МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1887ВЕ9Т

Руководство пользователя

2015

Содержание

Введение

В настоящем руководстве КФДЛ.431295.052 представлен обзор архитектуры и функционального построения микросхемы 1887ВЕ9Т, приведена система команд и указаны особенности применения.

Микросхема представляет собой СБИС 16-разрядного RISC микроконтроллера архитектуры С166 с тактовой частотой до 80 МГц и встроенными ОЗУ и ПЗУ. В состав микросхемы входят модули захвата, обработки и аналого-цифрового преобразования сигналов, блок кодирования/ декодирования информации, таймеры-счетчики, часы реального времени, генераторы ШИМ-сигналов, система отладки и др. Периферия представлена последовательными асинхронными и синхронными приемопередатчиками, квадратурным декодером и контроллерами интерфейсов CAN, I2C, SPI, ГОСТ Р 52070-2003.

Встроенный специализированный сопроцессор для улучшения выполнения алгоритмов обработки сигналов в комбинации с интегрированной интеллектуальной периферией со сниженной необходимостью вмешательства ЦПУ обеспечивают высокую производительность, гибкость и универсальность.

Эффективное программирование микросхемы 1887ВЕ9Т достигается благодаря мощной системе команд, поддерживающей вычисления над 8-, 16- и 32-разрядными операндами, операции умножения и деления, контроль границ стека, управление периферией через регистры специальных функций. Следует также отметить высокую пропускную способность, мощную систему адресации и поддержку программирования на языке высокого уровня.

Настоящее руководство может служить практическим пособием по применению микроконтроллера для разработчиков систем на основе ИС 1887ВЕ9Т и программистов.

.

1 Назначение и области применения

Микросхемы 1887ВЕ9Т предназначены для работы во встроенных системах реального времени и имеют малое время переключения между выполняемыми задачами. Наличие разнообразных периферийных модулей делает возможным применение микросхем для решения сложных задач управления.

Входящие в состав микросхем блоки захвата/сравнения и генерации ШИМ-сигналов с программируемым интервалом «мертвого времени», квадратурный декодер и модуль обработки сигналов от датчиков Холла позволяют использовать их в системах управления приводами двигателей постоянного и переменного тока. Внешний аварийный TRAP-сигнал для быстрого отключения выводов обеспечивает высокий уровень безопасности систем в целом.

Поддержка нескольких протоколов последовательного асинхронного и синхронного обмена данными позволяет организовывать системы, состоящие из нескольких микроконтроллеров и/или различных по принципу действия и скорости работы устройств.

Наличие контроллера магистрального последовательного интерфейса электронных модулей (ГОСТ Р 52070-2003) и контроллера интерфейса, реализующего последовательный протокол связи CAN, определяет перспективу широкого применения микросхем в таких областях, как автомобильный, железнодорожный и морской транспорт, промышленная автоматика, авиация, системы доступа и контроля.

2 Технические характеристики и параметры микросхемы

В состав микроконтроллера входят:

- 16-разрядное ЦПУ с пятиуровневым конвейером команд и тактовой частотой до 80 МГц, включающее в свой состав контроллер прерываний, блок управления сбросом, сторожевой таймер и блок контроля пониженного энергопотребления (три режима);
- объем адресуемой памяти 16 Мбайт;
- флеш-память программ (Flash) объемом 512 Кбайт;
- ОЗУ программ/данных (PSRAM) объемом 16 Кбайт;
- двухпортовое ОЗУ (DPRAM) объемом 3 Кбайт с интерфейсом прямого доступа к памяти;
- ОЗУ данных X-периферии (XRAM) объемом 16 Кбайт;
- синтезатор частоты на основе ФАПЧ (PLL);
- часы реального времени (RTC);
- 12-разрядный 16 канальный АЦП;
- блок кодирования по ГОСТ 28147-89;
- квадратурный декодер;
- пять таймеров общего назначения, объединенные в два блока (GPT1 и GPT2);
- два 16-канальных блока захвата/сравнения (CAPCOM1 и CAPCOM2);
- 3-канальный блок ШИМ и 1-канальный блок сравнения (CAPCOM6);
- два асинхронных последовательных канала передачи данных (ASC0 и ASC1);
- два синхронных последовательных канала передачи данных (SSC0 и SSC1);
- асинхронно-синхронный приемопередатчик (USART2);
- контроллеры последовательных интерфейсов:
    - SPI (один порт);
    - I2C (один порт);
    - CAN (протокол 2.0B, четыре порта);
    - ГОСТ Р 52070-2003 (два порта, совместимость со стандартом MIL-STD-1553B);
- по четыре 8- и 16-разрядных параллельных порта ввода-вывода и один 7-разрядный параллельный порт;
- система отладки с отладочным интерфейсом JTAG.

2.1 Конструктивные характеристики микросхемы

Кристалл микросхемы выполнен по КМОП технологии. Микросхема разработана в металлокерамическом корпусе 4248.144-1 с четырехсторонним планарным расположением выводов.

2.2 Функциональные параметры

Условное графическое обозначение микроконтроллера приведено на рисунке 2.1.

В таблице 2.1 указано функциональное назначение выводов микроконтроллера, имеющих альтернативные функции – в графе «Обозначение вывода» первым указывается обозначение основной функции вывода, далее – обозначения альтернативных функций. В таблице 2.2 указано функциональное назначение выводов микроконтроллера без альтернативных функций.

В таблицах 2.1 и 2.2 в графе «Номер вывода» указывается номер вывода корпуса микроконтроллера, в графе «Тип вывода» используются обозначения: I – вход, O – выход, Z – третье состояние, 2 – режим открытого стока.

После сброса микроконтроллера все выводы конфигурируются как выводы общего назначения.

