержденного каждым узлом) каждый узел проверяет идентификатор в сообщении и сохраняет сообщение, если это требуется. В противном случае, сообщение сбрасывается. 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», значит, 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 сообщения одновременно разными узлами. Для этого программно должно быть обеспечено, чтобы узлы, передающие данные, не имели одинаковых идентификаторов. Оригинальная спецификация в версии CAN 2.0b (так называемая расширенная версия CAN) определяет возможность идентификатора иметь длину 11 или 29 бит.

Протокол CAN предусматривает следующие типы сообщений:

- сообщение данных (стандартное и расширенное);
- удаленный запрос данных;
- сообщение об ошибке;
- сообщение о перезагрузке.

17.1 Типы и структура сообщений CAN

Стандартное сообщение данных

Формируется, когда узел желает передать данные. Формат сообщения показан на рисунке 17.2.

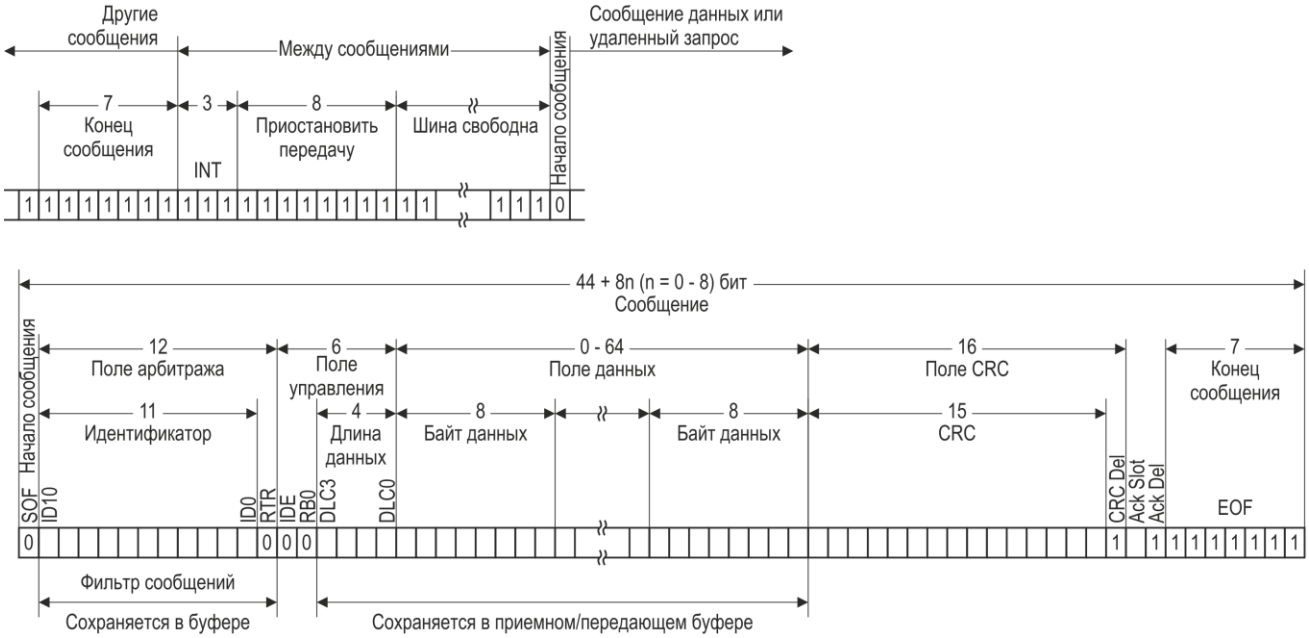


Рисунок 17.2 – Стандартное сообщение данных

Стандартное сообщение 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) и бит RTR передачи по удаленному запросу (RTR = «0» соответствует сообщению данных, RTR = «1» соответствует удаленному запросу);

- поле управления (6 бит), включающее бит IDE указатель расширенного идентификатора (IDE = «0» соответствует стандартному идентификатору, IDE = «1» соответствует расширенному идентификатору), бит RB0 резервный доминантный бит и 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 и 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) и 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 бит (поле INT простоя). Если после появления трех рецессивных битов (поле INT) ни один узел не начал передачу, шина переходит в состояние бездействия IDLE и находится в рецессивном состоянии до появления доминантного бита сообщения.

Расширенное сообщение данных

Формируется, когда узел желает передать данные. Формат сообщения показан на рисунке 17.3.

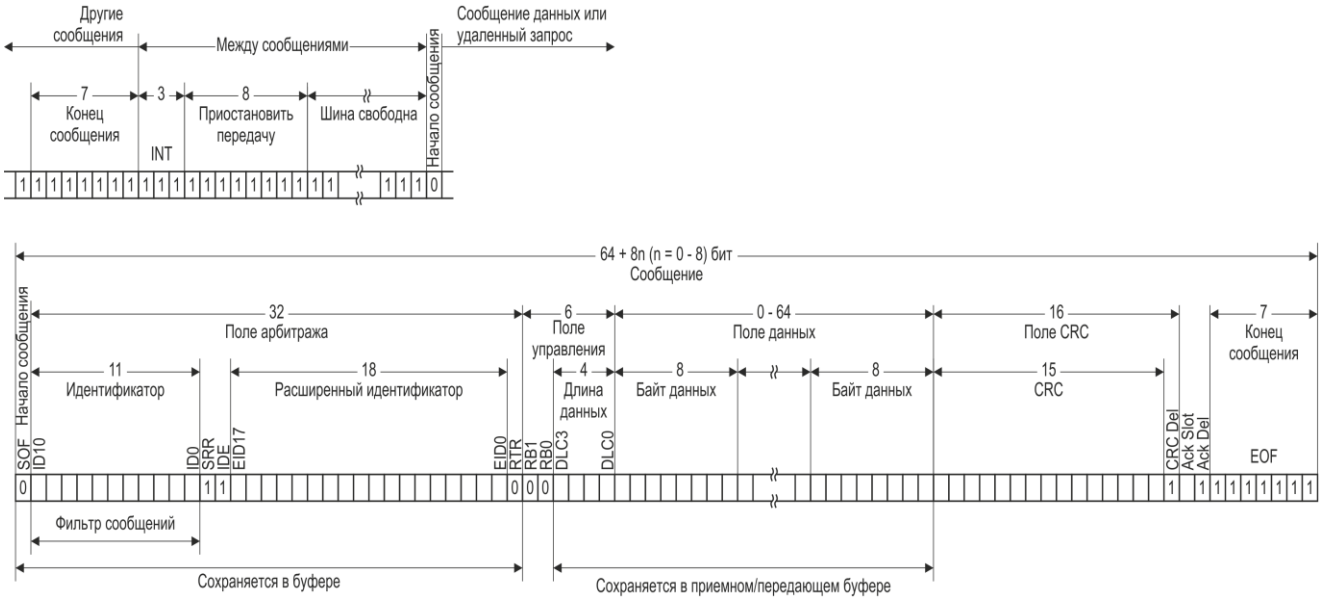


Рисунок 17.3 – Расширенное сообщение данных

Расширенное сообщение 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) и поле расширенного идентификатора (18 бит);
- бит RTR передачи по удаленному запросу (RTR = «0» соответствует сообщению данных, RTR = «1» соответствует удаленному запросу);
- поле управления (6 бит), включающее бит RB0 резервный доминантный бит, бит RB1 резервный доминантный бит и 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 и 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) и бит ACK Del рецессивный разделитель подтверждения;
- поле EOF конца сообщения (7 бит).

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 рисунке17.4 представлен вариант удаленного запроса со стандартным идентификатором).

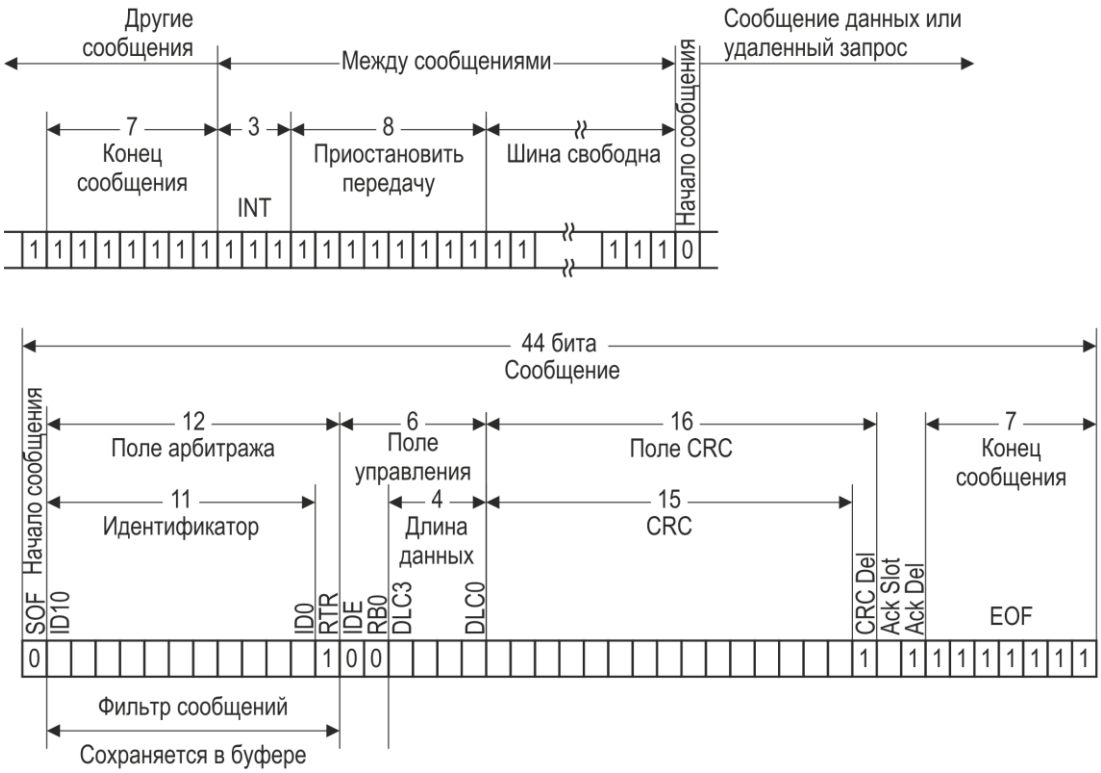


Рисунок 17.4 –Удаленный запрос данных (стандартный формат)

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).

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. Формат сообщения показан на рисунке 17.5.

Сообщение об ошибке состоит из двух полей: поле разделителя ошибки и поле флага ошибки. Возможны два типа поля флага ошибки, в зависимости от вида ошибки узла, обнаружившего ее.

Если ошибку обнаружил активный узел (как в примере на рисунке 17.5), 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 (сформированных одним узлом или более). Поле флага ошибки дополняется разделителем ошибки, состоящим из 8 рецессивных битов и позволяющим перезапускать связь с шиной после обнаружения ошибки. После перехода шины в нормальное состояние узлы возобновляют передачу данных, остановленный узел повторяет передачу сообщения, переданного до этого с ошибкой.

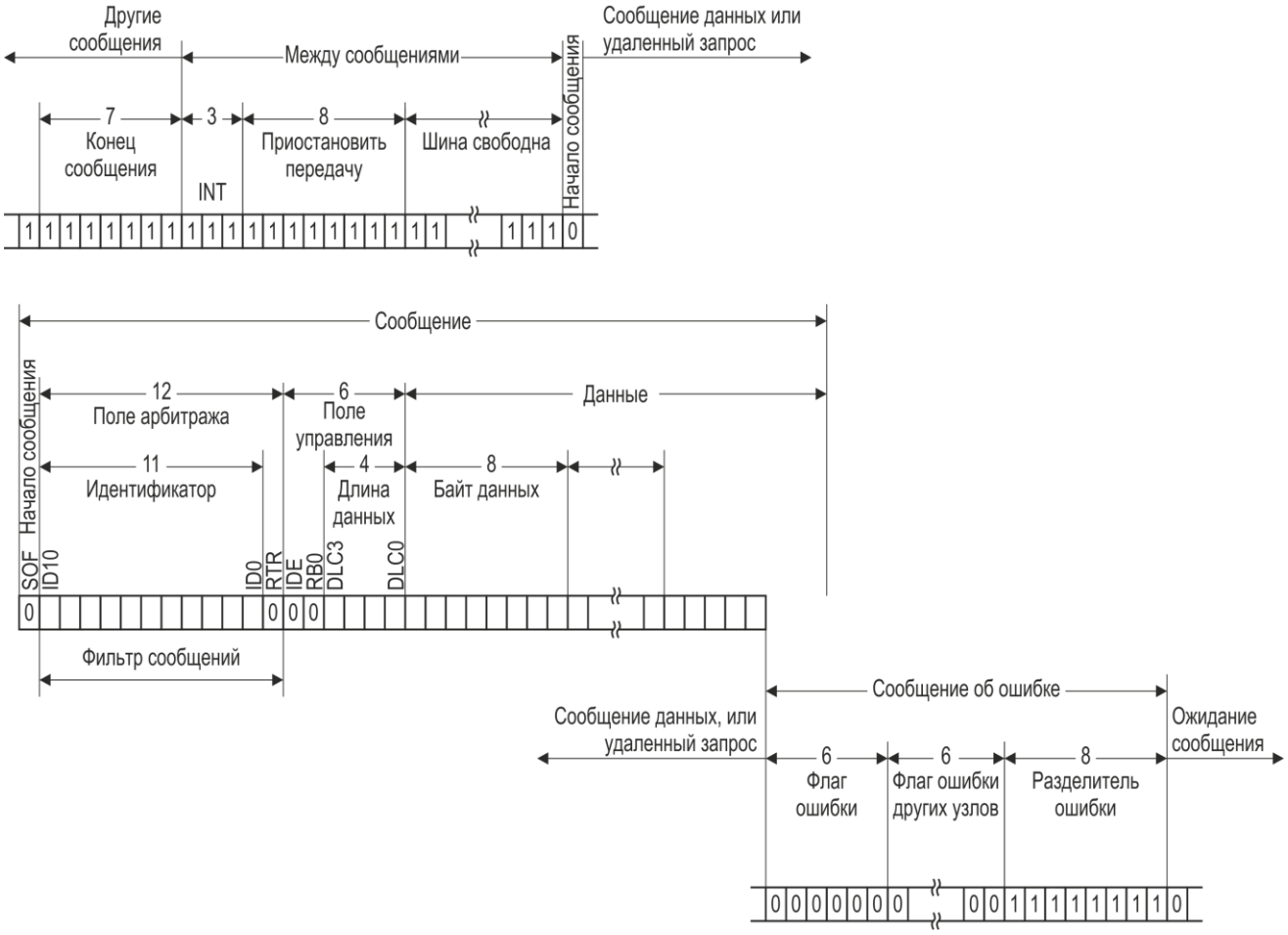


Рисунок 17.5 – Сообщение об ошибке

Если ошибку обнаружил пассивный узел, тогда он формирует флаг пассивной ошибки, состоящий из 6 последовательных рецессивных битов и затем разделитель ошибки. Таким образом, сообщение о пассивной ошибке состоит из 14 рецессивных битов. Это не нарушает правила бит-стафинга на шине и не оказывает влияния на передачи других узлов. Исключение составляет узел, который передает данные узлу, обнаружившему ошибку. В этом случае правила бит-стафинга нарушаются и передача данных прекращается. После передачи пассивной ошибки узел должен ожидать 6 последовательных рецессивных битов для восстановления связи с шиной.

Сообщение о перезагрузке

Формат сообщения о перезагрузке аналогичен формату сообщения об ошибке, но может быть сформирован только, когда шина простаивает.

Сообщение о перезагрузке проиллюстрировано на рисунке 17.6.

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.

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 до 12 доминантных битов.

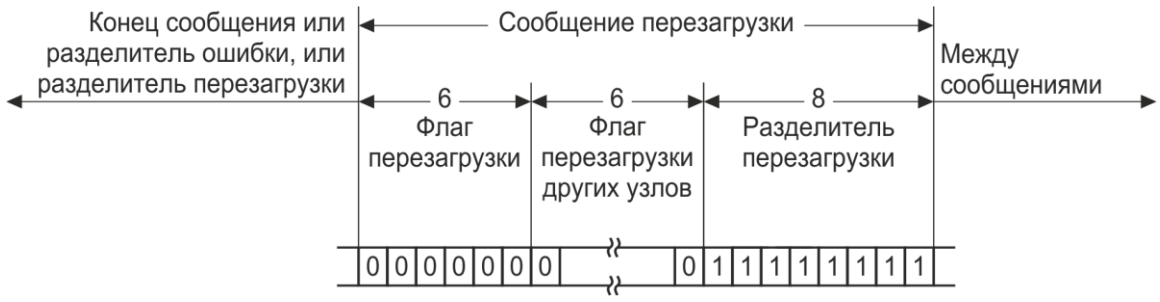


Рисунок 17.6 – Сообщение о перезагрузке

17.2 Структура и функционирование модуля CAN

В состав контроллера CAN входят два идентичных независимых узла CAN0 и CAN1, ОЗУ для хранения сообщений, которое является общим для узлов, и система управления контроллером. Между узлами есть отличие лишь в подключении к выводам микроконтроллера, в связи с этим при описании работы узлов индексы 0 и 1 будут заменены символом «x». Дополнительно в контроллере реализовано «логическое И» выходов узлов с выводом результата на выводы микроконтроллера.

Контроллер CAN имеет следующие функциональные особенности:

- соответствие ISO 11898;
- функционирование согласно спецификации CAN 2.0b (активная версия);
- отдельные управляющие регистры для каждого из двух узлов;
- программируемая скорость передачи информации до 1 Мбит/с;
- гибкий и полный контроль передачи сообщений и обработки ошибок;
- 16 линий прерываний;
- 64 объекта сообщения для хранения сообщений и их параметров в ОЗУ. Каждый объект сообщения может быть привязан к любому из узлов, сконфигурирован для передачи или приема как стандартных, так и расширенных сообщений и удаленных запросов. Каждый объект имеет индивидуальную маску для фильтрации принимаемых сообщений. Объекты сообщений могут объединяться в классы, с разными уровнями приоритета, могут объединяться для построения структур FIFO произвольных размеров (до 64 объектов в одной структуре). Кроме того, реализована возможность попарного соединения объектов для формирования шлюзов для автоматической передачи сообщений между узлами. Параллельно с вышеуказанными свойствами объекты сообщений могут организовываться в списки с постоянно доступной реорганизацией (совместимость с TwinCan-устройствами, которые не имеют списков).

Структура контроллера интерфейса CAN приведена на рисунке 17.7.

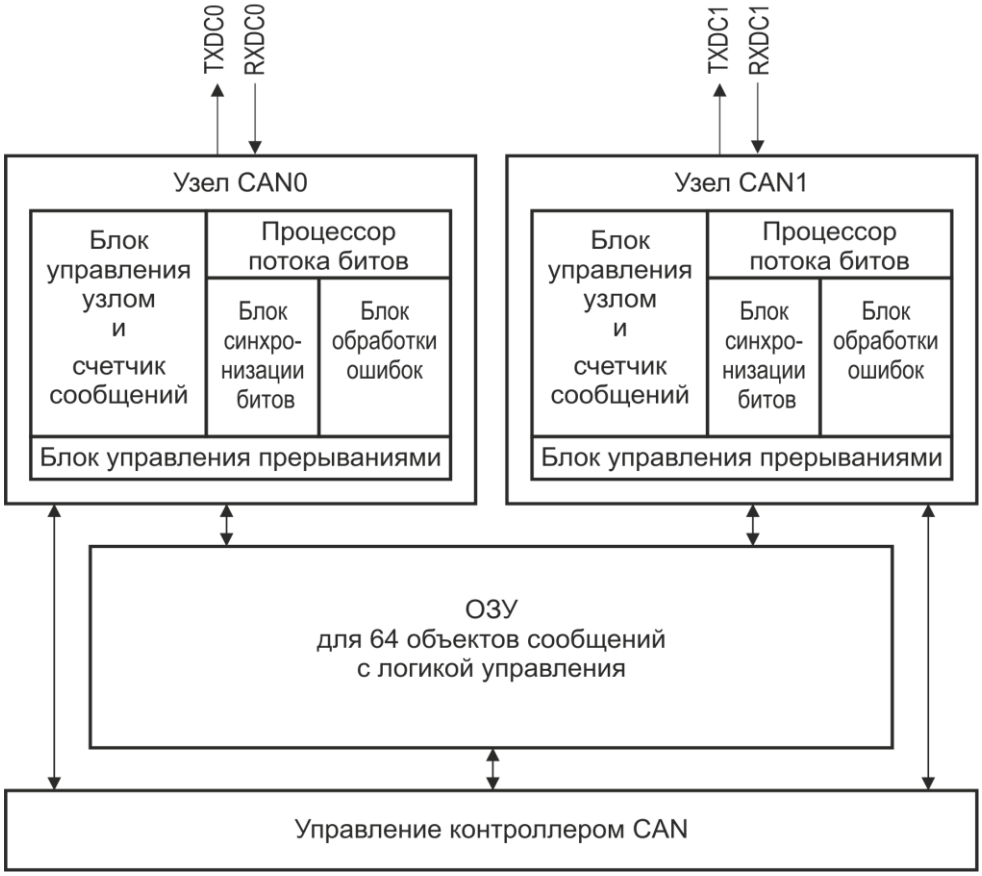


Рисунок 17.7 – Структура контроллера интерфейса CAN

Синхронизация

Тактирующим сигналом контроллера CAN является сигнал Fclc (Fin), приходящий с генератора тактовых сигналов. На основе этого сигнала посредством программируемого дробного делителя частоты формируется внутренний сигнал Fcan (Fout), синхронизирующий работу контроллера и являющийся базовым синхросигналом для передачи/приема сообщений по внешней шине CAN.

Включение контроллера CAN

По умолчанию, после сброса микроконтроллера контроллер CAN выключен. На это также указывает состояние флага DISS регистра CLC. Когда контроллер выключен, этот флаг установлен.

Для включения контроллера CAN следует записать ноль в бит DISR регистра CLC. После этого флаг DISS сбросится. Рекомендуется проверять состояние флага DISS, перед началом программирования регистров контроллера, которые не доступны в выключенном состоянии.

Выключение контроллера CAN

Программно можно перевести контроллер CAN в режим выключения установкой бита DISR. Контроллер завершает все текущие операции, после чего устанавливает флаг DISS и отключает внутреннее тактирование, в связи с чем, все регистры становятся недоступными для обращения.

Выключение контроллера CAN в режиме отладки

Во время эмуляции или отладки может возникнуть необходимость мониторинга мгновенного состояния устройства, включая все или большинство его модулей, в момент достижения программой точки останова. В этом случае может быть нежелательно, чтобы ядро контроллера CAN завершало процесс текущей передачи перед остановкой, поскольку это может стать причиной потери важных состояний контроллера. Решением

проблемы является использование режима быстрого выключения, который выбирается установкой бита FSOE (доступен только в режиме отладки и маскируется битом SBWE) регистра CLC. Если бит FSOE сброшен, то выключение можно осуществить, как указано выше – посредством бита DISR. После установки бита FSOE сигнал внешнего тактирования отключается практически мгновенно, и все процессы (в том числе и передача на шине) прекращаются, что позволяет «заморозить» контроллер CAN в состоянии, максимально приближенном к состоянию в момент достижения программой точки останова.

Простой шины

Между передачами сообщений шина CAN находится в рецессивном состоянии. Для выполнения условий простой шины необходимо, чтобы было получено, как минимум, три рецессивных бита после завершения передачи/приема очередного сообщения.

Анализ работы контроллера CAN

Для анализа работы контроллера доступны два режима – общего анализа и внутренней петли.

Режим общего анализа включается установкой бита CALM регистра NCRx узла и позволяет осуществлять независимый мониторинг работы узла, не затрагивая шину CAN. В этом режиме сообщения данных и удаленные запросы отслеживаются без участия узла в операциях на шине. Выходы 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 объектах сообщений. 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 объекта сообщений MOSTATn. Прерывания после приема генерируются (если это разрешено) для всех принятых сообщений, не содержащих ошибок.

Режим внутренней петли включается установкой бита LBM регистра NPCRx и позволяет проводить внутреннее тестирование контроллера CAN, а также отладку управляющей программы без доступа к внешней шине CAN. Внутренняя петля состоит из внутренней шины CAN (внутри контроллера CAN) и переключателя выбора шины для каждого узла (см. рисунок 17.8). С помощью переключателя каждый узел CAN может быть подключен либо к внутренней шине (режим внутренней петли), либо к внешней шине (нормальный режим работы). Если выбран режим внутренней петли, то на внешнем передающем выводе узла CAN поддерживается рецессивный уровень сигнала, а состояние принимающего вывода игнорируется.

Если оба узла CAN функционируют в режиме внутренней петли, они взаимодействуют друг с другом посредством внутренней шины CAN, не оказывая влияние на работу других модулей, функционирующих в нормальном режиме.

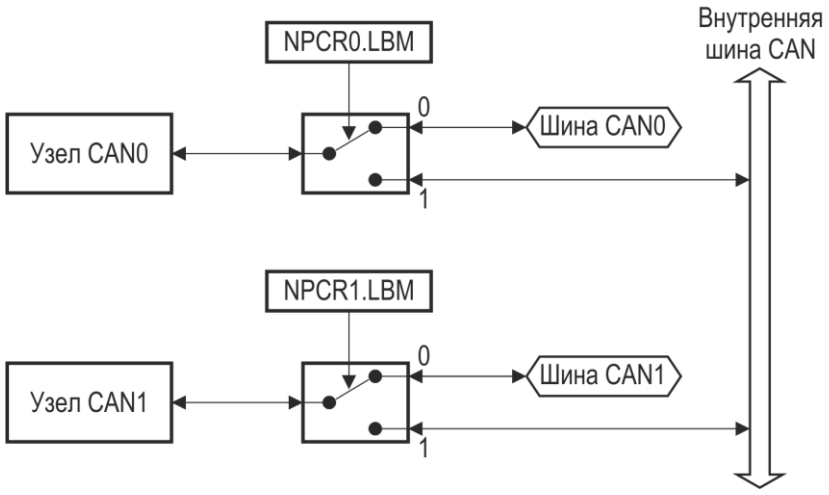


Рисунок 17.8– Режим внутренней петли

Дробный делитель

Дробный делитель позволяет генерировать частоту fout из входной тактовой частоты fin (fclc) путем программирования делителя посредством регистра FDR.

Примечание – Задаваемое значение входной частоты fin зависит от длительности передачи одного бита информации и должно быть n-кратно ей. Поскольку длительность передачи бита определяется количеством квантов времени (Ntq, см. далее), то для расчета частоты fin в МГц следует пользоваться формулой:

$$fin = n \times Ntq,$$

где Ntq – количество квантов времени tq,

n – целое число, начиная с 1 (для задания кратности).

Дробный делитель делит частоту fin путем умножения на величину 1/val или величину 1024/val для любого val от 0 до 1023, выдавая на выходе тактовый сигнал fout (fcan).

На рисунке 17.9 показана блок-схема дробного делителя. Логика дробного делителя работает по-разному, в зависимости от режима, задаваемого полем DM.

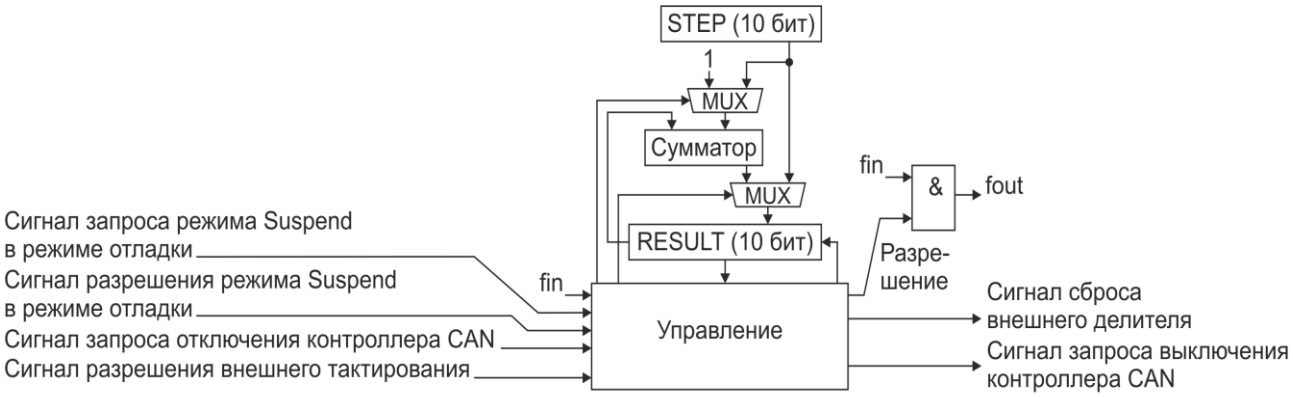


Рисунок 17.9 – Схема дробного делителя

В режиме нормального деления (DM = 01b) делитель работает как перегружаемый счетчик с шагом инкрементирования, равным единице. Состояние счетчика доступно посредством поля RESULT. Каждый раз, при переполнении (т.е. когда RESULT = 3FFh), формируется импульс сигнала Fout, после чего в счетчик загружается значение из поля STEP.

Выходная частота fout определяется по формуле:

fout = fin× 1/ (1024 – STEPd),

где STEPd – значение поля STEP в десятичном формате.

Отсюда следует, что для получения частоты fout = fin, значение STEP должно быть равно 3FFh. На рисунке 17.10 показано формирование сигнала Fout при значении STEP = = 3FDh (1021d).

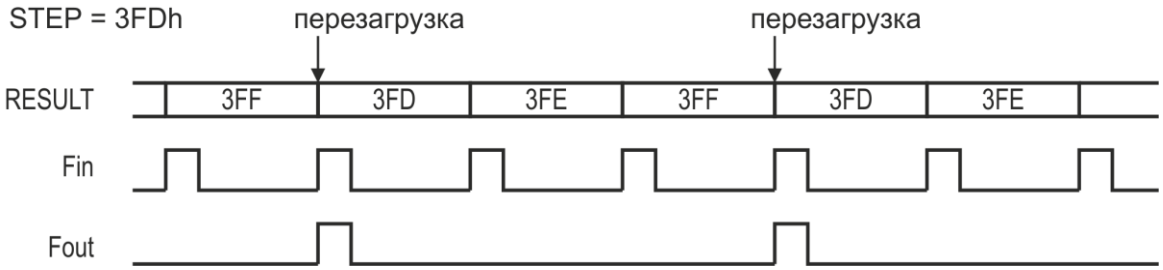


Рисунок 17.10 – Формирование сигнала с частотой fout в нормальном режиме

В режиме дробного деления (DM = 10b) делитель работает как перезагружаемый счетчик, но шаг инкрементирования в этом случае равен значению поля STEP. Если результат инкрементирования значения RESULT на величину STEP превышает 3FFh, возникает переполнение счетчика, формируется импульс сигнала Fout, после чего в счетчик загружается значение, на которое результат инкрементирования превысил 3FFh.

Выходная частота fout определяется по формуле:

fout = fin× STEPd / 1024d.

В целом, режим дробного деления позволяет программировать частоту fout с более высокой точностью, чем нормальный режим, но сигнал может иметь «джиттер» периода, не превышающий одного периода fin, в связи с чем не рекомендуется использовать режим дробного деления при высоких скоростях передач.

На рисунке 17.11 показано формирование сигнала Fout при значении STEP = 234h (564d).

fout = fin× 564/1024 = 0,55 × fin

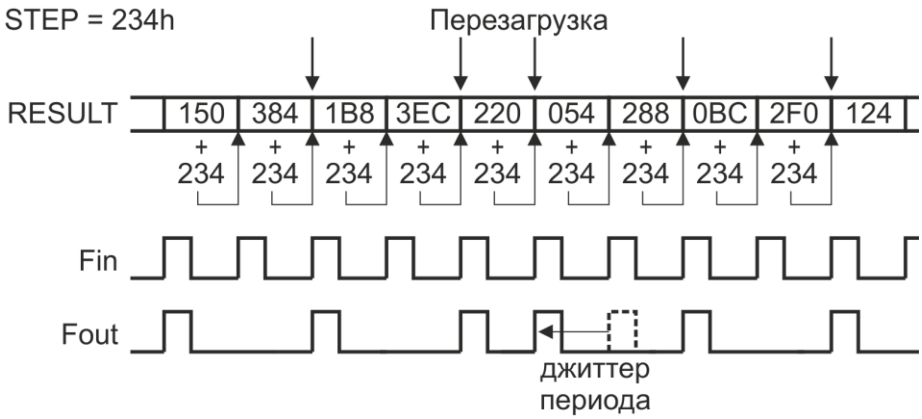


Рисунок 17.11 – Формирование сигнала с частотой fout в режиме дробного деления

Во время отладки контроллера CAN функционирование дробного делителя может управляться внутренним сигналом системы отладки. В режиме Suspend (см. далее)

регистры контроллера доступны как для записи, так и для чтения, но остальные функции контроллера «заморожены».

Вход в режим приостановки – Suspend – осуществляется с задержкой в один такт сигнала Fin после установки флага SUSACK в ответ на запрос режима Suspend, сформированный битом SUSREQ (регистр FDR) при условии, что SC≠00b (генерация выходного тактового сигнала запрещена). После запроса перехода в режим Suspend, флаг SUSACK устанавливается не сразу, а после того, как узел CAN завершит текущие операции и перейдет в состояние Idle или отключится от шины.

Мгновенный переход в режим Suspend осуществляется одновременной установкой битов SM и SUSREQ при условии, что SC≠00b.

Для функционирования в режиме Suspend биты SUSREQ и один из битов SUSACK или бит SM (в зависимости от того, как произошел переход в режим Suspend) должны оставаться установленными.

На рисунке 17.12 показана взаимосвязь различных сигналов управления делителя.

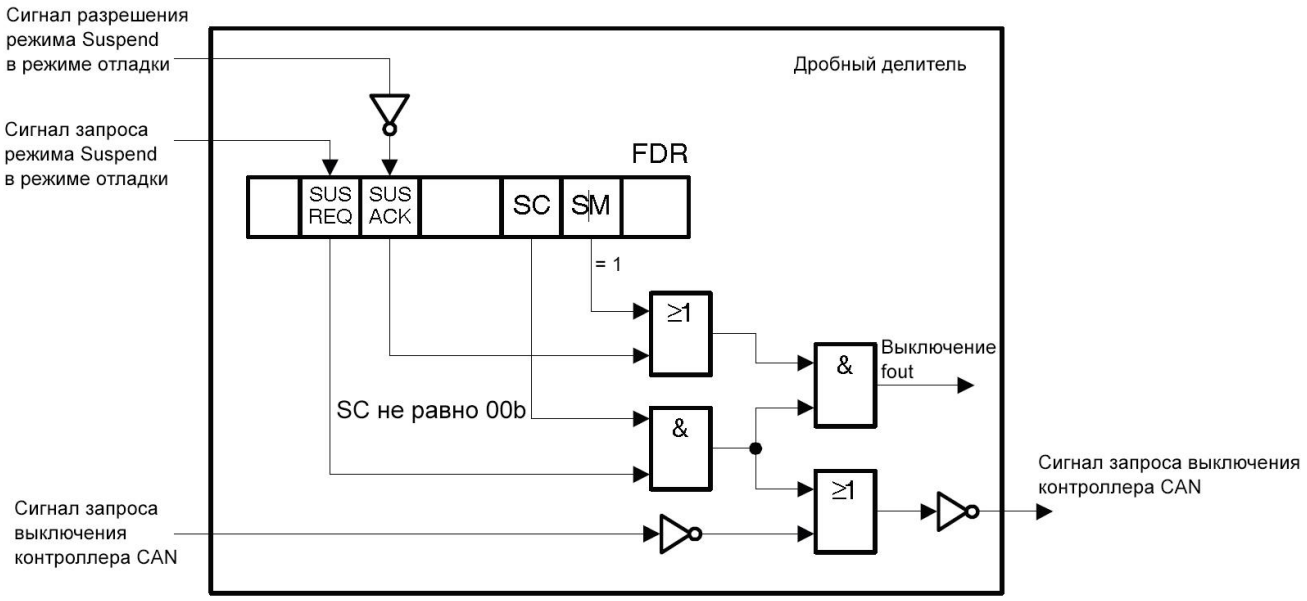


Рисунок 17.12 – Состояние сигналов запроса режима Suspend

Независимо от того, включен ли режим Suspend или нет, процесс выключения делителя начинается одновременно с возникновением запроса выключения контроллера CAN.

В таблице А.116 приложения А представлены варианты конфигурации дробного делителя в нормальном режиме работы и режиме Suspend.

Контроллер сообщений

Управляет обменом сообщениями между CAN узлами и памятью сообщений и выполняет следующие функции:

- фильтрация входящих сообщений для определения корректного объекта сообщений для сохранения полученных данных;
- определение объекта сообщений, содержимое которого будет передано в первую очередь (для каждого узла индивидуально);
- передача содержимого объекта сообщений к CAN узлу с параллельной вставкой в сообщение битов управления и состояния;
- осуществление буферизации FIFO и функционирования шлюза;
- объединение битов уведомления ждущих обработки сообщений.

Управление прерываниями модуля CAN

На рисунке 17.13 показана структура формирования запроса на прерывание.

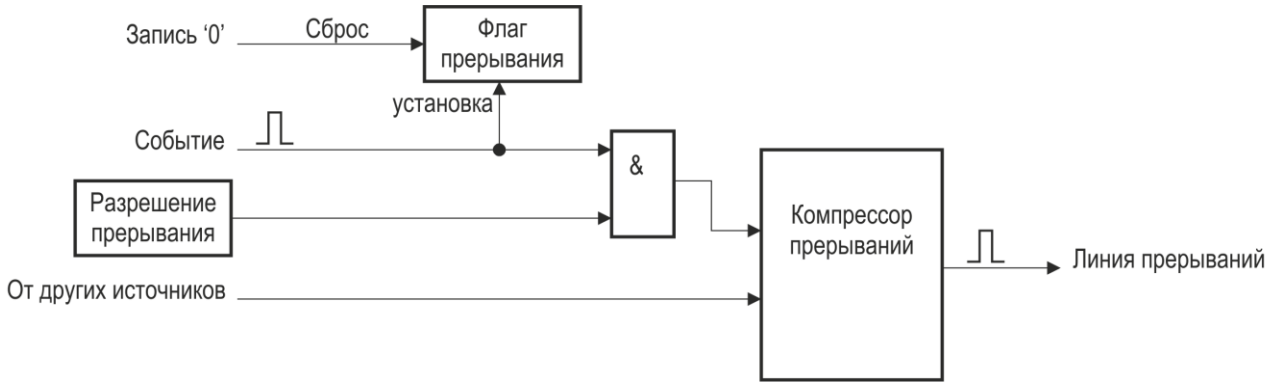


Рисунок 17.13 – Структура формирования запроса на прерывание

Событие, по которому должен быть сгенерирован запрос на прерывание, устанавливает флаг прерывания и (если разрешено) формирует запрос на прерывание на одной из 16 линий прерываний. Импульс запроса на прерывание генерируется независимо от состояния флага прерывания. Флаг прерывания может быть сброшен программно, записью нуля. 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.

Источниками прерываний являются:

- CAN узлы (8 источников – по 4 для каждого узла);
- объекты сообщений (128 источника – по 2 для каждого объекта);
- программное прерывание (источник – регистр MITR).

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. На рисунке 17.14 представлен компрессор прерываний.

Когда объект 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 объекта сообщения n. Если количество объектов 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 объектов сообщений.

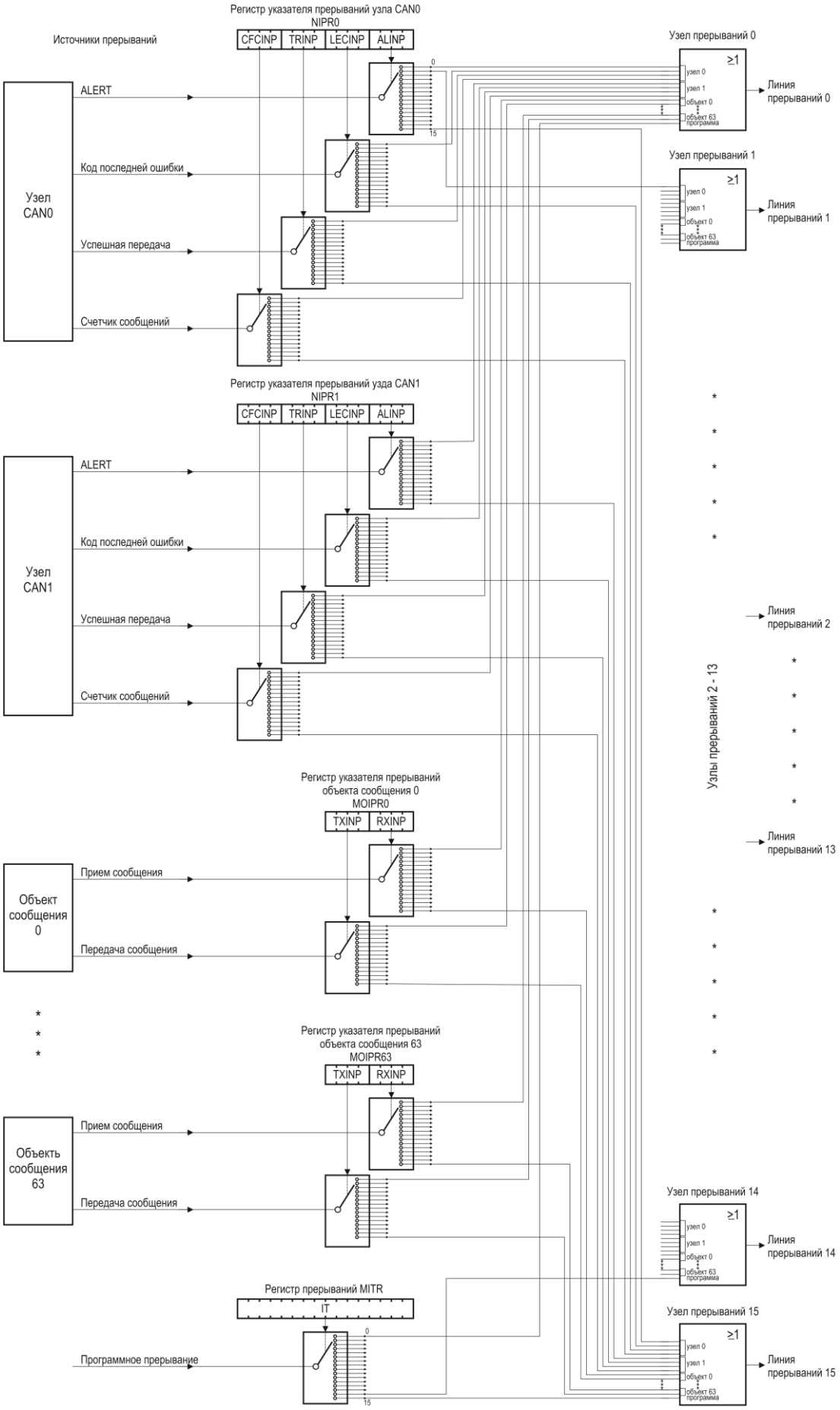


Рисунок 17.14 – Компрессор прерываний

Выводы микроконтроллера, связанные с выводами контроллера CAN

Выводы для приема и передачи сообщений закреплены за выводами портов P4, P7 и P9 (см. таблицу 2.1).

С каждым узлом CAN связано 7 входных линий. Вывод микроконтроллера, через который будут приниматься данные, выбирается посредством поля RXSEL регистра NPCR.

Режим Suspend

Переход в режим Suspend (режим приостановки работы) контроллера CAN возможен только в режиме отладки микроконтроллера. Когда прикладная программа приостановлена отладочной системой, ЦПУ начинает/возобновляет выполнение специальных отладочных программ.

Режим Suspend может быть включен системой отладки, чтобы «заморозить» состояние контроллера CAN и открыть доступ к его регистрам (каждый узел имеет бит SUSEN в регистре NCRx, который разрешает/запрещает перевод узла в режим Suspend). Для разрешения перехода контроллера в режим Suspend следует установить бит SPEN в регистре CLC и далее остановить контроллер одним из двух способов.

1 Быстрый переход в режим Suspend применяется одновременно к обоим узлам CAN и осуществляется одновременной установкой битов SUSREQ и SM регистра FDR при условии, что SC≠00b. Как только возникает запрос режима, тактовые сигналы Fin и Fout выключаются немедленно, все операции контроллера CAN прекращаются (в том числе и операции на шине) и регистры становятся недоступными. Для выхода из режима Suspend следует сбросить микроконтроллер.

Примечание – Следует помнить о том, что вероятен случай, когда быстрый переход в режим Suspend может привести к разрыву связи с другими CAN-устройствами и блокированию шины (например, в случае перехода в режим Suspend в момент передачи доминантного бита).

2 Плавный переход в режим Suspend возможен индивидуально для каждого из двух узлов (бит SUSEN регистра NCRx) и осуществляется установкой бита SUSREQ при условии, что бит SM сброшен и SC≠00b. После возникновения запроса режима тактовые сигналы не отключаются. Функционирование узла останавливается автоматически после завершения внутренних операций (например, после окончания передачи фрейма). Об окончании внутренних операций сообщает дробный делитель подачей внутреннего сигнала разрешения режима Suspend. Далее, если узлы подадут сигнал о том, что они могут быть переведены в режим Suspend – установится флаг SUSACK – дробный делитель отключит тактовый сигнал Fin и произойдет переход в режим Suspend. При плавном переходе в режим Suspend шина CAN не блокируется и все регистры доступны для чтения и записи. Отладчик контролирует остановку работы контроллера и может изменять значения регистров. Эти изменения останутся в силе после выхода из режима Suspend (очисткой бита SPEN).

Примечание – В режиме отладки любой из двух узлов может быть переведен в режим Suspend, даже если он не принимал участие в работе контроллера CAN (т.е. был отключен).

17.3 Узел модуля CAN

Каждый узел CAN имеет свою собственную логику управления и выдачи информации о состоянии и может быть сконфигурирован и работать независимо от другого узла.

Режим конфигурации включается установкой бита CCE регистра NCRx. Режим конфигурации позволяет изменять параметры синхронизации битов и состояния счетчиков ошибок.

Конфигурация прерываний задается битами TRIE, ALIE и LECIE:

- бит TRIE управляет разрешением прерывания после передачи сообщения;

- бит ALIE управляет разрешением прерываний по ошибке.

- бит LECIE управляет разрешением прерывания по коду последней ошибки.

Регистр NSRx отражает текущее состояние, содержит информацию о передачах и ошибках узла.

Блок управления узлом

Координирует работу:

- разрешает/запрещает действия узла на шине;

- разрешает/запрещает и генерирует различные события, касающиеся работы узла (ошибка на шине, успешное завершение передачи сообщения), которые приводят к формированию запросов на прерывания;

- управляет счетчиком сообщений.

Блок синхронизации битов

Согласно стандарту ISO 11898 время передачи одного бита разделено на сегменты, которые, в свою очередь, составлены из целочисленных отрезков времени, называемых квантами времени tq (см. рисунок 17.15). Квант времени – фиксированная единица времени, получаемая из частоты синхронизации и делителя модуля CAN.

Сегмент синхронизации Tsync позволяет синхронизировать начало обмена данными между передатчиком и приемником. Длительность сегмента всегда равна одному кванту времени.

Сегмент распространения – Tprop. 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.

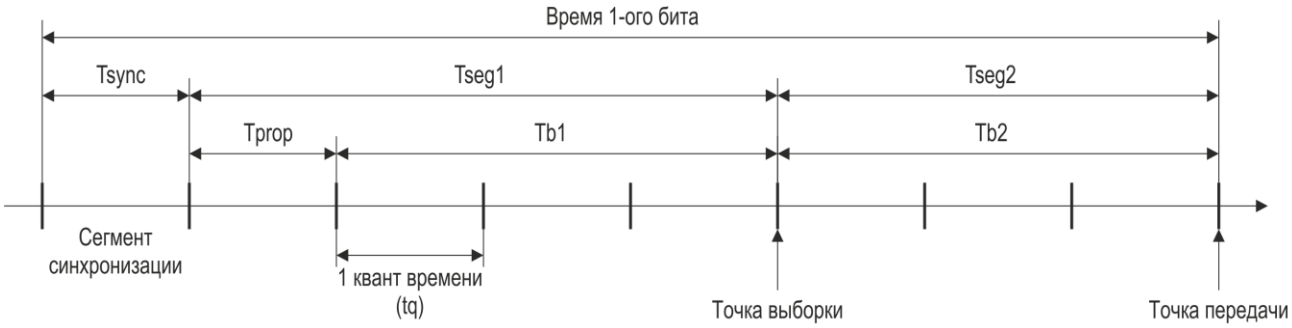


Рисунок 17.15 – Структура одного бита

Сегменты буфера фазы 1 и буфера фазы 2 – Tb1 и Tb2, 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 (Tseg1), который определяется битовым полем TSEG1 регистра синхронизации битов NBTRx (может быть записан, только если установлен бит CCE регистра NCRx). Согласно стандарту ISO, минимальная длительность сегмента параметра 1 должна составлять три кванта времени.

Сегмент параметра 2 (Tseg2) определяется битовым полем TSEG2 регистра NBTRx и охватывает сегмент буфера фазы 2. Минимальная длительность сегмента параметра 2 составляет два кванта времени.

Согласно стандарту ISO, минимальная длительность одного бита, получающаяся сложением сегментов Tsync, Tseg1 и Tseg2 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 Tbit:

при DIV8 = 0 значение кванта времени tq = (BRP + 1) / fout;
при DIV8 = 1 значение кванта времени tq = 8 × (BRP + 1) / fout;
Tsync = 1 × tq;
Tseg1 = (TSEG1 + 1) × tq ≥ 3tq;
Tseg2 = (TSEG2 + 1) × tq ≥ 2tq;
Tbit = Tsync + Tseg1 + Tseg2 ≥ 8tq.

Чтобы компенсировать смещение фазы между частотами генераторов различных узлов шины, каждое устройство должно синхронизироваться по фронту смены уровня сигнала на шине от рецессивного к доминантному. Как только фронт обнаруживается, логика синхронизации сравнивает его текущее положение с ожидаемым и выполняет настройку значений параметров Tseg1 и Tseg2.

Контроллер CAN использует два механизма синхронизации – аппаратный и ресинхронизацию (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 Tseg1 или укорачиванием сегмента Tseg2. Максимальное значение изменения сегментов 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.

Если величина фазового смещения меньше или равна запрограммированному значению ширины перехода ресинхронизации Tsjw, выполняется аппаратная синхронизация.

Если величина смещения фазы больше, чем Tsjw, а фазовое смещение положительно, то удлиняется сегмент Tseg1, в случае отрицательного фазового смещения укорачивается сегмент Tseg2.

Значение Tsjw определяется полем SJW регистра NBTRx по формуле:

Tsjw = (SJW + 1) × tq.

Помимо прочего, должны соблюдаться следующие правила:

Tseg1 ≥ Tsjw + Tprop и Tseg2 ≥ Tsjw.

Соотношения между максимальным отклонением частоты fout и сегментами буферов фаз и шириной перехода ресинхронизации следующие:

$\Delta fout \le T/2 \times (13 \times Tbit - Tb2)$,

$\Delta fout \le Tsjw / 20 \times Tbit$,

где T – меньшее из Tb1 и Tb2.

В итоге:

- Tsync составляет 1 квант времени;
- Tprop – от 1 до 8 квантов времени;
- Tb1 – от 1 до 8 квантов времени;
- Tb2 – выбирается равным двум квантам времени или равным сегменту Tb1, если его значение более двух квантов времени;
- Tsjw может составлять максимально 4 кванта времени, однако, в типовых приложениях достаточно 1.

Корректные значения параметров синхронизации битов должны быть записаны в регистр NBTRx (доступен, если установлен бит CCE) до окончания инициализации (до сброса бита INIT регистра NCRx), т.е. до начала работы CAN узла.

Процессор потока битов

Процессор потока битов формирует (на основе содержимого объектов сообщений) сообщения данных и удаленные запросы непосредственно перед отправкой на шину CAN. Процессор потока управляет генератором CRC (генератор контрольной суммы) 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 NSRx.

Корректность получаемых данных проверяется и подтверждается или не подтверждается кодом CRC. В случае неподтверждения 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 (поле REC в регистре NECNTx) и счетчик ошибок передачи (поле TEC). Инкрементированием и декрементированием счетчиков управляет процессор потока битов.

Если процессор потока битов сам выявляет ошибку в процессе передачи, то счетчик ошибок передачи (поле TEC) инкрементируется на 8. Инкрементирование на 1 происходит, если об ошибке сообщено внешним CAN-устройством путем генерирования сообщения об ошибке. Направление передачи с ошибочным сообщением и узел, сообщивший об ошибке передачи, указывают на соответствующие узлы CAN в регистрах NECNTx, что используется для анализа ошибки.

В зависимости от значений счетчиков ошибок узел CAN может находиться в одном из трех состояний:

- активной ошибки;
- пассивной ошибки;
- отключен от шины.

Узел находится в состоянии активной ошибки, если значение каждого из счетчиков ошибок меньше 128. Узел в состоянии активной ошибки присоединен к шине и посылает флаг активной ошибки при обнаружении ошибок.

Узел находится в состоянии пассивной ошибки, если значение хотя бы одного из счетчиков ошибок больше или равно 128. Узел подключен к шине, но при обнаружении

ошибок посылает флаг пассивной ошибки. После передачи узел в состоянии пассивной ошибки будет ждать инициализации дальнейшей передачи.

Узел находится в состоянии отключения от шины, если значение счетчика ошибок TEC больше или равно 256. О том, что CAN узел находится в состоянии отключения от шины, сигнализирует флаг BOFF регистра NSRx. Узел в состоянии отключения от шины не может работать с шиной (выходные передатчики отключены).

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 перестанут превышать лимит ошибок, флаг EWRN сбросится.

Счетчик сообщений

Счетчик сообщений может использоваться для проверки последовательности передаваемых битов объектом сообщений или получения информации о завершении передачи/приема сообщения соответствующего узла CAN. Подсчет сообщений осуществляется 16 разрядным счетчиком, который управляется регистром NFCRx. Битовые поля CFMOD и CFSEL определяют режим работы и событие для инкрементирования счетчика.

Каждый узел CAN имеет в своем составе 16-разрядный счетчик сообщений/ синхросчетчик, который подсчитывает количество принятых и переданных сообщений. Битовое поле CFSEL определяет один из трех режимов работы счетчика.

В режиме подсчета сообщений после успешной передачи и/или успешного приема сообщения, содержимое счетчика копируется в битовое поле CFCVAL регистра MOIPRn объекта сообщений n, участвующего в пересылке данных. После чего счетчик сообщений инкрементируется.

Прерывания узла CAN

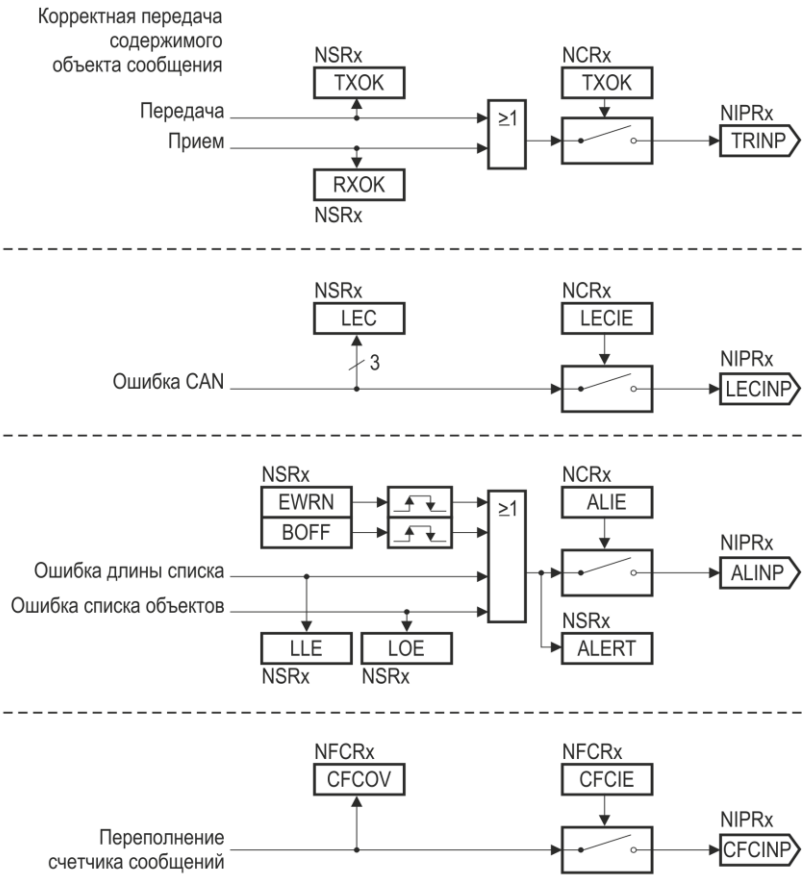


Рисунок 17.16 – Прерывания CAN узла

Узел может генерировать запросы на прерывания (см. рисунок 17.16) в случае:

- успешной передачи/приема сообщения;
- обнаружения кода последней ошибки;
- переполнения счетчика сообщений;
- состояния ALERT (состояние, возникающее, когда хотя бы один из счетчиков ошибок узла достиг значения своего лимита, изменяется состояние «отключен от шины», возникает ошибка длины списка или ошибка списка объектов).

После каждой успешной передачи или успешного приема сообщения генерируется (если разрешено соответствующими битами TXOK и RXOK) прерывание. Битовое поле TRINP регистра NIPRx задает одну (из 16) линию прерывания.

Прерывание узла при возникновении кода последней ошибки формируется (если разрешено битом LECIE), если после модификации поля LEC его значение больше нуля. Битовое поле LECINP задает линию прерывания.

Прерывание узла при переполнении счетчика сообщений генерируется, если оно разрешено битом CFCIE регистра NFCRx. Битовое поле CFCINP задает линию прерывания.

ALERT прерывание может быть сформировано (если разрешено битом ALERT) любым из следующих событий:

- изменение состояния бита BOFF;
- изменение состояния бита EWRN;
- ошибка длины списка, которая также выставляет бит LLE;
- ошибка элемента списка, которая также выставляет бит LOE;
- бит INIT выставлен аппаратно.

Битовое поле ALINP задает линию прерывания.

В дополнение к аппаратным прерываниям есть возможность программного генерирования прерываний с использованием регистра прерываний MITR. Запись единицы в n-й разряд битового поля IT генерирует сигнал запроса прерывания на соответствующей ему n-ой линии прерываний (одной из 16). Установка нескольких битов приводит к параллельному генерированию запросов прерываний на соответствующих установленным битам линиях прерываний.

17.4 Объекты сообщений

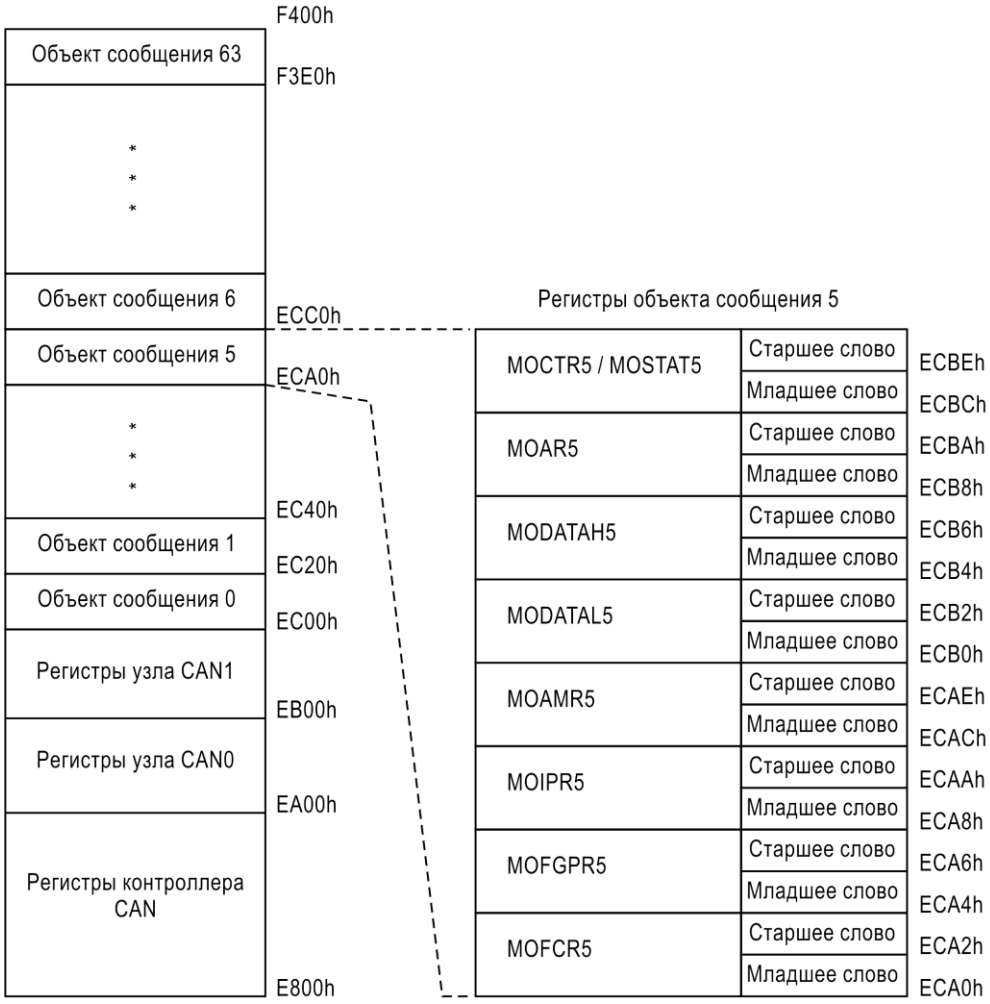


Рисунок 17.17 – Структура объектов сообщений

Регистры управления и состояния объектов сообщений

В состав каждого объекта сообщения входят девять 32-разрядных регистров:

- управления и состояния – MOCTRn (только запись) и MOSTATn (только чтение), расположенные по одному адресу;
- арбитража – MOARn;
- данных – MODATAHn и MODATALn;
- маски – MOAMRn;
- указателя прерываний – MOIPRn;
- указателя FIFO/шлюза – MOFGPRn;
- управления функционированием – MOFCRn.

Расположение регистров представлено на рисунке 17.17, где для примера взят пятый объект сообщения. Расположение регистров всех объектов сообщений идентично.

Контроллер списка

Структура списка объектов сообщений

Объекты сообщений модуля CAN могут быть организованы в восемь списков (см. рисунок 17.18).

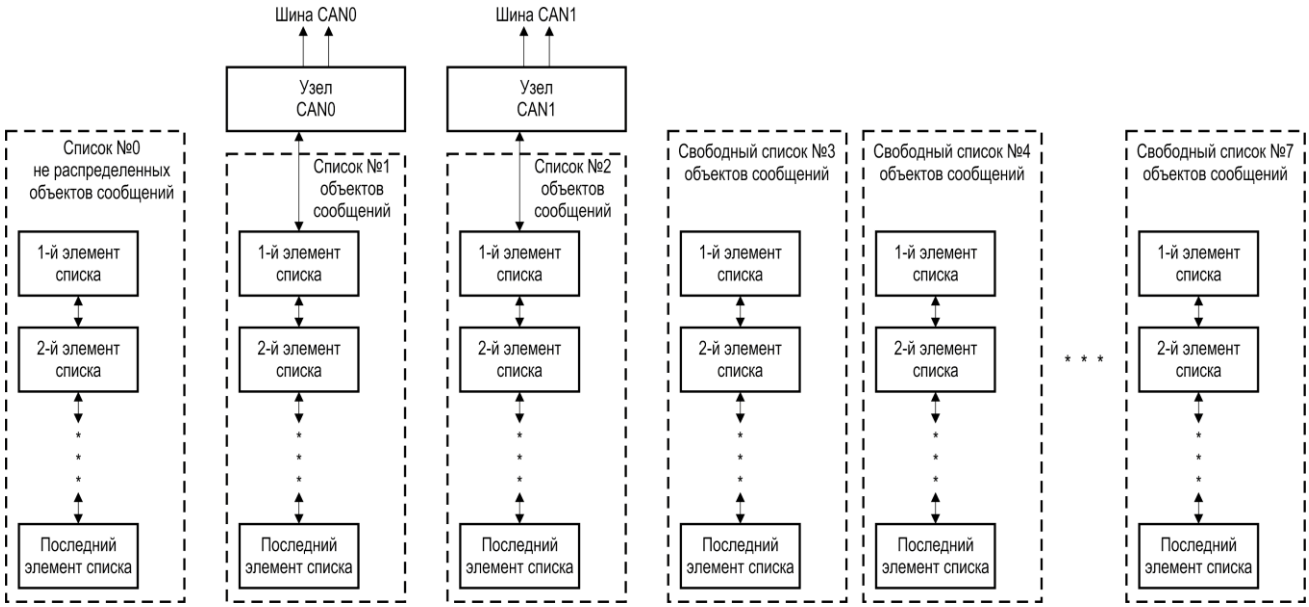


Рисунок 17.18 – Списки контроллера CAN

Каждый объект сообщения может быть добавлен в один из списков. Каждый узел CAN имеет свой список и соответствующий регистр списка. Регистр LIST1 отражает состояние списка №1 узла CAN0, регистр LIST2 – списка №2 узла CAN1.

Примечание – Узел может оперировать только с теми объектами сообщений, которые занесены в принадлежащий ему список.

Положение объекта сообщения n в списке определяется посредством регистра MOSTATn, который содержит указатели на предшествующий ему и следующий за ним элементы списка (объекты). Нераспределенные между узлами CAN объекты сообщений по умолчанию организуются в отдельный список №0, состояние которого отражается в регистре LIST0. Остальные пять списков с номерами от 3 до 7 являются свободными (не принадлежат ни одному узлу) и имеют соответствующие регистры LIST3 – LIST7.

Примечание – Объекты сообщений, распределенные в списки с 3 по 7, не могут быть использованы узлами CAN.

Механизмы FIFO и шлюза (см. далее) оперируют с объектами сообщений независимо от их распределения по спискам, что дает возможность работы со всеми восемью списками. Следовательно, при использовании механизмов FIFO и шлюза следует внимательно следить за содержимым списков.

На рисунке 17.19 представлен вариант, когда объекты сообщений с номерами 3, 5 и 16 занесены в список № 2, принадлежащий узлу CAN1. Состояние списка отражено в регистре LIST2.

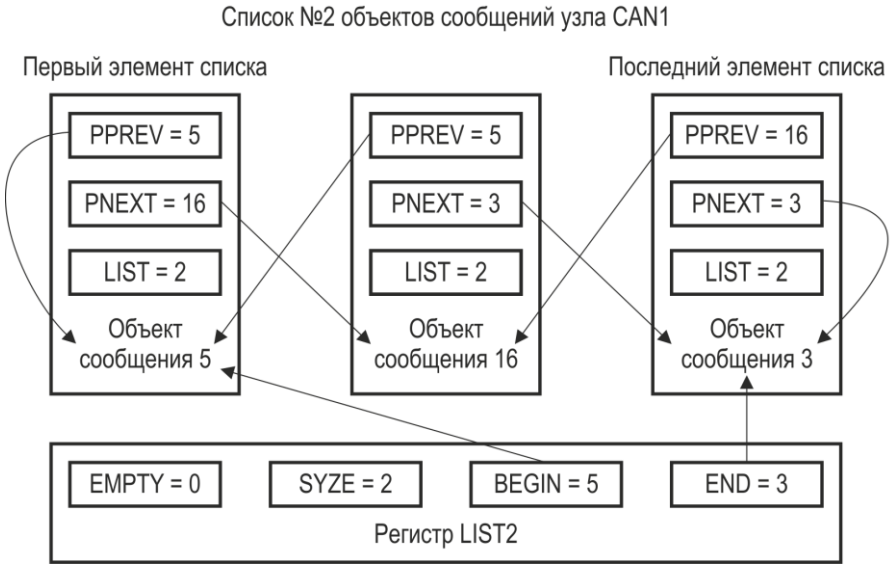


Рисунок 17.19 – Пример списка объектов сообщений

Значение поля BEGIN регистра LIST2 указывает на первый элемент списка (объект сообщения 5). Значение поля END указывает на последний элемент списка (объект сообщения 3). Количество элементов списка (количество объектов сообщений в списке) отражается в поле SIZE (значение SIZE всегда на единицу меньше количества элементов списка). Бит EMPTY является индикатором заполнения списка. Если список пуст, бит EMPTY установлен, в противном случае бит сброшен.

Каждый объект сообщений содержит номер списка (поле LIST регистра MOSTATn), к которому он относится, а также указатели PNEXT и PPREV на следующий по списку объект сообщения и предшествующий, соответственно. Поле PPREV первого по списку объекта сообщения должно указывать на этот же объект. Поле PNEXT последнего по списку объекта сообщения должно указывать на этот же объект.

На рисунке 17.14 указатель PPREV пятого объекта сообщения (первого в списке) имеет значение 5h, а указатель PNEXT третьего объекта сообщения (последнего в списке) имеет значение 3h. Значение поля LIST всех трех объектов сообщений равно 2h.

Объект сообщения, у которого LIST = 0h относится к нулевому списку нераспределенных объектов. После сброса все объекты сообщений считаются нераспределенными. По умолчанию, порядок элементов списка № 0 следующий: объект сообщений n–1 является предыдущим объекта сообщения n, а объект сообщения n+1 – следующим.

Панель команд списка

Для просмотра структуры списка объектов сообщений узла достаточно обратиться к соответствующим регистрам LIST1/LIST2 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.

Регистр панели команд PANCTR полностью определяет действия контроллера списка по построению и реорганизации списков объектов сообщений.

Панель команд запускается записью соответствующей команды в битовое поле PANCMD. До записи кода команды в поле PANCMD должны быть записаны соответствующие аргументы команды в битовые поля PANAR1 и PANAR2.

Примечание – Запись новых значений в поля PANAR1 и PANAR2 не изменяет сразу их содержимого. Новые значения сначала попадают в специальный теневой регистр.

Далее, одновременно с записью кода команды в поле PANCMD, новые значения из теневого регистра переносятся в поля PANAR1 и PANAR2.

С записью корректного кода команды выставляется флаг BUSY 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 микроконтроллера контроллер списка формирует список №0 нераспределенных объектов сообщений. Во время этой операции флаг BUSY установлен и все обращения к ОЗУ объектов сообщений запрещены. ОЗУ становится доступным только после сброса флага BUSY.

Примечание – После сброса ОЗУ объектов сообщений автоматически инициализируется контроллером списка для обеспечения каждого объекта сообщений 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 переносится в другой указанный список, наряду с битом BUSY, устанавливается бит RBUSY. Это указывает на то, что значения битовых полей PANAR1 и PANAR2 будут обновлены контроллером списка следующим образом:

- номер объекта сообщения, переносимого из списка № 0 нераспределенных объектов сообщений, записывается в PANAR1;

- если установлен бит ERR (седьмой бит поля PANAR2), значит, список №0 пуст и выполнение команды завершается; если бит ERR сброшен – 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 сбрасывается. Это позволяет пользователю запрограммировать настройки желаемого объекта сообщения, в то время как контроллер списка распределяет объекты. Во время операций со списками доступ к объектам сообщений не запрещен, но следует помнить, что любой доступ к регистрам ОЗУ объектов сообщений в течение процесса распределения объектов вносит задержку (в процесс), равную длительности доступа.

Код команды «нет операции» автоматически записывается в битовое поле PANCMD.

Новая команда может быть записана в любое время, когда бит BUSY сброшен.

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 установленным, 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 объект сообщения, занесенный в список активного узла CAN, будет перенесен на другую позицию этого же списка или перенесен в другой список, бит MSGVAL регистра MOSTATn объекта сообщения n должен быть очищен.

Примечание – Если требуется перераспределить объекты сообщений в списки повторно, необходимо приостановить работу узлов CAN (установить бит INIT регистра NCRx), а после занесения объектов в списки возобновить ее (сбросить бит INIT).

17.5 Прием сообщения

После завершения приема сообщение сохраняется в объекте сообщения в соответствии с установленным алгоритмом (см. рисунок 17.20). Помимо сохранения данных в объекте сообщения, модуль CAN осуществляет обмен данными с ЦП.

Бит MSGVAL корректности объекта сообщения

При приеме сообщения информация сохраняется в объекте сообщения только в том случае, если установлен бит MSGVAL регистра MOSTATn. Если ЦП очищает бит MSGVAL, модуль CAN останавливает запись в объект сообщения, и далее объект может быть реконфигурирован центральным процессором с последующей записью в него информации без участия модуля CAN.

Бит RTSEL распределения объекта сообщений

Реконфигурация объекта сообщения центральным процессором во время работы модуля CAN (например, сброс бита MSGVAL, изменение объекта сообщения и повторная установка бита MSGVAL) происходят следующим образом:

- объект сообщения получает приоритет;
- ЦП очищает бит MSGVAL для реконфигурации объекта сообщения;
- после реконфигурации ЦП снова устанавливает бит MSGVAL;
- завершается получение сообщения;
- если установлен бит MSGVAL полученные данные сохраняются в объекте сообщения, генерируется запрос на прерывание, устанавливается соответствующий флаг;
- если сконфигурировано, производятся шлюзовые и FIFO операции.

Примечание – После реконфигурации объекта сохранение данных по завершении получения сообщения может быть нежелательным. Запретить запись данных в объект сообщения можно посредством бита RTSEL.

После получения объектом сообщения приоритета его бит RTSEL устанавливается контроллером CAN, открывая, таким образом, объект сообщения для записи. После приема сообщения контроллер CAN дополнительно проверяет возможность записи в объект сообщения, а именно – установлен ли все еще бит RTSEL. И только в том случае, если бит RTSEL установлен, полученные данные сохраняются в объекте сообщения (вместе со всеми последующими действиями, которые указаны выше).

Если во время операций контроллера CAN объект сообщения становится некорректным (сброс бита MSGVAL), бит RTSEL должен быть сброшен до того, как бит MSGVAL будет установлен снова, или, по крайней мере, одновременно с ним. Это необходимо для предотвращения сохранения старой информации в объекте сообщения.

Реконфигурация объекта сообщения должна происходить следующим образом:

- сброс бита MSGVAL;
- реконфигурация объекта сообщения, пока бит MSGVAL сброшен;
- сброс бита RTSEL и далее установка бита MSGVAL.

Бит RXEN разрешения приема

Полученное с шины сообщение может быть сохранено в объекте сообщения только в случае, если установлен бит RXEN. Контроллер CAN проверяет состояние бита RXEN только во время фильтрации (см. далее) принимаемого сообщения. После того, как сообщение принято, состояние бита не имеет значения и не оказывает влияния на дальнейшее сохранение данных в объекте сообщения.

Бит RXEN позволяет управлять блокированием объекта сообщения – после сброса бита RXEN полученное сообщение сохраняется в объекте сообщения, который получил приоритет, но в сохранении последующих сообщений этот объект не принимает участия.

Флаги RXUPD, NEWDAT и MSGLST

Индикатором процесса сохранения (изменения) данных в объекте 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.

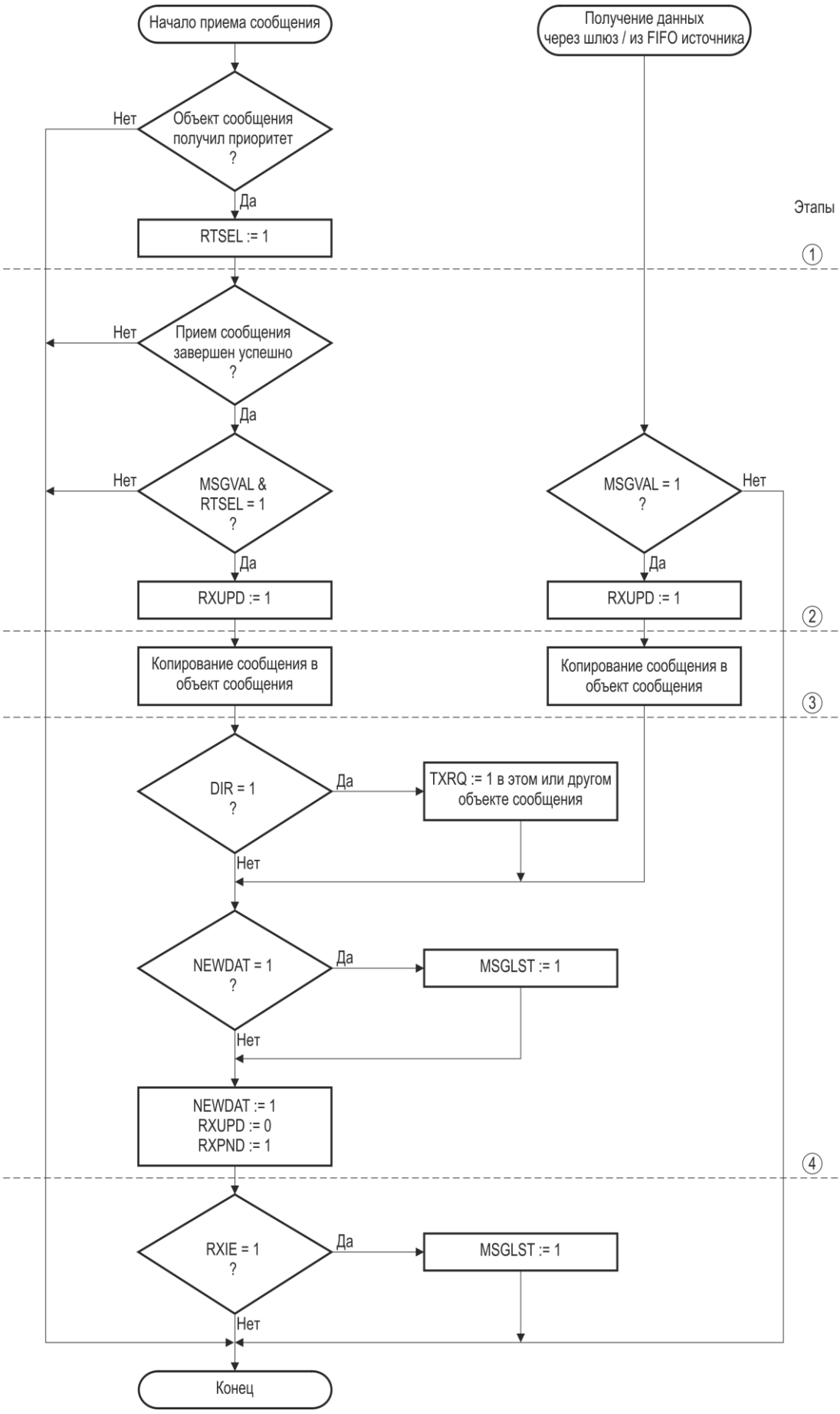


Рисунок 17.20 – Прием сообщения

Флаги RXUPD и NEWDAT позволяют произвести чтение корректных данных из объекта сообщения во время текущих операций контроллера CAN. Рекомендуемая последовательность действий следующая:

- сброс флага NEWDAT;
- чтение данных (идентификатор, данные и т. д.) из объекта сообщения;
- проверка флагов NEWDAT и RXUPD – оба флага должны быть сброшены. В случае невыполнения этого условия возвращение к первому действию;
- если флаги NEWDAT и RXUPD сброшены, то содержимое объекта сообщения корректно и не используется контроллером CAN в течение операции чтения.

Поведение флагов RXUPD, NEWDAT и MSGLST идентично как для сообщений данных, так и для сообщений удаленных запросов.

17.6 Передача сообщения

Алгоритм передачи сообщений показан на рисунке 17.21. Одновременно с копированием данных (идентификатора, бита IDE, бита RTR, равного биту DIR, кода длины данных и собственно данных) из объекта сообщения, содержимое которого должно быть передано во внутренний передающий буфер соответствующего узла CAN, для контроля соблюдения четкой последовательности выполнения всех операций устанавливаются биты состояния.

Биты MSGVAL, TXEN0, TXEN1 и TXRQ

Сообщение может быть передано только в случае, когда все четыре бита установлены.

Бит RTSEL

Бит RTSEL выставляется после того, как объект сообщения получает приоритет для передачи своего содержимого. Когда данные объекта сообщения копируются в передающий буфер, бит RTSEL проверяется, и если он установлен, сообщение передается. После успешной передачи сообщения бит RTSEL проверяется снова, и если он установлен, осуществляются дальнейшие операции.

Для полной и завершенной реконфигурации корректного объекта сообщения должны быть выполнены следующие шаги:

- очистка бита MSGVAL;
- реконфигурация объекта сообщения, пока бит MSGVAL сброшен;
- сброс бита RTSEL и установка бита MSGVAL.

Сброс бита RTSEL гарантирует как полное отключение объекта сообщения от текущей передачи, так и то, что никакие операции (копирование данных в передающий буфер, включая сброс бита NEWDAT, очистка бита TXRQ, прерывание сообщения и т. д.), относящиеся к старой конфигурации этого объекта сообщения, не повлияют на новую конфигурацию после установки бита MSGVAL.

Флаг NEWDAT

После завершения передачи содержимого объекта сообщения в передающий буфер узла CAN, флаг NEWDAT аппаратно сбрасывается, тем самым обозначая, что объект сообщения открыт для записи новых данных.

Если после успешной передачи сообщения (на CAN-шину) флаг NEWDAT все еще остается сброшенным (в объект сообщения не были записаны новые данные), флаг TXRQ аппаратно сбрасывается. Если же флаг NEWDAT был установлен программно (в связи с необходимостью передачи новых данных), флаг TXRQ не сбрасывается, тем самым разрешая передачу новых данных.

Дополнительные режимы передачи

Дополнительно имеются два режима, каждый из которых может быть выбран индивидуально:

- режим передачи данных с защитой от повторений;
- режим однократной пересылки данных.

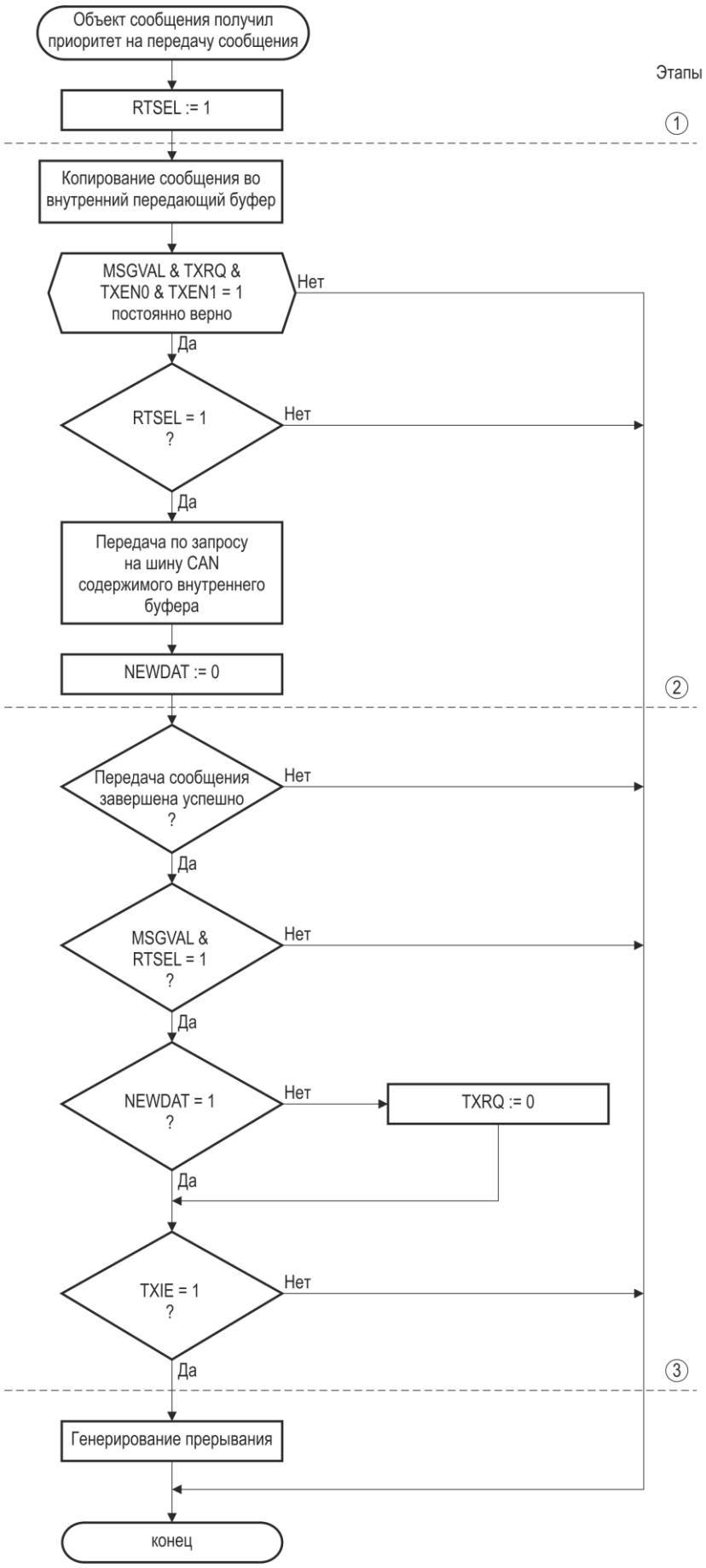


Рисунок 17.21 – Передача сообщения

Режим передачи данных с защитой от повторения

Выбирается установкой бита SDT регистра MOFCRn.

После приема сообщения и сохранения его в выбранном объекте сообщения, бит MSGVAL этого объекта аппаратно сбрасывается, что делает объект некорректным, а, следовательно, недоступным для последующей записи.

После однократной успешной передачи данных бит MSGVAL передающего объекта сообщения будет сброшен, и объект станет недоступным для передачи.

При приеме сообщения удаленного запроса поведение объекта сообщения n, который его принял, зависит от состояния бита FRREN регистра MOFCRn этого объекта.

Если бит FRREN установлен, то в ответ на сообщение удаленного запроса будут переданы данные из объекта, на который указывает поле CUR объекта сообщения, принявшего удаленный запрос. После успешной передачи бит MSGVAL объекта принявшего удаленный запрос будет сброшен. Объект сообщения, передавший данные, останется корректным.

Примечание – Если бит FRREN сброшен, то в ответ на сообщение удаленного запроса объект сообщения выполнит передачу собственных данных. По окончании передачи бит MSGVAL этого объекта сообщения не сбросится, т.е. объект останется корректным и может участвовать в дальнейших операциях (защита от повторений не включается).

Режим однократной пересылки данных

Выбирается установкой бита STT регистра MOFCRn.

Бит TXRQ сбрасывается, когда содержимое объекта сообщения копируется в передающий буфер узла CAN. Таким образом, в дальнейшем, при неудачной (вследствие ошибок) пересылке сообщения по CAN-шине, повторной передачи не будет.

17.7 Фильтрация сообщений

Контроллер CAN использует фильтрацию для контроля приема и передачи сообщений.

Фильтрация при получении сообщений

При получении узлом CAN сообщения определяется объект сообщения, в котором будут сохранены получаемые данные в случае успешного приема.

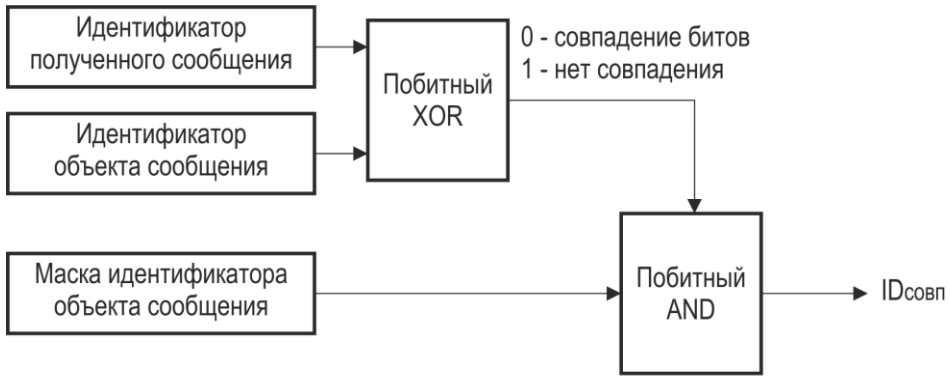


Рисунок 17.22 – Проверка идентификатора полученного сообщения

Объект сообщения считается корректным для приема, если одновременно соблюдаются условия:

-объект сообщения распределен в список объектов сообщений узла, который принимает сообщение;

- бит MSGVAL установлен;

- бит RXEN установлен;

- бит DIR равен биту RTR принимаемого сообщения. Если бит DIR установлен (объект передачи) объект сообщения может принять только сообщение удаленного запроса. Если бит DIR сброшен (объект приема), объект сообщения может принять только сообщение данных;

- если бит MIDE установлен, то бит IDE получаемого сообщения оказывает следующее влияние:

- если бит IDE (регистр MOARn) установлен, то бит IDE принимаемого сообщения должен быть равен единице (расширенный идентификатор);

- если бит IDE сброшен, бит IDE принимаемого сообщения должен быть равен нулю (стандартный идентификатор);

- если бит MIDE сброшен, значение бита IDE принимаемого сообщения не важно, т.е. допускаются сообщения как со стандартным, так и с расширенным идентификатором;

- идентификатор полученного сообщения полностью (побитно) совпадает с идентификатором, хранящимся в регистре MOARn объекта сообщений, за исключением битов, закрытых маской регистра MOAMRn, значение которых не важно. На рисунке 17.22 показан пример проверки идентификатора.

Среди всех объектов сообщений, которые отвечают указанным выше критериям, для сохранения полученного сообщения выбирается объект с наивысшим приоритетом. Для задания приоритета используется поле PRI в регистре MOARn. Объект сообщения, у которого значение поля PRI меньше, имеет больший приоритет. При равенстве значений поля PRI приоритетным считается объект сообщения, который предшествует следующему в списке.

Фильтрация при передаче сообщений

Когда требуется передача содержимого какого-либо объекта сообщения, в соответствующих управляющих регистрах выставляются флаги, указывающие на необходимость передачи. Объект сообщения считается корректным для передачи, если одновременно соблюдаются условия:

- объект сообщения распределен в список объектов сообщений CAN узла;

- флаг MSGVAL установлен;

- флаг TXRQ установлен;

- флаги TXEN0 и TXEN1 установлены.

Может возникнуть ситуация, когда передачи требуют одновременно несколько объектов сообщений. Среди всех объектов, которые отвечают указанным выше критериям, для передачи выбирается объект с наивысшим приоритетом.

Объект сообщения, у которого значение поля PRI (регистр MOARn) меньше, имеет больший приоритет. При равенстве значений поля PRI разных объектов приоритет определяется следующим образом:

- при PRI = 10b – согласно правилам арбитража передачи сообщения;

- при PRI = 01b/11b приоритет имеет объект сообщения, который предшествует следующему в списке.

Объект сообщения, являющийся корректным для передачи и имеющий приоритет, будет осуществлять передачу первым. Остальные объекты сообщений будут переданы по очереди, согласно их приоритетам.

Объект сообщения определяется как стандартный объект сообщения, если в регистре MOFCRn значение битового поля MMC равно нулю. Стандартный объект сообщения может принимать и передавать сообщения, согласно правилам, описанным выше.

На рисунке 17.23 показано формирование запроса на передачу объекта сообщения.

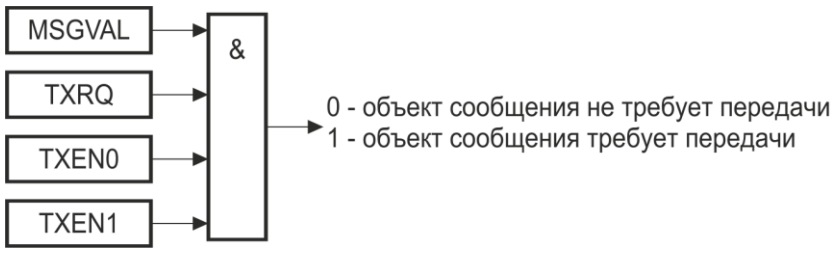


Рисунок 17.23 – Формирование запроса на передачу объекта сообщения

17.8 FIFO структура объектов сообщений

Регистр MOFGPRn объекта сообщения n содержит установки указателей на объекты сообщений, которые используются при операциях FIFO и шлюзовых операциях.

В случае сильной загрузки ЦП обработка серии сообщений может быть затруднена – например, 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 может функционировать автоматически и позволяет избежать потери принимаемых сообщений, минимизировать время подготовки сообщений к отправке, а также генерировать прерывания по окончании операций.

Допускается организация нескольких параллельных FIFO структур. Число структур и их составляющих зависит только от количества доступных объектов сообщений. FIFO структура может быть создана, изменена и удалена в любой момент времени, даже во время операций контроллера CAN.