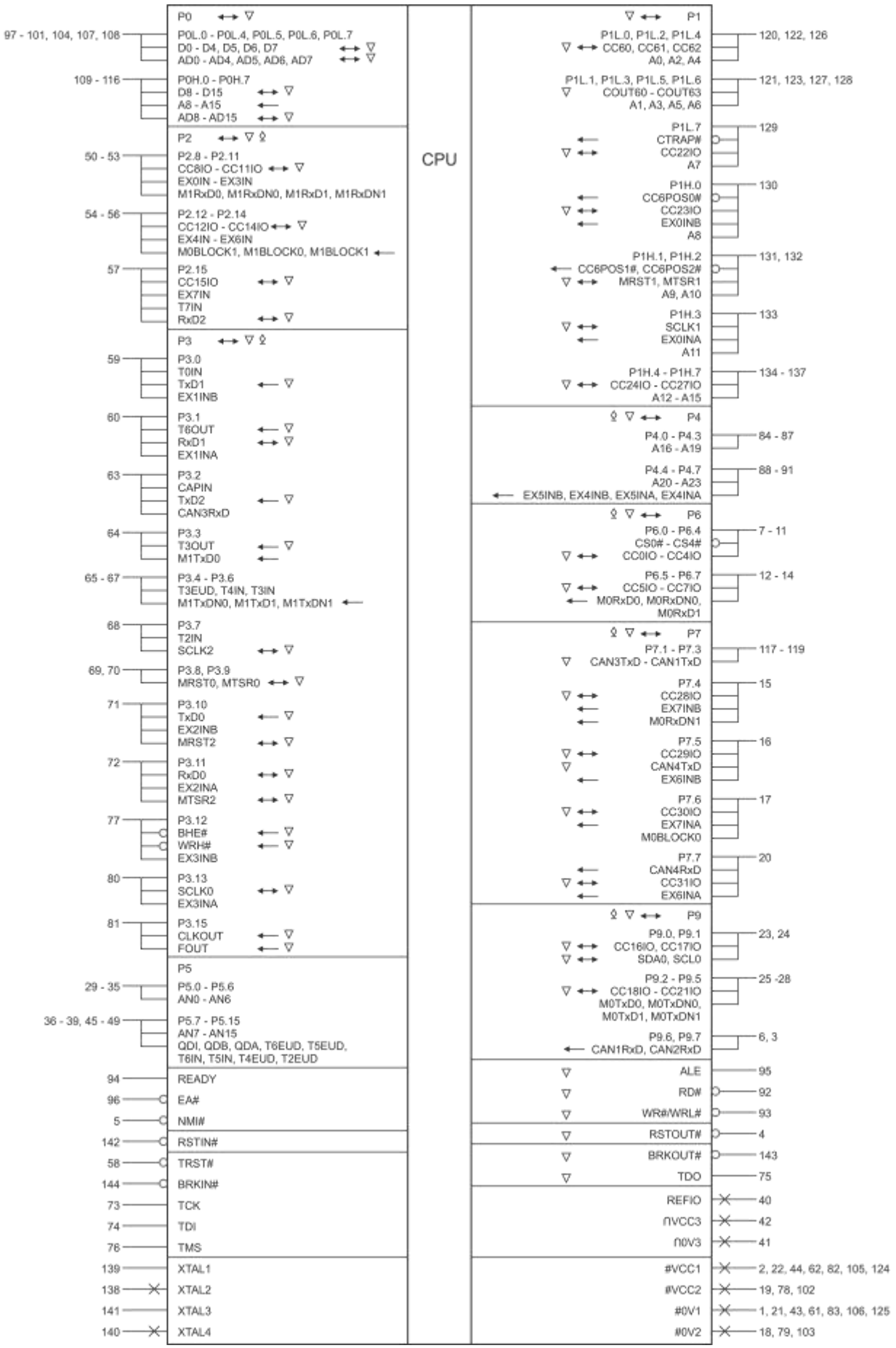


Рисунок 2.1 – Условное графическое изображение микросхемы 1887ВЕ9Т

Таблица 2.1 – Функциональное назначение выводов с альтернативными функциями

Обозначение вывода		Номер вывода	Тип вывода	Функциональное назначение вывода
1		2	3	4
P0L.0		97	I/O/Z	Вход/выход «порт 0, младший байт», 0 разряд
	D0		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 0 разряд
	AD0		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 0 разряд
P0L.1		98	I/O/Z	Вход/выход «порт 0, младший байт», 1 разряд
	D1		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 1 разряд
	AD1		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 1 разряд
P0L.2		99	I/O/Z	Вход/выход «порт 0, младший байт», 2 разряд
	D2		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 2 разряд
	AD2		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 2 разряд
P0L.3		100	I/O/Z	Вход/выход «порт 0, младший байт», 3 разряд
	D3		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 3 разряд
	AD3		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 3 разряд
P0L.4		101	I/O/Z	Вход/выход «порт 0, младший байт», 4 разряд
	D4		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 4 разряд
	AD4		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 4 разряд
P0L.5		104	I/O/Z	Вход/выход «порт 0, младший байт», 5 разряд
	D5		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 5 разряд
	AD5		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 5 разряд
P0L.6		107	I/O/Z	Вход/выход «порт 0, младший байт», 6 разряд
	D6		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 6 разряд
	AD6		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 6 разряд
P0L.7		108	I/O/Z	Вход/выход «порт 0, младший байт», 7 разряд
	D7		I/O/Z	Вход/выход демультиплексированной 8/16-разрядной шины данных, 7 разряд
	AD7		I/O/Z	Вход/выход мультиплексированной 8-разрядной шины адреса/данных, 7 разряд
P0H.0		109	I/O/Z	Вход/выход «порт 0, старший байт», 0 разряд
	D8		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 8 разряд
	A8		O	Выход мультиплексированной 16-разрядной шины адреса, 8 разряд
	AD8		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 8 разряд

Продолжение таблицы 2.1

1		2	3	4
P0H.1		110	I/O/Z	Вход/выход «порт 0, старший байт», 1 разряд
	D9		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 9 разряд
	A9		O	Выход мультиплексированной 16-разрядной шины адреса, 9 разряд
	AD9		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 9 разряд
P0H.2		111	I/O/Z	Вход/выход «порт 0, старший байт», 2 разряд
	D10		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 10 разряд
	A10		O	Выход мультиплексированной 16-разрядной шины адреса, 10 разряд
	AD10		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 10 разряд
P0H.3		112	I/O/Z	Вход/выход «порт 0, старший байт», 3 разряд
	D11		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 11 разряд
	A11		O	Выход мультиплексированной 16-разрядной шины адреса, 11 разряд
	AD11		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 11 разряд
P0H.4		113	I/O/Z	Вход/выход «порт 0, старший байт», 4 разряд
	D12		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 12 разряд
	A12		O	Выход мультиплексированной 16-разрядной шины адреса, 12 разряд
	AD12		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 12 разряд
P0H.5		114	I/O/Z	Вход/выход «порт 0, старший байт», 5 разряд
	D13		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 13 разряд
	A13		O	Выход мультиплексированной 16-разрядной шины адреса, 13 разряд
	AD13		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 13 разряд
P0H.6		115	I/O/Z	Вход/выход «порт 0, старший байт», 6 разряд
	D14		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 14 разряд
	A14		O	Выход мультиплексированной 16-разрядной шины адреса, 14 разряд
	AD14		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 14 разряд
P0H.7		116	I/O/Z	Вход/выход «порт 0, старший байт», 7 разряд
	D15		I/O/Z	Вход/выход демультиплексированной 16-разрядной шины данных, 15 разряд
	A15		O	Выход мультиплексированной 16-разрядной шины адреса, 15 разряд
	AD15		I/O/Z	Вход/выход мультиплексированной 16-разрядной шины адреса/данных, 15 разряд

Продолжение таблицы 2.1

1		2	3	4
P1L.0		120	I/O/Z	Вход/выход «порт 1, младший байт», 0 разряд
	CC60		I/O/Z	Вход/выход блока CAPCOM6, 0 канал
	A0		O	Выход демультиплексированной 8/16-разрядной шины адреса, 0 разряд
P1L.1		121	I/O/Z	Вход/выход «порт 1, младший байт», 1 разряд
	COUT60		O/Z	Выход блока CAPCOM6, 0 канал
	A1		O	Выход демультиплексированной 8/16-разрядной шины адреса, 1 разряд
P1L.2		122	I/O/Z	Вход/выход «порт 1, младший байт», 2 разряд
	CC61		I/O/Z	Вход/выход блока CAPCOM6, 1 канал
	A2		O	Выход демультиплексированной 8/16-разрядной шины адреса, 2 разряд
P1L.3		123	I/O/Z	Вход/выход «порт 1, младший байт», 3 разряд
	COUT61		O/Z	Выход блока CAPCOM6, 1 канал
	A3		O	Выход демультиплексированной 8/16-разрядной шины адреса, 3 разряд
P1L.4		126	I/O/Z	Вход/выход «порт 1, младший байт», 4 разряд
	CC62		I/O/Z	Вход/выход блока CAPCOM6, 2 канал
	A4		O	Выход демультиплексированной 8/16-разрядной шины адреса, 4 разряд
P1L.5		127	I/O/Z	Вход/выход «порт 1, младший байт», 5 разряд
	COUT62		O/Z	Выход блока CAPCOM6, 2 канал
	A5		O	Выход демультиплексированной 8/16-разрядной шины адреса, 5 разряд
P1L.6		128	I/O/Z	Вход/выход «порт 1, младший байт», 6 разряд
	COUT63		O/Z	Выход блока CAPCOM6, 3 канал
	A6		O	Выход демультиплексированной 8/16-разрядной шины адреса, 6 разряд
P1L.7		129	I/O/Z	Вход/выход «порт 1, младший байт», 7 разряд
	CTRAP#		I	Вход «системное прерывание» блока CAPCOM6
	CC22IO		I/O/Z	Вход/выход блока CAPCOM2, 22 канал
	A7		O	Выход демультиплексированной 8/16-разрядной шины адреса, 7 разряд
P1H.0		130	I/O/Z	Вход/выход «порт 1, старший байт», 0 разряд
	CC6POS0#		I	Вход «датчик 0» блока CAPCOM6
	CC23IO		I/O/Z	Вход/выход блока CAPCOM2, 23 канал
	EX0INB		I	Вход альтернативного сигнала B прерывания 0
	A8		O	Выход демультиплексированной 8/16-разрядной шины адреса, 8 разряд
P1H.1		131	I/O/Z	Вход/выход «порт 1, старший байт», 1 разряд
	CC6POS1#		I	Вход «датчик 1» блока CAPCOM6
	MRST1		I/O/Z	Вход/выход «данные» блока SSC1
	A9		O	Выход демультиплексированной 8/16-разрядной шины адреса, 9 разряд

Продолжение таблицы 2.1

1		2	3	4
P1H.2		132	I/O/Z	Вход/выход «порт 1, старший байт», 2 разряд
	CC6POS2#		I	Вход «датчик 2» блока CAPCOM6
	MTSR1		I/O/Z	Вход/выход «данные» блока SSC1
	A10		O	Выход демультиплексированной 8/16-разрядной шины адреса, 10 разряд
P1H.3		133	I/O/Z	Вход/выход «порт 1, старший байт», 3 разряд
	SCLK1		I/O/Z	Вход/выход «тактовый сигнал» блока SSC1
	EX0INA		I	Вход альтернативного сигнала A прерывания 0
	A11		O	Выход демультиплексированной 8/16-разрядной шины адреса, 11 разряд
P1H.4		134	I/O/Z	Вход/выход «порт 1, старший байт», 4 разряд
	CC24IO		I/O/Z	Вход/выход блока CAPCOM2, 24 канал
	A12		O	Выход демультиплексированной 8/16-разрядной шины адреса, 12 разряд
P1H.5		135	I/O/Z	Вход/выход «порт 1, старший байт», 5 разряд
	CC25IO		I/O/Z	Вход/выход блока CAPCOM2, 25 канал
	A13		O	Выход демультиплексированной 8/16-разрядной шины адреса, 13 разряд
P1H.6		136	I/O/Z	Вход/выход «порт 1, старший байт», 6 разряд
	CC26IO		I/O/Z	Вход/выход блока CAPCOM2, 26 канал
	A14		O	Выход демультиплексированной 8/16-разрядной шины адреса, 14 разряд
P1H.7		137	I/O/Z	Вход/выход «порт 1, старший байт», 7 разряд
	CC27IO		I/O/Z	Вход/выход блока CAPCOM2, 27 канал
	A15		O	Выход демультиплексированной 8/16-разрядной шины адреса, 15 разряд
P2.8		50	I/O/Z/2	Вход/выход «порт 2», 8 разряд
	CC8IO		I/O/Z	Вход/выход блока CAPCOM1, 8 канал
	EX0IN		I	Вход «внешнее прерывание 0»
	M1RxD0		I	Прямой вход 0 канала 1 блока ГОСТ Р 52070-2003
P2.9		51	I/O/Z/2	Вход/выход «порт 2», 9 разряд
	CC9IO		I/O/Z	Вход/выход блока CAPCOM1, 9 канал
	EX1IN		I	Вход «внешнее прерывание 1»
	M1RxDN0		I	Инверсный вход 0 канала 1 блока ГОСТ Р 52070-2003
P2.10		52	I/O/Z/2	Вход/выход «порт 2», 10 разряд
	CC10IO		I/O/Z	Вход/выход блока CAPCOM1, 10 канал
	EX2IN		I	Вход «внешнее прерывание 2»
	M1RxD1		I	Прямой вход 1 канала 1 блока ГОСТ Р 52070-2003
P2.11		53	I/O/Z/2	Вход/выход «порт 2», 11 разряд
	CC11IO		I/O/Z	Вход/выход блока CAPCOM1, 11 канал
	EX3IN		I	Вход «внешнее прерывание 3»
	M1RxDN1		I	Инверсный вход 1 канала 1 блока ГОСТ Р 52070-2003
P2.12		54	I/O/Z/2	Вход/выход «порт 2», 12 разряд
	CC12IO		I/O/Z	Вход/выход блока CAPCOM1, 12 канал
	EX4IN		I	Вход «внешнее прерывание 4»
	M0BLOCK1		O	Выход блокировки 1 канала 0 блока ГОСТ Р 52070-2003

Продолжение таблицы 2.1

1		2	3	4
P2.13		55	I/O/Z/2	Вход/выход «порт 2», 13 разряд
	CC13IO		I/O/Z	Вход/выход блока CAPCOM1, 13 канал
	EX5IN		I	Вход «внешнее прерывание 5»
	M1BLOCK0		O	Выход блокировки 0 канала 1 блока ГОСТ Р 52070-2003
P2.14		56	I/O/Z/2	Вход/выход «порт 2», 14 разряд
	CC14IO		I/O/Z	Вход/выход блока CAPCOM1, 14 канал
	EX6IN		I	Вход «внешнее прерывание 6»
	M1BLOCK1		O	Выход блокировки 1 канала 1 блока ГОСТ Р 52070-2003
P2.15		57	I/O/Z/2	Вход/выход «порт 2», 15 разряд
	CC15IO		I/O/Z	Вход/выход блока CAPCOM1, 15 канал
	EX7IN		I	Вход «внешнее прерывание 7»
	T7IN		I	Вход «синхронизация» таймера T7
	RxD2		I/O/Z	Вход/выход блока USART2
P3.0		59	I/O/Z/2	Вход/выход «порт 3», 0 разряд
	T0IN		I	Вход «синхронизация» таймера T0
	TxD1		O/Z	Выход блока ASC1
	EX1INB		I	Вход альтернативного сигнала B прерывания 1
P3.1		60	I/O/Z/2	Вход/выход «порт 3», 1 разряд
	T6OUT		O/Z	Выход «триггер» таймера T6
	RxD1		I/O/Z	Вход/выход блока ASC1
	EX1INA		I	Вход альтернативного сигнала A прерывания 1
P3.2		63	I/O/Z/2	Вход/выход «порт 3», 2 разряд
	CAPIN		I	Вход «захват» таймера T5
	TxD2		O/Z	Выход блока USART2
	CAN3RxD		I	Вход узла CAN3
P3.3		64	I/O/Z/2	Вход/выход «порт 3», 3 разряд
	T3OUT		O/Z	Выход «триггер» таймера T3
	M1TxD0		O	Прямой выход 0 канала 1 блока ГОСТ Р 52070-2003
P3.4		65	I/O/Z/2	Вход/выход «порт 3», 4 разряд
	T3EUD		I	Вход «направление счета» таймера T3
	M1TxDN0		O	Инверсный выход 0 канала 1 блока ГОСТ Р 52070-2003
P3.5		66	I/O/Z/2	Вход/выход «порт 3», 5 разряд
	T4IN		I	Вход «синхронизация» таймера T4
	M1TxD1		O	Прямой выход 1 канала 1 блока ГОСТ Р 52070-2003
P3.6		67	I/O/Z/2	Вход/выход «порт 3», 6 разряд
	T3IN		I	Вход «синхронизация» таймера T3
	M1TxDN1		O	Инверсный выход 1 канала 1 блока ГОСТ Р 52070-2003
P3.7		68	I/O/Z/2	Вход/выход «порт 3», 7 разряд
	T2IN		I	Вход «синхронизация» таймера T2
	SCLK2		I/O/Z	Вход/выход «тактовый сигнал» блока SPI
P3.8		69	I/O/Z/2	Вход/выход «порт 3», 8 разряд
	MRST0		I/O/Z	Вход/выход «данные» блока SSC0
P3.9		70	I/O/Z/2	Вход/выход «порт 3», 9 разряд
	MTSR0		I/O/Z	Вход/выход «данные» блока SSC0
P3.10		71	I/O/Z/2	Вход/выход «порт 3», 10 разряд
	TxD0		O/Z	Выход блока ASC0
	EX2INB		I	Вход альтернативного сигнала B прерывания 2
	MRST2		I/O/Z	Вход/выход «данные» блока SPI

Продолжение таблицы 2.1

1		2	3	4
P3.11		72	I/O/Z/2	Вход/выход «порт 3», 11 разряд
	RxD0		I/O/Z	Вход/выход «данные» блока ASC0
	EX2INA		I	Вход альтернативного сигнала A прерывания 2
	MTSR2		I/O/Z	Вход/выход «данные» блока SPI
P3.12		77	I/O/Z/2	Вход/выход «порт 3», 12 разряд
	BHE#		O/Z	Выход «выбор старшего байта»
	WRH#		O/Z	Выход «запись старшего байта»
	EX3INB		I	Вход альтернативного сигнала B прерывания 3
P3.13		80	I/O/Z/2	Вход/выход «порт 3», 13 разряд
	SCLK0		I/O/Z	Вход/выход «тактовый сигнал» блока SSC0
	EX3INA		I	Вход альтернативного сигнала A прерывания 3
P3.15		81	I/O/Z/2	Вход/выход «порт 3», 15 разряд
	CLKOUT		O/Z	Выход «системный тактовый сигнал»
	FOUT		O/Z	Выход «программируемая частота»
P4.0		84	I/O/Z/2	Вход/выход «порт 4», 0 разряд
	A16		O	Выход шины адреса, 16 разряд
P4.1		85	I/O/Z/2	Вход/выход «порт 4», 1 разряд
	A17		O	Выход шины адреса, 17 разряд
P4.2		86	I/O/Z/2	Вход/выход «порт 4», 2 разряд
	A18		O	Выход шины адреса, 18 разряд
P4.3		87	I/O/Z/2	Вход/выход «порт 4», 3 разряд
	A19		O	Выход шины адреса, 19 разряд
P4.4		88	I/O/Z/2	Вход/выход «порт 4», 4 разряд
	A20		O	Выход шины адреса, 20 разряд
	EX5INB		I	Вход альтернативного сигнала B прерывания 5
P4.5		89	I/O/Z/2	Вход/выход «порт 4», 5 разряд
	A21		O	Выход шины адреса, 21 разряд
	EX4INB		I	Вход альтернативного сигнала B прерывания 4
P4.6		90	I/O/Z/2	Вход/выход «порт 4», 6 разряд
	A22		O	Выход шины адреса, 22 разряд
	EX5INA		I	Вход альтернативного сигнала A прерывания 5
P4.7		91	I/O/Z/2	Вход/выход «порт 4», 7 разряд
	A23		O	Выход шины адреса, 23 разряд
	EX4INA		I	Вход альтернативного сигнала A прерывания 4
P5.0		29	I	Вход «порт 5», 0 разряд
	AN0		I	Вход АЦП, 0 канал
P5.1		30	I	Вход «порт 5», 1 разряд
	AN1		I	Вход АЦП, 1 канал
P5.2		31	I	Вход «порт 5», 2 разряд
	AN2		I	Вход АЦП, 2 канал
P5.3		32	I	Вход «порт 5», 3 разряд
	AN3		I	Вход АЦП, 3 канал
P5.4		33	I	Вход «порт 5», 4 разряд
	AN4		I	Вход АЦП, 4 канал
P5.5		34	I	Вход «порт 5», 5 разряд
	AN5		I	Вход АЦП, 5 канал
P5.6		35	I	Вход «порт 5», 6 разряд
	AN6		I	Вход АЦП, 6 канал