На рисунке 17.24 представлена основная FIFO структура. Она состоит из одного базового объекта и n-ого числа вспомогательных объектов. Вспомогательные объекты объединяются последовательно в списки (подобно спискам объектов сообщений). Базовый объект может быть занесен в любой список. Хотя на рисунке базовый объект не относится ни к одному из списков, он может быть вставлен в любую последовательность вспомогательных объектов. Это означает, что базовый объект одновременно является и вспомогательным объектом (шлюзовые операции не возможны). Порядковые номера объектов сообщений (0, 1, 2 и т. д.) не имеют никакого значения при FIFO операциях с объектами.

Базовый объект не нуждается в обязательном занесении его в какой-либо список, в отличие от вспомогательных объектов, 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 базовый объект к вспомогательным объектам, независимо от того, принадлежат базовый и вспомогательный объекты одному списку или разным спискам.

Минимальная FIFO структура может состоять из одного объекта сообщения, который будет одновременно являться и базовым, и вспомогательным (фактически не используется). Максимальная FIFO структура может включать в себя все 64 объекта сообщений.

В базовом объекте FIFO границы установлены: поле BOT указывает на самый младший элемент FIFO структуры, поле TOP – на самый старший элемент, поле CUR – на вспомогательный объект, который в настоящий момент выбран котроллером CAN для передачи сообщения. Как только начинается передача, в CUR записывается номер следующего по списку вспомогательного объекта сообщения (CUR = PNEXT используемого объекта). Если значение битового поля CUR достигло номера старшего элемента списка (CUR = TOP), то следующим значением будет 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.

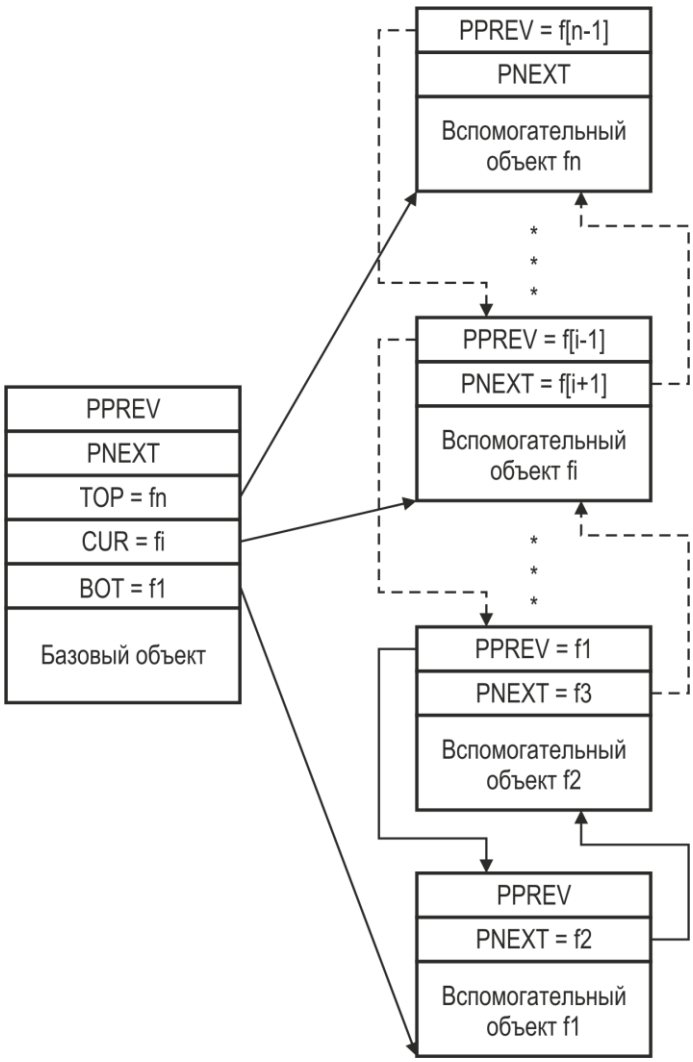


Рисунок 17.24 – FIFO структура с базовым объектом и n вспомогательными объектами

Битовое поле SEL позволяет определить вспомогательный объект в пределах списка, для которого генерируется прерывание всякий раз, когда указатель CUR достигает значения указателя SEL. Также битовое поле SEL позволяет отследить окончание запланированной передачи серии сообщений или выдать прерывание, предупреждающее о том, что FIFO структура становится заполненной.

FIFO структура для приема

FIFO структура для приема используется для буферизации входящих сообщений данных и удаленных запросов.

FIFO структура для приема активируется записью значения 0001b в битовое поле MMC регистра MOFCRn базового объекта. Эта запись автоматически определяет объект как базовый объект приема FIFO. Типы вспомогательных объектов FIFO не имеют значения при операциях.

Когда базовый объект FIFO получает сообщение от узла CAN, которому он принадлежит, сообщение сохраняется не в этом базовом объекте, а во вспомогательном объекте сообщения, на который указывает битовое поле CUR. При этом по умолчанию предполагается, что для вспомогательного объекта MMC = 0000b (действительное значение MMC игнорируется), и никаких операций фильтрации принимаемого сообщения не производится.

Одновременно с приемом сообщения текущее значение указателя CUR базового объекта меняется на номер следующего по списку вспомогательного объекта FIFO

структуры. Этот вспомогательный объект будет использован для приема следующего сообщения.

Если установлен флаг OVIE регистра MOFCRn базового объекта и значение указателя CUR становится равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем TXINP базового объекта сразу после сохранения полученного сообщения во вспомогательном объекте. Прерывания генерируются, если это разрешено битом TXIE.

Следует помнить, что сообщение сохраняется в базовом и вспомогательном объектах FIFO, только если установлен бит MSGVAL.

Во избежание непосредственного приема сообщения вспомогательным объектом, как если бы он был независимым объектом и не принадлежал FIFO структуре, флаги RXEN всех вспомогательных объектов должны быть сброшены. Состояние флага RXEN не важно в случае, когда вспомогательный объект занесен в список, не связанный с узлом CAN.

FIFO структура для передачи

FIFO структура для передачи используется для буферизации серий сообщений данных или удаленных запросов, которые должны быть отправлены. FIFO структура для передачи состоит из базового объекта и одного или более вспомогательных объектов.

FIFO структура для передачи активируется записью значения 0010b в поле MMC регистра MOFCRn базового объекта. В отличие от FIFO структуры для приема, в битовые поля MMC вспомогательных объектов (FIFO структуры для передачи) должно быть записано значение 0011b. Указатели CUR всех вспомогательных объектов должны указывать на базовый объект FIFO передачи (чтобы инициализироваться программно).

Флаги TXEN1 всех вспомогательных объектов сообщений, за исключением одного, на который указывает указатель CUR базового объекта, должны быть программно сброшены. Флаг TXEN1 указанного объекта должен быть установлен. Указатель CUR базового объекта может быть инициализирован для любого вспомогательного объекта.

При определении корректности объектов сообщений FIFO структуры для начала FIFO-операций базовый объект должен быть определен первым как корректный, т.е. MSGVAL должен быть установлен.

В случае необходимости удаления FIFO структуры, прежде, чем начнется операция удаления, все вспомогательные объекты, принадлежащие этой FIFO структуре, должны быть определены как некорректные (биты MSGVAL должны быть сброшены).

FIFO структура для передачи использует флаги TXEN1 всех своих объектов для выбора сообщения для передачи. В результате фильтрации право передавать сообщение получает тот объект, у которого выставлен флаг TXEN1. После передачи сообщения флаг TXEN1 аппаратно сбрасывается, а в указатель CUR записывается номер следующего объекта, требующего отправки сообщения, для которого уже выставлен (аппаратно) свой флаг TXEN1, и так далее для всей FIFO структуры.

Если установлен флаг OVIE регистра MOFCRn базового объекта и значение указателя CUR становится равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем RXINP базового объекта после завершения операций получения сообщения. Прерывания приема базового объекта генерируются, если это разрешено битом RXIE.

Рекомендации по программированию регистров для FIFO структуры

1 Для передающего базового объекта:

- сбросить бит MSGVAL;
- задать поля CUR, BOT, TOP, SEL;
- записать значение 0010b в поле MMC, задать DLC, установить биты OVIE и RXIE (если необходимо).

Примечание – Состояние регистров MOARn и MOAMRn передающего базового объекта не важно, поскольку в передаче участвуют передающие вспомогательные

объекты и принимающий базовый объект. Поле RXINP указывает линию, на которую будет выдаваться прерывание переполнения (CUR = SEL).

2 Для передающих вспомогательных объектов:

- сбросить бит MSGVAL;

- установить биты DIR, TXEN1 (только для того вспомогательного объекта, на который указывает поле CUR передающего базового объекта, у остальных вспомогательных объектов бит TXEN1 должен быть сброшен), TXEN0;

- записать в поле CUR номер передающего базового объекта;

- записать значение 0011b в поле MMC, задать DLC.

Примечание – Значения регистров MOARn передающих вспомогательных объектов должно совпадать со значением регистра MOAR принимающего базового объекта, так как процесс передачи фактически происходит между ними (или иного принимающего объекта, если на приеме используется не FIFO структура).

3 Для принимающего базового объекта:

- установить бит RXEN;

- задать поля CUR, BOT, TOP, SEL;

- записать значение 0001b в поле MMC, задать DLC, установить биты OVIE и TXIE (если необходимо).

Примечание – Значение регистра MOARn принимающего базового объекта должно быть равно значению регистров MOARn передающих вспомогательных объектов передачи. Поле TXINP указывает, на какую линию будет выдаваться прерывание переполнения (прерывание после операции сохранения полученного сообщения во вспомогательных объектах при CUR = SEL).

4 Для принимающих вспомогательных объектов:

- сбросить бит RXEN (не требуется, если вспомогательные объекты занесены в список, не связанный с узлом CAN);

- задать поле DLC (состояние поля MMC не важно).

Примечание – Состояние регистров MOARn принимающих вспомогательных объектов не важно.

5 Установить бит MSGVAL в первую очередь у передающего базового объекта, а затем у всех остальных объектов.

6 Установить бит TXRQ для всех передающих вспомогательных объектов, начиная с того, на который указывает поле CUR передающего базового объекта.

17.9 Шлюзы

Режим позволяет реализовывать автоматическую передачу информации через шлюз между двумя независимыми шинами CAN без участия ЦП.

Шлюз можно сформировать на уровне объектов сообщений и осуществлять передачу информации между узлами CAN. Шлюз может быть сформирован между двумя любыми объектами сообщений, принадлежащими разным узлам CAN. Количество шлюзов зависит только от количества объектов сообщений, допускающих формирование шлюзов.

Режим шлюза активируется записью значения 0100b в битовое поле MMC регистра MOFCRn объекта сообщения n и инициализирует его как шлюзовый объект-источник. Объект сообщения, который будет являться шлюзовым объектом-приемником, выбирается указателем CUR объекта-источника. Для формирования шлюза достаточно, чтобы объект-приемник был корректным (установлен бит MSGVAL). Остальные параметры не влияют на возможность осуществления передачи между объектами от источника к приемнику.

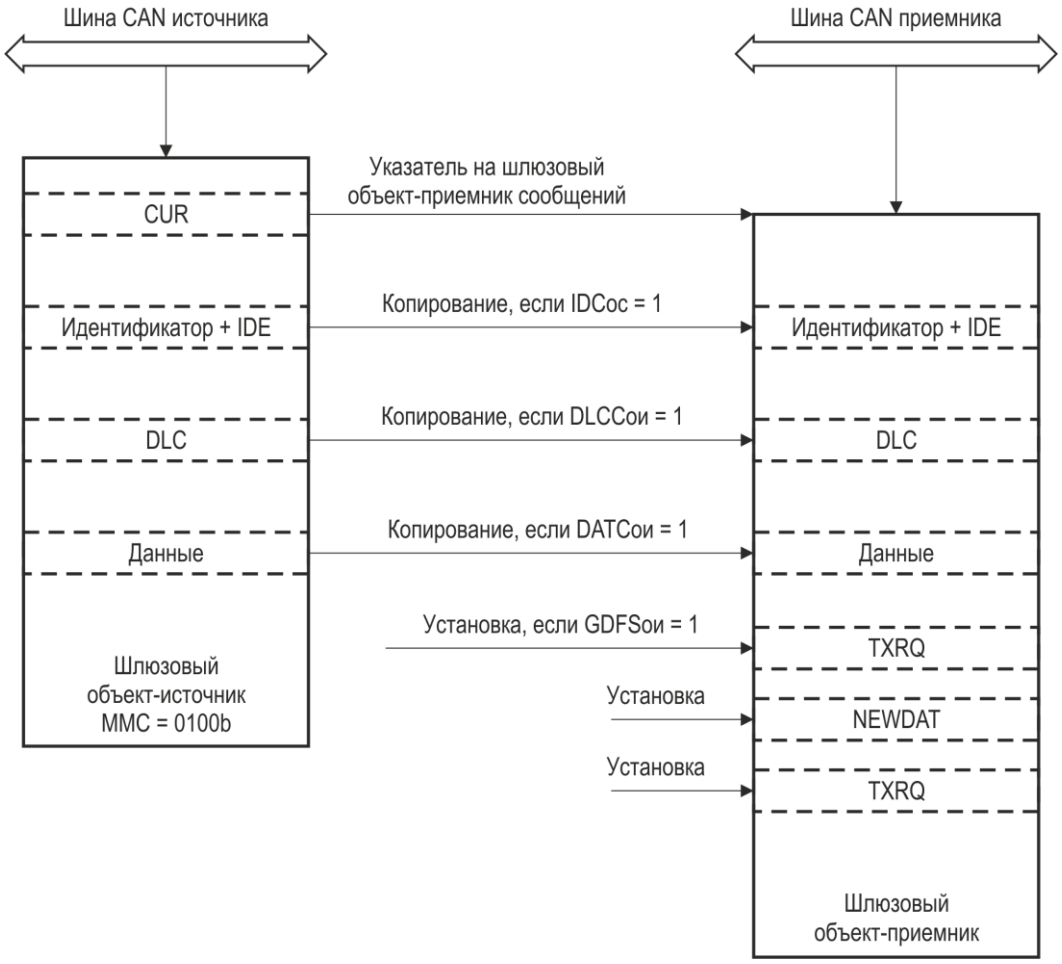


Рисунок 17.25 – Передача через шлюз от источника к приемнику

Шлюзовый объект-источник (см. рисунок 17.25) функционирует как обычный объект сообщений с тем отличием, что возможны дополнительные действия контроллера CAN при приеме и сохранении сообщения в объекте-приемнике:

1 Если установлен флаг DLCC регистра MOFCRn объекта-источника, код длины данных DLC копируется из шлюзового объекта-источника в шлюзовый объект-приемник.

2 Если установлен флаг IDC объекта-источника, идентификатор ID и расширение IDE копируются из шлюзового объекта-источника в шлюзовый объект-приемник.

3 Если установлен флаг DATC объекта-источника, байты данных, хранящиеся в двух регистрах MODATALn и MODATAHn объекта-источника, копируются из шлюзового объекта-источника в шлюзовый объект-приемник. Копируются все 8 байт данных, вне зависимости от значения поля DLC.

4 Если установлен флаг GDFS объекта-источника, то устанавливается бит запроса передачи TXRQ объекта-приемника.

5 Устанавливаются флаги RXPND и NEWDAT регистра MOSTATn объекта-приемника.

6 Если установлен флаг RXIE регистра MOSTATn объекта-приемника, то генерируется запрос на прерывание.

7 Указатель CUR объекта-источника переводится на следующий объект-приемник по правилам FIFO структуры. Сформировать шлюз между объектом-источником и одним объектом-приемником (значение указателя CUR будет оставаться неизменным) возможно программированием:

TOP = BOT = CUR = номер объекта-приемника.

Организация шлюза «объект-источник – объект-приемник» аналогична организации FIFO структуры «базовый объект-вспомогательный объект», что указывает на возможность формирования шлюза с интегрированным FIFO-приемником. На рисунке 17.25 объект сообщения слева – шлюзовый объект-источник, а объект сообщения справа – шлюзовый объект-приемник.

При получении сообщения данных (объект-источник является объектом приема, т.е. его бит DIR сброшен) и при получении удаленного запроса (объект-источник является объектом передачи) через шлюз используется один и тот же механизм.

Удаленные запросы

После получения узлом CAN сообщения удаленного запроса и сохранения его в объекте сообщения выставляется бит запроса передачи для ответа на удаленный запрос (отправка сообщения данных) или для автоматического повторения запроса. Если сброшен бит FRREN объекта сообщения, в который было сохранено сообщение удаленного запроса, то выставляется флаг TXRQ этого объекта.

У объекта сообщений, передающего сообщение удаленного запроса, должны быть выставлены биты регистра MOSTATn следующим образом: DIR = 0 (объект передает сообщение удаленного запроса), TXEN0 = 1, TXEN1 = 1, далее MSGVAL = 1, TXRQ = 1. Значение идентификатора, содержащегося в регистре MOARn передающего объекта сообщений, должно быть равно значению идентификатора (или подходить, если есть маскируемые биты идентификатора у принимающего объекта, установленные в регистре MOAMRn) принимающего объекта сообщений, чтобы сообщение удаленного запроса было принято принимающим объектом другого узла. Само сообщение удаленного запроса должно содержать идентификатор принимающего объекта сообщений, поэтому значение регистра MODATALn передающего объекта сообщений должно быть равно значению регистра MOARn принимающего объекта.

У объекта сообщений, принимающего сообщение удаленного запроса, должны быть выставлены биты регистра MOSTATn следующим образом: DIR = 1 (объект принимает сообщение удаленного запроса); TXEN0 = 1 и TXEN1 = 1 (если отвечать на запрос будет сам); RXEN = 1; далее MSGVAL = 1. Регистры объекта MODATALn и MODATAHn должны содержать данные, которые будут выдаваться в ответ на запрос.

Если установлен бит FRREN объекта сообщения, который принял сообщение удаленного запроса, то флаг TXRQ устанавливается у того объекта, на который указывает поле CUR объекта, принявшего удаленный запрос. При этом указатель CUR не меняет своего значения.

Последовательность записи в регистры объекта сообщений, передающего сообщение удаленного запроса, аналогична указанной выше.

У объекта сообщений, принимающего сообщение удаленного запроса, должны быть выставлены биты регистра MOSTATn следующим образом: DIR = 1 (объект принимает сообщение удаленного запроса), RXEN = 1, далее MSGVAL = 1. Битовое поле CUR регистра MOFGPRn должно указывать на номер объекта сообщения (должен находиться в том же узле, что и принявший сообщение удаленного запроса объект), содержащего данные, предназначенные для передачи в ответ на поступивший удаленный запрос.

У объекта сообщений, хранящего данные для отправки в ответ на запрос, должны быть установлены следующие биты регистра MOSTATn: DIR = 1 (объект передает сообщение данных), TXEN0 = 1, TXEN1 = 1, далее MSGVAL = 1. Бит TXRQ регистра MOSTATn объекта сообщения, хранящего данные для отправки в ответ на запрос, устанавливается автоматически при приеме сообщения удаленного запроса принимающим объектом сообщения.

Прием ответа на запрос (переданного сообщения данных) осуществляется стандартным объектом сообщения запрашивающего узла CAN (обмен данными происходит между объектом сообщения, хранящим данные для отправки в ответ на запрос, и объектом сообщения запрашивающего узла CAN).

Несмотря на то, что механизм удаленных запросов работает независимо от типа объекта сообщения, он наиболее полезен при использовании шлюзов, для формирования удаленных запросов на шине шлюзового объекта-источника после получения удаленного запроса на шине шлюзового объекта-приемника. В зависимости от значения бита FRREN шлюзового объекта-приемника, есть два варианта обработки удаленного запроса, возникшего с той стороны шлюза, где расположен объект-приемник (при условии, что происходит передача из объекта-источника в объект-приемник, т. е. DIR (источника) = 0 и DIR (приемника) =1).

1 Обработка запроса шлюзового объекта-приемника с FRREN = 0b:

- сообщение удаленного запроса принимается шлюзовым объектом-приемником;
- бит TXRQ шлюзового объекта-приемника устанавливается автоматически;
- сообщение данных с текущей информацией, хранящейся в объекте-приемнике, передается на шину приемника.

2 Обработка запроса шлюзового объекта-приемника с FRREN = 1b:

- сообщение удаленного запроса принимается шлюзовым объектом-приемником;
- бит TXRQ шлюзового объекта-источника (объект должен быть указан в поле CUR объекта-приемника), устанавливается автоматически;
- сообщение данных передается объектом-источником на шину CAN источника;
- получатель удаленного запроса в ответ выдает сообщение данных на шину источника;
- сообщение данных сохраняется в объекте-источнике;
- сообщение данных копируется в объект-приемник (через шлюз);
- выставляется бит TXRQ объекта-приемника (при условии, что $GDFS_{источника} = 1$);
- новые данные, сохраненные в объекте-приемнике, передаются на шину приемника, в ответ на удаленный запрос на шине приемника.

Рекомендации по записи в регистры в режиме шлюза при передаче удаленного запроса с FRREN = 1.

Обмен запрос-данные происходит в данном случае между стандартным объектом сообщений одного узла и объектом-приемником шлюза другого узла. Но при этом данные для ответа на запрос в шлюзовый объект-приемник поступают по шлюзу от объекта-источника. При получении удаленного запроса от объекта сообщения объектом-приемником флаг TXRQ устанавливается не у самого объекта-приемника, а у объекта-источника, благодаря установленному биту FRREN и битовому полю CUR (указывает на объект-источник) объекта-приемника. Данные из MODATALn и MODATAHn объекта-источника копируются в MODATALn и MODATAHn объекта-приемника (установлен бит DATC регистра MOFCRn объекта-источника), вследствие чего автоматически устанавливается бит TXRQ регистра MOCTRn объекта-приемника (установлен бит GDFS объекта-источника шлюза), и осуществляется передача сообщения данных (ответ на запрос) запрашивающему объекту сообщения.

После успешного приема/передачи сообщения ЦП получает уведомление о завершении операции для задания дальнейших действий, связанных с объектом сообщения.

17.10 Прерывания объектов сообщений

После сохранения принятого сообщения в объект сообщения или успешной передачи формируется соответствующее прерывание. Каждый объект сообщения может формировать прерывания. Каждое прерывание направляется на одну из 16 выходных линий прерываний. Прерывания приема (после сохранения сообщения) также формируются после операций FIFO и шлюзовых операций. Флаги TXPND и RXPND всегда устанавливаются после успешной операции передачи/приема, независимо от состояния соответствующих флагов разрешения прерываний.

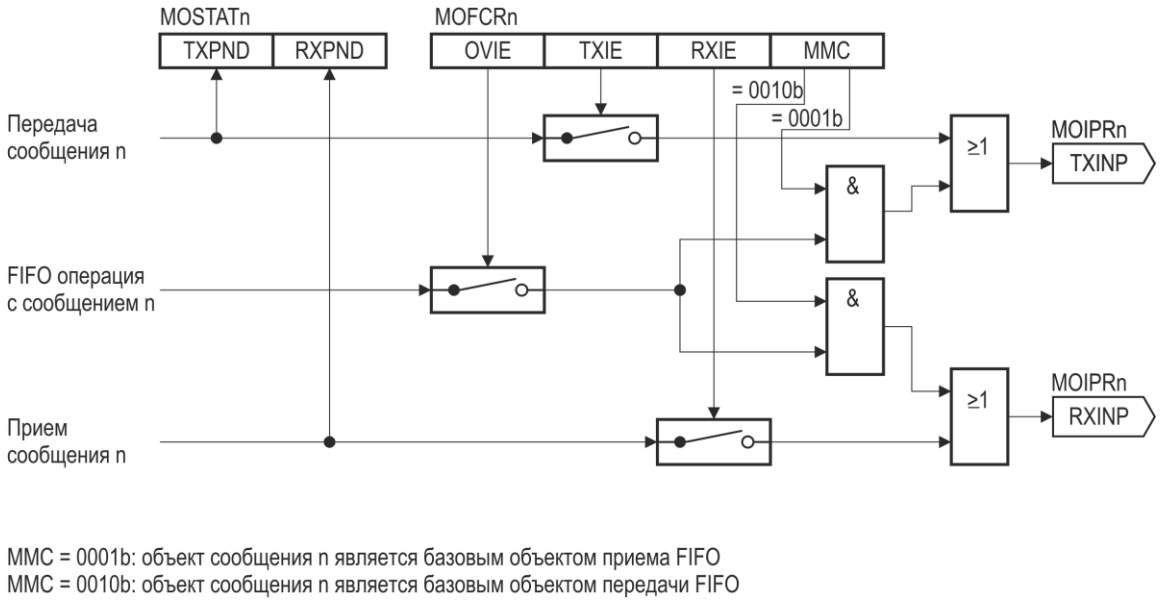


Рисунок 17.26 – Распределение прерываний

Объект сообщения может формировать FIFO прерывания. Если флаг OVIE регистра MOFCRn установлен, то формирование FIFO прерывания будет зависеть от типа объекта сообщений (см. рисунок 17.26):

- если объект сообщения является принимающим базовым объектом, то выходная линия прерываний для этого объекта определяется битовым полем TXINP регистра MOIPRn.
- если объект сообщения является передающим базовым объектом, то выходная линия прерываний определяется битовым полем RXINP.

Ждущие сообщения

Когда генерируется запрос на прерывание (после приема/передачи сообщения), в одном из четырех регистров ждущих прерываний MSPNDx (x – 0, 1, 2, 3) выставляется флаг ждущего сообщения. Четыре 32-разрядных регистра образуют область из 128 битов – по два бита (один бит для операций приема и один бит для операций передачи) для каждого из объектов сообщений. Позиция флага ждущего сообщения определяется демультиплексорами DMUX.

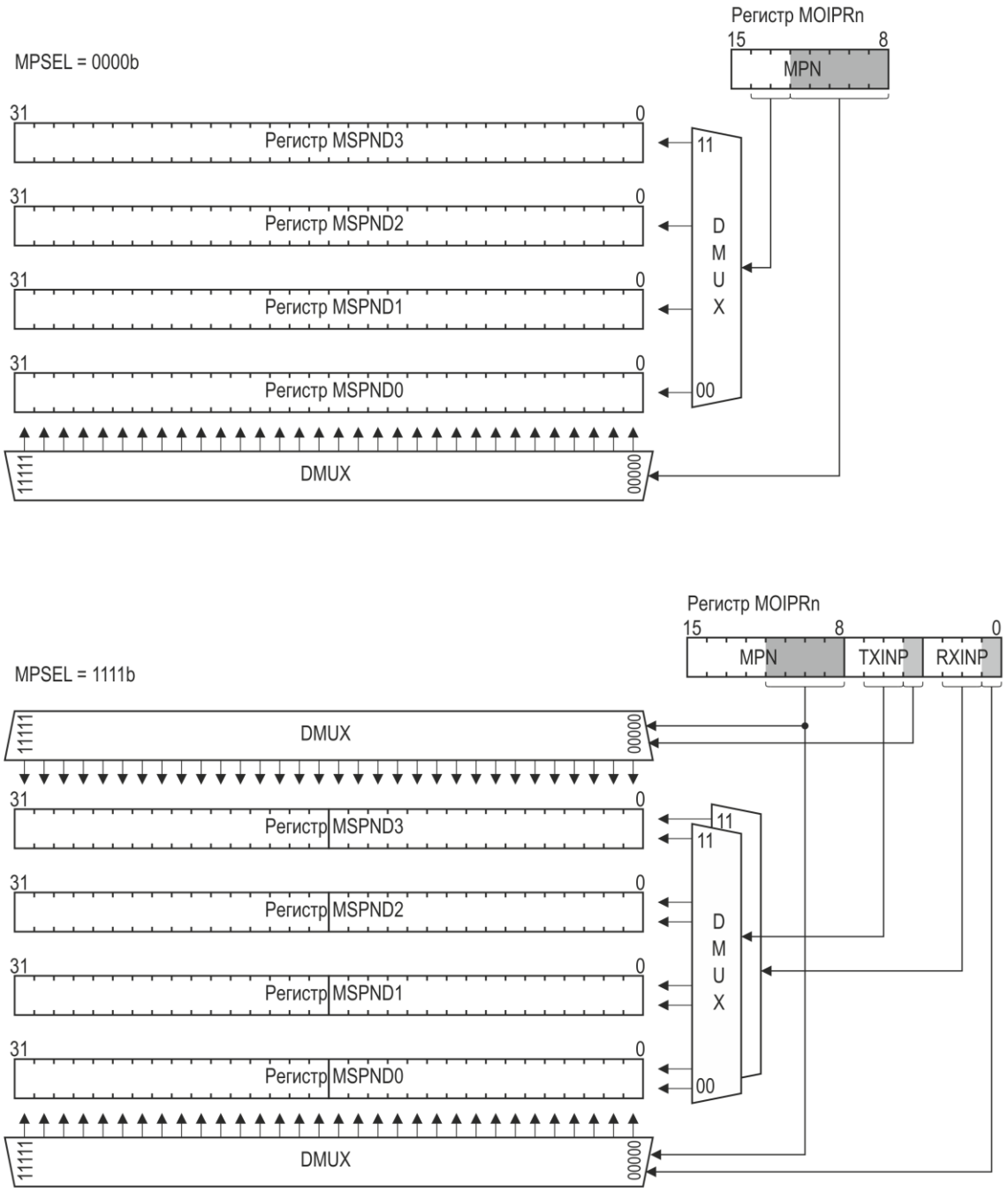


Рисунок 17.27 – Два режима выбора и установки флагов ждущих сообщений

В зависимости от значения поля MPSEL регистра MCR, реализуется один из двух режимов выбора и установки флагов ждущих сообщения (см. рисунок 17.27):

- Режим 1 в случае MPSEL = 0h;
- Режим 2 в случае MPSEL = Fh;

Если нет необходимости в определении источника прерывания (прием или передача сообщения), то можно использовать любой из двух режимов, в противном случае, следует использовать второй режим.

В первом режиме установка флага ждущего сообщения происходит следующим образом:

- 6 и 5 биты поля MPN выбирают регистр MSPNDx, в котором будет установлен флаг ждущего сообщения;
- пять младших бит поля MPN (на рисунке 17.27 выделены серым цветом) выбирают позицию флага (от 0 до 31), который будет установлен в выбранном регистре MSPNDx.

Во втором режиме при определении позиции флага ждущего сообщения принимаются в расчет значения поля MPN и полей RXINP (для приема) и TXINP (для передачи). При этом для флагов могут использоваться любые биты выбранного регистра MSPNDx. Установка флага ждущего сообщения происходит следующим образом:

- 2 и 1 биты поля TXINP/RXINP выбирают регистр MSPNDx, в котором будет установлен флаг по окончании передачи/приема сообщения;

- четыре младших бита поля MPN (на рисунке 17.27 выделены серым цветом) совместно с нулевыми битами полей TXINP и RXINP выбирают позицию флага (от 0 до 31). Фактически нулевой бит поля TXINP/RXINP выбирает старшее или младшее слово выбранного регистра MSPNDx, а четыре бита поля MPN задают позицию в выбранном слове.

Регистры MSPNDx могут быть записаны программно. Биты, в которые записываются единицы, остаются без изменений, а биты, в которые записываются нули, очищаются. Такой механизм записи позволяет избежать конфликта между одновременной аппаратной установкой и программной очисткой битов регистра.

Каждый регистр MSPNDx связан с соответствующим регистром индекса сообщения MSIDx, который отражает позицию самого младшего бита из всех установленных в регистре MSPNDx. Регистры MSIDx доступны только для чтения и обновляются незамедлительно после изменения (как аппаратного, так и программного) содержимого соответствующих регистров MSPNDx.

Регистр маски индекса сообщения MSIMASK содержит маску для регистров MSPNDx. Только незакрытые маской биты могут обслуживаться. Регистр MSIMASK используется одновременно для всех регистров MSPNDx и соответствующих им регистров MSIDx.

17.11 Рекомендации по программированию модуля CAN

Для корректного запуска и работы контроллера и конфигурирования его узлов следует соблюдать порядок программирования регистров:

- записать регистр CLC;
- записать регистр FDR;
- в регистре NCRx установить биты INIT и CCE, после чего регистры NBTRx и NPCRx станут доступны для записи и чтения, а регистр NECNTx – только для чтения;
- записать регистр NPCRx;
- записать регистр NIPRx;
- записать регистр NBTRx;
- записать регистр NFCRx (если необходимо);
- в регистре NCRx сбросить биты INIT и CCE, после чего регистры NBTRx и NPCRx будут не доступны для записи.

Для корректной работы объектов общений регистры каждого из них в обязательном порядке должны быть проинициализированы. Исключение составляют объекты сообщений, использование которых не предусматривается. Для таких объектов в регистре MOCTRn бит MSGVAL должен оставаться сброшенным.

Обязательный порядок инициализации регистров объекта сообщений n:

- установить бит DIR в регистре MOSTATn для передачи сообщения данных/приема удаленного запроса или сбросить бит DIR для приема сообщения данных/передачи удаленного запроса; установить биты TXEN0 и TXEN1 (для передачи) или RXEN (для приема) в регистре MOCTRn;
- записать регистр MOFCRn;
- записать регистр MOARn;
- записать регистр MOAMRn (если необходимо);
- записать регистр MOFGPRn (если будут использоваться FIFO структуры);
- записать регистр MOIPRn;

- записать регистры MODATALn и MODATAHn;
- распределить объекты сообщений в списки с помощью регистра PANCTR;
- установить бит MSGVAL корректности объекта сообщения в регистре MOCTRn (для неиспользуемых объектов этот бит должен быть сброшен);
- для активирования передачи установить бит TXRQ регистра MOCTRn.

Примечание – Нарушение последовательности программирования регистров перед началом работы может привести к некорректной работе узлов или невозможности функционирования контроллера CAN в целом.

18 Блоки захвата/сравнения (CAPCOM1 и CAPCOM2)

Модуль захвата/сравнения (в дальнейшем – модуль CAPCOM) предназначен для отслеживания заданных внешних (по отношению к микроконтроллеру) и внутренних событий, а также для формирования программируемых последовательностей импульсов (в том числе и ШИМ).

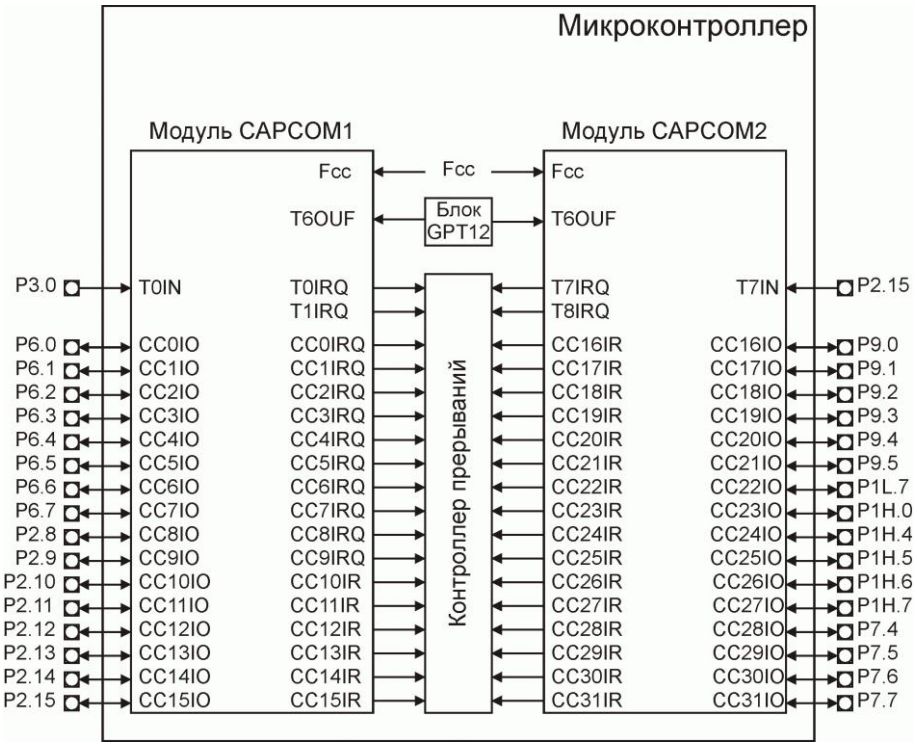


Рисунок 18.1 – Интерфейс модулей CAPCOM1 и CAPCOM2

В микроконтроллере присутствуют два идентичных (с функциональной точки зрения) модуля CAPCOM – CAPCOM1 и CAPCOM2 – отличающиеся только выводами микроконтроллера, к которым они подключены (см. таблицу 18.1 и рисунок 18.1), и регистрами (рисунок 18.2). Интерфейс модулей представлен на рисунке 18.1, а структурные схемы обоих модулей – на рисунке 18.2.

Далее приводится полное описание модуля CAPCOM1 и его режимов работы.

При работе с модулем CAPCOM2 следует названия таймеров/счетчиков, регистров и линий прерываний модуля CAPCOM1 заменять на соответствующие им в модуле CAPCOM2. Для уточнения можно обратиться к таблице 18.1 и рисунку 18.2.

Таблица 18.1 – Идентичные таймеры, регистры и линии прерываний модулей CAPCOM1 и CAPCOM2

Модуль CAPCOM1	Модуль CAPCOM2
Регистры таймеров/счетчиков	
T0	T7
T1	T8
Регистры управления	
CC1_T01CON	CC2_T78CON
CC1_DRM	CC2_DRM
CC1_M0	CC2_M4
CC1_M1	CC2_M5

Окончание таблицы 18.1

CC1_M2	CC2_M6
CC1_M3	CC2_M7
CC1_OUT	CC2_OUT
CC1_SEM	CC2_SEM
CC1_SEE	CC2_SEE
CC1_IOC	CC2_IOC
Регистры захвата/сравнения каналов модуля	
CC0 – CC15	CC16 – CC31
Линии прерываний	
CC0IRQ – CC15IRQ	CC16IRQ – CC31IRQ

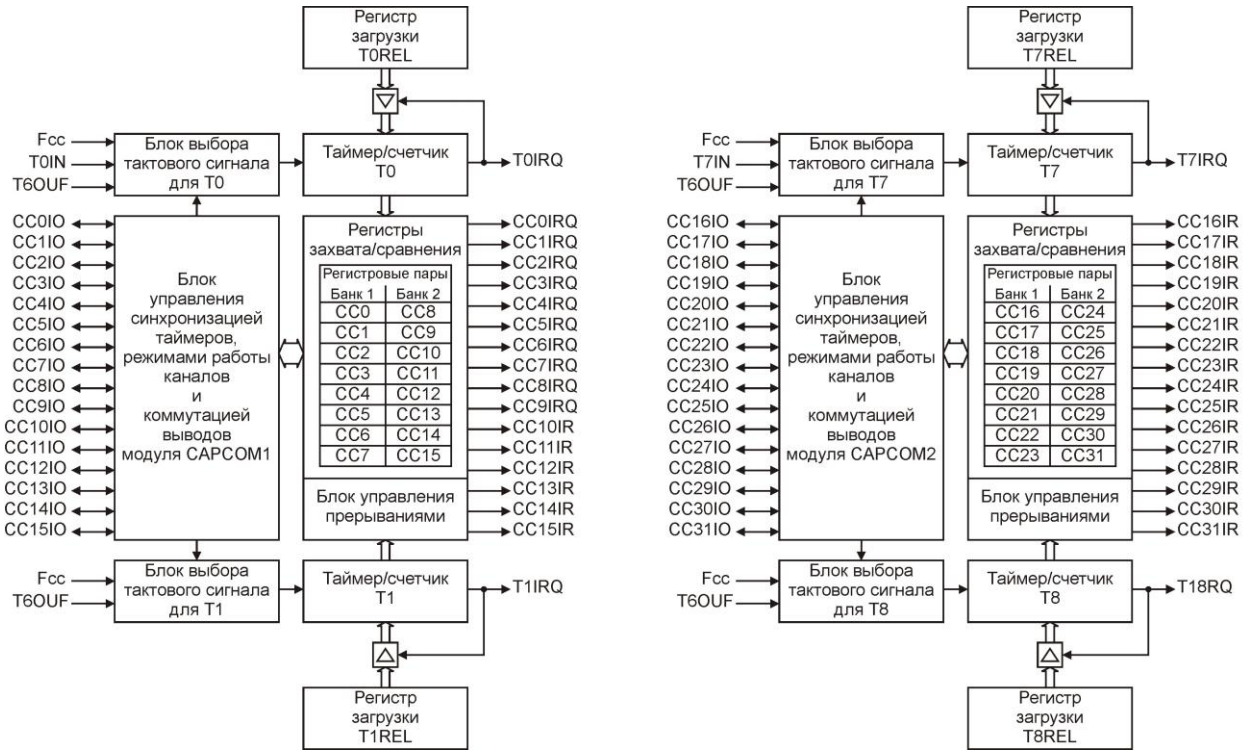


Рисунок 18.2 – Структурные схемы модулей CAPCOM1 и CAPCOM2

Структура модуля CAPCOM1

Модуль состоит из двух таймеров/счетчиков T0 и T1, регистров загрузки T0REL и T1REL, 16 регистров захвата/сравнения CC0 – CC15, а также двух блоков выбора тактовых сигналов для таймеров/счетчиков. Блоком, контролирующим всю работу модуля, является блок управления синхронизацией таймеров/счетчиков, режимами работы каналов и коммутацией выводов. Регистры захвата/сравнения логически разбиты на два банка регистров и образуют регистровые пары, что позволяет использовать каждый из 16 каналов модуля как независимо от других, так и попарно. Также в состав модуля входит блок управления прерываниями.

18.1 Таймер/счетчик T0

Таймером/счетчиком T0 является 16-разрядный регистр, который инкрементируется каждый такт сигнала тактирования. Таймер/счетчик запускается установкой бита T0R регистра CC1_T01CON. Программно, в таймер/счетчик может быть записано любое 16-разрядное значение. Таймер/счетчик всегда доступен для записи. Когда таймер/счетчик переполняется, в него загружается значение из регистра T0REL и, если разрешено,

формируется запрос на прерывание от T0, который по линии прерывания T0IRQ передается на контроллер прерываний.

На рисунке 18.3 показано положение T0 в структуре модуля CAPCOM1.

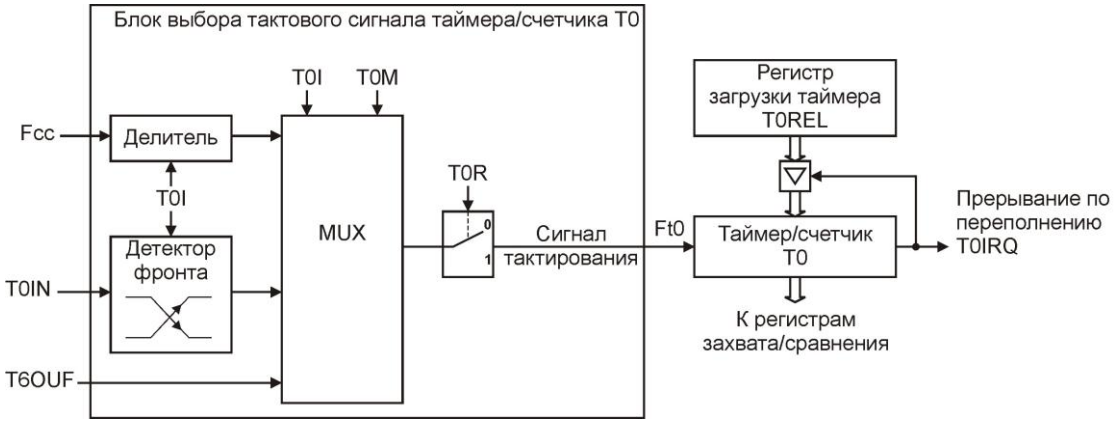


Рисунок 18.3 – Таймер/счетчик T0 и его схема тактирования

Примечание – Остановка таймера/счетчика T0 сбросом бита T0R не приводит к обнулению его содержимого. После запуска таймер/счетчик будет продолжать счет с того значения, на котором был остановлен.

Таймер/счетчик T0 может работать в двух режимах:

- таймера для расчета временных интервалов между событиями формирования последовательностей импульсов (в том числе и ШИМ);
- счетчика для подсчета количества возникающих запрограммированных событий.

В режиме таймера (бит T0M сброшен), T0 тактируется сигналом Ft0, который формируется на основе сигнала внешней тактовой частоты Fcc (fcc = fper). В зависимости от заданного в поле T0I коэффициента деления, частота переключения таймера T0 будет находиться:

- в ступенчатом режиме в диапазоне от fcc/1024 до fcc/8;
- в нормальном режиме в диапазоне от fcc/128 до fcc.

В режиме счетчика (бит T0M установлен), T0 инкрементируется каждый раз, когда возникает запрограммированное событие. Для корректной работы счетчика, сигнал тактирования, приходящий от выбранного источника, должен иметь частоту не ниже fcc, а желательно fcc/2. Если ведется подсчет фронтов сигнала, то минимальное время удержания уровня сигнала в новом состоянии от момента появления фронта до момента появления следующего фронта должно быть не менее длительности одного такта Fcc, а для более стабильной работы – двух тактов Fcc.

После переполнения таймер/счетчик T0 загружается значением из регистра T0REL. Если значение регистра T0REL не меняется, то можно рассчитать период переполнения таймера/счетчика Pt0 по формулам:

- для ступенчатого режима:

$$Pt0 = \frac{(2^{16} - \langle T0REL \rangle) \times 2^{(\langle T0I \rangle + 3)}}{fcc} \quad (18.1)$$

- для нормального режима:

$$Pt0 = \frac{(2^{16} - \langle T0REL \rangle) \times 2^{\langle T0I \rangle}}{fcc} \quad (18.2)$$

18.2 Таймер/счетчик T1

Таймером/счетчиком T1 является 16-разрядный регистр. Функционально идентичен таймеру/счетчику T0, за исключением режима счетчика.

В режиме счетчика T1 может тактироваться только сигналом переполнения таймера T6 блока GPT. Поэтому в поле указания источника T1I следует записывать значение 000b.

На рисунке 18.4 показано положение T1 в структуре модуля CAPCOM1.

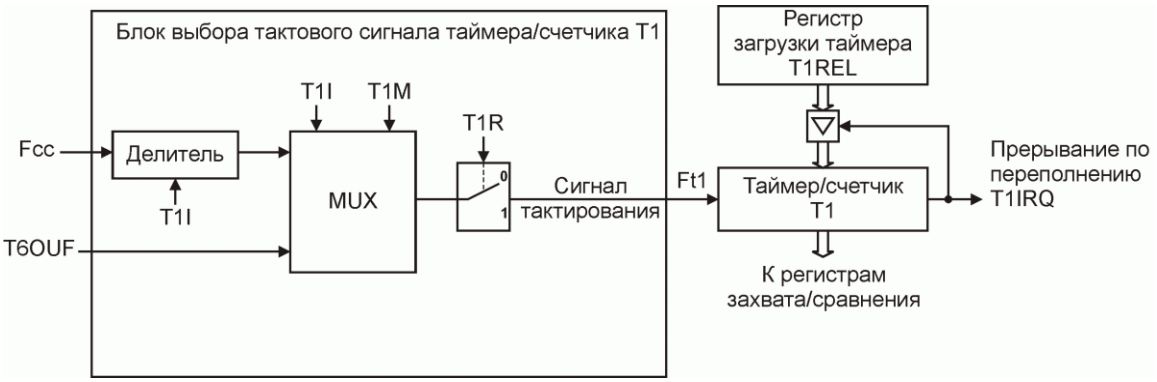


Рисунок 18.4 – Таймер/счетчик T0 и его схема тактирования

Регистры загрузки таймеров T0REL и T1REL

Регистр T0REL является 16-разрядным буфером для хранения значения, которое загружается в таймер/счетчик T0, каждый раз при его переполнении.

Регистр T1REL является 16-разрядным буфером для хранения значения, которое загружается в таймер/счетчик T1, каждый раз при его переполнении.

Оба регистра программно всегда доступны и позволяют легко и точно управлять периодами переполнения таймеров.

18.3 Каналы модуля CACPCOM1 и их режимы работы

Модуль CAPCOM имеет 16 каналов. Каждому каналу соответствует один регистр захвата/сравнения, который может взаимодействовать с любым из двух таймеров/счетчиков модуля CAPCOM1, и один вывод микроконтроллера, который в зависимости от выбранного режима должен быть программно сконфигурирован как вход или как выход.

Здесь будет приведено описание нулевого канала с регистром CC0. Поскольку все каналы функционально идентичны, то представленное описание применимо ко всем остальным каналам модуля. Исключение составляет 31-й канал, который дополнительно может использоваться для активации канала АЦП (при условии, что АЦП включен) в любом режиме работы.

Функциональная схема нулевого канала представлена на рисунке 18.5.

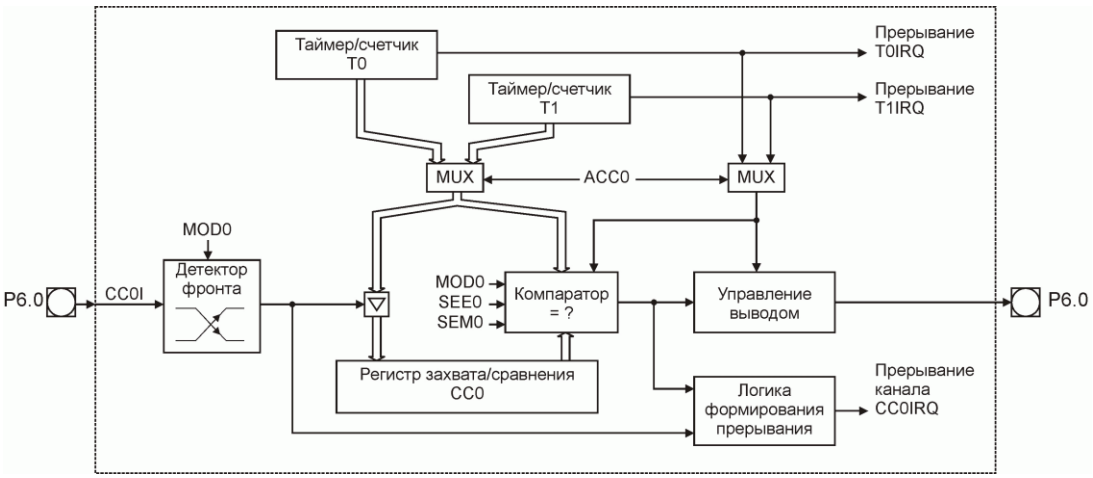


Рисунок 18.5 – Функциональная схема нулевого канала

Регистр захвата/сравнения CC0 нулевого канала является 16-разрядным регистром, программно доступным всегда. Через логику загрузки и компаратор он соединен с таймерами/счетчиками T0 и T1.

Битом ACC0 регистра CC1_M0 можно выбрать таймер/счетчик для регистра CC0, а битовым полем MOD0 – режим работы нулевого канала. Доступны три режима захвата и четыре режима сравнения.

Режимы захвата нулевого канала

В режиме захвата логика управления с помощью детектора фронта входного сигнала отслеживает события на выводе CC0IO. Во всех режимах захвата вывод CC0IO должен быть сконфигурирован как вход.

Событием является появление положительного или/и отрицательного перепада уровня сигнала, т.е., соответственно, переднего или/и заднего фронта сигнала на входе CC0IO. Минимальное время удержания уровня сигнала в новом состоянии от момента появления фронта до момента появления следующего фронта должно быть не менее длительности одного такта Fcc.

Как только запрограммированное событие обнаруживается, происходит захват (копирование) содержимого выбранного таймер/счетчика в регистр CC0, генерируется запрос на прерывание на линии CC0IRQ и далее, если разрешено (установлен бит IEN в регистре PSW и бит CC0IE в регистре CC0IC), запрос на прерывание передается в контроллер прерываний.

Если вход CC0IO используется как дополнительный внешний вход прерывания, то при возникновении события генерируется только запрос на прерывание, а захвата содержимого таймера/счетчика не происходит.

Для управления прерываниями нулевого канала используется регистр CC0IC. Все регистры управления прерываниями каналов идентичны.

Режимы сравнения нулевого канала

В режиме сравнения логика управления с помощью компаратора отслеживает момент совпадения запрограммированного значения регистра CC0 со значением выбранного таймера/счетчика. Событием является совпадение их значений.

Компаратор срабатывает только в момент инкрементирования выбранного таймера/счетчика, чтобы предотвратить повторные совпадения, если последний выключен. Программная запись в таймер/счетчик значения, совпадающего со значением регистра CC0, не вызовет срабатывания компаратора.

Как только запрограммированное событие обнаруживается, генерируется запрос на прерывание на линии CC0IRQ и далее, если разрешено (установлены бит IEN в регистре PSW и бит CC0IE в регистре CC0IC), запрос на прерывание передается в контроллер прерываний.

В зависимости от выбранного режима сравнения, вывод CC0IO используется как выход, управляемый логикой модуля CAPCOM1 или как вывод общего назначения микроконтроллера.

Режим сравнения 0

При возникновении события, генерируется запрос на прерывание на линии CC0IRQ. Модуль CAPCOM1 не оказывает влияния на вывод CC0IO и поэтому он может использоваться как вывод общего назначения.

За период переполнения выбранного таймера/счетчика может возникать более одного события, если в регистр CC0 программно записывать разные значения, каждое из которых заведомо больше текущего значения таймера/счетчика.

На рисунке 18.6 показан пример функционирования модуля CAPCOM1 в режиме сравнения 0.

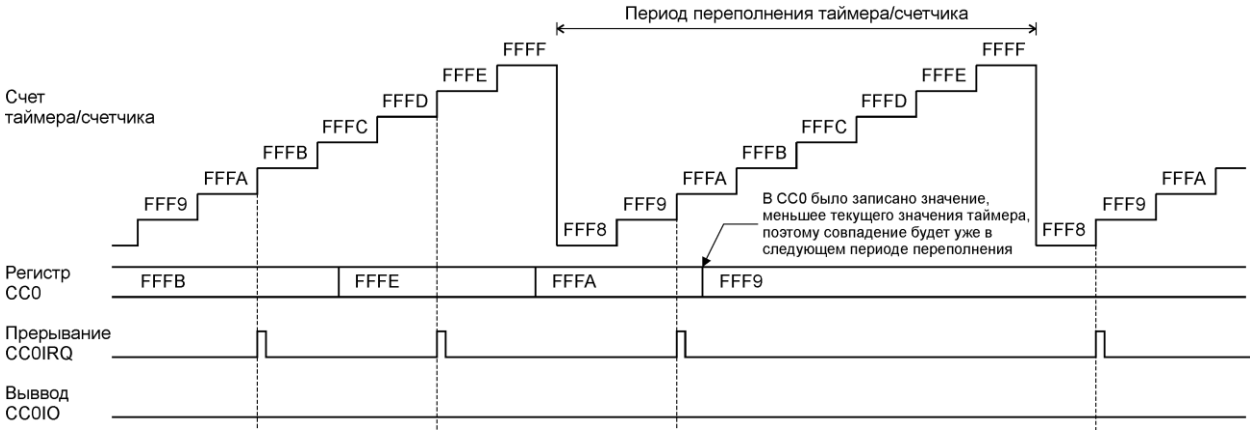


Рисунок 18.6 – Режим сравнения 0, значение регистра загрузки – FFF8h

Режим сравнения 1

При возникновении события генерируется запрос на прерывание на линии CC0IRQ, состояние сигнала на выводе CC0IO (функционирует как выход) инвертируется.

За период переполнения выбранного таймера/счетчика может возникать более одного события, если в регистр CC0 программно записывать разные значения, каждое из которых заведомо больше текущего значения таймера/счетчика.

На рисунке 18.7 показан пример функционирования модуля CAPCOM1 в режиме сравнения 1.

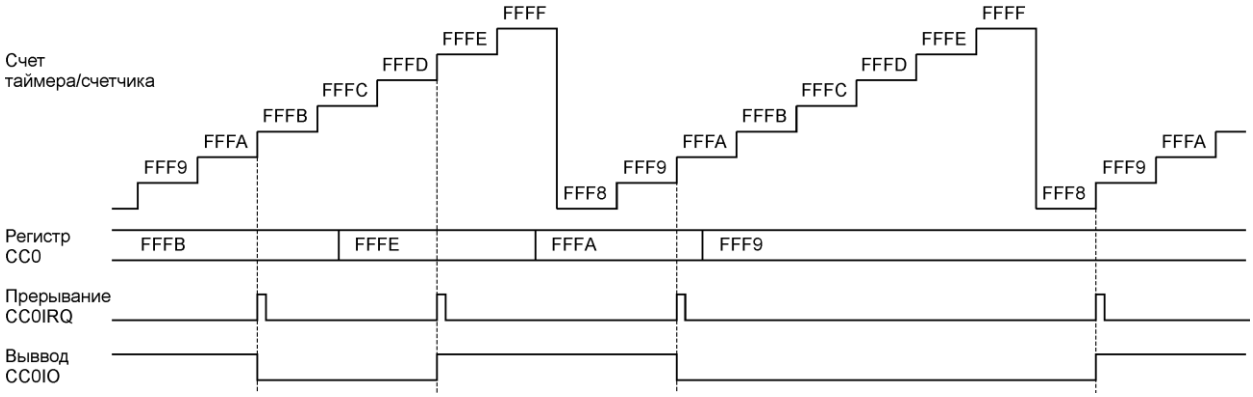


Рисунок 18.7 – Режим сравнения 1, значение регистра загрузки – FFF8h

Режим сравнения 2

При возникновении события, генерируется запрос на прерывание на линии CC0IRQ. Модуль CAPCOM1 не оказывает влияния на вывод CC0IO и поэтому он может использоваться как вывод общего назначения.

За период переполнения выбранного таймера/счетчика может возникать только одно событие. После этого все последующие совпадения (если они будут) игнорируются до начала следующего периода переполнения.

На рисунке 18.8 показан пример функционирования модуля CAPCOM1 в режиме сравнения 2.

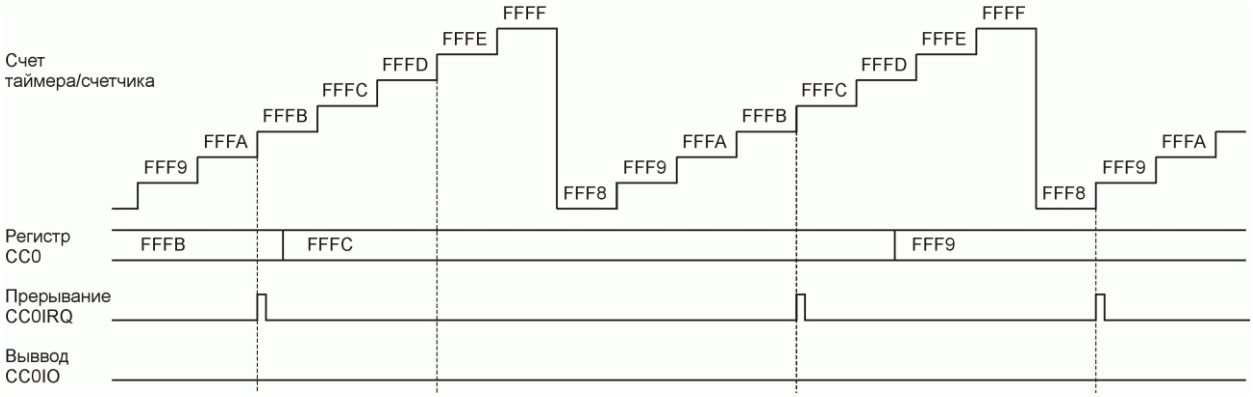


Рисунок 18.8 – Режим сравнения 2, значение регистра загрузки – FFF8h

Режим сравнения 3

При возникновении события генерируется запрос на прерывание на линии CC0IRQ, на выводе CC0IO (функционирует как выход) устанавливается логическая единица.

За период переполнения выбранного таймера/счетчика может возникать только одно событие. После этого все последующие совпадения (если они будут) игнорируются до начала следующего периода переполнения.

В момент начала очередного периода таймера/счетчика на выводе CC0IO устанавливается логический ноль.

Если значение регистра CC0 совпадает со значением, которое загружается в таймер/счетчик при переполнении, то состояние сигнала на выводе CC0IO не изменяется. В то же время это считается событием. Поэтому генерируется прерывание, и все дальнейшие события в пределах текущего периода переполнения игнорируются.

На рисунке 18.9 показан пример функционирования модуля CAPCOM1 в режиме сравнения 3.

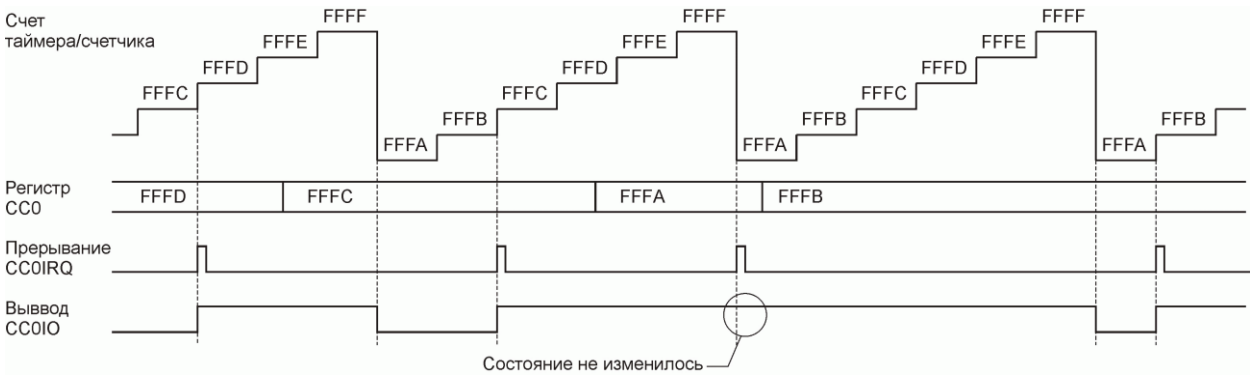


Рисунок 18.9 – Режим сравнения 3, значение регистра загрузки – FFFAh

Двухрегистровый режим сравнения

Специальный режим, при котором один выход модуля CAPCOM1 управляется двумя регистрами захвата/сравнения.

Все регистры модуля CAPCOM1 собраны в два банка регистров – Банк1 и Банк2. Два регистра – один из Банка1, а второй, соответствующий ему, из Банка 2 – образуют регистровую пару. Регистровая пара управляет выводом, соответствующим регистру Банка 1, при этом вывод, соответствующий регистру Банка 2, может использоваться как вывод общего назначения микроконтроллера.

Регистровые пары и выводы, которыми они управляют, указаны в таблице 18.2.

Таблица 18.2 – Регистровые пары

Блок CAPCOM 1				Блок CAPCOM 2			
Регистровая пара		Исполь-зуемый вывод	Управляющее поле в регистре CC1_DRM	Регистровая пара		Исполь-зуемый вывод	Управляющее поле в регистре CC2_DRM
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

Управление двухрегистровым режимом для каждой пары каналов осуществляется с помощью регистра CC1_DRM.

Функциональная схема объединения двух каналов для двухрегистрового режима работы показана на рисунке 18.10.

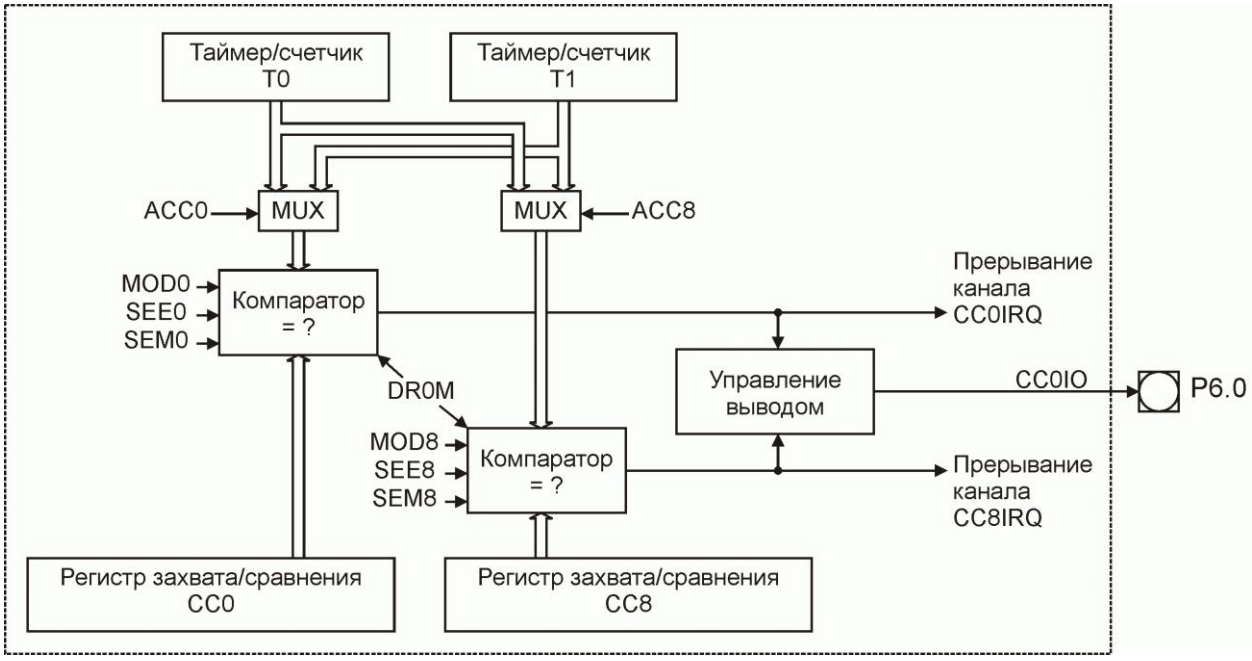


Рисунок 18.10 – Объединение каналов для двухрегистрового режима

Описание функционирования регистров в двухрегистровом режиме здесь приводится на примере регистров CC0 и CC8 модуля CAPCON1 (см. рисунок 18.10). Пара

управляет выводом CC0IO в то время, как вывод CC8IO может использоваться как вывод общего назначения микроконтроллера.

Если двухрегистровый режим активирован, то при возникновении события для любого регистра пары на соответствующей ему линии прерывания генерируется запрос на прерывание. При этом выход CC0IO инвертируется.

Если событие возникло для двух регистров одновременно, то вывод CC0IO будет проинвертирован один раз, но запросов на прерывание будет два – на каждой из линий CC0IRQ и CC8IRQ. На рисунке 18.11 показан пример функционирования регистров CC0 и CC8 в двухрегистровом режиме сравнения. Оба регистра работают с одним таймером.

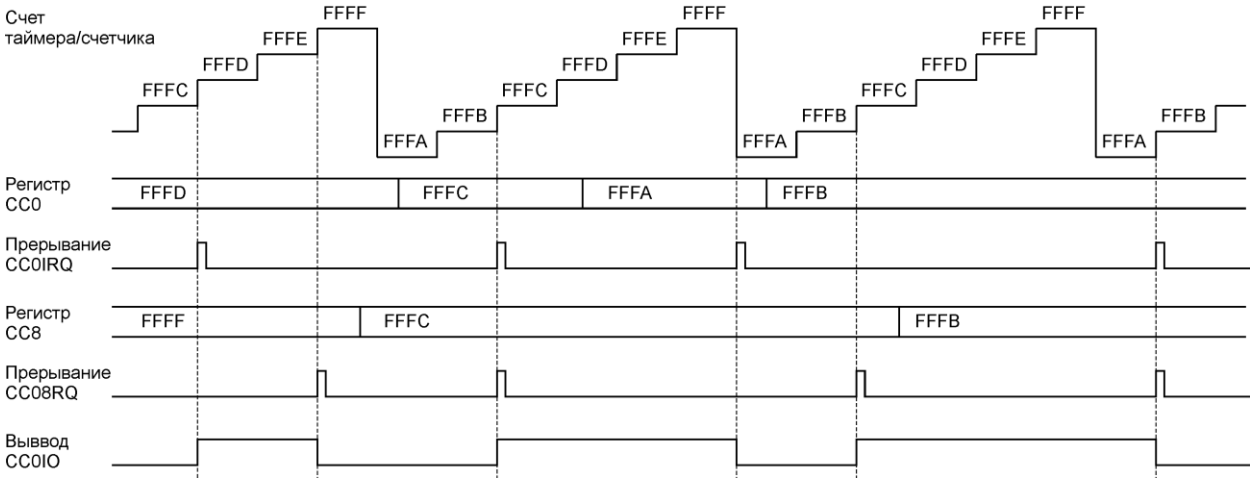


Рисунок 18.11 – Функционирование регистров CC0 и CC8 в двухрегистровом режиме сравнения

В двухрегистровом режиме каждый регистр пары может функционировать, будучи предварительно запрограммированным в индивидуальном порядке на любой желаемый режим сравнения с любым из двух таймеров/счетчиков, что значительно расширяет возможности модуля захвата/сравнения.

18.4 Управление выходными сигналами

Каждый канал захвата/сравнения подключается к соответствующему ему выводу микроконтроллера. Выходной сигнал всегда запоминается в выходной защелке. Помимо этого, если альтернативная функция вывода не включена (регистр ALTSEL), то выходной сигнал попадает непосредственно в защелку порта. В нормальном режиме непосредственное воздействие на защелку порта запрещено.

Функциональная схема передачи сигнала от канала захвата/сравнения до соответствующего вывода порта представлена на рисунке 18.12.

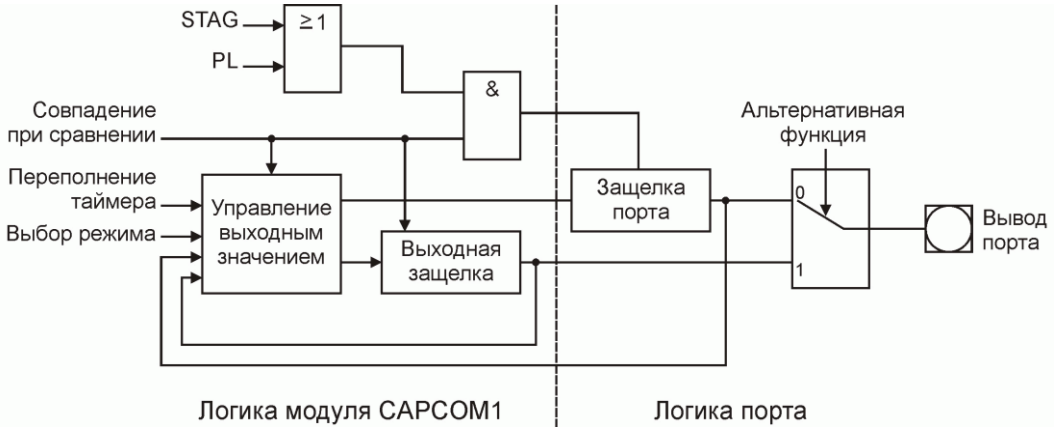


Рисунок 18.12 – Функциональная схема взаимодействия выходной логики канала модуля и логики ссоответствующего вывода порта

Обе защелки доступны программно. Следует помнить, если одновременно происходит аппаратное и программное изменение состояния защелки, то программное изменение имеет приоритет.

Выходные защелки всех каналов модуля CAPCOM1 доступны через регистр CC1_OUT.

Режим однократного срабатывания

Этот режим может быть включен параллельно только с одним из режимов сравнения. Режим позволяет отследить только одно желаемое событие за любой промежуток времени, что в некоторых случаях позволяет упростить программу.

Управление осуществляется посредством двух регистров CC1_SEM и CC1_SEE.

Последовательность действий для программирования режима однократного срабатывания для любого из 16 каналов захвата/сравнения идентична. Рассмотрим эту последовательность на примере нулевого канала.

После того, как выбран один из режимов сравнения и запрограммирован регистр CC0, следует установить бит SEM0 в регистре CC1_SEM. Включать режим можно в любой момент.

После установки бита SEM0 отслеживание событий для соответствующего канала прекращается до тех пор, пока не будет выставлен флаг SEE0. Установка бита SEE0 включает механизм сравнения содержимого регистра CC0 и выбранного таймера. Как только совпадение обнаруживается, флаг SEE0 сбрасывается и дальнейшее отслеживание событий прекращается. Формируется прерывание и, в зависимости от выбранного режима, переключается или остается в неизменном состоянии вывод CC0IO.

Для повторного запуска механизма сравнения следует повторно установить бит SEE0.

Ступенчатый и нормальный режимы работы

Модуль CAPCOM1 может работать в двух основных режимах: ступенчатом и нормальном. Оба режима предназначены для расширения возможностей управления выходными сигналами при формировании последовательностей импульсов, в том числе сигналов ШИМ. Переключение между режимами осуществляется с помощью бита STAG регистра CC1_IOC.

Ступенчатый режим

Включен по умолчанию.

Если при этом один или несколько регистров захвата/сравнения запрограммированы на захват, то функционирование каждого из них будет происходить согласно ранее описанному режиму захвата нулевого канала.

В случае, если один или несколько регистров захвата/сравнения запрограммированы на сравнение, то переключение сигналов на выводах модуля CAPCOM1 будет зависеть как от выбранного режима сравнения, так и от режима, в котором работает выбранный таймер.

Рассмотрим случай, когда таймер/счетчик запрограммирован как таймер.

Функционирование регистров захват/сравнения в режимах сравнения 0 и 2 будет происходить, согласно описанным ранее режиму сравнения 0 и режиму сравнения 2.

Функционирование регистров в режимах сравнения 1 и 3 будет происходить, согласно описанным ранее режиму сравнения 1 и режиму сравнения 3, с той особенностью, что каждый из каналов захвата/сравнения в случае возникновения события, может переключить соответствующий вывод только в свой, заранее определенный, промежуток времени. Время между двумя последовательными переключениями таймера аппаратно разбивается на восемь равных промежутков. Промежутки от первого до восьмого соответствуют каналам захвата/сравнения от нулевого до седьмого, а также – от восьмого до 15-го.

На рисунке 18.13 приведен пример для случая, когда таймер работает на своей максимальной (для ступенчатого режима) частоте, равной fcc/8. Значение загрузки таймера по переполнению равно FFFCh. Все регистры модуля CAPCOM1 работают в режиме сравнения 3. В момент инкрементирования таймера до значения FFFEh происходит сравнение всех 16 регистров с таймером. Совпадения обнаруживаются для регистров CC0, CC1, CC2, CC9, CC10, CC12 и CC15. С запаздыванием в один такт сигнала Fcc переключится в единицу вывод CC0IO, соответствующий регистру CC0 (первый промежуток). Далее, еще через такт – вывод CC1IO (второй промежуток). На третьем и пятом промежутках переключатся, соответственно, выходы CC2IO и CC4IO.

Для регистров CC3, CC5 и CC6 совпадения будут обнаружены в следующем такте таймера, когда его значение станет равно FFFFh, и соответствующие выходы будут переключены в единицы на четвертом, шестом и седьмом промежутках. Регистр CC7 содержит значение FFFDh, поэтому для него совпадение с таймером произошло раньше. Выход CC7IO переключится в единицу на соответствующем ему восьмом промежутке, когда значение таймера было равно FFFDh.

Порядок переключения выходов с CC8IO по CC15IO, соответствующих регистрам с CC8 по CC15 такой же, как для регистров CC0 по CC7 (регистры этих двух групп – банков – могут работать попарно). Так, например, на рисунке 18.13 для значения таймера FFFEh пары выходов CC1IO и CC9IO, CC2IO и CC10IO, CC4IO и CC12IO переключаются одновременно.

В отличие от выходных сигналов, прерывания от всех каналов, для которых произошло совпадение, генерируются одновременно в момент обнаружения совпадения.

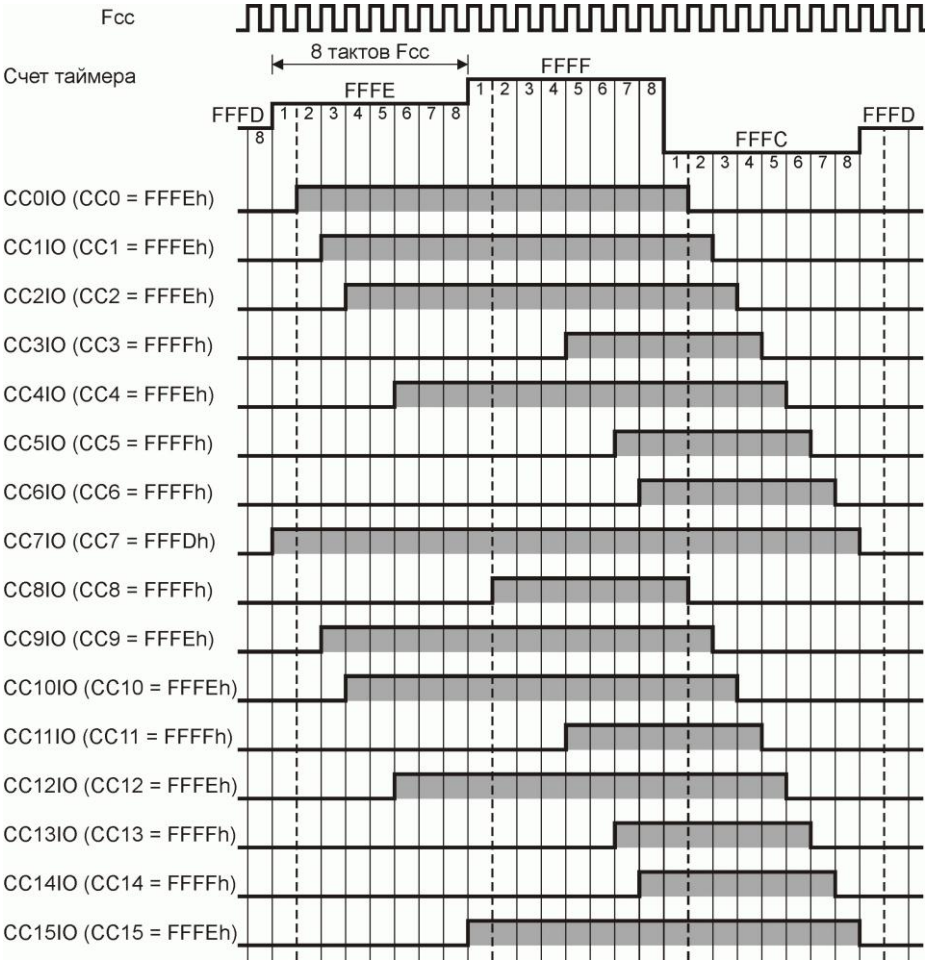


Рисунок 18.13 – Ступенчатый режим. Частота работы таймера fcc/8

Поскольку все регистры работают в режиме сравнения 3, то после загрузки таймера значением FFFCh, вызванной его переполнением, все выходы будут сброшены в ноль. Переключение выходов в ноль также происходит последовательно. Каждый выход переключается в свой временной промежуток.

На рисунках 18.14 и 18.15 показаны примеры для случаев, когда таймер работает на частотах fcc/16 и fcc/32 соответственно.

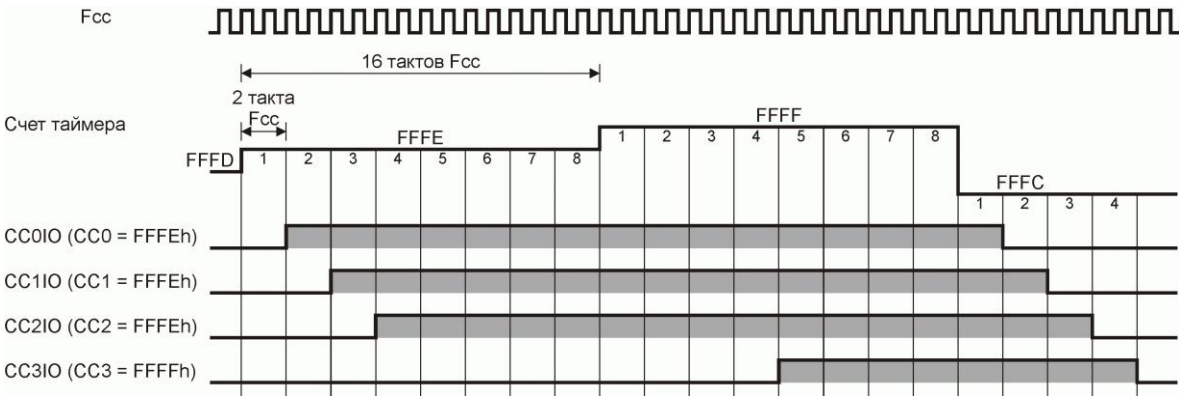


Рисунок 18.14 – Ступенчатый режим. Частота работы таймера fcc/16

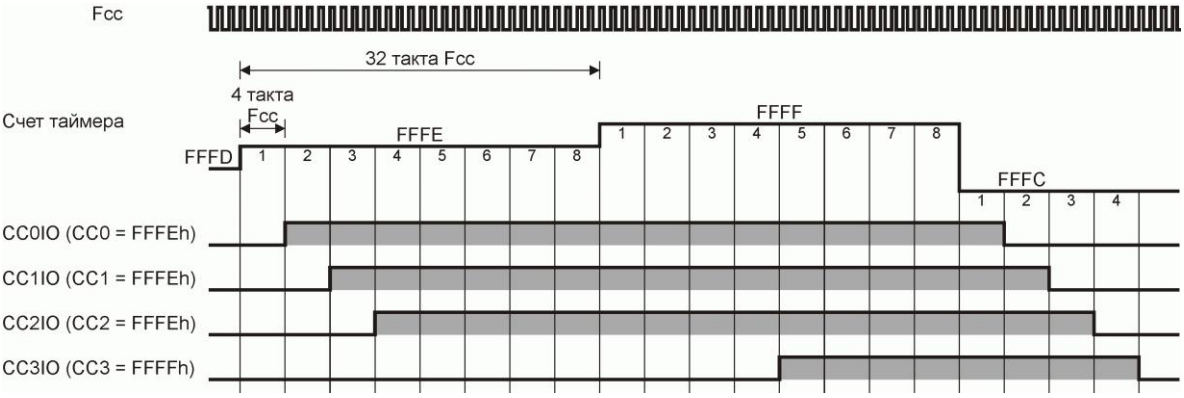


Рисунок 18.15 – Ступенчатый режим. Частота работы таймера fcc/32

В случае, если один или несколько регистров запрограммированы на работу в одном (любом) из режимов сравнения, а таймер/счетчик запрограммирован как счетчик, то регистры будут аппаратно переключены на работу в режиме сравнения 0.

Примечание – В ступенчатом режиме возможно непосредственное влияние на портовую защелку.

Нормальный режим

Включается установкой бита STAG регистра CC1IOC и применяется для достижения максимальной скорости работы модуля CAPCOM1.

В нормальном режиме функционирование каждого канала захвата/сравнения не зависит от режима, в котором работает выбранный таймер/счетчик, а зависит только от выбранного режима работы регистра канала.

Если один или несколько регистров захвата/сравнения запрограммированы на захват, то функционирование каждого из них будет происходить согласно ранее описанному режиму захвата нулевого канала.

Если один или несколько регистров захвата/сравнения запрограммированы на сравнение, то функционирование каждого из них будет происходить согласно ранее описанному, начиная с режимов сравнения нулевого канала.

В отличие от ступенчатого режима, в нормальном режиме при одновременном возникновении совпадений для одного или более регистров, соответствующие им выходы будут переключены одновременно.

Прерывания от всех каналов, для которых произошло совпадение, генерируются одновременно в момент обнаружения совпадения.

На рисунке 18.16 показан пример для случая, когда таймер/счетчик работает на своей максимальной частоте (для нормального режима), равной fcc. Значение загрузки таймера по переполнению равно FFFCh. Все регистры модуля CAPCOM1 работают в режиме сравнения 3. В момент инкрементирования таймера до значения FFFEh происходит сравнение всех 16 регистров с таймером. Совпадения обнаруживаются для регистров CC0, CC1, CC2, CC4, CC9, CC10, CC12 и CC15. С запаздыванием в один такт сигнала Fcc, а именно в момент очередного инкрементирования таймера/счетчика выводы CC0IO, CC1IO, CC2IO, CC4IO, CC9IO, CC10IO, CC12IO и CC15IO переключатся в единицы. Аналогично, в свое время, переключатся и остальные выходы.

Регистр CC7 содержит значение FFFDh, поэтому для него совпадение с таймером произошло раньше и выход CC7IO переключился в единицу в момент инкрементирования таймера/счетчика до значения FFFEh.

Поскольку все регистры работают в режиме сравнения 3, то после загрузки таймера значением FFFCh, вызванной его переполнением, все выходы одновременно будут сброшены в ноль, с запаздыванием в один такт сигнала Fcc.

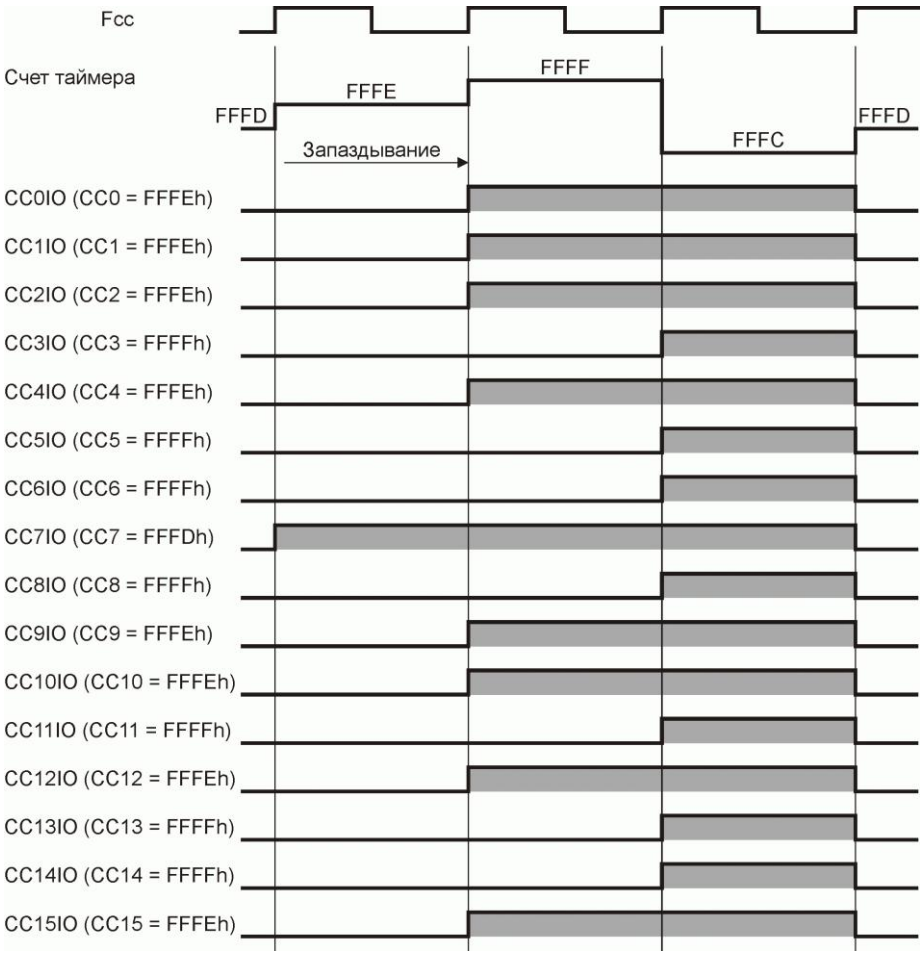


Рисунок 18.16 – Нормальный режим. Частота работы таймера/счетчика fcc

На рисунке 18.17 показан пример для случая, когда таймер работает на частоте меньшей, чем fcc (в данном примере fcc/8).

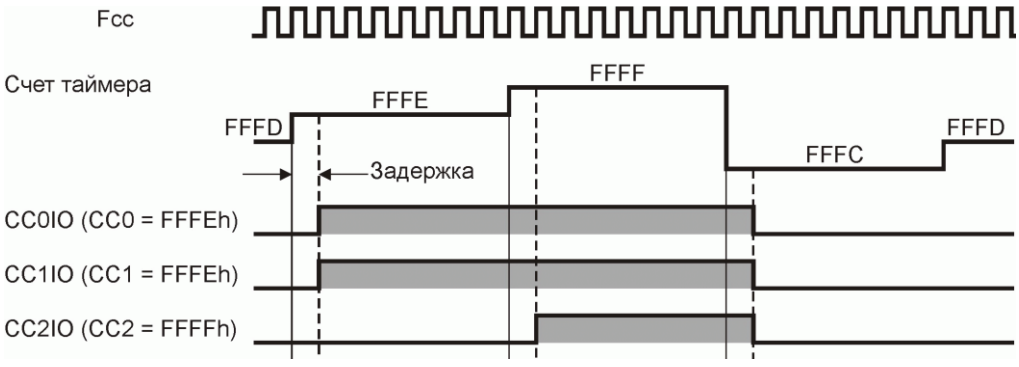


Рисунок 18.17 – Нормальный режим. Частота работы таймера/счетчика fcc/8

В момент, когда таймер/счетчик достигает значения FFFEh, возникают совпадения для регистров CC0 и CC1. Соответствующие им выходы CC0IO и CC1IO переключатся в единицы с запаздыванием в один такт Fcc («Задержка» на рисунке 18.17). Аналогично переключится выход CC2IO после достижения таймером/счетчиком значения FFFFh. После перезагрузки таймера/счетчика на всех выходах установится ноль.

Независимо от скорости работы таймера/счетчика, задержка в переключении сигналов на выходах будет оставаться постоянной и равной одному такту сигнала Fcc.

Примечание – В нормальном режиме непосредственное влияние на портовую защелку запрещено.

Общие замечания по работе модулей CAPCOM1 и CAPCOM2

1 Выводы порта P2, соединенные с выводами CC8IO – CC15IO модуля CAPCOM1, могут быть сконфигурированы как входы внешних прерываний. В этом случае сигналы с этих выводов поступают в контроллер прерываний.

2 В режиме захвата внешние сигналы, которые являются событиями, могут быть сгенерированы программно.

3 У модулей захвата/сравнения есть один общий вывод микроконтроллера –вывод 15 порта P2 (см. рисунок 18.18).

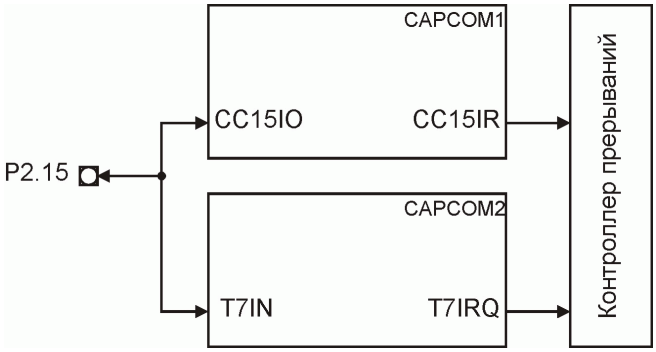


Рисунок 18.18 – Общий вывод модулей захвата/сравнения

Посредством этого вывода работа обоих модулей может быть скомбинирована:

- таймер T7 модуля CAPCOM2 может тактироваться сигналом с выхода CC15IO модуля CAPCOM1;
- сигнал, поступающий с вывода P2.15 и тактирующий таймер T7, может быть параллельно захвачен каналом 15 модуля CAPCOM1 с входа CC15IN.

19 Блок захвата/сравнения и ШИМ (CAPCOM6)

Блок CAPCOM6 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.

Структурная схема блока CAPCOM6 представлена на рисунке 19.1.

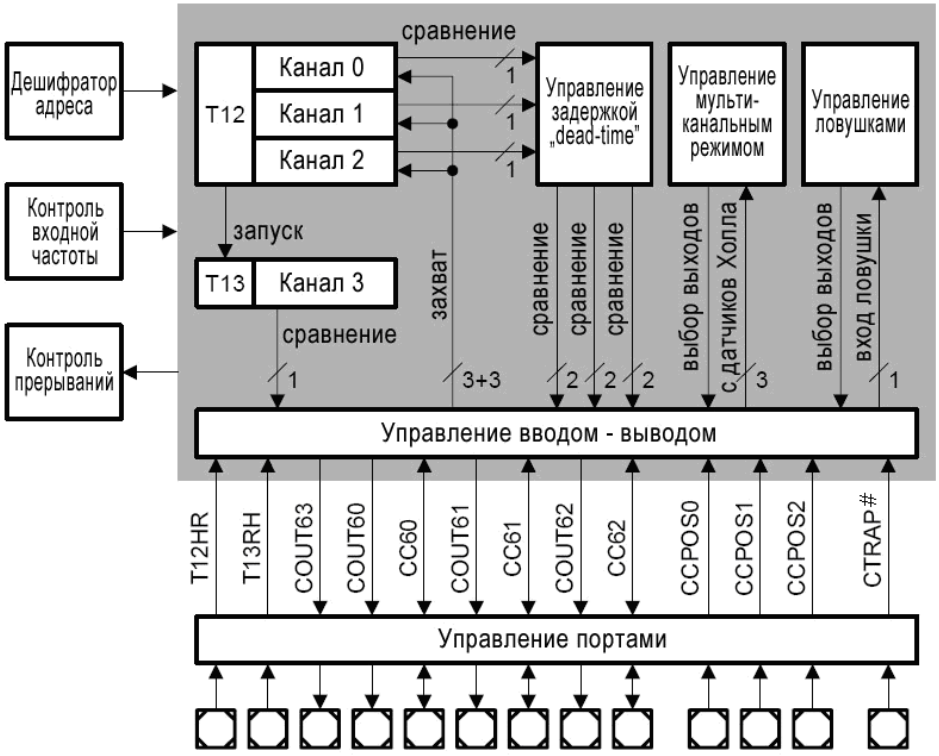


Рисунок 19.1 – Структурная схема блока CAPCOM6

Особенности блока таймера T12:

- три 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-сигнала;
- 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.