Продолжение таблицы 2.1

1		2	3	4
P5.7		36	I	Вход «порт 5», 7 разряд
	AN7		I	Вход АЦП, 7 канал
	QDI		I	Индексный вход квадратурного декодера
P5.8		37	I	Вход «порт 5», 8 разряд
	AN8		I	Вход АЦП, 8 канал
	QDB		I	Вход B квадратурного декодера
P5.9		38	I	Вход «порт 5», 9 разряд
	AN9		I	Вход АЦП, 9 канал
	QDA		I	Вход A квадратурного декодера
P5.10		39	I	Вход «порт 5», 10 разряд
	AN10		I	Вход АЦП, 10 канал
	T6EUD		I	Вход «направление счета» таймера T6
P5.11		45	I	Вход «порт 5», 11 разряд
	AN11		I	Вход АЦП, 11 канал
	T5EUD		I	Вход «направление счета» таймера T5
P5.12		46	I	Вход «порт 5», 12 разряд
	AN12		I	Вход АЦП, 12 канал
	T6IN		I	Вход «синхронизация» таймера T6
P5.13		47	I	Вход «порт 5», 13 разряд
	AN13		I	Вход АЦП, 13 канал
	T5IN		I	Вход «синхронизация» таймера T5
P5.14		48	I	Вход «порт 5», 14 разряд
	AN14		I	Вход АЦП, 14 канал
	T4EUD		I	Вход «направление счета» таймера T4
P5.15		49	I	Вход «порт 5», 15 разряд
	AN15		I	Вход АЦП, 15 канал
	T2EUD		I	Вход «направление счета» таймера T2
P6.0		7	I/O/Z/2	Вход/выход «порт 6», 0 разряд
	CS0#		O	Выход «выбор кристалла 0»
	CC0IO		I/O/Z	Вход/выход блока CAPCOM1, 0 канал
P6.1		8	I/O/Z/2	Вход/выход «порт 6», 1 разряд
	CS1#		O	Выход «выбор кристалла 1»
	CC1IO		I/O/Z	Вход/выход блока CAPCOM1, 1 канал
P6.2		9	I/O/Z/2	Вход/выход «порт 6», 2 разряд
	CS2#		O	Выход «выбор кристалла 2»
	CC2IO		I/O/Z	Вход/выход блока CAPCOM1, 2 канал
P6.3		10	I/O/Z/2	Вход/выход «порт 6», 3 разряд
	CS3#		O	Выход «выбор кристалла 3»
	CC3IO		I/O/Z	Вход/выход блока CAPCOM1, 3 канал
P6.4		11	I/O/Z/2	Вход/выход «порт 6», 4 разряд
	CS4#		O	Выход «выбор кристалла 4»
	CC4IO		I/O/Z	Вход/выход блока CAPCOM1, 4 канал
P6.5		12	I/O/Z/2	Вход/выход «порт 6», 5 разряд
	CC5IO		I/O/Z	Вход/выход блока CAPCOM1, 5 канал
	M0RxD0		I	Прямой вход 0 канала 0 блока ГОСТ Р 52070-2003
P6.6		13	I/O/Z/2	Вход/выход «порт 6», 6 разряд
	CC6IO		I/O/Z	Вход/выход блока CAPCOM1, 6 канал
	M0RxDN0		I	Инверсный вход 0 канала 0 блока ГОСТ Р 52070-2003

Окончание таблицы 2.1

1		2	3	4
P6.7		14	I/O/Z/2	Вход/выход «порт 6», 7 разряд
	CC7IO		I/O/Z	Вход/выход блока CAPCOM1, 7 канал
	M0RxD1		I	Прямой вход 1 канала 0 блока ГОСТ Р 52070-2003
P7.1		117	I/O/Z/2	Вход/выход «порт 7», 1 разряд
	CAN3TxD		O/Z	Выход узла CAN3
P7.2		118	I/O/Z/2	Вход/выход «порт 7», 2 разряд
	CAN2TxD		O/Z	Выход узла CAN2
P7.3		119	I/O/Z/2	Вход/выход «порт 7», 3 разряд
	CAN1TxD		O/Z	Выход узла CAN1
P7.4		15	I/O/Z/2	Вход/выход «порт 7», 4 разряд
	CC28IO		I/O/Z	Вход/выход блока CAPCOM2, 28 канал
	EX7INB		I	Вход альтернативного сигнала B прерывания 7
	M0RxDN1		I	Инверсный вход 1 канала 0 блока ГОСТ Р 52070-2003
P7.5		16	I/O/Z/2	Вход/выход «порт 7», 5 разряд
	CC29IO		I/O/Z	Вход/выход блока CAPCOM2, 29 канал
	CAN4TxD		O/Z	Выход узла CAN4
	EX6INB		I	Вход альтернативного сигнала B прерывания 6
P7.6		17	I/O/Z/2	Вход/выход «порт 7», 6 разряд
	CC30IO		I/O/Z	Вход/выход блока CAPCOM2, 30 канал
	EX7INA		I	Вход альтернативного сигнала A прерывания 7
	M0BLOCK0		O	Выход блокировки 0 канала 0 блока ГОСТ Р 52070-2003
P7.7		20	I/O/Z/2	Вход/выход «порт 7», 7 разряд
	CC31IO		I/O/Z	Вход/выход блока CAPCOM2, 31 канал
	CAN4RxD		I	Вход узла CAN4
	EX6INA		I	Вход альтернативного сигнала A прерывания 6
P9.0		23	I/O/Z/2	Вход/выход «порт 9», 0 разряд
	CC16IO		I/O/Z	Вход/выход блока CAPCOM2, 16 канал
	SDA0		I/O/Z	Вход/выход «данные» блока I2C
P9.1		24	I/O/Z/2	Вход/выход «порт 9», 1 разряд
	CC17IO		I/O/Z	Вход/выход блока CAPCOM2, 17 канал
	SCL0		I/O/Z	Вход/выход «тактовый сигнал» блока I2C
P9.2		25	I/O/Z/2	Вход/выход «порт 9», 2 разряд
	CC18IO		I/O/Z	Вход/выход блока CAPCOM2, 18 канал
	M0TxD0		O	Прямой выход 0 канала 0 блока ГОСТ Р 52070-2003
P9.3		26	I/O/Z/2	Вход/выход «порт 9», 3 разряд
	CC19IO		I/O/Z	Вход/выход блока CAPCOM2, 19 канал
	M0TxDN0		O	Инверсный выход 0 канала 0 блока ГОСТ Р 52070-2003
P9.4		27	I/O/Z/2	Вход/выход «порт 9», 4 разряд
	CC20IO		I/O/Z	Вход/выход блока CAPCOM2, 20 канал
	M0TxD1		O	Прямой выход 1 канала 0 блока ГОСТ Р 52070-2003
P9.5		28	I/O/Z/2	Вход/выход «порт 9», 5 разряд
	CC21IO		I/O/Z	Вход/выход блока CAPCOM2, 21 канал
	M0TxDN1		O	Инверсный выход 1 канала 0 блока ГОСТ Р 52070-2003
P9.6		6	I/O/Z/2	Вход/выход «порт 9», 6 разряд
	CAN1RxD		I	Вход узла CAN1
P9.7		3	I/O/Z/2	Вход/выход «порт 9», 7 разряд
	CAN2RxD		I	Вход узла CAN2

Таблица 2.2 – Функциональное назначение выводов микроконтроллера

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение
NMI#	5	I	Вход «немаскируемое прерывание»
READY	94	I	Вход «готовность»
ALE	95	O/Z	Выход «строб адреса»
RD#	92	O/Z	Выход «чтение команд/данных»
WR#/WRL#	93	O/Z	Выход «запись данных/запись младшего байта»
EA#	96	I	Вход «внешний доступ»
TRST#	58	I	Вход «сброс системы тестирования»
TMS	76	I	Вход «выбор тестового режима JTAG»
TCK	73	I	Вход «синхронизация JTAG»
TDI	74	I	Вход «данные JTAG»
TDO	75	O/Z	Выход «данные JTAG»
BRKIN#	144	I	Вход «останов» системы отладки
BRKOUT#	143	O/Z	Выход «останов» системы отладки
RSTIN#	142	I	Вход «сброс»
RSTOUT#	4	O/Z	Выход «индикация сброса»
XTAL1	139	I –	Вход основного тактового сигнала Вывод для подключения кварцевого резонатора
XTAL2	138	–	Вывод для подключения кварцевого резонатора
XTAL3	141	I –	Вход дополнительного тактового сигнала Вывод для подключения кварцевого резонатора (32 кГц)
XTAL4	140	–	Вывод для подключения кварцевого резонатора (32 кГц)
∩VCC3	42	–	Вывод опорного напряжения АЦП
∩0V3	41	–	Общий вывод АЦП
REFIO	40	–	Внешнее опорное напряжение АЦП
#VCC1	2, 22, 44, 62, 82, 105, 124	–	Выводы питания периферийной части микросхемы
#VCC2	19, 78, 102	–	Выводы питания ядра микросхемы
#0V1	1, 21, 43, 61, 83, 106, 125	–	Выводы общей шины периферийной части микросхемы
#0V2	18, 79, 103	–	Выводы общей шины ядра микросхемы

2.3 Электрические параметры

Полный перечень электрических параметров микросхемы при приемке и поставке и предельно допустимые значения параметров приведены в таблицах 2.3 и 2.4 соответственно.

Таблица 2.3 – 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_{CC1} = 3,0 В, U_{CC2} = 1,8 В, U_{CC3} = 3,0 В, I_{OL} = 5 мА		U_{OL}	–	0,45	–60 ± 3 25 ± 10 125 ± 5
2 Выходное напряжение высокого уровня на выводах P0 – P4, P6, P7, P9, ALE, RD#, WR#/WRL#, RSTOUT#[1], В, U_{CC1} = 3,0 В, U_{CC2} = 1,8 В, U_{CC3} = 3,0 В, I_{OH} = –5 мА		U_{OH}	U_{CC1}–0,5	–	
3 Ток утечки на входах P5.0 – P5.15[2], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{IL} = 0 В	I_{ILL1}	–10	10	
	U_{IH} = 3,6 В	I_{ILH1}	–10	10	
4 Ток утечки на входах NMI#, TRST#, TCK, TDI, TMS, RSTIN#, BRKIN#, EA#[2], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{IL} = 0,45 В	I_{ILL2}	–10	10	
	U_{IH} = 3,6 В	I_{ILH2}	–10	10	
5 Ток утечки в состоянии «Выключено» по выводам P0 – P4, P6, P7, P9, TDO[2], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{IL} = 0,45 В	I_{OZL}	–10	10	
	U_{IH} = 3,6 В	I_{OZH}	–10	10	
6 Входной ток по выводам P0, RD#, WR#/WRL#, READY[2, 3], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{IL} = 0,8 В	I_{IL1}	–190	–	
	U_{IH} = 1,8 В	I_{IH1}	–	–10	
7 Входной ток по выводам XTAL1, XTAL3[2], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 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_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{OL} = 0,45 В	I_{OL1}	10	–	
	U_{OH} = 2,5 В	I_{OH1}	–	170	
9 Выходной ток по выводам P6.0 – P6.4[2, 3], мкА, U_{CC1} = 3,6 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В	U_{OL} = 0,45 В	I_{OL2}	–190	–	
	U_{OH} = 2,5 В	I_{OH2}	–	–10	
10 Динамический ток потребления по источнику питания U_{CC2}[4], мА, U_{CC1} = 3,6 В, U_{CC2} = 2,0 В, U_{CC3} = 3,6 В, $f_{CIXTAL1}$ = 66 МГц		I_{OCC2}	–	180	
11 Ток потребления по источнику питания U_{CC2} в режиме холостого хода[4], мА, U_{CC1} = 3,6 В, U_{CC2} = 2,0 В, U_{CC3} = 3,6 В, $f_{CIXTAL1}$ = 66 МГц		I_{CCI}	–	80	

Окончание таблицы 2.3

1	2	3	4	5
12 Ток потребления по источнику питания U_{CC1} в режиме хранения, мА, U_{CC1} = 3,6 В, U_{CC2} = 2,0 В, U_{CC3} = 3,6 В, $f_{CIXTAL1}$ = 33 МГц	I_{CCS1}	–	10	
13 Ток потребления по источнику питания U_{CC2} в режиме хранения, мА, U_{CC1} = 3,6 В, U_{CC2} = 2,0 В, U_{CC3} = 3,6 В, $f_{CIXTAL3}$ = 32 кГц	I_{CCS2}	–	9	
14 Отношение сигнал/(шум + искажения) в каналах АЦП, дБ, U_{CC1} = 3,3 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В, f_{IN} = 1 МГц	SINAD	51	–	–60 ± 3 25 ± 10 125 ± 5
15 Дифференциальная нелинейность АЦП (режим преобразования – 12 бит), LSB, U_{CC1} = 3,3 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В,	E_{LD}	–2	2	
16 Интегральная нелинейность АЦП (режим преобразования – 12 бит), LSB, U_{CC1} = 3,3 В, U_{CC2} = 1,8 В, U_{CC3} = 3,6 В,	E_L	–4	4	
17 Функциональный контроль, U_{CC1} = (3,0; 3,6) В, U_{CC2} = (1,6; 2,0) В, U_{CC3} = (3,0; 3,6) В, $f_{CIXTAL1}$ = (1; 66; 80) МГц	ФК	–	–	

———————

[1)] Параметр не действителен 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 2.4 – 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 источника питания периферийной части ИС, В	U_{CC1}	3,0	3,6	–0,5	5,0
2 Напряжение источника питания ядра микросхемы, В	U_{CC2}	1,6	2,0	–0,5	2,5
3 Напряжение питания АЦП, В	U_{CC3}	3,0	3,6	–0,5	5,0
4 Входное напряжение низкого уровня по выводам XTAL1, XTAL3, В	U_{IL}	–0,5	$0,3U_{CC1}$	–0,6	–
5 Входное напряжение высокого уровня по выводам XTAL1, XTAL3, В	U_{IH}	$0,7U_{CC1}$	$U_{CC1}+0,5$	–	$U_{CC1}+0,6$
6 Выходной ток низкого уровня, мА	I_{OL}	–	5	–	10
7 Выходной ток высокого уровня, мА	I_{OH}	–5	–	–10	–
8 Емкость нагрузки, пФ	C_L	–	50	–	–
9 Частота следования импульсов тактового сигнала, МГц	$f_{CIXTAL1}$	1,0	80,0	–	–
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.					

3 Архитектура изделия

Архитектура микроконтроллера 1887ВЕ9Т объединяет в себе хорошо сбалансированные между собой преимущества архитектур RISC и CISC процессоров. Микроконтроллер имеет мощное процессорное ядро и набор периферийных модулей, объединенных высокоэффективной системой взаимодействия.

Структурная схема микроконтроллера представлена на рисунке 3.1.