Три канала таймера T12 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.

19.1 Блок таймера T12

Блок таймера T12 (см. рисунок 19.2) является основным генератором трехфазной ШИМ. 16-разрядный 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 также возможность включения временной задержки «dead-time» и блокирования.

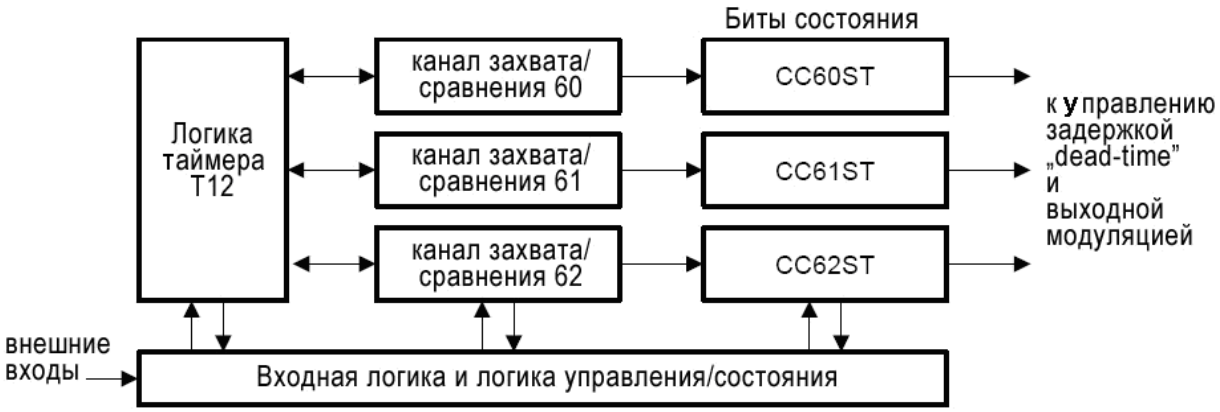


Рисунок 19.2 – Блок таймера T12

На рисунке 19.3 показана подробная схема структуры таймера T12. Таймер T12 получает входную тактовую частоту ft12 из частоты fosc 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 от выбранного режима. 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 однонаправленного счетчика по достижении значения периода, таймер сбрасывается в ноль. В режиме двунаправленного счетчика по достижении значения периода направление счета таймера меняется на противоположное (включается декрементирование). Следует помнить, что в режиме двунаправленного счетчика

значение счетчика таймера T12 превысит значение величины периода на один прежде, чем начнется обратный счет.

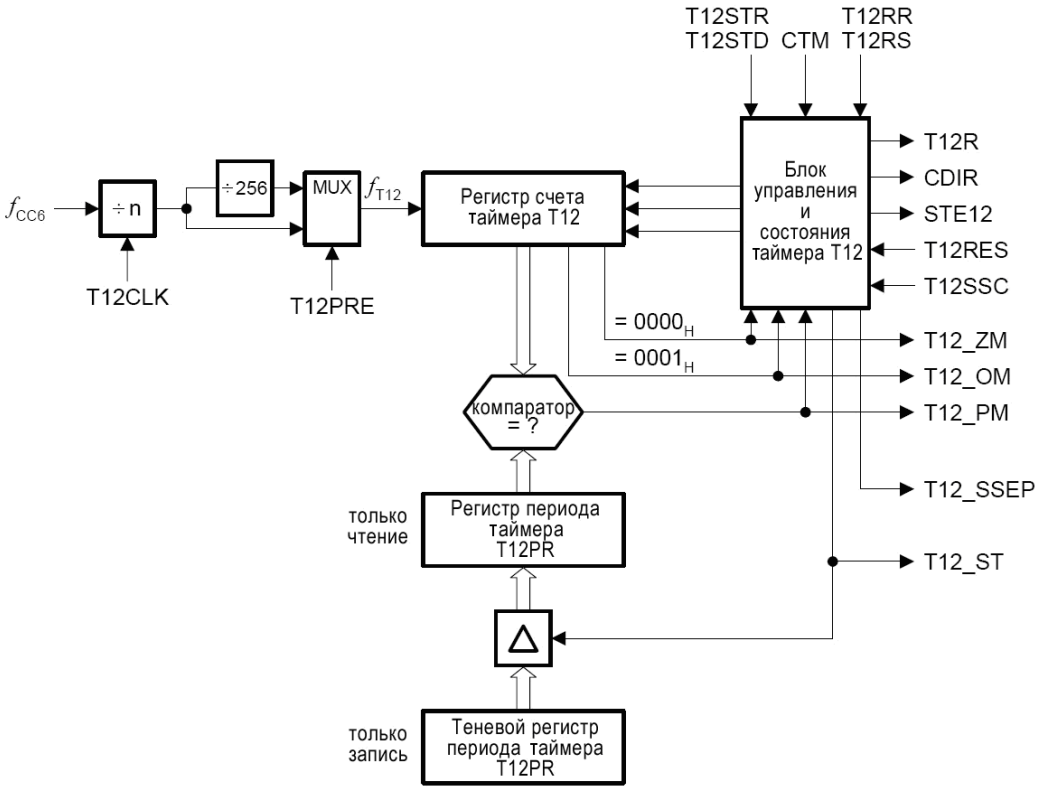


Рисунок 19.3 – Компаратор периода и логика таймера T12

В обоих режимах при достижении таймером значения периода генерируется сигнал T12_PM (совпадение по периоду).

Регистр периода CCU6_T12 таймера T12PR 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 0001h, и сигнал T12_ZM совпадения с нулем.

Режим работы таймера T12 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.

Регистр T12PR содержит значение периода таймера T12. Значение периода сравнивается с текущим значением счетчика таймера T12, и действия, производимые после, зависят от правил счета. 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 ft12 таймера T12 получается из частоты fosc посредством программируемого делителя и делителя 1/256 (см. таблицу 19.1).

Таблица 19.1 – Входная частота таймера T12

T12CLK/T13CLK	Входная частота таймера ft12	
	Дополнительный делитель выключен (T12PRE/T13PRE = 0)	Дополнительный делитель включен (T12PRE/T13PRE = 1)
000	fosc	fosc/256
001	fosc/2	fosc/512
010	fosc 4	fosc/1 024
011	fosc/8	fosc/2 048
100	fosc/16	fosc/4 096
101	fosc/32	fosc/8 192
110	fosc/64	fosc/16 384
111	fosc/128	fosc/32 768

Период счета таймера определяется значением в регистре периода T12PR и режимом таймера.

В режиме однонаправленного счетчика период равен:

T12per = «значение периода» + 1, в тактах таймера T12.

В режиме двунаправленного счетчика:

T12per = («значение периода» + 1) × 2, в тактах таймера T12.

Таймер T12 в режиме однонаправленного счетчика

Если с приходом очередного такта ft12 возникает совпадение по периоду, счетчик сбрасывается в ноль (см. рисунок 19.4).

Направление счета всегда «вверх», CDIR = 0b.

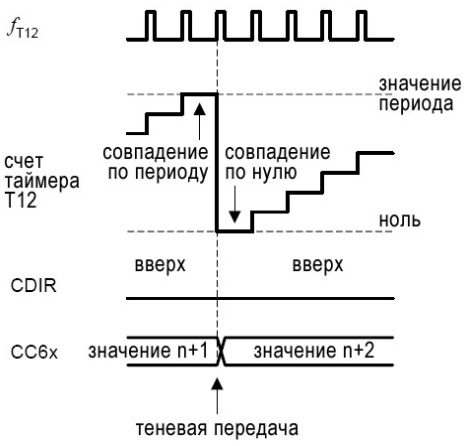


Рисунок 19.4 – Режим однонаправленного счетчика таймера T12

Таймер T12 в режиме двунаправленного счетчика

Если с приходом очередного такта ft12 возникает совпадение по периоду при счете «вверх», направление счета устанавливается «вниз» и CDIR принимает значение «1» (см. рисунок 19.5).

Если с приходом очередного такта ft12 при счете «вниз» возникает совпадение с единицей (счетчик = 0001h), направление счета устанавливается «вверх» и CDIR принимает значение «0».

Бит CDIR является индикатором направления счета таймера T12 (если CDIR = 0b, значит таймер считает «вверх»; если CDIR = 1b, 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.

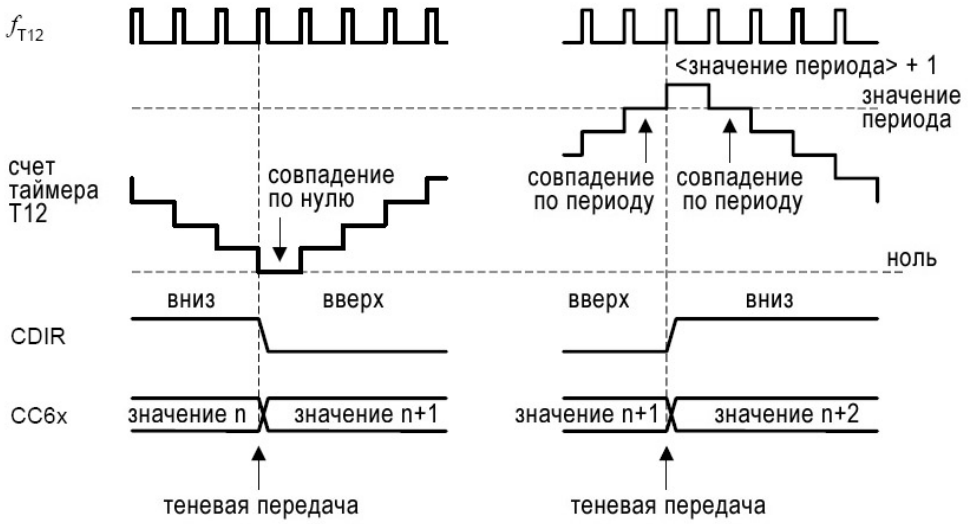


Рисунок 19.5 – Режим двунаправленного счетчика таймера T12

Примечание – На рисунках 19.4 – 19.7, 19.10 – 19.12 показаны варианты изменения сигналов для случаев, когда частота переключения счетчика таймера максимальна и совпадает с fcc, т. е. T12CLK = 000b и T12PRE = 0b.

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 = 1000b, 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. 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 0000h), но не оказывает другого влияния, например, не останавливает его.

Режим однократного срабатывания

В режиме однократного срабатывания (см. рисунок 19.6) 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, то счетчик таймера T12 после очередного запуска остановится, отсчитав один период таймера. В режиме однонаправленного счетчика это произойдет тогда, когда таймер обнулится по достижении заданного значения периода. В режиме двунаправленного счетчика период закончится, когда таймер досчитает «вниз» до 0 (см. рисунок 19.7).

Для выхода из режима необходимо записать «0» в бит T12SSC.

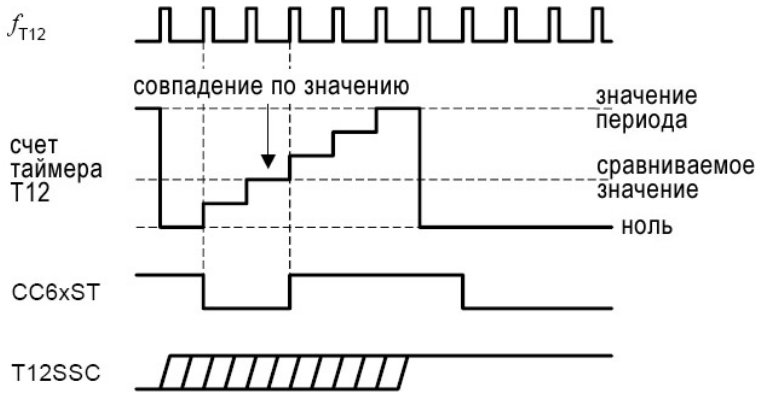


Рисунок 19.6 – Однократное срабатывание в режиме однонаправленного счетчика

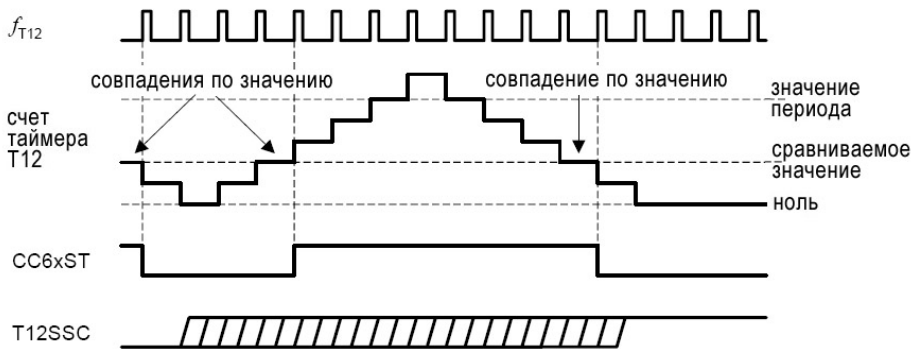


Рисунок 19.7 – Однократное срабатывание в режиме двунаправленного счетчика

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 19.8) три канала могут работать как независимо друг от друга, так и совместно для генерирования трехфазного ШИМ-сигнала.

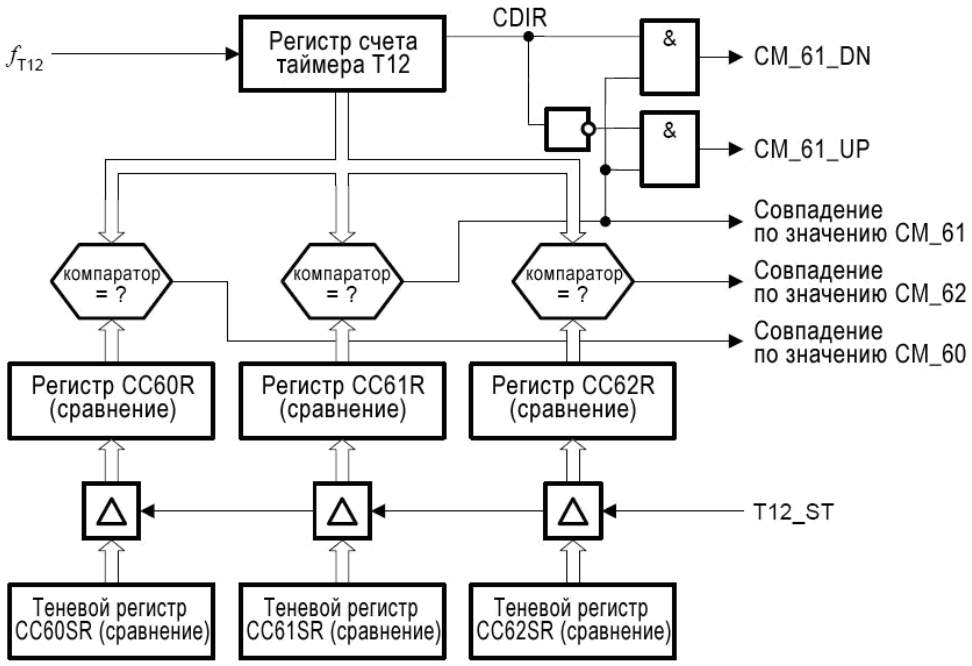


Рисунок 19.8 – Компараторы каналов таймера T12

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 19.9.

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.

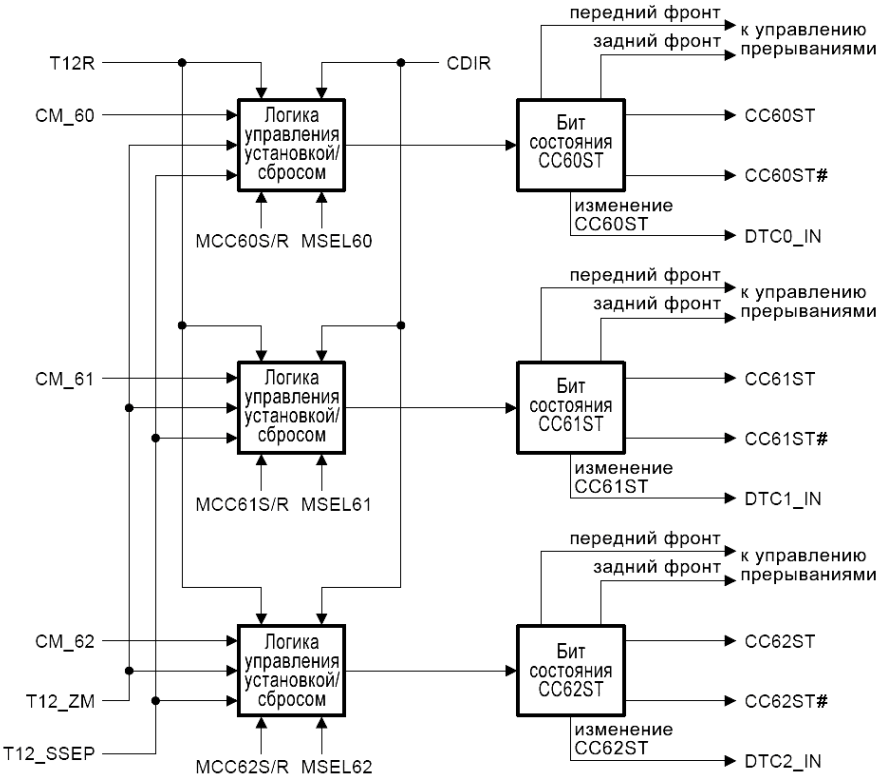


Рисунок 19.9 – Биты состояния

Аппаратное изменение бита состояния CC6xST возможно только во время работы таймера T12 (T12R = 1b). Поведение бита состояния в разных режимах показано на рисунках 19.10 и 19.11.

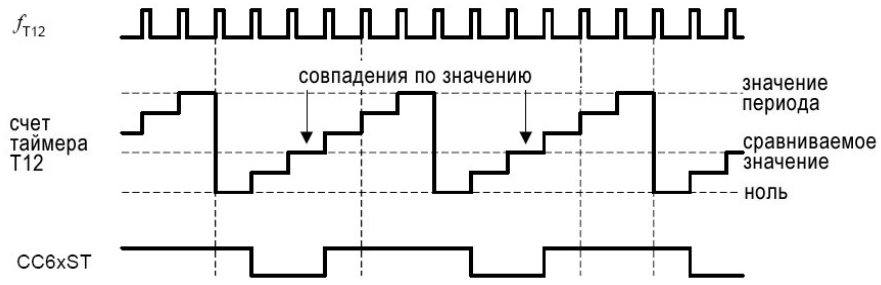


Рисунок 19.10 – Операция сравнения в режиме однонаправленного счетчика

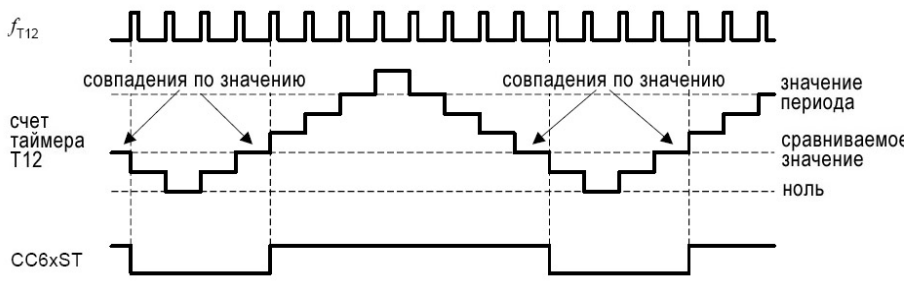


Рисунок 19.11 – Операция сравнения в режиме двунаправленного счетчика

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 0000h;
- при переключении направления счета счетчика таймера («симметричный») из «вниз» в «вверх» возникает совпадение по значению, равному 0000h.

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 0000h.

На рисунке 19.12 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).

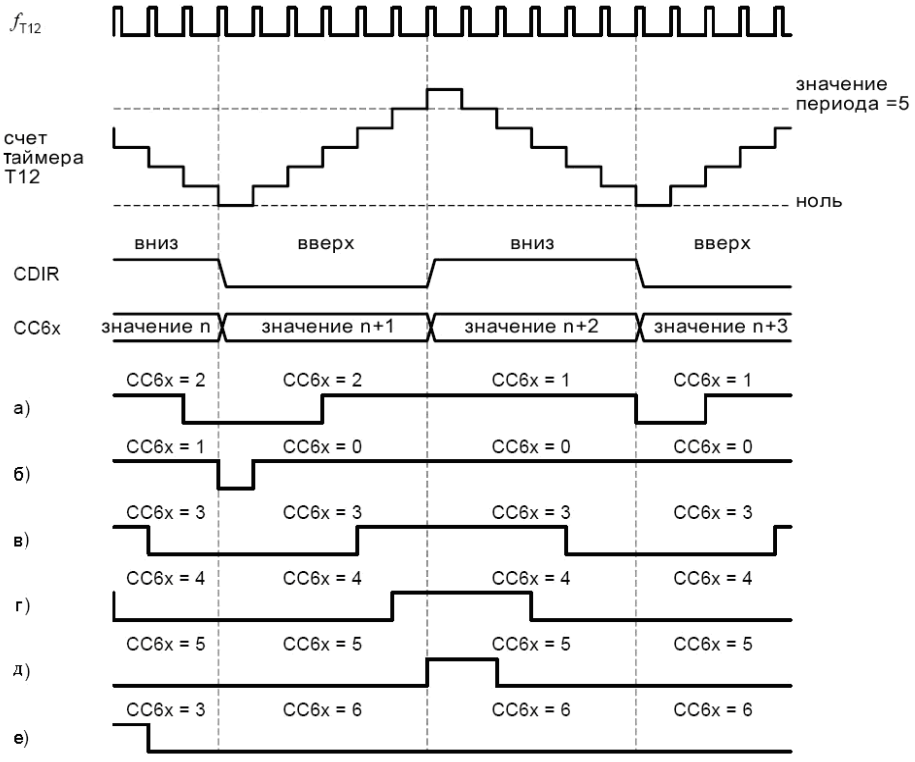


Рисунок 19.12 – Примеры переключения сигналов в режиме сравнения

Пример б) иллюстрирует передачу при 100 % заполнении. Используется сравниваемое значение 0001h, меняющееся затем на 0000h. Следует заметить, что отрицательный импульс с длиной в один такт появляется в период, когда вступает в силу новое значение 0000h. Этот импульс возникает из предыдущего значения 0001h. 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 е) показывает передачу при 0 % заполнении. Новое сравниваемое значение устанавливается в значение, равное «значение периода» плюс 1, и бит состояния CC6xST остается равным «0».

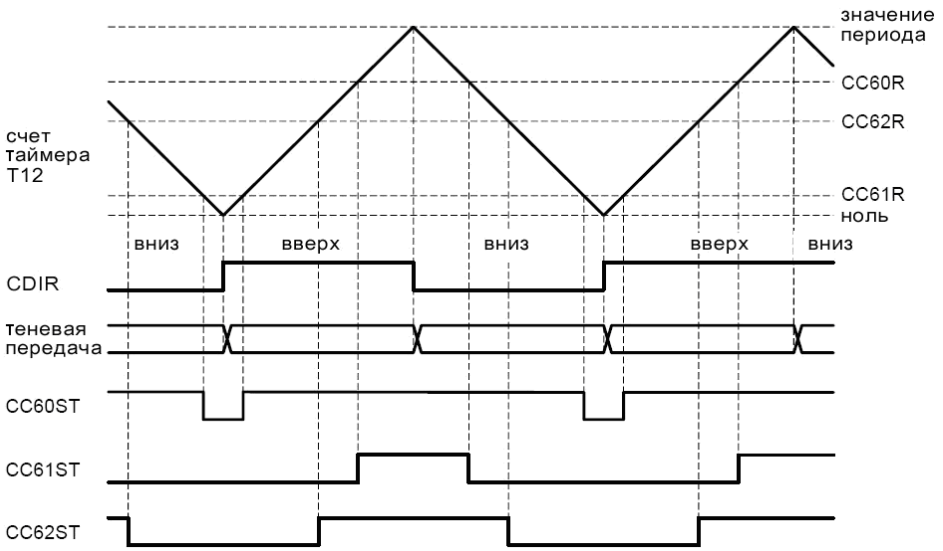


Рисунок 19.13 – Сигналы сравнения трех каналов

На рисунке 19.13 показаны сигналы трех каналов. С применением задержки «dead-time» и контроля выходной модуляции может быть сгенерирован достаточно хороший ШИМ сигнал.

Вывод сигнала в режиме сравнения

На рисунке 19.14 отражен в общем виде путь сигнала от бита состояния канала до выхода.

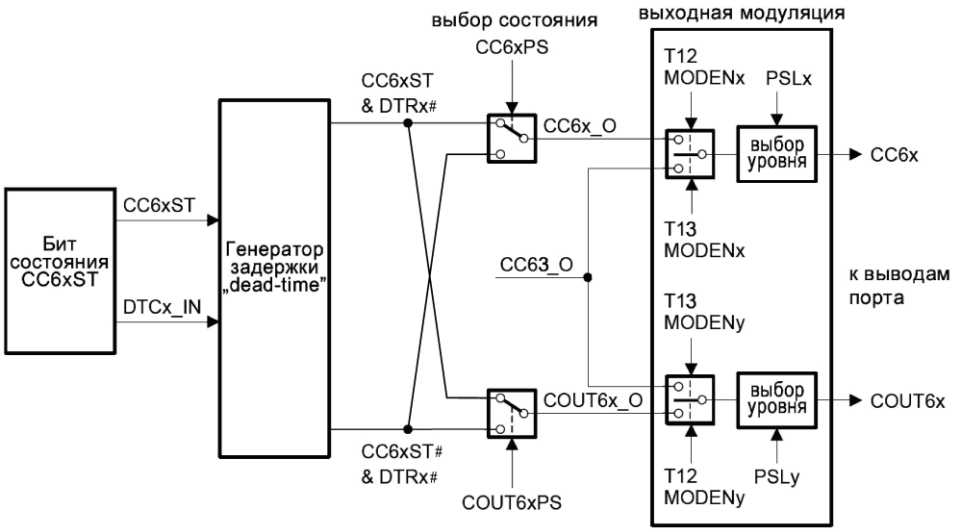


Рисунок 19.14 – Вывод сигнала в режиме сравнения

19.2 Регистры захвата/сравнения

Регистры CC6xR (x – номер канала = 0, 1, 2) 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.

Регистры CC6xR в режиме сравнения

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

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.

19.3 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 19.15, 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 T12DTC, и происходит запуск счета.

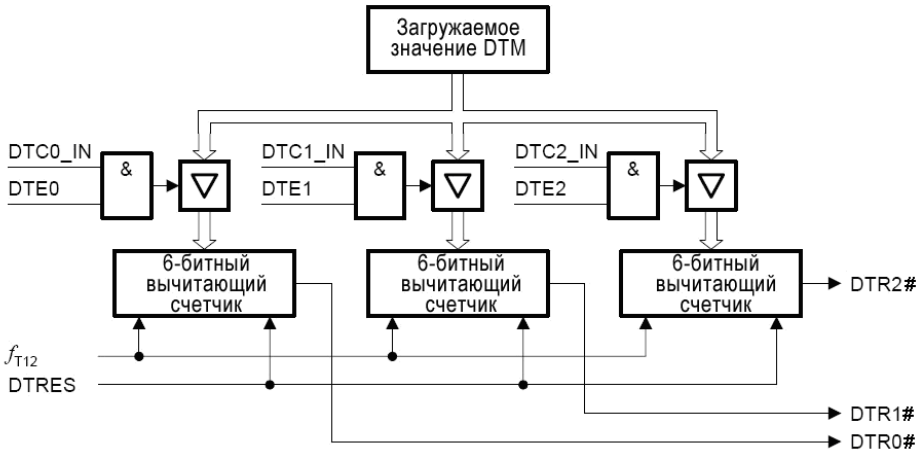


Рисунок 19.15 – Генерация задержки «dead-time»

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 19.16 показаны переключения сигналов с внесенной задержкой «dead-time».

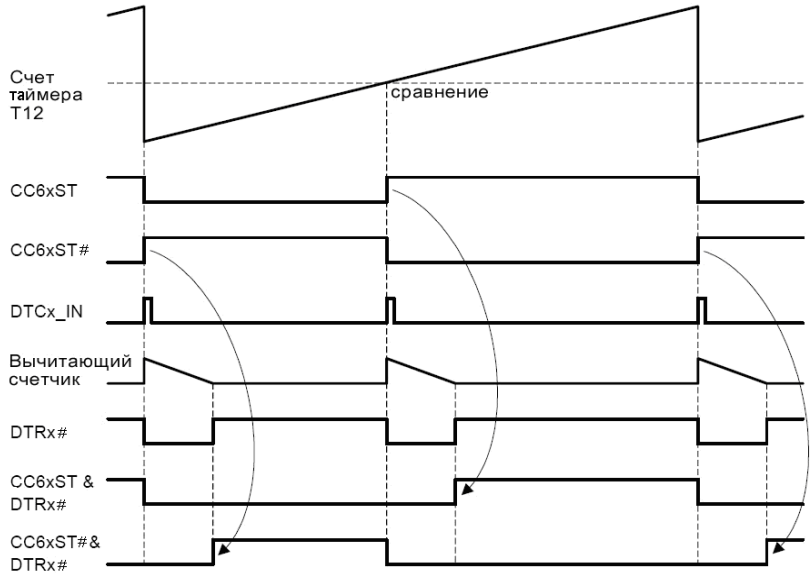


Рисунок 19.16 – Сигналы при внесении задержки «dead-time»

Регистр T12DTC управляет внесением задержки «dead-time» в изменение сигналов каналов таймера T12 (каждый канал программируется индивидуально).

19.4 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.

Режим захвата 1

На рисунке 19.17 показан режим захвата 1 MSEL = 0100b. 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 заносится в CC6xSR.

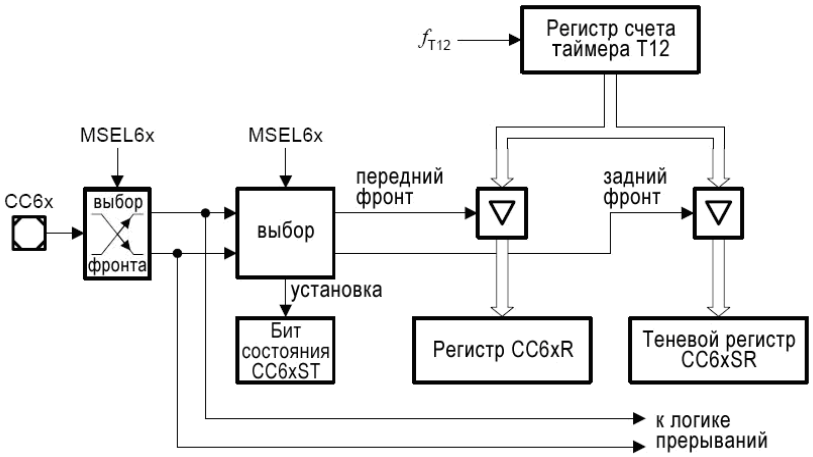


Рисунок 19.17 – Режим захвата 1 таймера T12

Режимы захвата 2, 3 и 4

Функциональная схема режимов захвата 2, 3 и 4 представлена на рисунке 19.18. 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.

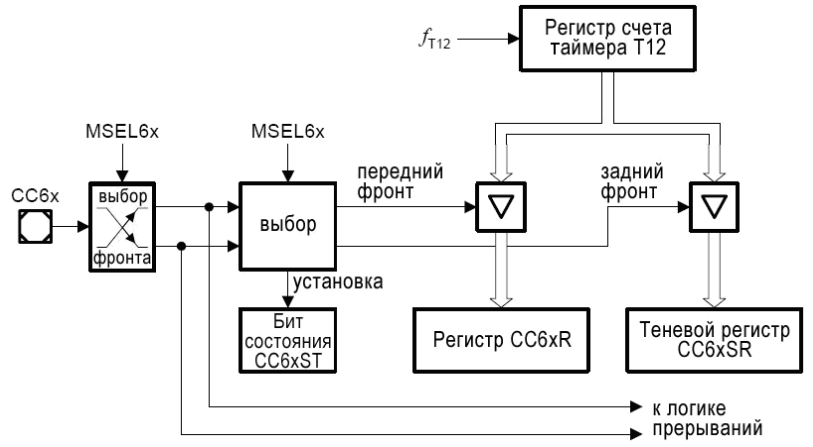


Рисунок 19.18 – Функциональная схема режимов захвата 2, 3 и 4 таймера T12

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 19.19.

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.

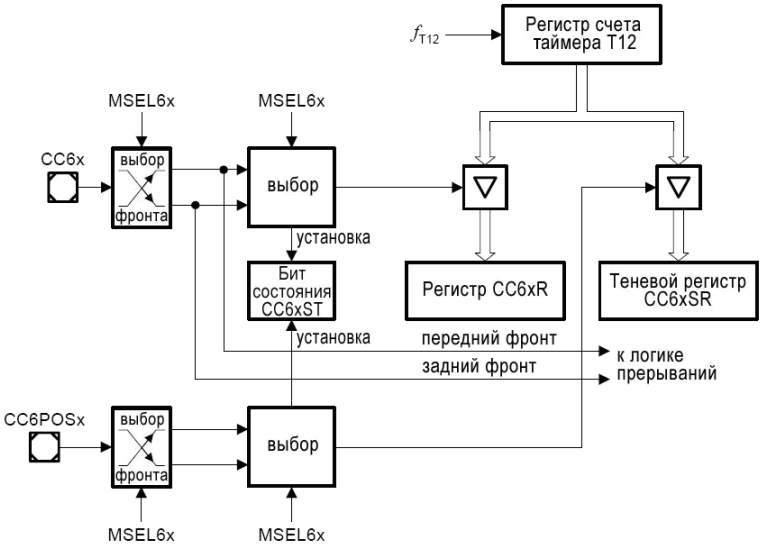


Рисунок 19.19 – Мультивходовый режим захвата таймера T12

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 представляет собой возможность блокирования изменений бита состояния канала x, согласно уровню сигнала на входе CCPOSx, см. рисунок 19.20.

Если вход CCPOSx становится 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.

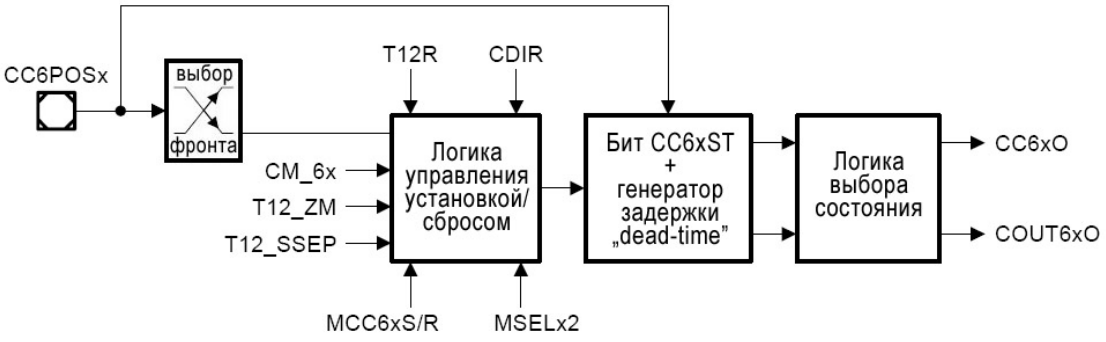


Рисунок 19.20 – Логика работы в режиме блокирования

Примечание – Если выходы канала x контролировались согласно заданному режиму MSELx и схема работала в режиме сравнения, то запись значения 1001b в MSELx не внесет изменений в работу, но режим блокирования включится. Чтение соответствующего поля MSELx вернет значение 1001b. Следует помнить, что схема будет продолжать работать в режиме, который был задан в поле MSELx на момент записи в него значения 1001b, 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.

19.5 Блок таймера T13

Таймер 13 подобен таймеру T12, но имеет только один канал, работающий в режиме сравнения, см. рисунок 19.21. 16-разрядный 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.

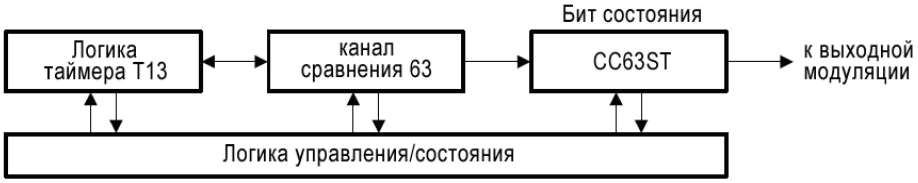


Рисунок 19.21 – Блок таймера T13

На рисунке 19.22 показана подробная схема таймера T13. Таймер получает входную тактовую частоту ft13 из частоты fosc 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 однонаправленного счетчика таймера T12.

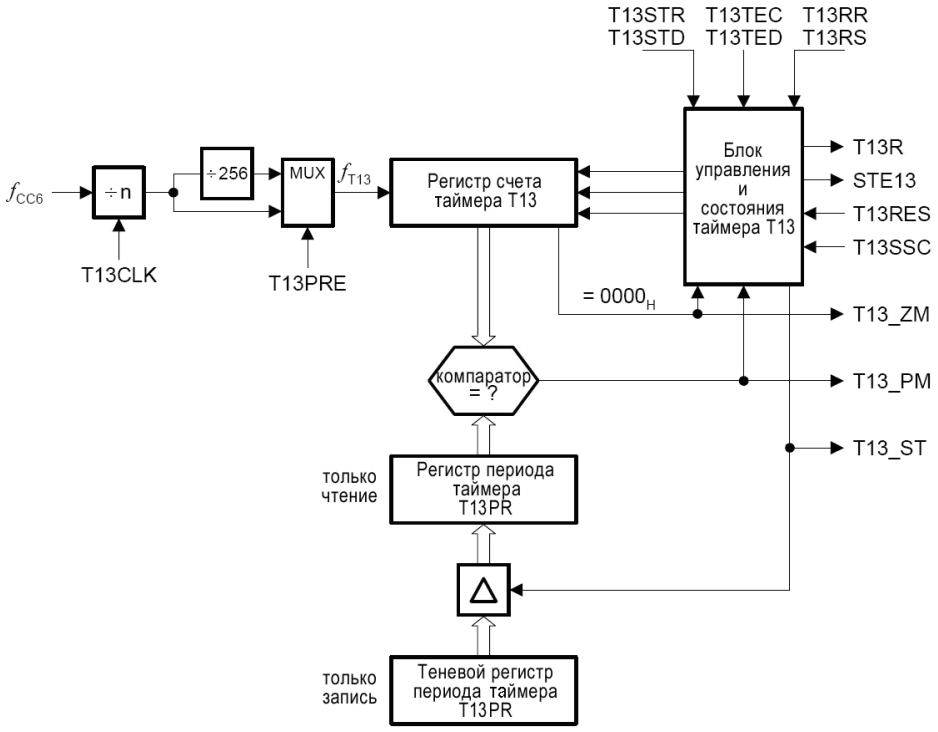


Рисунок 19.22 – Компаратор периода и логика таймера T13

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 0000h.

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 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.

Регистр T13PR содержит значение периода таймера T13. Значение периода сравнивается с текущим значением счетчика таймера T13 и действия, производимые после, зависят от правил счета.

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 ft13 таймера T13, см. таблицу 19.1, получается из частоты fosc посредством программируемого делителя и делителя 1/256.

Период счета таймера определяется значением в регистре периода T13PR и режимом таймера.

T13per = «значение периода» + 1, в тактах таймера T13.

Если с приходом очередного такта ft13 возникает совпадение по периоду, счетчик сбрасывается в ноль, см. рисунок 19.23.

Примечание – Далее на рисунках показаны варианты изменения сигналов для случаев, когда частота переключения счетчика таймера максимальна и совпадает с f_{CC}, т.е. T13CLK = 000b и T13PRE = 0.

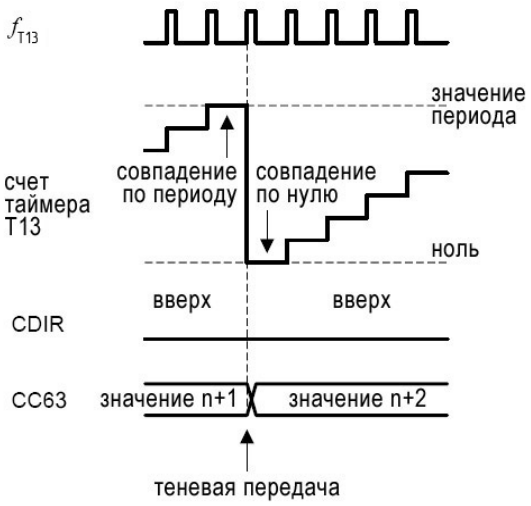


Рисунок 19.23 – Работа таймера T13

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. 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 0000h) синхронно с инкрементированием, но не оказывает другого влияния, например, не останавливает его.

Режим однократного срабатывания

В режиме однократного срабатывания, см. рисунок 19.24, бит запуска T13R находится под программным и аппаратным влиянием.

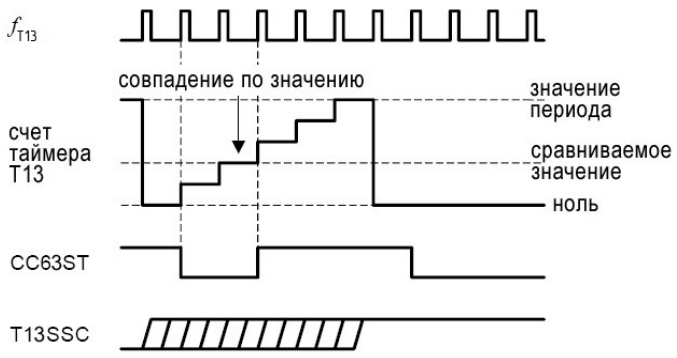


Рисунок 19.24 – Однократное срабатывание

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.

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 19.25 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.

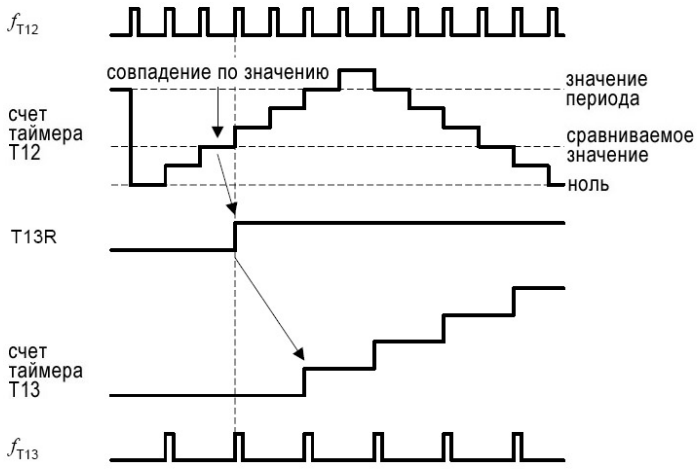


Рисунок 19.25 – Синхронизация запуска таймера T13 по таймеру T12

Битовое поле T13TEC выбирает событие (связанное с таймером T12), по которому следует запустить таймер T13, а битовое поле T13TED определяет, какое направление («вверх» или «вниз») счета таймера T12 при этом используется.

19.6 Режимы сравнения T13

С таймером T13 связан один канал сравнения, см. рисунок 19.26.

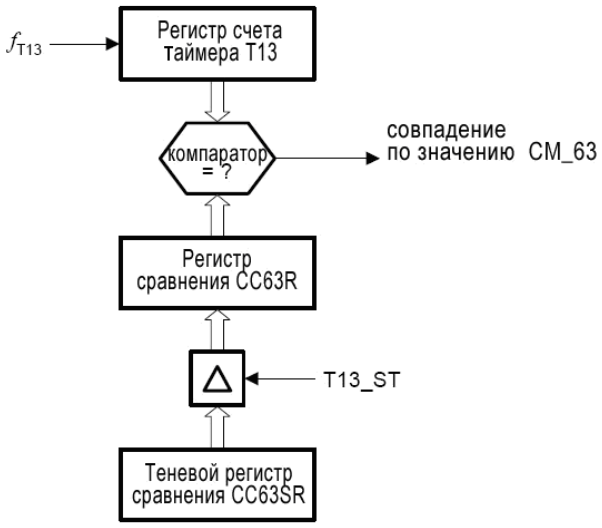


Рисунок 19.26 – Компаратор канала таймера T13

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 19.27.

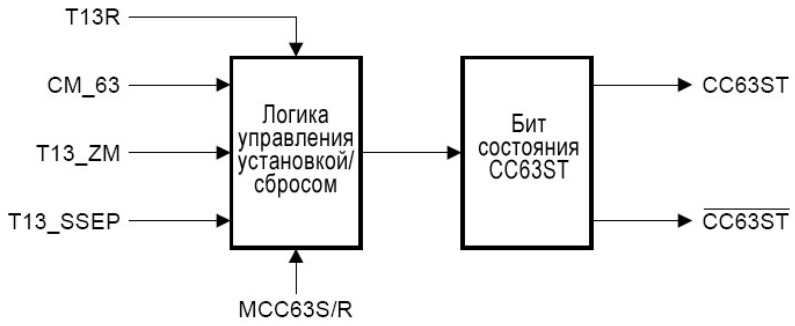


Рисунок 19.27 – Бит состояния канала таймера T13

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). 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 0000h.

Бит CC63ST переключается в «0», если при сбросе счетчика таймера в ноль по переполнению не возникает совпадение по значению, равное 0000h.

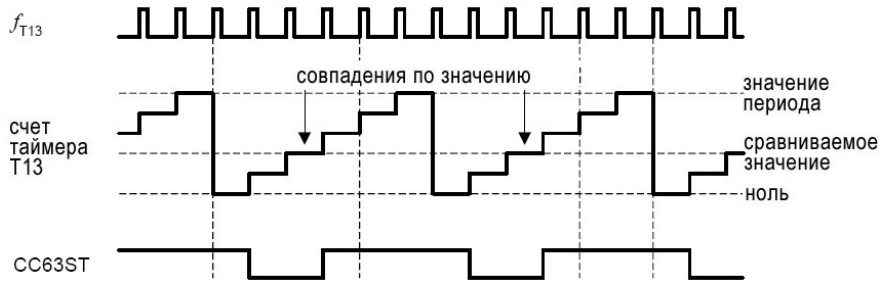


Рисунок 19.28 – Операция сравнения таймера T13

Примечание – Сигналы на рисунке 19.28 аналогичны сигналам таймера T12.

Вывод сигнала в режиме сравнения

На рисунке 19.29 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.

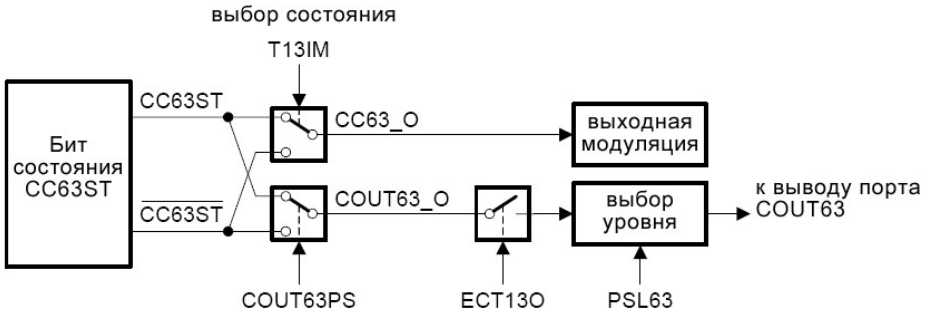


Рисунок 19.29 – Вывод сигнала в режиме сравнения

19.7 Регистры сравнения

Регистр CC63R является регистром, который может быть только программно прочитан. Запись в этот регистр осуществляется через соответствующий ему теневой регистр CC63SR, который может программно читаться и записываться. Перенос значения из теневого регистра в регистр CC63R происходит по сигналу теневой передачи T13_ST.

19.8 Мультиканальный режим

Мультиканальный режим позволяет управлять модуляцией всех шести выходных сигналов таймера T12. Для включения режима необходимо установить бит MCMEN.

Регистр CCU6_MODCTR состоит из битов, управляющих комбинацией и модуляцией выходных сигналов каналов обоих таймеров.

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 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 19.30 показан путь формирования сигнала теневой передачи MCM_ST.

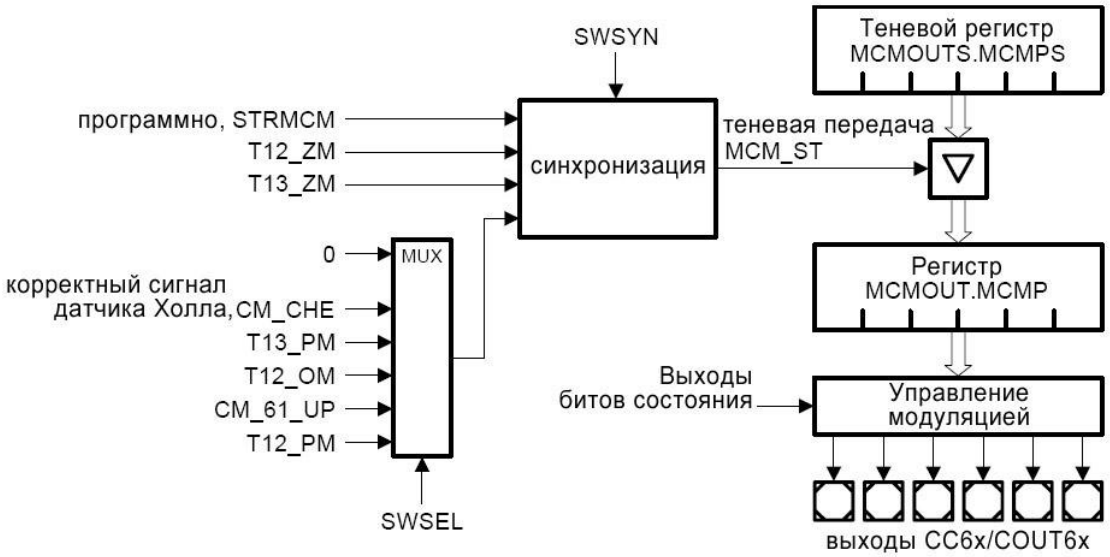


Рисунок 19.30 – Мультиканальный режим

Управление осуществляется посредством регистра CCU6_MCMCTR.

Примечание – Аппаратное генерирование сигнала теневой передачи MCM_ST возможно лишь в случае, если бит MCMEN = 1.

Битовое поле SWEL указывает одно из событий, 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.

В случае, когда не требуется аппаратная синхронизация появления сигнала теневой передачи MCM_ST с работой таймеров SWSEL = 00b, 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 обнулится/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 (равным «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.

19.9 Режим работы с датчиками Холла

Выбирается записью 1000b 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 текущем 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 19.31 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.

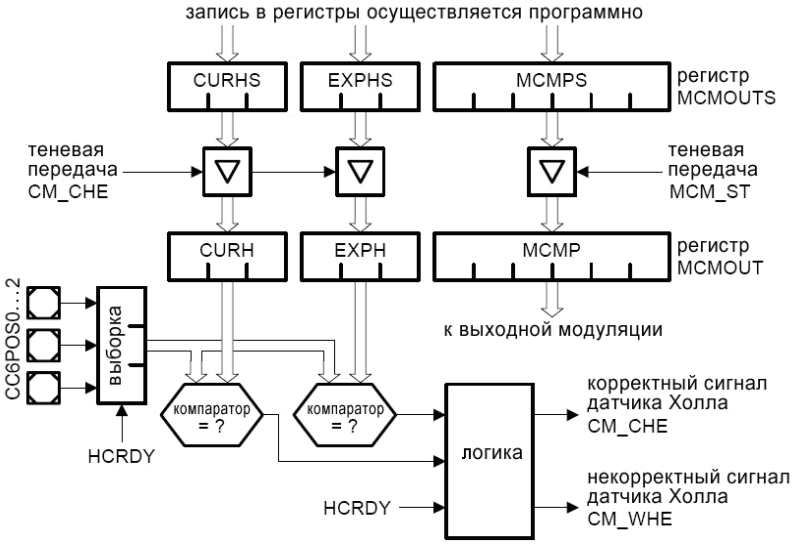


Рисунок 19.31 – Логика работы с датчиками Холла

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 19.32.

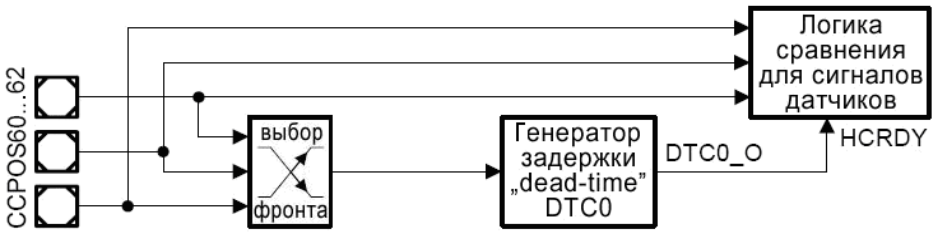


Рисунок 19.32 – Логика работы фильтра шумов

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 1000b.

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 19.33.

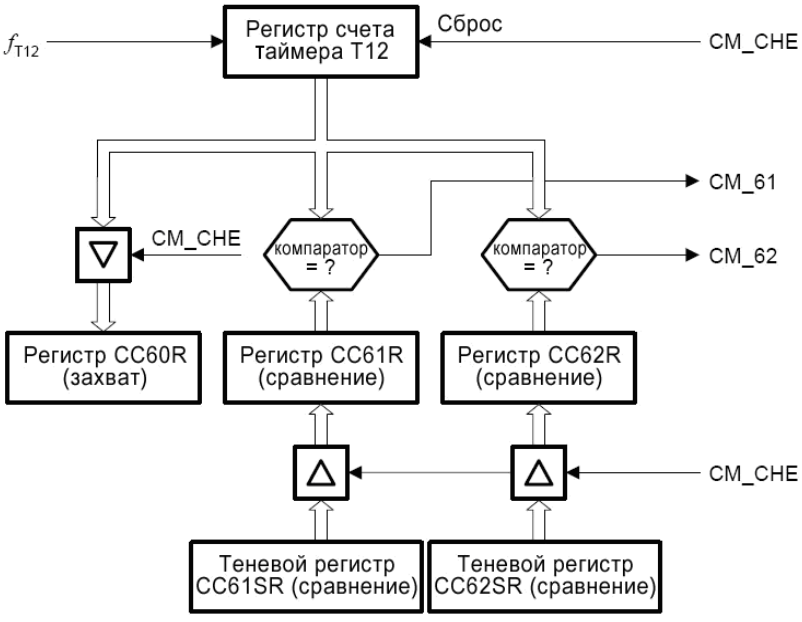


Рисунок 19.33 – Таймер T12 в режиме работы с датчиками Холла

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 0000h 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 следующим образом (см. рисунок 19.34):

1 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.

2 Отслеживание комбинаций сигналов с датчиков, которым необходима шумовая фильтрация, и управление выходной модуляцией могут быть засинхронизированы с работой источника модуляции.

3 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).

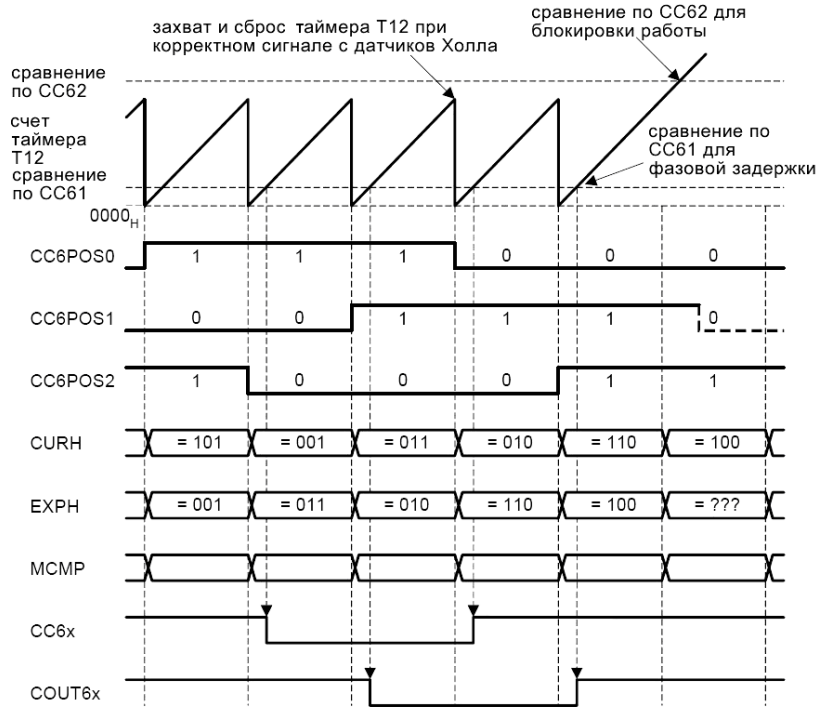


Рисунок 19.34 – Пример управления бесщеточным ДПТ (все MSEL6x = 1000b)

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 19.35.

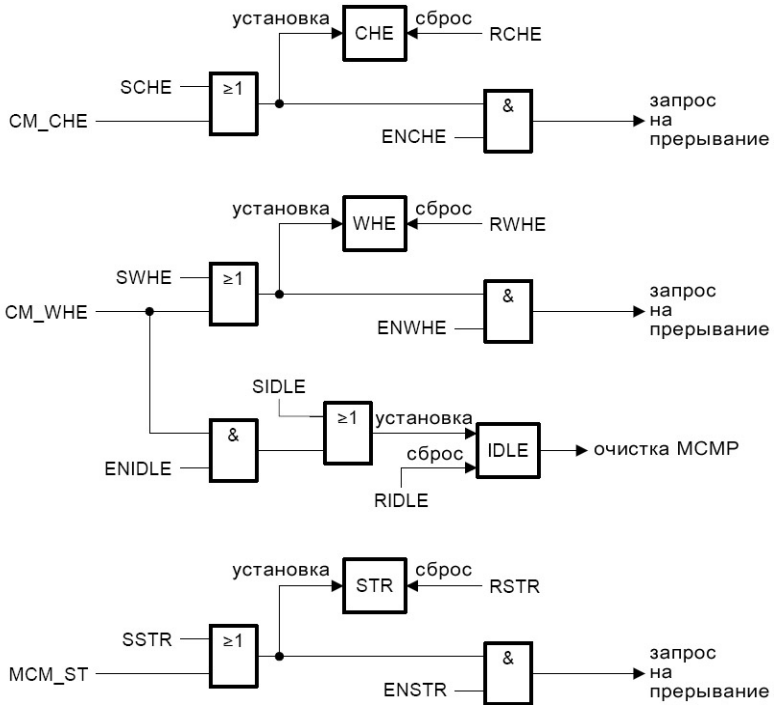


Рисунок 19.35 – Логика работы флагов в режиме датчиков Холла

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.

Регистры режима работы с датчиками Холла

Основные регистры, используемые при работе с датчиками Холла – регистр MCMOUT и его теневой регистр MCMOUTS.

19.10 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, выбранные выходные линии сигналов переводятся в пассивное состояние и на них поддерживается низкий уровень до тех пор, пока TRPS = 1. 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 становится неактивным;
- под программным контролем;
- синхронно с генерацией выходного сигнала таймерами T12 или T13.

Наилучший путь выхода из состояния ловушки – программный.

На рисунке 19.36 показана структурная схема блока обработки состояний ловушки.

Когда входной сигнал CTRAP# становится равным «0» (активный уровень), выставляется флаг TRPF, если TRPPEN = 1 (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.

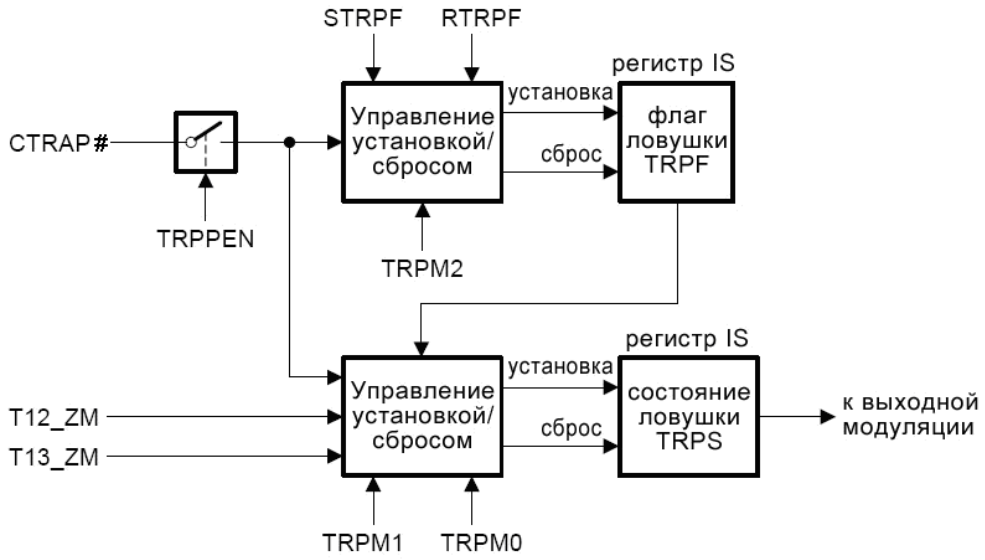


Рисунок 19.36 – Блок обработки состояния ловушки

При условии, что TRPPEN = 1 и CTRAP# = 0, 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 = 0b (см. рисунок 19.37) флаг TRPF сбрасывается аппаратно, как только CTRAP# становится неактивным, то есть равным «1». В случае если TRPM2 = 1, 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.

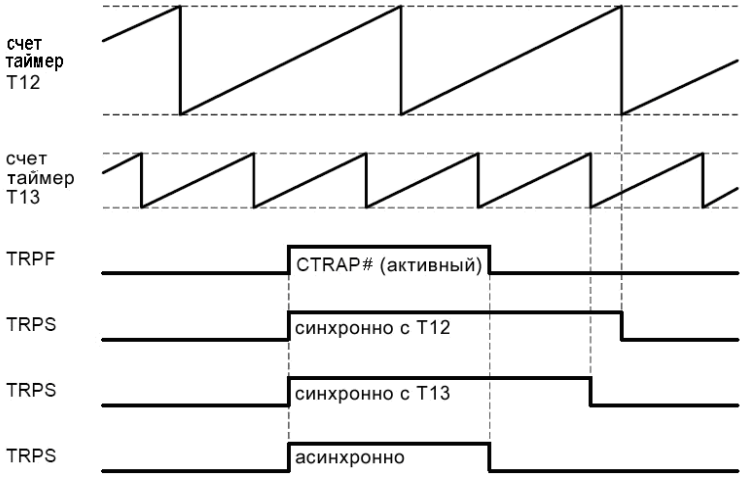


Рисунок 19.37 – Синхронизация в состоянии ловушки при TRPM2 = 0

Регистр ловушки

Регистр CCU6_TRPCTR управляет функционированием ловушки. Он содержит независимые биты, управляющие разрешением возможности попадания в ловушку каждой линии каналов выходных сигналов, и биты управления режимами ловушки.

19.11 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 19.38 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;
- модулируемый сигнал CC63_O, генерируемый в канале таймера T13, приходит с выхода логики выбора состояния;
- сигнал управления модуляцией MCMPy приходит с регистра MCMOUT в случае, если включен мультиканальный режим, бит MCMEN = 1;
- 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.

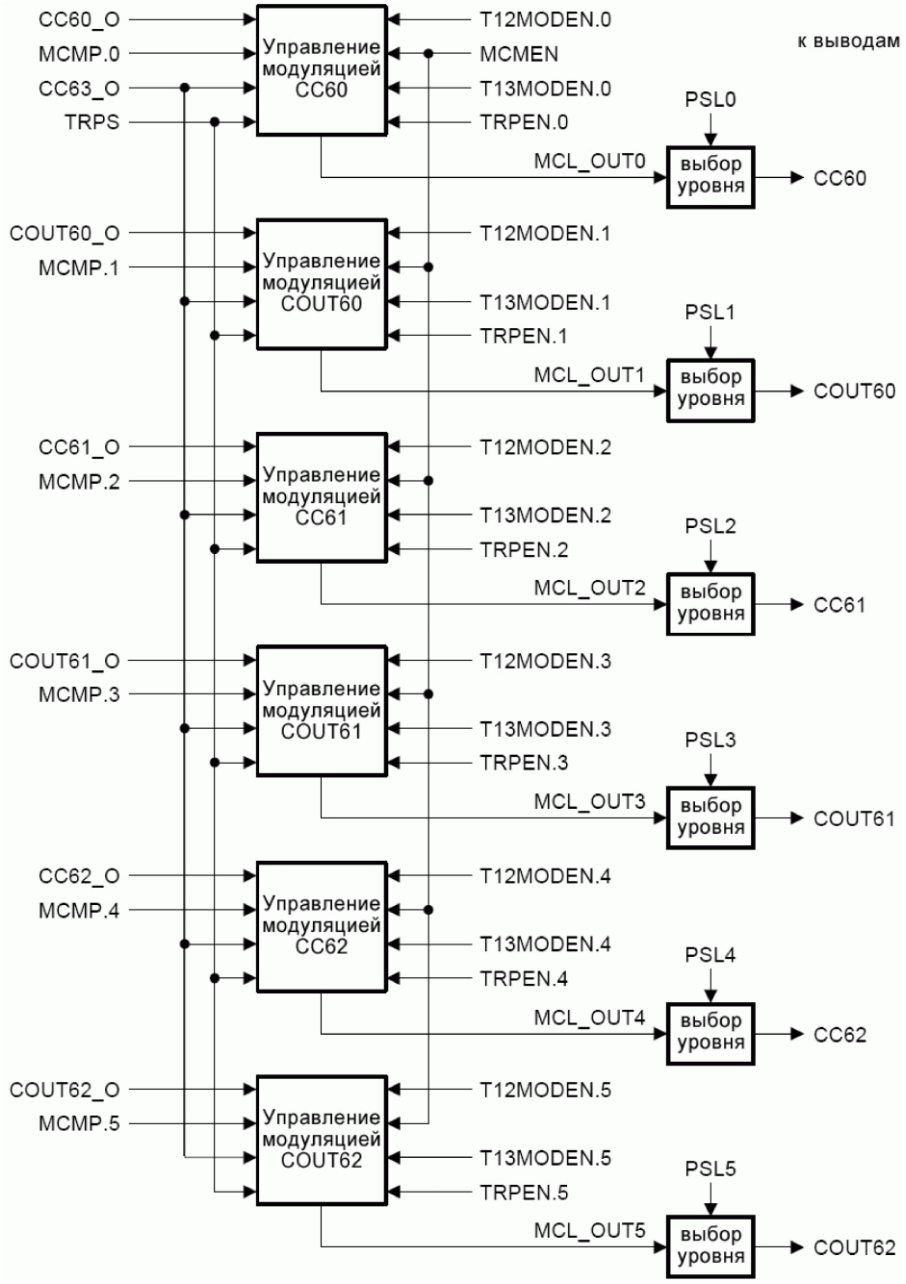


Рисунок 19.38 – Выходная модуляция

На рисунках 19.39 и 19.40 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 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.

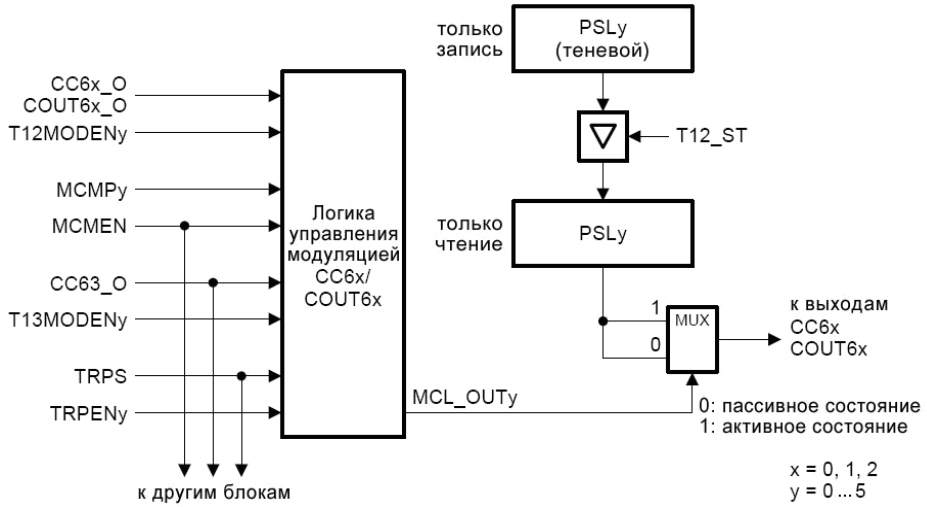


Рисунок 19.39 – Выходная модуляция в каналах таймера T12

Для канала таймера T13 построение схемы управления выводом модулируемого сигнала аналогично схеме каналов таймера T12.

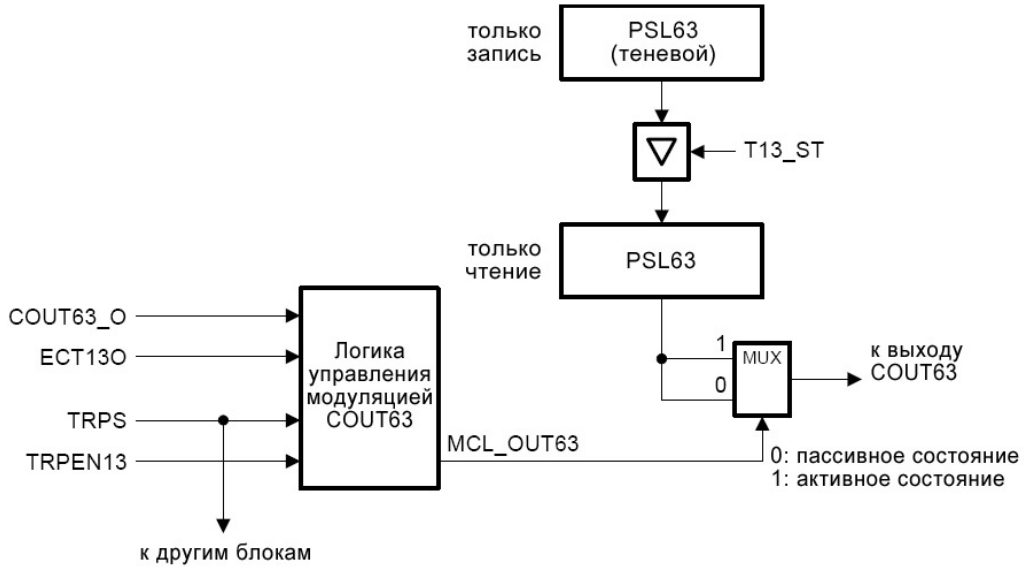


Рисунок 19.40 – Выходная модуляция в канале таймера T13

19.12 Двухрегистровые структуры и теневая передача

На рисунках 19.41 и 19.42 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.

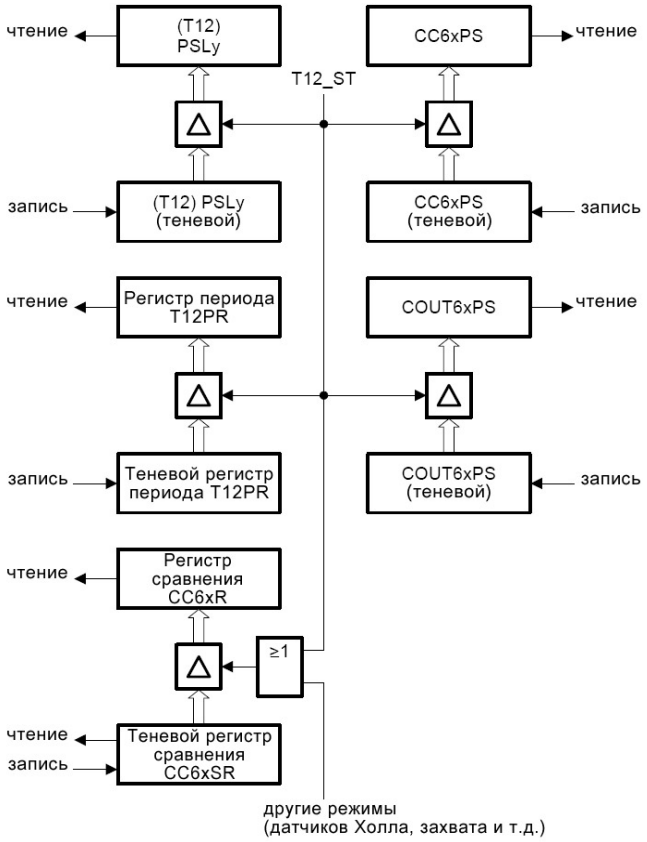


Рисунок 19.41 – Двухрегистровые структуры и сигнал теневой передачи каналов таймера T12

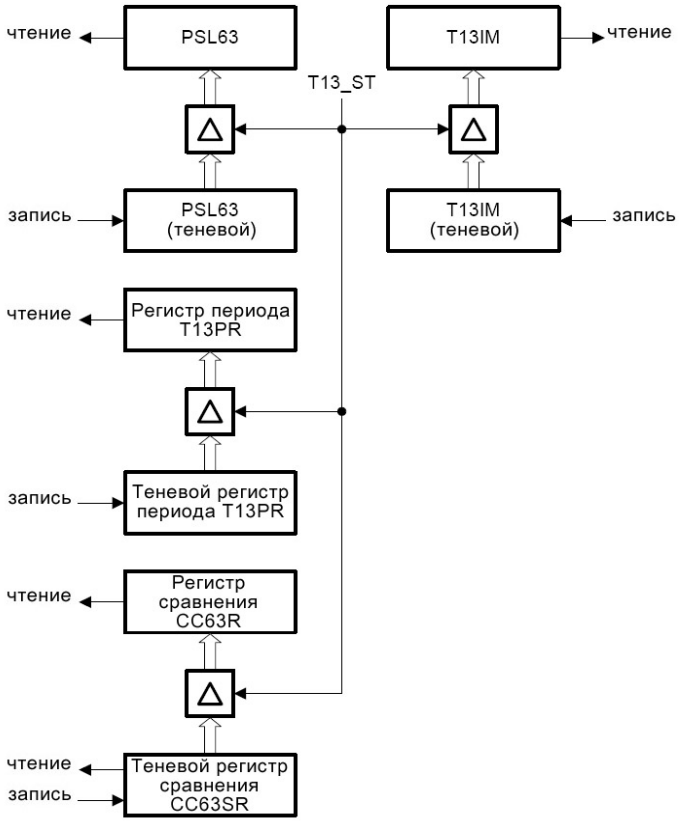


Рисунок 19.42 – Двухрегистровые структуры и сигнал теневой передачи канала таймера T13

19.13 Прерывания

Общая схема взаимодействия регистров прерываний показана на рисунке 19.43.

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.

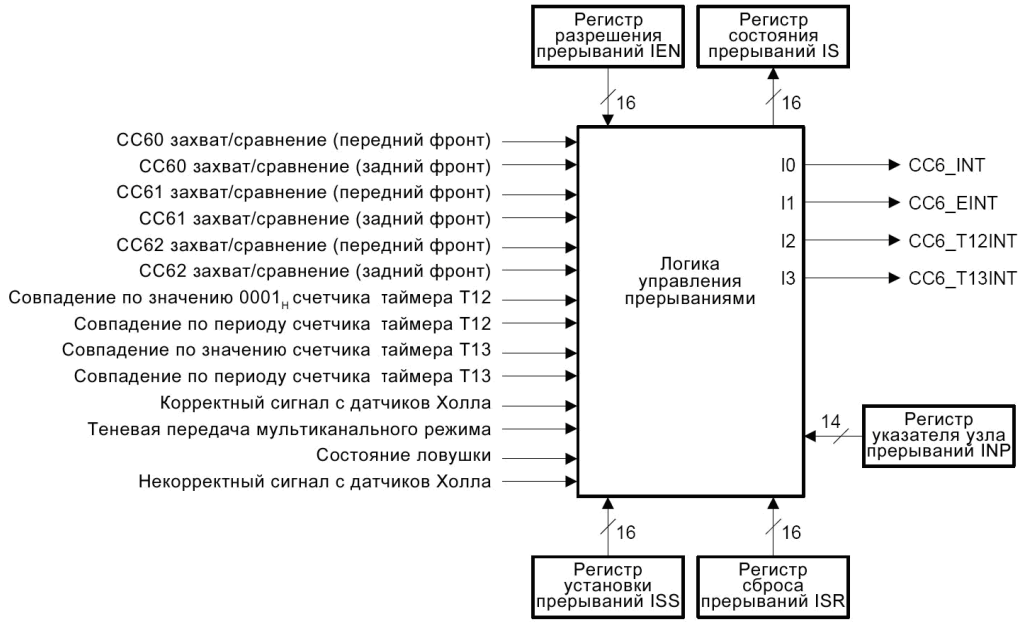


Рисунок 19.43 – Блок обработки прерываний

Подробная схема одного из семи узлов (одной из семи пар) прерываний показана на рисунке 19.44.

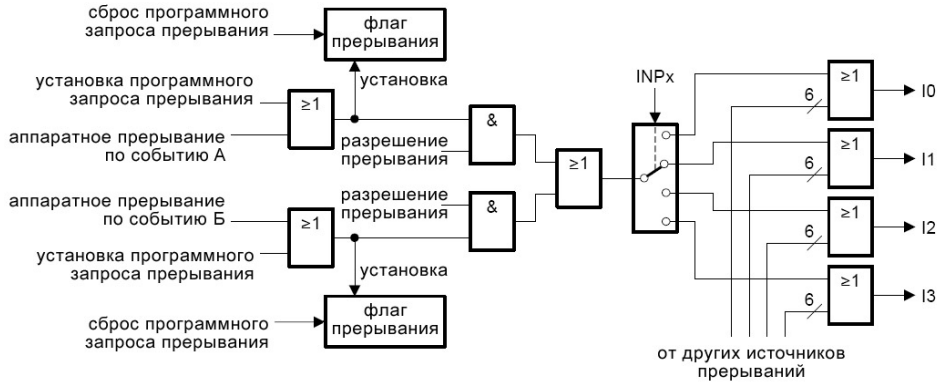


Рисунок 19.44 – Структура узла управления парой прерываний

Установка флагов прерываний в регистре IS может осуществляться как аппаратно (прерывания по событиям A и B), 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).

Примечание – В режиме сравнения (и режиме датчиков Холла) аппаратные прерывания таймера могут генерироваться только в течение работы таймера TxR = 1. 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.

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 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.

Источники прерываний модуля CAPCOM6 могут быть собраны программно в четыре узла прерываний (выходы I0, I1, I2, I3). Управляет этим регистр указателя узла прерываний INP.

Каждое битовое поле регистра INP указывает на один из четырех выходов (I0, …, I3), на который следует передавать запрос на прерывание от пары источников.

В таблице 19.2 отражены соответствия «по умолчанию» источников запросов на прерывания по узлам (выходам) и их соответствующим регистрам.

Таблица 19.2 – Соответствия «по умолчанию» источников запросов на прерывания по узлам (выходам) и их соответствующим регистрам

Источник прерывания	Выход	Управляющий регистр прерываний	Адрес регистра
Прерывания канала 0	I0	CCU6_IC	F140h
Прерывания канала 1	I0		
Прерывания канала 2	I0		
Прерывания по корректному сигналу датчика	I1	CCU6_EIC	F188h
Прерывания при возникновении ошибок	I1		
Прерывания таймера T12	I2	CCU6_T12IC	F190h
Прерывания таймера T13	I3	CCU6_T13IC	F198h

20 Отладочная система OCDS

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 20.1.

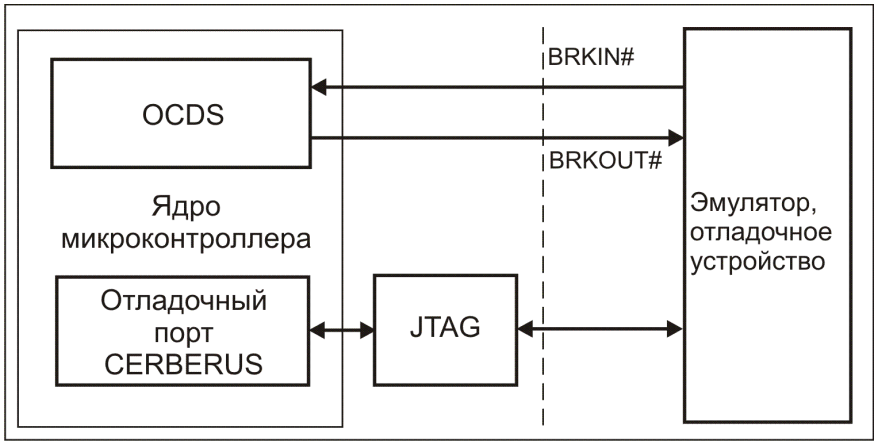


Рисунок 20.1 – Отладочная система микроконтроллера

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.

В таблицах 20.1 – 20.3 представлены все регистры отладочной системы.

Таблица 20.1 – Регистры блока OCDS

Мнемоническое название регистра	Адрес в ESFR	Назначение
DBGSR	F0FCh	Регистр состояния отладки
DEXEVT	F0F2h	Определяет действие, если выбрано внешнее событие по выводу BRKIN#
DSWEVT	F0F4h	Определяет действие, если выполняется отладочная команда
DTREVT	F0F0h	Комбинации критериев для выбора аппаратных отладочных событий
DCMPDP	F0EEh	Регистр программирования данных для регистров сравнения DCMPx
DCMPSP	F0ECh	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	F0F8h	Регистр указателя команд
DIPX	F0FAh	Регистр расширения указателя команд
DTIDR	F0D8h	Регистр ID задачи
Примечание – Регистры DCMP0, DCMP1, DCMP2, DCMPG, DCMPL доступны с помощью регистров DCMPSP и DCMPSD.		

Таблица 20.2 – Регистры блока JTAG

Мнемоника	Разряд-ность	Назначение	Сброс TRST#
1	2	3	4
BYPASS	1 бит	Стандартный регистр обхода (bypass) JTAG. Если выбрана команда BYPASS, то сигнал на TDO равен сигналу на TDI, задержанному на один цикл сигнала TCK	Xb

Окончание таблицы 20.2

1	2	3	4
CCONF	16 бит	Регистр конфигурации кристалла	0000h
ID	32 бита	Дополнительный регистр ID стандарта JTAG. Содержание ID выдается наружу, когда поле INSTRUCTION содержит команду IDCODE	–
INSTRUCTION (IR)	8 бит	Регистр команд стандарта JTAG. В отличие от остальных регистров, он устанавливается во время состояния «IR scan»	04h
IOPATH	2 бита	Выбор блока CERBERUS	00b

Таблица 20.3 – Регистры блока CERBERUS

Регистр	Разряд-ность	Адрес	Описание	Сброс JTAG	Сброс микроконтроллера
CLIENT_ID	16 бит	–	ID код клиента	Аппаратно закодировано	Аппаратно закодировано
IOADDR	24 бита	–	Адрес для доступа к следующему режиму чтения-записи	000000h	Не изменяется
IOCONF	8 бит	–	Регистр конфигурации	00h	Не изменяется
IOINFO	16 бит	–	Регистр анализа состояния микроконтроллера	Спецификации кристалла	Спецификации кристалла
TRADDR	4 бита	–	Адрес в режиме трассировки внешней шины	0h	Не изменяется
COMDATA	16 бит	F068h	Регистр данных режима коммуникации	Не изменяется	0000h
RWDATA	16 бит	F06Ah	Регистр данных режима чтения-записи	Не изменяется	0000h
IOSR	16 бит	F06Ch	Регистр состояния	См. таблицу 20.4	

Таблица 20.4 – Состояние регистра IOSR после сброса

Сброс JTAG	Сброс микроконтроллера
UUUU UUUU UUUUU0UUb UUUU UU00 UUUU U0UUb	0000 0000 0000 0U00b 0000 00UU 0000 0U00b
Примечание – С точки зрения программного обеспечения биты 9 и 8 отличаются таким поведением, потому что источником является сброс JTAG области регулирования (U – бит не изменяется). Только их синхронизация в триггерах типа «flip-flop» связана со сбросом микроконтроллера.	

20.1 Блок OCDS

Основная концепция работы блока состоит в обработке отладочных событий и определении действий, когда отладочное событие произошло, то есть запуск процесса отладки. Структурная схема блока OCDS приведена на рисунке 20.2.

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#.

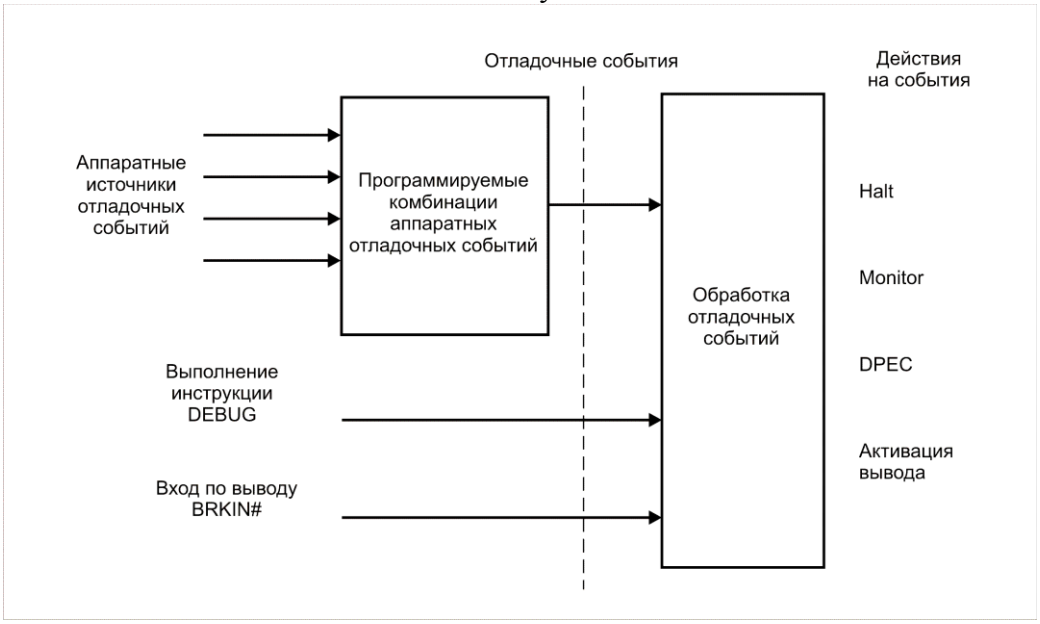


Рисунок 20.2 – Структурная схема блока OCDS

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:

а) записать F0FCh в DCMP0 (адрес DBGSR);

б) записать 2200h в DTREVT;

в) записать 0001h в DBGSR.

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

Сброс с переходом в режим останова 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:

1 Установить бит CCONF.RST_HLT до выполнения сброса.

2 Установить бит DBGSR.DEBUG_STATE для перевода в режим останова после выполнения сброса RESET.

3 Очистить бит CCONF.RST_HLT.

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

Источники отладочных событий

Приведены в таблице 20.5.

Таблица 20.5 – 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.

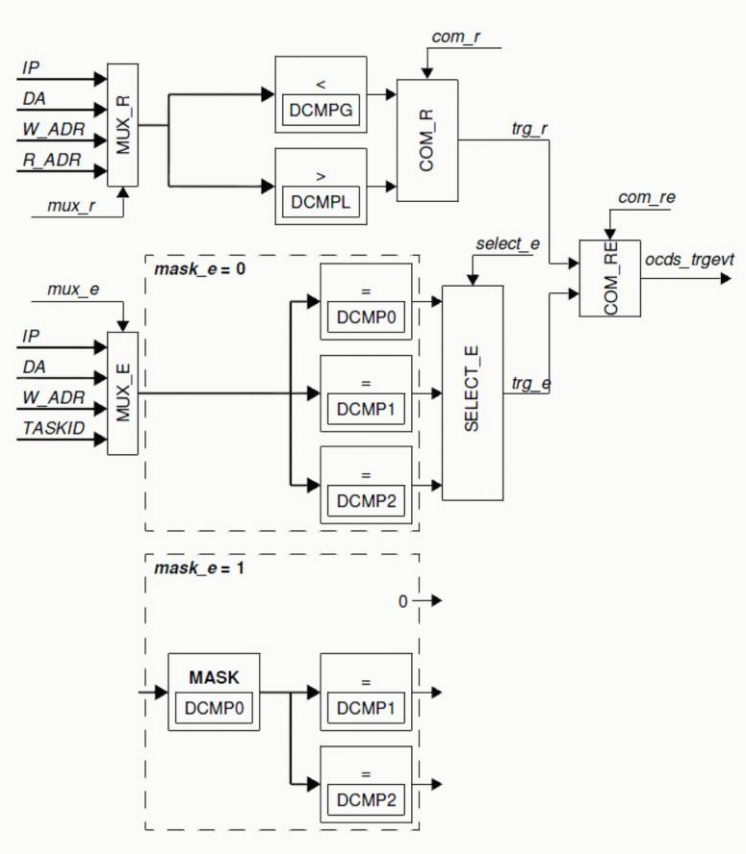


Рисунок 20.3 – Структурная схема аппаратного устройства генерации сигнала запуска отладочного события

На структурной схеме, приведенной на рисунке 20.3, часть блока, отвечающего за сравнения на равенство, может быть сконфигурирована для двух маскируемых сравнений на равенство.

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 D140h) определяет, что выполняется команда «Пользовательский режим», и ее выполнение зависит от того, разрешен ли OCDS. Если OCDS разрешен, команда DEBUG, 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 20.6.

Таблица 20.6 – 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, приведенных в таблице 20.7.

Таблица 20.7 – 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 20h) 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 20.4.

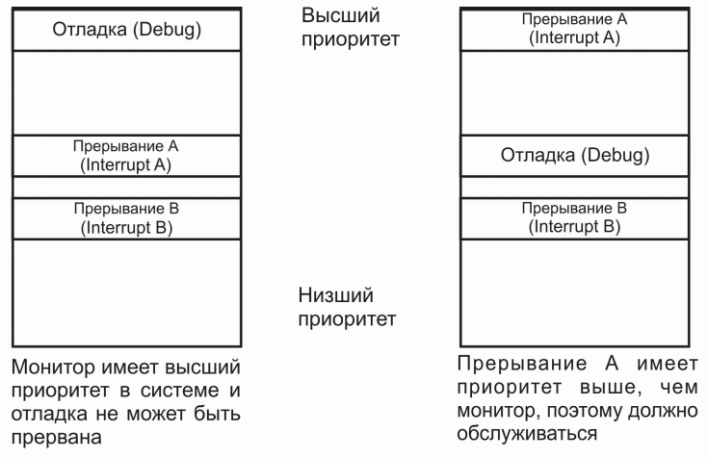


Рисунок 20.4 – Простая и расширенная модели отладки

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

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

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 = 000000h, DCMPG = 000001h и битовыми полями COM_R = 11b, COM_RE = 0b, BREAK_AFTER_MAKE = 1b 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.

Регистры управления отладочными событиями DEXEVT, DSWEVT, DTREVT

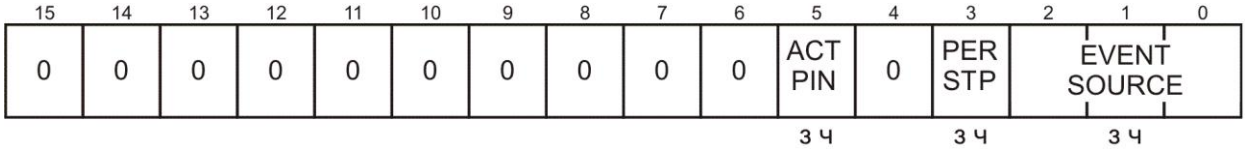
Каждый возможный источник отладочного события имеет соответствующий регистр, который определяет действие, предпринятое в случае появления отладочного события (см. таблицы 20.8 и 20.9). Регистры управления отладочными событиями имеют одинаковую структуру для всех определяемых текущих источников.

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.

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.