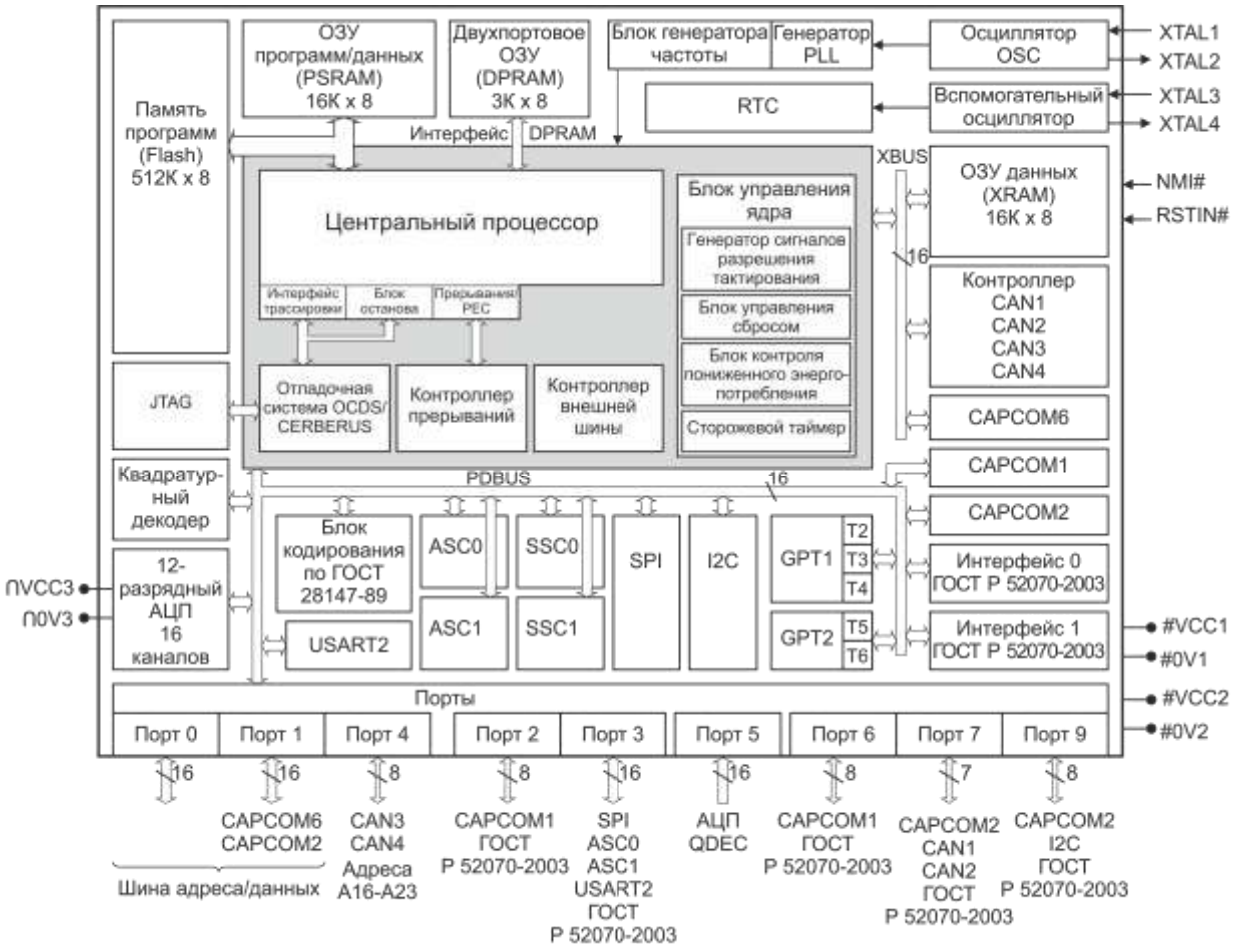


Рисунок 3.1 – Структурная схема микроконтроллера 1887ВЕ9Т

3.1 ЦПУ

Основным компонентом микроконтроллера является центральное процессорное устройство (ЦПУ). Оно имеет в своем составе ядро, блок управления ядра, контроллеры прерываний и внешней шины и систему отладки.

Ядро включает в свой состав пятиступенчатый конвейер команд, 16-разрядное арифметическое и логическое устройство (АЛУ), модуль умножения и деления, генератор битовой маски и циклический сдвигатель, а также регистры специальных функций (регистры SFR) и имеет отдельный блок умножения-накопления (MAC). Блок управления ядра контролирует синхронизацию, сброс и энергопотребление ЦПУ.

Работа микроконтроллера основана на тактовом сигнале ЦПУ. Внутренний генератор получает сигнал от кристалла или внешнего тактового генератора и формирует три таковых сигнала: прямой и инверсный для ЦПУ и синхросигнал периферии. Тактовый сигнал периферии синхронизирован с тактовым сигналом ЦПУ, но является независимым и имеет фазовый сдвиг на 180° относительно него. В режиме холостого хода IDLE

тактовый сигнал ЦПУ не генерируется, а периферия продолжает свою работу. Для каждого периферийного блока генерируются индивидуальный тактовый сигнал.

Блок управление сбросом отвечает за аппаратный сброс (немедленный переход в состояние сброса, асинхронно по отношению к тактовой частоте ЦПУ), программный сброс и сброс по сигналу сторожевого таймера (оба выполняются синхронно по отношению к тактовой частоте ЦПУ).

Специальным механизмом предотвращения (посредством аппаратного сброса) неправильного функционирования микроконтроллера является сторожевой таймер. Сторожевой таймер всегда запущен после сброса, но может быть программно запрещен.

Управление режимами холостого хода Idle и пониженного энергопотребления PowerDown осуществляется блоком контроля пониженного энергопотребления.

Для возможности обработки большого количества прерываний в состав ЦПУ включены многоприоритетный контроллер прерываний и контроллер периферийных событий (PEC), а также переключаемые банки регистров. Реализован механизм распознавания и обработки некорректных или ошибочных состояний (аппаратные ловушки). Вследствие возможности прерывания многотактных команд (умножения и деления) укорочено время начала обработки прерываний.

Контроллер периферийных событий используется для разгрузки центрального процессора, позволяя исключить дополнительные операции при обслуживании прерывания, путем однотактного перемещения байта/слова, вызванного запросом на прерывание, между двумя адресами в нулевом сегменте памяти.

Многоприоритетный контроллер прерываний управляет распределением приоритетов и позволяет предотвращать прерывание друг другом одинаковых по приоритету задач. Для каждого из векторов прерываний существует отдельный регистр управления, который включает в себя флаги запроса и разрешения прерывания, а также поле приоритета. В случае начала обслуживания запроса прерывания его можно прервать только запросом более высокого приоритета. Для стандартной обработки прерываний каждый источник прерываний имеет вектор прерывания.

Укороченное время начала прерывания доступно вследствие возможности прерывания многотактных команд (умножения и деления).

Микроконтроллер распознает и обрабатывает некорректные или ошибочные состояния, которые возникают в течение работы (так называемые «аппаратные ловушки»). Аппаратные ловушки вызывают немедленную немаскируемую реакцию системы, аналогичную стандартной процедуре обслуживания прерываний (переход к определенной точке таблицы прерываний). Возможностью использования аппаратных ловушек управляет регистр TFR. Аппаратные ловушки прерывают исполнение любой программы, за исключением программ обслуживания других ловушек с более высоким приоритетом. В свою очередь обслуживание ловушки не может быть прервано стандартными или PEC прерываниями.

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

Встроенный контроллер внешней шины позволяет получить доступ к внешней памяти и/или к внешним периферийным устройствам. В микроконтроллере реализована возможность подключения через интерфейс внешней шины до 16 Мбайт внешнего ОЗУ и/или ПЗУ. Возможна организация внешней демультиплексной/мультиплексной шины с 8/16-разрядными данными и 16/18/20/24-разрядным адресом.

Программируемые важные временные характеристики, такие как время обращения к памяти, циклы ожидания, длина ALE, задержка чтения/записи и др., позволяют пользователю адаптировать различные типы памяти и/или периферии. Доступ к очень медленной памяти или периферии поддерживается с помощью специальной функции готовности.

Встроенная в микроконтроллер шина XBUS является внутренним отображением внешней шины и позволяет организовывать доступ к интегрированным модулям и встроенной периферии как к внешним компонентам. Память XRAM и контроллер интерфейса CAN являются представителями X-периферии.

Система отладки (OCDS) позволяет выполнять отладку программы пользователя, посредством управления группой специальных регистров. Эти регистры также доступны посредством интерфейса JTAG.

3.2 Блоки памяти

Пространство памяти микроконтроллера устроено по принципу архитектуры Фон-Неймана: память программ и память данных находятся в одном линейном адресном пространстве, объемом до 16 Мбайт. Доступ к данным осуществляется побайтно и пословно. Часть памяти доступна побитно.

Флеш-память программ (Flash) объемом 512 Кбайт и дополнительное ОЗУ программ/данных (PSRAM) объемом 16 Кбайт соединены с центральным процессором через 32-разрядную шину локальной памяти.

Область регистров специальных функций имеет объем 1 Кбайт и состоит из двух равных областей – стандартной области регистров SFR и области расширенных регистров SFR (регистры ESFR, каждый длиною в слово).

Область регистров специальных функций X-периферии (регистры XSFR для устройств, подключенных к шине XBUS) имеет объем 26 Кбайт.

Внутреннее 16-разрядное ОЗУ (DPRAM) объемом 3 Кбайта отведено под регистры общего назначения (регистры GPR), пользовательские данные (переменные) и системный стек, а также может использоваться как память программ и доступно посредством прямой и косвенной адресации. В любой области DPRAM могут быть организованы пользовательские банки регистров, каждый объемом по 16 байт/слов. Базовый адрес активного банка регистров, доступного для центрального процессора, определяется контекстным указателем CP. Для простой передачи параметров области банков могут перекрываться друг другом. Количество банков регистров ограничено только объемом ОЗУ. Для временного хранения данных используется системный стек, доступ к которому организован с помощью регистра указателя стека SP. Границы стека задаются специальными регистрами общего назначения.

ОЗУ данных **(**XRAM) имеет объем 16 Кбайт и предназначено для хранения данных, а также может использоваться для организации системного стека.

3.3 Периферия

Четкое разделение периферийных модулей и ядра позволяет выполнять максимальное количество параллельных операций. Каждый функциональный блок обрабатывает данные независимо, передача данных и управление осуществляются через общую шину.