Таблица 20.8 – Регистры управления отладочными событиями вывода BREAK и программного обеспечения

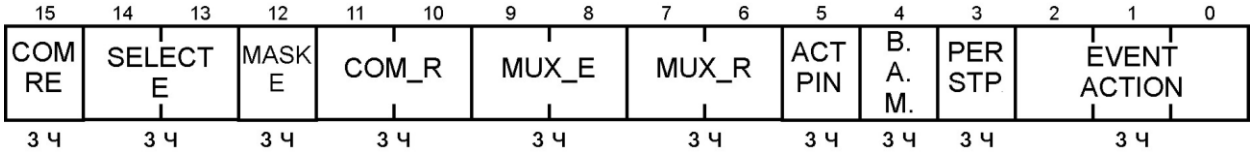
DEXEVT XSFR (F0F2h) Сброс: 0000h
DSWEVT XSFR (F0F4h)



Поле	Биты	Описание	
ACTIVATE_PIN	5	Активность внешнего вывода	
		0	Неактивен
		1	Активен в течение отладочного события
PERIPHERALS_STOP	3	Бит управления приостановкой периферии	
		0	Событие не оказывает влияние на периферийные устройства
		1	Периферийные устройства приостанавливают операции в случае события
EVENT_SOURCE	2–0	Действие в ответ на отладочное событие	
		000	Нет действий
		001	Режим программной отладки
		010	Остановочный отладочный режим Halt
		011	Зарезервировано
		100	Зарезервировано
		101	Выполнение DPEC
		110	Зарезервировано
		111	Установка бита источника в DBGSR
0	15–6, 4	Зарезервировано	

Таблица 20.9 – Регистр управления комбинациями аппаратных отладочных событий

DTREVT XSFR (F0F0h) Сброс: 0000h



Поле	Биты	Описание	
1	2	3	
COM_RE	15	Комбинация сравнения по равенству и диапазону:	
		0	Сигнал ocds_trgevt формируется логической функцией (trg_r | trg_e)
		1	ocds_trgevt формируется логической функцией (trg_r & 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

Окончание таблицы 20.9

1	2	3	
SELECT_E	14–13	Выбор сравнения на равенство (см. таблицу 20.10)	
MASK_E	12	Бит включения маски	
		0	Немаскированное сравнение на равенство
		1	Маскированное сравнение на равенство
COM_R	11–10	Выбор диапазона сравнения (см. таблицу 20.11)	
MUX_E	9–8	Управление входным мультиплексором сравнения на равенство	
		00	Указатель команд IP
		01	Значение данных DA
		10	Адрес записи W_ADR
		11	ID задачи TASKID
MUX_R	7–6	Область сравнения входного мультиплексора	
		00	Указатель команд IP
		01	Значение данных DA
		10	Адрес записи W_ADR
		11	Адрес чтения R_ADR
ACTIVATE_PIN	5	Идентично полю ACTIVATE_PIN регистра DEXEVT	
BREAK_AFTER_ MAKE	4	Бит управления остановом	
		0	Остановка до выполнения (только IP)
		1	Остановка после выполнения (только IP)
PERIPHERALS_ STOP	3	Идентично полю PERIPHERALS_STOP регистра DEXEVT	
EVENT_ ACTION	2–0	Идентично полю EVENT_SOURCE регистра DEXEVT	

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

Таблица 20.10 – Формирование сигнала trg_e

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

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

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.

Таблица 20.11 – Формирование сигнала trg_r

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

Регистр состояния отладки DBGSR

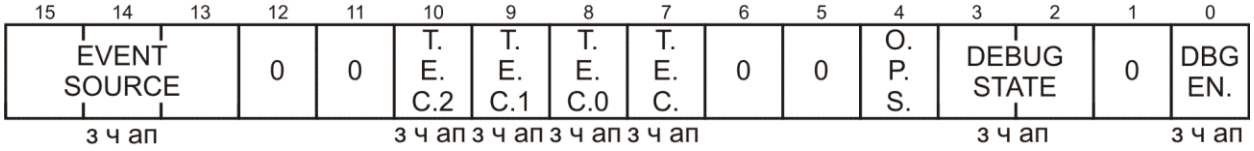
Регистр содержит некоторую информацию о текущем состоянии OCDS, включающую:

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

Описание регистра приведено в таблице 20.12

Таблица 20.12 – Регистр состояния отладки

DBGSR XSFR (F0FCh) Сброс: 0000h



Поле	Биты	Описание	
1	2	3	
EVENT_ SOURCE	15–13	Источник сообщения о последнем отладочном событии	
		xx1	Внешний вывод приостановки DEXEVT
		x1x	Выполняется команда DEBUG (DSEWT)
		1xx	Комбинация аппаратных источников событий DTREVT
TRGEVT_E_CMP2	10	Второй флаг сравнения	
		0	Сравнение 2 на равенство не соответствует
		1	Сравнение соответствует для текущего события
TRGEVT_E_CMP1	9	Первый флаг сравнения	
		0	Сравнение 1 на равенство не соответствует
		1	Сравнение соответствует для текущего события
TRGEVT_E_CMP0	8	Нулевой флаг сравнения	
		0	Сравнение 0 на равенство не соответствует
		1	Сравнение соответствует для текущего события
TRGEVT_E_CMP	7	Флаг сравнения	
		0	Диапазон сравнения не соответствует
		1	Сравнение соответствует для текущего события
OCDS_P_SUSPEND	4	Флаг приостановки	
		0	Нет действий
		1	Соответствующая периферия приостанавливает свои операции

Окончание таблицы 20.12

1	2	3	
DEBUG_STATE	3, 2	Текущее состояние отладки	
		00	Пользовательский режим
		01	Программный отладочный режим
		10	Остановочный отладочный режим (Halt)
		11	Зарезервировано
DEBUG_ENABLED	0	Бит разрешения отладки	
		0	Запрещено
		1	Разрешено
0	12, 11, 6, 5, 1	Зарезервировано	

Биты EVENT_SOURCE устанавливаются независимо от поля EVENT_ACTION в соответствующем регистре управления отладочным событием, за исключением значения 000b. 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.

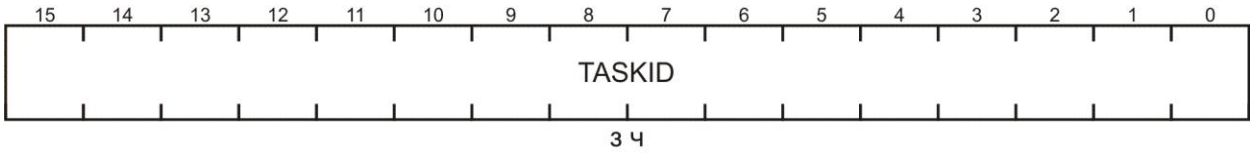
Бит OCDS_P_SUSPEND управляет сигналом, останавливающим периферию. Если он установлен в «1», то соответствующая периферия приостанавливается. Этот бит устанавливается в ходе отладочного события в соответствии с битом PERIPHERALS_STOP в активном регистре управления отладочным событием. Этот бит должен 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.

Регистр ID задачи DTIDR

Описание регистра приведено в таблице 20.13

Таблица 20.13 – Регистр ID задачи

DTIDR XSFR (F0D8h) Сброс: 0000h



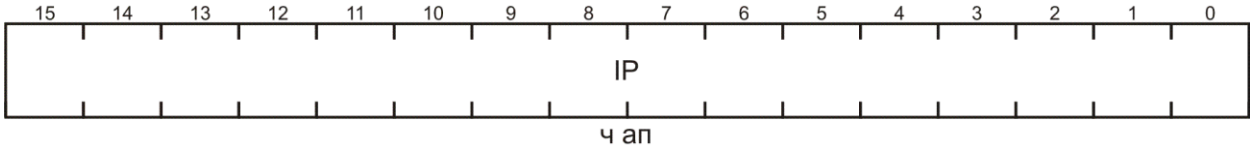
Поле	Биты	Описание
TASKID	15–0	Входные данные на аппаратное устройство генерации событий. Предназначено для использования в передовых системах реального времени для запоминания ID активной задачи

Регистры указания команд DIP и DIPX

Регистры (см. таблицы 20.14 и 20.15) предусмотрены для того, чтобы сделать видимым регистр указатель команд IP, когда CPU находится в режиме Halt.

Таблица 20.14 – Регистр указатель команд

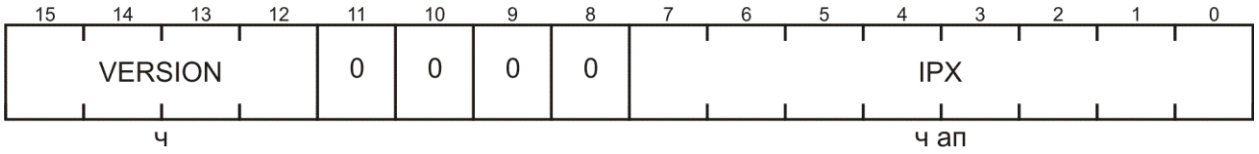
DIP XSFR (F0F8h) Сброс: 0000h



Поле	Биты	Описание
IP	15–0	Биты 15–0 текущего указателя команд в режиме Halt. Значение верно только в режиме Halt

Таблица 20.15 – Регистр расширения указателя команд

DIPX XSFR (F0FAh) Сброс: 0000h



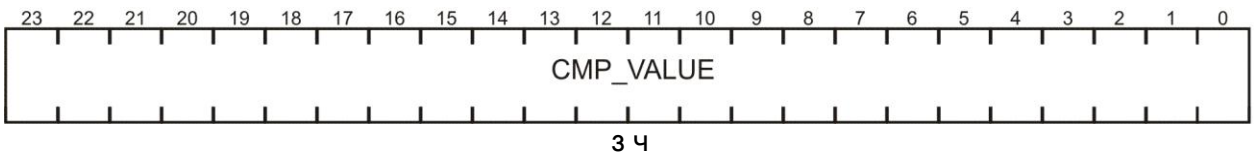
Поле	Биты	Описание
VERSION	15–12	Аппаратно закодировано как 0011b
IPX	7–0	Биты 23–16 расширенной части текущего указателя команд (CSP) в режиме Halt
0	11–8	Зарезервировано

Регистры аппаратного генератора сигнала запуска отладочного события

Регистры DCMP0, DCMP1, DCMP2, DCMPG, DCMPL (см. таблицу 20.16) 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 (см. таблицы 20.17 и 20.18). Поле SELECT_DCMP регистра DCMPSP выбирает сравниваемый регистр и записывает его старший байт. Младшие 16 бит могут быть доступны для записи с помощью регистра DCMPDP.

Таблица 20.16 – Регистры сравнения аппаратных событий

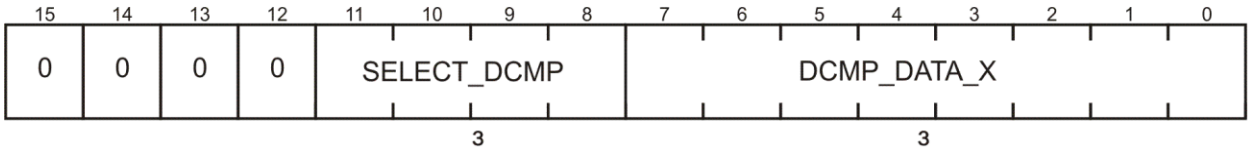
DCMP0, DCMP1, DCMP2 Сброс: 0000h
DCMPG, DCMPL



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

Таблица 20.17 – Регистр выбора и программирования DCMPx

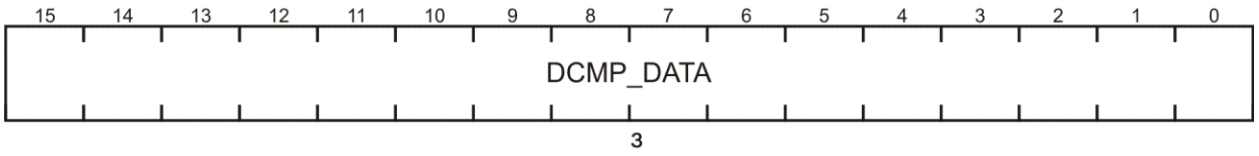
DCMPSP XSFR (F0ECh) Сброс: 0000h



Поле	Биты	Описание	
SELECT_DCMP	11–8	Выбор регистров сравнения	
		0000	Выбор DCMP0
		0001	Выбор DCMP1
		0010	Выбор DCMP2
		0011	Зарезервировано
		0100	Выбор DCMPL
		0101	Выбор DCMPG
		0110-1111	Зарезервировано
DCMP_DATA_X	7–0	Устанавливает биты 23–16 выбранного (SELECT_DCMP) DCMP регистра	

Таблица 20.18 – Регистр программирования данных для DCMPx

DCMPDP XSFR (F0EEh) Сброс: 0000h



Поле	Биты	Описание
DCMP_DATA	15–0	Устанавливает биты 15–0 выбранного SELECT_DCMP

Общие положения при обращении к регистрам OCDS

Функциями OCDS в основном управляют записи в регистре 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, выполняемыми 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 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, достаточно 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.

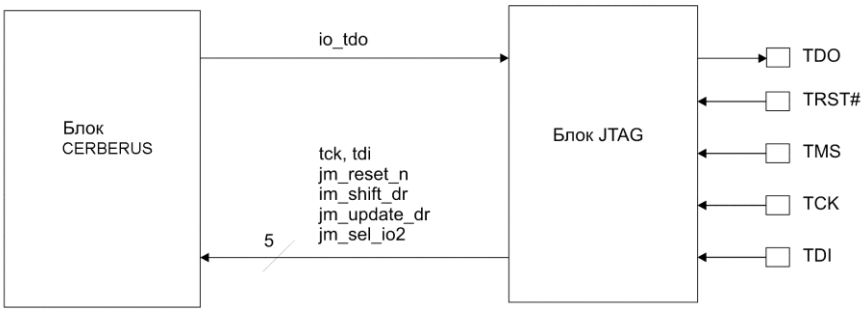


Рисунок 20.5 – Соединение блоков JTAG, CERBERUS, порта JTAG

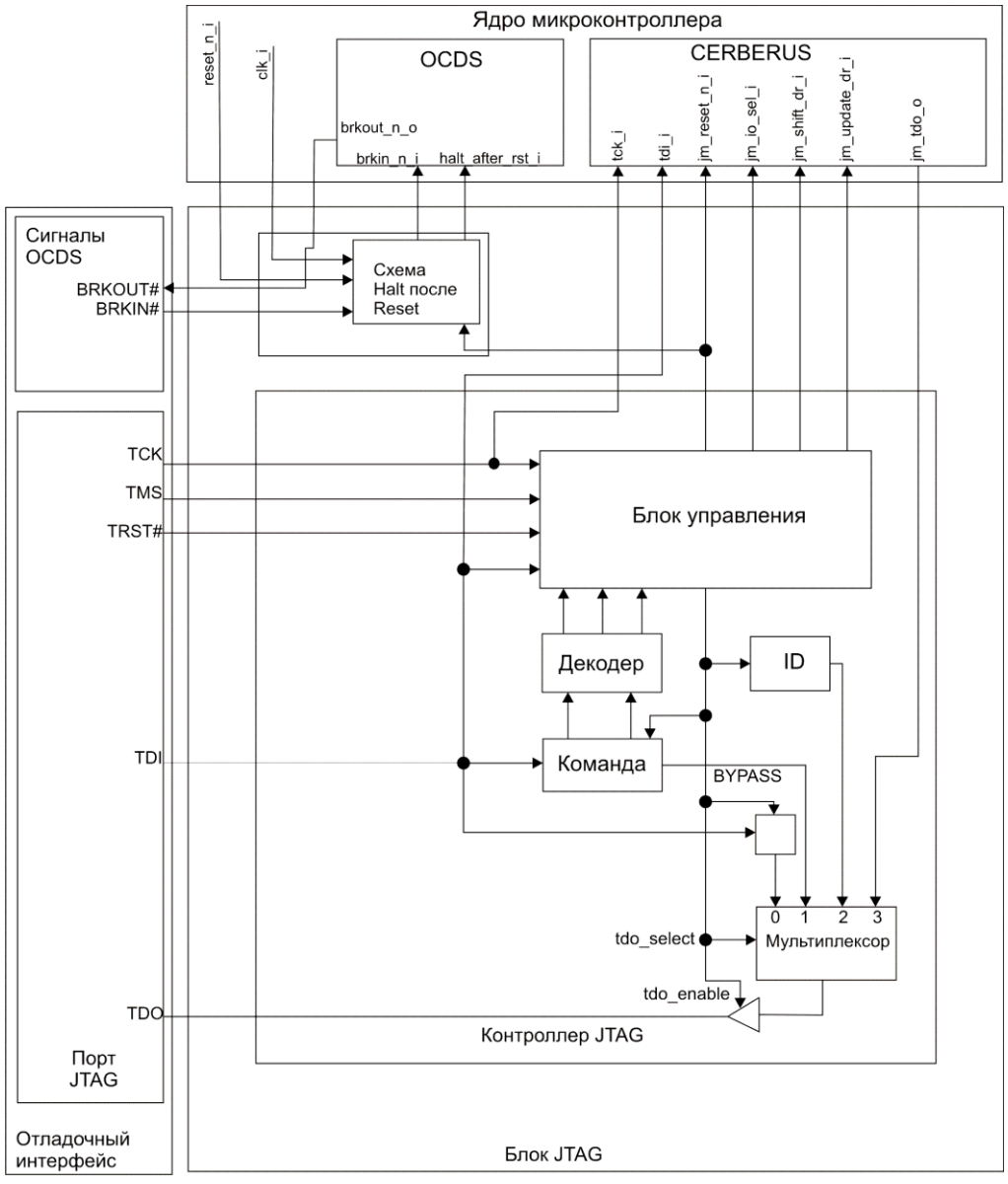


Рисунок 20.6 – Структурная схема блока JTAG

20.2 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 20.5 изображено межсоединение блоков JTAG, CERBERUS, порта JTAG. На рисунке 20.6 показана структурная схема блока JTAG.

Схема состояний контроллера JTAG

Схема состояний контроллера JTAG (JTAG Controller State Machine IEEE.1149) является сердцем блока JTAG. Это также относится к контроллеру порта тестового доступа TAP. Все переключения состояний происходят по положительному фронту, управляемому 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 20.7 приведена схема состояний контроллера JTAG.

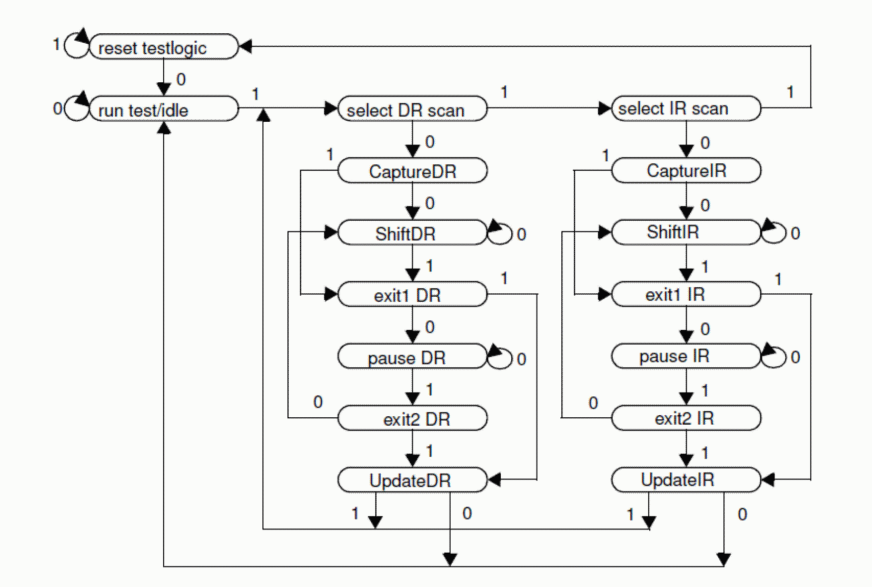


Рисунок 20.7 – Схема состояний контроллера JTAG

Все сигналы, приведенные на рисунке 20.7, 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 20.7, 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 20.7, 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 20.19.

Таблица 20.19 – Команды модуля JTAG в течение состояний «IR scan»

Код операции	Диапазон	Тип	Команда
00000000b-00001111b	00 – 0Fh, 16 команд	Зарезервировано	–
00010000b	10h	Конфигурация схемы	CCONF_SET
00010001b-10111111b	11h – BFh, 174 команды	Зарезервировано	–
11000000b	C0h	Режим чтения-записи JTAG	JTAG_IO_SELECT_PATH
11000001b	C1h		JTAG_IO_INSTRUCTION1
11000010b-11111110b	C2h – FEh, 60 команд	Зарезервировано	–
11111111b	FFh	IEEE1149	BYPASS

Регистры блока JTAG

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

Регистр BYPASS

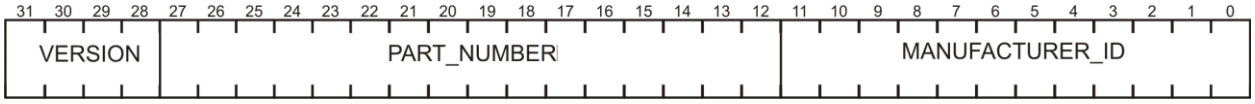
Это обязательный однобитный JTAG регистр. Если происходит выбор, то сигнал на выходе TDO равен сигналу на входе TDI, задержанный на один цикл сигнала TCK.

Регистр ID

Регистр 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 в таблице 20.20.

Таблица 20.20 – Регистр идентификации

ID (формат команды IDCODE) Сброс: UUUUUUUUh



Поле	Биты	Описание
VERSION	31–28	Версия кристалла
PART_NUMBER	27–12	Номер кристалла
MANUFACTURER_ID	11–0	Производственный номер

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 20.8, 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.

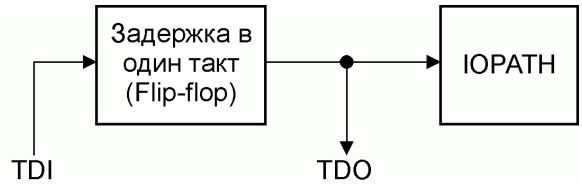


Рисунок 20.8 – Регистр IOPATH

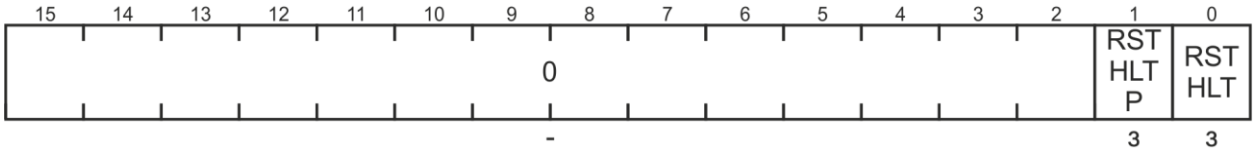
Регистр IOPATH используется для выбора блока CERBERUS. Если команда JTAG находится в диапазоне адресов и не равна C0h, то соответствующий сигнал выбора активен. Регистр IOPATH имеет ширину 2 бита и установлен командой JTAG_IO_SELECT_PATH как регистр регулярного сканирования. Для выбора CERBERUS он должен быть установлен как 10b (это рекомендуется устанавливать по умолчанию для аппаратных средств).

Регистр CCONF

Регистр предусмотрен для конфигурирования специальных состояний микроконтроллера. Это можно рассматривать как альтернативный механизм перезагрузки конфигурации. Описание регистра приведено в таблице 20.21. 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 подачей команды CCONF_SET и ведет себя так же, как и регистр IOPATH.

Таблица 20.21 – Регистр конфигурирования

CCONF Сброс: 0000h



Поле	Биты	Описание	
RST_HLT_P	1	Бит разрешения изменения состояния бита RST_HLT	
		0	Запрещено
		1	Разрешено
RST_HLT	0	Включение режима Halt после сброса	
		0	Нет действий
		1	Режим включен после сброса
–	15–2	Зарезервировано	

Инициализация блока CERBERUS

Используемые в микроконтроллере коды команд, приведенные в таблице 20.22, заносятся в регистр INSTRUCTION, когда JTAG модуль находится в состоянии «IR scan».

Таблица 20.22 – Команды модуля JTAG

Код команды	Команда	Описание
10h	CCONF_SET	Выбирает регистр CCONF
C0h	JTAG_IO_SELECT_PATH	Выбирает регистр IOPATH
$C1_H$	JTAG_IO_INSTRUCTION1	Выбирает модуль CERBERUS
FFh	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 (10h). DR scan: загрузка 0003h в регистр CCONF для останова после сброса, иначе 0000h.

На вход TDI надо послать 16 бит младшим битом вперед.

3 Загрузка регистра IOPATH. «IR scan»: загрузка команды JTAG_IO_SELECT_PATH (C0h). DR scan: загрузка 10b 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 (C1H). После выполнения этих операций блок CERBERUS готов к работе.

20.3 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 20.9.

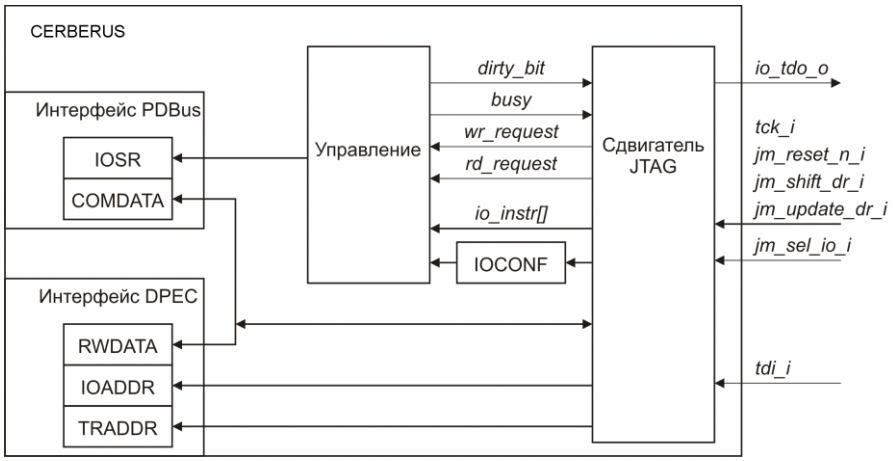


Рисунок 20.9 – Структурная схема блока CERBERUS

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 20.10 приведен синтаксис битового потока 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 не выполняема, выдается неопределенное число битов занятости.

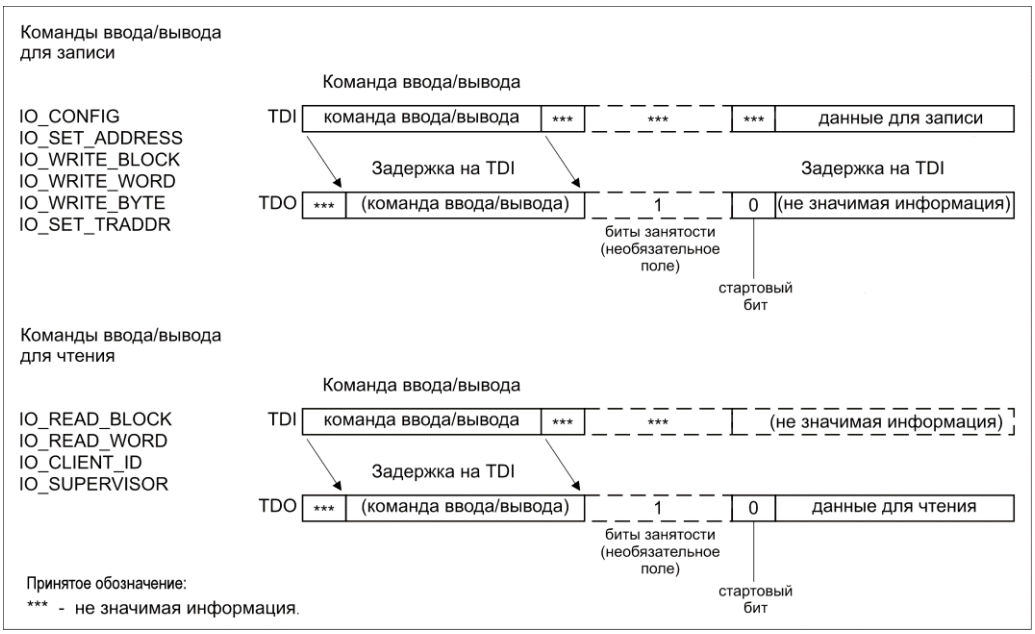


Рисунок 20.10 – Синтаксис последовательной передачи для выводов TDI и TDO в состоянии ShiftDR

Команды чтения-записи блока CERBERUS

В таблице 20.23 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 20.23 – Команды чтения-записи блока CERBERUS

Команда	Код	Тип	Описание
IO_CONFIG	0h	Запись	Устанавливает регистр конфигурации IOCONF
IO_SET_ADDRESS	1h	Запись	Устанавливает регистр адреса IOADDR
IO_WRITE_BLOCK	2h	Запись	Старт записи блока данных, начиная с адреса в IOADDR
IO_READ_BLOCK	3h	Чтение	Старт чтения блока данных, начиная с адреса в IOADDR
IO_WRITE_WORD	4h	Запись	Режим чтения-записи: запись слова Режим коммуникации: передача слова
IO_READ_WORD	5h	Чтение	Режим чтения-записи: чтение слова Режим коммуникации: запрос слова
Зарезервировано	7h-6h	–	Не использовать
IO_WRITE_BYTE	8h	Запись	Режим чтения-записи: запись байта Режим коммуникации: зарезервировано
Зарезервировано	9h	–	–
IO_SET_TRADDR	Ah	Запись	Установка регистра TRADDR
IO_SUPERVISOR	Bh	Чтение	Подтверждение сброса и анализ состояния захвата шины
Зарезервировано	Eh-Ch	–	–
IO_CLIENT_ID	Fh	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 (также 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 байт. 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 20.11 показана взаимосвязь выводов TDI, TDO и содержания сдвигового регистра блока CERBERUS после ввода клиентской команды.

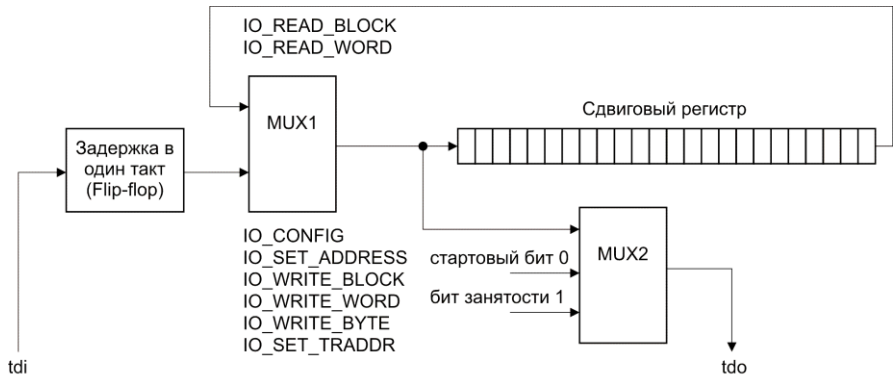


Рисунок 20.11 – Сдвиговый регистр в состоянии ShiftDR

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).

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 01h RWDATA 0000h IOADDR 000000h
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 01h RWDATA 0000h IOADDR 000033h
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 01h RWDATA 0000h IOADDR 000033h
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 01h RWDATA 1234h IOADDR 000033h
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 01h RWDATA 1234h IOADDR 000033h
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**

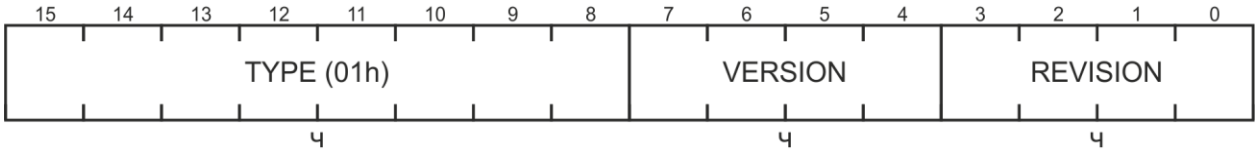
2 IO_READ_WORD
IOCONF 01h RWDATA 1234h IOADDR 000033h
TDI: **1010**11111111111111111111111111111111111111111111111111
TDO: 0**1010**1110**0010110001001000**xxxxxxxx**0010110001001000**xxxxxxxx.

Регистр CLIENT_ID

Регистр позволяет внешнему отладочному устройству проверять аппаратные средства в автоконфигурационном режиме. Описание регистра приведено в таблице 20.24.

Таблица 20.24 – Регистр идентификации типа клиента

CLIENT_ID Сброс: 0121h



Поле	Биты	Описание
TYPE	15–8	Тип. Зарезервированное значение – 01h
VERSION	7–4	Версия. Зарезервированное значение – 2h
REVISION	3–0	Ревизия. Зарезервированное значение – 1h

Регистр IOADDR

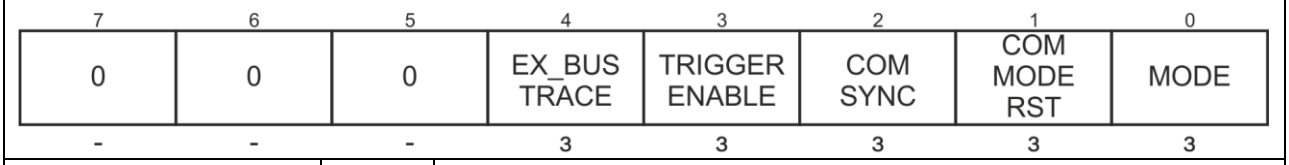
Регистр захватывает 24-битный адрес для следующего обращения к блоку CERBERUS. Регистр IOADDR обновляется в состоянии UpdateDR содержимым сдвигового регистра, когда команда IO_SET_ADDRESS активна или увеличивается на два (16-битное слово), если были выполнены команды IO_READ_BLOCK или IO_WRITE_BLOCK.

Регистр IOCONF

Регистр используется для конфигурации блока CERBERUS и записывается со стороны внешнего интерфейса (не доступен со стороны CPU). Описание регистра приведено в таблице 20.25.

Таблица 20.25 – Регистр конфигурации

IOCONF Сброс: 00h



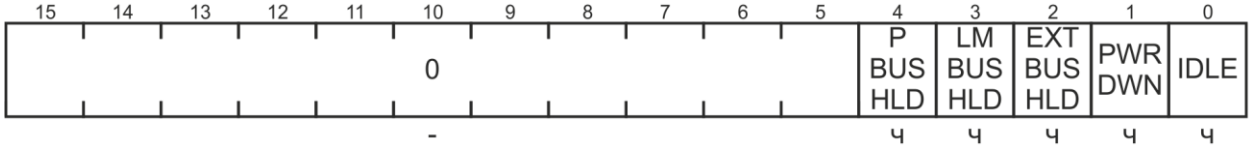
Поле	Биты	Описание	
1	2	3	
EX_BUS_TRACE	4	Бит разрешает включение приема-передачи по адресам внешней шины	
TRIGGER_ENABLE	3	Бит разрешает включение приема-передачи в режиме чтения-записи	
COM_SYNC	2	Запись единицы устанавливает бит COMSYNC в регистре IOSR	
COM_MODE_RST	1	Запись единицы сбрасывает биты CRSYNC и CWSYNC в регистре IOSR для прекращения запросов в коммуникационном режиме. Этот сброс нестатический, он выполняется только один раз, когда обновляется регистр IOCONF	
MODE	0	Выбор режима блока CERBERUS	
		0	Режим коммуникации
		1	Режим чтения-записи
–	7-5	Зарезервировано	

Регистр IOINFO

Регистр предусмотрен, чтобы анализировать ситуации, блокирующие передачу, или определять другие ошибочные состояния микроконтроллера. Это не физический регистр, но он предоставляет определенную информацию о состоянии микроконтроллера (см. таблицу 20.26).

Таблица 20.26 – Регистр состояния для анализа ошибок

IOINFO Сброс: 0000h



Поле	Биты	Описание
P_BUS_HLD	4	Шина PBus занята
LM_BUS_HLD	3	Не используется
EXT_BUS_HLD	2	Внешняя/XBus-шина занята
PWR_DWN	1	Контроллер в режиме PowerDown
IDLE	0	Контроллер в режиме Idle
–	15–5	Зарезервировано

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.

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

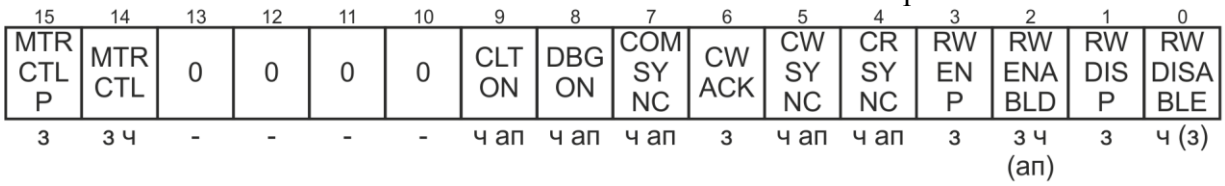
Регистр RWDATA является регистром данных для обоих типов передач: чтения и записи в режиме чтения-записи (RW Mode). Регистр COMDATA является эквивалентом для режима коммуникации (Communication Mode).

Регистр IOSR

Регистр 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 (см. таблицу 20.27).

Таблица 20.27 – Регистр управления и состояния

IOSR Сброс: 000000UU 00000U0b



Поле	Биты	Описание	
1	2	3	
MTR_CTL_P	15	Бит разрешения изменения состояния бита MTR_CTL	
		0	Запрещено
		1	Разрешено
MTR_CTL	14	Бит разрешения трассировки памяти	
CLT_ON	9	Индикатор выбора блок CERBERUS	
		0	Не выбран
		1	Выбран
DBG_ON	8	Индикатор наличия внешнего отладчика	
		0	Отсутствует
		1	Присутствует

Окончание таблицы 20.27

COM_SYNC	7	Индикатор состояния бита COMSYNC регистра IOCONF в коммуникационном режиме	
		0	Сброшен
		1	Установлен
CW_ACK	6	Отклик на запрос записи в коммуникационном режиме	
		0	Нет действий
		1	Сообщение о том, что отправленное значение было прочитано монитором из COMDATA
CW_SYNC	5	Бит записи в коммуникационном режиме	
		0	Данные не действительны или монитор (CPU) читает COMDATA
		1	Внешний интерфейс запрашивает у монитора чтение данных из COMDATA
CR_SYNC	4	Бит чтения в коммуникационном режиме	
		0	Нет запроса на чтение
		1	Внешний интерфейс запрашивает у монитора запись данных в COMDATA
RW_EN_P	3	Бит разрешения изменения состояния бита RW_ENABLED	
		0	Запрещено
		1	Разрешено
RW_ENABLED	2	Бит используется пользовательской программой. Сбрасывается только блоком JTAG. ЦПУ не может сбросить этот бит	
RW_DISP	1	Бит разрешения изменения состояния бита RW_DISABLE	
		0	Запрещено
		1	Разрешено
RW_DISABLE	0	Бит запрета режима чтения-записи	
		0	Режим разрешен
		1	Режим запрещен
–	13–10	Зарезервировано	

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.

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 20.4 «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.

20.4 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 байт 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 регистра RWDATA или в него. 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 из COMDATA или в него. 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 транзакции 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.

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.

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

$$NINSTR = 30/(TINSTR \times fjtag), \quad (20.1)$$

где NINSTR – число циклов команд, которые необходимы между двумя обращениями CPU к определенным областям памяти (memory location);

TINSTR – время цикла команды CPU, нс;

fjtag – тактовая частота интерфейса JTAG (TCK), МГц.

Например, если TINSTR = 100 нс и fjtag = 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 20.28.

Таблица 20.28 – 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 = 0b, TRIGGER_ENABLE = = 1b, EX_BUS_TRACE = 1b (все в регистре IOCONF). Адрес по внешней шине имеет формат, приведенный на рисунке 20.12.

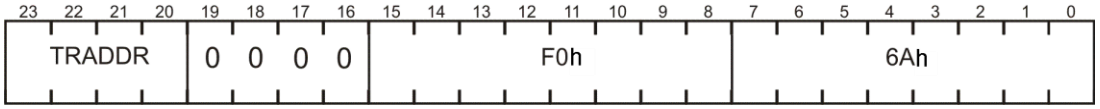


Рисунок 20.12 – Адрес по внешней шине

Регистр TRADDR устанавливает старшие значащие биты, остальные аппаратно установлены в 0F06Ah.

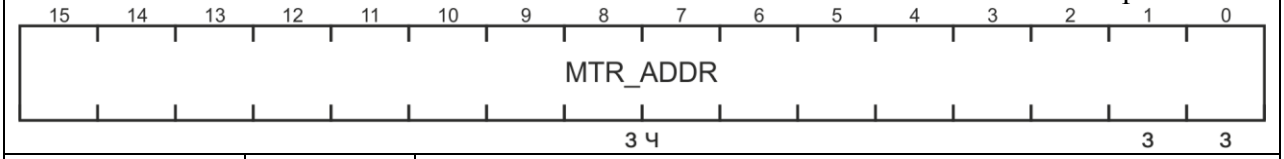
Управляемая монитором трассировка

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 и регистра RWDATA 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.

Регистры данных режима коммуникации приведены в таблицах 20.29 и 20.30.

Таблица 20.29 – Регистры данных режима коммуникации

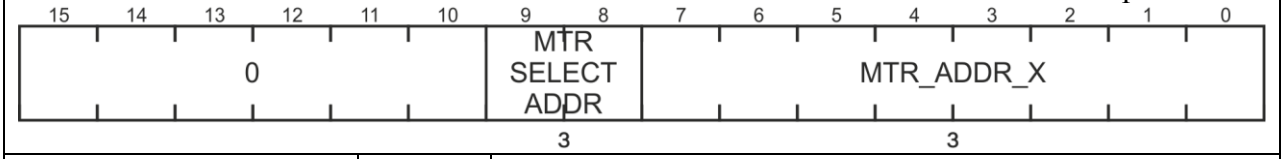
COMDATA Сброс: 0000h



Поле	Биты	Описание
MTR_ADDR	15–0	Данные

Таблица 20.30 – Регистры данных режима чтения-записи

RWDATA Сброс: 0000h



Поле	Биты	Описание	
MTR_SELECT_ADDR	9, 8	Поле указателя принадлежности адреса	
		00	Не определено
		01	Адрес источника
		10	Адрес назначения
		11	Зарезервировано
MTR_ADDR_X	7–0	Адрес	
0	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 = 0b, MTR_CTRL = 1b) 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 = 1b, если RW_ENABLED = 0b.

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_ENABLED.

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.

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.

21 Система команд

Таблицы 21.1 и 21.2 дают краткую информацию о командах и кодах команд микроконтроллера 1887ВЕ6Т.

Таблица 21.1 – Перекрестная таблица команд и кодов команд

	x0	x1	x2	x3	x4	x5	x6	x7
0x	ADD	ADDB	ADD	ADDB	ADD	ADDB	ADD	ADDB
1x	ADDC	ADDCB	ADDC	ADDCB	ADDC	ADDCB	ADDC	ADDCB
2x	SUB	SUBB	SUB	SUBB	SUB	SUBB	SUB	SUBB
3x	SUBC	SUBCB	SUBC	SUBCB	SUBC	SUBCB	SUBC	SUBCB
4x	CMP	CMPB	CMP	CMPB	–	–	CMP	CMPB
5x	XOR	XORB	XOR	XORB	XOR	XORB	XOR	XORB
6x	AND	ANDB	AND	ANDB	AND	ANDB	AND	ANDB
7x	OR	ORB	OR	ORB	OR	ORB	OR	ORB
8x	CMPI1	NEG	CMPI1	–	MOV	–	CMPI1	IDLE
9x	CMPI2	CPL	CMPI2	–	MOV	–	CMPI2	PWRDN
Ax	CMPD1	NEGB	CMPD1	–	MOVB	DISWDT	CMPD1	SRVWDT
Bx	CMPD2	CPLB	CMPD2	–	MOVB	EINIT	CMPD2	SRST
Cx	MOVBZ	–	MOVBZ	–	MOV	MOVBZ	SCXT	–
Dx	MOVBS	AT/EXTR	MOVBS	–	MOV	MOVBS	SCXT	EXTP/S/R
Ex	MOV	MOVB	PCALL	–	MOVB	–	MOV	MOVB
Fx	MOV	MOVB	MOV	MOVB	MOVB	–	MOV	MOVB

Таблица 21.2 – Перекрестная таблица команд и кодов команд

	x8	x9	xA	xB	xC	xD	xE	xF
0x	ADD	ADDB	BFLDL	MUL	ROL	JMPR	BCLR	BSET
1x	ADDC	ADDCB	BFLDH	MULU	ROL	JMPR	BCLR	BSET
2x	SUB	SUBB	BCMP	PRIOR	ROR	JMPR	BCLR	BSET
3x	SUBC	SUBCB	BMOVN	–	ROR	JMPR	BCLR	BSET
4x	CMP	CMPB	BMOV	DIV	SHL	JMPR	BCLR	BSET
5x	XOR	XORB	BOR	DIVU	SHL	JMPR	BCLR	BSET
6x	AND	ANDB	BAND	DIVL	SHR	JMPR	BCLR	BSET
7x	OR	ORB	BXOR	DIVLU	SHR	JMPR	BCLR	BSET
8x	MOV	MOVB	JB	–	–	JMPR	BCLR	BSET
9x	MOV	MOVB	JNB	TRAP	JMPI	JMPR	BCLR	BSET
Ax	MOV	MOVB	JBC	CALLI	ASHR	JMPR	BCLR	BSET
Bx	MOV	MOVB	JNBS	CALLR	ASHR	JMPR	BCLR	BSET
Cx	MOV	MOVB	CALLA	RET	NOP	JMPR	BCLR	BSET
Dx	MOV	MOVB	CALLS	RETS	EXTP/S/R	JMPR	BCLR	BSET
Ex	MOV	MOVB	JMPA	RETP	PUSH	JMPR	BCLR	BSET
Fx	–	–	JMPS	RETI	POP	JMPR	BCLR	BSET

Список кодов команд

Коды команд показаны в таблицах 21.3 – 21.6.

1 Команды кодируются посредством добавочных битов в поле операнда команды.

x0h – x7h: Rw, #data 3 или Rb, #data3

x8h – xBh: Rw, [Rw] или Rw, [Rw]

xCh – xFh: 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.xxxxb: EXTS или ATOMIC
01xx.xxxxb: EXTP
10xx.xxxxb: EXTSR или EXTR
11xx.xxxxb: 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.

Условные обозначения в таблицах 21.3 – 21.6:

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 бит-адресуемом пространстве памяти.
bitoff – Указатель слова в бит-адресуемом 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.

Условные обозначения для операций 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.

Условные обозначения для расширенных операций

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.

Условные обозначения для кодов 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 21.3 – Команды с кодами 00h – 3Fh

Код	Мнемокод	Операнды	Байт	Код	Мнемокод	Операнды	Байт
1	2	3	4	5	6	7	8
00h	ADD	Rw, Rw	2	20h	SUB	Rw, Rw	2
01h	ADDB	Rb, Rb	2	21h	SUBB	Rb, Rb	2
02h	ADD	reg, mem	4	22h	SUB	reg, mem	4
03h	ADDB	reg, mem	4	23h	SUBB	reg, mem	4
04h	ADD	mem, reg	4	24h	SUB	mem, reg	4
05h	ADDB	mem, reg	4	25h	SUBB	mem, reg	4
06h	ADD	reg, #data16	4	26h	SUB	reg, #data16	4
07h	ADDB	reg, #data8	4	27h	SUBB	reg, #data8	4
08h	ADD	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2	28h	SUB	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2
09h	ADDB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2	29h	SUBB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2
0Ah	BFLDL	bitoff, #mask8, #data8	4	2Ah	BCMP	bitaddr, bitaddr	4
0Bh	MUL	Rw, Rw	2	2Bh	PRIOR	Rw, Rw	2
0Ch	ROL	Rw, Rw	2	2Ch	ROR	Rw, Rw	2
0Dh	JMPR	cc_UC, rel	2	2Dh	JMPR	cc_EQ, rel или cc_Z, rel	2
0Eh	BCLR	bitoff.0	2	2Eh	BCLR	bitoff.2	2
0Fh	BSET	bitoff.0	2	2Fh	BSET	bitoff.2	2
10h	ADDC	Rw, Rw	2	30h	SUBC	Rw, Rw	2
11h	ADDCB	Rb, Rb	2	31h	SUBCB	Rb, Rb	2
12h	ADDC	reg, mem	4	32h	SUBC	reg, mem	4
13h	ADDCB	reg, mem	4	33h	SUBCB	reg, mem	4

Окончание таблицы 21.3

1	2	3	4	5	6	7	8
14h	ADDC	mem, reg	4	34h	SUBC	Mem, reg	4
15h	ADDCB	mem, reg	4	35h	SUBCB	Mem, reg	4
16h	ADDC	reg, #data16	4	36h	SUBC	reg, #data16	4
17h	ADDCB	reg, #data8	4	37h	SUBCB	reg, #data8	4
18h	ADDC	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2	38h	SUBC	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2
19h	ADDCB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2	39h	SUBCB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2
1Ah	BFLDH	bitoff, #mask8, #data8	4	3Ah	BMOVN	bitaddr, bitaddr	4
1Bh	MULU	Rw, Rw	2	3Bh	–	–	–
1Ch	ROL	Rw, #data4	2	3Ch	ROR	Rw, #data4	2
1Dh	JMPR	cc_NET, rel	2	3Dh	JMPR	cc_NE, rel или cc_NZ	2
1Eh	BCLR	bitoff.1	2	3Eh	BCLR	bitoff.3	2
1Fh	BSET	bitoff.1	2	3Fh	BSET	bitoff.3	2

Таблица 21.4 – Команды с кодами 40h – 7Fh

Код	Мнемокод	Операнды	Байт	Код	Мнемокод	Операнды	Байт
1	2	3	4	5	6	7	8
40h	CMP	Rw, Rw	2	60h	AND	Rw, Rw	2
41h	CMPB	Rb, Rb	2	61h	ANDB	Rb, Rb	2
42h	CMP	reg, mem	4	62h	AND	reg, mem	4
43h	CMPB	reg, mem	4	63h	ANDB	reg, mem	4
44h	–	–	–	64h	AND	mem, reg	4
45h	–	–	–	65h	ANDB	mem, reg	4
46h	CMP	reg, #data16	4	66h	AND	reg, #data16	4
47h	CMPB	reg, #data8	4	67h	ANDB	reg, #data8	4
48h	CMP	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2	68h	AND	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2
49h	CMPB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2	69h	ANDB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2
4Ah	BMOV	bitaddr, bitaddr	4	6Ah	BAND	bitaddr, bitaddr	4
4Bh	DIV	Rw	2	6Bh	DIVL	Rw	2
4Ch	SHL	Rw, Rw	2	6Ch	SHR	Rw, Rw	2
4Dh	JMPR	cc_V, rel	2	6Dh	JMPR	cc_N, rel	2
4Eh	BCLR	Bitoff.4	2	6Eh	BCLR	bitoff.6	2
4Fh	BSET	Bitoff.4	2	6Fh	BSET	bitoff.6	2
50h	XOR	Rw, Rw	2	70h	OR	Rw, Rw	2
51h	XORB	Rb, Rb	2	71h	ORB	Rb, Rb	2
52h	XOR	reg, mem	4	72h	OR	reg, mem	4
53h	XORB	reg, mem	4	73h	ORB	reg, mem	4
54h	XOR	mem, reg	4	74h	OR	mem, reg	4
55h	XORB	mem, reg	4	75h	ORB	mem, reg	4
56h	XOR	reg, #data16	4	76h	OR	reg, #data16	4

Окончание таблицы 21.4

1	2	3	4	5	6	7	8
57h	XORB	reg, #data8	4	77h	ORB	reg, #data8	4
58h	XOR	Rw, [Rw+] или Rw, [Rw] или Rw, data3	2	78h	OR	Rw, [Rw+] или Rw, [Rw] или Rw, #data3	2
59h	XORB	Rb, [Rw+] или Rb, [Rw] или Rb, #data3	2	79h	ORB	Rb, [Rw +] или Rb, [Rw] или Rb, #data3	2
5Ah	BOR	bitaddr, bitaddr	4	7Ah	BXOR	bitaddr, bitaddr	4
5Bh	DIV	Rw	2	7Bh	DIVLU	Rw	2
5Ch	SHL	Rw, #data4	2	7Ch	SHR	Rw, #data4	2
5Dh	JMPR	cc_NV, rel	2	7Dh	JMPR	cc_NN, rel	2
5Eh	BCLR	Bitoff.5	2	7Eh	BCLR	bitoff.7	2
5Fh	BSET	Bitoff.5	2	7Fh	BSET	bitoff.7	2

Таблица 21.5 – Команды с кодами 80h – BFh

Код	Мнемокод	Операнды	Байт	Код	Мнемокод	Операнды	Байт
80h	CMPI1	Rw, #data4	2	A0h	CMPD1	Rw, #data4	2
81h	NEG	Rw	2	A1h	NEGB	Rb	2
82h	CMPI1	Rw, mem	4	A2h	CMPD1	Rw, mem	4
83h	CoXXX	xx	4	A3h	CoXXX	xx	4
84h	MOV	[Rw], mem	–	A4h	MOVB	[Rw], mem	4
85h	–	–	–	A5h	DISWDT	–	4
86h	CMPI1	Rw, #data16	4	A6h	CMPID	Rw, #data16	4
87h	IDLE	–	4	A7h	SRVWDT		4
88h	MOV	[–Rw], Rw	2	A8h	MOV	Rw, [Rw]	2
89h	MOVB	[–Rw], Rb	2	A9h	MOVB	Rb, [Rw]	2
8Ah	JB	bitaddr, rel	4	AAh	JBC	bitaddr, rel	4
8Bh	–	–	2	ABh	CALLI	cc, [Rw]	2
8Ch	–	–	2	ACh	ASHR	Rw, Rw	2
8Dh	JMPR	cc_C, rel или cc_ULT, rel	2	ADh	JMPR	cc_SGT, rel	2
8Eh	BCLR	Bitoff.8	2	AEh	BCLR	bitoff.10	2
8Fh	BSET	Bitoff.8	2	AFh	BSET	bitoff.10	2
90h	CMPI2	Rw, #data4	2	B0h	CMPD2	Rw, #data4	2
91h	CPL	Rw	2	B1h	CPLB	Rb	2
92h	CMPI2	Rw, mem	4	B2h	CMPD2	Rw, mem	4
93h	–	–	4	B3h	–	–	4
94h	MOV	mem, [Rw]	4	B4h	MOVB	mem, [Rw]	4
95h	–	–	4	B5h	EINIT	–	4
96h	CMPI2	Rw, #data16	4	B6h	CMPD2	Rw, #data16	4
97h	PWRDN	–	4	B7h	SRST	–	4
98h	MOV	Rw, [Rw+]	2	B8h	MOV	[Rw], Rw	2
99h	MOVB	Rb, [Rw+]	2	B9h	MOVB	[Rw], Rb	2
9Ah	JNB	bitaddr, rel	4	BAh	JNBS	bitaddr, rel	4
9Bh	TRAP	#trap7	2	BBh	CALLR	rel	2
9Ch	JMPI	cc, [Rw]	2	BCh	ASHR	Rw, #data4	2
9Dh	JMPR	cc_NC, rel или cc_UGE, rel	2	BDh	JMPR	cc_SLE, rel	2
9Eh	BCLR	Bitoff.9	2	BEh	BCLR	bitoff.11	2
9Fh	BSET	Bitoff.9	2	BFh	BSET	bitoff.11	2

Таблица 21.6 – Команды с кодами C0h – FFh

Код	Мнемокод	Операнды	Байт	Код	Мнемокод	Операнды	Байт
C0h	MOVBZ	Rw, Rb	2	E0h	MOV	Rw, #data4	2
C1h	–	–	2	E1h	MOVB	Rb, #data4	2
C2h	MOVBZ	reg, mem	4	E2h	PCALL	reg, caddr	4
C3h	–	–	4	E3h	–	–	4
C4h	MOV	[Rw + #data16], Rw	–	E4h	MOVB	[Rw + #data16], Rb	4
C5h	MOVBZ	mem, reg	–	E5h	–	–	4
C6h	SCXT	reg, #data16	4	E6h	MOV	reg, #data16	4
C7h	–	–	4	E7h	MOVB	reg, #data8	4
C8h	MOV	[Rw], [Rw]	2	E8h	MOV	[Rw], [Rw+]	2
C9h	MOVB	[Rw], [Rw]	2	E9h	MOVB	[Rw], [Rw+]	2
CAh	CALLA	cc, addr	4	EAh	JMPA	cc, caddr	4
CBh	RET	–	2	EBh	RETP	reg	2
CCh	NOP	–	2	ECh	PUSH	reg	2
CDh	JMPR	cc_SLT, rel	2	EDh	JMPR	cc_UGT, rel	2
CEh	BCLR	Bitoff.12	2	EEh	BCLR	bitoff.14	2
CFh	BSET	Bitoff.12	2	EFh	BSET	bitoff.14	2
D0h	MOVBS	Rw, Rb	2	F0h	MOV	Rw, Rw	2
D1h	ATOMIC / EXTR	#irang2	2	F1h	MOVB	Rb, Rb	2
D2h	MOVBS	reg, mem	4	F2h	MOV	reg, mem	4
D3h	–	–	4	F3h	MOVB	reg, mem	4
D4h	MOV	Rw, [Rw + #data16]	4	F4h	MOVB	Rb, [Rw + #data16]	4
D5h	MOVBS	mem, reg	4	F5h	–	–	4
D6h	SCXT	reg, mem	4	F6h	MOV	mem, reg	4
D7h	EXTP(R), EXTS(R)	#pag10, #irang2 #seg8, #irang2	4	F7h	MOVB	mem, reg	4
D8h	MOV	[Rw+], [Rw]	2	F8h	–	–	2
D9h	MOVB	[Rw+], [Rw]	2	F9h	–	–	2
DAh	CALLS	seg, caddr	4	FAh	JMPS	seg, caddr	4
DBh	RETS	–	2	FBh	RETI		2
DCh	EXTP(R), EXTS(R)	Rw, #irang2	2	FCh	POP	reg	2
DDh	JMPR	cc_SGE, rel	2	FDh	JMPR	cc_ULE, rel	2
DEh	BCLR	Bitoff.13	2	FEh	BCLR	bitoff.15	2
DFh	BSET	Bitoff.13	2	FFh	BSET	bitoff.15	

Подробное описание команд микроконтроллера 1887ВЕ6Т приводится в приложении Г настоящего РП.

Заключение

В настоящем руководстве пользователя приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения микроконтроллера 1887ВЕ6Т, который представляет собой 16-разрядную однокристальную микро-ЭВМ с RISC-архитектурой с повышенной стойкостью к СВВФ на базе КНИ-технологии. Разработанную СБИС предполагается использовать в модернизированных и перспективных образцах ВВСТ космического назначения, а также в аппаратуре систем управления ракет-носителей.

Руководство пользователя может служить практическим руководством по применению микроконтроллеров 1887ВЕ6Т для разработчиков систем на их основе и программистов.

Приложение А
(обязательное)
Регистры микроконтроллера

А.1 Регистры блока ЦПУ

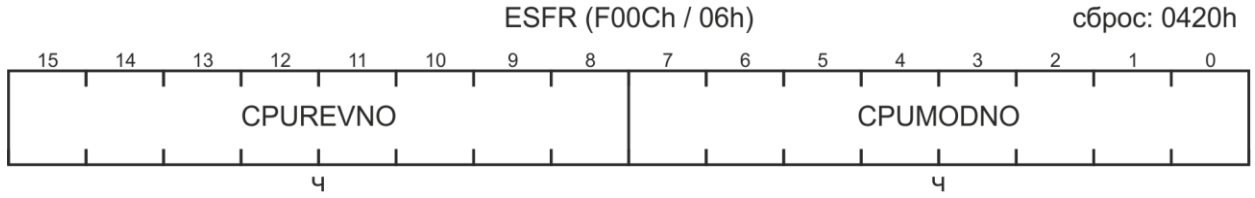
Регистры блока ЦПУ представлены в таблицах А.1 – А.23.

Таблица А.1 – Список регистров

Мнемоническое обозначение	Область памяти	Адрес		Сброс
ADDRSEL1	SFR	FE18h	0Ch	0000h
ADDRSEL2	SFR	FE1Ah	0Dh	0000h
ADDRSEL3	SFR	FE1Ch	0Eh	0000h
ADDRSEL4	SFR	FE1Eh	0Fh	0000h
BUSCON0	SFR-b	FF0Ch	86h	0000h
BUSCON1	SFR-b	FF14h	8Ah	0000h
BUSCON2	SFR-b	FF16h	8Bh	0000h
BUSCON3	SFR-b	FF18h	8Ch	0000h
BUSCON4	SFR-b	FF1Ah	8Dh	0000h
CP	SFR	FE10h	08h	FC00h
CPUID	ESFR	F00Ch	06h	0420h
CSP	SFR	FE08h	04h	0000h
DPP0	SFR	FE00h	00h	0000h
DPP1	SFR	FE02h	01h	0001h
DPP2	SFR	FE04h	02h	0002h
DPP3	SFR	FE06h	03h	0003h
MDC	SFR-b	FF0Eh	87h	0000h
MDH	SFR	FE0Ch	06h	0000h
MDL	SFR	FE0Eh	07h	0000h
ONES	SFR-b	FF1Eh	8Fh	FFFFh
PSW	SFR-b	FF10h	88h	0000h
RP0H	ESFR-b	F108h	84h	--XXh
SP	SFR	FE12h	09h	FC00h
STKOV	SFR	FE14h	0Ah	FA00h
STKUN	SFR	FE16h	0Bh	FC00h
SYSCON	SFR-b	FF12h	89h	XXXXh
XADRS1	ESFR	F014h	0Ah	XXXXh
XADRS2	ESFR	F016h	0Bh	XXXXh
XADRS3	ESFR	F018h	0Ch	XXXXh
XADRS4	ESFR	F01Ah	0Dh	XXXXh
XADRS5	ESFR	F01Ch	0Eh	XXXXh
XADRS6	ESFR	F01Eh	0Fh	XXXXh
XBCON1	ESFR-b	F114h	8Ah	XXXXh
XBCON2	ESFR-b	F116h	8Bh	XXXXh
XBCON3	ESFR-b	F118h	8Ch	XXXXh
XBCON4	ESFR-b	F11Ah	8Dh	XXXXh
XBCON5	ESFR-b	F11Ch	8Eh	XXXXh
XBCON6	ESFR-b	F11Eh	8Fh	XXXXh
XPERCON	ESFR	F024h	12h	0000h
ZEROS	SFR-b	FF1Ch	8Eh	0000h

Таблица А.2 – Регистр идентификации ЦПУ

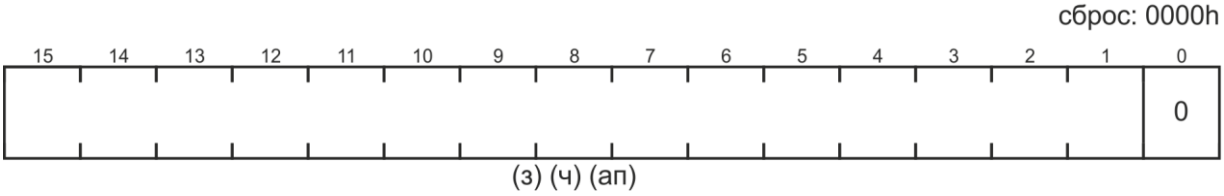
CPUID



Поле	Биты	Описание
CPUREVNO	15–8	Номер микроконтроллера
CPUMODNO	7–0	Версия микроконтроллера

Таблица А.3 – Указатель команд

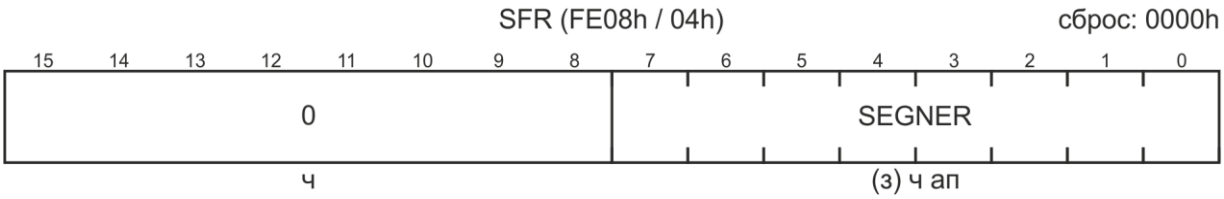
IP



Поле	Биты	Описание
IP	15–1	Внутрисегментное смещение, по которому необходимо произвести выборку инструкции. IP относится к сегменту, указанному в битовом поле SEGNR регистра CSP. Выровнен пословно

Таблица А.4 – Указатель сегмента кода

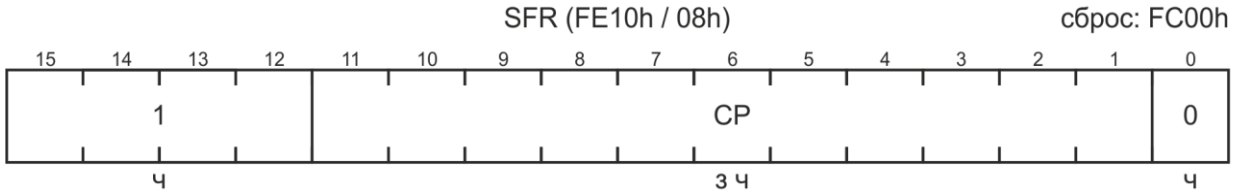
CSP



Поле	Биты	Описание
SEGNER	7–0	Поле содержит значение сегмента кода, откуда происходит выборка команды
0	15–8	Зарезервировано. Не использовать

Таблица А.5 – Контекстный указатель

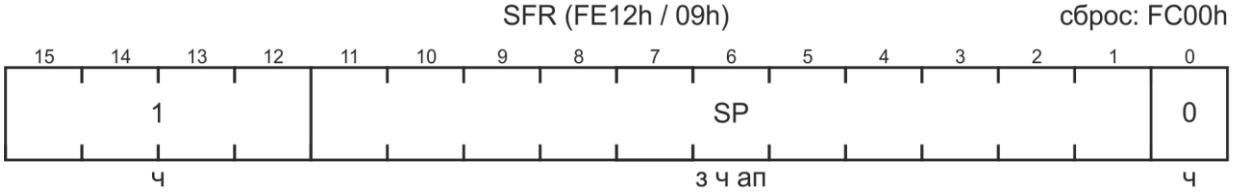
CP



Поле	Биты	Описание
CP	11–1	Изменяемое битовое поле указателя (всегда указывает на DPRAM) Определяет адрес слова/байта используемого банка регистров. Если при записи нового значения содержимое битов CP.11 – CP.9 записываемого значения равно нулю, то в биты CP.10 и CP.11 автоматически записываются единицы. Выровнен пословно.
1	15–12	Зарезервировано. Всегда возвращает единицы

Таблица А.6 – Указатель стека

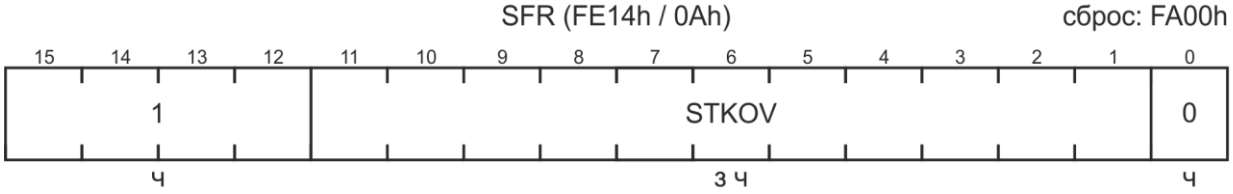
SP



Поле	Биты	Описание
SP	11–1	Указатель вершины системного стека. Выровнен пословно.
1	15–12	При чтении всегда возвращает значение единицы

Таблица А.7 – Указатель переполнения стека

STKOV



Поле	Биты	Описание
STKOV	11–1	Битовое поле, определяющее смещение адреса сегмента нижней границы системного стека. Выровнен пословно
1	15–12	Зарезервировано. Всегда возвращает единицы

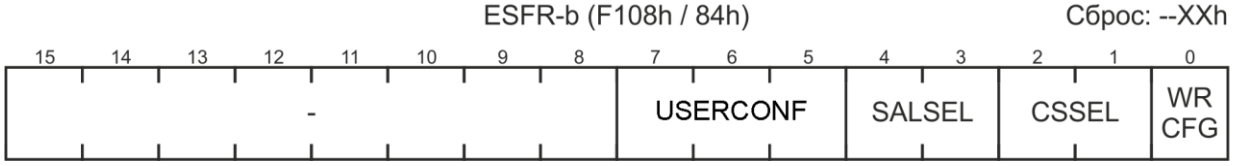
Таблица А.8 – Указатель опустошения стека

STKUN

Поле	Биты	Описание
STKUN	11–1	Битовое поле, определяющее смещение адреса сегмента верхней границы системного стека. Выровнен пословно.
1	15–12	Зарезервировано. Всегда возвращает единицы

Таблица А.9 – Регистр начальной конфигурации внешней шины

RP0H

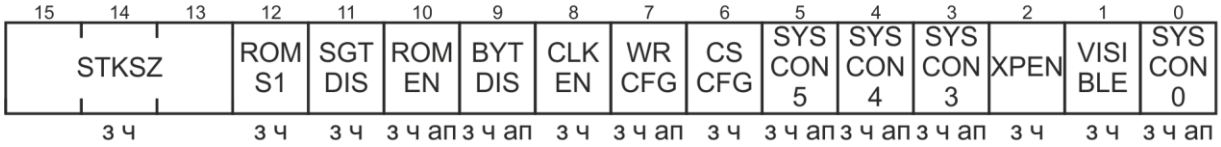


Поле	Биты	Описание	
USERCONF	7–5	Код пользователя. Нефункциональные биты	
SALSEL	4–3	Разрядность внешней шины. Поле задает количество разрядов старшей части адреса	
		00	Сегментная часть адреса A19–A16. Адресное пространство – 1 Мбайт
		01	Адресное пространство – 64 Мбайт
		10	Сегментная часть адреса A23–A16. Адресное пространство – 16 Мбайт
		11	Сегментная часть адреса A17, A16. Адресное пространство 256 Кбайт
CSSEL	2–1	Количество используемых каналов выборки	
		00	3 канала – CS0#, CS1#, CS2#
		01	2 канала – CS0#, CS1#
		10	Каналы не используются
		11	5 каналов – CS0#, CS1#, CS2#, CS3#, CS4#. Оставшиеся свободные каналы без pull-down резисторов
WRCFG	0	Бит конфигурации сигнала записи	
		0	WR# и BHE# работают в соответствии со своими функциями
		1	WR# работает как WRL#, BHE# – как WRH#
–	15–8	Зарезервировано	

Таблица А.10 – Регистр управления системы

SYSCON

SFR (FF12h / 89h) сброс: 00000XX0 X0000000b



Поле	Биты	Описание	
1	2	3	
STKZ	15–13	Биты выбора размера системного стека в DPRAM	
		000	256 слов (FBFEh…FA00h)
		001	128 слов (FBFEh…FB00h)
		010	64 слова (FBFEh…FB80h)
		011	32 слова (FBFEh…FBC0h)
		100	512 слов (FBFEh…F800h)
		101	Зарезервировано
		110	Зарезервировано
		111	1536 слов (FDFEh …F200h)
ROMS1	12	Бит локализации внутренней памяти	
		0	Внутренняя память программ размещается в нулевом сегменте (0000h – 7FFFh)
		1	Внутренняя память программ размещается в первом сегменте (10000h – 17FFFh)
SGTDIS	11	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)
ROMEN	10	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
BYTDIS	9	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
CLKEN	8	Разрешение вывода системного тактового сигнала CLKOUT	
WRCFG	7	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#

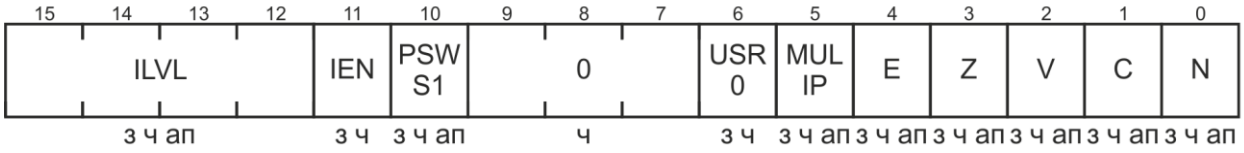
Окончание таблицы А.10

1	2	3	
CSCFG	6	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
SYSCON3–SYSCON5	3–5	Зарезервировано	
XPEN	2	Бит разрешения XBUS периферии	
		0	Доступ к XBUS периферии и обращение к ней запрещены
		1	Доступ к XBUS периферии и обращение к ней разрешены
VISIBLE	1	Управление режимом видимости шины XBUS	
		0	Обращение по шине XBUS производится только внутри кристалла
		1	Обращение по шине XBUS дублируется на внешней шине
SYSCON0	0	Бит системной конфигурации	
Примечание – Регистр SYSCON не может быть изменен после выполнения команды EINIT.			

Таблица А.11 – Слово состояния процессора

PSW

SFR (FF10h / 88h) сброс: 0000h



Поле	Бит	Описание	
1	2	3	
ILVL	15–12	Уровень приоритета ЦПУ. Изменяется аппаратно при входе в подпрограмму обслуживания прерываний. Для запрещения прерываний ILVL может быть программно изменен. В случае присвоения ЦПУ максимально возможного уровня 15, текущая деятельность ЦПУ не может быть прервана, за исключением аппаратных ловушек и внешних немаскируемых прерываний	
		Fh	Самый высокий уровень приоритета
		…	…
		0h	Самый низкий уровень приоритета
		После сброса микроконтроллера все прерывания запрещены, и ЦПУ присвоен самый низкий уровень приоритета – 0	
IEN	11	Бит глобального разрешения прерываний	
		0	Запрещены
		1	Разрешены
PSWS1	10	Зарезервировано для системы	

Продолжение таблицы А.11

1	2	3	
USR0	6	Пользовательский флаг общего назначения	
MULIP	5	Флаг выполнения операции умножения и деления	
		0	Отсутствие выполнения операции умножения и деления
		1	Умножение и деление были прерваны
		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	
E	4	Флаг конца таблицы	
		0	Адрес исходного операнда команды не 8000h и не 80h
		1	Адрес исходного операнда команды 8000h или 80h
		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 (8000h для слов данных или 80h для байтов данных), E-флаг устанавливается	
Z	3	Флаг нулевого результата. 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	

Продолжение таблицы А.11

1	2	3
V	2	Флаг переполнения результата вычислений. 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 8000h до плюс 7FFFh или для байтов от минус 80h до плюс 7Fh). 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 C-флага ошибка сдвига, вызываемая командой правого сдвига, может быть оценена до величины результата половины LSB. Вместе с V-флагом, C-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
C	1	Флаг переноса. 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 C-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 C-флаг устанавливается в «0», когда дополнительное сложение приводит к переносу. C-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 C-флаг подставляется значение бита, который был изменен в слове или байте данных последним. Если итог операции был «0», то C-флаг принимает нулевое значение. Для логических операций с битами с одним операндом C-флаг всегда принимает «0» значение. Для логических битовых операций с двумя операндами C-флаг принимает результат операции логического «И» с двумя соответствующими битами

Окончание таблицы А.11

1	2	3
N	0	Флаг отрицательного результата. 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=1b, положительный N=0b). Отрицательные числа всегда представляются в виде дополнительных чисел. Возможный диапазон чисел со знаком: в случае данных из одного слова – от минус 8000h до плюс 7FFFh, либо для однобайтных данных – от минус 80h до плюс 7Fh. 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
–	9–7	Зарезервировано

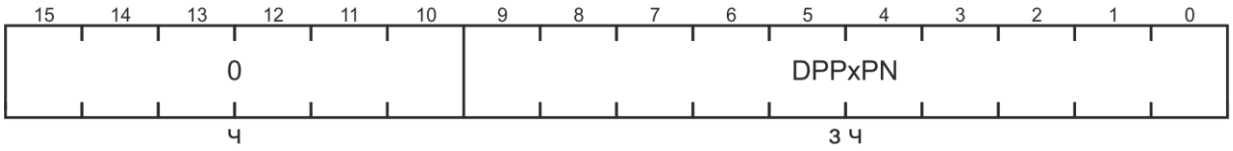
Для оценки ошибки округления при сдвиге вправо следует пользоваться флагами C и V:

Величина ошибки	Состояние флагов	
	C	V
Нет ошибки	0	0
0 < ошибка < 1/2 LSB (самый младший бит)	0	1
ошибка = 1/2 LSB	1	0
1/2 LSB < ошибка	1	1

Таблица А.12 – Указатель страницы данных (x – номер страницы данных)

DPPx

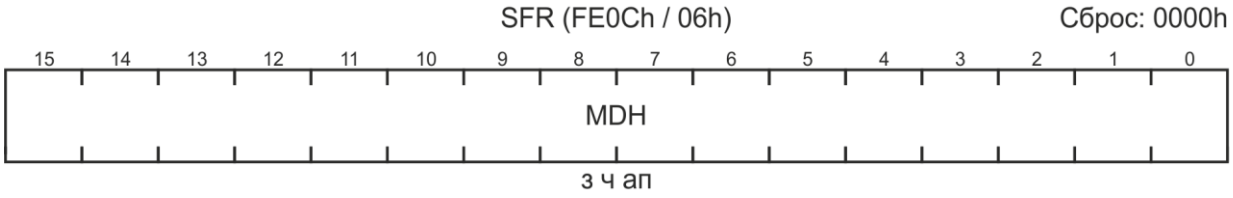
Сброс: 0000h



Поле	Биты	Описание
DPPxPN	9–0	Номер страницы данных в пределах выбранного регистра DPPx
0	15–10	Зарезервировано. Всегда возвращает нули

Таблица А.13 – Старший и младший регистры умножения/деления

MDH

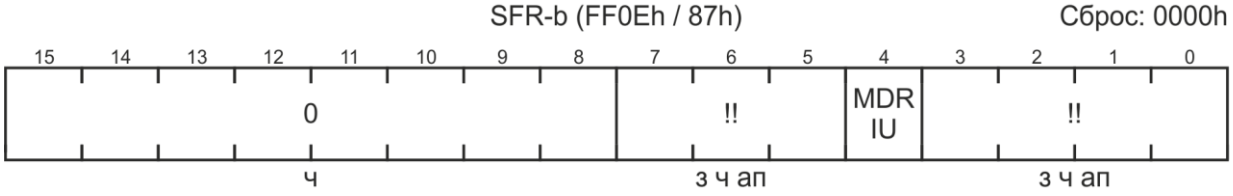


MDL

Поле	Биты	Описание
MDH	15–0	Старшее слово 32-рязрядного регистра умножения/деления
MDL	15–0	Младшее слово 32-рязрядного регистра умножения/деления

Таблица А.14 – Регистр управления умножением/делением

MDC

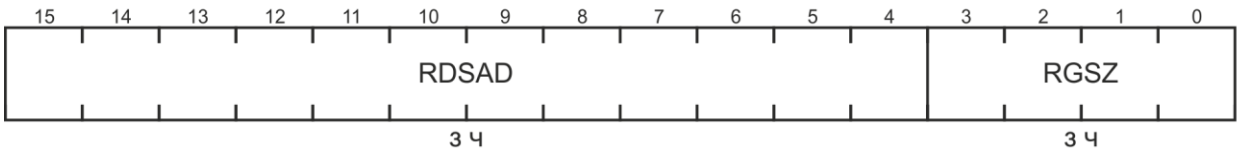


Поле	Биты	Описание	
!!	7–5, 3–0	Биты используются модулем умножения/деления для внутренних операций. Не следует изменять состояния этих битов без сохранения и восстановления значений регистра MDC	
MDRIU	4	Флаг использования регистра умножения/деления	
		0	Сбрасывается при чтении регистра MDL
		1	Устанавливается при записи в регистры MDH и MDL и во время выполнения операций умножения/деления
0	15–8	Зарезервировано	

Таблица А.15 – Регистр выбора адреса

ADDRSELx

Сброс: 0000h

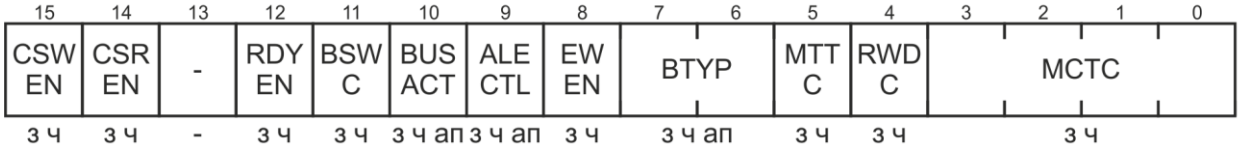


Поле	Бит	Описание
RDSAD	15–4	Биты разрядов A23 – A12 начального адреса окна
RGSZ	3–0	Биты управления размером адресного окна, управляемого парой регистров BUSCONx – ADDRSELx

Таблица А.16 – Регистры управления шиной

BUSCONx

Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
CSWEN	15	Бит разрешения сигнала записи WR#, WRL#, WRH# (см. таблицу А.17)	
		0	Формирование сигнала CSx# не зависит от сигнала записи
		1	Формирование сигнала CSx# во время активного сигнала записи WR#, WRL#, WRH#, BHE#
CSREN	14	Бит разрешения сигнала CSx# во время цикла чтения (см. таблицу А.17)	
		0	Сигнал CSx# не зависит от сигнала RD#
		1	Сигнал CSx# генерируется во время активности RD#
RDYEN	12	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#
BSWC	11	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
BUSACT	10	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 А.16

1	2	3	
ALECTL	9	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
EWEN	8	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
BTYP	7–6	Биты управления внешней конфигурацией:	
		00	Разрядная демультиплексная шина
		01	Разрядная мультиплексная шина
		10	6-разрядная демультиплексная шина
MTTC	5	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
RWDC	4	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
MCTC	3–0	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 = 1b старший бит (3) поля MCTC не используется. При этом число циклов ожидания может выбираться от 0 до 7	
–	13	Зарезервировано	

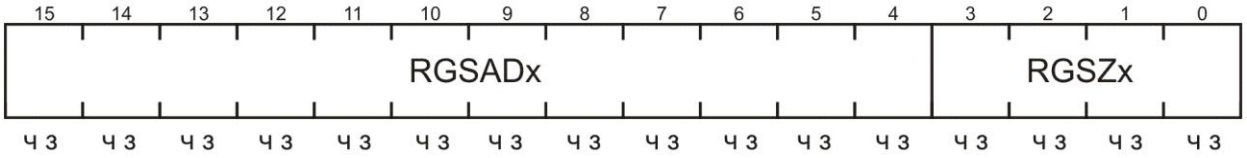
Таблица А.17 – Комбинации состояния битов CSWEN и CSREN

CSWEN	CSREN	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#

Таблица А.18 – Регистр выбора адреса шины XBUS (x – номер окна)

XADRSx

Сброс: XXXXh



Поле	Бит	Описание
RGSADx	15–4	Выбор стартового адреса адресного окна x
RGSZx	3–0	Выбор размера адресного окна x. Допустимые коды см. в таблице А.19

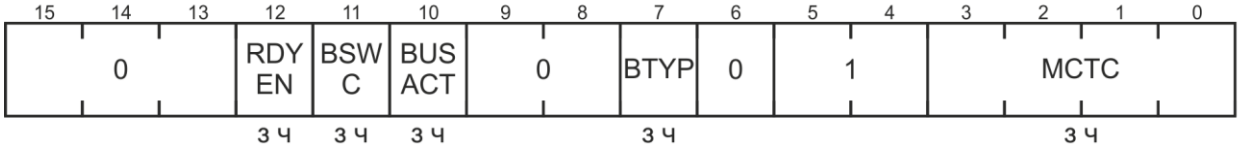
Таблица А.19 – Коды значений поля RGSZ

RGSZ	Размер адресного окна	Стартовый адрес выбранного адресного окна		
			Соответствующие (R) биты выбора стартового адреса RGSAD	
0h	256 байт	0000	RRRR RRRR RRRR	0000 0000
1h	512 байт	0000	RRRR RRRR RRR0	0000 0000
2h	1 кбайт	0000	RRRR RRRR RR00	0000 0000
3h	2 кбайта	0000	RRRR RRRR R000	0000 0000
4h	4 кбайта	0000	RRRR RRRR 0000	0000 0000
5h	8 кбайт	0000	RRRR RRR0 0000	0000 0000
6h	16 кбайт	0000	RRRR RR00 0000	0000 0000
7h	32 кбайта	0000	RRRR R000 0000	0000 0000
8h	64 кбайта	0000	RRRR 0000 0000	0000 0000
9h	128 кбайт	0000	RRR0 0000 0000	0000 0000
Ah	256 кбайт	0000	RR00 0000 0000	0000 0000
Bh	512 кбайт	0000	R000 0000 0000	0000 0000
Примечание – Остальные коды поля RGSZ зарезервированы.				

Таблица А.20 – Регистр управления шиной XBUS

XBCONx

Сброс: XXXXh



Поле	Бит	Описание	
1	2	3	
RDYEN	12	Разрешение READY	
		0	Длина шинного цикла управляется полем MCTC
		1	Длина шинного цикла управляется сигналом READY#
BSWC	11	Управление переключением BUSCON	
		0	Адресное окно переключается сразу
		1	Вставляется задержка третьего состояния (2TCL) для следующего окна

Окончание таблицы А.20

1	2	3
BUSACT	10	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
BTYP	7	Определение типа XBUS
		0 8-битная демультиплексная шина
		1 16-битная демультиплексная шина
MCTC	3–0	Количество тактов задержки.
1	5–4	Зарезервировано. Всегда возвращает единицы
0	15–13, 9–8, 6	Зарезервировано. Всегда возвращает нули

Таблица А.21 – Регистр управления X-периферией

XPERCON

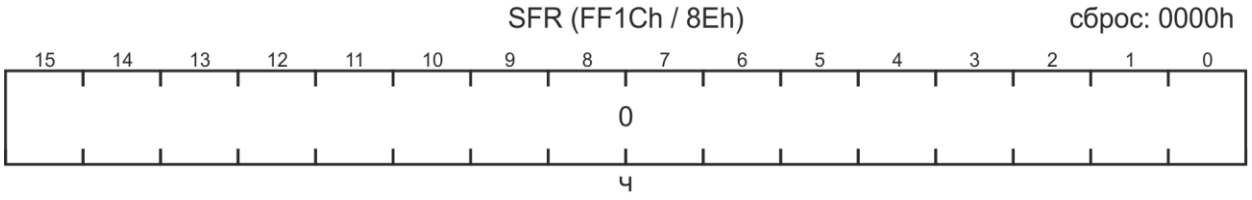
XSFR (F024h/12h) Сброс: 0000h

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
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 З

Поле	Биты	Описание	
PER3	3	Бит включения сигнала XCS4 для блока ШИМ (CAPCOM6)	
		0	Неактивен
		1	Активен
PER2	2	Бит включения сигнала XCS3 для старших 2 Кбайт XRAM	
		0	Неактивен
		1	Активен
PER1	1	Бит включения сигнала XCS2 для младших 2 Кбайт XRAM	
		0	Неактивен
		1	Активен
PER0	0	Бит включения сигнала XCS1 для контроллера CAN	
		0	Неактивен
		1	Активен
PER15–PER4	15–4	Не используются	

Таблица А.22 – Регистр постоянного нуля

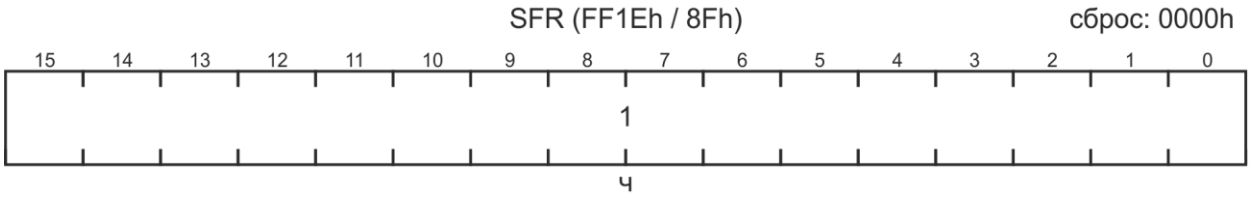
ZEROS



Поле	Биты	Описание
0	15–0	Всегда возвращает нули

Таблица А.23 – Регистр постоянной единицы

ONES



Поле	Биты	Описание
1	15–0	Всегда возвращает единицы

А.2 Регистры блока умножения-накопления (MAC)

Регистры блока умножения-накопления MAC представлены в таблицах А.24 – А.31.

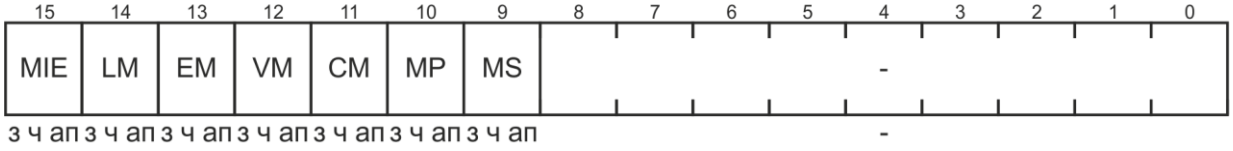
Таблица А.24 – Список регистров

Мнемоническое обозначение	Область памяти	Адрес		Сброс
IDX0	SFR-b	FF08h	84h	0000h
IDX1	SFR-b	FF0Ah	85h	0000h
MAH	SFR	FE5Eh	2Fh	0000h
MAL	SFR	FE5Ch	2Eh	0000h
MCW	SFR-b	FFDCh	EEh	0000h
MRW	SFR-b	FFDAh	EDh	0000h
MSW	SFR-b	FFDEh	EFh	0200h
QR0	ESFR	F004h	02h	0000h
QR1	ESFR	F006h	03h	0000h
QX0	ESFR	F000h	00h	0000h
QX1	ESFR	F002h	01h	0000h

Таблица А.25 – Регистр управления блока MAC

MCW

SFR (FFDCh / EEh) Сброс: 0000h

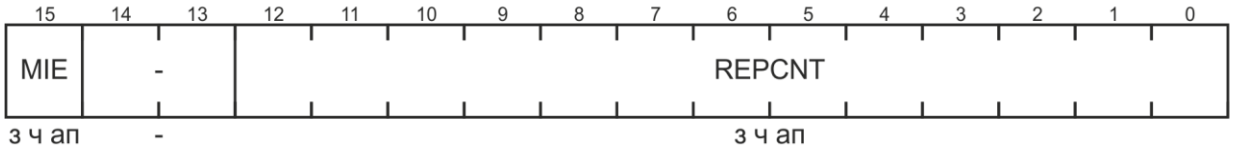


Поле	Бит	Описание	
MIE	15	Разрешение на прерывание блока MAC	
		0	Общее запрещение на прерывание блока MAC
		1	Общее разрешение на прерывание блока MAC
LM	14	Маска ограничения	
		0	Флаг SL не может генерировать прерывание
		1	Флаг SL может генерировать прерывание
EM	13	Маска расширения	
		0	Флаг E не может генерировать прерывание
		1	Флаг E может генерировать прерывание
VM	12	Маска переполнения	
		0	Флаг SL не может генерировать прерывание
		1	Флаг SL может генерировать прерывание
CM	11	Маска переноса	
		0	Флаг C не может генерировать прерывание
		1	Флаг C может генерировать прерывание
MP	10	Управление однобитным делителем	
		0	Запрещение сдвига результата умножения
		1	Разрешение сдвига результата умножения
MS	9	Контроль ограничения	
		0	Запрещение автоматического ограничения
		1	Разрешение автоматического ограничения
–	8–0	Зарезервированы	

Таблица А.26 – Регистр повторения блока MAC

MRW

SFR (FFDAh / EDh) Сброс: 0000h

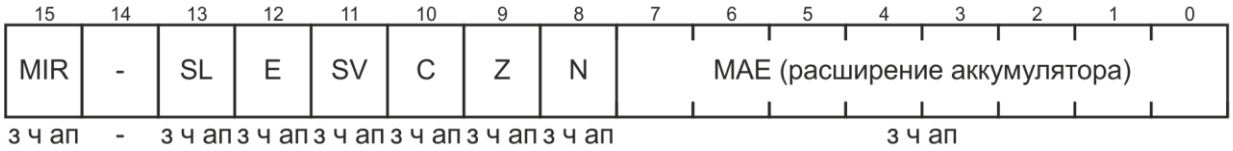


Поле	Бит	Описание
MIE	15	Флаг повторения. Устанавливается, когда выполняется повторяемая команда
REPCNT	12–0	Счетчик повторений. 13-битное целое число без знака. Число повторений команды минус 1
–	14, 13	Зарезервировано

Таблица А.27 – Статусный регистр блока MAC

MSW

SFR (FFDEh / EFh) Сброс: 0000h

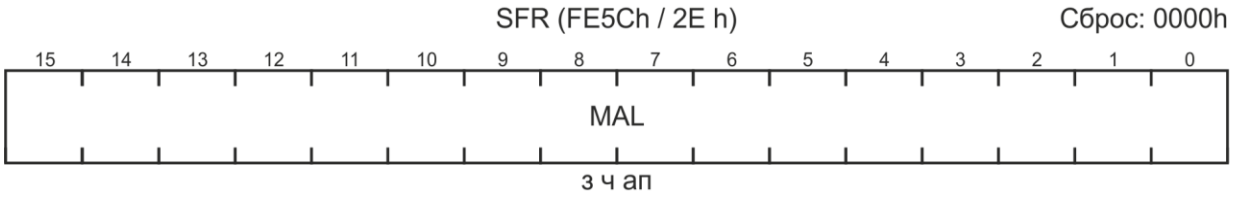


Поле	Бит	Описание	
MIR	15	Запрос на прерывание блока MAC	
		0	Нет запроса на прерывание блока MAC
		1	Запрос на прерывание блока MAC
SL	13	Флаг ограничения	
		0	Не произошло автоматического 32-битного ограничения
		1	Произошло автоматическое 32-битное ограничение
E	12	Флаг расширения	
		0	MAE не содержит значащих битов
		1	MAE содержит значащие биты
SV	11	Флаг переполнения	
		0	Не произошло 40-битного переполнения
		1	Произошло 40-битное переполнение
C	10	Флаг переноса	
		0	Не произошел перенос/заем
		1	Произошел перенос/заем
Z	9	Флаг нуля	
		0	Результат не равен нулю
		1	Результат равен нулю
N	8	Флаг отрицательного результата	
		0	Отрицательный результат блока MAC
		1	Положительный результат блока MAC
MAE	7–0	Расширение аккумулятора. Восемь старших значащих битов 40-битного аккумулятора блока MAC (биты с 40 по 33)	
–	14	Зарезервировано	

Таблица А.28 – Регистры старшего и младшего байт аккумулятора

MAH

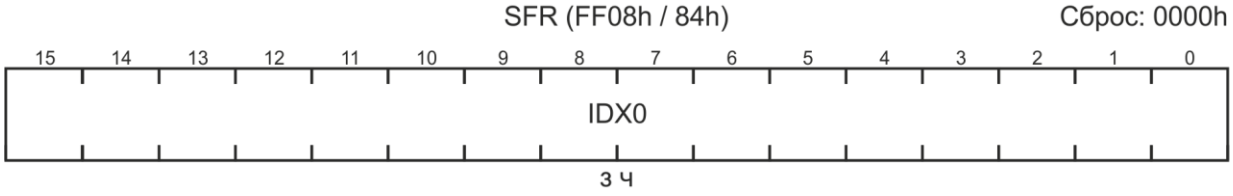
MAL



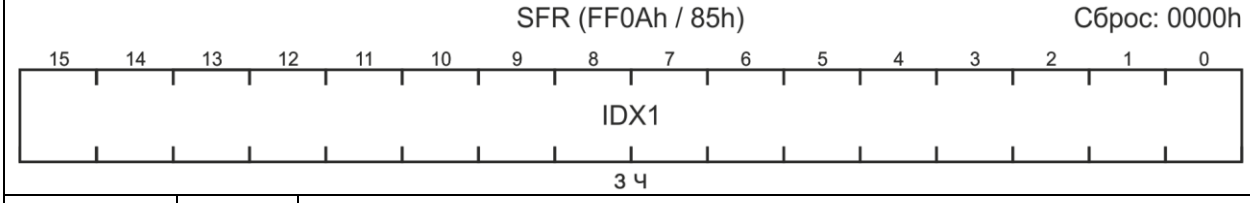
Поле	Бит	Описание
MAH	15–0	Поле содержит 16 бит 40-битного аккумулятора блока MAC (биты с 31 по 16)
MAL	15–0	Поле содержит младшие 16 бит 40-битного аккумулятора блока MAC (биты с 15 по 0)

Таблица А.29 – Регистры нулевого и первого указателей адреса

IDX0



IDX1

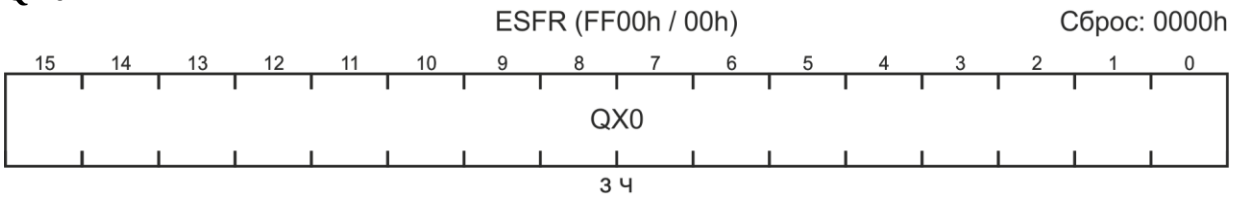


Поле	Бит	Описание
IDX0	15–0	Изменяемая часть указателя адреса
IDX1	15–0	

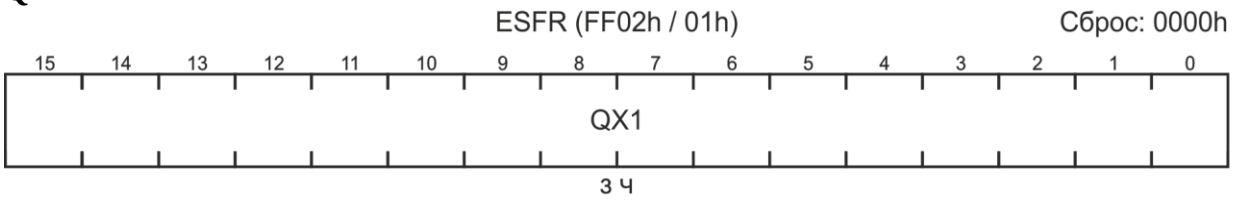
Примечание – Поскольку поля IDX0 и IDX1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю.

Таблица А.30 – Регистры смещения для нулевого и первого указателей адреса

QX0



QX1

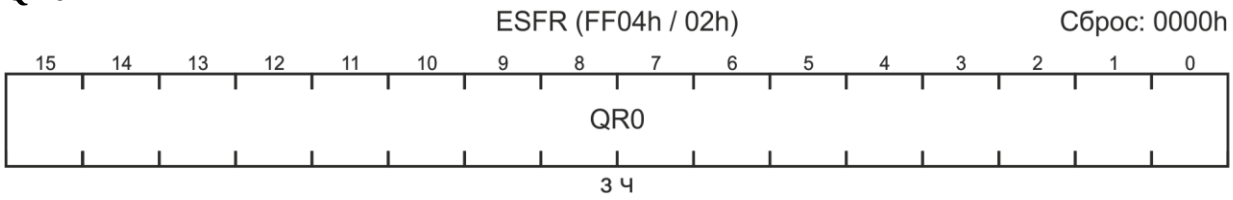


Поле	Бит	Описание
QX0	15–0	16-разрядное смещение для указателя адреса IDX0/IDX1
QX1	15–0	

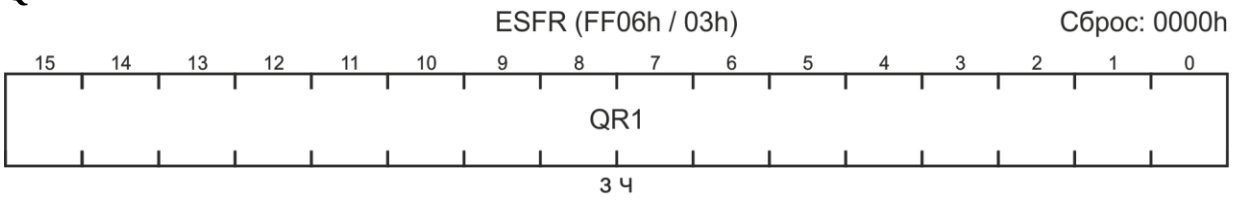
Примечание – Поскольку поля QR0 и QR1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю.

Таблица А.31 – Регистры смещения для нулевого и первого указателей адреса GPR

QR0



QR1



Поле	Бит	Описание
QR0	15–0	16-разрядное смещение для указателя адреса GPR
QR1	15–0	

Примечание – Поскольку поля QR0 и QR1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю.

А.3 Регистры системы прерываний

Регистры системы прерываний представлены в таблицах А.32 – А.38.

Таблица А.32 – Список регистров управления

Мнемоническое обозначение	Область памяти	Адрес		Сброс
EOPIC	ESFR-b	F180h	C0h	0000h
EXICON	ESFR-b	F1C0h	E0h	0000h
EXISEL0	ESFR-b	F1DAh	EDh	0000h
EXISEL1	ESFR-b	F1D8h	ECh	0000h
TFR	SFR-b	FFACh	D6h	0000h

Таблица А.33 – Список регистров

Мнемоническое обозначение (xxIC)	Область памяти	Адрес		Сброс
1	2	3		4
CAN_IC0	ESFR-b	F128h	94h	0000h
CAN_IC1	ESFR-b	F12Ah	95h	0000h
CAN_IC2	ESFR-b	F12Ch	96h	0000h
CAN_IC3	ESFR-b	F12Eh	97h	0000h
CAN_IC4	ESFR-b	F132h	99h	0000h
CAN_IC5	ESFR-b	F134h	9Ah	0000h
CAN_IC6	ESFR-b	F136h	9Bh	0000h
CAN_IC7	ESFR-b	F138h	9Ch	0000h
CAN_IC8	ESFR-b	F140h	A0h	0000h
CAN_IC9	ESFR-b	F142h	A1h	0000h
CAN_IC10	ESFR-b	F144h	A2h	0000h
CAN_IC11	ESFR-b	F146h	A3h	0000h
CAN_IC12	ESFR-b	F148h	A4h	0000h
CAN_IC13	ESFR-b	F14Ah	A5h	0000h
CAN_IC14	ESFR-b	F14Ch	A6h	0000h
CAN_IC15	ESFR-b	F14Eh	A7h	0000h
CC0IC	SFR-b	FF78h	BCh	0000h
CC1IC	SFR-b	FF7Ah	BDh	0000h
CC2IC	SFR-b	FF7Ch	BEh	0000h
CC3IC	SFR-b	FF7Eh	BFh	0000h
CC4IC	SFR-b	FF80h	C0h	0000h
CC5IC	SFR-b	FF82h	C1h	0000h
CC6IC	SFR-b	FF84h	C2h	0000h
CC7IC	SFR-b	FF86h	C3h	0000h
CC8IC	SFR-b	FF88h	C4h	0000h
CC9IC	SFR-b	FF8Ah	C5h	0000h
CC10IC	SFR-b	FF8Ch	C6h	0000h
CC11IC	SFR-b	FF8Eh	C7h	0000h
CC12IC	SFR-b	FF90h	C8h	0000h
CC13IC	SFR-b	FF92h	C9h	0000h
CC14IC	SFR-b	FF94h	CAh	0000h
CC15IC	SFR-b	FF96h	CBh	0000h
CC16IC	ESFR-b	F160h	B0h	0000h
CC17IC	ESFR-b	F162h	B1h	0000h
CC18IC	ESFR-b	F164h	B2h	0000h

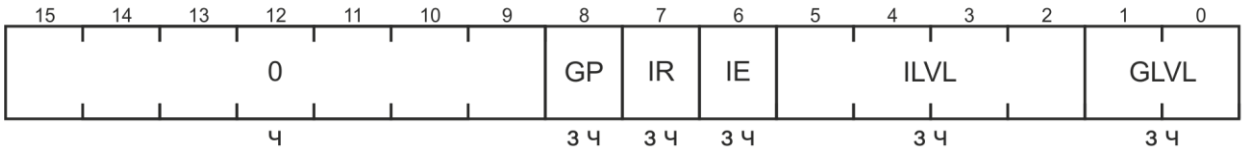
Окончание таблицы А.33

1	2	3	4	5
CC19IC	ESFR-b	F166h	B3h	0000h
CC20IC	ESFR-b	F168h	B4h	0000h
CC21IC	ESFR-b	F16Ah	B5h	0000h
CC22IC	ESFR-b	F16Ch	B6h	0000h
CC23IC	ESFR-b	F16Eh	B7h	0000h
CC24IC	ESFR-b	F170h	B8h	0000h
CC25IC	ESFR-b	F172h	B9h	0000h
CC26IC	ESFR-b	F174h	BAh	0000h
CC27IC	ESFR-b	F176h	BBh	0000h
CC28IC	ESFR-b	F178h	BCh	0000h
CC29IC	ESFR-b	F184h	C2h	0000h
CC30IC	ESFR-b	F18Ch	C6h	0000h
CC31IC	ESFR-b	F194h	CAh	0000h
CRIC	SFR-b	FF6Ah	B5h	0000h
CCU6_EIC	ESFR-b	F188h	C4h	--00h
CCU6_IC	ESFR-b	F120h	90h	--00h
CCU6_T12IC	ESFR-b	F190h	C8h	--00h
CCU6_T13IC	ESFR-b	F198h	CCh	--00h
I2C_DIC	ESFR-b	F186h	C3h	--00h
S0RIC	SFR-b	FF6Eh	B7h	0000h
S0TBIC	ESFR-b	F19Ch	CEh	0000h
S0TIC	SFR-b	FF6Ch	B6h	0000h
S0EIC	SFR-b	FF70h	B8h	0000h
S1RIC	ESFR-b	F18Ah	C5h	0000h
S1TBIC	ESFR-b	F13Ch	9Eh	0000h
S1TIC	ESFR-b	F182h	C1h	0000h
S1EIC	ESFR-b	F192h	C9h	0000h
SSC0EIC	SFR-b	FF76h	BBh	0000h
SSC0RIC	SFR-b	FF74h	BAh	0000h
SSC0TIC	SFR-b	FF72h	B9h	0000h
SSC1EIC	ESFR-b	F126h	93h	0000h
SSC1RIC	ESFR-b	F124h	92h	0000h
SSC1TIC	ESFR-b	F122h	91h	0000h
T0IC	SFR-b	FF9Ch	CEh	--00h
T1IC	SFR-b	FF9Eh	CFh	--00h
T2IC	SFR-b	FF60h	B0h	0000h
T3IC	SFR-b	FF62h	B1h	0000h
T4IC	SFR-b	FF64h	B2h	0000h
T5IC	SFR-b	FF66h	B3h	0000h
T6IC	SFR-b	FF68h	B4h	0000h
T7IC	ESFR-b	F17Ah	BDh	--00h
T8IC	ESFR-b	F17Ch	BEh	--00h
RTC_IC	ESFR-b	F13Ah	9Dh	0000h
RTC_IC1	ESFR-b	F150h	A8h	0000h
RTC_IC2	ESFR-b	F152h	A9h	0000h
RTC_IC3	ESFR-b	F154h	AAh	0000h
WDTIC	ESFR-b	F19Ah	CDh	0000h

Таблица А.34 – Регистр управления прерываниями

xxIC

Сброс: 0000h



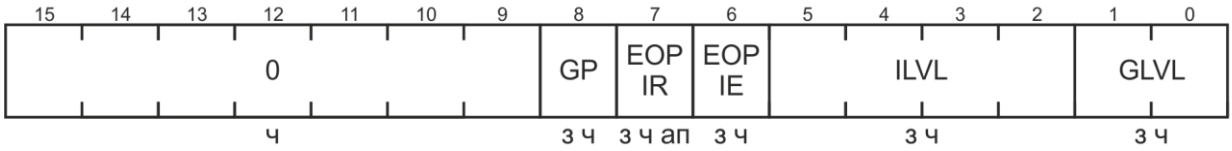
Поле	Бит	Описание	
GP	8	Бит расширения группового приоритета. Определяет значение старшего разряда уровня группы.	
IR	7	Флаг запроса прерываний	
		0	Ожидание запроса
		1	Источник выставил запрос на прерывание
		Этот бит поддерживает бит-защиту	
IE	6	Бит разрешения прерываний. Индивидуальное разрешение/запрет прерывания определенного источника	
		0	Запрос на прерывание запрещен
		1	Запрос на прерывание разрешен
ILVL	5–2	Уровень приоритета прерываний	
		1111	Наивысший уровень приоритета
		…	…
		0000	Низший уровень приоритета
GLVL	1–0	Уровень группового приоритета. Определяет внутренний приоритет в случае одновременного выставления запроса на прерывание с одинаковым приоритетом	
0	15–9	Зарезервировано	

Таблица А.35 – Регистр управления прерываниями

EOPIC

SFR-b (F180h / C0h)

Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
0	15–9	Зарезервированы	
GP	8	Расширение группового приоритета. Определяет значение старшего бита уровня группового приоритета	
EOPIR	7	Флаг запроса прерывания	
		0	Нет запроса прерывания
		1	Источник выставил запрос на прерывание
		Этот бит поддерживает бит-защиту	
EOPIE	6	Бит разрешения прерывания	
		0	Запрос прерывания разрешен
		1	Запрос прерывания запрещен

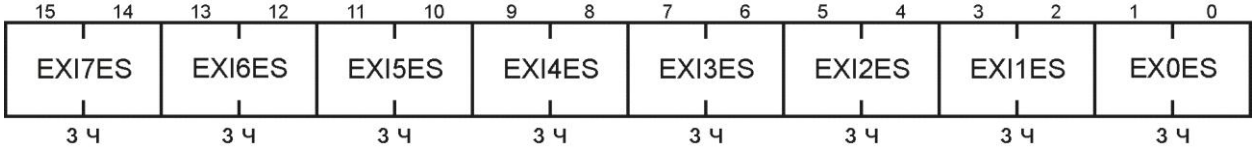
Окончание таблицы А.35

1	2	3	
ILVL	5–2	Уровень приоритета прерываний	
		1111	Наивысший уровень приоритета
		…	…
		0000	Низший уровень приоритета
GLVL	1, 0	Уровень группового приоритета	
		11	Наивысший уровень приоритета
		…	…
		00	Низший уровень приоритета
–	15–9	Зарезервировано	

Таблица А.36 – Регистр управления внешними прерываниями

EXICON

ESFR (F1C0h/ E0h) Сброс: 0000h

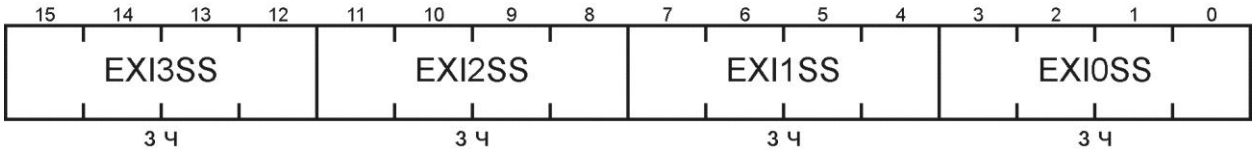


Поле	Бит	Описание	
EXIxES	15–14, 13–12, 11–10, 9–8, 7–6, 5–4, 3–2, 1–0	Поле выбора события для внешнего прерывания с номером x	
		00	Стандартный режим. Быстрые внешние прерывания запрещены
		01	Прерывание по положительному фронту
		10	Прерывание по отрицательному фронту
		11	Прерывание по любому фронту

Таблица А.37 – Регистр выбора источника для каждого внешнего прерывания

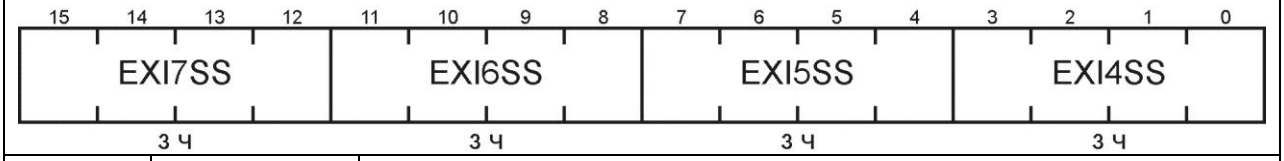
EXISEL0

ESFR (F1DAh / EDh) Сброс: 0000h



EXISEL1

ESFR (F1D8h / ECh) Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
EXIxSS	15–12, 11–8, 7–4, 3–0	Поле выбора события для внешнего прерывания x	
		0000	Вход EXxIN
		0001	Альтернативный вход AltA

Окончание таблицы А.37

1	2	3	4
EXIxSS	15–12, 11–8, 7–4, 3–0	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	

Таблица А.38 – Регистр флагов ловушек

TFR

SFR (FFACh / D6h) сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMI	STK OF	STK UF	0	0	0	0	0	UND OPC	MAC TRP	0	0	PRT FLT	ILL OPA	ILL INA	ILL BUS
з	ч	ап	з	ч	ап	з	ч	ап	ч	ч	ч	ч	ч	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	

Поле	Бит	Описание	
NMI	15	Флаг немаскируемого прерывания	
		0	Нет немаскируемого прерывания
		1	Обнаружен отрицательный перепад на выводе NMI#
STKOF	14	Флаг переполнения стека	
		0	Нет переполнения стека
		1	Обнаружено переполнение стека, текущее значение указателя стека меньше содержимого регистра STKOV
STKUF	13	Флаг опустошения стека	
		0	Нет опустошения стека
		1	Обнаружено опустошение стека, текущее значение указателя стека превышает содержимое регистра STKUN
UND OPC	7	Флаг неопределенного программного кода	
		0	Нет ошибочного программного кода
		1	Обнаружен неверный программный код, текущая декодируемая команда не является командой контроллера
MAC TRP	6	Флаг прерывания MAC	
		0	Нет прерывания MAC
		1	Обнаружено прерывание MAC
PRTFLT	3	Флаг ошибки защиты	
		0	Нет ошибки защиты
		1	Обнаружена защищенная команда в неправильном формате
ILLOPA	2	Флаг неправильного доступа к слову операнда	
		0	Нет ошибки доступа
		1	Обнаружена попытка доступа (чтения или записи) по нечетному адресу слова операнда
ILLINA	1	Флаг неправильного командного доступа	
		0	Нет ошибки командного доступа
		1	Обнаружена попытка перехода к нечетному адресу
ILLBUS	0	Флаг неправильного доступа к внешней шине	
		0	Нет ошибки доступа
		1	Обнаружена попытка внешнего доступа с неопределенной внешней шиной
0	12–8, 5, 4	Зарезервированы	

A.4 Регистры PEC

Регистры PEC представлены в таблицах A.39 – A.46.

Таблица A.39– Список регистров

Мнемоническое обозначение	Область памяти	Адрес		Сброс
PECC0	SFR	FEC0h	60h	0000h
PECC1	SFR	FEC2h	61h	0000h
PECC2	SFR	FEC4h	62h	0000h
PECC3	SFR	FEC6h	63h	0000h
PECC4	SFR	FEC8h	64h	0000h
PECC5	SFR	FECAh	65h	0000h
PECC6	SFR	FECCh	66h	0000h
PECC7	SFR	FECEh	67h	0000h
PECC8	SFR	FEE8h	74h	0000h
PECC9	SFR	FEEAh	75h	0000h
PECC10	SFR	FEECh	76h	0000h
PECC11	SFR	FEEEh	77h	0000h
PECC12	SFR	FEF8h	7Ch	0000h
PECC13	SFR	FEFAh	7Dh	0000h
PECC14	SFR	FEFCh	7Eh	0000h
PECC15	SFR	FEFEh	7Fh	0000h
PECISNCL	SFR-b	FFA6h	D3h	0000h
PECISNCH	SFR-b	FFA8h	D4h	0000h
PECSN0	SFR	FED0h	68h	0000h
PECSN1	SFR	FED2h	69h	0000h
PECSN2	SFR	FED4h	6Ah	0000h
PECSN3	SFR	FED6h	6Bh	0000h
PECSN4	SFR	FED8h	6Ch	0000h
PECSN5	SFR	FEDAh	6Dh	0000h
PECSN6	SFR	FEDCh	6Eh	0000h
PECSN7	SFR	FEDEh	6Fh	0000h
PECSN8	SFR	FEE0h	70h	0000h
PECSN9	SFR	FEE2h	71h	0000h
PECSN10	SFR	FEE4h	72h	0000h
PECSN11	SFR	FEE6h	73h	0000h
PECSN12	SFR	FEB8h	5Ch	0000h
PECSN13	SFR	FEBAh	5Dh	0000h
PECSN14	SFR	FEBCh	5Eh	0000h
PECSN15	SFR	FEBEh	5Fh	0000h
PECXC0	SFR	FEF0h	78h	0000h
PECXC2	SFR	FEF2h	79h	0000h

Таблица A.40 – Дополнительный список регистров

Мнемоническое обозначение	Адрес	Сброс	Область памяти
1	2	3	4
DSTP0	FCE2h	0000h	DPRAM
DSTP1	FCE6h	0000h	
DSTP2	FCEAh	0000h	
DSTP3	FCEEh	0000h	
DSTP4	FCF2h	0000h	

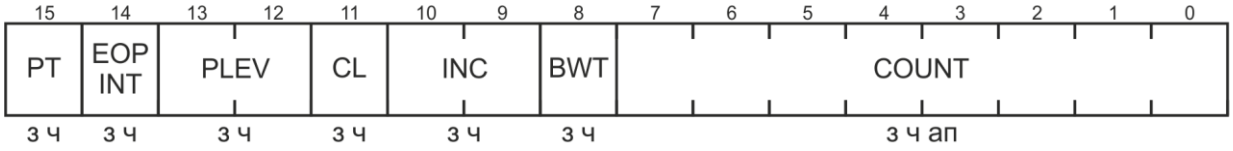
Окончание таблицы А.40

1	2	3	4
DSTP5	FCF6h	0000h	DPRAM
DSTP6	FCFAh	0000h	
DSTP7	FCFEh	0000h	
DSTP8	FCD2h	0000h	
DSTP9	FCD6h	0000h	
DSTP10	FCDAh	0000h	
DSTP11	FCDEh	0000h	
DSTP12	FCC2h	0000h	
DSTP13	FCC6h	0000h	
DSTP14	FCCAh	0000h	
DSTP15	FCCEh	0000h	
SPCP0	FCE0h	0000h	
SPCP1	FCE4h	0000h	
SPCP2	FCE8h	0000h	
SPCP3	FCECh	0000h	
SPCP4	FCF0h	0000h	
SPCP5	FCF4h	0000h	
SPCP6	FCF8h	0000h	
SPCP7	FCFCh	0000h	
SPCP8	FCD0h	0000h	
SPCP9	FCD4h	0000h	
SPCP10	FCD8h	0000h	
SPCP11	FCDCh	0000h	
SPCP12	FCC0h	0000h	
SPCP13	FCC4h	0000h	
SPCP14	FCC8h	0000h	
SPCP15	FCCCh	0000h	

Таблица А.41 – Регистр управления каналом PEC

PECCx

Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
PT	15	Режим передачи	
		0	Режим короткой передачи
		1	Режим долгой передачи
EOPINT	14	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)

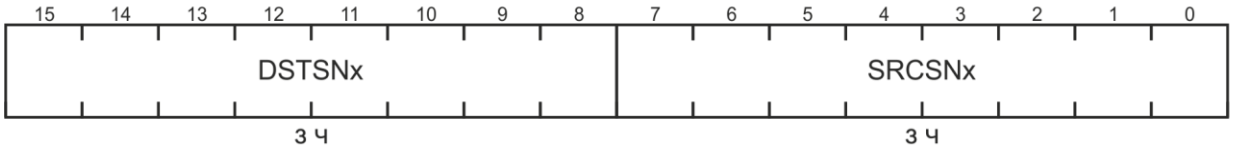
Окончание таблицы А.41

1	2	3	
PLEV	13–12	Выбор уровня приоритета PEC	
CL	11	Управление каналами PEC	
		0	Каналы PEC работают независимо
		1	Каналы PEC объединены попарно
INC	10–9	Поле управления инкрементом указателей адреса	
		00	Указатели не изменяются
		01	Инкремент указателя адреса приемника DSTPx на 1 (если BWT=1) или 2 (если BWT=0)
		10	Инкремент указателя адреса источника SRCPx на 1 (если BWT=1) или 2 (если BWT=0)
		11	Зарезервировано
BWT	8	Выбор формата данных для передачи	
		0	Слово
		1	Байт
COUNT	7–0	Счетчик передач PEC	

Таблица А.42 – Регистр указателя сегмента канала x

PECSNx

Сброс: 0000h

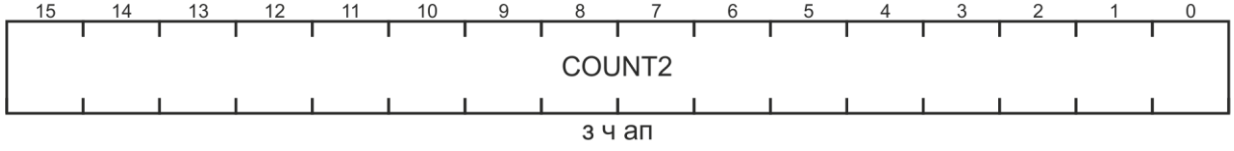


Поле	Бит	Описание
DSTSNx	15–8	Указатель сегмента адреса приемника канала x
SRCSNx	7–0	Указатель сегмента адреса источника канала x

Таблица А.43 – Регистры расширенного управления каналами PEC

PECXCx

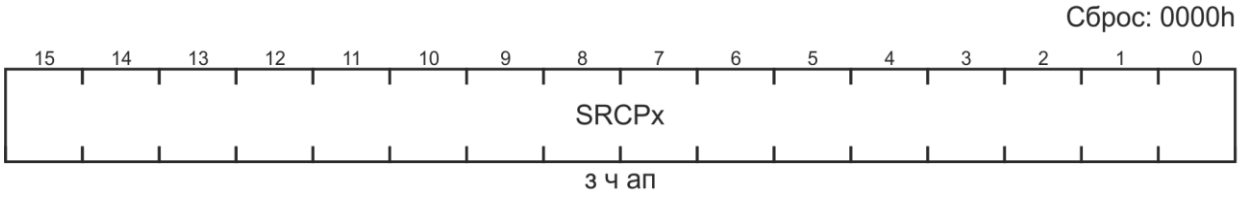
Сброс: 0000h



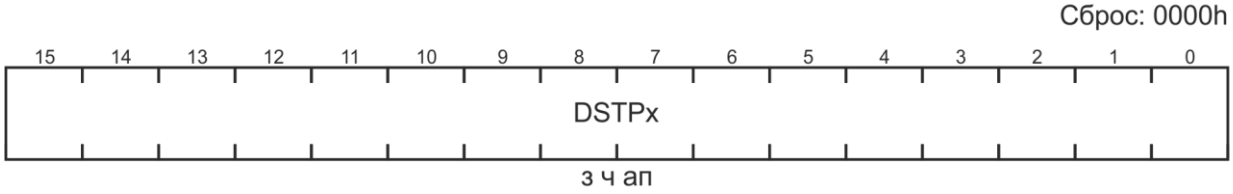
Поле	Бит	Описание
COUNT2	15–0	Счетчик расширенного управления каналом PEC

Таблица А.44 – Регистры адреса источника и приемника канала x

SRCPx



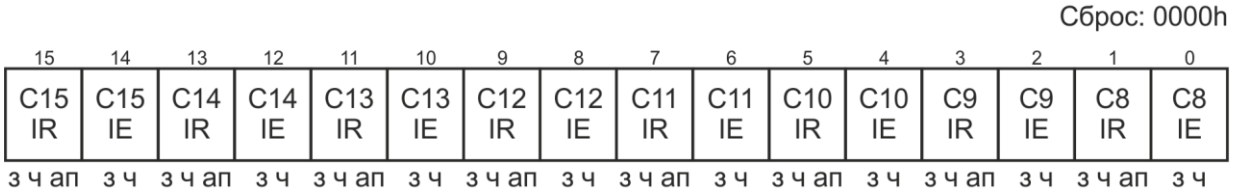
DSTPx



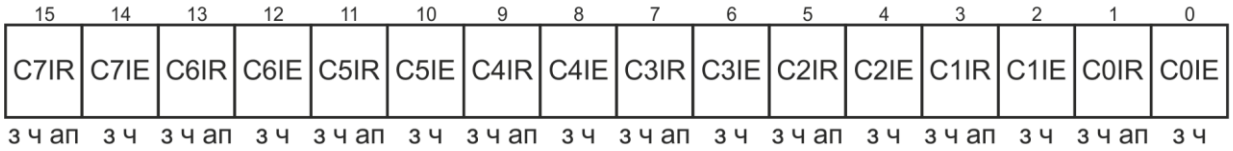
Поле	Бит	Описание
SRCPx	15–0	Адрес источника канала x
DSTPx	15–0	Адрес приемника канала x

Таблица А.45 – Регистр управления узлом прерываний канала x

PECISNCH



PECISNCL



Поле	Бит	Описание	
CxIR	15, 13, 11, 9, 7, 5, 3, 1	Флаг запроса прерывания канала x	
		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	Бит разрешения генерирования запроса на прерывание от канала x	
		0	Запрещено
		1	Разрешено генерирование запроса на прерывание по окончании обслуживания канала x

Примечание – Флаги запросов прерываний не очищаются аппаратно и должны быть очищены подпрограммой обработки прерываний.

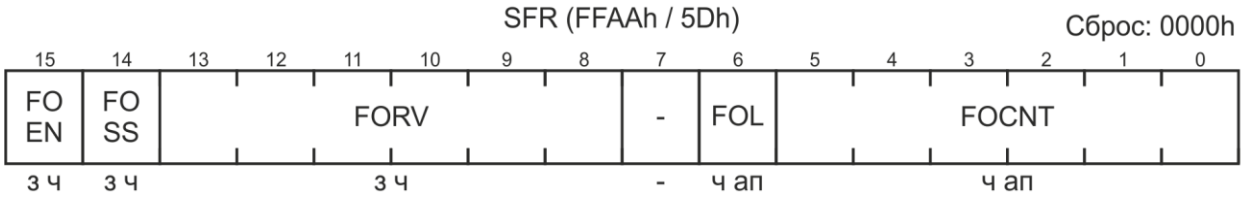
Рекомендуется производить очистку флага запроса прерываний CxIR перед разрешением соответствующего прерывания. В противном случае, ожидающие решения прежние запросы немедленно вызовут запрос на прерывание после установки бита разрешения.

A.5 Регистр генератора тактовых сигналов

Регистр генератора тактовых сигналов представлен в таблице А.46.

Таблица А.46– Регистр управления выходным тактовым сигналом

FOCON



Поле	Бит	Описание	
FOEN	15	Разрешение вывода частоты. Генератор выходной частоты выключается, когда сигнал Fout переходит в низкий уровень. FOCNT работает, Fout выводится на выход контролера. Первая перезагрузка выполнится после переключения из «0» в «1»	
FOSS	14	Выбор типа выходного сигнала:	
		0	Выходной сигнал с производительностью 50 %
		1	Производительность зависит от FORV
FORV	13–8	Значение перезагрузки счетчика выходной частоты. Копируется в FOCNT после каждого обнуления FOCNT	
FOL	6	Защелка переключателя выходной частоты. Переключается после каждого обнуления FOCNT	
FOCNT	5–0	Счетчик выходной частоты	
–	7	Зарезервировано	

Примечание – Запись в битовые поля FOCNT и FOTL регистра FOCON запрещена. Это ограничение не позволит изменить частоту тактового сигнала или остановить тактовый генератор.

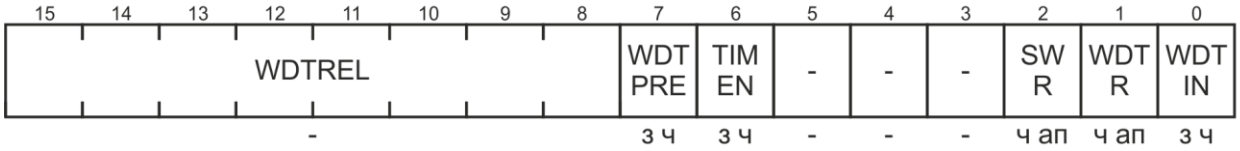
A.6 Регистр сторожевого таймера

Регистр сторожевого таймера представлен в таблицах А.47 – А.49.

Таблица А.47 – Регистр управления сторожевым таймером

WDTCON

SFR (FFAEh / D7h) Сброс: 00000000_10000xx1b



Поле	Бит	Описание
WDTREL	15–8	Значение перезагрузки сторожевого таймера
WDTPRE	7	Бит выбор делителя входной частоты. Функционирует в паре с битом WDTIN (см. таблицу А.48)
TIMEN	6	Бит разрешения таймера. Если этот бит установлен, то WDT-сброс запрещается после выполнения команды DISWDT. При переполнении сторожевого таймера генерируется запрос на прерывание.
SWR	2	Флаг программного сброса (см. таблицу А.49)
WDTR	1	Флаг WDT-сброса (см. таблицу А.49)
WDTIN	0	Бит выбор делителя входной частоты. Функционирует в паре с битом WDTPRE (см. таблицу А.48)
–	5–3	Зарезервированы

Таблица А.48 – Комбинации битов выбора делителя входной частоты

WDTPRE	WDTIN	Делитель
0	0	2
1	0	4
0	1	128
1	1	256

Таблица А.49 – Функционирование флагов программного сброса и WDT-сброса

Событие	Состояние флага	
	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

A.7 Регистры портов ввода-вывода

Регистры портов ввода-вывода представлены в таблицах А.50 – А.59.

Таблица А.50 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
ALTSEL0P1H	ESFR	F1ECh	F6h	0000h	Регистр управления альтернативными функциями выводов P1H.7 – P1H.0 порта P1
ALTSEL0P1L	ESFR	F1EAh	F5h	0000h	Регистр управления альтернативными функциями выводов P1L.7 – P1L.0 порта P1
ALTSEL0P2	ESFR	F1EEh	F7h	0000h	Регистр управления альтернативными функциями порта P2
ALTSEL0P3	ESFR	F1F0h	F8h	0000h	Регистр управления альтернативными функциями порта P3
ALTSEL0P4	ESFR	F1F4h	FAh	0000h	Регистр управления альтернативными функциями порта P4
ALTSEL0P6	ESFR	F1F6h	FBh	0000h	Регистр управления альтернативными функциями порта P6
ALTSEL0P7	ESFR	F1F8h	FCh	0000h	Регистр управления альтернативными функциями порта P7
ALTSEL0P9	ESFR	F1FCh	FEh	0000h	Регистр управления альтернативными функциями порта P9
ALTSEL1P3	ESFR	F1F2h	F9h	0000h	Дополнительный регистр управления альтернативными функциями порта P3
ALTSEL1P7	ESFR	F1FAh	FDh	0000h	Дополнительный регистр управления альтернативными функциями порта P7
ALTSEL1P9	ESFR	F1FEh	FFh	0000h	Дополнительный регистр управления альтернативными функциями порта P9
DP0H	ESFR-b	F102h	81h	0000h	Регистр задания направления выводов P0H.7 – P0H.0 порта P0
DP0L	ESFR-b	F100h	80h	0000h	Регистр задания направления выводов P0L.7 – P0L.0 порта P0
DP1H	ESFR-b	F106h	83h	0000h	Регистр задания направления выводов P1H.7 – P1H.0 порта P1
DP1L	ESFR-b	F104h	82h	0000h	Регистр задания направления выводов P1L.7 – P1L.0 порта P1
DP2	SFR-b	FFC2h	E1h	0000h	Регистр задания направления выводов порта P2
DP3	SFR-b	FFC6h	E3h	0000h	Регистр задания направления выводов порта P3
DP4	SFR-b	FFCAh	E5h	0000h	Регистр задания направления выводов порта P4
DP6	SFR-b	FFCEh	E7h	0000h	Регистр задания направления выводов порта P6
DP7	SFR-b	FFD2h	E9h	0000h	Регистр задания направления выводов порта P7
DP9	SFR-b	FFD6h	EBh	0000h	Регистр задания направления выводов порта P9

Окончание таблицы А.50

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
ODP2	ESFR-b	F1C2h	E1h	0000h	Регистр управления режимом открытого стока выводов порта P2
ODP3	ESFR-b	F1C6h	E3h	0000h	Регистр управления режимом открытого стока выводов порта P3
ODP4	ESFR-b	F1CAh	E5h	0000h	Регистр управления режимом открытого стока выводов порта P4
ODP6	ESFR-b	F1CEh	E7h	0000h	Регистр управления режимом открытого стока выводов порта P6
ODP7	ESFR-b	F1D2h	E9h	0000h	Регистр управления режимом открытого стока выводов порта P7
ODP9	ESFR-b	F1D6h	EBh	0000h	Регистр управления режимом открытого стока выводов порта P9
P0H	SFR-b	FF02h	81h	--00h	Регистр состояния выводов P0H.7 – P0H.0 порта P0
P0L	SFR-b	FF00h	80h	--00h	Регистр состояния выводов P0L.7 – P0L.0 порта P0
P1H	SFR-b	FF06h	83h	--00h	Регистр состояния выводов P1H.7 – P1H.0 порта P0
P1L	SFR-b	FF04h	82h	--00h	Регистр состояния выводов P1L.7 – P1L.0 порта P0
P2	SFR-b	FFC0h	E0h	0000h	Регистр состояния выводов порта P2
P3	SFR-b	FFC4h	E2h	0000h	Регистр состояния выводов порта P3
P4	SFR-b	FFC8h	E4h	0000h	Регистр состояния выводов порта P4
P5	SFR-b	FFA2h	D1h	0000h	Регистр состояния выводов порта P5
P6	SFR-b	FFCCh	E6h	0000h	Регистр состояния выводов порта P6
P7	SFR-b	FFD0h	E8h	0000h	Регистр состояния выводов порта P7
P9	SFR-b	FFD4h	EAh	0000h	Регистр состояния выводов порта P9

Таблица А.51 – Регистры порта 0

P0H

SFR (FF02h / 81h) Сброс: --00h

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

DP0H

ESFR (F102h / 81h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

P0L

SFR (FF00h / 80h) Сброс: --00h

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

DP0L

ESFR (F100h / 80h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то соответствующий вывод работает как вход, иначе – как выход.

3 Биты с 15 по 8 регистров не используются.

Таблица А.52 – Регистры порта 1

P1H

SFR (FF06h / 83h) Сброс: --00h

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

DP1H

ESFR (F106h / 83h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ALTSEL0P1H

ESFR (F1ECh / F6h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								CC27 OUT	CC26 OUT	CC25 OUT	CC24 OUT	SCLK 1	MTSR 1 M	MTSR 1 S	CC23 OUT
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

P1L

SFR (FF04h / 82h) Сброс: --00h

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

DP1L

ESFR (F104h / 82h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ALTSEL0P1L

ESFR (F1EAh / F5h) Сброс: --00h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								COUT 63	CC63 OUT	COUT 62	CC62 OUT	COUT 61	CC61 OUT	COUT 60	CC60 OUT
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 7 по 0 регистра ALTSEL0P1H/L являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит. Биты с 15 по 8 регистров не используются.

Таблица А.53 – Регистры порта 2

P2

SFR (FFC0h / E0h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	-							
з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап	з ч	ап

DP2

SFR (FFC2h / E1h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15 OUT EN	P14 OUT EN	P13 OUT EN	P12 OUT EN	P11 OUT EN	P10 OUT EN	P9 OUT EN	P8 OUT EN	-							
з ч	з ч	з ч	з ч	з ч	з ч	з ч	з ч	-							

ODP2

ESFR (F1C2h / E1h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15 OD EN	P14 OD EN	P13 OD EN	P12 OD EN	P11 OD EN	P10 OD EN	P9 OD EN	P8 OD EN	-							
з ч	з ч	з ч	з ч	з ч	з ч	з ч	з ч	-							

ALTSEL0P2

ESFR (F1EEh / F7h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC15 OUT	CC14 OUT	CC13 OUT	CC12 OUT	CC11 OUT	CC10 OUT	CC9 OUT	CC8 OUT	-							
з ч	з ч	з ч	з ч	з ч	з ч	з ч	з ч	-							

Примечания

1 Биты P15, P14, …, P8 являются битами состояния выводов порта.

2 Биты P15OUTEN, P14OUTEN, …, P8OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 15 по 8 регистра ODP2 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

4 Биты с 15 по 8 регистра ALTSEL0P2 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты с 8 по 0 регистров не используются.

Таблица А.54 – Регистры порта 3

P3

SFR (FFC4h / E2h) Сброс: 0000h

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

DP3

SFR (FFC6h / E3h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15 OUT EN	P14 OUT EN	P13 OUT EN	P12 OUT EN	P11 OUT EN	P10 OUT EN	P9 OUT EN	P8 OUT EN	P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
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

ODP3

ESFR (F1C6h / E3h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-		P13 OD EN	-	P11 OD EN	P10 OD EN	P9 OD EN	P8 OD EN	P7 OD EN	P6 OD EN	P5 OD EN	P4 OD EN	P3 OD EN	P2 OD EN	P1 OD EN	P0 OD EN
-		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

ALTSEL0P3

ESFR (F1F0h / F8h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK OUT	-	SCLK 0	WR H#	RXD0 OUT	TXD0	MTSR 0 M	MTSR 0 S			-		T3 OUT	-	T6 OUT	TXD1
З Ч	-	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч			-		З Ч	-	З Ч	З Ч

ALTSEL1P3

ESFR (F1F2h / F9h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FOUT	-		P12			-				TXD1		-		T6 OUT	-
З Ч	-		З Ч			-				З Ч		-		З Ч	-

Примечания

1 Биты P15, P14, …, P0 являются битами состояния выводов порта.

2 Биты P15OUTEN, P14OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты 13 и с 11 по 0 регистра ODP3 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

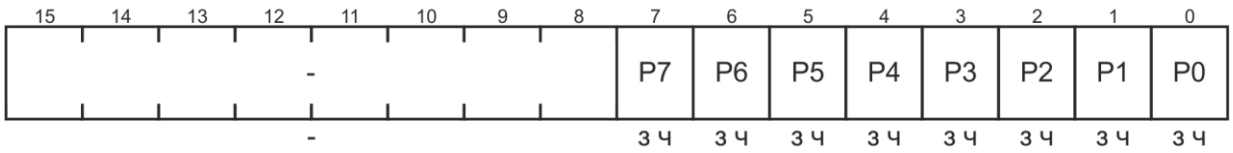
4 Биты регистров ALTSEL0P3 и ALTSEL1P3 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты регистров, отмеченные символом «-», не используются.

Таблица А.55 – Регистры порта 4

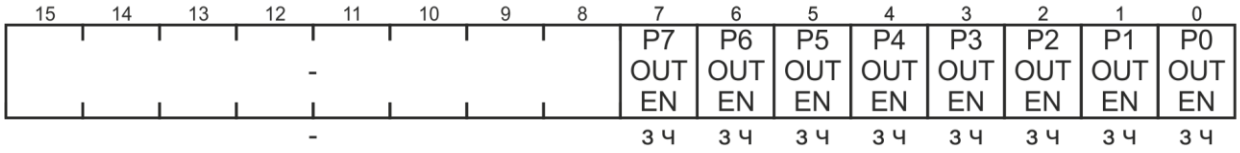
P4

SFR (FFC8h / E4h) Сброс: 0000h



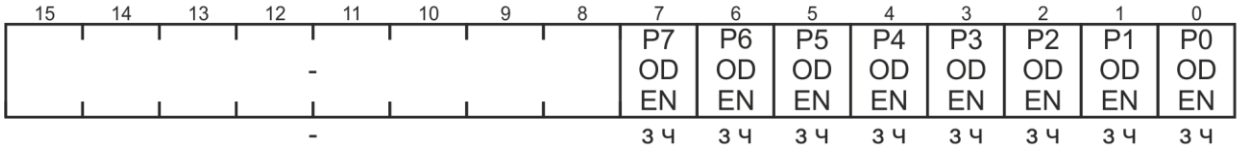
DP4

SFR (FFCAh / E5h) Сброс: 0000h



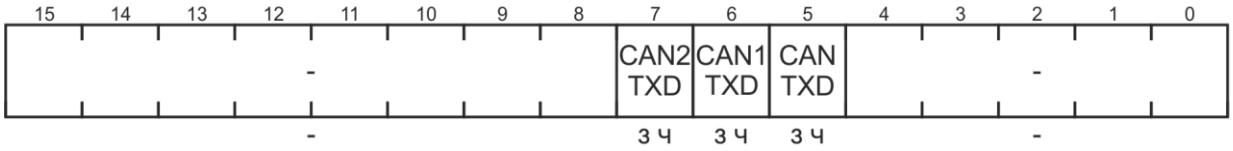
ODP4

ESFR (F1CAh / E5h) Сброс: 0000h



ALTSEL0P4

ESFR (F1F4h / FAh) Сброс: 0000h



Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 7 по 0 регистра ODP4 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

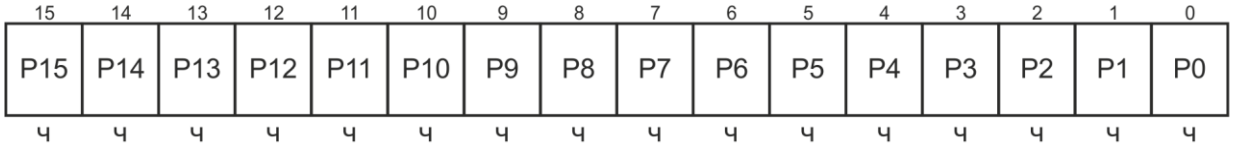
4 Биты с 7 по 5 регистра ALTSEL0P4 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты регистров, отмеченные символом «-», не используются.

Таблица А.56– Регистр порта 5

P5

SFR (FFA2h / D1h) Сброс: 0000h



Примечание – Биты P15, P14, …, P0 являются битами состояния выводов порта.

Таблица А.57 – Регистры порта 6

P6

SFR (FFCCh / E6h) Сброс: 0000h

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

DP6

SFR (FFCEh / E7h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ODP6

ESFR (F1CEh / E7h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OD EN	P6 OD EN	P5 OD EN	P4 OD EN	P3 OD EN	P2 OD EN	P1 OD EN	P0 OD EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ALTSEL0P6

ESFR (F1F6h / FBh) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								CC7OUT	CC6 OUT	CC5 OUT	CC4 OUT	CC3 OUT	CC2 OUT	CC1 OUT	CC0 OUT
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 7 по 0 регистра ODP6 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

4 Биты с 7 по 5 регистра ALTSEL0P6 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты регистров, отмеченные символом «-», не используются.

Таблица А.58 – Регистры порта 7

P7

SFR (FFD0h / E8h) Сброс: 0000h

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

DP7

SFR (FFD2h / E9h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ODP7

ESFR (F1D2h / E9h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OD EN	P6 OD EN	P5 OD EN	P4 OD EN	P3 OD EN	P2 OD EN	P1 OD EN	P0 OD EN
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

ALTSEL0P7

ESFR (F1F8h / FCh) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								CAN1 TXD	CAN TXD	CAN2 TXD	-	CAN1 TXD	CAN2 TXD	CAN TXD	-
-								З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	-

ALTSEL1P7

ESFR (F1FAh / FDh) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								CC31 OUT	CC30 OUT	CC29 OUT	CC20 OUT		-		
-								З Ч	З Ч	З Ч	З Ч		-		

Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 7 по 0 регистра ODP6 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

4 Биты регистров ALTSEL0P7 и ALTSEL1P7 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты регистров, отмеченные символом «-», не используются.

Таблица А.59– Регистры порта 9

P9

SFR (FFD4 h / EA h) Сброс: 0000h

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	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

DP9

SFR (FFD6 h / EB h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OUT EN	P6 OUT EN	P5 OUT EN	P4 OUT EN	P3 OUT EN	P2 OUT EN	P1 OUT EN	P0 OUT EN
-								з ч	з ч	з ч	з ч	з ч	з ч	з ч	з ч

ODP9

ESFR (F1D6 h / EB h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								P7 OD EN	P6 OD EN	P5 OD EN	P4 OD EN	P3 OD EN	P2 OD EN	P1 OD EN	P0 OD EN
-								з ч	з ч	з ч	з ч	з ч	з ч	з ч	з ч

ALTSEL0P9

ESFR (F1FC h / FE h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-										SCL 2	SDA 2	SCL 1	SDA 1	SCL 0	SDA 0
-										з ч	з ч	з ч	з ч	з ч	з ч

ALTSEL1P9

ESFR (F1FE h / FF h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-										CC21 OUT	CC20 OUT	CAN1 TXD	CC18 OUT	CAN2 TXD	CC16 OUT
-										з ч	з ч	з ч	з ч	з ч	з ч

Примечания

1 Биты P7, P6, …, P0 являются битами состояния выводов порта.

2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.

3 Биты с 7 по 0 регистра ODP9 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

4 Биты регистров ALTSEL0P9 и ALTSEL1P9 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит.

5 Биты регистров, отмеченные символом «-», не используются.

A.8 Регистры блока RTC

Регистры блока RTC представлены в таблицах A.60 – A.70.

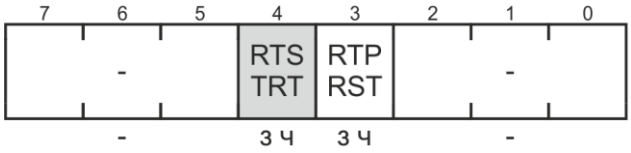
Таблица A.60 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
RTCCMP1_H	ESFR	F042h	21h	0000h	Регистр сравнения 1, старшие разряды
RTCCMP1_L	ESFR	F044h	22h	0200h	Регистр сравнения 1, младшие разряды
RTCCMP2_H	ESFR	F046h	23h	FFFFh	Регистр сравнения 2, старшие разряды
RTCCMP2_L	ESFR	F048h	24h	FFFFh	Регистр сравнения 2, младшие разряды
RTCCMP3_H	ESFR	F04Ah	25h	FFFFh	Регистр сравнения 3, старшие разряды
RTCCMP3_L	ESFR	F04Ch	26h	FFFFh	Регистр сравнения 3, младшие разряды
RTCCST	ESFR	F030h	18h	--00h	Регистр управления и статуса
RTCEIST	ESFR	F034h	1Ah	--00h	Регистр состояния прерываний
RTCIEN	ESFR	F036h	1Bh	--00h	Регистр разрешения прерываний
RTCLD_H	ESFR	F03Eh	1Fh	0000h	Перезагружаемый регистр, старшие разряды
RTCLD_L	ESFR	F040h	20h	0000h	Перезагружаемый регистр, младшие разряды
RTCRD_H	ESFR	F03Ah	1Dh	0000h	Регистр реального времени, старшие разряды
RTCRD_L	ESFR	F03Ch	1Eh	0001h	Регистр реального времени, младшие разряды
RTPRD	ESFR	F038h	1Ch	0000h	Регистр прескалера
RTUDST	ESFR	F032h	19h	--00h	Регистр состояния обновлений

Таблица A.61 – Регистр управления и состояния

RTCCST

ESFR (F030h /18h) сброс: 00h

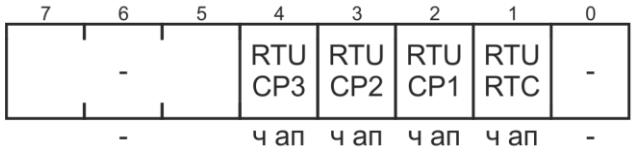


Поле	Биты	Описание	
RTSTRT	4	Запуск часов реального времени. Установка бита включает счетчик импульсов и счетчик реального времени. Бит сбрасывается только при Power-On-Reset одновременно с остановкой счетчиков и инициализацией регистров модуля RTC	
		0	Счетчик выключен
		1	Счетчик включен
RTPRST	3	Перезапуск счетчика импульсов	
		0	Нет действий
		1	Если бит установлен, то счетчик при обнаружении очередного импульса сигнала тактирования обнуляется. Одновременно с этим бит аппаратно сбрасывается
–	7–5, 2–0	Зарезервировано	

Таблица А.62 – Регистр ожиданий загрузок

RTUDST

ESFR (F032h /19h) сброс: 00h

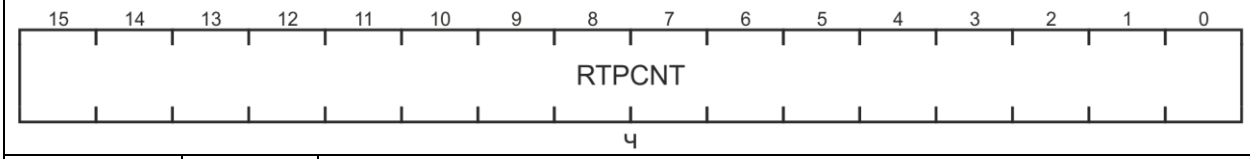


Поле	Биты	Описание	
RTUCP3	4	Бит ожидания загрузки в регистр RTCCMP3. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	Есть данные для загрузки
RTUCP2	3	Бит ожидания загрузки в регистр RTCCMP2. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	Есть данные для загрузки
RTUCP1	2	Бит ожидания загрузки в регистр RTCCMP1. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	Есть данные для загрузки
RTURTC	1	Бит ожидания загрузки в счетчик реального времени. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	Есть данные для загрузки
–	7–5, 0	Зарезервировано	

Таблица А.63 – Регистр счетчика импульсов

RTPRD

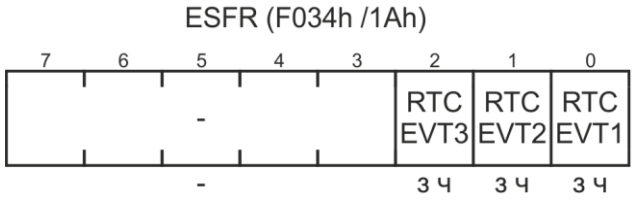
ESFR (F038h /1Ch) только Power-On-Reset: 0000h



Поле	Биты	Описание
RTPCNT	15–0	Значение 16-разрядного счетчика импульсов

Таблица А.64 – Регистр совпадений

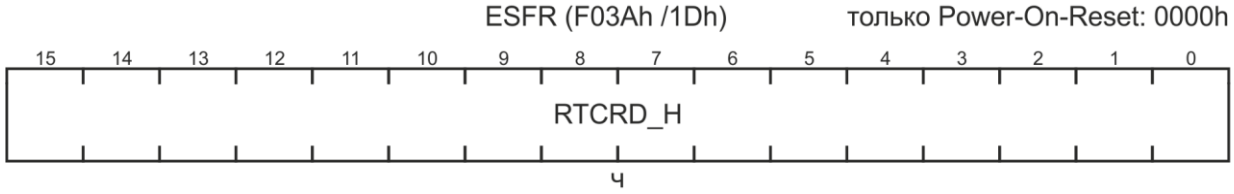
RTCEIST



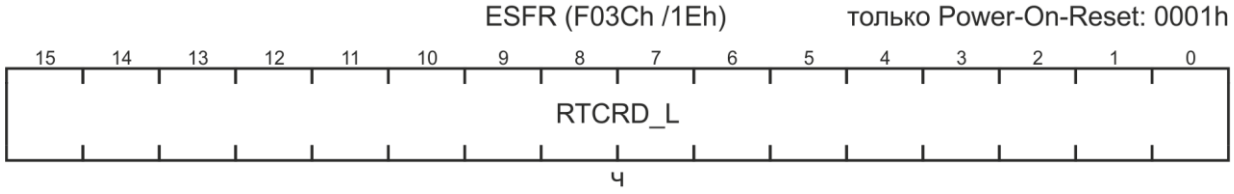
Поле	Биты	Описание	
RTCEVT3	2	Флаг совпадения значения регистра RTCCMP3 со значением счетчика реального времени. Бит сбрасывается программно, записью в него единицы	
		0	Нет совпадения
		1	Совпадение обнаружено
RTCEVT2	1	Флаг совпадения значения регистра RTCCMP2 со значением счетчика реального времени. Бит сбрасывается программно, записью в него единицы	
		0	Нет совпадения
		1	Совпадение обнаружено
RTCEVT1	0	Флаг совпадения суммы значения поля RTCCMP1 и константы 7DFFh со значением счетчика импульсов. Бит сбрасывается программно, записью в него единицы	
		0	Нет совпадения
		1	Совпадение обнаружено
–	7–3	Зарезервировано	

Таблица А.65 – Регистры старшего и младшего слов счетчика реального времени

RTCRD_H



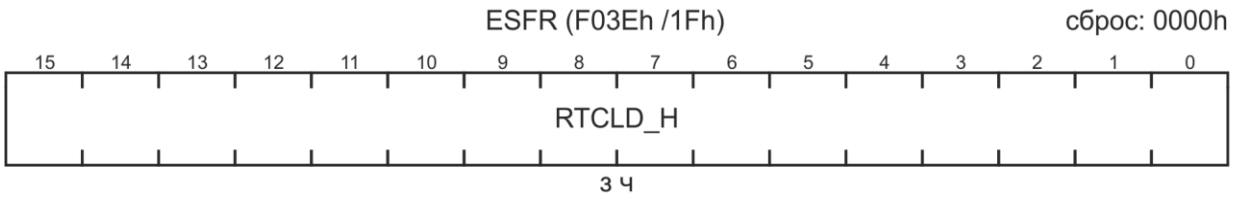
RTCRD_L



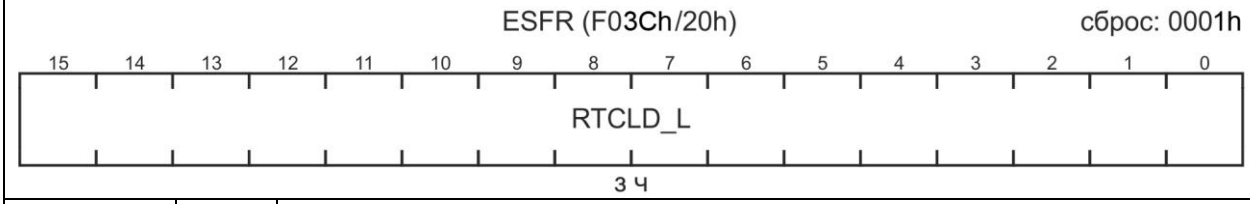
Поле	Биты	Описание
RTCRD_H	15–0	Значение старшего слова 32-разрядного счетчика реального времени
RTCRD_L	15–0	Значение младшего слова 32-разрядного счетчика реального времени

Таблица А.66 – Регистры загрузки старшего и младшего слов счетчика реального времени

RTCLD_H



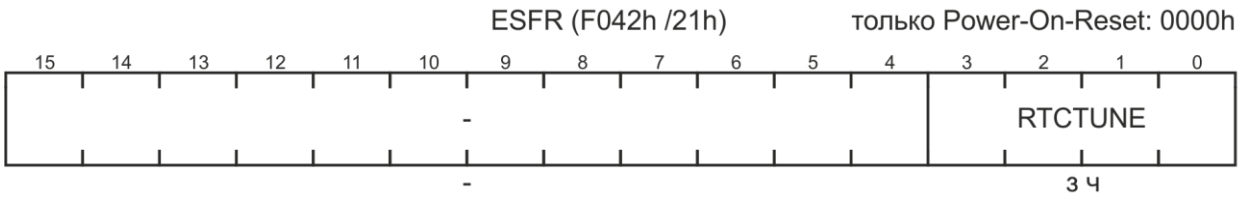
RTCRD_L



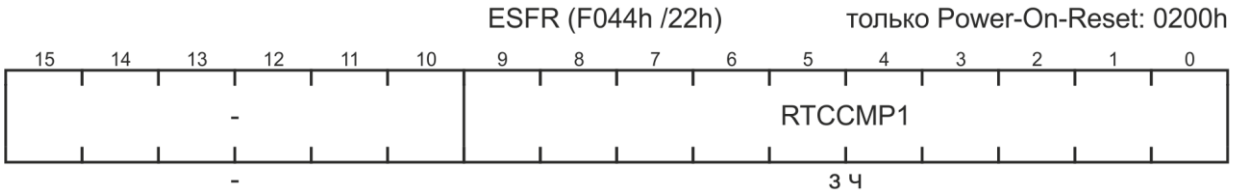
Поле	Биты	Описание
RTCLD_H	15–0	Значение для загрузки в старшее слово счетчика реального времени
RTCLD_L	15–0	Значение для загрузки в младшее слово счетчика реального времени

Таблица А.67 – Первый регистр сравнения (старшее и младшее слова)

RTCCMP1_H



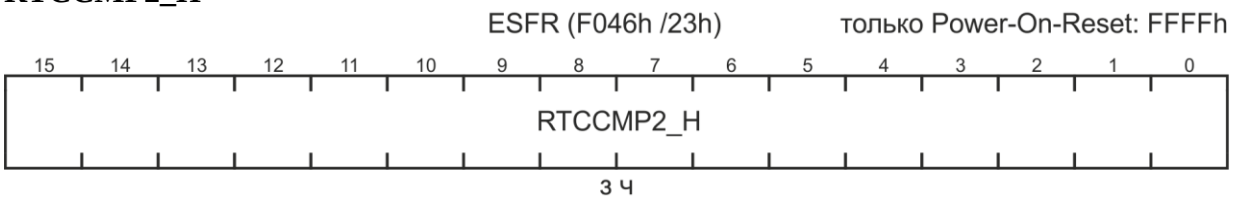
RTCCMP1_L



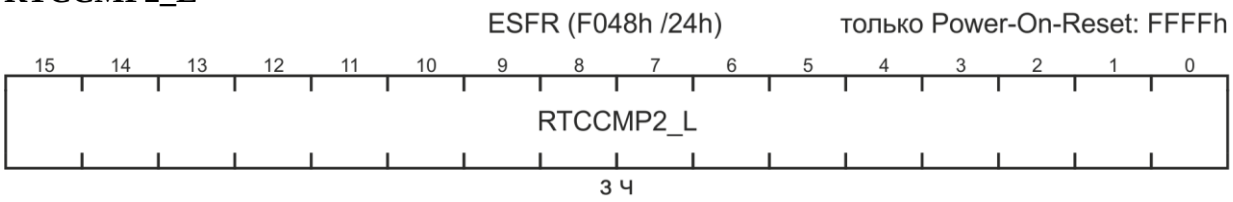
Поле	Биты	Описание
RTCTUNE	3–0	Поле задания количества шагов подстройки
RTCCMP1	9–0	Поле задания значения компенсации при настройке модуля RTC
–	15–10/ 15–4	Зарезервировано

Таблица A.68 – Второй регистр сравнения (старшее и младшее слова)

RTCCMP2_H



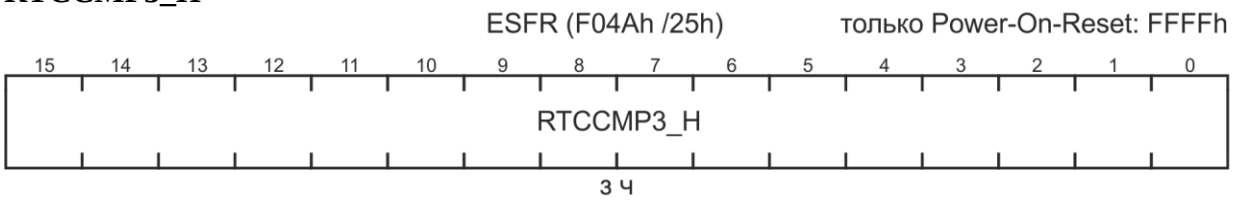
RTCCMP2_L



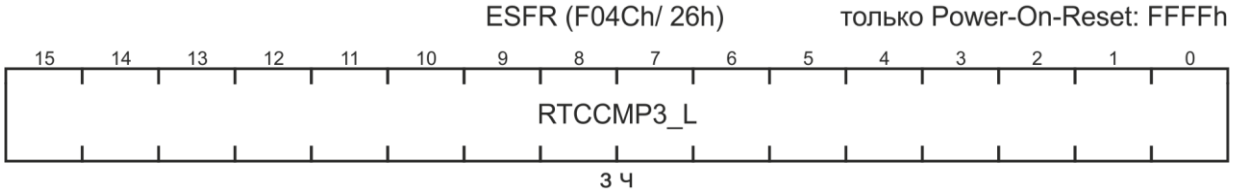
Поле	Биты	Описание
RTCCMP2_H	15–0	Значение для загрузки в старшее слово регистра сравнения RTCCMP2
RTCCMP2_L	15–0	Значение для загрузки в младшее слово регистра сравнения RTCCMP2

Таблица A.69 – Третий регистр сравнения (старшее и младшее слова)

RTCCMP3_H



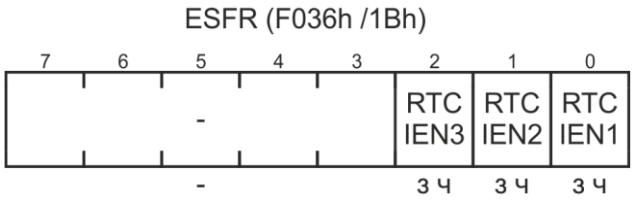
RTCCMP3_L



Поле	Биты	Описание
RTCCMP3_H	15–0	Значение для загрузки в старшее слово регистра сравнения RTCCMP3
RTCCMP3_L	15–0	Значение для загрузки в младшее слово регистра сравнения RTCCMP3
–	7–3	Зарезервировано

Таблица А.70 – Регистр разрешения прерываний

RTCIEN



Поле	Биты	Описание	
RTCIEN3	2	Бит разрешения прерывания после установки бита RTCEVT3	
		0	Запрещено
		1	Разрешено
RTCIEN2	1	Бит разрешения прерывания после установки бита RTCEVT2	
		0	Запрещено
		1	Разрешено
RTCIEN1	0	Бит разрешения прерывания после установки бита RTCEVT1	
		0	Запрещено
		1	Разрешено
–	7–3	Зарезервировано	

A.9 Регистры таймеров общего назначения (GPT)

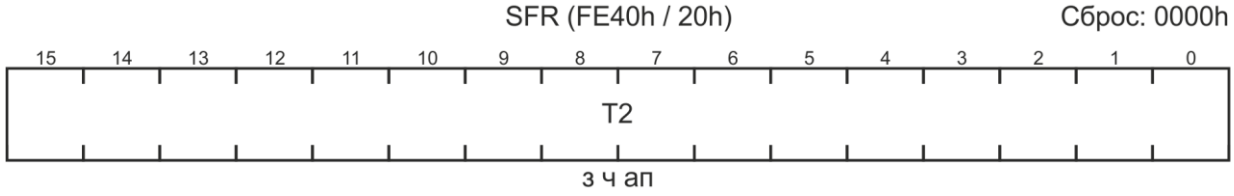
Регистры таймеров общего назначения GPT представлены в таблицах А.71 – А.84.

Таблица А.71 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
CAPREL	SFR	FE4Ah	25h	0000h	Регистр захвата/перезагрузки
GPTID	SFR-b	FFE6h	F3h	58XXh	Регистр идентификации
T2	SFR	FE40h	20h	0000h	Регистр таймера 2
T2CON	SFR-b	FF40h	A0h	0000h	Регистр управления таймера 2
T3	SFR	FE42h	21h	0000h	Регистр таймера 3
T3CON	SFR-b	FF42h	A1h	0000h	Регистр управления таймера 3
T4	SFR	FE44h	22h	0000h	Регистр таймера 4
T4CON	SFR-b	FF44h	A2h	0000h	Регистр управления таймера 4
T5	SFR	FE46h	23h	0000h	Регистр таймера 5
T5CON	SFR-b	FF46h	A3h	0000h	Регистр управления таймера 5
T6	SFR	FE48h	24h	0000h	Регистр таймера 6
T6CON	SFR-b	FF48h	A4h	0000h	Регистр управления таймера 6

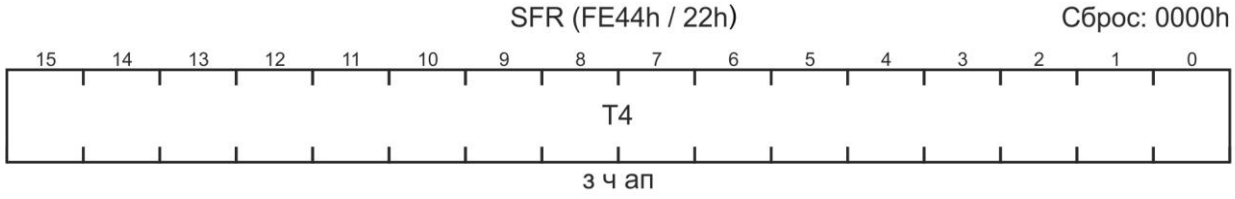
Таблица А.72 – Регистры таймеров T2, T3, T4

T2



T3

T4

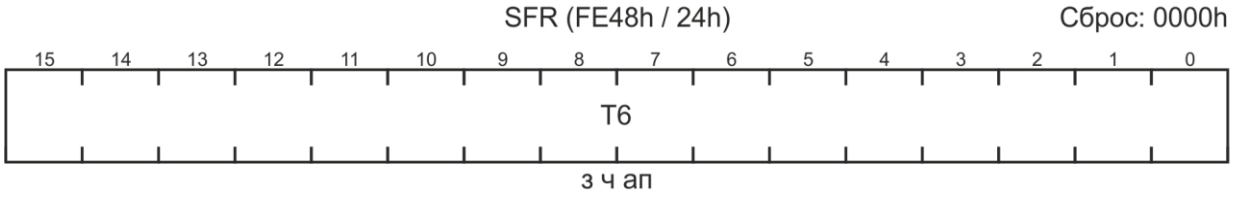


Поле	Бит	Описание
T2/T3/T4	15–0	Текущее значение таймера T2/T3/T4

Таблица А.73 – Регистры таймеров T5, T6

T5

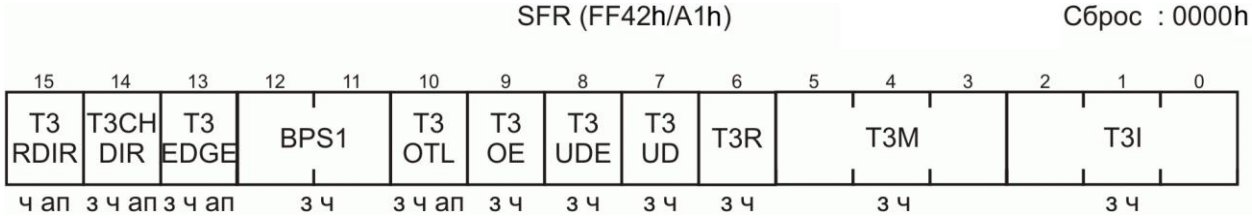
T6



Поле	Бит	Описание
T5/T6	15–0	Текущее значение таймера T5/T6

Таблица А.74 – Регистр управления таймера T3

T3CON



Поле	Бит	Описание	
1	2	3	
T3RDIR	15	Бит направления счета таймера	
		0	Таймер считает «вверх» (инкрементирование)
		1	Таймер считает «вниз» (декрементирование)
T3CHDIR	14	Флаг изменения направления счета таймера. Этот бит устанавливается каждый раз при изменении направления счета	
		0	Изменения направления счета не обнаружено
		1	Обнаружено изменение направления счета
		Бит должен сбрасываться программно	
T3EDGE	13	Флаг обнаружения фронта сигнала на входе таймера. Флаг устанавливается при каждом обнаружении корректного фронта входного сигнала	
		0	Фронт сигнала не обнаружен
		1	Фронт сигнала обнаружен
		Бит должен очищаться программно	
BPS1	12–11	Делитель частоты. Дополнительная информация – в таблице А.75	
		00	fosc/8
		01	fosc/4
		10	fosc/32
		11	fosc/16

Окончание таблицы А.74

1	2	3	
T3OTL	10	Бит выходной защелки таймера. Этот бит переключается каждый раз при переполнении/ опустошении таймера T3. Бит может быть установлен/сброшен программно	
T3OE	9	Бит разрешения вывода сигнала о переполнении/опустошении таймера	
		0	Переполнение/опустошение таймера не детектируется извне
		1	Переполнение/опустошение таймера детектируется по сигналу на линии T3OUT
T3UDE	8	Бит разрешения внешнего управления направлением счета	
		0	Направление счета таймера контролируется программно
		1	Направление счета таймера контролируется посредством линии T3EUD
T3UD	7	Бит установки направления счета таймера	
		0	Таймер считает «вверх»
		1	Таймер считает «вниз»
T3R	6	Бит работы таймера	
		0	Таймер/счетчик остановлен
		1	Таймер/счетчик работает
T3M	5–3	Выбор режима	
		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)
T3I	2–0	Выбор режима тактирования таймера T3. Для режимов таймера и внешнего управления комбинации – в таблице А.75. Для режима счетчика комбинации в таблице А.76. Для режима внешнего инкрементирования – по двум входам комбинации в таблице А.77	

Таблица А.75 – Режимы таймера и внешнего управления таймером Tx (x = 2, 3, 4)

TxI	Делитель для fosc			
	BPS1 = 00b	BPS1 = 01b	BPS1 = 10b	BPS1 = 11b
000b	8	4	32	16
001b	16	8	64	32
010b	32	16	128	64
011b	64	32	256	128
100b	128	64	512	256
101b	256	128	1024	512
110b	512	256	2048	1024
111b	1024	512	4096	2048

Таблица А.76 – Режим счетчика Tx (x = 2, 3, 4)

TxI	Ожидаемый фронт входного сигнала для инициирования переключения таймера
X00b	Счетчик Tx отключен
001b	Положительный фронт на входе TxIN
010b	Отрицательный фронт на входе TxIN
011b	Положительный и отрицательный фронты на входе TxIN
101b	Положительный фронт на входе T3OTL
110b	Отрицательный фронт на входе T3OTL
111b	Положительный и отрицательный фронты на входе T3OTL

Таблица А.77 – Режим внешнего инкрементирования T3

T3I	Выбор фронта для положительного или отрицательного счета
000b	Счетчик не считает
001b	Любой фронт (положительный или отрицательный) на T3IN
010b	Любой фронт (положительный или отрицательный) на T2EUD
011b	Любой фронт (положительный или отрицательный) на любом из входов T3IN и T3EUD
1XXb	Зарезервировано. Не использовать

Таблица А.78 – Регистр управления таймером 2 и 4

T2CON

SFR (FF40h / A0h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T2 RDIR	T2 CH DIR	T2 EDG E	T2 IR DIS	-		T2 RC	T2 UDE	T2 UD	T2R	T2M			T2I		
ч ап	з ч ап	з ч ап	з ч	-		з ч	з ч	з ч	з ч	з ч			з ч		

T4CON

SFR (FF44h / A2h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T4 RDIR	T4 CH DIR	T4 EDG E	T4 IR DIS	-		T4 RC	T4 UDE	T4 UD	T4R	T4M			T4I		
ч ап	з ч ап	з ч ап	з ч	-		з ч	з ч	з ч	з ч	з ч			з ч		

Поле	Бит	Описание	
1	2	3	
T2RDIR T4RDIR	15	Индикатор направления счета таймера	
		0	Таймер считает вверх
		1	Таймер считает вниз
T2CHDIR T4CHDIR	14	Индикатор изменения направления счета таймера. Этот бит устанавливается каждый раз при изменении направления счета таймера	
		0	Нет изменения направления счета
		1	Направления счета изменилось
		Бит должен очищаться программно	
T2EDGE T4EDGE	13	Индикатор обнаружения фронта входного сигнала таймера	
		0	Фронт не обнаружен
		1	Фронт обнаружен
		Бит должен очищаться программно	

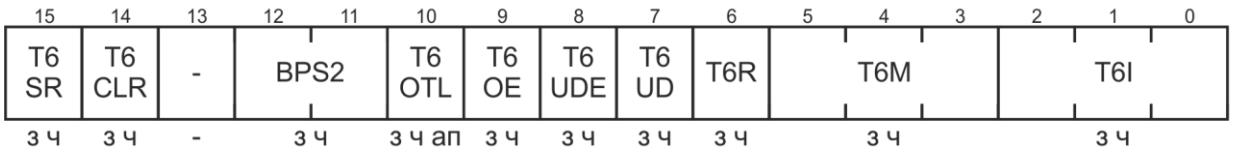
Окончание таблицы А.78

1	2	3	
T2IRDIS T4IRDIS	12	Бит запрета прерываний таймера	
		0	Формирование запросов прерываний T2CHDIR/T4CHDIR и T2EDGE/T4EDGE в режиме внешнего инкрементирования разрешено
		1	Формирования запросов прерываний запрещено
T2RC T4RC	9	Бит выбора управления включением таймера	
		0	Таймер/счетчик включается собственным битом работы T2R/T4R
		1	Таймер/счетчик включается битом основного таймера Т3
T2UDE T4UDE	8	Бит разрешения внешнего управления направлением счета таймера	
		0	Направление счета таймера контролируется программно
		1	Направление счета таймера контролируется посредством линии T2EUD/T4EUD
T2UD T4UD	7	Бит установки направления счета таймера (при T2UDE/T4UDE = 0)	
		0	Таймер считает вверх
		1	Таймер считает вниз
T2R T4R	6	Бит запуска таймера	
		0	Остановлен
		1	Запущен
T2M T4M	5–3	Поле задания режима таймера	
		000	Таймер
		001	Счетчик
		010	Внешнее управление активным низким уровнем
		011	Внешнее управления активным высоким уровнем
		100	Перезагрузка
		101	Захват
		110	Внешнее инкрементирование (детектирование периодов)
		111	Внешнее инкрементирование (детектирование фронтов)
T2I T4I	2–0	Поле задания режима тактирования таймера. Режим таймера: комбинации в таблице А.75. Режим внешнего управления таймером: комбинации в таблице А.76. Режим счетчика: комбинации в таблице А.76. Режим внешнего инкрементирования: комбинации в таблице А.77	
–	11–10	Зарезервировано. Не использовать	

Таблица А.79 – Регистр управления таймером 6

T6CON

SFR (FF48h / A4h) Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
T6SR	15	Бит разрешения загрузи таймера T6:	
		0	Загрузка значения из регистра CAPREL запрещена
		1	Загрузка значения из регистра CAPREL разрешена
T6CLR	14	Бит очистки таймера T6:	
		0	Таймер T6 не очищается при захвате
		1	Таймер T6 очищается после захвата
BPS2	12–11	Делитель блока таймеров GPT2: Максимальная входная частота сигнала тактирования[1)]	
		00	fosc/4
		01	fosc/2
		10	fosc/16
		11	fosc/8
T6OTL	10	Бит выходной защелки таймера T6. Этот бит переключается каждый раз при переполнении/ опустошении таймера T6. Бит может быть установлен/сброшен программно	
T6OE	9	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
T6UDE	8	Бит разрешения внешнего управления направлением счета таймера T6:	
		0	Направление счета таймера контролируется программно
		1	Направление счета таймера контролируется посредством линии T6EUD
T6UD	7	Бит установки направления счета таймера T6 (если T6UDE = 0b):	
		0	Таймер считает «вверх» (инкрементирование)
		1	Таймер считает «вниз» (декрементирование)
T6R	6	Бит работы таймера T6:	
		0	Таймер/счетчик остановлен
		1	Таймер/счетчик работает
T6M	5–3	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

Окончание таблицы А.79

1	2	3
T6I	2–0	Выбор режима тактирования таймера T6. Режим таймера: комбинации в таблице А.80. Режим внешнего управления таймером: комбинации в таблице А.80. Режим счетчика: комбинации в таблице А.81
–	13	Зарезервировано

———————

[1)] Дополнительно частота входного сигнала тактирования может быть изменена (см. T6I) для режима таймера, режима счетчика и режима внешнего управления

Таблица А.80 – Режимы таймера и внешнего управления таймером Tx (x = 5, 6)

Tx	Делитель для fosc			
	BPS2= 00b	BPS2 = 01b	BPS2 = 10b	BPS2 = 11b
000b	4	2	16	8
001b	8	4	32	16
010b	16	8	64	32
011b	32	16	128	64
100b	64	32	256	128
101b	128	64	512	256
110b	256	128	1024	512
111b	512	256	2048	1024

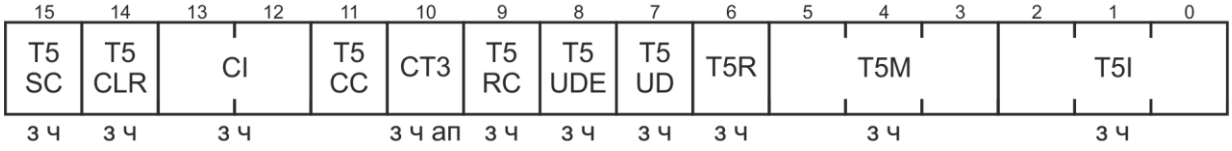
Таблица А.81 – Режим счетчика T6

T6I	Ожидаемый фронт входного сигнала для инициирования переключения таймера
X00b	Счетчик отключен
001b	Положительный фронт на входе T6IN
010b	Отрицательный фронт на входе T6IN
011b	Положительный и отрицательный фронты на входе T6IN
1XXb	Зарезервировано. Не использовать

Таблица А.82 – Регистр управления таймером 5

T5CON

SFR (FF46h / A3h) Сброс: 0000h



Поле	Бит	Описание	
1	2	3	
T5SC	15	Бит разрешения захвата содержимого таймера T5:	
		0	Захват содержимого таймера T5 в регистр CAPREL запрещен
		1	Захват содержимого таймера T5 в регистр CAPREL разрешен
T5CLR	14	Бит очистки таймера T5:	
		0	Таймер T5 не сбрасывается после захвата его содержимого
		1	Таймер T5 сбрасывается после захвата его содержимого
CI	13–12	Выбор события на линии таймера T3 для инициации захвата содержимого таймера T5 (в зависимости от бита CT3):	
		00	Захват запрещен

Окончание таблицы А.82

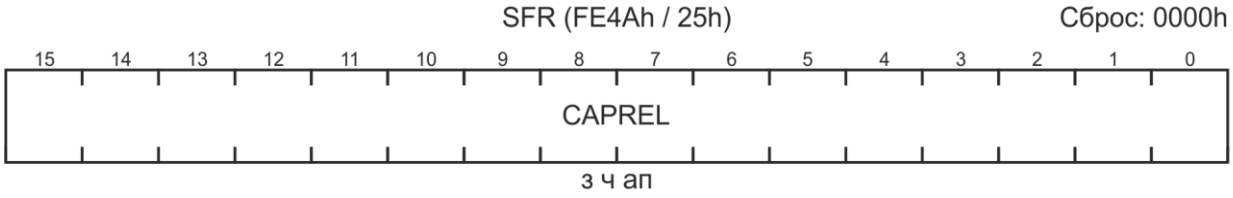
1	2	3	
CI		01	Передний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3IN
		10	Задний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3EUD
		11	Любой фронт сигнала на входе CAPIN или любой фронт сигнала на входах T3IN или T3EUD
T5CC	11	Бит коррекции при захвате содержимого таймера T5:	
		0	Содержимое таймера T5 захватывается без изменений
		1	Содержимое таймера T5 декрементируется на 1 перед захватом
CT3	10	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
T5RC	9	Бит удаленного контроля таймером Tx:	
		0	Таймер/счетчик T5 управляется собственным битом работы T5R
		1	Захват содержимого таймера T5 по сигналу на линии T3IN и/или T3EUD
T5UDE	8	Бит разрешения внешнего управления направлением счета таймера T5:	
		0	Направление счета таймера контролируется программно
		1	Направление счета таймера контролируется посредством линии T5EUD
T5UD	7	Бит установки направления счета таймера T5 (когда T5UDE = 0b):	
		0	Таймер считает «вверх» (инкрементирование)
		1	Таймер считает «вниз» (декрементирование)
T5R	6	Бит работы таймера T5:	
		0	Таймер/счетчик остановлен
		1	Таймер/счетчик работает
T5M	5–3	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
T5I	2–0	Выбор режима тактирования таймера T5. Режим счетчика: комбинации в таблице А.83.	

Таблица А.83 – Режим счетчика T5

T5I	Ожидаемый фронт входного сигнала для инициирования переключения таймера
X00b	Счетчик T5 отключен
001b	Положительный фронт на входе T5IN
010b	Отрицательный фронт на входе T5IN
011b	Положительный и отрицательный фронты на входе T5IN
101b	Положительный фронт на входе T6OTL
110b	Отрицательный фронт на входе T6OTL
111b	Положительный и отрицательный фронты на входе T6OTL

Таблица А.84 – Регистр захвата/перезагрузки

CAPREL



Поле	Бит	Описание
CAPREL	15–0	Значение регистра захвата/перезагрузки

A.10 Регистры асинхронно/синхронных последовательных интерфейсов ASC0 и ASC1

Регистры асинхронно/синхронных последовательных интерфейсов ASC0 и ASC1 представлены в таблицах А.85 – А.95.