Для управления работой блоков и хранения результатов используются наборы регистров специального назначения, расположенные в областях памяти SFR и ESFR, а также в области XSFR (для устройств X-периферии).

Для связи с внешним оборудованием используются выводы портов микроконтроллера.

Параллельные порты

103 канала ввода-вывода микроконтроллера организованы в девять параллельных портов. Каналы имеют программируемые альтернативные функции. Разряды портов являются бит-адресуемыми и могут индивидуально программироваться на ввод или вывод.

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.

Блок АЦП

12-разрядный аналого-цифровой преобразователь с поддержкой режимов однократного и многократного преобразования для каждого из 16 каналов с возможностью включения автоматического сканирования каналов.

Блоки таймеров общего назначения

Блоки GPT1 и GPT2 являются многофункциональными блоками, объединяющими пять независимых таймеров-счетчиков, которые могут использоваться для формирования временных интервалов, измерения длительности внешних импульсов и вывода импульсов. Блок GPT1 содержит три таймера (поддерживают каскадирование и захват/перезагрузку с тактовой частотой работы до 1/4 тактовой частоты периферии). Блок GPT2 содержит два таймера (с тактовой частотой работы до 1/2 тактовой частоты периферии) и специальный регистр захвата/перезагрузки CAPREL.

Блоки захвата/сравнения

Два блока захвата/сравнения CAPCOM1 и CAPCOM2 обеспечивают управление временными последовательностями посредством 32 каналов. Блоки обычно используются для высокоскоростного формирования сигналов сложной формы, широтно-импульсной модуляции и регистрации моментов времени возникновения определенных событий.

Каждый блок имеет в своем составе два 16-разрядных таймера-счетчика, синхронизируемых тактовым сигналом ЦПУ или сигналом переполнения таймера блока GPT2, а также шестнадцать регистров захвата/сравнения.

Блок широтно-импульсной модуляции

Блок CAPCOM6 обеспечивает управление временными последовательностями посредством 4 каналов. Включает в свой состав два подблока таймеров, каждый со своими регистрами захвата/сравнения. Специальные режимы работы позволяют также осуществлять управление бесщеточными ДПТ с датчиками Холла или контролем обратной ЭДС.

Квадратурный декодер

Квадратурный декодер является блоком преобразования цифровых сигналов с датчиков положения для вычисления скорости, направления вращения и текущего положения вала двигателя. Декодер использует два квадратурных входа микроконтроллера и один индексный вход.

Блок кодирования/декодирования

Блок является аппаратной реализацией методов кодирования/декодирования блоков данных (по 8 байт) на основе алгоритмов ГОСТ 28147–89 симметричными методами с возможностью формирования контрольной комбинации.

Асинхронные/синхронные последовательные каналы передачи данных

Блоки ASC0, ASC1 и приемопередатчик USART2 обеспечивают последовательную связь с другими микроконтроллерами и внешними устройствами, позволяя осуществлять дуплексный асинхронный и полудуплексный синхронный обмен данными. Каждый из трех блоков при внутренней частоте тактирования 50 МГц, позволяет осуществлять обмен данными на скоростях до 1,5 Мбод.

Синхронные последовательные каналы передачи данных

Блоки SSC0, SSC1 обеспечивают последовательную связь с другими микроконтроллерами и внешними устройствами, позволяя осуществлять дуплексный и полудуплексный синхронный обмен данными. Контроллер интерфейса SPI позволяет осуществлять последовательную передачу данных в дуплексном режиме по протоколу SPI.

Контроллер интерфейса I2C

Контроллер обеспечивает полную поддержку двухпроводного последовательного синхронного интерфейса I2C/SMBus и, как следствие, легкое соединение со многими устройствами ввода-вывода, запоминающими устройствами, АЦП, ЦАП и др. Блок поддерживает стандартный, скоростной и высокоскоростной режимы работы.

Контроллер интерфейса CAN

Контроллер обеспечивает полную поддержку последовательного интерфейса связи CAN версии 2.0B (активная версия) и может функционировать в таких условиях как обрыв одного из трех проводов шины и закорачивание одного из проводов шины на питание/общий провод (стандарт ISO 11898).

Контроллер имеет в своем составе четыре идентичных узла CAN, каждый из которых реализует программируемую скорость обмена данными до 1 Мбит/с.

ОЗУ, общее для узлов, рассчитано на 256 объектов сообщений. Каждый объект сообщения может быть связан с любым из узлов, сконфигурирован для передачи или приема как стандартных, так и расширенных сообщений и удаленных запросов.

Контроллеры интерфейса ГОСТ Р 52070-2003

В состав микроконтроллера входят два идентичных контроллера. Каждый контроллер поддерживает обмен данными с другими устройствами посредством магистрального последовательного интерфейса (ГОСТ Р 52070-2003), совместимого с интерфейсом стандарта MIL-STD-1553B. На физическом уровне интерфейс представляет собой последовательную шину данных (экранированная витая пара), к которой подключены устройства. Контроллер может функционировать в режиме контроллера шины (с возможностью обращения к любому из 31 оконечных устройств с 5-битным адресом), монитора шины или оконечного устройства (удаленного терминала).

4 Ядро микроконтроллера

4.1 АЛУ и конвейер команд

Все стандартные арифметические, сдвиговые и логические операции производятся в 16-разрядном арифметическом и логическом устройстве (АЛУ). АЛУ оптимизировано для работы с 8- и 16-разрядными числами. Большинство инструкций микроконтроллера выполняется за один машинный цикл, который равен двум тактам частоты ЦПУ. Деление 32-разрядного числа на 16-разрядное занимает 10 тактов частоты ЦПУ, а умножение 16-разрядных чисел – 5 тактов.

После каждой арифметической и логической операции, операции с битами и операции перемещения данных автоматически обновляются флаги в регистре слова состояния процессора PSW. Поддержка знаковой и беззнаковой арифметики обеспечивается с помощью определяемых пользователем условий.

Длинные команды умножения и деления могут быть прерваны во время выполнения, что позволяет получать более быстрые ответы на прерывания.

Применение 5-ступенчатого конвейера команд для обработки инструкций существенно сокращает время выполнения программ, поскольку позволяет ядру ЦПУ параллельно обрабатывать разные этапы выполнения четырех последовательно идущих команд. Конвейерная обработка состоит из четырех этапов: выборка, декодирование, выполнение и запись результата.

На этапе выборки команда считывается из внутренней или внешней памяти по адресу, определяемому значениями указателей команд IP и сегмента кода CSP. Далее выбранная команда декодируется, и считываются необходимые операнды, после чего декодированная команда выполняется. После этого, если требуется, производится запись результатов выполнения команды.

4.2 Операции с битами

Для обработки битов предназначено большое число команд, которые обеспечивают постоянный доступ к двум операндам в побитно адресуемом пространстве памяти, без необходимости перемещения данных в промежуточные регистры.

Пространство DPRAM, регистры GPR, старшие 256 байт области SFR и область ESFR поддерживают побитную адресацию. Регистры внешнего адресного пространства не доступны побитно.

Следует учитывать, что механизм чтение-модификация-запись влияет на биты, к которым происходит обращение, и их состояние может аппаратно меняться во время обратной записи. При одновременно аппаратном и программном обращении к биту с целью его модификации, программный запрос имеет приоритет.

4.3 Блок умножения и деления

В состав ЦПУ входит отдельный блок умножения и деления. Регистры MDH и MDL являются регистрами старшего и младшего, соответственно, слов 32-разрядного регистра, используемого ЦПУ при совершении операций умножения и деления.

Перед началом операции умножения в регистры загружаются множители. После умножения регистры содержат старшее и младшее слова 32-разрядного результата.

При делении 16-разрядного числа, делимое загружается в регистр MDL. Далее загружается делитель. При длинном делении для старшего и младшего слов делимого используются оба регистра MDH и MDL. После деления регистр MDL содержит 16-битное целое частное, а регистр MDH – 16-битный остаток.

Индикатором использования регистров MDH и MDL является флаг MDRIU регистра MDC, который устанавливается при записи в регистры и сбрасывается при чтении MDL (в связи с этим при чтении результата первым следует читать регистр MDH).

Если результат умножения/деления не может быть представлен словом (содержит больше 16 разрядов), устанавливается флаг V в регистре PSW. Этот флаг используется для определения необходимости копирования обоих слов результата из регистров MDH и MDL.