Таблица А.85 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
ASC0ID	SFR-b	FFE2h	F1h	44XXh	Регистр идентификации ASC0
S0BG	SFR	FEB4h	5Ah	0000h	Регистр таймера генератора скорости передачи ASC0
S0CON	SFR-b	FFB0h	D8h	0000h	Регистр управления ASC0
S0FDV	SFR	FEB6h	5Bh	0000h	Регистр дробного делителя ASC0
S0RBUF	SFR	FEB2h	59h	0000h	Буферный регистр приемника ASC0
S0TBUF	SFR	FEB0h	58h	0000h	Буферный регистр передатчика ASC0
S1BG	SFR	FEAAh	55h	0000h	Регистр таймера генератора скорости ASC1
S1CON	ESFR-b	F1B8h	DCh	0000h	Регистр управления ASC1
S1FDV	SFR	FEACh	56h	0000h	Регистр дробного делителя ASC1
S1RBUF	SFR	FEA8h	54h	0000h	Буферный регистр приемника ASC1
S1TBUF	SFR	FEA6h	53h	0000h	Буферный регистр передатчика ASC1

Таблица А.86 – Регистр управления ASC0 и ASC1

S0CON

SFR-b (FFB0h / D8h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LB	BRS	ODD	FDE	OE	FE	PE	OEN	FEN/ RXDI	PEN	REN	STP	M		
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		

S1CON

ESFR-b (F1B8h / DCh) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LB	BRS	ODD	FDE	OE	FE	PE	OEN	FEN/ RXDI	PEN	REN	STP	M		
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		

Поле	Бит	Описание	
1	2	3	
R	15	Бит работы генератора baudrate	
		0	Генератор baudrate отключен (ASC выключен)
		1	Генератор baudrate включен
LB	14	Бит разрешения режима loopback	
		0	Режим стандартной передачи/получения данных
		1	Включен режим loopback
BRS	13	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

Окончание таблицы А.86

1	2	3	
ODD	12	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)
FDE	11	Бит включения дробного делителя	
		0	Дробный делитель выключен
		1	Дробный делитель включен и используется как делитель для генератора скорости передачи
OE	10	Флаг ошибки переполнения. Устанавливается аппаратными средствами при ошибке переполнения, если OEN = 1b. Должен быть очищен программно	
FE	9	Флаг ошибки фрейма. Устанавливается аппаратными средствами при ошибке фрейма, если OEN = 1b. Должен быть очищен программно	
PE	8	Флаг ошибки четности. Устанавливается аппаратными средствами при ошибке четности, если FEN = 1b. Должен быть очищен программно	
OEN	7	Проверка ошибки переполнения	
		0	Игнорировать ошибки переполнения
		1	Проверять ошибки переполнения
FEN/RXDI	6	Проверка ошибки фрейма (только асинхронный режим)	
		0	Игнорировать ошибки фрейма
		1	Проверять ошибки фрейма
PEN	5	Бит включения проверки четности. Все асинхронные режимы.	
		0	Игнорировать проверку четности
		1	Проверять четность
REN	4	Бит разрешения приема	
		0	Прием запрещен
		1	Прием разрешен
STP	3	Выбор количества стоповых бит	
		0	Один стоповый бит
		1	Два стоповых бита
M	2–0	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

Таблица А.87 – Формулы для определения скорости пересылки данных при использовании фиксированного тактового сигнала

FDE	BRS	BG	Формула
0	0	0, …, 8191	$baudrate = \frac{f_{CLK}}{32 \times (BG + 1)}$ (A.1) $BG = \frac{f_{CLK}}{32 \times baudrate} - 1$ (A.2)
	1		$baudrate = \frac{f_{CLK}}{48 \times (BG + 1)}$ (A.3) $BG = \frac{f_{CLK}}{48 \times baudrate} - 1$ (A.4)

Таблица А.88 – Типовые скорости пересылки данных при соответствующих значениях регистров перезагрузки и отклонения скоростей

Скорость пересылки (baudrate)	BRS = 0b, f_{CLK} = 20 МГц		BRS = 1b, f_{CLK} = 20 МГц	
	Ошибка отклонения, %	Перезагружаемое значение	Ошибка отклонения, %	Перезагружаемое значение
10,200 Кбод	0,44/–1,18	003CH / 003DH	0,12/–0,37	0027H / 0028H
9600 бод	0,16/–1,36	0040H / 0041H	0,94/–1,36	002AH / 002BH
4800 бод	0,16/–0,6	0081H / 0082H	0,94/–0,22	0055H / 0056H
2400 бод	0,16/–0,22	0103H / 0104H	0,35/–0,22	00ACH / 00ADH
1200 бод	0,16/–0,03	0207H / 0208H	0,06/–0,22	015AH / 015BH

Таблица А.89 – Формулы расчета скорости пересылки данных при использовании дробного делителя

FDE	BRS	BG	FDV	Формула
1	–	1, …, 8191	1, …, 511	$baudrate = \frac{FDV}{512} \times \frac{f_{CLK}}{16 \times (BG + 1)}$ (A.5)
			0	$baudrate = \frac{f_{CLK}}{16 \times (BG + 1)}$ (A.6)

Таблица А.90 – Типовые скорости пересылки данных при использовании дробного делителя и возможные отклонения

f_{CLK}, МГц	Требуемая скорость, Кбод	BG	FDV	Итоговая скорость, Кбод	Отклонение, %
20	115,2	9h	1D8h	115,234	0,02
	57,6	14h	1EFh	57,547	0,09
	38,4	1Fh	1F7h	38,376	0,06
	19,2	40h	1FFh	19,193	0,03

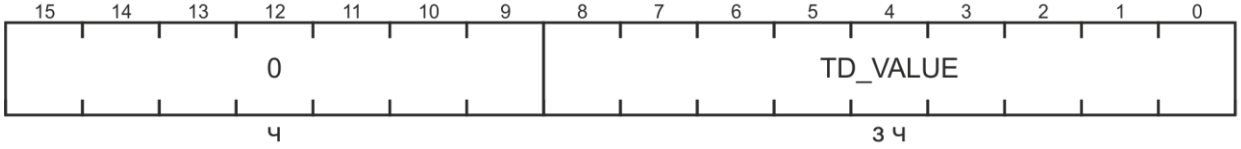
Таблица А.91 – Формулы для определения скорости передачи данных (baudrate) в синхронном режиме

BRS	BG	Формула
0	1, …, 8191	$baudrate = \frac{f_{CLK}}{8 \times (BG + 1)}$ (A.7) $baudrate = \frac{f_{CLK}}{8 \times (BG + 1)} - 1$ (A.8)
1		$baudrate = \frac{f_{CLK}}{12 \times (BG + 1)}$ (A.9) $baudrate = \frac{f_{CLK}}{12 \times (BG + 1)} - 1$ (A.10)

Таблица А.92 – Буферный регистр передатчика ASC0 и ASC1

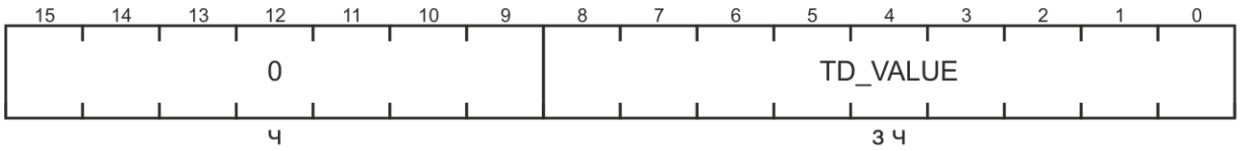
S0TBUF

SFR (FEB0h / 58h) Сброс: 0000h



S1TBUF

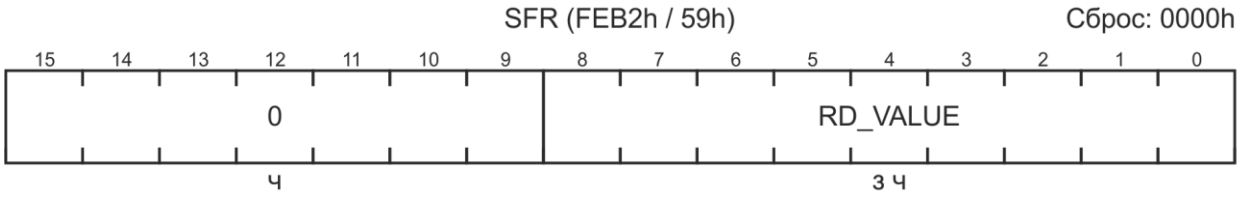
SFR (FEA6h / 53h) Сброс: 0000h



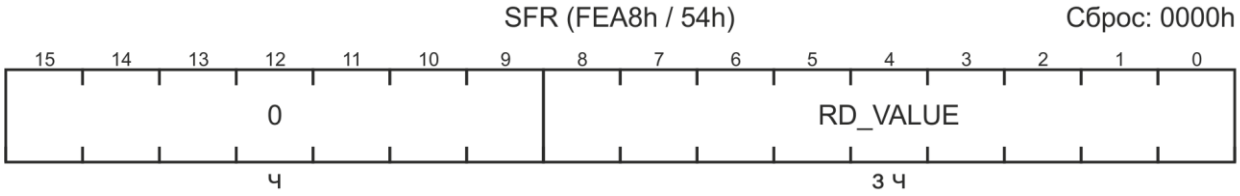
Поле	Бит	Описание
0	15–9	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна
TD_VALUE	8–0	Значение данных буферного регистра передатчика. Содержит данные, которые будут переданы в синхронном или асинхронном режимах ASC. Передача с двойной буферизацией

Таблица А.93 – Буферный регистр приемника ASC0 и ASC1

S0RBUF



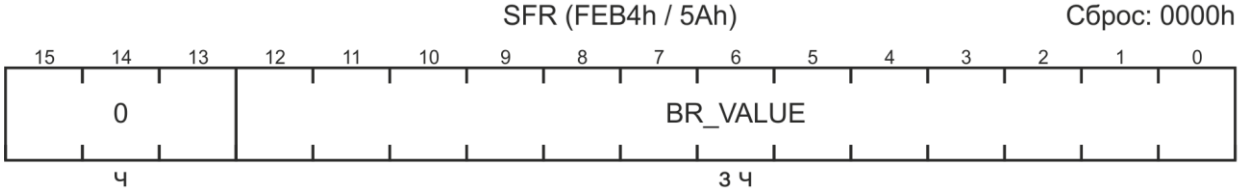
S1RBUF



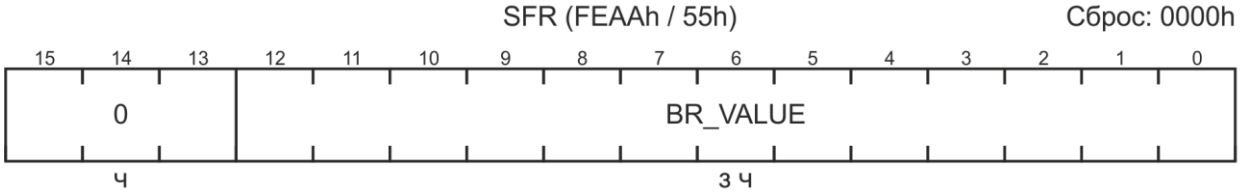
Поле	Бит	Описание
0	15–9	Зарезервировано. При чтении возвращает значение «0». Запись в эти биты не эффективна
RD_VALUE	8–0	Значение данных буферного регистра приемника. Содержит полученные данные и, в зависимости от выбранного режима, бит четности. В асинхронном режиме с CON_M = 011b (7-битные данные 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

Таблица А.94 – Регистр таймера генератора скорости передачи ASC0 и ASC1

S0BG



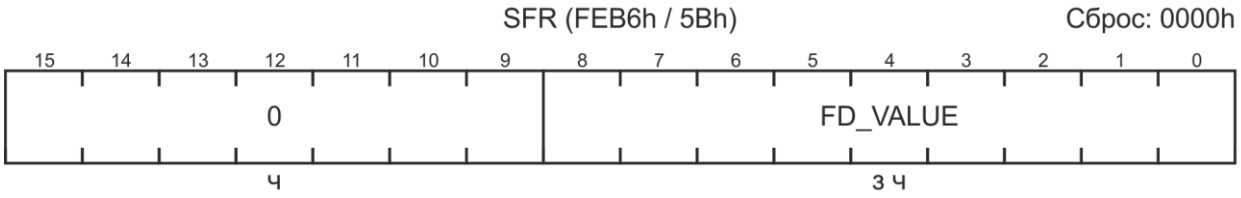
S1BG



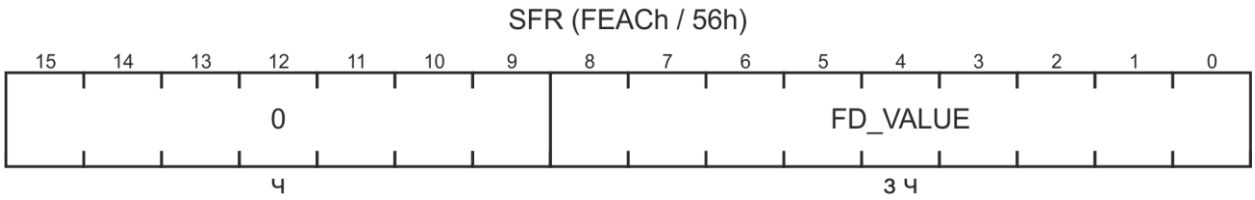
Поле	Бит	Описание
BR_VALUE	12–0	Значение перезагружаемых данных. Чтение возвращает 13-битовое содержимое регистра Запись загружает значение в таймер. Примечание – BG должен быть записан только при CON_R = 0b
–	15–13	Зарезервировано

Таблица А.95 – Регистр дробного делителя ASC0 и ASC1

S0FDV



S1FDV



Поле	Бит	Описание
FD_VALUE	8–0	Значение дробного делителя. Определяет дробное отношение n/512, где n = 0, …, 512. При n = 0 дробный делитель выключен: $f_{DIV} = f_{CLK}$
–	15–9	Зарезервировано

А.11 Регистры синхронных последовательных интерфейсов SSC0 и SSC1

Регистры синхронных последовательных интерфейсов SSC0 и SSC1 представлены в таблицах А.96 – А.102.

Таблица А.96 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
SSC0BR	ESFR	F0B4h	5Ah	0000h	Регистр скорости пересылки
SSC0CON	SFR-b	FFB2h	D9h	0000h	Регистр управления
SSC0ID	SFR-b	FFE4h	F2h	45XXh	Регистр идентификации
SSC0RB	ESFR	F0B2h	59h	0000h	Буферный регистр приемника
SSC0TB	ESFR	F0B0h	58h	0000h	Буферный регистр передатчика
SSC1BR	ESFR	F05Eh	2Fh	0000h	Регистр скорости пересылки
SSC1CON	SFR-b	FF5Eh	AFh	0000h	Регистр управления
SSC1RB	ESFR	F05Ch	2Eh	0000h	Буферный регистр приемника
SSC1TB	ESFR	F05Ah	2Dh	0000h	Буферный регистр передатчика

Таблица А.97 – Регистр управления

SSC0CON (режим программирования EN = 0)

SFR (FFB2h / D9h) Сброс: 0000h

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			

SSC1CON

SFR (FF5Eh / AFh) Сброс: 0000h

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			

Поле	Бит	Описание	
1	2	3	
EN	15	Бит разрешения работы SSC = 0b. Передача и прием данных отключены, имеется доступ к битам управления	
MS	14	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
0	13	Зарезервирован. При чтении возвращает значение «0». Всегда должен записываться как «0»	
AREN	12	Бит разрешения автоматического перезапуска блока SSC	
		0	Не требуется дополнительных действий при определении ошибки скорости передачи
		1	Автоматический перезапуск SSC при определении ошибки скорости передачи

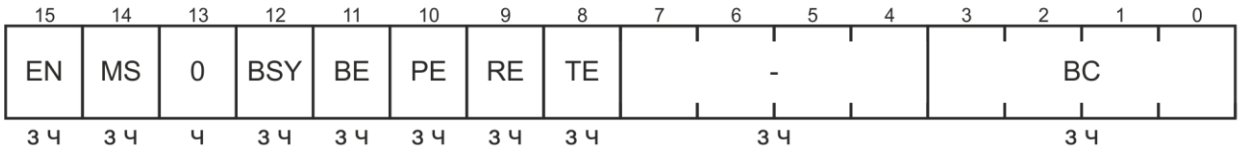
Окончание таблицы А.97

1	2	3	
BEN	11	Бит разрешения определения ошибки скорости передачи	
		0	Игнорировать ошибки скорости передачи
		1	Проверять ошибки скорости передачи
PEN	10	Бит разрешения определения ошибки фазы	
		0	Игнорировать ошибки фазы
		1	Проверять ошибки фазы
REN	9	Бит разрешения определения ошибки приема	
		0	Игнорировать ошибки приема
		1	Проверять ошибки приема
TEN	8	Бит разрешения определения ошибки передачи	
		0	Игнорировать ошибки передачи
		1	Проверять ошибки передачи
LB	7	Бит управления режимом loopback	
		0	Нормальный выход
		1	Вход приемника подключен к выходу передатчика (полудуплексный режим)
PO	6	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»
PH	5	Бит управления фазой тактового сигнала	
		0	Передача по переднему фронту, захват по заднему фронту
		1	Передача по заднему фронту, захват по переднему фронту
HB	4	Бит управления типом заголовка	
		0	Первый передается/принимается LSB
		1	Первый передается/принимается MSB
BM	3–0	Выбор ширины данных	
		0000	Зарезервировано
		0001	–
		1111	Ширина передаваемых данных 2, …, 16 бит (BM + 1).

Таблица А.98 – Регистр управления

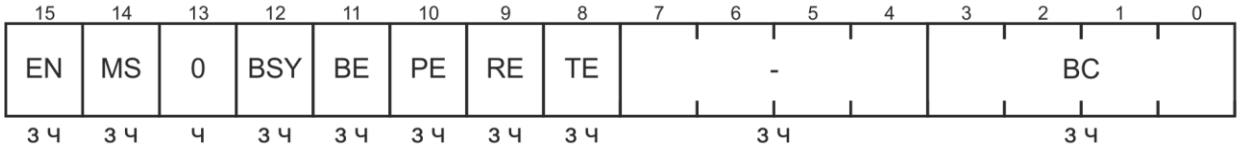
SSC0CON (операционный режим EN = 1)

SFR (FFB2h / D9h) Сброс: 0000h



SSC1CON

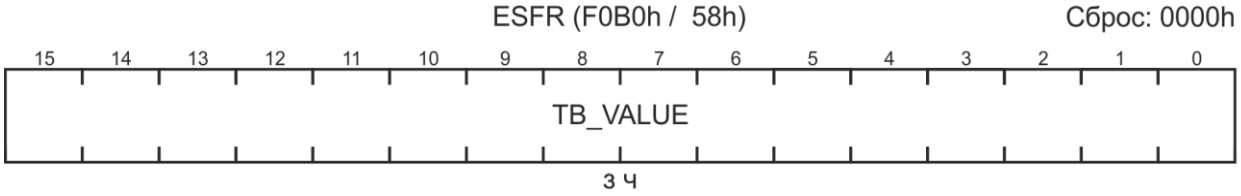
SFR (FF5Eh / AFh) Сброс: 0000h



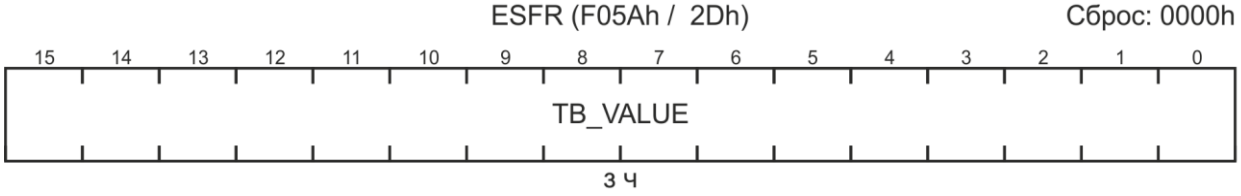
Поле	Бит	Описание	
1	2	3	
EN	15	Бит разрешения работы SSC = 1b. Передача и прием данных разрешены, доступны флаги состояния и управление M/S	
MS	14	Бит выбора режима master	
		0	Режим slave, работа с внешним тактовым сигналом на входе SCLK
		1	Режим master, генерация собственного тактового сигнала и вывод его через SCLK
BSY	12	Флаг занятости. Устанавливается во время передачи Примечание – Не производить запись в это поле!	
BE	11	Флаг ошибки скорости передачи	
		0	Нет ошибки
		1	Фактическая и ожидаемая скорость передачи данных отличается больше, чем диапазон (0,5 – 2) раз
PE	10	Флаг ошибки фазы	
		0	Нет ошибки
		1	Принимаемые данные изменяются во время считывания значения
RE	9	Флаг ошибки приема	
		0	Нет ошибки
		1	Прием был завершен до того, как приемный буфер был прочитан
TE	8	Флаг ошибки передачи	
		0	Нет ошибки
		1	Передача началась, но при этом не было изменено значение в slave буфере передачи
BC	3–0	Поле отсчета битов. Изменяет свое значение на единицу после каждого переданного бита. Примечание – Не производить запись в это поле!	
–	13, 7–4	Зарезервирован. При чтении возвращает значение «0». Всегда должен записываться как «0»	

Таблица А.99 – Буферный регистр передатчика

SSC0TB



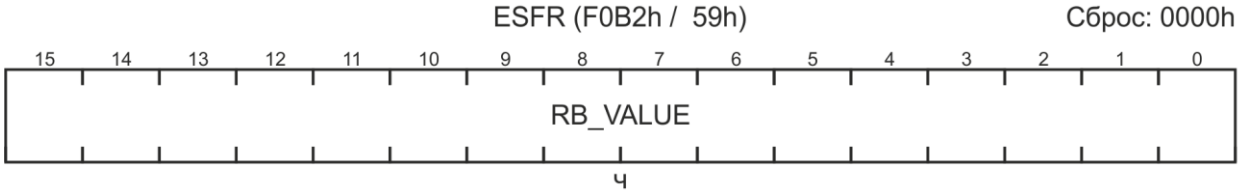
SSC1TB



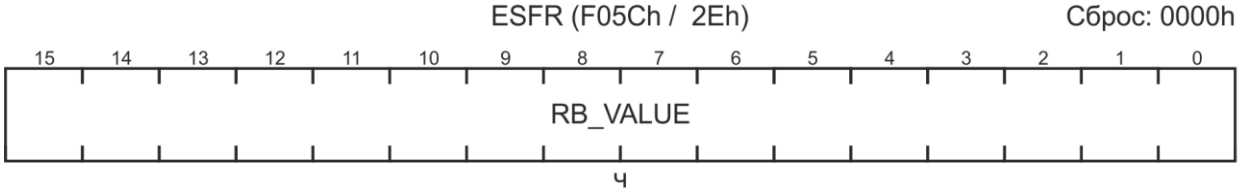
Поле	Бит	Описание
TB_VALUE	15 – 0	Содержит значение данных, которое будет передано. Невыбранные биты SSCxTB игнорируются во время передачи

Таблица А.100 – Буферный регистр приемника

SSC0RB



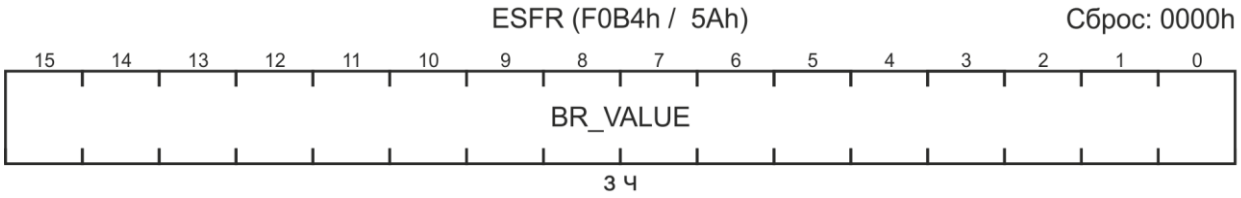
SSC1RB



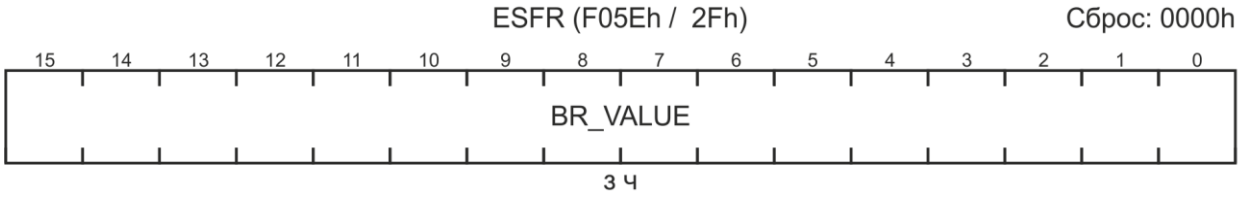
Поле	Бит	Описание
RD_VALUE	15–0	Содержит значение принятых данных. Невыбранные биты SSCxRB являются недействительными и должны быть проигнорированы

Таблица А.101 – Регистр скорости пересылки

SSC0BR



SSC1BR



Поле	Бит	Описание
BR_VALUE	15–0	Чтение возвращает записанное ранее значение перезагрузки. Запись загружает в таймер скорости передачи перезагружаемое значение регистра BR_VALUE

$$\text{baudrate} = \frac{f_{CLK}}{2 \times (
 + 1)} \qquad \text{(A.11)}$$

$$BR = \frac{f_{CLK}}{2 \times \text{baudrate}} - 1. \qquad \text{(A.12)}$$

Таблица А.102 – Типовые скорости пересылки данных SSCx (f_{MC_CLK/SS_CLK} = 20 МГц)

Значение перезагрузки	Скорость пересылки f_{MC_CLK/SS_CLK}	Отклонение, %
0013h	500 Кбод	0
0031h	200 Кбод	0
0063h	100 Кбод	0
FFFFh	152,59 бод	0

A.12 Регистры контроллера интерфейса I2C

Регистры контроллера интерфейса I2C представлены в таблицах A.103 – A.111.

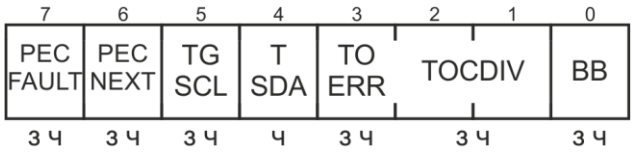
Таблица A.103 – Список регистров

Мнемоническое обозначение	Область памяти	Адрес		Сброс
SMBADDR	ESFR	F0E2h	71h	00h
SMBCST	ESFR	F0DEh	6Fh	00h
SMBCTRL1	ESFR	F0E0h	70h	00h
SMBCTRL2	ESFR	F0E4h	72h	00h
SMBCTRL3	ESFR	F0E8h	74h	00h
SMBSDA	ESFR	F0DAh	6Dh	XXh
SMBST	ESFR	F0DCh	6Eh	00h
SMBTOPR	ESFR	F0E6h	73h	00h

Таблица A.104 – Регистр управления и статуса

SMBCST

ESFR (F0DEh / 6Fh) Сброс: 00h



Поле	Бит	Тип	Описание
1	2	3	4
PECFAULT	7	Запись Чтение	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной суммой, значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Запись Чтение	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, что следующий передаваемый байт будет байтом CRC (байт контрольной суммы). Реакция на установку бита PECNEXT зависит от режима работы. В режиме мастера передатчика установка бита PECNEXT вызовет загрузку результата вычисления CRC в регистр SMBSDA. После сброса флага INT начнется передача байта CRC. В режиме приемника установка этого бита будет указывать логике управления на то, что следующий байт, который будет принят, будет байтом CRC. В режиме ведомого приемника модуль I2C автоматически будет квитировать или не квитировать прием байта CRC, в зависимости от того, будет ли выявлена ошибка пакета данных или нет. В режиме мастера приемника по окончании приема байта CRC, будет отправлено значение бита ACK регистра SMBCTRL1

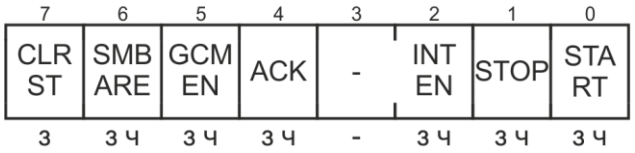
Окончание таблицы А.104

1	2	3	4
TGSCL	5	Запись Чтение	Бит переключения SCL. Бит позволяет переключать вывод SCL во время восстановления после ошибки. Когда на выводе SDA – низкий уровень сигнала, запись «1» в бит TGSCL переключит вывод SCL на один такт. Когда на SDA высокий уровень сигнала, запись «1» в бит TGSCL игнорируется. Бит очищается аппаратно по окончании такта
TSDA	4	Чтение	Бит тестирования SDA. Содержит текущее значение SDA. Этот бит можно использовать для отслеживания окончания процесса восстановления после ошибки, в течение которого ведомый постоянно поддерживает низкий уровень сигнала на выводе SDA
TOERR	3	Запись Чтение	Флаг ошибки простоя на шине. Если TOERR = 1b, это указывает на то, что на линии SCL был обнаружен простой. Флаг TOERR выставляется по обнулению основного счетчика времени простоя и может быть сброшен записью «1» в бит CLRST регистра SMBCTRL1
TOCDIV	2–1	Запись Чтение	Поле коэффициента делителя. Устанавливает коэффициент деления системного тактового сигнала, подаваемого на предделитель времени простоя линии SCL
			00 – Тактовый сигнал отсутствует
			01 – Деление на 4
			10 – Деление на 8
			11 – Деление на 16
BB	0	Запись Чтение	Флаг занятости шины. Если BB = 1b, это указывает на то, что шина занята. Устанавливается, как только шина переходит в активное состояние (одновременное появление низкого уровня сигнала на выводах SDA и SCL или хотя бы на одном из них) или в стартовое состояние. Сбрасывается при выключении интерфейса I2C, либо при обнаружении состояния останова.

Таблица А.105 – Регистр 1 управления

SMBCTRL1

ESFR (F0E0h / 70h) Сброс: 00h



Поле	Бит	Тип	Описание	
CLRST	7	Запись	Бит сброса флага прерывания INT. Запись «0» в бит CLR игнорируется. Запись «1» в бит CLR сбросит флаг INT в регистре SMBST. Чтение этого бита всегда возвращает «0»	
SMBARE	6	Запись Чтение	Бит управления реакцией на получение адреса отклика	
			0	Полученный адрес не проверяется на совпадение с адресом отклика
			1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом отклика (0001_100b)
			Бит очищается при выходе ведомого из режима IDLE	
GCMEN	5	Запись Чтение	Бит управления реакцией на получение адреса общего вызова	
			0	Полученный адрес не проверяется на совпадение с адресом общего вызова
			1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом общего вызова (0000_000b)
			Бит очищается при выходе ведомого из режима IDLE	
ACK	4	Запись Чтение	Бит квитирования приема. В режиме передатчика не используется. В режиме приемника (мастера/ведомого) содержит значение, которое передается в течение цикла отклика на запрос передатчика подтвердить прием. Передача нуля по окончании передачи байта (квитирование) означает, что данные успешно получены. Передача единицы (неквитирование) означает, что приемник не может продолжать работу по каким-либо причинам. Бит ACK очищается аппаратно по окончании цикла отклика	
INTEN	2	Запись Чтение	Бит разрешения прерывания	
			0	Запрещено
			1	Разрешено
STOP	1	Запись Чтение	Бит останова. В режиме мастера установка бита STOP генерирует состояние останова, которое завершает или прерывает текущую передачу. После прекращения передачи бит STOP очищается аппаратно	
START	0	Запись Чтение	Бит старта. Этот бит устанавливается, когда требуется сформировать стартовое состояние на шине. Бит START очищается аппаратно по окончании цикла стартового состояния, а также при обнаружении ошибки на шине (состояние с кодом 1Fh)	
–	3	–	Зарезервировано	

Таблица А.106 – Регистр 2 управления

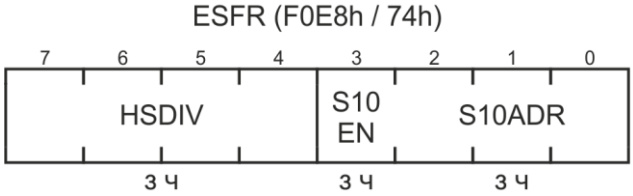
SMBCTRL2

ESFR (F0E4h / 72h) Сброс: 00h

Поле	Бит	Тип	Описание	
SCLFRQ	7–1	Запись Чтение	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме мастера. Длительности высокого(T_{SCLH}) и низкого(T_{SCLL}) уровней сигнала SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формуле $T_{SCLH} = T_{SCLL} = 2 \times SCLFRQ \times (1/fosc)$. (A.13) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{SCLH} + T_{SCLL})$. (A.14) В поле SCLFRQ можно записать любое значение в диапазоне от 04h до 7Fh. При попытке записи любого значения меньше «04h», оно будет записано со смещением 04h. Например, при записи числа 02h, к нему будет аппаратно добавлено смещение 04h и, в итоге, в поле SCLFRQ окажется значение «06h»	
ENABLE	0	Запись Чтение	Бит включения модуля I2C	
			0	Модуль выключен. Тактирование не осуществляется. Регистры SMBCTRL1, SMBST, SMBCST сброшены
			1	Модуль включен

Таблица А.107 – Регистр 3 управления

SMBCTRL3



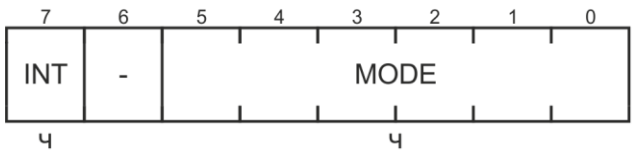
Поле	Бит	Тип	Описание
HSDIV	7–4	Запись Чтение	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме HS мастера. Длительности высокого(T_{HSCLH}) и низкого(T_{HSCLL}) уровней сигнала на выводе SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формулам $T_{HSCLH} = HSDIV \times (1/fosc)$, (A.15) $T_{HSCLL} = 2 \times HSDIV \times (1/fosc)$. (A.16) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{HSCLH} + T_{HSCLL})$. (A.17) В поле HSDIV можно записать любое значение в диапазоне от 2h до Fh. При попытке записи любого значения меньше «2h» в поле HSDIV, оно будет записано со смещением 2h. Например, при записи числа 1h к нему будет аппаратно добавлено смещение 2h и, в итоге, в поле SCLFRQ окажется значение «3h»
S10EN	3	Запись Чтение	Бит разрешения 10-битной адресации ведомого 0 – Запрещена 1 – Разрешена при условии, что установлен бит SAEN в регистре SMBADDR
S10ADR	2–0	Запись Чтение	Поле старших битов 10-битного адреса ведомого. Поле содержит старшие три разряда адреса ведомого при 10-битной адресации. Первый принятый байт адреса сравнивается со значением [11110b, S10ADR[2:1]], второй байт адреса – со значением [S10ADR[0], ADDR]*

* Скобки указывают на то, что заключенное в них число получается конкатенацией значений чисел, разделенных запятой; S10ADR[2:1] и S10ADR[0] – соответственно значения старших двух битов и младшего бита поля S10ADR; ADDR – значение поля регистра SMBADDR.

Таблица А.108 – Регистр состояния

SMBST

ESFR (F0DCh / 6Eh) Сброс: 00h

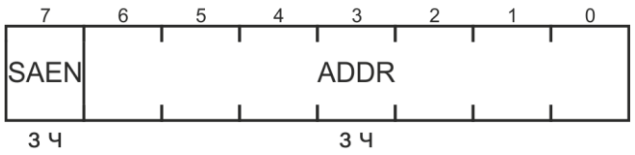


Поле	Бит	Тип	Описание
INT	7	Чтение	Флаг прерывания. Устанавливается после девятого такта сигнала SCL (когда SCL = 0) в любое запрограммированное время. Условия выставления флага INT: - во время приема/передачи как в режиме мастера, так и в режиме ведомого; - при совпадении адреса (адреса ведомого, адреса отклика или адреса общего вызова) содержимое регистра SMBSDA должно контролироваться программно для определения типа полученного адреса; - после успешного формирования стартового состояния или состояния повторного старта; - в случае неквитирования переданной информации; - при потере арбитража во время передачи последнего бита; - при обнаружении валидного состояния останова или состояния повторного старта; - при обнаружении ошибки на шине. Пока установлен флаг INT, на линии SCL удерживается низкий уровень сигнала. Флаг INT может быть сброшен установкой бита CLRST в регистре SMBCTRL1 или выключением модуля I2C (обнуление бита ENABLE в регистре SMBCTRL2). Условия выставления флага INT (не влияющие на уровень сигнала на линии SCL): - простой на линии SCL; - состояние останова в режиме ведомого (MODE = 1Ch); - потеря арбитража, вследствие чего ведомый переключился в безадресный режим (MODE = 03h или MODE = 23h); - неквитированная передача байта данных (MODE = 17h)
MODE	5–0	Чтение	Код состояния. Возникновение того или иного состояния в течение функционирования модуля I2C сопровождается записью соответствующего кода в поле MODE
–	6	–	Зарезервировано

Таблица А.109 – Регистр собственного адреса

SMBADDR

ESFR (F0E2h / 71h) Сброс: 00h

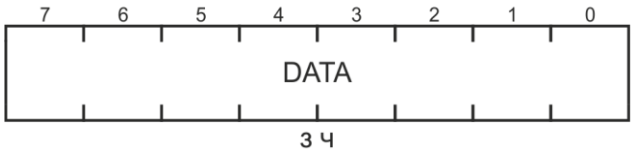


Поле	Бит	Тип	Описание	
SAEN	7	Запись Чтение	Бит разрешения распознавания адреса	
			0	Безадресный режим
			1	Включена функция распознавания принятого адреса
ADDR	6–0	Запись Чтение	Поле собственного 7-битного адреса. При работе в режиме ведомого первые 7 бит, принятые после стартового состояния, сравниваются со значением ADDR. Если обнаружено совпадение и установлен бит SAEN, ведомый переходит в режим приемника или передатчика (в зависимости от состояния бита направления R/W#)	

Таблица А.110 – Сдвиговый регистр данных

SMBSDA

ESFR (F0DAh / 6Dh) Сброс: XXh

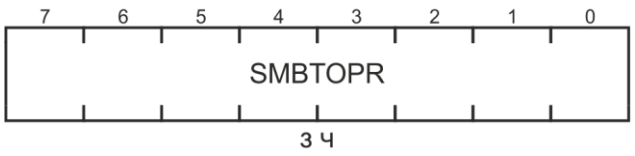


Поле	Бит	Тип	Описание
DATA	7–0	Запись Чтение	Поле данных

Таблица А.111 – Регистр загрузки предделителя

SMBTOPR

ESFR (F0E6h / 73h) Сброс: 00h



Поле	Бит	Тип	Описание
SMBTOPR	7–0	Запись Чтение	Поле значения перезагрузки предделителя

A.13 Регистры контроллера интерфейса CAN

Контроллер CAN оперирует тремя группами регистров: регистры контроллера, см. таблицу А.112, регистры узлов с символом «x» в названии, см. таблицы А.121 – А.131 и регистры объектов сообщений с символом «n» в названии, таблицы А.134 – А.144.

Таблица А.112 – Регистры контроллера CAN и его узлов CAN0 и CAN1

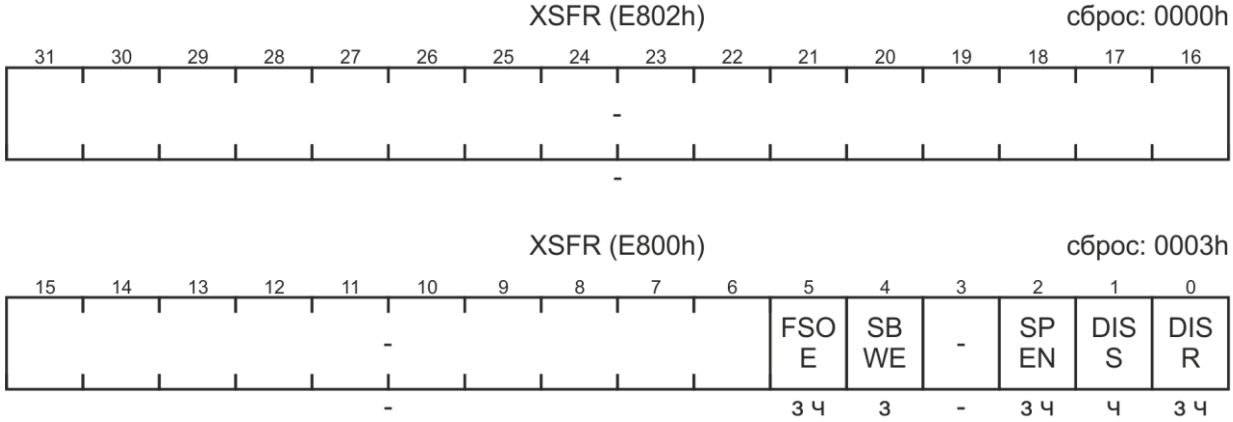
Мнемоника	Адреса слов регистра – старшего/младшего области XSFR	Состояние регистра после сброса
Регистры узла CAN1		
NFCR1	EB1Ah / EB18h	0000 0000h
NECNT1	EB16h / EB14h	0060 0000h
NBTR1	EB12h / EB10h	0000 0000h
NPCR1	EB0Eh / EB0Ch	0000 0000h
NIPR1	EB0Ah / EB08h	0000 0000h
NSR1	EB06h / EB04h	0000 0000h
NCR1	EB02h / EB00h	0000 0001h
Регистры узла CAN0		
NFCR0	EA1Ah / EA18h	0000 0000h
NECNT0	EA16h / EA14h	0060 0000h
NBTR0	EA12h / EA10h	0000 0000h
NPCR0	EA0Eh / EA0Ch	0000 0000h
NIPR0	EA0Ah / EA08h	0000 0000h
NSR0	EA06h / EA04h	0000 0000h
NCR0	EA02h / EA00h	0000 0001h
Регистры контроллера		
MITR	E9CEh / E9CCh	0000 0000h
MCR	E9CAh / E9C8h	0000 0000h
PANCTR	E9C6h / E9C4h	0000 0301h
MSIMASK	E9C2h / E9C0h	0000 0000h
MSID3	E98Eh / E98Ch	0000 0020h
MSID2	E98Ah / E988h	0000 0020h
MSID1	E986h / E984h	0000 0020h
MSID0	E982h / E980h	0000 0020h
MSPND3	E94Eh / E94Ch	0000 0000h
MSPND2	E94Ah / E948h	0000 0000h
MSPND1	E946h / E944h	0000 0000h
MSPND0	E942h / E940h	0000 0000h
LIST7	E91Eh / E91Ch	0100 0000h
LIST6	E91Ah / E918h	0100 0000h
LIST5	E916h / E914h	0100 0000h
LIST4	E912h / E910h	0100 0000h
LIST3	E90Eh / E90Ch	0100 0000h
LIST2	E90Ah / E908h	0100 0000h
LIST1	E906h / E904h	0100 0000h
LIST0	E902h / E900h	007F 7F00h
FDR	E80Eh / E80Ch	0000 0000h
ID	E80Ah / E808h	002B C051h
CLC	E802h / E800h	0000 0003h

В таблице А.112 приведен список регистров, которые контролируют работу всего контроллера CAN, а также список регистров управления узлами CAN0 и CAN1 с указанием их мнемонических наименований, адресов и состояний после сброса микроконтроллера.

В таблицах А.113 – А.144 приведено подробное описание регистров. Все регистры 32-разрядные. Биты 31–16 составляют старшее слово регистра, биты 15–0 составляют младшее слово.

Таблица А.113 – Регистр управления частотой

CLC

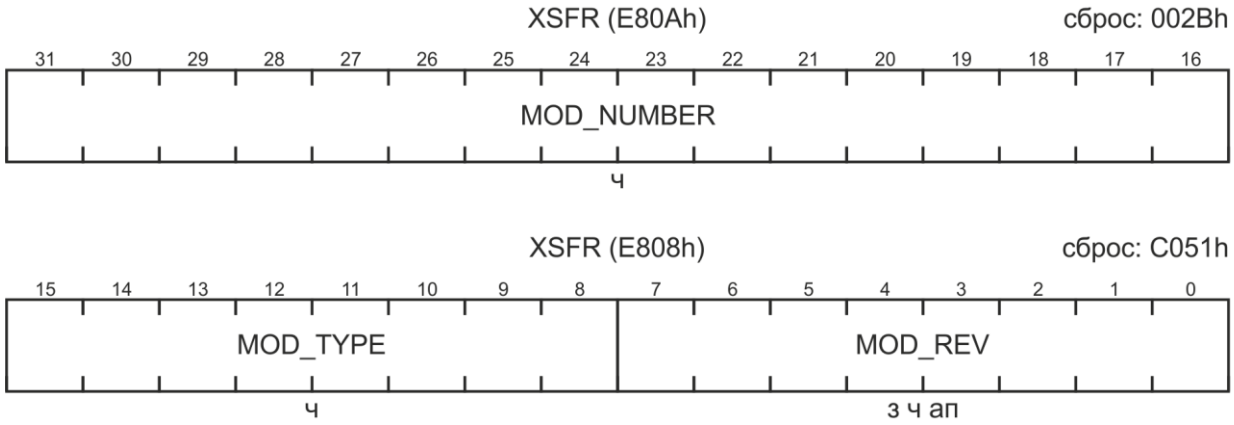


Поле	Бит	Описание
FSOE*	5	Бит активации быстрого выключения контроллера CAN для системы отладки
		0 Нет действий
		1 Запись единицы запускает механизм быстрого выключения
SBWE*	4	Бит разрешения записи в биты FSOE и SPEN для системы отладки
		0 Запись запрещена
		1 Запись разрешена
		Бит доступен только для записи. При чтении возвращается ноль
SPEN*	2	Бит активации режима приостановки работы (Suspend) для системы отладки
		0 Нет действий
		1 Запись единицы включает режим Suspend (если установлен бит SBWE)
DISS	1	Бит состояния контроллера CAN
		0 Включен
		1 Выключен
DISR	0	Бит выключения контроллера CAN
		0 Нет действий
		1 Запись единицы запускает механизм выключения
–	31–16, 15–6, 3	Зарезервировано

Примечание – Биты, отмеченные «*», доступны только в отладочном режиме работы контроллера CAN.

Таблица А.114 – Регистр идентификации

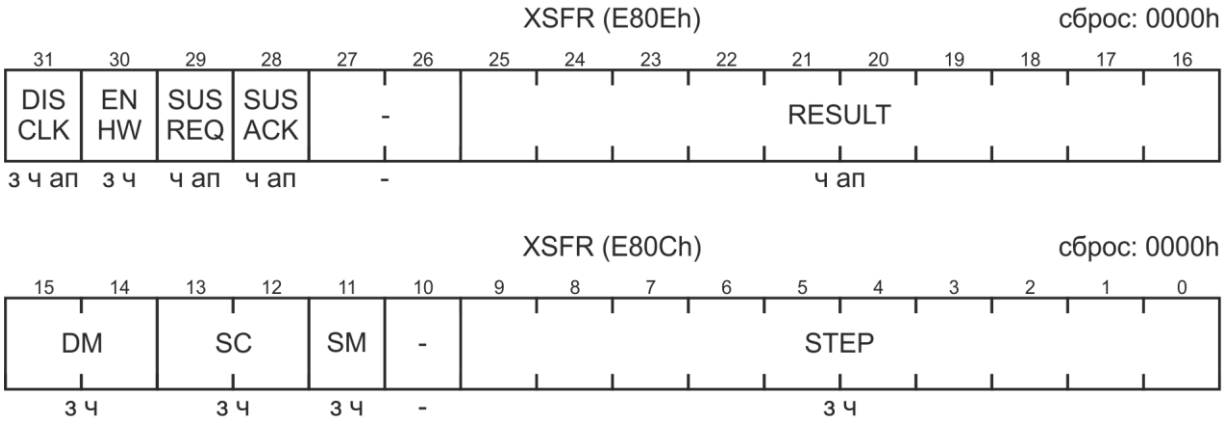
ID



Поле	Бит	Описание
MOD_NUMBER	31–16	Идентификационный номер контроллера CAN
MOD_TYPE	15–8	Разрядность контроллера CAN
MOD_REV	7–0	Число модификаций контроллера CAN

Таблица А.115 – Регистр делителя

FDR



Поле	Бит	Описание	
DISCLK	31	Бит запрета внутреннего тактирования	
		0	Генерирование сигнала Fcan разрешено
		1	Генерирование сигнала Fcan запрещено
ENHW	30	Бит контроля синхронизации. Это бит аппаратно удерживается в сброшенном состоянии и не может быть установлен	
SUSREQ*	29	Бит активации режима приостановки работы (Suspend)	
		0	Нет действий
		1	Установка бита запускает механизм приостановки. Контроллер CAN перейдет в режим Suspend только тогда, когда будут установлены биты SUSREQ и SUSACK
SUSACK*	28	Индикатор режима Suspend	
		0	Контроллер CAN функционирует в нормальном режиме
		1	Контроллер CAN в режиме приостановки
RESULT	25–16	Счетчик делителя частоты	
DM	15–14	Поле задания режима делителя частоты	
		00	Счетчик выключен. Синхросигнал Fout не генерируется. Сигнал сброса внешнего делителя в состоянии логической единицы
		01	Нормальный режим работы
		10	Режим дробного деления
		11	Счетчик выключен. Синхросигнал Fout не генерируется
SC*	13–12	Поле задания конфигурации делителя частоты в режиме Suspend (см. таблицу А.116)	
SM*	11	Бит выбора режима приостановки работы	
		0	Промежуточный режим Suspend
		1	Основной режим Suspend
STEP	9–0	Шаг делителя. Поле хранит значение, которое загружается в RESULT при переполнении счетчика делителя	
–	27, 26, 10	Зарезервировано	

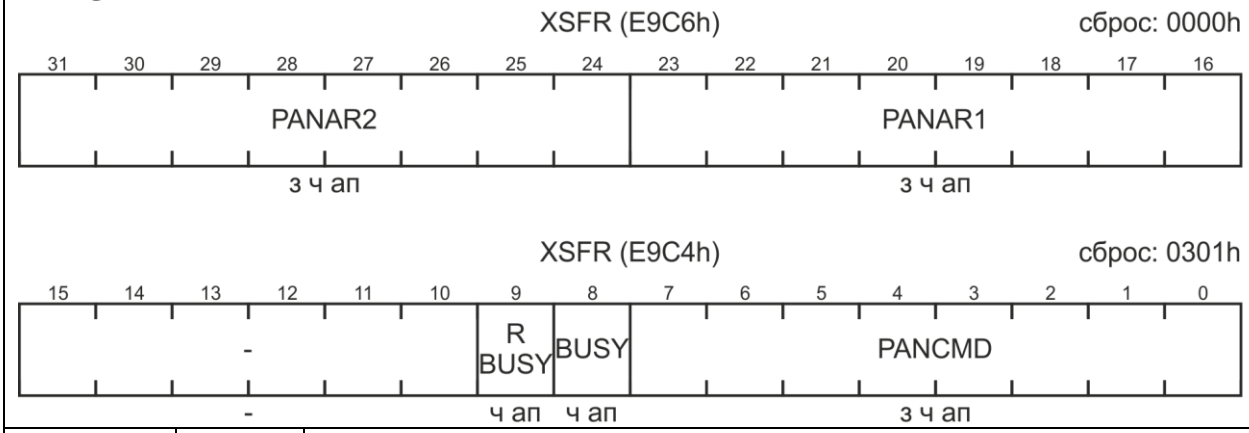
Примечание – Биты, отмеченные «*», доступны только в отладочном режиме работы контроллера CAN.

Таблица А.116 – Варианты конфигурации делителя частоты

Режим	Поле SC	Поле DM	Состояние сигнала сброса внешнего делителя	Поле RESULT	Формирование сигнала Fout	Состояние/ режим делителя
Нормальный	–	00	1	Не меняется	Нет	Выключен
		01	0	При активации режима загружается значением 3FFh. Далее периодически загружается значением из STEP	Да	Нормальный
		10				Дробный
		11		Не меняется	Нет	Выключен
Suspend	00	00	1	Не меняется	Нет	Выключен
		01	0	При активации режима загружается значением 3FFh. Далее периодически загружается значением из STEP	Да	Нормальный
		10				Дробный
		11		Не меняется	Нет	Выключен
	01	00	1	Не меняется	Нет	Выключен
		01	0	Загружается значением 3FFh		Остановлен
		10				
		11		Не меняется		Выключен
	10	00	1	Загружается значением 3FFh		Выключен
		01	0			Остановлен
		10				
		11				Выключен
	11	–	1			Выключен

Таблица А.117 – Регистр панели команд

PANCTR



Поле	Бит	Описание
1	2	3
PANAR2	31–24	Панель аргумента 2 (см. таблицу А.118)

Окончание таблицы А.117

1	2	3	
PANAR1	23–16	Панель аргумента 1 (см. таблицу А.118)	
RBUSY	9	Флаг занятости панелей аргументов	
		0	Нет действий
		1	Выполняется команда списка, результат выполнения которой будет записан в PANAR1 и PANAR2
BUSY	8	Флаг занятости панелей аргументов	
		0	Панели готовы для записи
		1	Панели заняты – ожидают записи по окончании выполнения команды
PANCMD	7–0	Поле команды (см. таблицу А.118). После выполнения команды в это поле записывается 00h	
–	15–10	Зарезервировано	

Таблица А.118 – Коды команд работы со списками

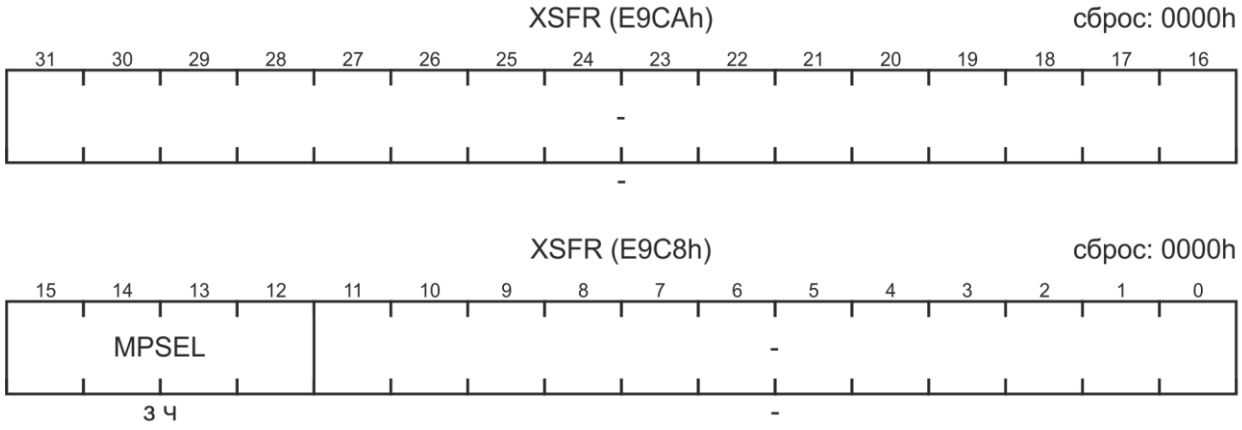
Код команды PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
1	2	3	4
00h	–	–	Нет операции. Никаких действий не выполняется
01h	Результат: бит 7 – ошибка, бит 6 – не определен	–	Инициализация списков. Запуск инициализации для очистки битовых полей CTRL и LIST всех объектов сообщений. Регистры LIST0 – LIST8 устанавливаются в свои значения после сброса. Это приводит к переносу всех объектов сообщений в список №0 (список нераспределенных объектов сообщений). Инициализация списков требует, чтобы биты INIT и CCE регистра NCR были установлены для обоих узлов. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – Инициализация завершена успешно; - 1 – Инициализация не завершена, поскольку не все биты INIT и CCE были установлены. Команда инициализации списков автоматически запускается при каждом сбросе контроллера CAN, за исключением случая, когда все регистры объектов сообщений уже сброшены
02h	Аргумент: номер списка	Аргумент: номер объекта сообщения	Статическое занесение объекта сообщения в список. Объект сообщения переносится из текущего списка в список, указанный полем PANAR2 и добавляется в его конец. Эта команда также используется для дераспределения объекта сообщения, т.е. переноса его в список № 0 (если PANAR2 равно 00h)

Окончание таблицы А.118

1	2	3	4
03h	Аргумент: номер списка Результат: бит 7 – ошибка, бит 6– не определен	Результат: номер объекта сообщения	Динамическое занесение объекта сообщения в список. Первый объект сообщения списка №0 переносится в список, указанный полем PANAR2, и добавляется в его конец. Номер объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – Операция выполнена; - 1 – Операция не выполнена – список №0 пуст
04h	Аргумент: номер объекта сообщения	Аргумент: текущий номер объекта сообщения	Перемещение по списку вверх. Перенос объекта сообщения с номером PANAR1 на одну позицию выше, чем расположен объект сообщения с номером PANAR2
05h	Аргумент: номер объекта сообщения Результат: бит 7 – ошибка, бит 6 – не определен	Результат: номер добавлен-ного объекта сообщения	Динамическая вставка в список. Первый объект сообщения списка №0 вставляется на одну позицию выше, чем расположен объект сообщения с номером PANAR2. Номер добавленного объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – Операция выполнена; - 1 – Операция не выполнена – список №0 пуст
06h	Аргумент: номер объекта сообщения	Аргумент: текущий номер объекта сообщения	Перемещение по списку вниз. Перенос объекта сообщения с номером PANAR1 на одну позицию ниже, чем расположен объект сообщения с номером PANAR2
07h	Аргумент: номер объекта сообщения Результат: бит 7– ошибка, бит 6 – не определен	Результат: номер добавлен-ного объекта сообщения	Динамическая вставка в список. Первый объект сообщения списка №0 вставляется на одну позицию ниже, чем расположен объект сообщения с номером PANAR2. Номер добавленного объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – Операция выполнена; - 1 – Операция не выполнена – список №0 пуст
08h – FFh	–	–	Зарезервировано

Таблица А.119 – Регистр управления

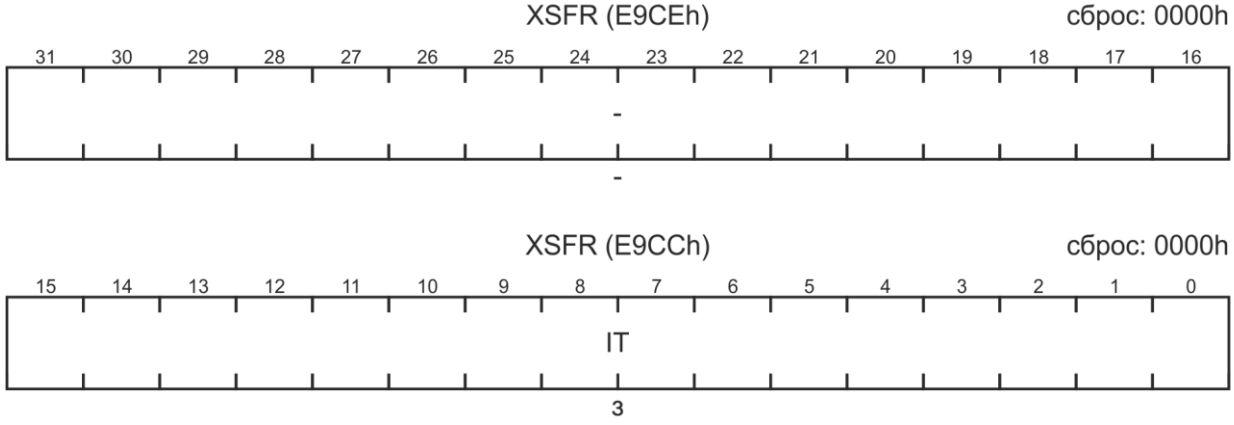
MCR



Поле	Бит	Описание
MPSEL	15–12	Поле задания позиции ждущего бита сообщения после приема/ передачи сообщения
–	31-16, 11–0	Зарезервировано

Таблица А.120 – Регистр прерываний

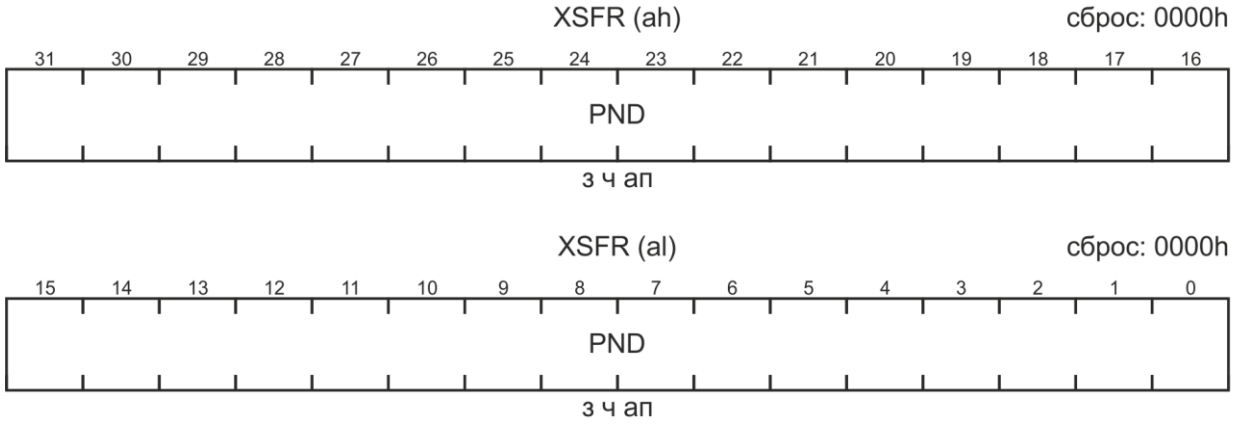
MITR



Поле	Бит	Описание
IT	15–0	Поле генератора прерываний. Каждый бит поля связан с одной из линий прерываний. Номера битов от 0 до 15 соответствуют номерам линий прерываний. Для того, чтобы сгенерировать одно или несколько прерываний, следует установить соответствующие биты. Установленные биты сбрасываются аппаратно
–	31–16	Зарезервировано

Таблица А.121 – Регистр ждущих прерываний

MSPNDx

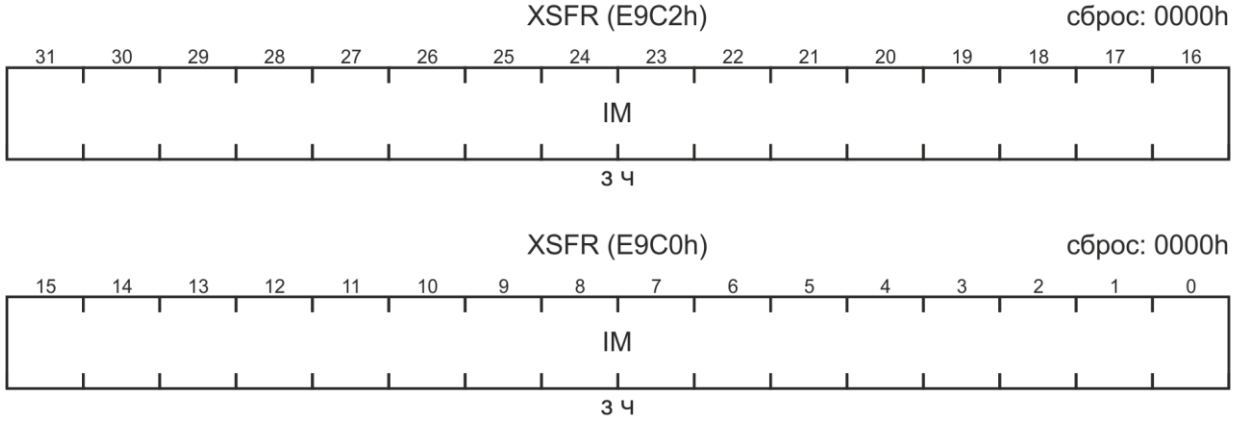


x	0	1	2	3
Регистр	MSPND0	MSPND1	MSPND2	MSPND3
ah	E942h	E946h	E94Ah	E94Eh
al	E940h	E944h	E948h	E94Ch

Поле	Бит	Описание
PND	31–0	Поле ждущих битов сообщений. Каждому объекту сообщений выделяется два бита. Биты устанавливаются только аппаратно. Установленные биты сбрасываются аппаратно по окончании обслуживания запроса прерывания или могут быть сброшены в любой момент программно

Таблица А.122 – Регистр маски индекса сообщения

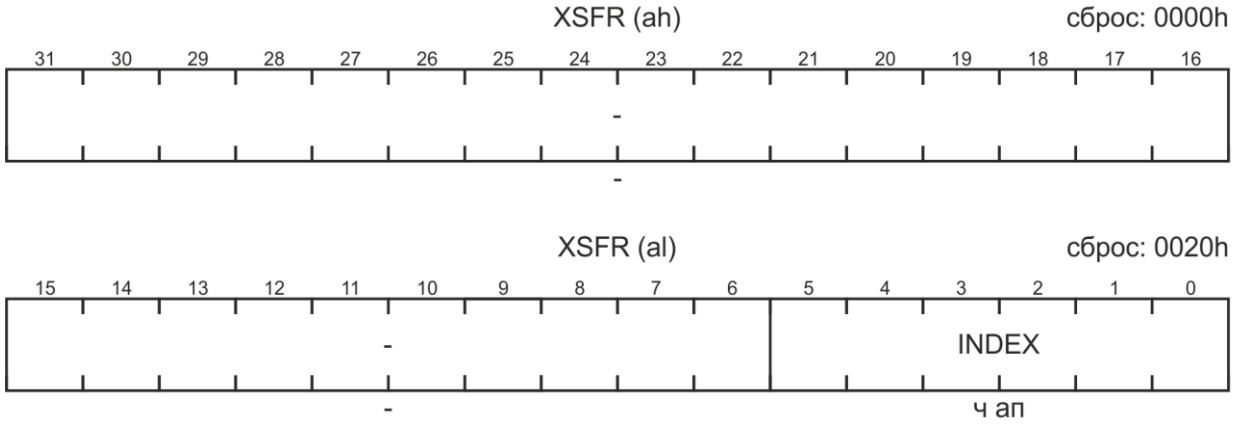
MSIMASK



Поле	Бит	Описание
IM	31–0	Маска для ждущих битов сообщений. Учитывается состояние только тех бит регистра MSPND, для которых в поле IM установлены соответствующие биты
–	31–6	Зарезервировано

Таблица А.123 – Регистр индекса сообщения

MSIDx

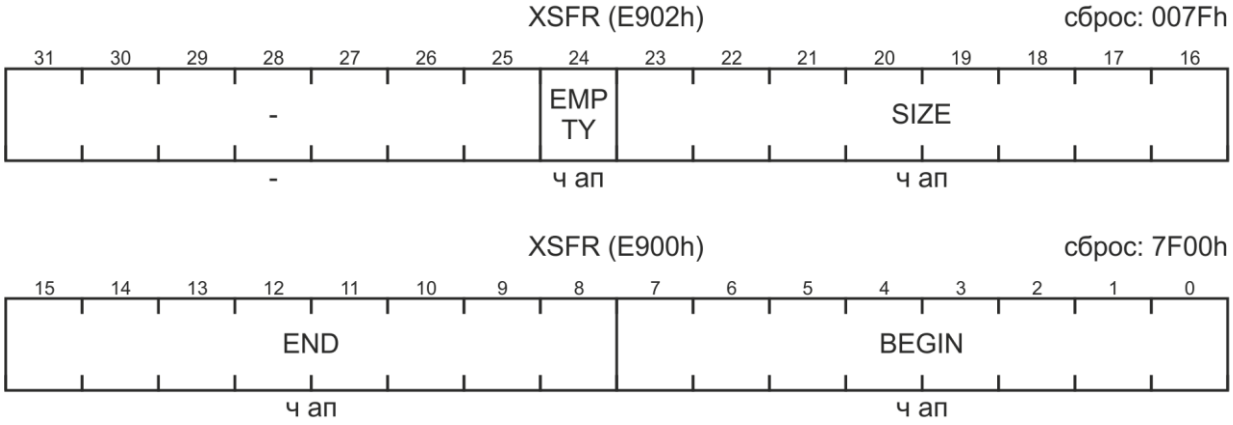


x	0	1	2	3
Регистр	MSID0	MSID1	MSID2	MSID3
ah al	E982h E980h	E986h E984h	E98Ah E988h	E98Eh E98Ch

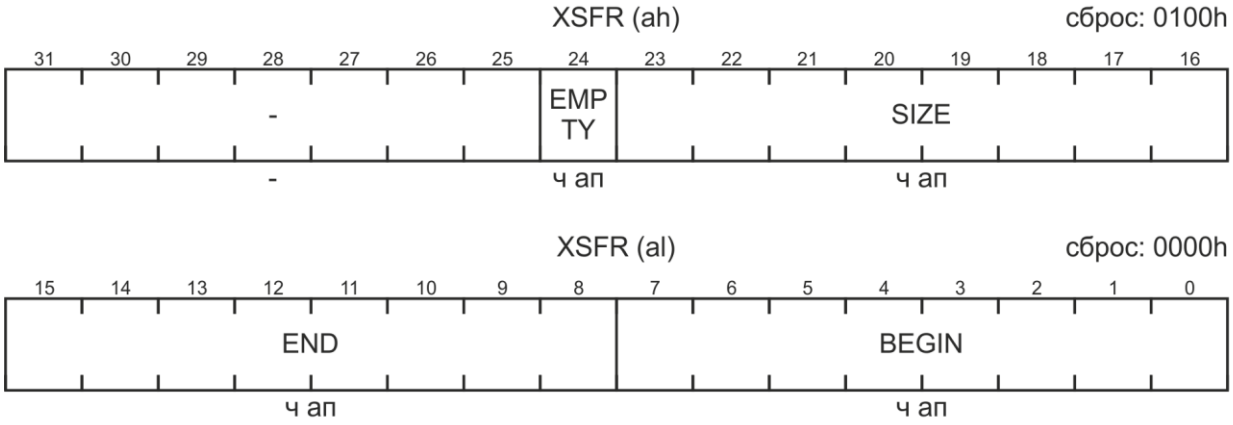
Поле	Бит	Описание
INDEX	5–0	Поле номера ждущего бита. Если в регистре MSPND есть установленные биты, которые не маскируются соответствующими битами регистра MSIMASK, то поле INDEX будет указывать на самый старший из них. Если в регистре MSPND нет установленных битов или они замаскированы, то в поле INDEX будет находиться значение 20h, указывающее на бит 31 регистра MSPND
–	31–6	Зарезервировано

Таблица А.124 – Регистры списков

LIST0 – регистр списка №0 (список нераспределенных объектов сообщений)



LISTx – регистр свободного списка



x	1	2	3	4	5	6	7
Регистр	LIST1	LIST2	LIST3	LIST4	LIST5	LIST6	LIST7
ah al	E906h E904h	E90Ah E908h	E90Eh E90Ch	E912h E910h	E916h E914h	E91Ah E918h	E91Eh E91Ch

Поле	Бит	Описание	
EMPTY	24	Индикатор пустого списка	
		0	В списке есть как минимум один элемент
		1	Список пуст
SYZE	23–16	Размер списка. Количество элементов (объектов сообщений) в списке. Значение поля SYZE всегда на единицу меньше числа элементов. Если список пуст, SYZE = 00h	
END	15–8	Номер объекта сообщения, находящегося последним в списке. Поле может принимать значения от 00h до 3Fh, согласно количеству объектов сообщений (64)	
BEGIN	7–0	Номер объекта сообщения, находящегося первым в списке. Поле может принимать значения от 00h до 3Fh, согласно количеству объектов сообщений	
–	31–25	Зарезервировано	

Таблица А.125 – Регистр управления узла

NCRx

XSFR (ah) сброс: 0000h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
-															

XSFR (al) сброс: 0001h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-							SUS EN	CAL M	CCE	-	CAN DIS	ALI E	LEC IE	TRIE	INIT
-							з ч	з ч	з ч	-	з ч	з ч	з ч	з ч	з ч ап

x	0	1
Регистр	NCR0	NCR1
ah	EA02h	EB02h
al	EA00h	EB00h

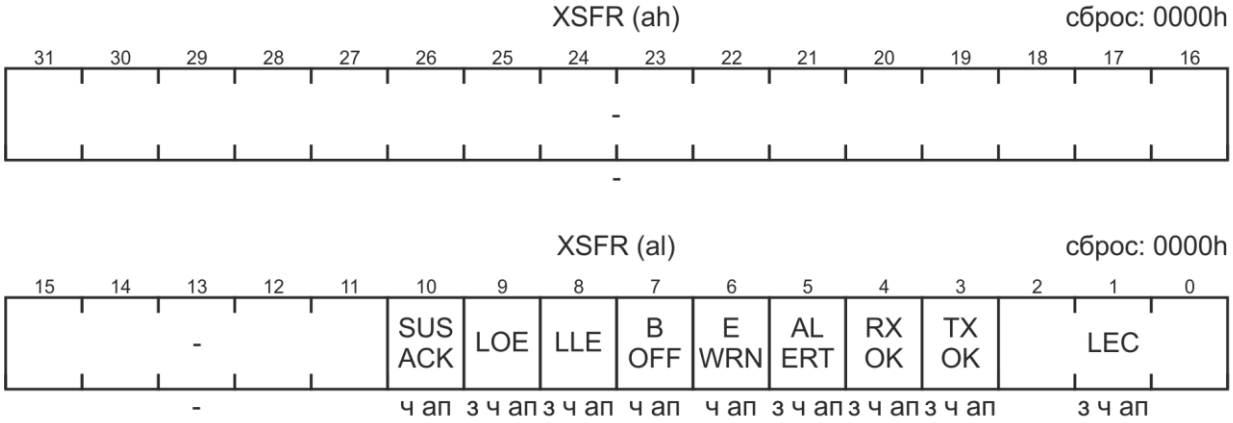
Поле	Бит	Описание	
1	2	3	
SUSEN	8	Бит разрешения режима приостановки работы узла	
		0	Отладочная система не может перевести узел в режим Suspend
		1	Отладочная система может перевести узел в режим Suspend. При активации режима узел перейдет в состояние «простоя» или «отключен от шины», после чего аппаратно установится бит INIT и будет находиться в этом состоянии до выхода из режима
		Бит сбрасывается при сбросе системы отладки	
CALM	7	Бит включения режима анализа узла	
		0	Режим выключен
		1	Установка бита включает режим анализа узла. В этом режиме сообщения могут только приниматься, бит подтверждения не посылается после успешного приема сообщения, флаг активной ошибки посылается рецессивным вместо доминантного. На линии отправки сообщений поддерживается высокий уровень сигнала
		Бит может быть установлен только, если установлен бит INIT	
CCE	6	Бит разрешения изменения конфигурации узла. Управляет доступом к регистрам NBTR, NPCR и NECNT	
		0	Только чтение
		1	Полный доступ
CANDIS	4	Бит выключения узла	
		0	Сброс бита включает узел
		1	Установка бита выключает узел. Сначала узел переходит в состояние «простоя» или «отключен от шины», далее аппаратно устанавливается бит INIT и, если разрешено, генерируется прерывание ALERT

Окончание таблицы А.125

1	2	3	
ALIE	3	Бит разрешения прерывания ALERT от узла	
		0	Запрещено
		1	Разрешено
LECIE	2	Бит разрешения прерывания от узла при обнаружении кода последней ошибки	
		0	Запрещено
		1	Разрешено
TRIE	1	Бит разрешения прерывания от узла по окончании передачи/приема	
		0	Запрещено
		1	Разрешено
INIT	0	Инициализация узла	
		0	Сброс бита разрешает участие узла в трафике CAN шины. Узел ожидает последовательность из 11 рецессивных бит на шине и включается в трафик. Если на момент сброса бита INIT узел находился в состоянии «отключен от шины», начинается процесс выхода из этого состояния в следующем порядке: получение 128 последовательностей бит (каждая из 11 рецессивных бит), выход из состояния «отключен от шины», включение в трафик
		1	Установка бита INIT прекращает участие узла в трафике. Все текущие передачи останавливаются, линии передач переходят в рецессивное состояние. Если на момент установки бита INIT узел находился в состоянии «отключен от шины», процесс выхода из этого состояния продолжается до его завершения. Далее узел остается неактивным до тех пор, пока установлен бит INIT
–	31–9, 5	Зарезервировано	

Таблица А.126 – Регистр состояния узла

NSRx



x	0	1
Регистр	NSR0	NSR1
ah al	EA06h EA04h	EB06h EB04h

Поле	Бит	Описание	
1	2	3	
SUSACK	10	Индикатор перехода узла в режим приостановки	
		0	Узел не переключился в режим Suspend
		1	Узел находится в режиме Suspend
LOE	9	Флаг ошибки номера списка	
		0	Ошибок не обнаружено
		1	Обнаружена ошибка при фильтрации принимаемого сообщения. В регистре MOSTAT объекта сообщения обнаружен неверный номер списка
		Бит должен сбрасываться программно записью нуля	
LLE	8	Флаг ошибки списка	
		0	Ошибок не обнаружено
		1	Обнаружена ошибка при фильтрации принимаемого сообщения. Количество элементов списка, принадлежащего узлу, отличается от указанного в поле SYZE соответствующего регистра списка
		Бит должен сбрасываться программно записью нуля	
BOFF	7	Флаг состояния «отключен от шины»	
		0	Узел не находится в состоянии «отключен от шины»
		1	Узел находится в состоянии «отключен от шины»
EWRN	6	Флаг критического количества ошибок	
		0	Лимит ошибок еще не достигнул
		1	По крайней мере, один из счетчиков ошибок (REC, TEC) достиг лимита ошибок, заданного полем EWRNLVL регистра NECNT узла

Продолжение таблицы А.126

1	2	3	
ALERT	5	Флаг предупреждения ALERT	
		0	Нет событий
		1	Произошло одно или несколько не взаимоисключающих событий: - модификация бита BOFF; - модификация/установка бита LOE; - установка бита LLE; - аппаратная установка бита INIT
		Бит должен сбрасываться программно записью нуля	
RXOK	4	Флаг успешного приема сообщения	
		0	Полученных сообщений нет
		1	Сообщение получено
		Бит должен сбрасываться программно записью нуля	
TXOK	3	Флаг успешной передачи сообщения	
		0	Переданных сообщений нет
		1	Сообщение передано без ошибок с получением подтверждения
		Бит должен сбрасываться программно записью нуля	
LEC	2–0	Код последней ошибки. Поле хранит код последней из обнаруженных ошибок работы узла	
		000	Ошибок нет
		001	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	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	Ошибка подтверждения (ACKNOWLEGMENT ERROR). Обнаруживается передатчиком всякий раз, когда он не обнаруживает доминантный бит АСК в «области подтверждения»
		100	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

Окончание таблицы А.126

1	2	3	
LEC	2–0	101	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 для отслеживания длительного простоя шины
		110	Ошибка циклического избыточного кода (CRC ERROR). Передатчик по установленному алгоритму вычисляет значение контрольной суммы (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. В случае не совпадения фиксируется ошибка
		111	Код разрешения аппаратной записи в поле LEC
		После аппаратной записи в поле LEC значения кода, отличного от 111b, поле становится закрытым для записи и далее центральный процессор не может изменить его состояние до тех пор, пока в это поле не будет программно записано значение 111b	
–	31–11	Зарезервировано	

Таблица А.127 – Регистр указателя прерываний узла

NIPRx

XSFR (ah) сброс: 0000h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

XSFR (al) сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFCINP				TRINP				LECINP				ALINP			
3 ч				3 ч				3 ч				3 ч			

x	0	1
Регистр	NIPR0	NIPR1
ah	EA0Ah	EB0Ah
al	EA08h	EB08h

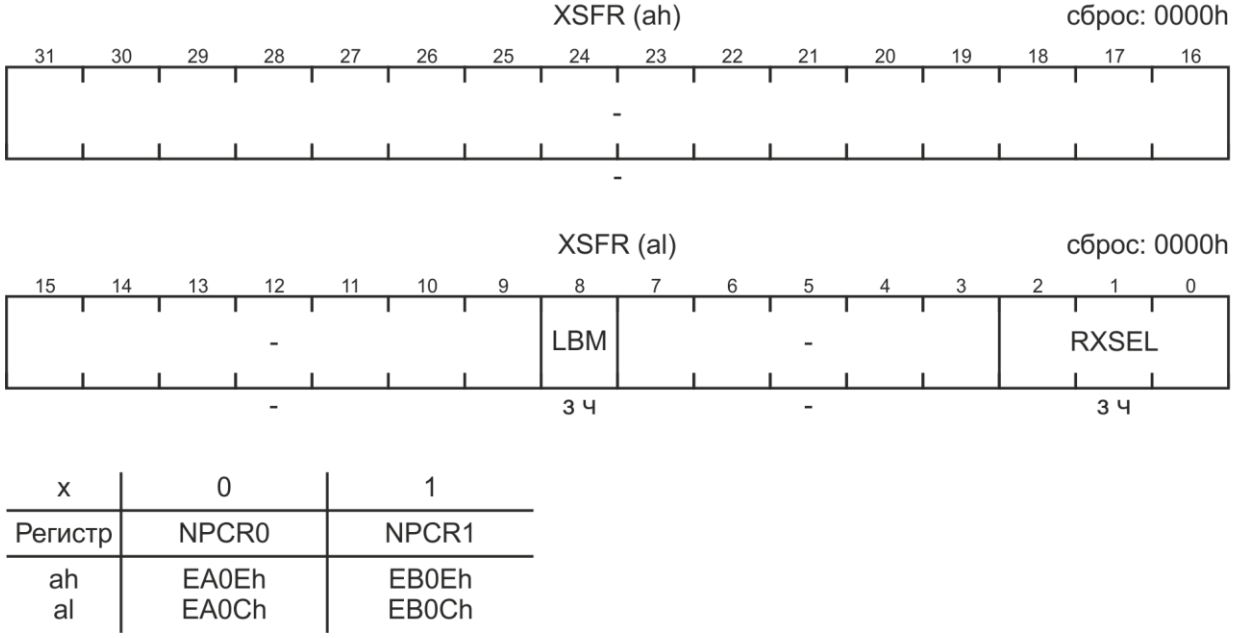
Поле	Бит	Описание
1	2	3
CFCINP	15–12	Указатель линии прерывания для прерывания при переполнении счетчика фреймов узла

Окончание таблицы А.127

1	2	3
TRINP	11–8	Указатель линии прерывания для прерывания по окончании передачи/приема сообщения
LECINP	7–4	Указатель линии прерывания для прерывания при записи кода последней ошибки
ALINP	3–0	Указатель линии прерывания для прерывания ALERT
–	31–16	Зарезервировано
Примечание – Каждый из указателей позволяет задать номер одной из 16 линий прерываний для каждого из четырех источников. Значение 00h соответствует нулевой линии прерываний, значение 01h – первой и так далее до значения FFh, которое соответствует линии 15 прерываний.		

Таблица А.128 – Регистр управления портом узла

NPCRx



x	0	1
Регистр	NPCR0	NPCR1
ah al	EA0Eh EA0Ch	EB0Eh EB0Ch

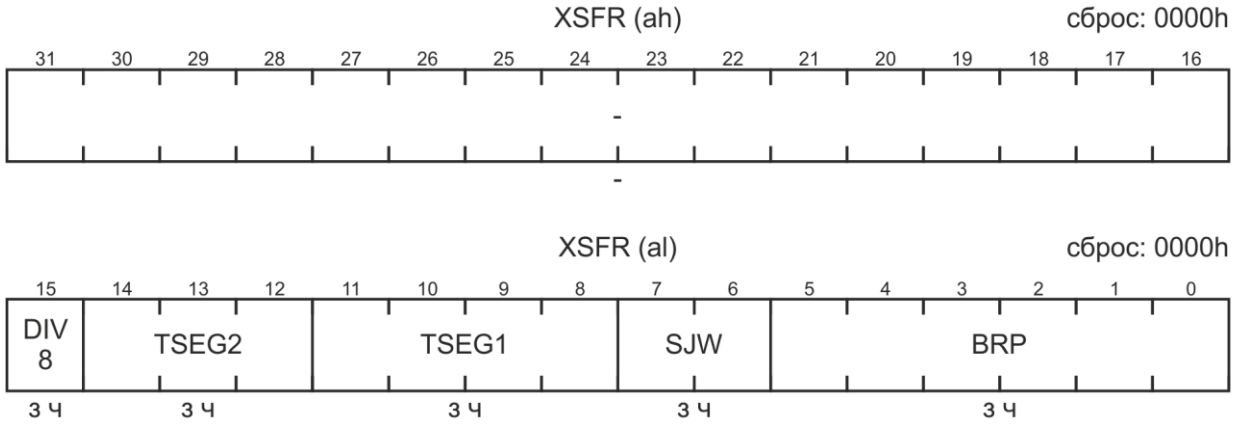
Поле	Бит	Описание	
1	2	3	
LBM	8	Бит включения режима обратной петли (Loop-Back)	
		0	Режим выключен
		1	Включен режим обратной петли. В этом режиме узел подсоединяется к внутренней виртуальной CAN шине. Если для обоих узлов включен режим обратной петли, то они объединяются виртуальной CAN шиной и могут взаимодействовать друг с другом. При этом на внешних выводах узлов, соединенных с внешней физической CAN шиной, поддерживается рецессивный уровень сигнала, т.е. узлы неактивны

Окончание таблицы А.128

1	2	3	
RXSEL	2–0	Поле выбора вывода микроконтроллера для приема сообщений	
		000	P4.5
		001	P4.7
		010	P7.6
		011	P9.2
		100	P4.4
		101	P7.4
		110	P9.0
		111	Зарезервировано
–	31–9, 7–3	Зарезервировано	

Таблица А.129 – Регистр синхронизации битов

NBTRx



x	0	1
Регистр	NBTR0	NBTR1
ah	EA12h	EB12h
al	EA10h	EB10h

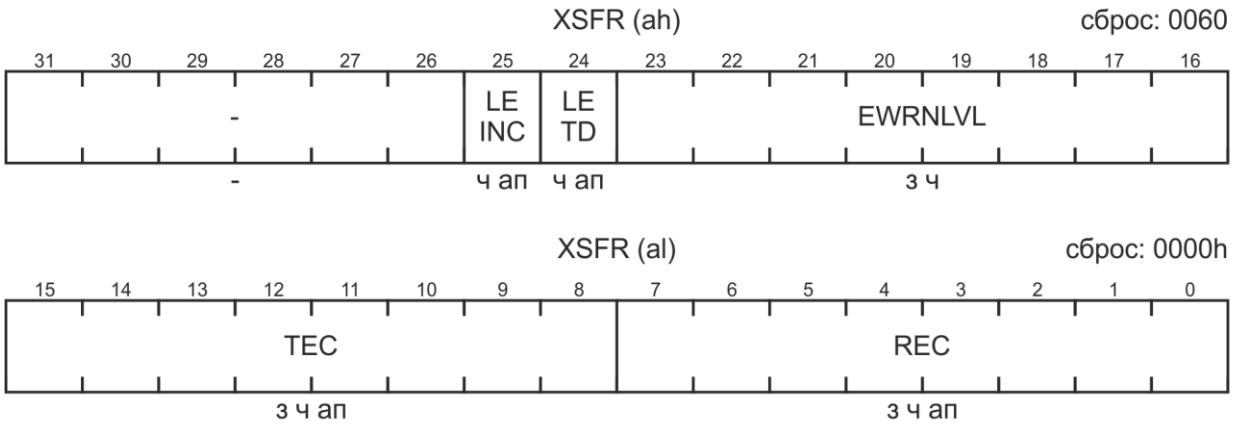
Поле	Бит	Описание	
1	2	3	
DIV8	15	Делитель частоты на восемь	
		0	Длительность кванта времени (BRP + 1), тактов частоты
		1	Длительность кванта времени 8 × (BRP + 1), тактов частоты
TSEG2	14–12	Параметр 2. Временной промежуток от точки выборки до точки передачи, определяемый пользователем. Длительность сегмента равна tq × (TSEG2 + 1) и может быть уменьшена за счет ресинхронизации. Допустимые значения для TSEG1: от 01h до 07h	
TSEG1	11–8	Параметр 1. Временной промежуток от сегмента синхронизации до точки выборки, определяемый пользователем и включающий в себя сегмент распространения. Длительность равна tq × (TSEG1 + 1) и может быть увеличена за счет ресинхронизации. Допустимые значения для TSEG1: от 02h до 0Fh	

Окончание таблицы А.129

1	2	3
SJW	7–6	Ширина перехода ресинхронизации. Длительность равна tq × (SJW + 1)
BRP	5–0	Предделитель скорости передачи. Если DIV8 = 0b, тогда длительность одного кванта времени равна (BRP + 1) тактам частоты. Если DIV8 = 1b, тогда длительность одного кванта времени равна 8×(BRP + 1) тактам частоты
–	31–16	Зарезервировано

Таблица А.130 – Регистр счетчика ошибок узла

NECNTx

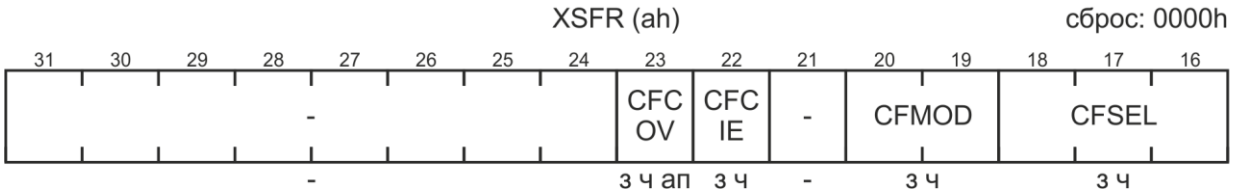


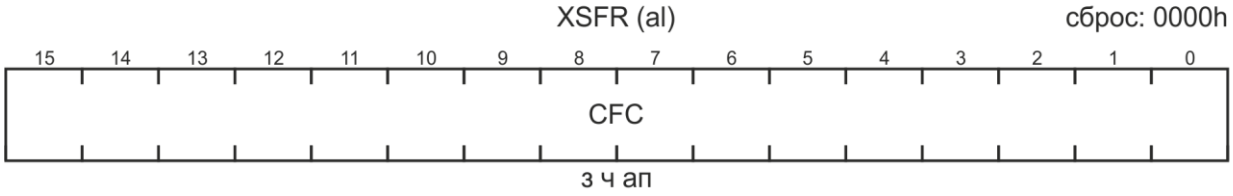
x	0	1
Регистр	NECNT0	NECNT1
ah	EA16h	EB16h
al	EA14h	EB14h

Поле	Бит	Описание	
LEINC	25	Индикатор инкрементирования при последней ошибке	
		0	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на единицу
		1	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на восемь
LETD	24	Флаг последней ошибки передачи	
		0	При приеме сообщения обнаружена ошибка, и произошло инкрементирование поля REC
		1	При передаче сообщения обнаружена ошибка, и произошло инкрементирование поля TEC
EWRNLVL	23–16	Поле задания лимита ошибок, по достижении которого выставляется флаг EWRN в регистре NSR (по умолчанию, количество ошибок –96)	
TEC	15–8	Поле счетчика ошибок приема сообщений	
REC	7–0	Поле счетчика ошибок приема сообщений	
–	31–26	Зарезервировано	

Таблица А.131 – Регистр счетчика сообщений

NFCRx





x	0	1
Регистр	NFCR0	NFCR1
ah al	EA1Ah EA18h	EB1Ah EB18h

Поле	Бит	Описание	
CFCOV	23	Флаг переполнения счетчика сообщений	
		0	Счетчик не переполнен
		1	Счетчик переполнился. В режиме синхросчетчика этот флаг устанавливается при изменении поля CFC и, если установлен бит CFCIE, формируется прерывание
		Бит сбрасывается программно	
CFCIE	22	Бит разрешения прерывания от счетчика сообщений	
		0	Запрещено
		1	Разрешено
CFMOD	20–19	Поле задания режима работы счетчика сообщений	
		00	Счетчик сообщений. Инкрементируется после каждого успешного приема/передачи сообщения
		01	Счетчик битов
		10	Синхросчетчик
		11	Зарезервировано
CFSEL	18–16	Поле задания параметров выбранного режима счетчика сообщений (см. таблицу А.132)	
CFC	15–0	Счетчик сообщений	
		Режим	Содержимое
		Счетчик сообщений	Количество успешно принятых/отправленных сообщений
		Счетчик битов	Количество обнаруженных битов от начала передачи/приема сообщения
		Синхросчетчик	Количество тактов сигнала Fclc минус один
–	31–24, 21	Зарезервировано	

Таблица А.132 – Коды задания параметров режима счетчика сообщений

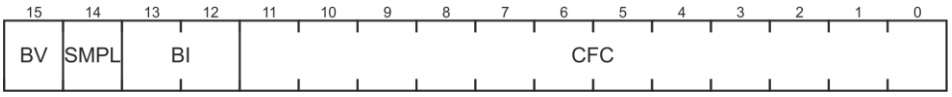
<table>
<tr><th colspan="3">Счетчик сообщений (CFMOD = 00b)</th></tr>
<tr><td>Код в поле CFSEL</td><td colspan="2">Действия</td></tr>
<tr><td>* * 1</td><td colspan="2">Счетчик инкрементируется каждый раз при получении сообщения не имеющего объекта сообщения</td></tr>
<tr><td>* 1 *</td><td colspan="2">Счетчик инкрементируется каждый раз при получении сообщения имеющего соответствующий объект сообщения</td></tr>
<tr><td>1 * *</td><td colspan="2">Счетчик инкрементируется каждый раз при успешной отправке сообщения</td></tr>
<tr><td colspan="3">Примечание – «*» указывает на то, что состояние этого бита поля CFSEL не важно для включения параметра режима. Все три параметра могут комбинироваться между собой (например, 110b или 101b)</td></tr>
<tr><th colspan="3">Счетчик битов (CFMOD = 01b)</th></tr>
<tr><td>000</td><td colspan="2">Счетчик инкрементируется каждый раз с началом очередного бита, начиная от начала передачи сообщения</td></tr>
<tr><td>001 – 111</td><td colspan="2">Счетчик выключен</td></tr>
<tr><th colspan="3">Синхросчетчик (CFMOD = 10b)</th></tr>
<tr><td>000</td><td colspan="2">В счетчик записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени между двумя последовательными ниспадающими фронтами сигнала на принимающем входе</td></tr>
<tr><td>001</td><td colspan="2">В счетчик записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени между двумя последовательными нарастающими фронтами сигнала на принимающем входе</td></tr>
<tr><td>010</td><td colspan="2">В счетчик записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени от обнаружения ниспадающего фронта сигнала до момента выборки, в результате которой бит определен как доминантный на принимающем входе</td></tr>
<tr><td>011</td><td colspan="2">В счетчик записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени от обнаружения нарастающего фронта сигнала до момента выборки, в результате которой бит определен как рецессивный на принимающем входе</td></tr>
<tr><td>100</td><td colspan="2">В счетчик записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени между обнаружением ниспадающего фронта сигнала – начала сегмента синхронизации – и точкой выборки на принимающем входе</td></tr>
<tr><td rowspan="5">101</td><td colspan="2">Поле CFC младшего слова счетчика условно разбивается на сектора:
 
</td></tr>
<tr><td>BV</td><td>Передаваемое значение бита</td></tr>
<tr><td>SMPL</td><td>Принятое значение бита</td></tr>
<tr><td>BI</td><td>Информация о шине (см. таблицу А.133)</td></tr>
<tr><td>CFC</td><td>Каждый раз в момент выборки в CFC записывается количество тактов сигнала Fclc, которые были подсчитаны в промежутке времени от начала передачи предыдущего бита до начала передачи текущего бита (для которого делается выборка) на принимающем входе</td></tr>
<tr><td>110, 111</td><td colspan="2">Зарезервировано</td></tr>
</table>

Таблица А.133 – Информация, записываемая в старшие разряды поля CFC при CFSEL = 101b

Код в секторе BI	Состояние шины CAN
00	Нет бита. Шина находится в состоянии простоя, осуществляет (де-) стаффинг бита или на шине находится один из сегментов: SOF, бит SRR, CRC, разделители, первые 6 битов EOF, поле простоя INT
01	Новый бит. Код указывает на то, что обнаружен первый бит сегмента сообщения. Текущий бит является первым битом одного из следующих сегментов: Бит 10 стандартного идентификатора ID10 (только при передаче), бит RTR, зарезервированные биты, бит IDE, старший бит поля длины данных DLC, седьмой бит каждого байта данных, первый бит расширения идентификатора EID17
10	Бит. Этот код указывает на то, что обнаруженный бит, является битом сегмента, длиной два и более бит, и не является первым битом сегмента. Текущий бит является битом одного из следующих сегментов: идентификатора (за исключением первого бита стандартного идентификатора и первого бита расширения идентификатора), бит поля длины данных DLC, биты с 6 по 0 каждого байта данных
11	Последний бит. Код указывает на то, что обнаружен последний бит сегмента. 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 «последних» бита формируют сигнал ошибки сообщения

Регистры объектов сообщений

Каждый из 64 объектов сообщений имеет набор из восьми регистров: MOARn, MODATAHn, MODATALn, MOAMRn, MOIPRn, MOFGPRn, MOFCRn и MOCTRn/MOSTATn (два регистра, обращение к которым производится по одному адресу – при записи данные попадают в MOCTRn, а при чтении возвращается значение из MOSTATn). Все наборы регистров идентичны, n – является номером объекта сообщения от 0 до 63. Все регистры имеют уникальные адреса. В таблице А.134 приведены адреса регистров, в зависимости от номера n объекта сообщения, которому они принадлежат.

Таблица А.134 – Адреса регистров объектов сообщений в зависимости от номера n объекта

n	MOCTR MOSTAT	MOAR	MODATAH	MODATAL	MOAMR	MOIPR	MOFGPR	MOFCR
0	EC1Eh EC1Ch	EC1Ah EC18h	EC16h EC14h	EC12h EC10h	EC0Eh EC0Ch	EC0Ah EC08h	EC06h EC04h	EC02h EC00h
1	EC3Eh EC3Ch	EC3Ah EC38h	EC36h EC34h	EC32h EC30h	EC2Eh EC2Ch	EC2Ah EC28h	EC26h EC24h	EC22h EC20h
2	EC5Eh EC5Ch	EC5Ah EC58h	EC56h EC54h	EC52h EC50h	EC4Eh EC4Ch	EC4Ah EC48h	EC46h EC44h	EC42h EC40h
3	EC7Eh EC7Ch	EC7Ah EC78h	EC76h EC74h	EC72h EC70h	EC6Eh EC6Ch	EC6Ah EC68h	EC66h EC64h	EC62h EC60h
4	EC9Eh EC9Ch	EC9Ah EC98h	EC96h EC94h	EC92h EC90h	EC8Eh EC8Ch	EC8Ah EC88h	EC86h EC84h	EC82h EC80h

Продолжение таблицы А.134

n	MOCTR MOSTAT	MOAR	MODATAH	MODATAL	MOAMR	MOIPR	MOFGPR	MOFCR
5	ECBEh ECBCh	ECBAh ECB8h	ECB6h ECB4h	ECB2h ECB0h	ECAEh ECACh	ECAAh ECA8h	ECA6h ECA4h	ECA2h ECA0h
6	ECDEh ECDCh	ECDAh ECD8h	ECD6h ECD4h	ECD2h ECD0h	ECCEh ECCCh	ECCAh ECC8h	ECC6h ECC4h	ECC2h ECC0h
7	ECFEh ECFCh	ECFAh ECF8h	ECF6h ECF4h	ECF2h ECF0h	ECEEh ECECh	ECEAh ECE8h	ECE6h ECE4h	ECE2h ECE0h
8	ED1Eh ED1Ch	ED1Ah ED18h	ED16h ED14h	ED12h ED10h	ED0Eh ED0Ch	ED0Ah ED08h	ED06h ED04h	ED02h ED00h
9	ED3Eh ED3Ch	ED3Ah ED38h	ED36h ED34h	ED32h ED30h	ED2Eh ED2Ch	ED2Ah ED28h	ED26h ED24h	ED22h ED20h
10	ED5Eh ED5Ch	ED5Ah ED58h	ED56h ED54h	ED52h ED50h	ED4Eh ED4Ch	ED4Ah ED48h	ED46h ED44h	ED42h ED40h
11	ED7Eh ED7Ch	ED7Ah ED78h	ED76h ED74h	ED72h ED70h	ED6Eh ED6Ch	ED6Ah ED68h	ED66h ED64h	ED62h ED60h
12	ED9Eh ED9Ch	ED9Ah ED98h	ED96h ED94h	ED92h ED90h	ED8Eh ED8Ch	ED8Ah ED88h	ED86h ED84h	ED82h ED80h
13	EDBEh EDBCh	EDBAh EDB8h	EDB6h EDB4h	EDB2h EDB0h	EDAEh EDACh	EDAAh EDA8h	EDA6h EDA4h	EDA2h EDA0h
14	EDDEh EDDCh	EDDAh EDD8h	EDD6h EDD4h	EDD2h EDD0h	EDCEh EDCCh	EDCAh EDC8h	EDC6h EDC4h	EDC2h EDC0h
15	EDFEh EDFCh	EDFAh EDF8h	EDF6h EDF4h	EDF2h EDF0h	EDEEh EDECh	EDEAh EDE8h	EDE6h EDE4h	EDE2h EDE0h
16	EE1Eh EE1Ch	EE1Ah EE18h	EE16h EE14h	EE12h EE10h	EE0Eh EE0Ch	EE0Ah EE08h	EE06h EE04h	EE02h EE00h
17	EE3Eh EE3Ch	EE3Ah EE38h	EE36h EE34h	EE32h EE30h	EE2Eh EE2Ch	EE2Ah EE28h	EE26h EE24h	EE22h EE20h
18	EE5Eh EE5Ch	EE5Ah EE58h	EE56h EE54h	EE52h EE50h	EE4Eh EE4Ch	EE4Ah EE48h	EE46h EE44h	EE42h EE40h
19	EE7Eh EE7Ch	EE7Ah EE78h	EE76h EE74h	EE72h EE70h	EE6Eh EE6Ch	EE6Ah EE68h	EE66h EE64h	EE62h EE60h
20	EE9Eh EE9Ch	EE9Ah EE98h	EE96h EE94h	EE92h EE90h	EE8Eh EE8Ch	EE8Ah EE88h	EE86h EE84h	EE82h EE80h
21	EEBEh EEBCh	EEBAh EEB8h	EEB6h EEB4h	EEB2h EEB0h	EEAEh EEACh	EEAAh EEA8h	EEA6h EEA4h	EEA2h EEA0h
22	EEDEh EEDCh	EEDAh EED8h	EED6h EED4h	EED2h EED0h	EECEh EECCh	EECAh EEC8h	EEC6h EEC4h	EEC2h EEC0h
23	EEFEh EEFCh	EEFAh EEF8h	EEF6h EEF4h	EEF2h EEF0h	EEEEh EEECh	EEEAh EEE8h	EEE6h EEE4h	EEE2h EEE0h
24	EF1Eh EF1Ch	EF1Ah EF18h	EF16h EF14h	EF12h EF10h	EF0Eh EF0Ch	EF0Ah EF08h	EF06h EF04h	EF02h EF00h
25	EF3Eh EF3Ch	EF3Ah EF38h	EF36h EF34h	EF32h EF30h	EF2Eh EF2Ch	EF2Ah EF28h	EF26h EF24h	EF22h EF20h
26	EF5Eh EF5Ch	EF5Ah EF58h	EF56h EF54h	EF52h EF50h	EF4Eh EF4Ch	EF4Ah EF48h	EF46h EF44h	EF42h EF40h
27	EF7Eh EF7Ch	EF7Ah EF78h	EF76h EF74h	EF72h EF70h	EF6Eh EF6Ch	EF6Ah EF68h	EF66h EF64h	EF62h EF60h

Продолжение таблицы А.134

n	MOCTR MOSTAT	MOAR	MODATAH	MODATAL	MOAMR	MOIPR	MOFGPR	MOFCR
28	EF9Eh EF9Ch	EF9Ah EF98h	EF96h EF94h	EF92h EF90h	EF8Eh EF8Ch	EF8Ah EF88h	EF86h EF84h	EF82h EF80h
29	EFBEh EFBCh	EFBAh EFB8h	EFB6h EFB4h	EFB2h EFB0h	EFAEh EFACh	EFAAh EFA8h	EFA6h EFA4h	EFA2h EFA0h
30	EFDEh EFDCh	EFDAh EFD8h	EFD6h EFD4h	EFD2h EFD0h	EFCEh EFCCh	EFCAh EFC8h	EFC6h EFC4h	EFC2h EFC0h
31	EFFEh EFFCh	EFFAh EFF8h	EFF6h EFF4h	EFF2h EFF0h	EFEEh EFECh	EFEAh EFE8h	EFE6h EFE4h	EFE2h EFE0h
32	F01Eh F01Ch	F01Ah F018h	F016h F014h	F012h F010h	F00Eh F00Ch	F00Ah F008h	F006h F004h	F002h F000h
33	F03Eh F03Ch	F03Ah F038h	F036h F034h	F032h F030h	F02Eh F02Ch	F02Ah F028h	F026h F024h	F022h F020h
34	F05Eh F05Ch	F05Ah F058h	F056h F054h	F052h F050h	F04Eh F04Ch	F04Ah F048h	F046h F044h	F042h F040h
35	F07Eh F07Ch	F07Ah F078h	F076h F074h	F072h F070h	F06Eh F06Ch	F06Ah F068h	F066h F064h	F062h F060h
36	F09Eh F09Ch	F09Ah F098h	F096h F094h	F092h F090h	F08Eh F08Ch	F08Ah F088h	F086h F084h	F082h F080h
37	F0BEh F0BCh	F0BAh F0B8h	F0B6h F0B4h	F0B2h F0B0h	F0AEh F0ACh	F0AAh F0A8h	F0A6h F0A4h	F0A2h F0A0h
38	F0DEh F0DCh	F0DAh F0D8h	F0D6h F0D4h	F0D2h F0D0h	F0CEh F0CCh	F0CAh F0C8h	F0C6h F0C4h	F0C2h F0C0h
39	F0FEh F0FCh	F0FAh F0F8h	F0F6h F0F4h	F0F2h F0F0h	F0EEh F0ECh	F0EAh F0E8h	F0E6h F0E4h	F0E2h F0E0h
40	F11Eh F11Ch	F11Ah F118h	F116h F114h	F112h F110h	F10Eh F10Ch	F10Ah F108h	F106h F104h	F102h F100h
41	F13Eh F13Ch	F13Ah F138h	F136h F134h	F132h F130h	F12Eh F12Ch	F12Ah F128h	F126h F124h	F122h F120h
42	F15Eh F15Ch	F15Ah F158h	F156h F154h	F152h F150h	F14Eh F14Ch	F14Ah F148h	F146h F144h	F142h F140h
43	F17Eh F17Ch	F17Ah F178h	F176h F174h	F172h F170h	F16Eh F16Ch	F16Ah F168h	F166h F164h	F162h F160h
44	F19Eh F19Ch	F19Ah F198h	F196h F194h	F192h F190h	F18Eh F18Ch	F18Ah F188h	F186h F184h	F182h F180h
45	F1BEh F1BCh	F1BAh F1B8h	F1B6h F1B4h	F1B2h F1B0h	F1AEh F1ACh	F1AAh F1A8h	F1A6h F1A4h	F1A2h F1A0h
46	F1DEh F1DCh	F1DAh F1D8h	F1D6h F1D4h	F1D2h F1D0h	F1CEh F1CCh	F1CAh F1C8h	F1C6h F1C4h	F1C2h F1C0h
47	F1FEh F1FCh	F1FAh F1F8h	F1F6h F1F4h	F1F2h F1F0h	F1EEh F1ECh	F1EAh F1E8h	F1E6h F1E4h	F1E2h F1E0h
48	F21Eh F21Ch	F21Ah F218h	F216h F214h	F212h F210h	F20Eh F20Ch	F20Ah F208h	F206h F204h	F202h F200h
49	F23Eh F23Ch	F23Ah F238h	F236h F234h	F232h F230h	F22Eh F22Ch	F22Ah F228h	F226h F224h	F222h F220h
50	F25Eh F25Ch	F25Ah F258h	F256h F254h	F252h F250h	F24Eh F24Ch	F24Ah F248h	F246h F244h	F242h F240h

Окончание таблицы А.134

n	MOCTR MOSTAT	MOAR	MODATAH	MODATAL	MOAMR	MOIPR	MOFGPR	MOFCR
51	F27Eh F27Ch	F27Ah F278h	F276h F274h	F272h F270h	F26Eh F26Ch	F26Ah F268h	F266h F264h	F262h F260h
52	F29Eh F29Ch	F29Ah F298h	F296h F294h	F292h F290h	F28Eh F28Ch	F28Ah F288h	F286h F284h	F282h F280h
53	F2BEh F2BCh	F2BAh F2B8h	F2B6h F2B4h	F2B2h F2B0h	F2AEh F2ACh	F2AAh F2A8h	F2A6h F2A4h	F2A2h F2A0h
54	F2DEh F2DCh	F2DAh F2D8h	F2D6h F2D4h	F2D2h F2D0h	F2CEh F2CCh	F2CAh F2C8h	F2C6h F2C4h	F2C2h F2C0h
55	F2FEh F2FCh	F2FAh F2F8h	F2F6h F2F4h	F2F2h F2F0h	F2EEh F2ECh	F2EAh F2E8h	F2E6h F2E4h	F2E2h F2E0h
56	F31Eh F31Ch	F31Ah F318h	F316h F314h	F312h F310h	F30Eh F30Ch	F30Ah F308h	F306h F304h	F302h F300h
57	F33Eh F33Ch	F33Ah F338h	F336h F334h	F332h F330h	F32Eh F32Ch	F32Ah F328h	F326h F324h	F322h F320h
58	F35Eh F35Ch	F35Ah F358h	F356h F354h	F352h F350h	F34Eh F34Ch	F34Ah F348h	F346h F344h	F342h F340h
59	F37Eh F37Ch	F37Ah F378h	F376h F374h	F372h F370h	F36Eh F36Ch	F36Ah F368h	F366h F364h	F362h F360h
60	F39Eh F39Ch	F39Ah F398h	F396h F394h	F392h F390h	F38Eh F38Ch	F38Ah F388h	F386h F384h	F382h F380h
61	F3BEh F3BCh	F3BAh F3B8h	F3B6h F3B4h	F3B2h F3B0h	F3AEh F3ACh	F3AAh F3A8h	F3A6h F3A4h	F3A2h F3A0h
62	F3DEh F3DCh	F3DAh F3D8h	F3D6h F3D4h	F3D2h F3D0h	F3CEh F3CCh	F3CAh F3C8h	F3C6h F3C4h	F3C2h F3C0h
63	F3FEh F3FCh	F3FAh F3F8h	F3F6h F3F4h	F3F2h F3F0h	F3EEh F3ECh	F3EAh F3E8h	F3E6h F3E4h	F3E2h F3E0h

Примечание – Первым указывается адрес старшего слова регистра, вторым – младшего.

Таблица А.135 – Состояние регистров объектов сообщений после сброса

n (код)	MOCTRn / MOSTATn			MOAMRn
	PNEXT (биты 31–24)	PPREV (биты 23–16)	Биты 15–0	
0 (00h)	01h	00h	0000h	3FFF FFFFh
1 (01h)	02h	00h		
2 (02h)	03h	01h		
3 (03h)	04h	02h		
…	…	…		
60 (3C)	3D	3B		
61 (3D)	3E	3C		
62 (3E)	3F	3D		
63 (3F)	3F	3E		

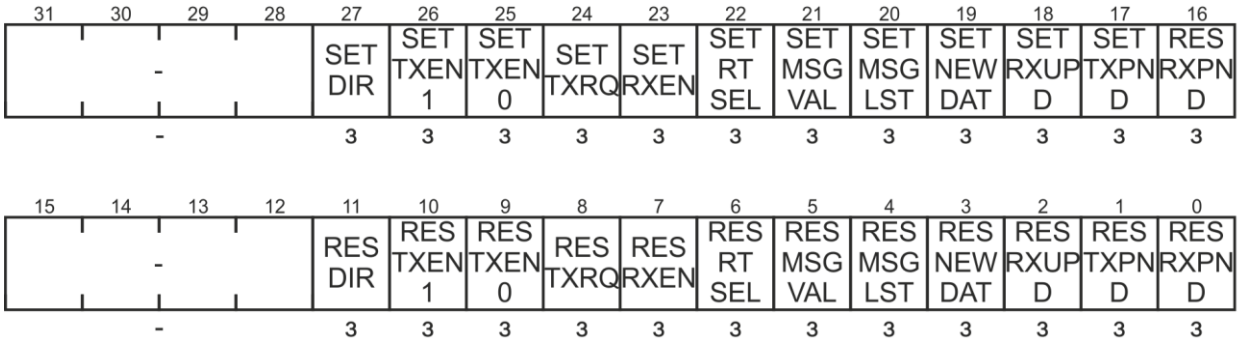
Примечание – Регистры MOARn, MODATAHn, MODATALn, MOIPRn, MOFGPRn и MOFCRn после сброса микроконтроллера всегда инициализируются в 0000 0000h.

В таблице А.135 указаны состояния регистров объектов сообщений после сброса микроконтроллера. В таблицах А.136 – А.144 приведено подробное описание регистров

объектов сообщений. Все регистры 32-разрядные. Биты 31–16 составляют старшее слово регистра, биты 15–0 составляют младшее слово.

Таблица А.136 – Регистр управления объектом сообщения

MOCTRn



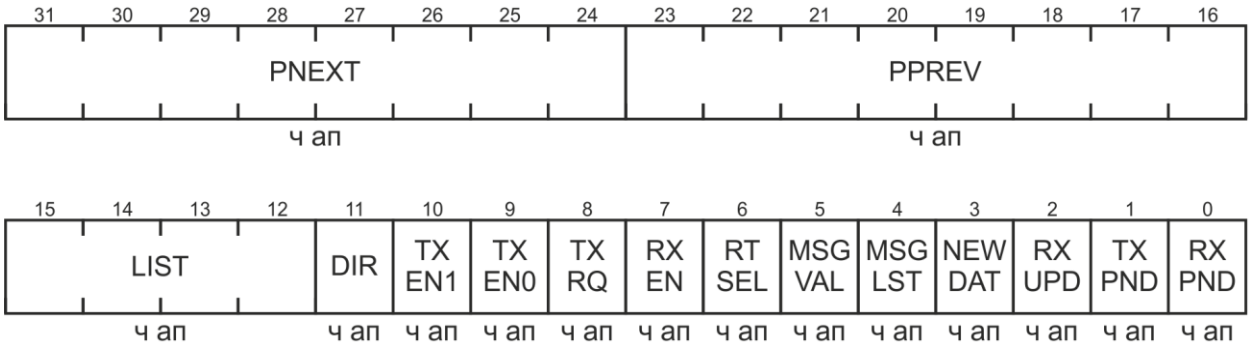
Биты регистра работают попарно. Комбинация состояний бит каждой пары оказывает влияние на один (соответствующий этой паре) бит регистра MOSTATn того же объекта сообщения. Так, пара SETDIR-RESDIR устанавливает и сбрасывает бит DIR, пара SETTXN1-RESTXN1 устанавливает и сбрасывает бит TXEN1 и т.д.

После записи старшего или младшего слова регистра MOCTRn аппаратная часть проверяет состояние бит каждой пары и, в зависимости от обнаруженной комбинации, выполняет соответствующее действие

Бит SET***	Бит RES***	Действие над битом ***
0	0	Нет
1	1	
1	0	Установка
0	1	Сброс
Биты 31–26 и 15–12 являются зарезервированными		

Таблица А.137 – Регистр состояния объекта сообщения

MOSTATn



Поле	Бит	Описание
1	2	3
PNEXT	31–24	Указатель на следующий элемент списка. В поле находится номер объекта сообщения, расположенного выше по списку относительно текущего
PPREV	23–16	Указатель на предыдущий элемент списка. В поле находится номер объекта сообщения, расположенного ниже по списку относительно текущего

Продолжение таблицы А.137

1	2	3	
LIST	15–12	Номер списка, которому принадлежит объект сообщения n. Поле обновляется аппаратно при распределении/перераспределении объекта сообщения.	
DIR	11	Бит распределения	
		0	Объект приема. При установленном TXRQ формируется фрейм удаленного запроса с идентификатором объекта сообщения n и передается. Полученный в ответ фрейм данных с соответствующим идентификатором сохраняется в объекте сообщения n
		1	Объект передачи. При получении фрейма удаленного запроса с соответствующим идентификатором устанавливается флаг TXRQ, после чего в ответ передается фрейм данных, расположенный в объекте сообщения n. Передача фрейма данных может быть инициирована программной установкой бита TXRQ
TXEN1	10	Бит разрешения передачи фрейма	
		0	Запрещено
		1	Передача фрейма разрешена. Объект сообщения n может участвовать в передаче только, если установлены оба бита – TXEN1 и TXEN0. Контроллер CAN использует бит TXEN1 для выбора активного объекта передачи сообщения из FIFO
TXEN0	9	Бит разрешения передачи фрейма	
		0	Запрещено
		1	Передача фрейма разрешена. Объект сообщения n может участвовать в передаче только если установлены оба бита – TXEN0 и TXEN1. Контроллер CAN использует бит TXEN1 для выбора активного объекта передачи сообщения из FIFO. Можно программно очищать бит TXEN0 для запрета передачи сообщения, которое в настоящий момент формируется, или для запрета автоматической передачи в ответ на удаленный запрос
TXRQ	8	Бит инициации передачи	
		0	Нет действий
		1	Установка бита инициирует передачу фрейма из объекта сообщения n. Инициация передачи фрейма возможна только в случае, если установлены биты TXRQ, TXEN0, TXEN1 и MSGVAL. Также бит TXRQ устанавливается аппаратно при получении фрейма удаленного запроса. Бит сбрасывается аппаратно при успешном завершении передачи и если при этом не был повторно программно установлен бит NEWDAT
RXEN	7	Бит разрешения приема	
		0	Запрещено
		1	Объект сообщения может принимать сообщения
		Состояние бита учитывается только при фильтрации принимаемых сообщений	

Продолжение таблицы А.137

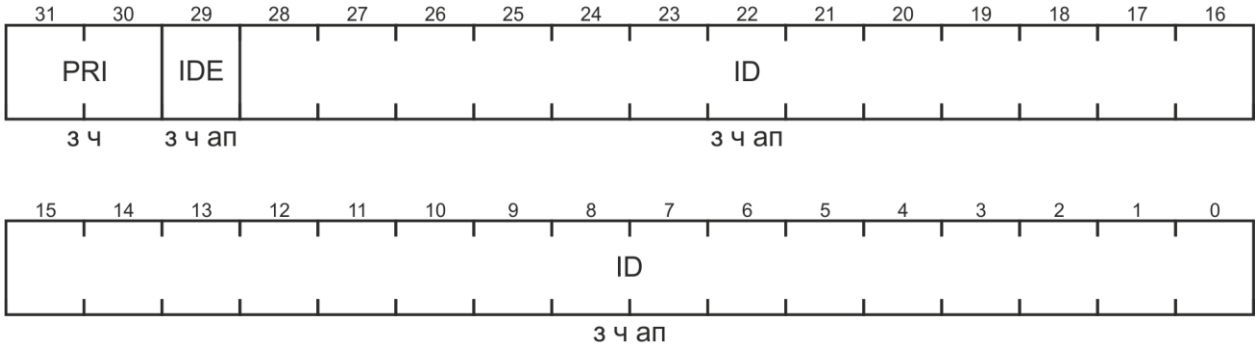
1	2	3	
RTSEL	6	Индикатор возможности приема/передачи	
		0	Объект сообщения не может принимать/передавать сообщения
		1	Объект сообщения может принимать/передавать сообщения
		Прием фрейма. Бит RTSEL устанавливается аппаратно после того, как выбран объект сообщения n для сохранения только что принятого фрейма. Прежде, чем записать принятые данные в объект сообщения n, аппаратная часть проверяет состояние бита RTSEL. ЦПУ может сбрасывать этот бит, чтобы запретить запись принятого фрейма в объект сообщения n. Передача фрейма. Бит RTSEL устанавливается аппаратно после того, как выбран следующий объект сообщения n для передачи фрейма. Аппаратная часть перед началом передачи проверяет: установлен ли бит RTSEL и сброшен ли бит NEWDAT. Бит RTSEL должен оставаться установленным до окончания передачи. Проверка состояния бита RTSEL производится только при попытке изменения содержимого объекта сообщения n во избежание одновременного выполнения операций передачи фрейма и его изменения. Бит не участвует в фильтрации сообщений, и не сбрасывается аппаратно	
MSGVAL	5	Бит активности объекта сообщения n	
		0	Не активен
		1	Активен
		Только те объекты сообщений, для которых установлен этот бит, могут использоваться для операций приема и передачи	
MSGLST	4	Бит потери сообщения	
		0	Ни одно сообщение не потеряно
		1	Принятое сообщение потеряно вследствие того, что контроллер CAN попытался установить бит NEWDAT по окончании приема сообщения при том, что флаг NEWDAT уже был установлен ранее после записи другого сообщения
NEWDAT	3	Индикатор новых данных	
		0	С момента сброса бита NEWDAT никаких изменений объекта сообщений n не обнаружено
		1	Объект сообщения был изменен. Бит устанавливается аппаратно после того, как принятое сообщение было сохранено в объекте сообщения n. Бит сбрасывается аппаратно после начала передачи объекта сообщения n. Бит NEWDAT следует устанавливать программно после того, как новые данные для передачи будут сохранены в объекте сообщения n для предотвращения автоматического сброса бита TRXQ в конце текущей передачи
RXUPD	2	Индикатор изменений	
		0	Нет текущих изменений
		1	Идентификатор сообщения, поле длины данных DLC и данные в объекте сообщения изменяются
TXPND	1	Индикатор окончания передачи	
		0	Переданных сообщений нет
		1	Сообщение объекта сообщения n было успешно передано

Окончание таблицы А.137

1	2	3	
RXPND	0	Индикатор окончания приема	
		0	Принятых сообщений нет
		1	Сообщение было успешно принято объектом сообщения n (напрямую или через шлюз). Бит должен сбрасываться программно

Таблица А.138 – Регистр арбитража объекта сообщения n

MOARn



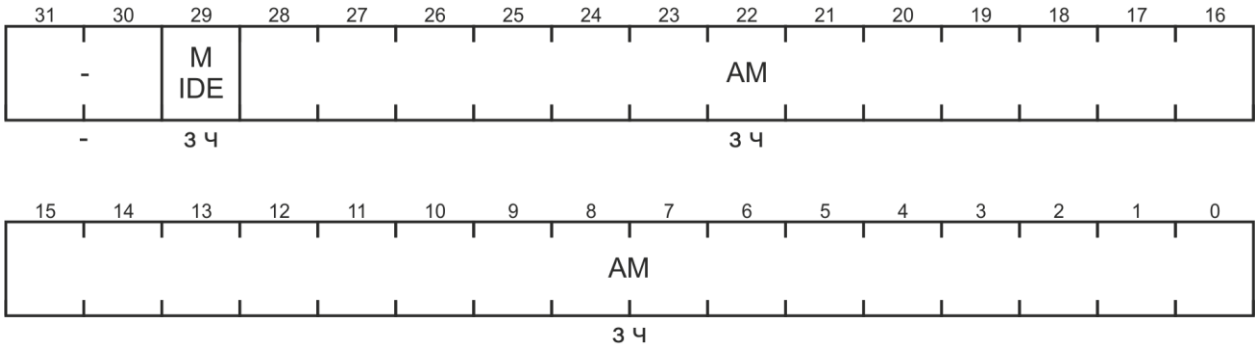
Поле	Бит	Описание	
PRI	31–30	Класс приоритета. Поле определяет один из четырех классов (0, 1, 2 и 3) приоритета объекта сообщения n. Нулевой класс устанавливает наивысший приоритет. Объекты сообщений с нулевым классом всегда выигрывают арбитраж при передаче и приеме сообщений. Фильтрация сообщений на основе идентификатора (маскируемого) и позиции в списке организуются только для объектов сообщений с равным приоритетом. Кроме этого, поле PRI определяет метод фильтрации	
		00	Зарезервировано
		01	Фильтрация в зависимости от положения объекта сообщения в списке. Объект сообщения n получает приоритет на передачу сообщения только в случае, если нет других объектов сообщений с установленными битами MSGVAL, TXEN0 и TXEN1, стоящих выше по списку
		10	Фильтрация в зависимости от значения идентификатора. Объект сообщения n получает приоритет на передачу сообщения только в случае, если в списке нет других объектов сообщений с «Идентификатор + IDE + DIR» более высокого приоритета (согласно правилам арбитража)
		11	Фильтрация в зависимости от положения объекта сообщения в списке (как при PRI = 01b)
IDE	29	Бит расширения идентификатора объекта сообщения n	
		0	Объект сообщения n оперирует с фреймами со стандартным 11-битным идентификатором
		1	Объект сообщения n оперирует с фреймами с расширенным 29-битным идентификатором
ID	28–0	Идентификатор объекта сообщения n. При оперировании с расширенными фреймами используются биты 28–0. При оперировании со стандартными фреймами используются биты 28–18, при этом состояние битов 17–0 не важно	

Таблица А.139 – Распределение приоритета между объектами сообщений согласно правилам арбитража

Установки для объектов сообщений 0 и 1, которые участвуют в арбитраже (приоритет объекта 0 выше приоритета объекта 1)	Пояснение
MOAR0[28:18] < MOAR1[28:18] (11-битный стандартный идентификатор объекта 0 меньше по числовому значению, чем 11-битный идентификатор объекта 1)	Стандартный фрейм с идентификатором, имеющим меньшее значение, обладает более высоким приоритетом
MOAR0[28:18] = MOAR1[28:18]. В регистре MOAR0 бит IDE = 0. В регистре MOAR1 бит IDE = 1.	При равенстве значений стандартных идентификаторов, стандартный фрейм имеет приоритет перед расширенным
MOAR0[28:18] = MOAR1[28:18]. Биты IDE обоих объектов сброшены. В регистре MOSTAT0 бит DIR = 1. В регистре MOSTAT1 бит DIR = 0.	При равенстве значений идентификаторов стандартный фрейм данных имеет приоритет перед стандартным фреймом удаленного запроса
MOAR0[28:0] = MOAR1[28:0] Биты IDE обоих объектов установлены. В регистре MOSTAT0 бит DIR = 1. В регистре MOSTAT1 бит DIR = 0.	При равенстве значений идентификаторов расширенный фрейм данных имеет приоритет перед расширенным фреймом удаленного запроса
MOAR0[28:0] < MOAR1[28:0] Биты IDE обоих объектов установлены. (29-битный идентификатор объекта 0 меньше по числовому значению, чем 29-битный идентификатор объекта 1)	Расширенный фрейм с идентификатором, имеющим меньшее значение, обладает более высоким приоритетом

Таблица А.140 – Регистр маски объекта сообщения n

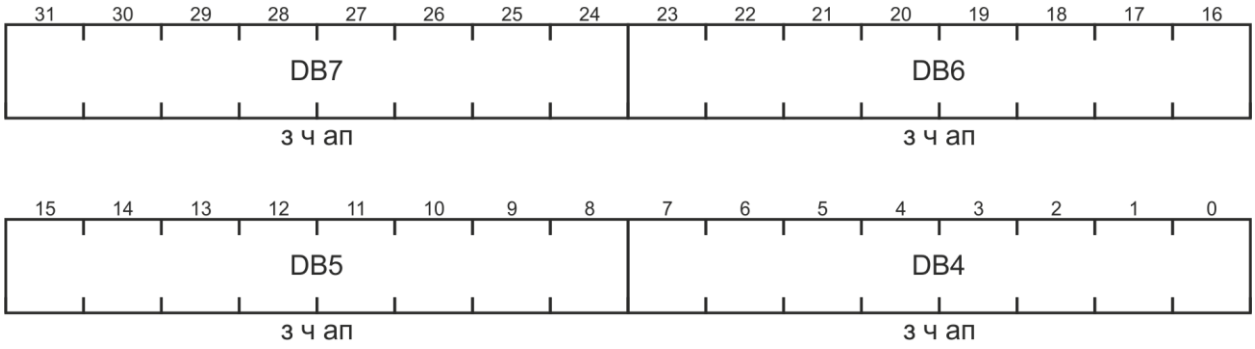
MOAMRn



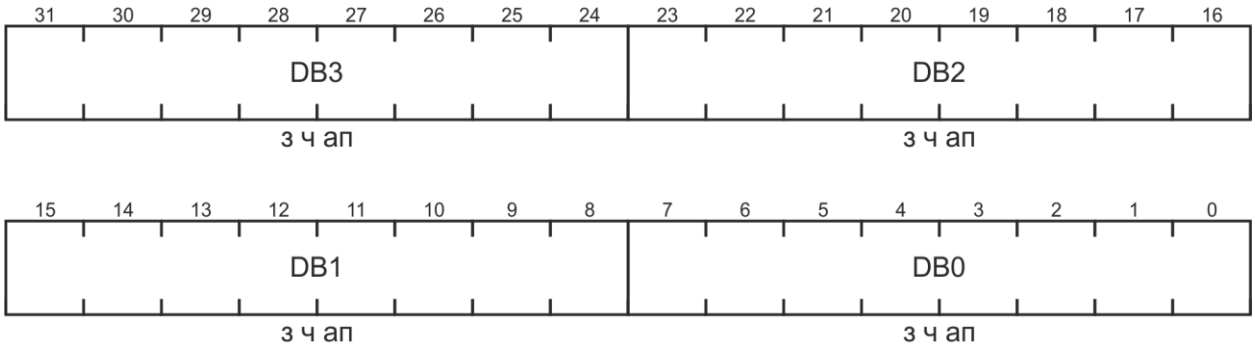
Поле	Бит	Описание	
MIDE	29	Маска бита IDE сообщения	
		0	Объект сообщения n может принимать как стандартные, так и расширенные фреймы
		1	Объект сообщения n может принимать только те фреймы, у которых состояние бита IDE совпадает с его битом IDE
AM	28–0	Маска идентификатора. При приеме расширенного сообщения используется вся маска. При приеме стандартного сообщения используются биты 28–18, при этом состояние битов 17–0 не важно	
–	31, 30	Зарезервировано	

Таблица А.141 – Регистры данных объекта сообщения n

MODATAHn



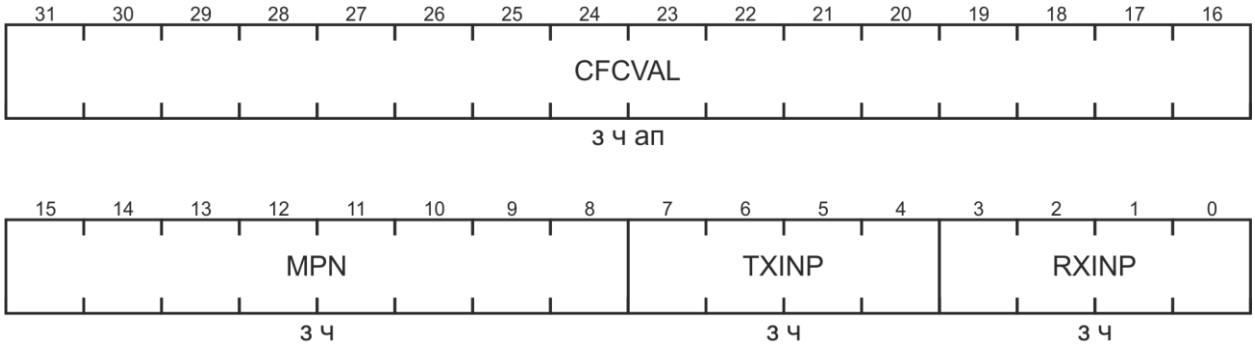
MODATALn



Поле	Бит	Описание
DB7	31–24	Седьмой байт данных
DB6	23–16	Шестой байт данных
DB5	15–8	Пятый байт данных
DB4	7–0	Четвертый байт данных
DB3	31–24	Третий байт данных
DB2	23–16	Второй байт данных
DB1	15–8	Первый байт данных
DB0	7–0	Нулевой байт данных

Таблица А.142 – Регистр указателя прерываний объекта сообщения n

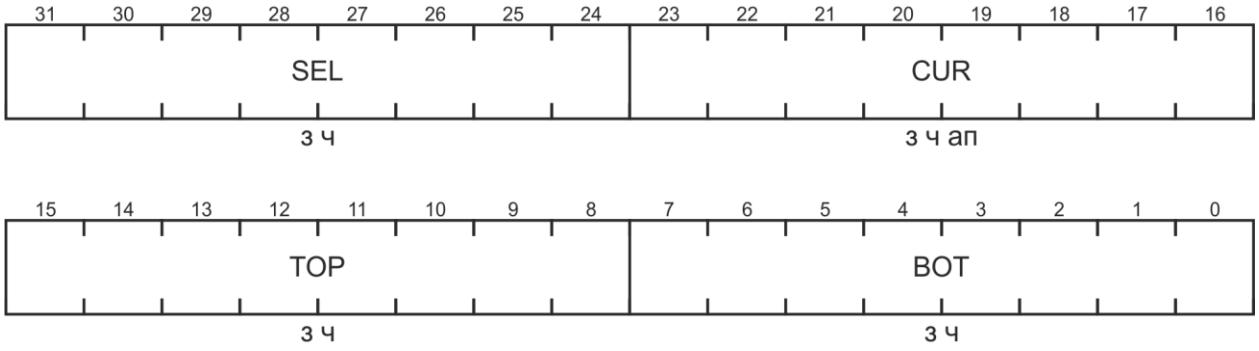
MOIPRn



Поле	Бит	Описание
CFCVAL	31–16	Количество фреймов. Каждый раз после записи принятого сообщение в объект сообщения n или успешной передачи объекта сообщения n, значение счетчика фреймов CFC (регистр NFCRn) копируется в CFCVAL
MPN	15–8	Номер ждущего бита сообщения. Указывает позицию бита, соответствующего объекту сообщения n в регистре MSPNDx
TXINP	7–4	Указатель линии прерываний для прерывания после передачи. Всего доступно 16 линий прерываний с номерами от 0 до 15. Значение 0000b, записанное в TXINP, выбирает нулевую линию прерываний, 0001b – первую, 0010b – вторую и т.д. Дополнительно бит TXINP используется для выбора позиции ждущего бита объекта сообщения n.
RXINP	3–0	Указатель линии прерываний для прерывания после приема. Всего доступно 16 линий прерываний с номерами от 0 до 15. Значение 0000b, записанное в TXINP, выбирает нулевую линию прерываний, 0001b – первую, 0010b – вторую и т.д. Дополнительно бит RXINP используется для выбора позиции ждущего бита объекта сообщения n

Таблица А.143 – Регистр указателя FIFO/шлюза объекта сообщения n

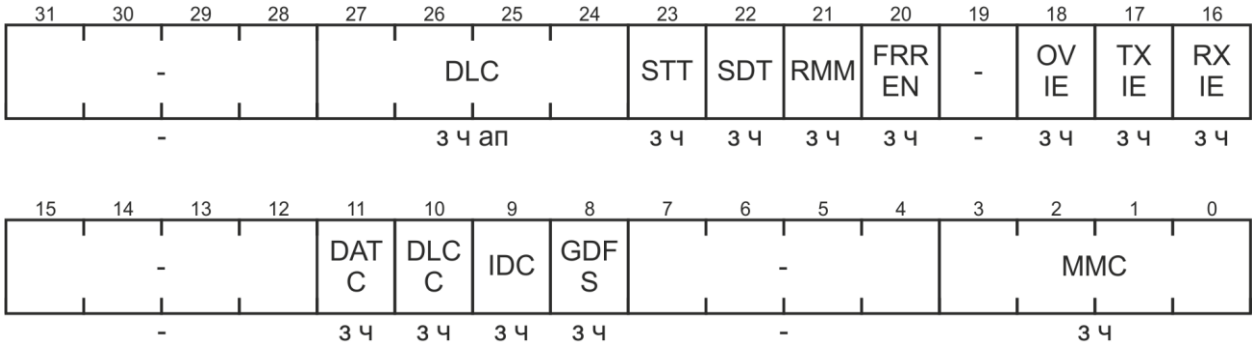
MOFGPRn



Поле	Бит	Описание
SEL	31–24	Указатель объекта сообщения. Второй (программный) указатель в дополнение к аппаратному указателю CUR при работе с FIFO. Поле SEL используется для общего мониторинга (генерирование прерываний FIFO)
CUR	23–16	Указатель на текущий объект в пределах FIFO или шлюза. После каждой операции FIFO или передачи через шлюз указатель CUR обновляется – в него заносится номер следующего объекта сообщения в списке (поле PNEXT регистра MOSTATn) – до тех пор, пока не будет достигнут верхний элемент FIFO (поле TOP), после чего CUR сбрасывается, и в него загружается номер нижнего элемента списка (из поля BOT)
TOP	15–8	Указатель верхнего элемента FIFO. В поле находится номер последнего элемента
BOT	7–0	Указатель нижнего элемента FIFO. В поле находится номер первого элемента

Таблица А.144 – Регистр управления функционированием объекта сообщения n

MOFCRn



Поле	Бит	Описание	
1	2	3	
DLC	27–24	Код длины данных. Показывает количество байт данных, находящихся в объекте сообщения. Диапазон – значение от 0 до 8. Если значение DLC больше 8, это автоматически указывает на 8 байт. Значение DLC полученного сообщения сохраняется таким, каким было получено	
STT	23	Бит задания однократной пересылки данных	
		0	Нет действий
		1	Если бит установлен, тогда бит TXRQ сбрасывается после начала передачи объекта сообщения n. В связи с этим, в случае неудачной передачи, повторной передачи сообщения не будет
SDT	22	Бит задания однократного участия объекта сообщения n в пересылке	
		0	Нет действий
		1	Если бит установлен и объект сообщения n не является объектом FIFO, тогда бит MSGVAL сбрасывается после успешной пересылки данных (приема или передачи). В случае, если объект сообщения является объектом FIFO, то бит MSGVAL сбрасывается, когда указатель CUR (указатель на текущий объект) достигает значения SEL в регистре MOFGPRn
RMM	21	Бит включения удаленного мониторинга объекта передачи	
		0	Выключен. Идентификатор, бит IDE и поле DLC объекта сообщения n остаются без изменений до получения корректного фрейма удаленного запроса
		1	Включен. Идентификатор, бит IDE и поле DLC корректного фрейма удаленного запроса копируются в объект передачи n в порядке получения битов фрейма удаленного запроса монитора
		Состояние бита оказывает влияние только на объекты передач	
FRREN	20	Бит разрешения удаленного запроса. Определяет, будет ли устанавливаться бит TXRQ в объекте сообщения n или в другом объекте сообщения, на который указывает CUR	
		0	Бит TXRQ объекта сообщения n устанавливается после получения корректного фрейма удаленного запроса
		1	Бит TXRQ другого объекта сообщения (на который указывает CUR) устанавливается после получения им корректного фрейма удаленного запроса

Продолжение таблицы А.144

1	2	3
OVIE	18	Бит разрешения прерывания по заполнению FIFO объекта сообщения n. Прерывание генерируется, когда указатель CUR (указатель на текущий объект) достигает значения SEL регистра MOFGPRn
		0 Запрещено
		1 Разрешено
		Если объект сообщения n является объектом приема FIFO, то поле TXINP (регистр MOIPRn) указывает на одну из 16 линий прерываний. Если объект сообщения n является объектом передачи FIFO, то поле RXINP (регистр MOIPRn) указывает на одну из 16 линий прерываний. Для всех других режимов объекта сообщения состояние бита OVIE не важно
TXIE	17	Бит разрешения прерывания по окончании передачи сообщения
		0 Запрещено
		1 Разрешено. Прерывание генерируется, если сообщение из объекта сообщения n было успешно передано. Поле TXINP (регистр MOIPRn) указывает на одну из 16 линий прерываний
RXIE	16	Бит разрешения прерывания по окончании приема сообщения
		0 Запрещено
		1 Разрешено. Прерывание генерируется, если сообщение было успешно принято объектом сообщения n (напрямую или через шлюз). Поле RXINP (регистр MOIPRn) указывает на одну из 16 линий прерываний
DATC	11	Индикатор копирования данных
		0 Данные не копируются
		1 Данные в регистрах MODATAHn и MODATALn объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируются через шлюз в объект-приемник
		Бит DATC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует
DLCC	10	Индикатор копирования кода длины данных DLC
		0 Код не копируется
		1 Код длины данных объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируется через шлюз в объект-приемник
		Бит DLCC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует
IDC	9	Индикатор копирования идентификатора
		0 Идентификатор не копируется
		1 Идентификатор объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируется через шлюз в объект-приемник
		Бит IDC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует

Окончание таблицы А.144

1	2	3	
GDFS	8	Индикатор отправки фрейма через шлюз	
		0	Состояние бита TXRQ объекта-приемника без изменений
		1	Установлен бит TXRQ объекта-приемника после внутренней передачи из объекта-источника
		Бит GDFS используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует	
MMC	3–0	Задание режима объекта сообщения n	
		0000	Объект сообщения
		0001	Объект-приемник FIFO
		0010	Объект-передатчик FIFO
		0011	Ведомый объект-передатчик FIFO
		0100	Объект-источник шлюза
		Остальные комбинации зарезервированы	
–	31–28, 19, 15–12, 7–4	Зарезервировано	
Примечание – Под корректным фреймом удаленного запроса подразумевается фрейм, идентификатор которого совпадает с идентификатором объекта сообщения.			

A.14 Регистры блоков захвата/сравнения (CAPCOM1 и CAPCOM2)

Регистры блоков CAPCOM1 и CAPCOM2 представлены в таблицах A.145 – A.154.

Таблица A.145 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
CC1_T01CON	SFR-b	FF50h	A8h	0000h	Регистр управления таймерами 0 и 1
CC1_DRM	SFR-b	FF5Ah	ADh	0000h	Регистр двухрегистрового режима сравнения
CC1_IOC	ESFR	F062h	31h	0000h	Регистр управления вводом-выводом
CC1_OUT	SFR-b	FF5Ch	AEh	0000h	Регистр выводов сравнения
CC1_SEE	SFR	FE2Eh	17h	0000h	Регистр однократного срабатывания
CC1_SEM	SFR	FE2Ch	16h	0000h	Регистр режима однократного срабатывания
CC1_M0	SFR-b	FF52h	A9h	0000h	Регистр управления режимом захвата/сравнения 0
CC1_M1	SFR-b	FF54h	AAh	0000h	Регистр управления режимом захвата/сравнения 1
CC1_M2	SFR-b	FF56h	ABh	0000h	Регистр управления режимом захвата/сравнения 2
CC1_M3	SFR-b	FF58h	ACh	0000h	Регистр управления режимом захвата/сравнения 3
T0	SFR	FE50h	28h	0000h	Регистр таймера T0
T0REL	SFR	FE54h	2Ah	0000h	Регистр загрузки таймера 0
T1	SFR	FE52h	29h	0000h	Регистр таймера T1
T1REL	SFR	FE56h	2Bh	0000h	Регистр загрузки таймера 1
CC0	SFR	FE80h	40h	0000h	Регистр захвата/сравнения 0
CC1	SFR	FE82h	41h	0000h	Регистр захвата/сравнения 1
CC2	SFR	FE84h	42h	0000h	Регистр захвата/сравнения 2
CC3	SFR	FE86h	43h	0000h	Регистр захвата/сравнения 3
CC4	SFR	FE88h	44h	0000h	Регистр захвата/сравнения 4
CC5	SFR	FE8Ah	45h	0000h	Регистр захвата/сравнения 5
CC6	SFR	FE8Ch	46h	0000h	Регистр захвата/сравнения 6
CC7	SFR	FE8Eh	47h	0000h	Регистр захвата/сравнения 7
CC8	SFR	FE90h	48h	0000h	Регистр захвата/сравнения 8
CC9	SFR	FE92h	49h	0000h	Регистр захвата/сравнения 9
CC10	SFR	FE94h	4Ah	0000h	Регистр захвата/сравнения 10
CC11	SFR	FE96h	4Bh	0000h	Регистр захвата/сравнения 11
CC12	SFR	FE98h	4Ch	0000h	Регистр захвата/сравнения 12
CC13	SFR	FE9Ah	4Dh	0000h	Регистр захвата/сравнения 13
CC14	SFR	FE9Ch	4Eh	0000h	Регистр захвата/сравнения 14
CC15	SFR	FE9Eh	4Fh	0000h	Регистр захвата/сравнения 15

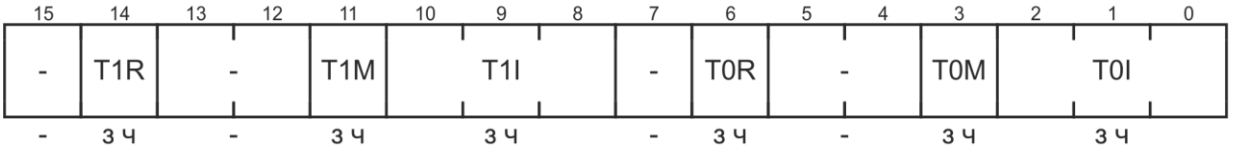
Таблица А.146 – Список регистров

Мнемо-ническое обозначение	Область памяти	Адрес		Сброс	Назначение
CC16	SFR	FE60h	30h	0000h	Регистр захвата/сравнения 16
CC17	SFR	FE62h	31h	0000h	Регистр захвата/сравнения 17
CC18	SFR	FE64h	32h	0000h	Регистр захвата/сравнения 18
CC19	SFR	FE66h	33h	0000h	Регистр захвата/сравнения 19
CC2_DRM	SFR-b	FF2Ah	95h	0000h	Регистр двухрегистрового режима сравнения
CC2_IOC	ESFR	F066h	33h	0000h	Регистр управления вводом/выводом
CC2_OUT	SFR-b	FF2Ch	96h	0000h	Регистр выводов сравнения
CC2_SEE	SFR	FE2Ah	15h	0000h	Регистр однократного срабатывания
CC2_SEM	SFR	FE28h	14h	0000h	Регистр режима однократного срабатывания
CC2_M4	SFR-b	FF22h	91h	0000h	Регистр управления режимом захвата/сравнения 4
CC2_M5	SFR-b	FF24h	92h	0000h	Регистр управления режимом захвата/сравнения 5
CC2_M6	SFR-b	FF26h	93h	0000h	Регистр управления режимом захвата/сравнения 6
CC2_M7	SFR-b	FF28h	94h	0000h	Регистр управления режимом захвата/сравнения 7
CC20	SFR	FE68h	34h	0000h	Регистр захвата/сравнения 20
CC21	SFR	FE6Ah	35h	0000h	Регистр захвата/сравнения 21
CC22	SFR	FE6Ch	36h	0000h	Регистр захвата/сравнения 22
CC23	SFR	FE6Eh	37h	0000h	Регистр захвата/сравнения 23
CC24	SFR	FE70h	38h	0000h	Регистр захвата/сравнения 24
CC25	SFR	FE72h	39h	0000h	Регистр захвата/сравнения 25
CC26	SFR	FE74h	3Ah	0000h	Регистр захвата/сравнения 26
CC27	SFR	FE76h	3Bh	0000h	Регистр захвата/сравнения 27
CC28	SFR	FE78h	3Ch	0000h	Регистр захвата/сравнения 28
CC29	SFR	FE7Ah	3Dh	0000h	Регистр захвата/сравнения 29
CC30	SFR	FE7Ch	3Eh	0000h	Регистр захвата/сравнения 30
CC31	SFR	FE7Eh	3Fh	0000h	Регистр захвата/сравнения 31
T7	ESFR	F050h	28h	0000h	Регистр таймера T7
CC2_T78CON	SFR-b	FF20h	90h	0000h	Регистр управления таймерами 7 и 8
T7REL	ESFR	F054h	2Ah	0000h	Регистр загрузки таймера T7
T8	ESFR	F052h	29h	0000h	Регистр таймера T8
T8REL	ESFR	F056h	2Bh	0000h	Регистр загрузки таймера T8

Таблица А.147 – Регистр управления таймерами/счетчиками T0 и T1

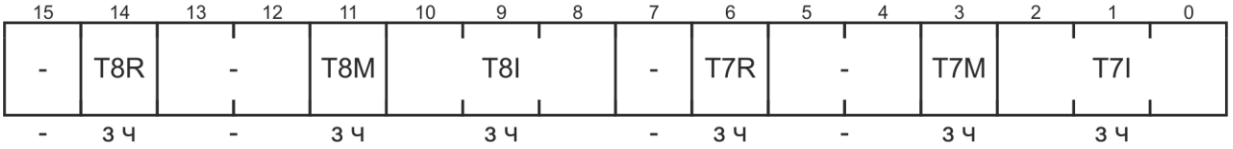
CC1_T01CON

SFR (FF50h / A8h) Сброс: 0000h



CC1_T78CON

SFR (FF20h / 90h) Сброс: 0000h



Поле	Бит	Тип	Описание
1	2	3	4
T1R (T8R)	14	Запись Чтение	Бит запуска T1(T8) 0 Выключен 1 Включен
T1M (T8M)	11	Запись Чтение	Бит управления режимом T1(T8) 0 Таймер 1 Счетчик
T1I (T8I)	10–8	Запись Чтение	В режиме таймера – поле задания коэффициента деления частоты fcc внешнего сигнала Fcc для получения желаемой частоты ft1 (ft8) сигнала тактирования Ft1 (Ft8) Ступенчатый режим*: $ft1 = \frac{fcc}{2(<T1I>+3)}$; $ft8 = \frac{fcc}{2(<T8I>+3)}$ Нормальный режим: $ft1 = \frac{fcc}{2<T1I>}$; $ft8 = \frac{fcc}{2<T8I>}$ В режиме счетчика в поле может быть записано только значение 000b (остальные зарезервированы). Источником сигнала тактирования T6OUF является таймер T6 блока таймеров GPT. Счетчик T1 (T8) инкрементируется каждый раз, когда переполняется таймер T6
T0R (T7R)	6	Запись Чтение	Бит запуска T0 (T7) 0 Выключен 1 Включен
T0M (T7M)	3	Запись Чтение	Бит управления режимом T0 (T7) 0 Таймер 1 Счетчик

Окончание таблицы А.147

<table>
<tr><th>1</th><th>2</th><th>3</th><th colspan="4">4</th></tr>
<tr><td rowspan="8">T0I
(T7I)</td><td rowspan="8">2–0</td><td rowspan="8">Запись
Чтение</td><td colspan="4">В режиме таймера – поле задания коэффициента деления частоты fcc внешнего сигнала Fcc для получения желаемой частоты ft0(ft7) сигнала тактирования Ft0(Ft7)</td></tr>
<tr><td>Ступенчатый режим*</td><td colspan="2">$ft0 = \frac{fcc}{2(<T1I>+3)}$</td><td>$ft7 = \frac{fcc}{2(<T7I>+3)}$</td></tr>
<tr><td>Нормальный режим</td><td colspan="2">$ft0 = \frac{fcc}{2<T1I>}$</td><td>$ft7 = \frac{fcc}{2<T7I>}$</td></tr>
<tr><td colspan="4">В режиме счетчика – поле указания источника, который формирует сигнал тактирования счетчика T0 (T7)</td></tr>
<tr><td>000</td><td colspan="3">Переполнение таймера T6 блока таймеров GPT</td></tr>
<tr><td>001</td><td colspan="3">Передний фронт сигнала на входе T0IN (T8IN)</td></tr>
<tr><td>010</td><td colspan="3">Задний фронт сигнала на входе T0IN (T8IN)</td></tr>
<tr><td>011</td><td colspan="3">Передний/задний фронт сигнала на входе T0IN (T8IN)</td></tr>
<tr><td>100 – 111</td><td colspan="3">Зарезервировано</td></tr>
<tr><td>–</td><td>15, 13,12, 7, 5, 4</td><td>–</td><td colspan="4">Зарезервировано</td></tr>
</table>

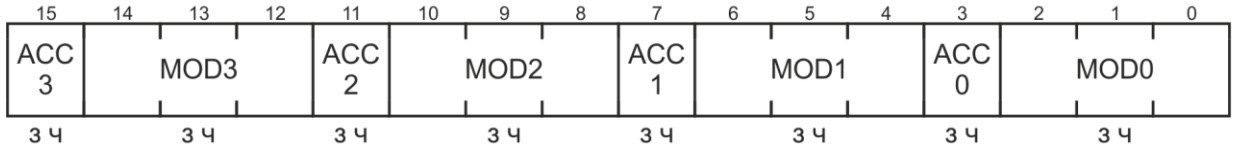
_____

* Ступенчатый и нормальный режимы работы подробно описаны ниже. Для таймеров разница между этими режимами заключается лишь в том, что в ступенчатом режиме частота сигнала тактирования таймера дополнительно делится на 8.

Таблица А.148 – Регистры управления режимами

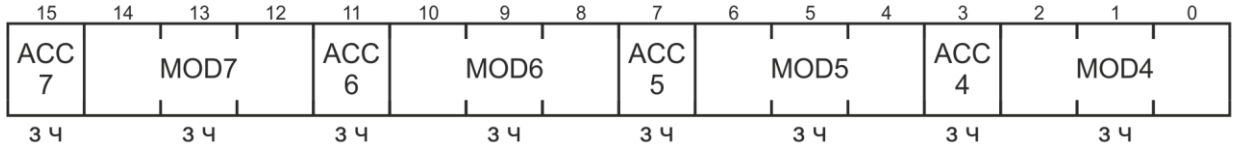
CC1_M0

SFR (FF52h / A9h) Сброс: 0000h



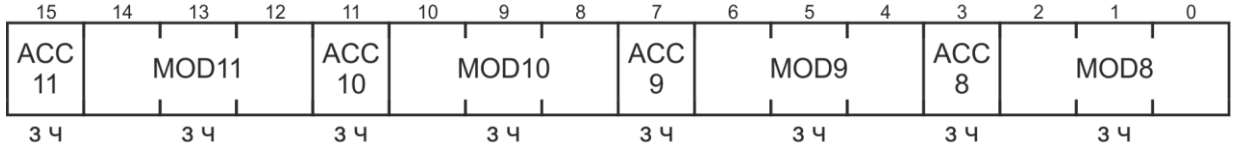
CC1_M1

SFR (FF54h / AAh) Сброс: 0000h



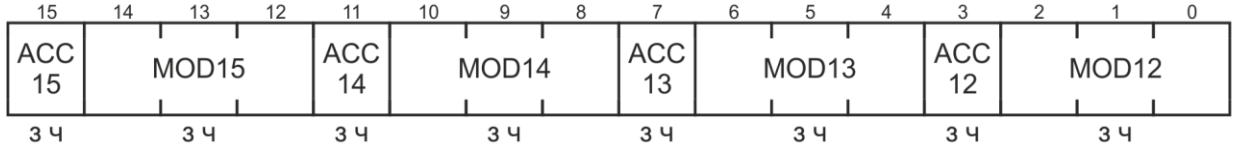
CC1_M2

SFR (FF56h / ABh) Сброс: 0000h



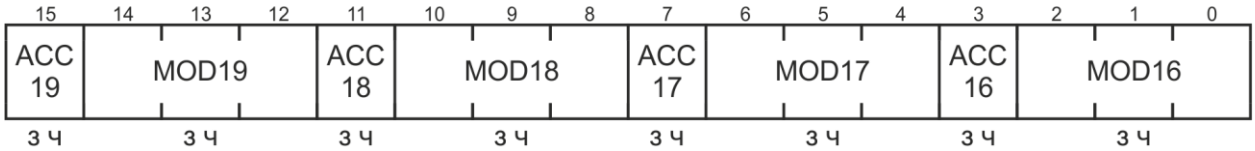
CC1_M3

SFR (FF58h / ACh) Сброс: 0000h



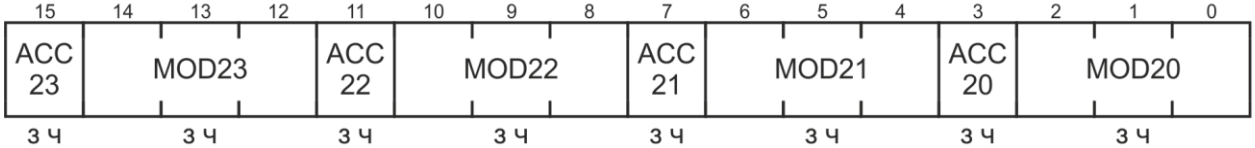
CC2_M4

SFR (FF22h / 91h) Сброс: 0000h



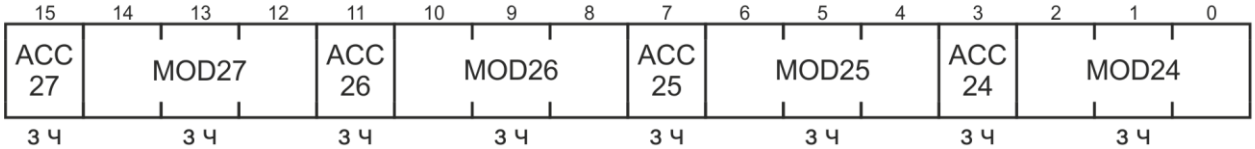
CC2_M5

SFR (FF24h / 92h) Сброс: 0000h



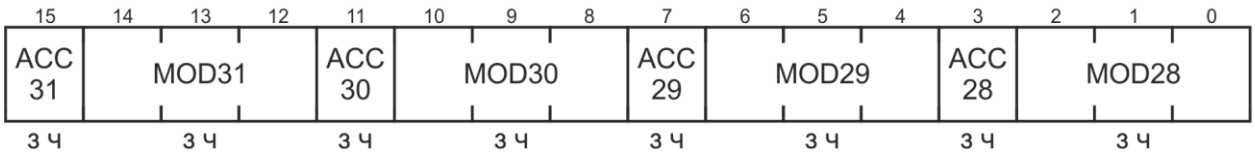
CC2_M6

SFR (FF26h / 93h) Сброс: 0000h



CC2_M7

SFR (FF28h / 94h) Сброс: 0000h



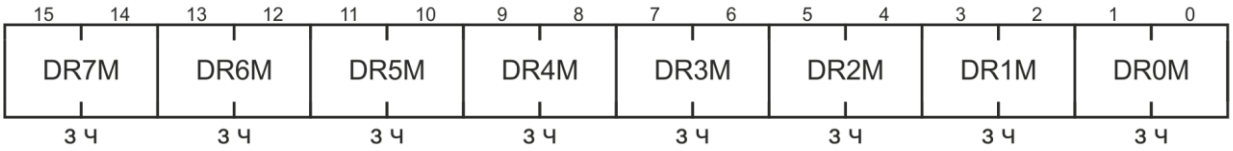
Окончание таблицы А.148

Поле	Бит	Тип	Описание	
ACCy	15, 11, 7, 3	Запись Чтение	Бит выбора таймера/счетчика для регистра CCy	
			0	Таймер/счетчик T0
			1	Таймер/счетчик T1
MODy	14 – 12, 10 – 8, 6 – 4, 2 – 0	Запись Чтение	Поле задания режима работы канала y	
			000	Регистр CCy используется как регистр общего назначения микроконтроллера
			001	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении переднего фронта сигнала на входе CCyIO и генерирование прерывания
			010	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении заднего фронта сигнала на входе CCyIO и генерирование прерывания
			011	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении переднего/заднего фронта сигнала на входе CCyIO и генерирование прерывания
			100	Режим сравнения 0. При совпадении значений регистра CCy и выбранного таймера генерируется прерывание, состояние выхода CCyIO не изменяется. Если регистр принадлежит Банку 2, может быть включен двухрегистровый режим
			101	Режим сравнения 1. При совпадении значений регистра CCy и выбранного таймера/счетчика генерируется прерывание, состояние выхода CCyIO инвертируется. Если регистр принадлежит Банку 1, может быть включен двухрегистровый режим
			110	Режим сравнения 2. При совпадении значений регистра CCy и выбранного таймера/счетчика генерируется прерывание, состояние выхода CCyIO не изменяется. За один период переполнения таймера допускается только одно совпадение – остальные игнорируются
			111	Режим сравнения 3. При совпадении значений регистра CCy и выбранного таймера/счетчика генерируется прерывание, на выходе CCyIO устанавливается логическая единица. За один период переполнения таймера допускается только одно совпадение – остальные игнорируются. При переполнении таймера/счетчика на выходе CCyIO устанавливается логический ноль

Таблица А.149 – Регистр двухрегистрового режима сравнения

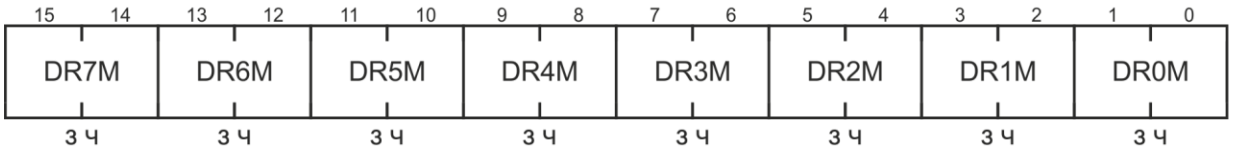
CC1_DRM

SFR (FF5Ah / ADh) Сброс: 0000h



CC2_DRM

SFR (FF2Ah / 95h) Сброс: 0000h

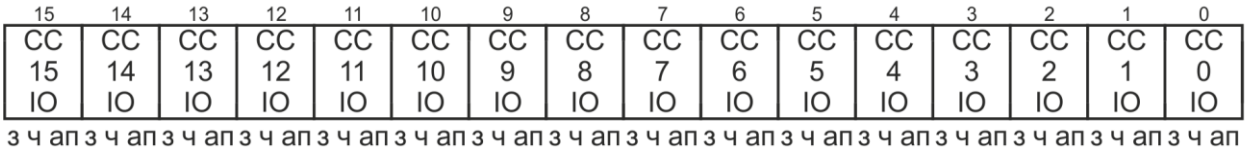


Поле	Бит	Тип	Описание	
DR7M, DR6M, DR5M, DR4M, DR3M, DR2M, DR1M, DR0M	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Запись Чтение	Поле управления двухрегистровым режимом	
			00	Режим включается автоматически при условии, что каждый из регистров пары запрограммирован на соответствующий режим сравнения: - младший регистр пары (Банк 1) на режим 1; - старший регистр пары (Банк 2) на режим 0
			01	Режим выключен
			10	Режим включен безусловно
			11	Зарезервировано

Таблица А.150 – Регистр выходных сигналов

CCx_OUT

Сброс: 0000h

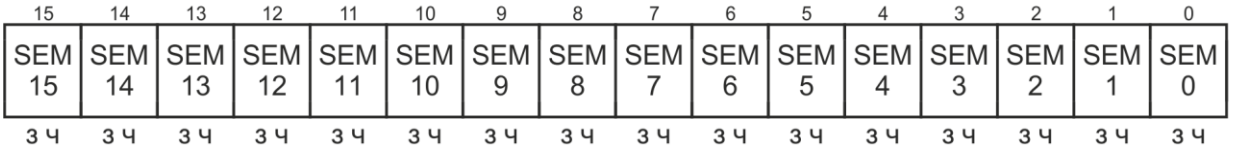


Поле	Бит	Тип	Описание
CC15IO, CC14IO, CC13IO, CC12IO, CC11IO, CC10IO, CC9IO, CC8IO, CC7IO, CC6IO, CC5IO, CC4IO, CC3IO, CC2IO, CC1IO, CC0IO	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Запись Чтение Аппаратное влияние	Выходная защелка канала

Таблица А.151 – Регистр режима однократного срабатывания

CCx_SEM

Сброс: 0000h

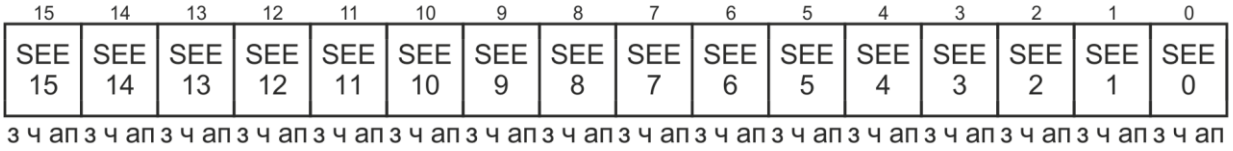


Поле	Бит	Тип	Описание	
SEM15, SEM14, SEM13, SEM12, SEM11, SEM10, SEM9, SEM8, SEM7, SEM6, SEM5, SEM4, SEM3, SEM2, SEM1, SEM0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Запись Чтение	Бит включения режима однократного срабатывания	
			0	Не включен
			1	Включен

Таблица А.152 – Регистр однократного срабатывания

CCx_SEE

Сброс: 0000h

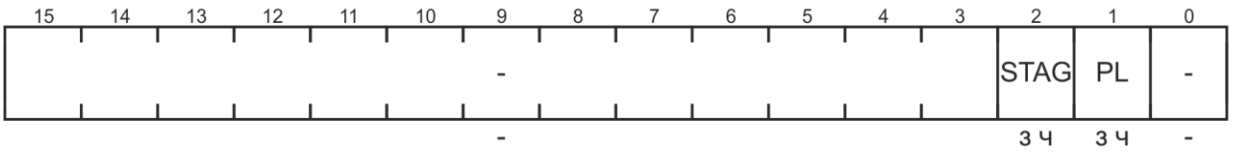


Поле	Бит	Тип	Описание	
SEE15, SEE14, SEE13, SEE12, SEE11, SEE10, SEE9, SEE8, SEE7, SEE6, SEE5, SEE4, SEE3, SEE2, SEE1, SEE0	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Запись Чтение	Флаг ожидания события	
			0	Ожидание запрограммированного события не запущено или событие уже произошло
			1	Ожидается событие. Примечание – Этот бит может быть установлен только, если установлен соответствующий ему бит в регистре CC1_SEM

Таблица А.153 – Регистры управления вводом-выводом

CCx_IOC

Сброс: 0000h

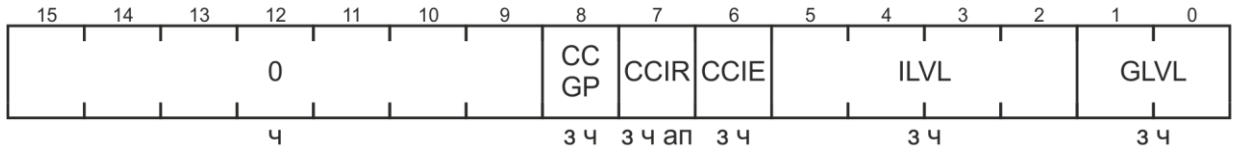


Поле	Бит	Тип	Описание	
STAG	2	Запись Чтение	Бит выключения ступенчатого режима	
			0	Ступенчатый режим
			1	Нормальный режим
PL	1	Запись Чтение	Бит управления подключением выходов модуля к выводам микроконтроллера	
			0	Все выходы подключены к соответствующим выводам
			1	Выходы отключены и не оказывают влияния на выводы
–	15–3, 0	–	Зарезервировано	

Таблица А.154 – Регистр управления прерыванием канала x

CCxIC

Сброс: 0000h



Поле	Бит	Тип	Описание	
CCGP	8	Запись Чтение	Расширение группового приоритета. Определяет значение старшего разряда уровня группы	
CCIR	7	Запись Чтение Аппаратное влияние	Флаг запроса прерывания	
			0	Нет запроса на прерывание
			1	Запрос на прерывание
CCIE	6	Запись Чтение	Бит разрешения прерывания	
			0	Запрещено
			1	Разрешено
ILVL	5–2	Запись Чтение	Поле задания уровня приоритета прерывания. Наивысший уровень приоритета задается значением Fh, низший уровень – значением 0h.	
GLVL	1–0	Запись Чтение	Поле задания уровня приоритета прерывания в группе. Определяет уровень приоритета прерывания в группе одновременно возникших прерываний с одинаковым приоритетом	
0	15–9	Чтение «0»	Зарезервировано	

A.15 Регистры блока CAPCOM6

Регистры блока CAPCOM6 представлены в таблицах А.155 – А.180.

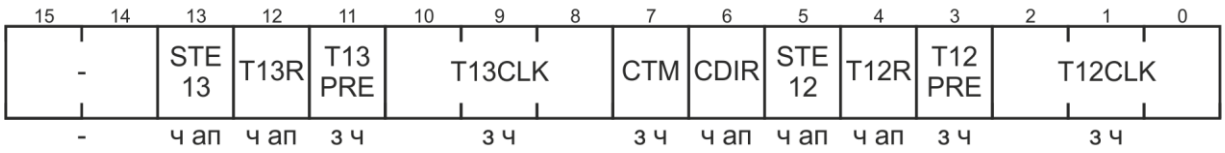
Таблица А.155 – Список регистров

Мнемоническое обозначение	Область памяти	Адрес	Сброс	Назначение
CCU6_T12	XSFR	E890h	0000h	Регистр счета таймера T12
CCU6_T12PR	XSFR	E892h	0000h	Регистр периода таймера T12
CCU6_T12DTC	XSFR	E894h	0000h	Регистр контроля задержки «dead-time» T12
CCU6_CC60R	XSFR	E898h	0000h	Регистр захвата/сравнения канала 0
CCU6_CC61R	XSFR	E89Ah	0000h	Регистр захвата/сравнения канала 1
CCU6_CC62R	XSFR	E89Ch	0000h	Регистр захвата/сравнения канала 2
CCU6_CC60SR	XSFR	E8A0h	0000h	Теневой регистр захвата/сравнения канала 0
CCU6_ CC61SR	XSFR	E8A2h	0000h	Теневой регистр захвата/сравнения канала 1
CCU6_ CC62SR	XSFR	E8A4h	0000h	Теневой регистр захвата/сравнения канала 2
CCU6_ TCTR4	XSFR	E8A6h	0000h	Регистр управления таймерами 4
CCU6_CMPSTAT	XSFR	E8A8h	0000h	Регистр состояния захвата/сравнения
CCU6_CMPMODIF	XSFR	E8AAh	0000h	Регистр изменения состояния захвата/сравнения
CCU6_TCTR0	XSFR	E8ACh	0000h	Регистр управления таймерами 0
CCU6_TCTR2	XSFR	E8AEh	0000h	Регистр управления таймерами 2
CCU6_T13	XSFR	E8B0h	0000h	Регистр счета таймера T13
CCU6_T13PR	XSFR	E8B2h	0000h	Регистр периода таймера T13
CCU6_CC63R	XSFR	E8B4h	0000h	Регистр захвата/сравнения канала 3
CCU6_CC63SR	XSFR	E8B6h	0000h	Теневой регистр захвата/сравнения 3
CCU6_MODCTR	XSFR	E8C0h	0000h	Регистр управления модуляцией
CCU6_TRPCTR	XSFR	E8C2h	0000h	Регистр управления ловушками
CCU6_PSLR	XSFR	E8C4h	0000h	Регистр уровня пассивного состояния
CCU6_T12MSEL	XSFR	E8C6h	0000h	Регистр выбора режима захвата/сравнения T12
CCU6_MCMOUTS	XSFR	E8CAh	0000h	Теневой регистр управления выходами в мультиканальном режиме
CCU6_MCMOUT	XSFR	E8CCh	0000h	Регистр управления выходами в мультиканальном режиме
CCU6_MCMCTR	XSFR	E8CEh	0000h	Регистр управления мультиканальным режимом
CCU6_IS	XSFR	E8D0h	0000h	Регистр состояния прерываний
CCU6_ISS	XSFR	E8D2h	0000h	Регистр установки состояния прерываний
CCU6_ISR	XSFR	E8D4h	0000h	Регистр сброса состояния прерываний
CCU6_INP	XSFR	E8D6h	3940h	Регистр указателя узла прерываний
CCU6_IEN	XSFR	E8D8h	0000h	Регистр разрешения прерываний

Таблица А.156 – Регистр управления таймерами

CCU6_TCTR0

XSFR (E8ACh) Сброс: 0000h



Поле	Бит	Тип	Описание		
1	2	3	4		
STE13 STE12	13 5	Чтение Аппаратное влияние	Бит программного запроса теневой загрузки регистра T13PR/T12PR. Для установки бита следует записать единицу в бит T13STR/T12STR регистра CCU6_TCTR4. Сброс бита происходит при записи единицы в бит T13STD/T12STD		
			0	Нет действий	
			1	Загрузить в регистр T13PR/T12PR число из соответствующего теневого регистра при следующем инкрементировании/ декрементировании таймера, если он включен. Если таймер выключен, загрузка происходит сразу. После выполнения теневой загрузки бит сбрасывается аппаратно	
T13R T12R	12 4	Чтение Аппаратное влияние	Бит запуска таймера T13/T12. Устанавливается и сбрасывается посредством битов T13RS/T12RS и T13RR/T12RR регистра CCU6_TCTR4. Аппаратно сбрасывается только в режиме однократного срабатывания		
			0	Таймер остановлен	
			1	Таймер запущен	
T13PRE T12PRE	11 3	Запись Чтение	Бит включения дополнительного делителя входной частоты		
			0	Делитель выключен	
			1	Делитель 1/256 включен	
T13CLK T12CLK	10–8, 2–0	Запись Чтение	Поле задания коэффициента деления частоты fosc внешнего сигнала для получения желаемой частоты ft13/ft12 сигнала тактирования таймера T13/T12		
			T12/13CLK	T12/13PRE = 0	T12/13PRE = 1
			000	1	1/256
			001	1/2	1/512
			010	1/4	1/1024
			011	1/8	1/2048
			100	1/16	1/4096
			101	1/32	1/8192
			110	1/64	1/16384
			111	1/128	1/32768

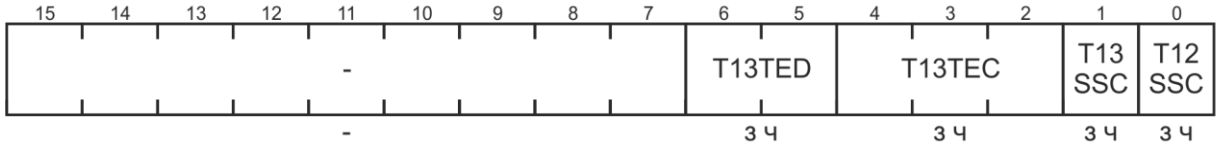
Окончание таблицы А.156

1	2	3	4	
CTM	7	Запись Чтение	Бит выбора режима счета таймера T12	
			0	Однонаправленный. Таймер считает только вверх
			1	Двунаправленный. Направление счета таймера может меняться в течение его работы
CDIR	6	Чтение Аппаратное влияние	Индикатор текущего направления счета таймера T12	
			0	Счет вверх
			1	Счет вниз
–	15, 14	–	Зарезервировано	

Таблица А.157 – Регистр управления таймерами

CCU6_TCTR2

XSFR (E8AEh) Сброс: 0000h

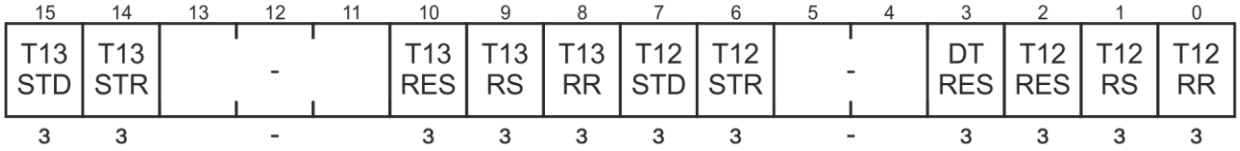


Поле	Бит	Тип	Описание	
T13TED	6, 5	Запись Чтение	Выбор направления счета таймера T13. Поле определяет, при каком направлении счета («вверх»/«вниз») таймера T12 детектировать ожидаемое событие	
			00	Зарезервировано
			01	Вверх
			10	Вниз
			11	Постоянно
T13TEC	4–2	Запись Чтение	Выбор события, связанного с таймером T12. Поле определяет ожидаемое событие, связанное с таймером T12, при обнаружении которого будет запущен таймер T13	
			000	Нет действий
			001	Совпадение Событие по каналу 0
			010	Совпадение Событие по каналу 1
			011	Совпадение Событие по каналу 2
			100	Совпадение Событие по любому из трех каналов
			101	Событие «Период»
			110	Совпадение Событие «Ноль» при счете вверх
			111	Любое изменение состояния датчиков Холла
T13SSC T12SSC	1 0	Запись Чтение	Бит включения режима однотактного запуска таймера T13/T12	
			0	Выключен
			1	Включен
–	15–7	–	Зарезервировано	

Таблица А.158 – Регистр управления таймерами

CCU6_TCTR4

XSFR (E8A6h) Сброс: 0000h



Поле	Бит	Тип	Описание	
1	2	3	4	
T13STD T12STD	15 7	Запись	Бит управления аппаратной теневой загрузкой регистра T13PR/T12PR	
			0	Для таймера T13. Теневая загрузка происходит аппаратно каждый раз, когда для таймера возникает событие «Период»
				Для таймера T12. Теневая загрузка происходит аппаратно каждый раз, когда для таймера возникает событие: 1 «Период» при счете вверх; 2 «Единица» при счете вниз
			1	Все аппаратные теневые загрузки, которые выполнялись при T13STD/T12STD = 0, запрещены. В тоже время программные теневые загрузки, инициируемые установкой бита T13STR/T12STR, выполняются. Установка бита T13STD/T12STD всегда вызывает запись нуля в бит T13STR/T12STR, что приводит к обнулению бита STE13/STE12, т.е. отмену ожидания теневой загрузки. Если дальнейшее запрещение всех аппаратных теневых загрузок не требуется, бит T13STD/T12STD нужно программно сбросить
T13STR T12STR	14 6	Запись	Бит создания запроса теневой загрузки регистра T13PR/T12PR. Бит может быть установлен программно, независимо от состояния бита T13STD/T12STD. Бит сбрасывается программно, либо при установке бита T13STD/T12STD	
			0	Нет действий
			1	Запись единицы вызывает установку бита STE13/STE12 в регистре CCU6_TCTR0
T13RES T12RES	10 2	Запись	Бит сброса таймера T13/T12. Установка бита возможна в любой момент времени. Бит не оказывает влияния на работу таймера, т.е. не останавливает его и не переключает направление счета. Сбрасывается аппаратно	
			0	Нет действий
			1	Содержимое таймера обнуляется
T13RS T12RS	9 1	Запись	Бит создания запроса запуска таймера T13/T12. Сбрасывается аппаратно после запуска таймера	
			0	Нет действий
			1	Запись единицы вызывает аппаратную установку бита T13R/T12R в регистре CCU6_TCTR0

Окончание таблицы А.158

1	2	3	4	
T13RR T12RR	8 0	Запись	Бит создания запроса остановки таймера T13/T12. Сбрасывается аппаратно после остановки таймера	
			0	Нет действий
			1	Запись единицы вызывает аппаратный сброс бита T13R/T12R в регистре CCU6_TCTR0
DTRES	3	Запись	Бит сброса и остановки счетчиков задержки. Сбрасывается аппаратно	
			0	Нет действий
			1	Счетчики задержки всех каналов обнуляются и останавливаются
–	13–11, 5, 4	–	Зарезервировано	
Примечание – Битовые пары T13RS–T13RR, T12RS–T12RR не допускают состояния «11», поэтому попытка записи значения 11b в любое из указанных полей игнорируется. Запись значения 00b также игнорируется.				

Таблица А.159 – Регистры блока таймера T12

Название регистра	Адрес	Назначение
1	2	3
CCU6_T12	E890h	Регистр таймера T12. 16-разрядный счетчик. Направление счета зависит от заданного режима работы. Регистр всегда доступен для чтения. Запись в регистр возможна только при остановленном таймере
CCU6_T12PR	E892h	16-разрядный регистр периода таймера T12. Хранит значение периода, досчитав до которого, таймер меняет направление счета или обнуляется, в зависимости от режима работы. Всегда доступен для чтения. Недоступен напрямую для записи. Имеет теневой регистр T12PRS. В результате записи в регистр периода, записываемое значение предварительно размещается в теневом регистре и затем аппаратно копируется в регистр периода только по сигналу теневой загрузки. Этот механизм используется для синхронизации обновления регистра периода и работы таймера
CCU6_CC60R CCU6_CC61R CCU6_CC62R	E898h E89Ah E89Ch	16-разрядный регистр захвата/сравнения блока таймера T12. В режиме сравнения хранит значение, сравниваемое со значением таймера. В режиме захвата захватывает текущее значение таймера при возникновении запрограммированного условия. Всегда доступен для чтения. Недоступен напрямую для записи. Имеет теневой регистр CCU6_CCxSR. В результате записи в регистр захвата/сравнения, записываемое значение предварительно размещается в теневом регистре и затем аппаратно копируется в регистр периода только по сигналу теневой загрузки. Этот механизм используется для синхронизации обновления регистра и работы таймера

Окончание таблицы А.159

1	2	3
CCU6_CC60SR CCU6_CC61SR CCU6_CC62SR	E8A0h E8A2h E8A4h	16-разрядный теневой регистр захвата блока таймера T12. Хранит значение, которое должно быть записано в регистр захвата/сравнения CCU6_CC6xR по сигналу теневой загрузки. При работе основного регистра в режиме захвата, выполняет роль дополнительного регистра захвата. Захватывает текущее значение таймера при возникновении запрограммированного условия. Всегда доступен для чтения и записи (по адресу основного регистра)

Таблица А.160 – Регистр состояния входов и выходов каналов

CCU6_CMPSTAT

XSFR (E8A8h) Сброс: 0000h

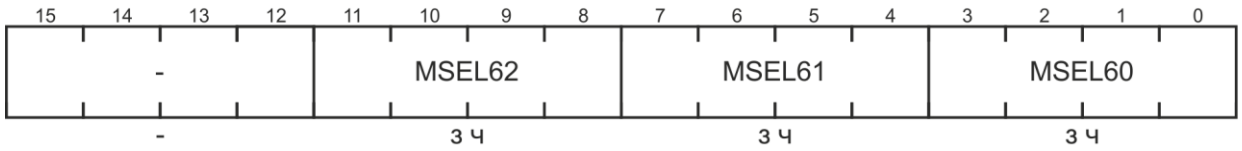
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13IM	COUT63PS	COUT62PS	CC62PS	COUT61PS	CC61PS	COUT60PS	CC60PS	-	CC63ST	CCPOS2	CCPOS1	CCPOS0	CC62ST	CC61ST	CC60ST
з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	-	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап	з ч ап

Поле	Бит	Тип	Описание	
T13IM	15	Запись Чтение Аппаратное влияние	Бит выбора сигнала для дальнейшей модуляции в канале 3	
			0	CC63ST
			1	CC63ST#
COUT63PS COUT62PS COUT61PS COUT60PS	14 13 11 9	Запись Чтение Аппаратное влияние	Бит выбора сигнала для линии CC6x_O. Имеет теневой бит и обновляется в момент теневой загрузки регистров CCU6_CC6xR	
			0	CC6xST & DTRx#
			1	CC6xST# & DTRx#
CC62PS CC61PS CC60PS	12 10 8	Запись Чтение Аппаратное влияние	Бит выбора сигнала для линии COUT6x_O. Имеет теневой бит и обновляется в момент теневой загрузки регистров CCU6_CC6xR	
			0	CC6xST & DTRx#
			1	CC6xST# & DTRx#
CC63ST CC62ST CC61ST CC60ST	6 2 1 0	Запись Чтение Аппаратное влияние	Бит состояния линии канала x	
			В режиме сравнения	
			0	Значение таймера меньше значения регистра CCU6_CC6xR
			1	Значение таймера больше значения регистра CCU6_CC6xR
			В режиме захвата (только каналы 0, 1 и 2)	
			0	Ожидаемый фронт сигнала еще не обнаружен
			1	Ожидаемый фронт сигнала обнаружен
CCPOS2 CCPOS1 CCPOS0	5 4 3	Запись Чтение Аппаратное влияние	Бит состояния входа, к которому подключается датчик Холла. Доступен только в режиме работы с датчиками Холла	
–	7	–	Зарезервировано	

Таблица А.161 – Регистр выбора режима захвата/сравнения T12

CCU6_T12MSEL

XSFR (E8C6h) Сброс: 0000h



Поле	Бит	Тип	Описание	
MSEL62 MSEL61 MSEL60	11–8 7–4 3–0	Запись Чтение	Поле выбора режима работы канала x. Каждый канал может программироваться независимо	
			Режимы сравнения	
			0000	Линии CC6x_O и COUT6x_O закрыты. Выводы CC6x и COUT6x микроконтроллера могут использоваться как выводы общего назначения
			0001	Вывод сигнала на линию CC6x_O. Линия COUT6x_O закрыта. Вывод COUT6x микроконтроллера может использоваться как вывод общего назначения
			0010	Вывод сигнала на линию COUT6x_O. Линия CC6x_O закрыта. Вывод CC6x микроконтроллера может использоваться как вывод общего назначения
			0011	Линии CC6x_O и COUT6x_O открыты
			Двухрегистровые и мультивходовые режимы захвата	
			0100	Режим 1
			0101	Режим 2
			0110	Режим 3
			0111	Режим 4
			1010	Режим 5
			1011	Режим 6
			1100	Режим 7
			1101	Режим 8
			1110	Режим 9
			Специальные режимы	
			1000	Режим работы с датчиками Холла
			1001	Режим блокирования
			1111	Зарезервировано
–	15–12	–	Зарезервировано	

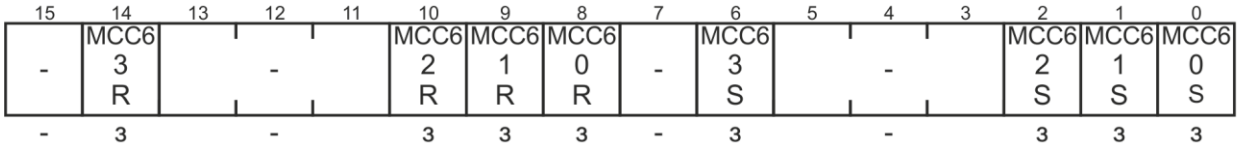
Таблица А.162 – Обзор режимов захвата/сравнения

MSEL6x	Выбранный режим				
0000B	Вывод сигнала сравнения запрещен. Выходы используются для простого ввода-вывода				
0001B	Вывод сигнала сравнения на вывод CC6x. Вывод COUT6x используется для простого ввода-вывода				
0010B	Вывод сигнала сравнения на вывод COUT6x. Вывод CC6x используется для простого ввода-вывода				
0011B	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
0100B	1	CC6x	0 – 1	–	CC6xR
		CC6x	1 – 0	–	CC6xSR
0101B	2	CC6x	0 – 1	CC6xR	CC6xSR
0110B	3	CC6x	1 – 0	CC6xR	CC6xSR
0111B	4	CC6x	любой	CC6xR	CC6xSR
1000B	Режим датчиков Холла				
1001B	Режим блокирования				
	Режим	Вывод	Активный перепад	Содержимое таймера T12 переносится в регистр	
1010B	5	CC6x	0 – 1	CC6xR	
		CCPOSx	1 – 0	CC6xSR	
1011B	6	CC6x	1 – 0	CC6xR	
		CCPOSx	0 – 1	CC6xSR	
1100B	7	CC6x	0 – 1	CC6xR	
		CCPOSx	0 – 1	CC6xSR	
1101B	8	CC6x	1 – 0	CC6xR	
		CCPOSx	1 – 0	CC6xSR	
1110B	9	CC6x	любой	CC6xR	
		CCPOSx	любой	CC6xSR	
1111B	–	Зарезервировано (никаких действий захвата/сравнения не производится).			
Примечание – Режим датчиков Холла должен программироваться одновременно для трех каналов таймера T12.					

Таблица А.163 – Регистр управления битами состояния

CCU6_CMPMODIF

XSFR (E8AAh) Сброс: 0000h



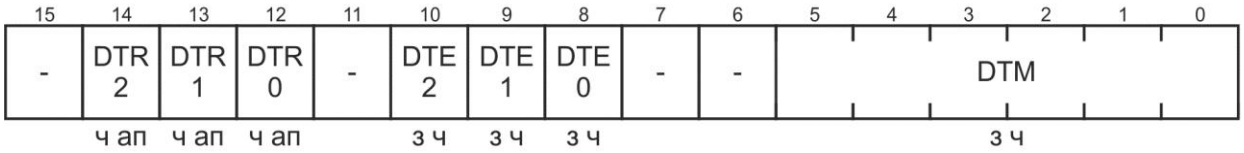
Поле	Бит	Тип	Описание	
MCC63R MCC62R MCC61R MCC60R	14 10 9 8	Запись	Бит программного сброса бита состояния CC6xST	
			0	Нет действий
			1	Сбросить
MCC63S MCC62S MCC61S MCC60S	6 2 1 0	Запись	Бит программной установки бита состояния CC6xST	
			0	Нет действий
			1	Установить
–	15, 13–11, 7, 5–3	–	Зарезервировано	

Примечание – Все биты сбрасываются аппаратно после того, как будет установлен/сброшен бит CC6xST. Одновременная установка какой-либо пары битов сброса и установки проинвертирует управляемый ими бит состояния.

Таблица А.164 – Регистр управления задержкой

CCU6_T12DTC

XSFR (E894h/-- h) Сброс: 0000h



Поле	Бит	Тип	Описание	
DTR2 DTR1 DTR0	14 13 12	Запись Чтение	Флаг задержки	
			0	Счетчик задержки остановлен
			1	Счетчик задержки работает
DTE2 DTE1 DTE0	10 9 8	Запись Чтение	Бит разрешения генерирования задержки	
			0	Запрещено
			1	Разрешено
DTM	5–0		Битовое поле программируемого значения задержки. Определяет программируемую задержку «dead-time» между переключением канала из пассивного состояния в активное (определяет значение, с которого начинает счет счетчик задержки)	
–	15, 11, 7, 6	–	Зарезервировано	

Примечание – Все счетчики задержки можно одновременно сбросить битом DTRES регистра CCU6_TCTR4.

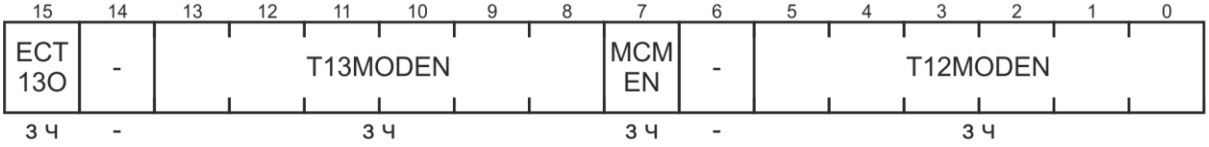
Таблица А.165 – Регистры блока таймера T13

Название регистра	Адрес	Назначение
CCU6_T13	E8B0h	Регистр таймера T13. 16-разрядный счетчик. Направление счета зависит от заданного режима работы. Регистр всегда доступен для чтения. Запись в регистр возможна только при остановленном таймере
CCU6_T13PR	E8B2h	16-разрядный регистр периода таймера T13. Хранит значение периода, досчитав до которого, таймер меняет направление счета или обнуляется, в зависимости от режима работы. Всегда доступен для чтения. Недоступен напрямую для записи. Имеет теневой регистр T13PRS. В результате записи в регистр периода, записываемое значение предварительно размещается в теневом регистре и затем аппаратно копируется в регистр периода только по сигналу теневой загрузки. Этот механизм используется для синхронизации обновления регистра периода и работы таймера
CCU6_CC63R	E8B4h	16-разрядный регистр захвата/сравнения блока таймера T13. В режиме сравнения хранит значение, сравниваемое со значением таймера. В режиме захвата захватывает текущее значение таймера при возникновении запрограммированного условия. Всегда доступен для чтения. Недоступен напрямую для записи. Имеет теневой регистр CCU6_CCxSR. В результате записи в регистр захвата/сравнения, записываемое значение предварительно размещается в теневом регистре и затем аппаратно копируется в регистр периода только по сигналу теневой загрузки. Этот механизм используется для синхронизации обновления регистра и работы таймера
CCU6_CC63SR	E8B6h	16-разрядный теневой регистр захвата блока таймера T13. Хранит значение, которое должно быть записано в регистр захвата/сравнения CCU6_CC6xR по сигналу теневой загрузки. При работе основного регистра в режиме захвата выполняет роль дополнительного регистра захвата. Захватывает текущее значение таймера при возникновении запрограммированного условия. Всегда доступен для чтения и записи (по адресу основного регистра)

Таблица А.166 – Регистр управления модуляцией

CCU6_MODCTR

XSFR (E8C0h) Сброс: 0000h

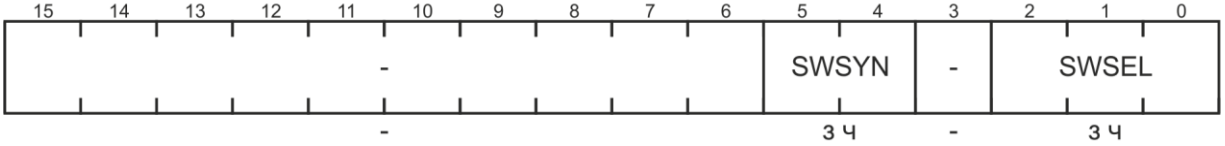


Поле	Биты	Тип	Описание
ECT13O	15	Чтение Запись	Разрешение вывода сигнала канала таймера T13: 0 Вывод сигнала запрещен (на выходе поддерживается низкий уровень сигнала) 1 Вывод сигнала разрешен
T13MODEN T12MODEN	13–8 5–0	Чтение Запись	Выбор выходных сигналов, каналов таймера T12. Биты управляют выбором 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
–	6, 14	–	Зарезервировано

Таблица А.167 – Регистр управления мультиканальным режимом

CCU6_MCMCTR

XSFR (E8CEh) Сброс: 0000h



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
–	15–6, 3	–	Зарезервировано

Таблица А.168 – Выбор события запуска теневой передачи в мультиканальном режиме

SWSEL	Выбранное событие
000b	Событие не выбрано
001b	Корректный сигнал датчика Холла CM_CHE
010b	Совпадение по периоду таймера T13 (T13_PM)
011b	Совпадение счетчика таймера T12 с единицей (T12_OM) при счете «вниз»
100b	Совпадение по значению для канала 1 при счете таймера T12 «вверх» (CM_61_UP)
101b	Совпадение по периоду таймера T12 (T12_PM) при счете «вверх»
11xb	Зарезервировано

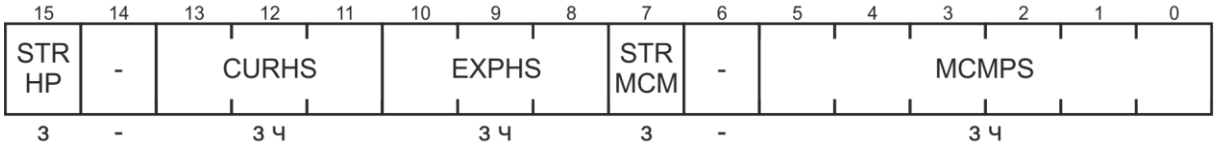
Таблица А.169 – Выбор источника синхронизации

SWSEL	Источник синхронизации
00b	Нет синхронизации
01b	Синхронизация по сигналу совпадения с нулем таймера T13 (T13_ZM)
10b	Синхронизация по сигналу совпадения с нулем таймера T12 (T12_ZM)
11b	Зарезервировано

Таблица А.170 – Теневой регистр управления выходами в мультиканальном режиме

CCU6_MCMOUTS

XSFR (E8CAh) Сброс: 0000h



Поле	Биты	Тип	Описание
1	2	3	4
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

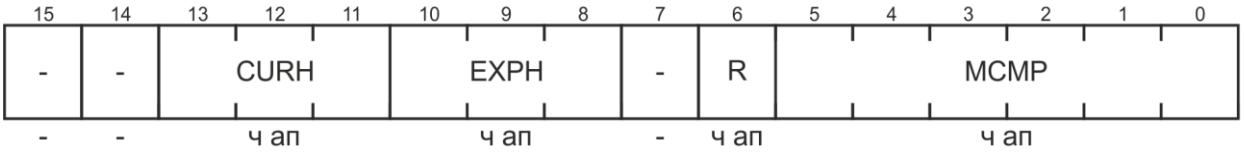
Окончание таблицы А.170

1	2	3	4
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
–	14, 6	–	Зарезервировано

Таблица А.171 – Регистр управления выходами в мультиканальном режиме

CCU6_MCMOUT

XSFR (E8CCh) Сброс: 0000h



Поле	Биты	Тип	Описание
1	2	3	4
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. По окончании передачи аппаратно сбрасывается. Также бит R равен «0» в случае, когда MCMEN = 0B. 0 Теневая передача не ожидается. 1 Теневая передача ожидается и еще не произошла

Окончание таблицы А.171

1	2	3	4
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
–	15–14, 7	–	Зарезервировано

_____

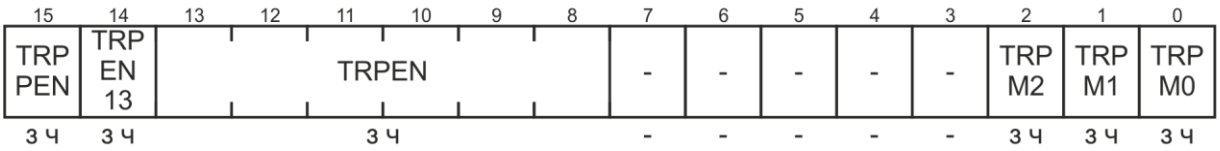
[1)] Биты в полях EXPH и CURH соответствуют сигналам с датчиков Холла на входах CCPOSx, x = 0, 1, 2. EXPH.2, EXPH.1, EXPH.0 и CURH.2, CURH.1, CURH.0 соответствуют побитно слева направо CCPOS2, CCPOS1, CCPOS0.

[2)] Поле очищается в то время, как бит IS.IDLE = 1B.

Таблица А.172 – Регистр управления ловушками

CCU6_TRPCTR

XSFR (E8C2h) Сброс: 0000h



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»

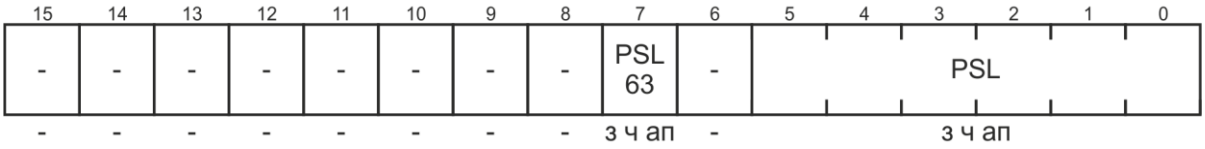
Окончание таблицы А.172

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# становится 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
–	7–3	–	Зарезервировано

Таблица А.173 – Регистр уровня пассивного состояния

CCU6_PSLR

XSFR (E8C4h) Сброс: 0000h

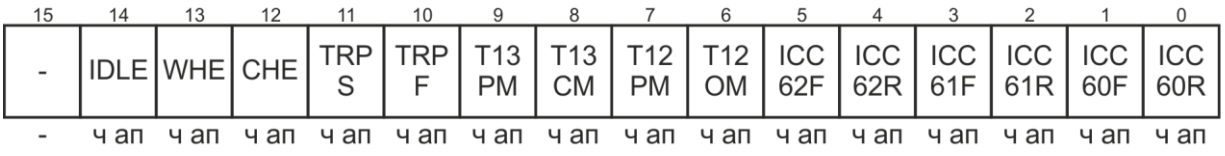


Поле	Биты	Тип	Описание
PSL63	7	Чтение Запись Аппаратное влияние	Бит уровня выходного сигнала канала таймера T13. Бит определяет, в каком виде будет передан на выход модулируемый сигнал. 0 Модулируемый сигнал передается в прямом виде. 1 Модулируемый сигнал передается в инвертированном виде
PSL	5–0	Чтение 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
–	15–8, 6	–	Зарезервировано

Таблица А.174 – Регистр состояния прерываний

CCU6_IS

XSFR (E8D0h) Сброс: 0000h



Поле	Биты	Тип	Описание
1	2	3	4
IDLE	14	Чтение Аппаратное влияние	Флаг режима ожидания Idle. Если разрешено ENIDLE = 1B, этот флаг устанавливается совместно WHE 0 Бездействие. 1 Поле MCMP очищается, на выходах каналов таймер T12 поддерживается низкий уровень сигнала

Продолжение таблицы А.174

1	2	3	4
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# = 0B) или программное попадание в ловушку
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

Окончание таблицы А.174

1	2	3	4
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

Таблица А.175 – Регистр установки состояния прерываний

CCU6_ISS

XSFR (E8D2h) Сброс: 0000h

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

Поле	Биты	Тип	Описание
1	2	3	4
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
–	15, 11	–	Зарезервировано

Таблица А.176 – Регистр сброса состояния прерываний

CCU6_ISR

XSFR (E8D4h /--) Сброс: 0000h

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

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 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.

Таблица А.177 – Регистр разрешения прерываний

CCU6_IEN

XSFR (E8D8h) Сброс: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	EN IDLE	EN WHE	EN CHE	-	EN TRPF	EN T13PM	EN T13CM	EN T12PM	EN T12OM	EN CC62F	EN CC62R	EN CC61F	EN CC61R	EN CC60F	EN CC60R
-	3	3	3	-	3	3	3	3	3	3	3	3	3	3	3

Поле	Биты	Тип	Описание
1	2	3	4
ENIDLE	14	Запись	Разрешение установки флага режима ожидания Idle
ENWHE	13	Запись	Разрешение формирования запроса на прерывания при появлении сигнала некорректного сигнала датчика Холла

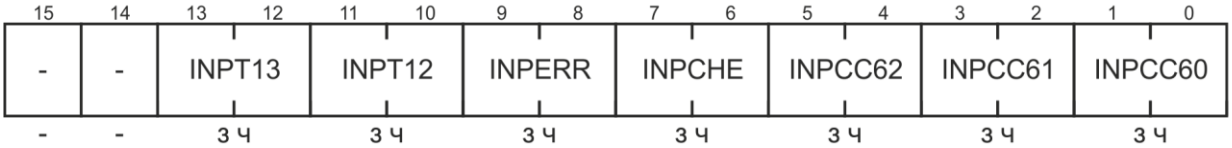
Окончание таблицы А.177

1	2	3	4
ENCHE	12	Запись	Разрешение формирования запроса на прерывания при появлении сигнала корректного сигнала датчика Холла
ENTRPF	10	Запись	Разрешение формирования запроса на прерывания при появлении сигнала перехода в состояние ловушки
ENT13PM	9	Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду счетчика таймера T13
ENT13CM	8	Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по значению счетчика таймера T13
ENT12PM	7	Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду при счете «вверх» счетчика таймера T12
ENT12OM	6	Запись	Разрешение формирования запроса на прерывания при появлении сигнала совпадения с единицей при счете «вниз» счетчика таймера T12
ENCC62F ENCC61F ENCC60F	5 3 1	Запись	Разрешение формирования запроса на прерывания при появлении сигнала события 1
ENCC62R ENCC61R ENCC60R	4 2 0	Запись	Разрешение формирования запроса на прерывания при появлении сигнала события 2
–	11, 15	–	Зарезервировано

Таблица А.178 – Регистр указателя узла прерываний

CCU6_INP

XSFR (E8D6h) Сброс: 0000h



Поле	Биты	Тип	Описание
1	2	3	4
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

Окончание таблицы А.178

1	2	3	4
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
–	15, 14	–	Зарезервировано

Таблица А.179 – Кодировка для полей регистра INP

Комбинация сигналов в поле INPxx	Выбранный выход
00b	I0
01b	I1
10b	I2
11b	I3

Таблица А.180 – 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	F140h
Прерывания канал 1	I0		
Прерывания канал 2	I0		
Прерывания по корректному сигналу датчика	I1	CCU6_EIC	F188h
Прерывания при возникновении ошибок	I1		
Прерывания таймера T12	I2	CCU6_T12IC	F190h
Прерывания таймера T13	I3	CCU6_T13IC	F198h

Приложение Б
(обязательное)
Таблицы регистров областей SFR, ESFR и XSFR по адресам

Регистры областей SFR, ESFR по адресам представлены в таблице Б.1, XSFR – в таблице Б.2.

Таблица Б.1

Адрес		Мнемоника	Область памяти	Сброс	Описание
1	2	3	4	5	6
F000h	00h	QX0	ESFR	0000h	Регистр смещения 0 для указателя адреса IDX0/1 (MAC)
F002h	01h	QX1	ESFR	0000h	Регистр смещения 1 для указателя адреса IDX0/1 (MAC)
F004h	02h	QR0	ESFR	0000h	Регистр смещения 0 для указателя адреса GPRi (MAC)
F006h	03h	QR1	ESFR	0000h	Регистр смещения 1 для указателя адреса GPRi (MAC)
F008h, F00Ah	04h, 05h	–	ESFR	–	Зарезервировано
F00Ch	06h	CPUID	ESFR	0420h	Регистр идентификации ЦПУ
F00Eh–F012h	07h–09h	–	ESFR	–	Зарезервировано
F014h	0Ah	XADRS1	ESFR	XXXXh	Регистр 1 выбора адреса (XBUS)
F016h	0Bh	XADRS2	ESFR	XXXXh	Регистр 2 выбора адреса (XBUS)
F018h	0Ch	XADRS3	ESFR	XXXXh	Регистр 3 выбора адреса (XBUS)
F01Ah	0Dh	XADRS4	ESFR	XXXXh	Регистр 4 выбора адреса (XBUS)
F01Ch	0Eh	XADRS5	ESFR	XXXXh	Регистр 5 выбора адреса (XBUS)
F01Eh	0Fh	XADRS6	ESFR	XXXXh	Регистр 6 выбора адреса (XBUS)
F020h, F022h	10h, 11h	–	ESFR	–	Зарезервировано
F024h	12h	XPERCON	ESFR	0000h	Регистр управления X-периферией (XBUS)
F026h–F02Eh	13h–17h	–	ESFR	–	Зарезервировано
F030h	18h	RTCCST	ESFR	--00h	Регистр управления и статуса (RTC)
F032h	19h	RTUDST	ESFR	--00h	Регистр состояния обновлений (RTC)
F034h	1Ah	RTCEIST	ESFR	--00h	Регистр состояния прерываний (RTC)
F036h	1Bh	RTCIEN	ESFR	--00h	Регистр разрешения прерываний (RTC)
F038h	1Ch	RTPRD	ESFR	0000h	Регистр прескалера (RTC)
F03Ah	1Dh	RTCRD_H	ESFR	0000h	Регистр реального времени, старшие разряды (RTC)

Продолжение таблицы Б.1

1	2	3	4	5	6
F03Ch	1Eh	RTCRD_L	ESFR	0001h	Регистр реального времени, младшие разряды (RTC)
F03Eh	1Fh	RTCLD_H	ESFR	0000h	Перезагружаемый регистр, старшие разряды (RTC)
F040h	20h	RTCLD_L	ESFR	0000h	Перезагружаемый регистр, младшие разряды (RTC)
F042h	21h	RTCCMP1_H	ESFR	0000h	Регистр сравнения 1, старшие разряды (RTC)
F044h	22h	RTCCMP1_L	ESFR	0200h	Регистр сравнения 1, младшие разряды (RTC)
F046h	23h	RTCCMP2_H	ESFR	FFFFh	Регистр сравнения 2, старшие разряды (RTC)
F048h	24h	RTCCMP2_L	ESFR	FFFFh	Регистр сравнения 2, младшие разряды (RTC)
F04Ah	25h	RTCCMP3_H	ESFR	FFFFh	Регистр сравнения 3, старшие разряды (RTC)
F04Ch	26h	RTCCMP3_L	ESFR	FFFFh	Регистр сравнения 3, младшие разряды (RTC)
F04Eh	27h	–	ESFR	–	Зарезервировано
F050h	28h	T7	ESFR	0000h	Регистр таймера T7 (CAPCOM2)
F052h	29h	T8	ESFR	0000h	Регистр таймера T8 (CAPCOM2)
F054h	2Ah	T7REL	ESFR	0000h	Регистр загрузки таймера T7 (CAPCOM2)
F056h	2Bh	T8REL	ESFR	0000h	Регистр загрузки таймера T8 (CAPCOM2)
F058h	2Ch	–	ESFR	–	Зарезервировано
F05Ah	2Dh	SSC1TB	ESFR	0000h	Буферный регистр передатчика (SSC1)
F05Ch	2Eh	SSC1RB	ESFR	0000h	Буферный регистр приемника (SSC1)
F05Eh	2Fh	SSC1BR	ESFR	0000h	Регистр скорости пересылки (SSC1)
F060h	30h	–	ESFR	–	Зарезервировано
F062h	31h	CC1_IOC	ESFR	0000h	Регистр управления вводом/выводом (CAPCOM1)
F064h	32h	–	ESFR	–	Зарезервировано
F066h	33h	CC2_IOC	ESFR	0000h	Регистр управления вводом/выводом (CAPCOM2)
F068h	34h	COMDATA	ESFR	0000h	Регистр режима коммуникации Cerberus (OCDS)
F06Ah	35h	RWDATA	ESFR	0000h	Регистр данных режима RW Cerberus (OCDS)
F06Ch	36h	IOSR	ESFR	0000h	Регистр статуса Cerberus (OCDS)
F06Eh–F0AEh	37h–57h	–	ESFR	37h	Зарезервировано

Продолжение таблицы Б.1

1	2	3	4	5	6
F0B0h	58h	SSC0TB	ESFR	0000h	Буферный регистр передатчика (SSC0)
F0B2h	59h	SSC0RB	ESFR	0000h	Буферный регистр приемника (SSC0)
F0B4h	5Ah	SSC0BR	ESFR	0000h	Регистр скорости пересылки (SSC0)
F0B6h–F0D6h	5Bh–6Bh	–	ESFR	–	Зарезервировано
F0D8h	6Ch	DTIDR	ESFR	0000h	Регистр идентификации задачи системы отладки (OCDS)
F0DAh	6Dh	SMBSDA	ESFR	--XXh	Сдвиговый регистр данных (I2C)
F0DCh	6Eh	SMBST	ESFR	--00h	Регистр статуса (I2C)
F0DEh	6Fh	SMBCST	ESFR	--00h	Регистр управления и статуса (I2C)
F0E0h	70h	SMBCTRL1	ESFR	--00h	Регистр управления 1 (I2C)
F0E2h	71h	SMBADDR	ESFR	--00h	Регистр собственного адреса (I2C)
F0E4h	72h	SMBCTRL2	ESFR	--00h	Регистр управления 2 (I2C)
F0E6h	73h	SMBTOPR	ESFR	--00h	Регистр делителя частоты времени ожидания (I2C)
F0E8h	74h	SMBCTRL3	ESFR	--00h	Регистр управления 3 (I2C)
F0EAh	75h	–	ESFR	–	Зарезервировано
F0ECh	76h	DCMPSP	ESFR	0000h	Регистр выбора и программирования для DCMPx (OCDS)
F0EEh	77h	DCMPDP	ESFR	0000h	Регистр данных программирования для DCMPx (OCDS)
F0F0h	78h	DTREVT	ESFR	0000h	Регистр управления комбинациями аппаратных отладочных событий (OCDS)
F0F2h	79h	DEXEVT	ESFR	0000h	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)
F0F4h	7Ah	DSWEVT	ESFR	0000h	Регистр управления отладочными событиями вывода BREAK и программного обеспечения (OCDS)
F0F6h	7Bh	–	ESFR	–	Зарезервировано
F0F8h	7Ch	DIP	ESFR	0000h	Регистр указателя машинной команды
F0FAh	7Dh	DIPX	ESFR	3000h	Регистр расширенного указателя инструкций
F0FCh	7Eh	DBGSR	ESFR	0000h	Регистр состояния отладки (OCDS)
F0FEh	7Fh	–	ESFR	–	Зарезервировано

Продолжение таблицы Б.1

1	2	3	4	5	6
F100h	80h	DP0L	ESFR-b	0000h	Регистр выбора направления порта P0L (порт P0)
F102h	81h	DP0H	ESFR-b	0000h	Регистр выбора направления порта P0H (порт P0)
F104h	82h	DP1L	ESFR-b	0000h	Регистр выбора направления порта P1L (порт P1)
F106h	83h	DP1H	ESFR-b	0000h	Регистр выбора направления порта P1H (порт P1)
F108h	84h	RP0H	ESFR-b	--XXh	Регистр начальной конфигурации внешней шины
F10Ah–F112h	85h–89h	–	ESFR-b	–	Зарезервировано
F114h	8Ah	XBCON1	ESFR-b	XXXXh	Регистр 1 управления 1 (XBUS)
F116h	8Bh	XBCON2	ESFR-b	XXXXh	Регистр 2 управления 2 (XBUS)
F118h	8Ch	XBCON3	ESFR-b	XXXXh	Регистр 3 управления 3 (XBUS)
F11Ah	8Dh	XBCON4	ESFR-b	XXXXh	Регистр 4 управления 4 (XBUS)
F11Ch	8Eh	XBCON5	ESFR-b	XXXXh	Регистр 5 управления 5 (XBUS)
F11Eh	8Fh	XBCON6	ESFR-b	XXXXh	Регистр 6 управления 6 (XBUS)
F120h	90h	CCU6_IC	ESFR-b	--00h	Регистр управления прерываниями (CAPCOM6)
F122h	91h	SSC1TIC	ESFR-b	0000h	Регистр контроля прерывания по передаче SSC1
F124h	92h	SSC1RIC	ESFR-b	0000h	Регистр контроля прерывания по приему SSC1
F126h	93h	SSC1EIC	ESFR-b	0000h	Регистр контроля прерывания по ошибке SSC1
F128h	94h	CAN_IC0	ESFR-b	0000h	Регистр контроля прерываний линии 0 (CAN)
F12Ah	95h	CAN_IC1	ESFR-b	0000h	Регистр контроля прерываний линии 1 (CAN)
F12Ch	96h	CAN_IC2	ESFR-b	0000h	Регистр контроля прерываний линии 2 (CAN)
F12Eh	97h	CAN_IC3	ESFR-b	0000h	Регистр контроля прерываний линии 3 (CAN)
F130h	98h	–	ESFR-b	–	–
F132h	99h	CAN_IC4	ESFR-b	0000h	Регистр контроля прерываний линии 4 (CAN)
F134h	9Ah	CAN_IC5	ESFR-b	0000h	Регистр контроля прерываний линии 5 (CAN)
F136h	9Bh	CAN_IC6	ESFR-b	0000h	Регистр контроля прерываний линии 6 (CAN)
F138h	9Ch	CAN_IC7	ESFR-b	0000h	Регистр контроля прерываний линии 7 (CAN)
F13Ah	9Dh	RTC_IC	ESFR-b	0000h	Регистр контроля прерываний RTC

Продолжение таблицы Б.1

1	2	3	4	5	6
F13Ch	9Eh	S1TBIC	ESFR-b	0000h	Регистр управления прерыванием по опустошению буфера передатчика ASC1
F13Eh	9Fh	–	ESFR-b	–	–
F140h	A0h	CAN_IC8	ESFR-b	0000h	Регистр контроля прерываний линии 8 (CAN)
F142h	A1h	CAN_IC9	ESFR-b	0000h	Регистр контроля прерываний линии 9 (CAN)
F144h	A2h	CAN_IC10	ESFR-b	0000h	Регистр контроля прерываний линии 10 (CAN)
F146h	A3h	CAN_IC11	ESFR-b	0000h	Регистр контроля прерываний линии 11 (CAN)
F148h	A4h	CAN_IC12	ESFR-b	0000h	Регистр контроля прерываний линии 12 (CAN)
F14Ah	A5h	CAN_IC13	ESFR-b	0000h	Регистр контроля прерываний линии 13 (CAN)
F14Ch	A6h	CAN_IC14	ESFR-b	0000h	Регистр контроля прерываний линии 14 (CAN)
F14Eh	A7h	CAN_IC15	ESFR-b	0000h	Регистр контроля прерываний линии 15 (CAN)
F150h	A8h	RTC_IC1	ESFR-b	0000h	Регистр контроля прерывания 1 RTC
F152h	A9h	RTC_IC2	ESFR-b	0000h	Регистр контроля прерывания 2 RTC
F154h	AAh	RTC_IC3	ESFR-b	0000h	Регистр контроля прерывания 3 RTC
F156h	ABh	–	ESFR-b	–	Зарезервировано
F158h–F15Eh	ACh–AFh	–	ESFR-b	–	Зарезервировано
F160h	B0h	CC16IC	ESFR-b	0000h	Регистр управления прерываниями 16 (CAPCOM2)
F162h	B1h	CC17IC	ESFR-b	0000h	Регистр управления прерываниями 17 (CAPCOM2)
F164h	B2h	CC18IC	ESFR-b	0000h	Регистр управления прерываниями 18 (CAPCOM2)
F166h	B3h	CC19IC	ESFR-b	0000h	Регистр управления прерываниями 19 (CAPCOM2)
F168h	B4h	CC20IC	ESFR-b	0000h	Регистр управления прерываниями 20 (CAPCOM2)
F16Ah	B5h	CC21IC	ESFR-b	0000h	Регистр управления прерываниями 21 (CAPCOM2)
F16Ch	B6h	CC22IC	ESFR-b	0000h	Регистр управления прерываниями 22 (CAPCOM2)

Продолжение таблицы Б.1

1	2	3	4	5	6
F16Eh	B7h	CC23IC	ESFR-b	0000h	Регистр управления прерываниями 23 (CAPCOM2)
F170h	B8h	CC24IC	ESFR-b	0000h	Регистр управления прерываниями 24 (CAPCOM2)
F172h	B9h	CC25IC	ESFR-b	0000h	Регистр управления прерываниями 25 (CAPCOM2)
F174h	BAh	CC26IC	ESFR-b	0000h	Регистр управления прерываниями 26 (CAPCOM2)
F176h	BBh	CC27IC	ESFR-b	0000h	Регистр управления прерываниями 27 (CAPCOM2)
F178h	BCh	CC28IC	ESFR-b	0000h	Регистр управления прерываниями 28 (CAPCOM2)
F17Ah	BDh	T7IC	ESFR-b	--00h	Регистр управления прерываниями таймера T7 (CAPCOM2)
F17Ch	BEh	T8IC	ESFR-b	--00h	Регистр управления прерываниями таймера T8 (CAPCOM2)
F17Eh	BFh	–	ESFR-b	–	Зарезервировано
F180h	C0h	EOPIC	ESFR-b	0000h	Регистр контроля за прерыванием по завершению PEC передачи
F182h	C1h	S1TIC	ESFR-b	0000h	Регистр управления прерыванием по передаче (ASC1)
F184h	C2h	CC29IC	ESFR-b	0000h	Регистр управления прерываниями 29 (CAPCOM2)
F186h	C3h	I2C_DIC	ESFR-b	--00h	Регистр управления прерыванием (I2C)
F188h	C4h	CCU6_EIC	ESFR-b	--00h	Регистр управления прерываниями по ошибке (CAPCOM6)
F18Ah	C5h	S1RIC	ESFR-b	0000h	Регистр управления прерыванием по приему (ASC1)
F18Ch	C6h	CC30IC	ESFR-b	0000h	Регистр управления прерываниями 30 (CAPCOM2)
F18Eh	C7h	–	ESFR-b	–	Зарезервировано
F190h	C8h	CCU6_T12IC	ESFR-b	--00h	Регистр управления прерываниями таймера T12 (CAPCOM6)
F192h	C9h	S1EIC	ESFR-b	0000h	Регистр управления прерыванием по ошибке (ASC1)
F194h	CAh	CC31IC	ESFR-b	0000h	Регистр управления прерываниями 31 (CAPCOM2)
F196h	CBh	–	ESFR-b	–	Зарезервировано
F198h	CCh	CCU6_T13IC	ESFR-b	--00h	Регистр управления прерываниями таймера T13 (CAPCOM6)
F19Ah	CDh	WDTIC	ESFR-b	0000h	Регистр управления прерыванием сторожевого таймера (WDT)
F19Ch	CEh	S0TBIC	ESFR-b	0000h	Регистр контроля прерывания по опустошению буфера передачи (ASC0)
F19Eh	CFh	–	–	–	Зарезервировано

Продолжение таблицы Б.1

1	2	3	4	5	6
F1A0h–F1B6h	D0h–DBh	–	ESFR-b	–	Зарезервировано
F1B8h	DCh	S1CON	ESFR-b	0000h	Регистр управления (ASC1)
F1BAh–F1BEh	DDh–DFh	–	ESFR-b	–	Зарезервировано
F1C0h	E0h	EXICON	ESFR-b	0000h	Регистр контроля внешних прерываний
F1C2h	E1h	ODP2	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P2
F1C6h	E3h	ODP3	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P3
F1C8h	E4h	–	ESFR-b	–	Зарезервировано
F1CAh	E5h	ODP4	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P4
F1CCh	–	–	–	–	–
F1CEh	E7h	ODP6	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P6
F1D0h	–	–	–	–	–
F1D2h	E9h	ODP7	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P7
F1D4h	EAh	–	ESFR-b	-	Зарезервировано
F1D6h	EBh	ODP9	ESFR-b	0000h	Регистр управления режимом с открытым стоком порта P9
F1D8h	ECh	EXISEL1	ESFR-b	0000h	Регистр 1 выбора источника внешних прерываний
F1DAh	EDh	EXISEL0	ESFR-b	0000h	Регистр 0 выбора источника внешних прерываний
F1DCh–F1E8h	EEh–F4h	–	ESFR-b	–	Зарезервировано
F1EAh	F5h	ALTSEL0P1L	ESFR	0000h	Регистр управления альтернативными функциями порта P1L (порт P1)
F1ECh	F6h	ALTSEL0P1H	ESFR	0000h	Регистр управления альтернативными функциями порта P1H (порт P1)
F1EEh	F7h	ALTSEL0P2	ESFR	0000h	Регистр управления альтернативными функциями порта P2
F1F0h	F8h	ALTSEL0P3	ESFR	0000h	Регистр 0 управления альтернативными функциями порта P3
F1F2h	F9h	ALTSEL1P3	ESFR	0000h	Регистр 1 управления альтернативными функциями порта P3
F1F4h	FAh	ALTSEL0P4	ESFR	0000h	Регистр управления альтернативными функциями порта P4

Продолжение таблицы Б.1

1	2	3	4	5	6
F1F6h	FBh	ALTSEL0P6	ESFR	0000h	Регистр управления альтернативными функциями порта P6
F1F8h	FCh	ALTSEL0P7	ESFR	0000h	Регистр 0 управления альтернативными функциями порта P7
F1FAh	FDh	ALTSEL1P7	ESFR	0000h	Регистр 1 управления альтернативными функциями порта P7
F1FCh	FEh	ALTSEL0P9	ESFR	0000h	Регистр 0 управления альтернативными функциями порта P9
F1FEh	FFh	ALTSEL1P9	ESFR	0000h	Регистр 1 управления альтернативными функциями порта P9
FE00h	00h	DPP0	SFR	0000h	Регистр указателя нулевой страницы данных (10 битов) ЦПУ
FE02h	01h	DPP1	SFR	0001h	Регистр указателя первой страницы данных (10 битов) ЦПУ
FE04h	02h	DPP2	SFR	0002h	Регистр указателя второй страницы данных (10 битов) ЦПУ
FE06h	03h	DPP3	SFR	0003h	Регистр указателя третьей страницы данных (10 битов) ЦПУ
FE08h	04h	CSP	SFR	0000h	Регистр указателя сегмента кода (8 битов) ЦПУ
FE0Ah	05h	–	SFR	–	Зарезервировано
FE0Ch	06h	MDH	SFR	0000h	Старший регистр умножения/деления ЦПУ
FE0Eh	07h	MDL	SFR	0000h	Младший регистр умножения/деления ЦПУ
FE10h	08h	CP	SFR	FC00h	Регистр контекстного указателя ЦПУ
FE12h	09h	SP	SFR	FC00h	Регистр указателя стека ЦПУ
FE14h	0Ah	STKOV	SFR	FA00h	Регистр указателя переполнения стека ЦПУ
FE16h	0Bh	STKUN	SFR	FC00h	Регистр опустошения стека ЦПУ
FE18h	0Ch	ADDRSEL1	SFR	0000h	Регистр выбора адреса 1
FE1Ah	0Dh	ADDRSEL2	SFR	0000h	Регистр выбора адреса 2
FE1Ch	0Eh	ADDRSEL3	SFR	0000h	Регистр выбора адреса 3
FE1Eh	0Fh	ADDRSEL4	SFR	0000h	Регистр выбора адреса 4
FE20h–FE26h	10h–13h	–	SFR	–	Зарезервировано
FE28h	14h	CC2_SEM	SFR	0000h	Регистр режима однократного срабатывания (CAPCOM2)
FE2Ah	15h	CC2_SEE	SFR	0000h	Регистр однократного срабатывания (CAPCOM2)
FE2Ch	16h	CC1_SEM	SFR	0000h	Регистр режима однократного срабатывания (CAPCOM1)

Продолжение таблицы Б.1

1	2	3	4	5	6
FE2Eh	17h	CC1_SEE	SFR	0000h	Регистр однократного срабатывания (CAPCOM1)
FE30h–FE3Eh	18h–1Fh	–	SFR	–	Зарезервировано
FE40h	20h	T2	SFR	0000h	Регистр таймера 2 (GPT1)
FE42h	21h	T3	SFR	0000h	Регистр таймера 3 (GPT1)
FE44h	22h	T4	SFR	0000h	Регистр таймера 4 (GPT1)
FE46h	23h	T5	SFR	0000h	Регистр таймера 5 (GPT2)
FE48h	24h	T6	SFR	0000h	Регистр таймера 6 (GPT2)
FE4Ah	25h	CAPREL	SFR	0000h	Регистр захвата/перезагрузки (GPT2)
FE4Ch, FE4Eh	26h, 27h	–	SFR	–	Зарезервировано
FE50h	28h	T0	SFR	0000h	Регистр таймера T0 (CAPCOM1)
FE52h	29h	T1	SFR	0000h	Регистр таймера T1 (CAPCOM1)
FE54h	2Ah	T0REL	SFR	0000h	Регистр загрузки таймера 0 (CAPCOM1)
FE56h	2Bh	T1REL	SFR	0000h	Регистр загрузки таймера 1 (CAPCOM1)
FE58h, FE5Ah	2Ch, 2Dh	–	SFR	–	Зарезервировано
FE5Ch	2Eh	MAL	SFR	0000h	Регистр младшего байта аккумулятора (MAC)
FE5Eh	2Fh	MAH	SFR	0000h	Регистр старшего байта аккумулятора (MAC)
FE60h	30h	CC16	SFR	0000h	Регистр захвата/сравнения 16 (CAPCOM2)
FE62h	31h	CC17	SFR	0000h	Регистр захвата/сравнения 17 (CAPCOM2)
FE64h	32h	CC18	SFR	0000h	Регистр захвата/сравнения 18 (CAPCOM2)
FE66h	33h	CC19	SFR	0000h	Регистр захвата/сравнения 19 (CAPCOM2)
FE68h	34h	CC20	SFR	0000h	Регистр захвата/сравнения 20 (CAPCOM2)
FE6Ah	35h	CC21	SFR	0000h	Регистр захвата/сравнения 21 (CAPCOM2)
FE6Ch	36h	CC22	SFR	0000h	Регистр захвата/сравнения 22 (CAPCOM2)
FE6Eh	37h	CC23	SFR	0000h	Регистр захвата/сравнения 23 (CAPCOM2)
FE70h	38h	CC24	SFR	0000h	Регистр захвата/сравнения 24 (CAPCOM2)
FE72h	39h	CC25	SFR	0000h	Регистр захвата/сравнения 25 (CAPCOM2)
FE74h	3Ah	CC26	SFR	0000h	Регистр захвата/сравнения 26 (CAPCOM2)
FE76h	3Bh	CC27	SFR	0000h	Регистр захвата/сравнения 27 (CAPCOM2)
FE78h	3Ch	CC28	SFR	0000h	Регистр захвата/сравнения 28 (CAPCOM2)

Продолжение таблицы Б.1

1	2	3	4	5	6
FE7Ah	3Dh	CC29	SFR	0000h	Регистр захвата/сравнения 29 (CAPCOM2)
FE7Ch	3Eh	CC30	SFR	0000h	Регистр захвата/сравнения 30 (CAPCOM2)
FE7Eh	3Fh	CC31	SFR	0000h	Регистр захвата/сравнения 31 (CAPCOM2)
FE80h	40h	CC0	SFR	0000h	Регистр захвата/сравнения 0 (CAPCOM1)
FE82h	41h	CC1	SFR	0000h	Регистр захвата/сравнения 1 (CAPCOM1)
FE84h	42h	CC2	SFR	0000h	Регистр захвата/сравнения 2 (CAPCOM1)
FE86h	43h	CC3	SFR	0000h	Регистр захвата/сравнения 3 (CAPCOM1)
FE88h	44h	CC4	SFR	0000h	Регистр захвата/сравнения 4 (CAPCOM1)
FE8Ah	45h	CC5	SFR	0000h	Регистр захвата/сравнения 5 (CAPCOM1)
FE8Ch	46h	CC6	SFR	0000h	Регистр захвата/сравнения 6 (CAPCOM1)
FE8Eh	47h	CC7	SFR	0000h	Регистр захвата/сравнения 7 (CAPCOM1)
FE90h	48h	CC8	SFR	0000h	Регистр захвата/сравнения 8 (CAPCOM1)
FE92h	49h	CC9	SFR	0000h	Регистр захвата/сравнения 9 (CAPCOM1)
FE94h	4Ah	CC10	SFR	0000h	Регистр захвата/сравнения 10 (CAPCOM1)
FE96h	4Bh	CC11	SFR	0000h	Регистр захвата/сравнения 11 (CAPCOM1)
FE98h	4Ch	CC12	SFR	0000h	Регистр захвата/сравнения 12 (CAPCOM1)
FE9Ah	4Dh	CC13	SFR	0000h	Регистр захвата/сравнения 13 (CAPCOM1)
FE9Ch	4Eh	CC14	SFR	0000h	Регистр захвата/сравнения 14 (CAPCOM1)
FE9Eh	4Fh	CC15	SFR	0000h	Регистр захвата/сравнения 15 (CAPCOM1)
FEA0h–FEA4h	50h–52h	–	–	–	Зарезервировано
FEA6h	53h	S1TBUF	SFR	0000h	Буферный регистр передатчика (ASC1)
FEA8h	54h	S1RBUF	SFR	0000h	Буферный регистр приемника (ASC1)
FEAAh	55h	S1BG	SFR	0000h	Регистр скорости пересылки (ASC1)
FEACh	56h	S1FDV	SFR	0000h	Регистр дробного делителя (ASC1)
FEAEh	57h	WDTCON	SFR	00000000 10000xx1b	Регистр сторожевого таймера (WDT)
FEB0h	58h	S0TBUF	SFR	0000h	Буферный регистр передатчика (ASC0)
FEB2h	59h	S0RBUF	SFR	0000h	Буферный регистр приемника (ASC0)
FEB4h	5Ah	S0BG	SFR	0000h	Регистр скорости пересылки (ASC0)

Продолжение таблицы Б.1

1	2	3	4	5	6
FEB6h	5Bh	S0FDV	SFR	0000h	Регистр дробного делителя (ASC0)
FEB8h	5Ch	PECSN12	SFR	0000h	Регистр указателя сегмента 12 (PEC)
FEBAh	5Dh	PECSN13	SFR	0000h	Регистр указателя сегмента 13 (PEC)
FEBCh	5Eh	PECSN14	SFR	0000h	Регистр указателя сегмента 14 (PEC)
FEBEh	5Fh	PECSN15	SFR	0000h	Регистр указателя сегмента 15 (PEC)
FEC0h	60h	PECC0	SFR	0000h	Регистр управления PEC-каналом 0
FEC2h	61h	PECC1	SFR	0000h	Регистр управления PEC-каналом 1
FEC4h	62h	PECC2	SFR	0000h	Регистр управления PEC-каналом 2
FEC6h	63h	PECC3	SFR	0000h	Регистр управления PEC-каналом 3
FEC8h	64h	PECC4	SFR	0000h	Регистр управления PEC-каналом 4
FECAh	65h	PECC5	SFR	0000h	Регистр управления PEC-каналом 5
FECCh	66h	PECC6	SFR	0000h	Регистр управления PEC-каналом 6
FECEh	67h	PECC7	SFR	0000h	Регистр управления PEC-каналом 7
FED0h	68h	PECSN0	SFR	0000h	Регистр указателя сегмента 0 (PEC)
FED2h	69h	PECSN1	SFR	0000h	Регистр указателя сегмента 1 (PEC)
FED4h	6Ah	PECSN2	SFR	0000h	Регистр указателя сегмента 2 (PEC)
FED6h	6Bh	PECSN3	SFR	0000h	Регистр указателя сегмента 3 (PEC)
FED8h	6Ch	PECSN4	SFR	0000h	Регистр указателя сегмента 4 (PEC)
FEDAh	6Dh	PECSN5	SFR	0000h	Регистр указателя сегмента 5 (PEC)
FEDCh	6Eh	PECSN6	SFR	0000h	Регистр указателя сегмента 6 (PEC)
FEDEh	6Fh	PECSN7	SFR	0000h	Регистр указателя сегмента 7 (PEC)
FEE0h	70h	PECSN8	SFR	0000h	Регистр указателя сегмента 8 (PEC)
FEE2h	71h	PECSN9	SFR	0000h	Регистр указателя сегмента 9 (PEC)
FEE4h	72h	PECSN10	SFR	0000h	Регистр указателя сегмента 10 (PEC)
FEE6h	73h	PECSN11	SFR	0000h	Регистр указателя сегмента 11 (PEC)
FEE8h	74h	PECC8	SFR	0000h	Регистр управления PEC-каналом 8
FEEAh	75h	PECC9	SFR	0000h	Регистр управления PEC-каналом 9
FEECh	76h	PECC10	SFR	0000h	Регистр управления PEC-каналом 10
FEEEh	77h	PECC11	SFR	0000h	Регистр управления PEC-каналом 11
FEF0h	78h	PECXC0	SFR	0000h	Регистр расширенного управления каналом 0 (PEC)
FEF2h	79h	PECXC2	SFR	0000h	Регистр расширенного управления каналом 2 (PEC)
FEF4h, FEF6h	7Ah, 7Bh	–	SFR	–	Зарезервировано
FEF8h	7Ch	PECC12	SFR	0000h	Регистр управления PEC-каналом 12
FEFAh	7Dh	PECC13	SFR	0000h	Регистр управления PEC-каналом 13
FEFCh	7Eh	PECC14	SFR	0000h	Регистр управления PEC-каналом 14
FEFEh	7Fh	PECC15	SFR	0000h	Регистр управления PEC-каналом 15
FF00h	80h	P0L	SFR-b	00h	Младший регистр порта P0
FF02h	81h	P0H	SFR-b	00h	Старший регистр порта P0
FF04h	82h	P1L	SFR-b	00h	Младший регистр порта P1
FF06h	83h	P1H	SFR-b	00h	Старший регистр порта P1
FF08h	84h	IDX0	SFR-b	0000h	Регистр 0 указателя адреса MAC
FF0Ah	85h	IDX1	SFR-b	0000h	Регистр 1 указателя адреса MAC
FF0Ch	86h	BUSCON0	SFR-b	0000h	Регистр конфигурации шины 0

Продолжение таблицы Б.1

1	2	3	4	5	6
FF0Eh	87h	MDC	SFR-b	0000h	Регистр управления умножением/делением ЦПУ
FF10h	88h	PSW	SFR-b	0000h	Регистр слова состояния процессора ЦПУ
FF12h	89h	SYSCON	SFR-b	XXXXh	Регистр конфигурации системы (ЦПУ)
FF14h	8Ah	BUSCON1	SFR-b	0000h	Регистр конфигурации шины 1
FF16h	8Bh	BUSCON2	SFR-b	0000h	Регистр конфигурации шины 2
F18h	8Ch	BUSCON3	SFR-b	0000h	Регистр конфигурации шины 3
FF1Ah	8Dh	BUSCON4	SFR-b	0000h	Регистр конфигурации шины 4
FF1Ch	8Eh	ZEROS	SFR-b	0000h	Регистр постоянного значения 0
FF1Eh	8Fh	ONES	SFR-b	FFFFh	Регистр постоянного значения 1
FF20h	90h	CC2_ T78CON	SFR-b	0000h	Регистр управления таймерами T7 и T8 (CAPCOM2)
FF22h	91h	CC2_M4	SFR-b	0000h	Регистр управления режимом захвата/сравнения 4 (CAPCOM1)
FF24h	92h	CC2_M5	SFR-b	0000h	Регистр управления режимом захвата/сравнения 5 (CAPCOM1)
FF26h	93h	CC2_M6	SFR-b	0000h	Регистр управления режимом захвата/сравнения 6 (CAPCOM1)
FF28h	94h	CC2_M7	SFR-b	0000h	Регистр управления режимом захвата/сравнения 7 (CAPCOM1)
FF2Ah	95h	CC2_DRM	SFR-b	0000h	Регистр двухрегистрового режима сравнения (CAPCOM2)
FF2Ch	96h	CC2_OUT	SFR-b	0000h	Регистр выводов сравнения (CAPCOM2)
FF2Eh–FF3Eh	97h–9Fh	–	SFR-b	–	Зарезервировано
FF40h	A0h	T2CON	SFR-b	0000h	Регистр управления таймера 2 (GPT1)
FF42h	A1h	T3CON	SFR-b	0000h	Регистр управления таймера 3 (GPT1)
FF44h	A2h	T4CON	SFR-b	0000h	Регистр управления таймера 4 (GPT1)
FF46h	A3h	T5CON	SFR-b	0000h	Регистр управления таймера 5 (GPT2)
FF48h	A4h	T6CON	SFR-b	0000h	Регистр управления таймера 6 (GPT2)
FF4Ah–FF4Eh	A5h–A7h	–	SFR-b	–	Зарезервировано
FF50h	A8h	CC1_T01CON	SFR-b	0000h	Регистр управления таймерами T0 и T1 (CAPCOM1)
FF52h	A9h	CC1_M0	SFR-b	0000h	Регистр управления режимом захвата/сравнения 0 (CAPCOM1)
FF54h	AAh	CC1_M1	SFR-b	0000h	Регистр управления режимом захвата/сравнения 1 (CAPCOM1)
FF56h	ABh	CC1_M2	SFR-b	0000h	Регистр управления режимом захвата/сравнения 2 (CAPCOM1)
FF58h	ACh	CC1_M3	SFR-b	0000h	Регистр управления режимом захвата/сравнения 3 (CAPCOM1)
FF5Ah	ADh	CC1_DRM	SFR-b	0000h	Регистр двухрегистрового режима сравнения (CAPCOM1)
FF5Ch	AEh	CC1_OUT	SFR-b	0000h	Регистр выводов сравнения (CAPCOM1)
FF5Eh	AFh	SSC1CON	SFR-b	0000h	Регистр управления (SSC1)

Продолжение таблицы Б.1

1	2	3	4	5	6
FF60h	B0h	T2IC	SFR-b	0000h	Регистр управления прерываниями таймера 2 (GPT1)
FF62h	B1h	T3IC	SFR-b	0000h	Регистр управления прерываниями таймера 3 (GPT1)
FF64h	B2h	T4IC	SFR-b	0000h	Регистр управления прерываниями таймера 4 (GPT1)
FF66h	B3h	T5IC	SFR-b	0000h	Регистр управления прерываниями таймерами 5 (GPT2)
FF68h	B4h	T6IC	SFR-b	0000h	Регистр управления прерываниями таймера 6 (GPT2)
FF6Ah	B5h	CRIC	SFR-b	0000h	Регистр управления прерыванием захвата/перезагрузки (GPT1, GPT2)
FF6Ch	B6h	S0TIC	SFR-b	0000h	Регистр контроля прерывания по передаче (ASC0)
FF6Eh	B7h	S0RIC	SFR-b	0000h	Регистр контроля прерывания по приему (ASC0)
FF70h	B8h	S0EIC	SFR-b	0000h	Регистр контроля прерывания по ошибке (ASC0)
FF72h	B9h	SSC0TIC	SFR-b	0000h	Регистр контроля прерывания по передаче (SSC0)
FF74h	BAh	SSC0RIC	SFR-b	0000h	Регистр контроля прерывания по приему (SSC0)
FF76h	BBh	SSC0EIC	SFR-b	0000h	Регистр контроля прерывания по ошибке (SSC0)
FF78h	BCh	CC0IC	SFR-b	0000h	Регистр управления прерываниями 0 (CAPCOM1)
FF7Ah	BDh	CC1IC	SFR-b	0000h	Регистр управления прерываниями 1 (CAPCOM1)
FF7Ch	BEh	CC2IC	SFR-b	0000h	Регистр управления прерываниями 2 (CAPCOM1)
FF7Eh	BFh	CC3IC	SFR-b	0000h	Регистр управления прерываниями 3 (CAPCOM1)
FF80h	C0h	CC4IC	SFR-b	0000h	Регистр управления прерываниями 4 (CAPCOM1)
FF82h	C1h	CC5IC	SFR-b	0000h	Регистр управления прерываниями 5 (CAPCOM1)
FF84h	C2h	CC6IC	SFR-b	0000h	Регистр управления прерываниями 6 (CAPCOM1)
FF86h	C3h	CC7IC	SFR-b	0000h	Регистр управления прерываниями 7 (CAPCOM1)
FF88h	C4h	CC8IC	SFR-b	0000h	Регистр управления прерываниями 8 (CAPCOM1)
FF8Ah	C5h	CC9IC	SFR-b	0000h	Регистр управления прерываниями 9 (CAPCOM1)
FF8Ch	C6h	CC10IC	SFR-b	0000h	Регистр управления прерываниями 10 (CAPCOM1)

Продолжение таблицы Б.1

1	2	3	4	5	6
FF8Eh	C7h	CC11IC	SFR-b	0000h	Регистр управления прерываниями 11 (CAPCOM1)
FF90h	C8h	CC12IC	SFR-b	0000h	Регистр управления прерываниями 12 (CAPCOM1)
FF92h	C9h	CC13IC	SFR-b	0000h	Регистр управления прерываниями 13 (CAPCOM1)
FF94h	CAh	CC14IC	SFR-b	0000h	Регистр управления прерываниями 14 (CAPCOM1)
FF96h	CBh	CC15IC	SFR-b	0000h	Регистр управления прерываниями 15 (CAPCOM1)
FF98h, FF9Ah	CCh, CDh	–	–	–	Зарезервировано
FF9Ch	CEh	T0IC	SFR-b	--00h	Регистр управления прерываниями таймера 0 (CAPCOM1)
FF9Eh	CFh	T1IC	SFR-b	--00h	Регистр управления прерываниями таймера 1 (CAPCOM1)
FFA0h	D0h	–	–	–	Зарезервировано
FFA2h	D1h	P5	SFR-b	0000h	Регистр данных порта P5
FFA4h	D2h	–	SFR-b	–	Зарезервировано
FFA6h	D3h	PECISNCL	SFR-b	0000h	Регистр управления узлом прерываний PEC
FFA8h	D4h	PECISNCH	SFR-b	0000h	Регистр управления узлом прерываний PEC
FFAAh	D5h	FOCON	SFR-b	0000h	Регистр управления выходным тактовым сигналом
FFACh	D6h	TFR	SFR-b	0000h	Регистр флагов ловушек
FFAEh	D7h	WDTCON	SFR-b	008Xh	Регистр управления сторожевым таймером WDT
FFB0h	D8h	S0CON	SFR-b	0000h	Регистр управления ASC0
FFB2h	D9h	SSC0CON	SFR-b	0000h	Регистр управления SSC0
FFB4h–FFBEh	DAh–DFh	–	SFR-b	–	Зарезервировано
FFC0h	E0h	P2	SFR-b	0000h	Регистр данных порта P2
FFC2h	E1h	DP2	SFR-b	0000h	Регистр выбора направления порта P2
FFC4h	E2h	P3	SFR-b	0000h	Регистр данных порта P3
FFC6h	E3h	DP3	SFR-b	0000h	Регистр выбора направления порта P3
FFC8h	E4h	P4	SFR-b	0000h	Регистр данных порта P4
FFCAh	E5h	DP4	SFR-b	0000h	Регистр выбора направления порта P4
FFCCh	E6h	P6	SFR-b	0000h	Регистр данных порта P6
FFCEh	E7h	DP6	SFR-b	0000h	Регистр выбора направления порта P6
FFD0h	E8h	P7	SFR-b	0000h	Регистр данных порта P7
FFD2h	E9h	DP7	SFR-b	0000h	Регистр выбора направления порта P7

Окончание таблицы Б.1

1	2	3	4	5	6
FFD4h	EAh	P9	SFR-b	0000h	Регистр данных порта P9
FFD6h	EBh	DP9	SFR-b	0000h	Регистр выбора направления порта P9
FFD8h	ECh	–	SFR-b	–	Зарезервировано
FFDAh	EDh	MRW	SFR-b	0000h	Регистр повторения блока MAC
FFDCh	EEh	MCW	SFR-b	0000h	Регистр управления блока MAC
FFDEh	EFh	MSW	SFR-b	0200h	Статусный регистр блока MAC
FFE0h	F0h	–	SFR-b	–	Зарезервировано
FFE2h	F1h	ASC0ID	SFR-b	44XXh	Регистр идентификации ASC01
FFE4h	F2h	SSC0ID	SFR-b	45XXh	Регистр идентификации SSC0
FFE6h	F3h	GPTID	SFR-b	58XXh	Регистр идентификации GPT1 и GPT2
FFE8h–FFFFh	F4h–FFh	–	SFR-b	–	Зарезервировано

Таблица Б.2

Адрес	Мнемоника	Область памяти	Сброс	Описание
1	2	3	4	5
E800h	CLC	XSFR	0003h	Регистр управления частотой (CAN)
E802h			0000h	
E804h–E806h	–	XSFR	–	Зарезервировано
E808h	ID	XSFR	C051h	Регистр идентификации (CAN)
E80Ah			002Bh	
E80Ch	FDR	XSFR	0000h	Регистр делителя (CAN)
E80Eh			0000h	
E810h–E88Eh	–	XSFR	–	Зарезервировано
E890h	CCU6_T12	XSFR	0000h	Регистр счета таймера T12 (CAPCOM6)
E892h	CCU6_T12PR	XSFR	0000h	Регистр периода таймера T12 (CAPCOM6)
E894h	CCU6_T12DTC	XSFR	0000h	Регистр контроля задержки «dead-time» T12 (CAPCOM6)
E896h	–	XSFR	–	Зарезервировано
E898h	CCU6_CC60R	XSFR	0000h	Регистр захвата/сравнения канала 0 (CAPCOM6)
E89Ah	CCU6_CC61R	XSFR	0000h	Регистр захвата/сравнения канала 1 (CAPCOM6)
E89Ch	CCU6_CC62R	XSFR	0000h	Регистр захвата/сравнения канала 2 (CAPCOM6)
E89Eh	–	XSFR	–	Зарезервировано
E8A0h	CCU6_CC60SR	XSFR	0000h	Теневой регистр захвата/сравнения канала 0 (CAPCOM6)
E8A2h	CCU6_CC61SR	XSFR	0000h	Теневой регистр захвата/сравнения канала 1 (CAPCOM6)
E8A4h	CCU6_CC62SR	XSFR	0000h	Теневой регистр захвата/сравнения канала 2 (CAPCOM6)
E8A6h	CCU6_TCTR4	XSFR	0000h	Регистр управления таймерами 4 (CAPCOM6)

Продолжение таблицы Б.2

1	2	3	4	5
E8A8h	CCU6_ CMPSTAT	XSFR	0000h	Регистр состояния захвата/сравнения (CAPCOM6)
E8AAh	CCU6_ CMPMODIF	XSFR	0000h	Регистр изменения состояния захвата/сравнения (CAPCOM6)
E8ACh	CCU6_TCTR0	XSFR	0000h	Регистр управления таймерами 0 (CAPCOM6)
E8AEh	CCU6_TCTR2	XSFR	0000h	Регистр управления таймерами 2 (CAPCOM6)
E8B0h	CCU6_T13	XSFR	0000h	Регистр счета таймера T13 (CAPCOM6)
E8B2h	CCU6_T13PR	XSFR	0000h	Регистр периода таймера T13 (CAPCOM6)
E8B4h	CCU6_CC63R	XSFR	0000h	Регистр захвата/сравнения канала 3 (CAPCOM6)
E8B6h	CCU6_CC63SR	XSFR	0000h	Теневой регистр захвата/сравнения 3 (CAPCOM6)
E8B8h– E8BEh	–	XSFR	–	Зарезервировано
E8C0h	CCU6_ MODCTR	XSFR	0000h	Регистр управления модуляцией (CAPCOM6)
E8C2h	CCU6_ TRPCTR	XSFR	0000h	Регистр управления ловушками (CAPCOM6)
E8C4h	CCU6_ PSLR	XSFR	0000h	Регистр уровня пассивного состояния (CAPCOM6)
E8C6h	CCU6_ T12MSEL	XSFR	0000h	Регистр выбора режима захвата/сравнения T12 (CAPCOM6)
E8C8h	–	XSFR	–	Зарезервировано
E8CAh	CCU6_ MCMOUTS	XSFR	0000h	Теневой регистр управления выходами в мультиканальном режиме (CAPCOM6)
E8CCh	CCU6_ MCMOUT	XSFR	0000h	Регистр управления выходами в мультиканальном режиме (CAPCOM6)
E8CEh	CCU6_ MCMCTR	XSFR	0000h	Регистр управления мультиканальным режимом (CAPCOM6)
E8D0h	CCU6_IS	XSFR	0000h	Регистр состояния прерываний (CAPCOM6)
E8D2h	CCU6_ISS	XSFR	0000h	Регистр установки состояния прерываний (CAPCOM6)
E8D4h	CCU6_ISR	XSFR	0000h	Регистр сброса состояния прерываний (CAPCOM6)
E8D6h	CCU6_INP	XSFR	3940h	Регистр указателя узла прерываний (CAPCOM6)
E8D8h	CCU6_IEN	XSFR	0000h	Регистр разрешения прерываний (CAPCOM6)
E8DAh– E8FEh	–	XSFR	–	Зарезервировано
E900h	LIST0	XSFR	7F00h	Регистр списка 0 (CAN)
E902h			007Fh	
E904h	LIST1	XSFR	0000h	Регистр свободного списка 1 (CAN)
E906h			0100h	
E908h	LIST2	XSFR	0000h	Регистр свободного списка 2 (CAN)
E90Ah			0100h	

Продолжение таблицы Б.2

1	2	3	4	5
E90Ch	LIST3	XSFR	0000h	Регистр свободного списка 3 (CAN)
E90Eh			0100h	
E910h	LIST4	XSFR	0000h	Регистр свободного списка 4 (CAN)
E912h			0100h	
E914h	LIST5	XSFR	0000h	Регистр свободного списка 5 (CAN)
E916h			0100h	
E918h	LIST6	XSFR	0000h	Регистр свободного списка 6 (CAN)
E91Ah			0100h	
E91Ch	LIST7	XSFR	0000h	Регистр свободного списка 7 (CAN)
E91Eh			0100h	
E920h–E93Eh	–	XSFR	–	Зарезервировано
E940h	MSPND0	XSFR	0000h	Регистр 0 ждущих прерываний (CAN)
E942h			0000h	
E944h	MSPND1	XSFR	0000h	Регистр 1 ждущих прерываний (CAN)
E946h			0000h	
E948h	MSPND2	XSFR	0000h	Регистр 2 ждущих прерываний (CAN)
E94Ah			0000h	
E94Ch	MSPND3	XSFR	0000h	Регистр 3 ждущих прерываний (CAN)
E94Eh			0000h	
E950h–E97Eh	–	XSFR	–	Зарезервировано
E980h	MSID0	XSFR	0020h	Регистр 0 индекса сообщения (CAN)
E982h			0000h	
E984h	MSID1	XSFR	0020h	Регистр 1 индекса сообщения (CAN)
E986h			0000h	
E988h	MSID2	XSFR	0020h	Регистр 2 индекса сообщения (CAN)
E98Ah			0000h	
E98Ch	MSID3	XSFR	0020h	Регистр 3 индекса сообщения (CAN)
E98Eh			0000h	
E990h–E9BEh	–	XSFR	–	Зарезервировано
E9C0h	MSIMASK	XSFR	0000h	Регистр маски индекса сообщения (CAN)
E9C2h			0000h	
E9C4h	PANCTR	XSFR	0301h	Регистр панели команд (CAN)
E9C6h			0000h	
E9C8h	MCR	XSFR	0000h	Регистр управления (CAN)
E9CAh			0000h	
E9CCh	MITR	XSFR	0000h	Регистр прерываний (CAN)
E9CEh			0000h	
E9D0h–E9FEh	–	XSFR	–	Зарезервировано
EA00h	NCR0	XSFR	0001h	Регистр управления узла 0 (CAN)
EA02h			0000h	
EA04h	NSR0	XSFR	0000h	Регистр состояния узла 0 (CAN)
EA06h			0000h	
EA08h	NIPR0	XSFR	0000h	Регистр указателя прерываний узла 0 (CAN)
EA0Ah			0000h	

Продолжение таблицы Б.2

1	2	3	4	5
EA0Ch	NPCR0	XSFR	0000h	Регистр управления портом узла 0 (CAN)
EA0Eh			0000h	
EA10h	NBTR0	XSFR	0000h	Регистр синхронизации битов 0 (CAN)
EA12h			0000h	
EA14h	NECNT0	XSFR	0000h	Регистр счетчика ошибок узла 0 (CAN)
EA16h			0060h	
EA18h	NFCR0	XSFR	0000h	Регистр счетчика сообщений 0 (CAN)
EA1Ah			0000h	
EA1Ch–FAFEh	–	XSFR	–	Зарезервировано
EB00h	NCR1	XSFR	0001h	Регистр управления узла 1 (CAN)
EB02h			0000h	
EB04h	NSR1	XSFR	0000h	Регистр состояния узла 1 (CAN)
EB06h			0000h	
EB08h	NIPR1	XSFR	0000h	Регистр указателя прерываний узла 1 (CAN)
EB0Ah			0000h	
EB0Ch	NPCR1	XSFR	0000h	Регистр управления портом узла 1 (CAN)
EB0Eh			0000h	
EB10h	NBTR1	XSFR	0000h	Регистр синхронизации битов 1 (CAN)
EB12h			0000h	
EB14h	NECNT1	XSFR	0000h	Регистр счетчика ошибок узла 1 (CAN)
EB16h			0060h	
EB18h	NFCR1	XSFR	0000h	Регистр счетчика сообщений 1 (CAN)
EB1Ah			0000h	
EB1Ch–FBFEh	–	XSFR	–	Зарезервировано
EC00h	MOFCR0	XSFR	0000h	Регистр управления функционированием объекта сообщения 0 (CAN)
EC02h			0000h	
EC04h	MOFGPR0	XSFR	0000h	Регистр указателя FIFO/шлюза объекта сообщения 0 (CAN)
EC06h			0000h	
EC08h	MOIPR0	XSFR	0000h	Регистр указателя прерываний объекта сообщения 0 (CAN)
EC0Ah			0000h	
EC0Ch	MOAMR0	XSFR	0000h	Регистр маски объекта сообщения 0 (CAN)
EC0Eh			0000h	
EC10h	MODATAL0	XSFR	0000h	Младший регистр данных объекта сообщения 0 (CAN)
EC12h			0000h	
EC14h	MODATAH0	XSFR	0000h	Старший регистр данных объекта сообщения 0 (CAN)
EC16h			0000h	
EC18h	MOAR0	XSFR	0000h	Регистр арбитража объекта сообщения 0 (CAN)
EC1Ah			0000h	
EC1Ch	MOCTR0/ MOSTAT0	XSFR	0000h	Регистр управления/состояния объекта сообщения 0 (CAN)
EC1Eh			0100h	
EC20h	MOFCR1	XSFR	0000h	Регистр управления функционированием объекта сообщения 1 (CAN)
EC22h			0000h	
EC24h	MOFGPR1	XSFR	0000h	Регистр указателя FIFO/шлюза объекта сообщения 1 (CAN)
EC26h			0000h	

Окончание таблицы Б.2

1	2	3	4	5
EC28h	MOIPR1	XSFR	0000h	Регистр указателя прерываний объекта сообщения 1 (CAN)
EC2Ah			0000h	
EC2Ch	MOAMR1	XSFR	0000h	Регистр маски объекта сообщения 1 (CAN)
EC2Eh			0000h	
EC30h	MODATAL1	XSFR	0000h	Младший регистр данных объекта сообщения 1 (CAN)
EC32h			0000h	
EC34h	MODATAH1	XSFR	0000h	Старший регистр данных объекта сообщения 1 (CAN)
EC36h			0000h	
EC38h	MOAR1	XSFR	0000h	Регистр арбитража объекта сообщения 1 (CAN)
EC3Ah			0000h	
EC3Ch	MOCTR1/ MOSTAT1	XSFR XSFR	0000h	Регистр управления/состояния объекта сообщения 1 (CAN)
EC3Eh			0200h	
Область объектов сообщений – с 2 по 62 (см. таблицы А.134, А.135)				
F3E0h	MOFCR63	XSFR	0000h	Регистр управления функционированием объекта сообщения 63 (CAN)
F3E2h			0000h	
F3E4h	MOFGPR63	XSFR	0000h	Регистр указателя FIFO/шлюза объекта сообщения 63 (CAN)
F3E6h			0000h	
F3E8h	MOIPR63	XSFR	0000h	Регистр указателя прерываний объекта сообщения 63 (CAN)
F3EAh			0000h	
F3ECh	MOAMR63	XSFR	0000h	Регистр маски объекта сообщения 63 (CAN)
F3EEh			0000h	
F3F0h	MODATAL63	XSFR	0000h	Младший регистр данных объекта сообщения 63 (CAN)
F3F2h			0000h	
F3F4h	MODATAH63	XSFR	0000h	Старший регистр данных объекта сообщения 63 (CAN)
F3F6h			0000h	
F3F8h	MOAR63	XSFR	0000h	Регистр арбитража объекта сообщения 63 (CAN)
F3FAh			0000h	
F3FCh	MOCTR63/ MOSTAT63	XSFR	0000h	Регистр управления/состояния объекта сообщения 63 (CAN)
F3FEh			3F3Eh	

Приложение В
(обязательное)
Таблица векторов прерываний

Таблица В.1 – Векторы прерываний

Номер прерывания	Локализация вектора	Источник прерывания	Регистр управления
1	2	3	4
00h	0000h	RESET	–
01h	0004h	–	–
02h	0008h	NMI (немаскируемое прерывание)	–
03h	000Ch	–	–
04h	0010h	Переполнение стека	–
05h	0014h	–	–
06h	0018h	Опустошение стека	–
07h	001Ch	–	–
08h	0020h	Программный останов	–
09h	0024h	–	–
0Ah	0028h	Прерывание класса В	–
0Bh	002Ch	–	–
0Ch	0030h	–	–
0Dh	0034h	–	–
0Eh	0038h	–	–
0Fh	003Ch	–	–
10h	0040h	Регистр сравнения 0 (CAPCOM1)	CC0IC
11h	0044h	Регистр сравнения 1 (CAPCOM1)	CC1IC
12h	0048h	Регистр сравнения 2 (CAPCOM1)	CC2IC
13h	004Ch	Регистр сравнения 3 (CAPCOM1)	CC3IC
14h	0050h	Регистр сравнения 4 (CAPCOM1)	CC4IC
15h	0054h	Регистр сравнения 5 (CAPCOM1)	CC5IC
16h	0058h	Регистр сравнения 6 (CAPCOM1)	CC6IC
17h	005Ch	Регистр сравнения 7 (CAPCOM1)	CC7IC
18h	0060h	Регистр сравнения 8 (CAPCOM1)	CC8IC
19h	0064h	Регистр сравнения 9 (CAPCOM1)	CC9IC
1Ah	0068h	Регистр сравнения 10 CAPCOM1)	CC10IC
1Bh	006Ch	Регистр сравнения 11 (CAPCOM1)	CC11IC
1Ch	0070h	Регистр сравнения 12 (CAPCOM1)	CC12IC
1Dh	0074h	Регистр сравнения 13 (CAPCOM1)	CC13IC
1Eh	0078h	Регистр сравнения 14 (CAPCOM1)	CC14IC
1Fh	007Ch	Регистр сравнения 15 (CAPCOM1)	CC15IC
20h	0080h	Таймер 0 (CAPCOM1)	T0IC
21h	0084h	Таймер 1 (CAPCOM1)	T1IC
22h	0088h	Таймер Т2 (GPT1, GPT2)	T2IC
23h	008Ch	Таймер Т3 (GPT1, GPT2)	T3IC
24h	0090h	Таймер Т4 (GPT1, GPT2)	T4IC
25h	0094h	Таймер Т5 (GPT1, GPT2)	T5IC
26h	0098h	Таймер Т6 (GPT1, GPT2)	T6IC
27h	009Ch	Захват/перезагрузка (GPT1, GPT2)	CRIC
28h	00A0h	Завершение преобразования АЦП	ASC_CIC
29h	00A4h	Ошибка АЦП	ADC_EIC
2Ah	00A8h	Передача (ASC0)	S0TIC

Продолжение таблицы В.1

1	2	3	4
2Bh	00ACh	Прием (ASC0)	S0RIC
2Ch	00B0h	Ошибка (ASC0)	S0EIC
2Dh	00B4h	Передача (SSC0)	SSC0TIC
2Eh	00B8h	Прием (SSC0)	SSC0RIC
2Fh	00BCh	Ошибка (SSC0)	SSC0EIC
30h	00C0h	Регистр сравнения 16 (CAPCOM2)	CC16IC
31h	00C4h	Регистр сравнения 17 (CAPCOM2)	CC17IC
32h	00C8h	Регистр сравнения 18 (CAPCOM2)	CC18IC
33h	00CCh	Регистр сравнения 19 (CAPCOM2)	CC19IC
34h	00D0h	Регистр сравнения 20 (CAPCOM2)	CC20IC
35h	00D4h	Регистр сравнения 21 (CAPCOM2)	CC21IC
36h	00D8h	Регистр сравнения 22 (CAPCOM2)	CC22IC
37h	00DCh	Регистр сравнения 23 (CAPCOM2)	CC23IC
38h	00E0h	Регистр сравнения 24 (CAPCOM2)	CC24IC
39h	00E4h	Регистр сравнения 25 (CAPCOM2)	CC25IC
3Ah	00E8h	Регистр сравнения 26 (CAPCOM2)	CC26IC
3Bh	00ECh	Регистр сравнения 27 (CAPCOM2)	CC27IC
3Ch	00F0h	Регистр сравнения 28 (CAPCOM2)	CC28IC
3Dh	00F4h	Таймер 7 (CAPCOM2)	T7IC
3Eh	00F8h	Таймер 8 (CAPCOM2)	T8IC
3Fh	00FCh	Программный запрос 15	IRQ15
40h	0100h	I2C	I2C_DIC
41h	0104h	Завершение преобразования АЦП2	ADC2_CIC
42h	0108h	Ошибка АЦП2	ADC2_EIC
43h	010Ch	Нахождение частоты PLL	PLLLOCK_IC
44h	0110h	Регистр сравнения 29 (CAPCOM2)	CC29IC
45h	0114h	Регистр сравнения 30 (CAPCOM2)	CC30IC
46h	0118h	Регистр сравнения 31 (CAPCOM2)	CC31IC
47h	011Ch	Буфер передатчика ASC0	S0TBIC
48h	0120h	Передача ASC1	S1TIC
49h	0124h	Прием ASC1	S1RIC
4Ah	0128h	Ошибка ASC1	S1EIC
4Bh	012Ch	Сторожевой таймер WDT	WDTIC
4Ch	0130h	Окончание PEC передач	EOPIC
4Dh	0134h	Таймер 12 (CAPCOM6)	CCU6_T12IC
4Eh	0138h	Таймер 13 (CAPCOM6)	CCU6_T13IC
4Fh	013Ch	Датчик (CAPCOM6)	CCU6_EIC
50h	0140h	CAPCOM6	CCU6_IC
51h	0144h	Передача SSC1	SSC1TIC
52h	0148h	Прием SSC1	SSC1RIC
53h	014Ch	Ошибка SSC1	SSC1EIC
54h	0150h	CAN0	CAN_0IC
55h	0154h	CAN1	CAN_1IC
56h	0158h	CAN2	CAN_2IC
57h	015Ch	CAN3	CAN_3IC
58h	0160h	Программный запрос 72	IRQ 72
59h	0164h	CAN4	CAN_4IC
5Ah	0168h	CAN5	CAN_5IC

Окончание таблицы В.1

1	2	3	4
5Bh	016Ch	CAN6	CAN_6IC
5Ch	0170h	CAN7	CAN_7IC
5Dh	0174h	RTC	RTC_IC
5Eh	0178h	Буфер передатчика ASC1	S1TBIC
5Fh	017Ch	Программный запрос 79	IRQ 79
60h	0180h	CAN8	CAN_8IC
61h	0184h	CAN9	CAN_9IC
62h	0188h	CAN10	CAN_10IC
63h	018Ch	CAN11	CAN_11IC
64h	0190h	CAN12	CAN_12IC
65h	0194h	CAN13	CAN_13IC
66h	0198h	CAN14	CAN_14IC
67h	019Ch	CAN15	CAN_15IC
68h	01A0h	Регистр сравнения RTCCMP1	RTC_IC1
69h	01A4h	Регистр сравнения RTCCMP2	RTC_IC2
6Ah	01A8h	Регистр сравнения RTCCMP3	RTC_IC3
6Bh	01Ach	Потеря частоты PLL	PLLUNLOCK_IC
6Ch	01B0h	Программный запрос 92	IRQ 92
6Dh	01B4h	Программный запрос 93	IRQ 93
6Eh	01B8h	Программный запрос 94	IRQ 94
6Fh	01BCh	Программный запрос 95	IRQ 95
70h	01C0h	Программный запрос 96	IRQ 96
71h	01C4h	Программный запрос 97	IRQ 97
72h	01C8h	Программный запрос 98	IRQ 98
73h	01CCh	Программный запрос 99	IRQ 99
74h	01D0h	Программный запрос 100	IRQ 100
75h	01D4h	Программный запрос 101	IRQ 101
76h	01D8h	Программный запрос 102	IRQ 102
77h	01DCh	Программный запрос 103	IRQ 103
78h	01E0h	Программный запрос 104	IRQ 104
79h	01E4h	Программный запрос 105	IRQ 105
7Ah	01E8h	Программный запрос 106	IRQ 106
7Bh	01ECh	Программный запрос 107	IRQ 107
7Ch	01F0h	Программный запрос 108	IRQ 108
7Dh	01F4h	Программный запрос 109	IRQ 109
7Eh	01F8h	Программный запрос 110	IRQ 110
7Fh	01FCh	Программный запрос 111	IRQ 111

Приложение Г
(обязательное)
Система команд микроконтроллера 1887ВЕ6Т

Условные обозначения (для таблиц Г.1 – Г.13):

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 бит-адресуемом пространстве памяти.
bitoff – Указатель слова в бит-адресуемом 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.

В таблицах Г.1 – Г.13 представлены команды и режимы адресации.

Таблица Г.1 – 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	Rwm #data4 [Rwm] [Rwm+] Rwn Rwn [Rwm] [Rwm] [Rwm +] #data16 [Rwm+#d16] Rwn mem [Rwn] mem	* * * * * * ** * * 	2 2 2 2 2 2 2 2 2 4 4 4 4 4 4

Продолжение таблицы Г.1

1	2			3
MOV[B]	mem	reg		4
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

Окончание таблицы Г.1

1	2	3
ATOMIC EXTR	#irang2	2
EXTP EXTPR	Rwm #irag2 #pag #irag2	2 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 Г.2 – 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 Г.2

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 Г.2

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 Г.3 – 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 Г.3

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 Г.3

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 Г.4 – 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 Г.4

1	2	3	4
BCMP	bitaddr, bitaddr	4	Сравнение бита с битом
BFLDH	bitoff, #mask8, #data8	4	Побитовая модификация маскированных битов старшего байта бит-адресуемого слова в памяти с непосредственным операндом
BFLDL	bitoff, #mask8, #data8	4	Побитовая модификация маскированных битов младшего байта бит-адресуемого 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 Г.5 – 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 Г.5

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 Г.6 – 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 Г.7 – 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 Г.8 – 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 Г.8

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 Г.9 – 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 Г.9

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 Г.10 – 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 Г.11 – 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 Г.12 – 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 Г.12

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 Г.13 – Описание вспомогательной команды

Мнемокод команды	Операнды	Размер, байт	Описание команды
NOP	–		Нет операции

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 также 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 операндами;

(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 Г.14 представлены возможные коды состояний (условий перехода), которые могут использоваться командами вызова подпрограмм и переходов.

Таблица Г.14 – Коды условий перехода (cc)

Мнемокод	Условие	Описание	Номер
cc_UC	1 = 1	Безусловный переход	0h
cc_Z	Z = 1	Если результат равен нулю	2h
cc_NZ	Z = 0	Если результат не равен нулю	3h
cc_V	V = 1	Если произошло переполнение во время выполнения команды	4h
cc_NV	V = 0	Если не произошло переполнения во время выполнения команды	5h
cc_N	N = 1	Если результат отрицательный	6h
cc_NN	N = 0	Если результат неотрицательный	7h
cc_C	C = 1	Если произошел перенос во время выполнения инструкции	8h
cc_NC	C = 0	Если не произошел перенос во время выполнения инструкции	9h
cc_EQ	Z = 1	Если сравниваемые операнды эквивалентны	2h
cc_NE	Z = 0	Если сравниваемые операнды не эквивалентны	3h
cc ULT	C = 1	Меньше (без знака)	8h
cc_ULE	(Z v C) = 1	Меньше или равно (без знака)	Fh
cc_UGE	C = 0	Больше или равно (без знака)	9h
cc_UGT	(Z v C) = 0	Больше (без знака)	Eh
cc_SLT	(N ⊕ V) = 1	Меньше (со знаком)	Ch
cc_SLE	(Z v (N ⊕ V)) = 1	Меньше или равно (со знаком)	Bh
cc_SGE	(N ⊕ V) = 0	Больше или равно (со знаком)	Dh
cc_SGT	(Z v (N ⊕ V)) = 0	Больше (со знаком)	Ah
cc_NET	(Z v E) = 0	Если сравниваемые операнды не эквивалентны и один из операндов не равен наименьшему отрицательному числу	1h

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 (8000h для данных типа WORD и 80h 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 неадресуемый байт очищается, а адресуемый (в который производится 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 Г.1 показано соотношение между форматом команды и соответствующей ей внутренней организацией (N = полубайт = 4 бита).

Для описания формата команд используются следующие символы:

00h – FFh – 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 – незначимый;
@@	– 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 1887ВЕ6Т – двух- и четырехбайтные.

Формат команды микроконтроллера 1887ВЕ6Т изображен на рисунке Г.1.

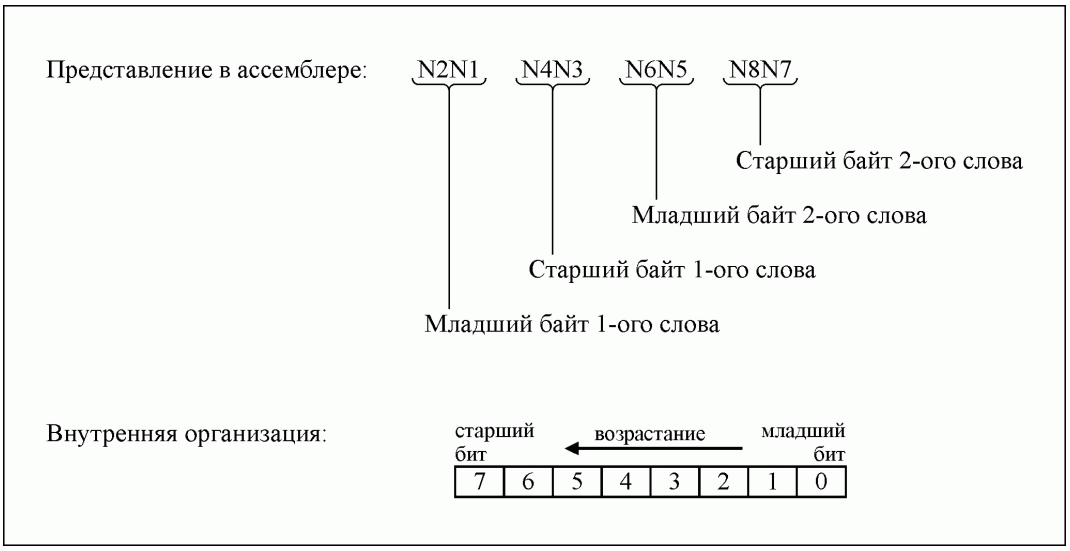


Рисунок Г.1 – Формат команды микроконтроллера 1887ВЕ6Т

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) $\Leftrightarrow$ (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) $\Leftrightarrow$ (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 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) ⇔ (op2)
(opl) ← (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

Действие

```
(count) ← op1 [ 1 ≤ op ≤ 4 ]
IRQ_Processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Segment = (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 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:l l 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 Д
(обязательное)
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 8000000000h. 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⊗]	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, [$Rw_m\otimes$]	83 nm D0 rrrr:rqqq	Да
CoMAC	[$IDX_i\otimes$], [$Rw_m\otimes$]	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
((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 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
((IDXi (- ⊗))) ← ((IDXi))
```

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	[IDXi⊗], [Rwm⊗]	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, [$Rw_m\otimes$], rnd	83 nm 01 0:0qqq	Нет
CoMULu	[$IDX_i\otimes$], [$Rw_m\otimes$], 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⊗], 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
CoSUB	[IDX_i⊗], [Rw_m⊗]	93 Xm 0A 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
CoSUB	[IDX_i⊗], [Rw_m⊗]	93	Xm	0A	rrrr:rqqq	Да

Приложение Е
(обязательное)
Коды состояний функционирования модуля I2C

В таблицах Е.1 – Е.11 представлена информация о соответствии кодов и операций.

Условные обозначения, принятые в таблицах:

- [ADR, 0], [ADR, 1] – 8-разрядное значение, состоящее из 7-разрядного адреса ADR и бита направления передачи R/W#, значение которого «0» или «1» указывается непосредственно;
- DAT – байт данных;
- код мастера – 8-разрядное значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств;
- «с ACK» – выражение, обозначающее, что после передачи адреса/байта в ответ на запрос подтверждения передачи (бит ACK) передатчик получает подтверждение передачи от ведомого (квитирование);
- «с NACK» – выражение, обозначающее, что после передачи адреса/байта в ответ на запрос подтверждения передачи (бит ACK) передатчик получает неподтверждение передачи от ведомого (неквитирование);
- X – бит может быть установленным (1b) или сброшенным (0b), в зависимости от режима работы, состояния и дальнейших действий модуля I2C.

Таблица Е.1 – Исключительные состояния

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
00h	IDLE	–	–	–	–	–	Ожидать завершения текущей передачи байта
1Fh	Ошибка на шине	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)

Таблица Е.2 – Режим FS мастера передатчика (дополнительно см. таблицу Е.4)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
01h	Старт	Код мастера	1	0	0	0	Передать код мастера и перейти в режим HS (0Ch/ 21h)
		[ADR, 0]					Передать адрес ведомого (04h/ 05h)
02h	Повторный Старт	[ADR, 0]	1	0	0	0	Передать адрес ведомого (04h/ 05h)
		[ADR, 1]					Передать адрес ведомого, после чего перейти в режим приемника (08h/ 09h)

Окончание таблицы Е2

1	2	3	4	5	6	7	8
03h	Потеря арбитража, мастер перешел в режим безадресного ведомого	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
04h	Отправлен адрес ведомого с ACK	DAT	1	0	0	0	Передать байт данных (06h/ 07h)
		–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
05h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
06h	Отправлен байт данных с ACK	DAT	1	0	0	0	Передать байт данных (06h/ 07h)
		–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
07h	Отправлен байт данных с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Е.3 – Режим FS мастера приемника

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
08h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)
09h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
0Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)

Окончание таблицы Е.3

1	2	3	4	5	6	7	8
0Bh	Принят байт данных и не квити-рован	DAT	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Е.4 – Режим FS мастера передатчика (дополнительно см. таблицу Е.2)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
0Ch	Отправлен код мастера, обнаружена ошибка (ACK)	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Е.5 – Режим FS ведомого приемника (дополнительно см. таблицу Е.7)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
10h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
11h	Принят адрес после потери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
12h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
13h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Е.6 – Режим FS ведомого передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
14h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
15h	Принят адрес после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
16h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
17h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
18h	Принят адрес отклика и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)
19h	Принят адрес отклика после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)
1Ah	Отправлен байт данных в ответ на получение адреса отклика с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)

Окончание таблицы Е.6

1	2	3	4	5	6	7	8
1Bh	Отправлен байт данных в ответ на получение адреса отклика с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Е.7 – Режим FS ведомого приемника (дополнительно см. таблицу Е.5)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
1Ch	Стоп	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
1Dh	Принят адрес общего вызова и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
1Eh	Принят адрес общего вызова после потери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)

Таблица Е.8 – Режим HS мастера передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
21h	Успешно отправлен код мастера, мастер перешел в режим HS	–	1	0	0	1	Сделать повторный старт (22h)
22h	Повторный старт	[ADR, 0]	1	0	0	0	Передать адрес ведомого (28h/29h)
		[ADR, 1]					Передать адрес ведомого, после квитирования/не квитирования переключиться в режим мастера приемника (28h/29h)
23h	Потеря арбитража, мастер перешел в режим HS безадресного ведомого	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
24h	Отправлен адрес ведомого с ACK	DAT	1	0	0	0	Передать байт данных (26h/27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
25h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
26h	Отправлен байт данных с ACK	DAT	1	0	0	0	Передать байт данных (26h/27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
27h	Отправлен байт данных с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица E.9 – Режим HS мастера приемника

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
28h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
29h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
2Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
2Bh	Принят байт данных и не квити-рован	DAT	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица E.10 – Режим HS ведомого приемника

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
30h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
32h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
33h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Е.11 – Режим HS ведомого передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
34h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (36h/37h)
36h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (36h/37h)
37h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Приложение Ж
(обязательное)
Стартовая конфигурация микроконтроллера 1887ВЕ6Т

Для конфигурирования микроконтроллера при внешнем сбросе используются выводы порта P0 и вывод EA#, состояние которых должно быть установлено во время активного уровня сигнала сброса RSTIN# (не менее 1 мкс). 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 Ж.1 представлены выводы порта P0 и регистры, состояние которых они изменяют.

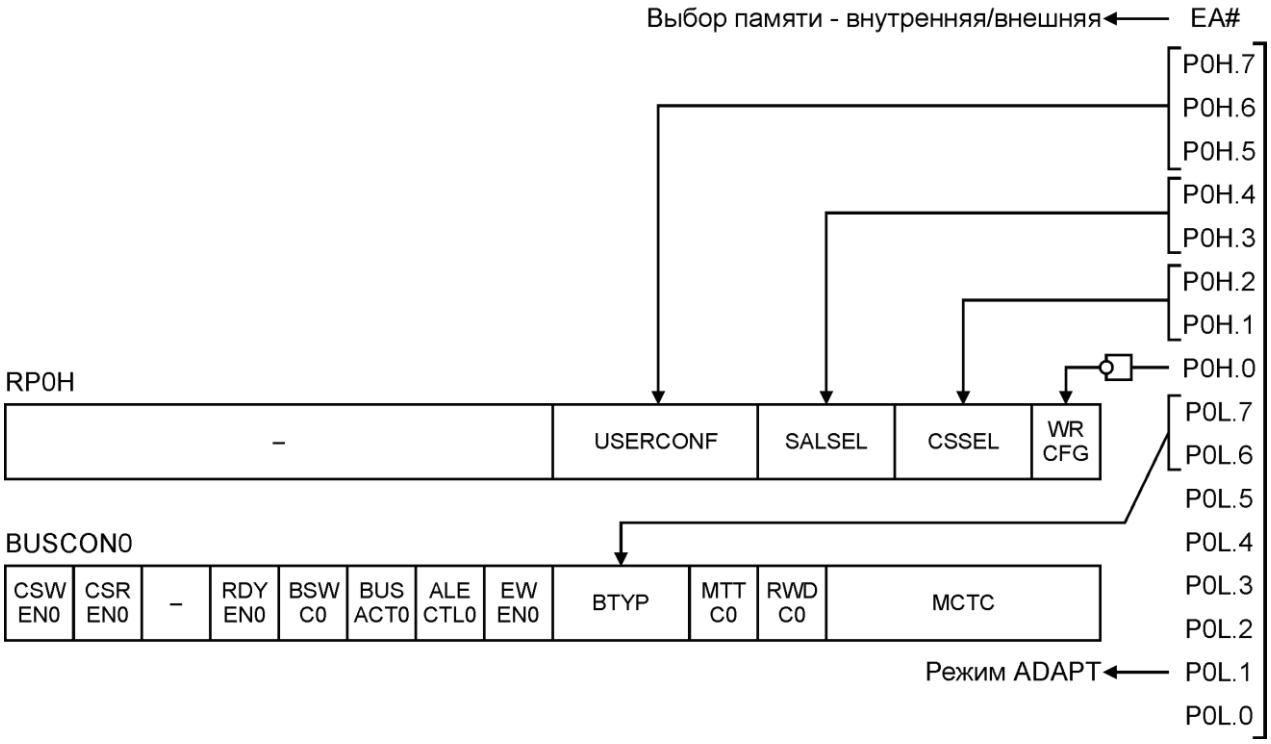


Рисунок Ж.1 – Порт P0 и его взаимодействие с регистрами микроконтроллера при конфигурировании

В таблице Ж.1 указано назначение выводов порта P0 и вывода EA# при конфигурировании микроконтроллера.

Таблица Ж.1 – Функциональное назначение выводов микроконтроллера при конфигурировании

Обозначение вывода	Функциональное назначение выводов при конфигурации микроконтроллера
1	2
P0H.7, P0H.6, P0H.5,	С выводов P0H.7 – P0H.5 считывается код, который записывается в битовое поле USERCONF регистра RP0H. Этот код не имеет функционального назначения

Окончание таблицы Ж.1

<table>
<tr><th>1</th><th>2</th></tr>
<tr><td>P0H.4,
P0H.3</td><td>С выводов P0H.4, P0H.3 считывается код, определяющий разрядность старшей части адреса. 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>С выводов P0H.2, P0H.1 считывается код, определяющий количество каналов CS#. 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>
<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>16-разрядная мультиплексная шина</td></tr>
</table></td></tr>
<tr><td>P0L.5–P0L.2,
P0L.0</td><td>Не участвуют в конфигурации</td></tr>
<tr><td>P0L.1</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>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	-	671			21.01.14
1	-	87	-	-	-			17.04.14
2	-	19, 179, 355, 386, 388, 389, 390, 461	-	-	-			05.05.14