В приведенном примере показана последовательность команд для выполнения умножения двух беззнаковых 16-битных чисел.

```
SAVE:
JNB MDRIU, START  ; Проверка использования регистров MDH и MDL
SCXT MDC, #0010h  ; Сохранение и очистка регистра управления, снятие флага
                  ; MDRIU (требуется только для прерываемых команд)
BSET SAVED        ; Указание сохранения
PUSH MDH          ; Сохранение значений MDH и MDL в стеке
PUSH MDL

START:
MULU R1, R2       ; Беззнаковое умножение с установкой флага MDRIU
JMPR cc_NV, COPYL ; Проверка результата на переполнение
MOV R3, MDH       ; Сохранение MDH в R3

COPYL:
MOV R4, MDL       ; Сохранение MDL в R4 и автоматический сброс MDRIU

RESTORE:
JNB SAVED, DONE   ; Проверка сохранения старого значения MDH и MDL в стеке
POP MDL           ; Восстановление значений регистров
POP MDH
POP MDC           ; (должно быть прочитано из стека до команды RETI)
BCLR SAVED        ; Умножение завершено
                  ; Программа продолжается
DONE:
```

Ниже приведен пример последовательности команд для выполнения деления 32-битного числа на 16-битное.

```
MOV MDH, R1       ; Запись делимого в регистры MDH и MDL с установкой флага
MOV MDL, R2       ; MDRIU
DIV R3            ; Деление на R3
JMPR cc_V, ERROR  ; Проверка результата на переполнения
MOV R3, MDH       ; Сохранения остатка
MOV R4, MDL       ;Сохранения целого в R4 и автоматический сброс флага
                  ; MDRIU
```

В случае если операции умножения/деления прерываются, и подпрограмме обслуживания прерывания требуется выполнить собственное умножение/деление, то содержимое регистров MDH, MDL и MDC должно быть сохранено и восстановлено по окончании вычислений во избежание ошибочных результатов.

В случае, когда команда умножения/деления прерывается во время выполнения, 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/деления читается из адреса, сохраненного в стеке, и завершается после выполнения команды RETI.

Форматы данных

Микроконтроллер поддерживает операции с битами, битовыми строками, символами и целочисленными типами данных. Форматы данных микроконтроллера и соответствие этих форматов и типов данных системы ANSI C указано в таблице 4.1.

Таблица 4.1 – Поддерживаемые типы данных ANSI C

Тип данных ANSI C	Размер	Диапазон	Формат данных ЦПУ
bit	1 бит	0 или 1	BIT
sfrbit	1 бит	0 или 1	BIT
esfrbit	1 бит	0 или 1	BIT
signed char	8 бит	от –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
unsigned int	16 бит	от 0 до 65 535	WORD
near pointer	16 бит	16 или 14 бит в зависимости от модели памяти	WORD
signed long, unsigned long, float, far pointer, huge pointer, shuge pointer	32 бит	Не поддерживаются	
double, long double	64 бит		
Примечание – Размер данных: BIT – 1 бит, BYTE – 8 бит (байт), WORD – 16 бит (слово).			

4.4 Системные регистры SFR

Регистры специальных функций доступны посредством любых команд, действительных для адресного пространства SFR. Пять регистров имеют особенности. Регистры IP и CSP не доступны – их значения могут быть изменены только косвенно (посредством команд перехода). Регистры PSW, SP и MDC могут быть изменены как с помощью прямых команд, так и посредством выполнения специальных команд ЦПУ.

Регистры SFR доступны для записи и чтения как пословно, так и побайтно. Часть регистров доступна побитно.

Если возникает необходимость обращения к регистрам области расширенных регистров ESFR, то при использовании 8-битной или прямой битовой адресации следует использовать команду расширения EXTR (при 16-битной и косвенной адресации не требуется):

```
EXTR #4                    ;  Включение ESFR для следующих четырех команд
MOV ODP2, #data16          ; ODP2 использует 8-битный адрес
BFLDL DP6, #mask, #data8
```

```
BSET DP6.7
MOV T8REL, R1          ; T8REL использует 16-битный адрес
                       ; (EXTR не требуются для этого доступа и не оказывает
                       ; влияния)
```

При выполнении программы адрес ячейки кода формируется путем конкатенации номера сегмента, определяемого регистром CSP и 16-разрядного сегментного адреса, задаваемого регистром IP. Доступ к коду выполняется в режиме сегментированной памяти (включен по умолчанию) или несегментированной памяти. Режим выбирается битом SGTDIS регистра SYSCON.

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.

Для конфигурации микроконтроллера и управления им используется побитно адресуемый регистр SYSCON. После любой команды изменяющей значение регистра SYSCON, необходимо выполнить, по крайней мере, одну команду, не использующую этот регистр. Состояние регистра после сброса зависит от состояния конфигурационных входов во время сброса.

Текущее состояние микроконтроллера (состояние АЛУ и прерываний ЦПУ) отражает побитно адресуемый регистр PSW. Биты N, C, V, Z, E, MULIP являются флагами состояния АЛУ после выполнения последней операции. Эти флаги устанавливаются по специальным правилам и в зависимости от операции в АЛУ. После любой команды, изменяющей значение регистра PSW, необходимо выполнить, по крайней мере, одну команду, выполнение которой не зависит от флагов АЛУ, уровня приоритета ILVL и бита IEN. Чтение регистра возвращает его состояние на момент предыдущей операции и изменяет состояние флагов.

Регистр CP содержит адрес активного банка регистров GPR (в пределах области DPRAM). После любой команды, изменяющей значение регистра CP (переключение банков), необходимо выполнить, по крайней мере, две команды, не использующие для доступа к регистрам нового банка короткие виды адресации (8/4/2-битную и битовую). В подпрограммах обработки запросов прерываний для этого удобно использовать команду SCXT, которая перед обновлением сохраняет в системном стеке содержимое регистра CP:

```
SCXT CP, #0FC00h    ; Задание нового значения базового адреса банка регистров

…                   ; Любая команда, не использующая регистры GPR

MOV R0, #data       ; Запись в регистр GPR нового значения
```

Указатели DPP0, DPP1, DPP2 и DPP3 содержат номера страниц, которые используются для полной 16-битной 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 изменяющей значение регистра DPPx необходимо выполнить, по крайней мере, одну команду, не использующую этот регистр:

```
MOV DPP0, #4        ; Выбор четвертой страницы данных
```

```
…                       ; Любая инструкция, не использующая DPP0

MOV 0000h, r1           ; Пересылка по адресу 01'0000h (четвертая страница)
```

Регистр SP содержит адрес вершины системного стека. После любой команды изменяющей значение регистра SP, необходимо выполнить, по крайней мере, одну команду, не использующую системный стек. Это необходимо для корректного выполнения команд RET, RETI, RETS, RETP и POP:

```
MOV SP, #0FA40h     ; Указание вершины стека

…                   ; Любая инструкция, не использующая стек

POP R0              ; Загрузка R0 значением из вершины стека
```

Параметры обращения по внешней шине во время доступа к адресному окну x (значения от 1 до 4) определяются парой регистров BUSCONx и ADDRSELx.

Изменение состояния регистра ADDRSELx и затем параметров (кроме временных) в регистре BUSCONx (только в такой последовательности) 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.

Побитно адресуемые доступные только для чтения регистры ZEROS и ONES являются регистрами постоянного нуля и единицы и могут использоваться в операциях с битами, а также для создания масок.

Регистр CPUID содержит номер и версию микроконтроллера 1887BE9T.

4.5 Режимы адресации

Микроконтроллер осуществляет несколько режимов адресации для пословного, побайтного и побитного обращения (короткого, длинного, косвенного) к данным.

Адресация посредством указателей сегмента и команд

16 Мбайт адресуемого пространства микроконтроллера состоит из 256 сегментов по 64 Кбайта.

Для доступа к ячейкам памяти используется 24-разрядный указатель адреса, который состоит из 8-разрядного указателя сегмента CSP и 16-разрядного указателя команды (смещения) IP, см. рисунок 4.1.

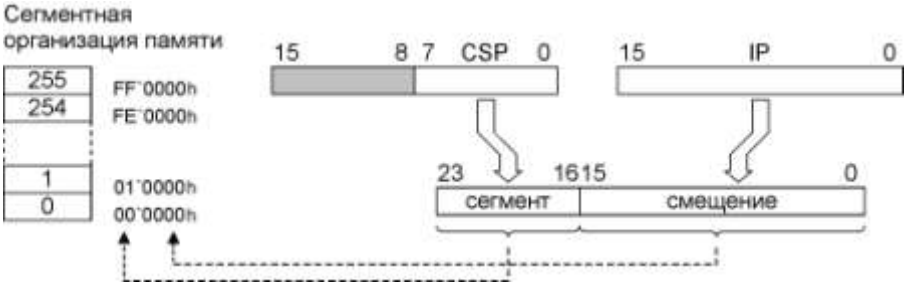


Рисунок 4.1 – Адресация посредством CSP и IP

Способы адресации при переходах представлены в таблице 4.2.

Таблица 4.2 – 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

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 регистра 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 В. 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.

Режимы короткой адресации

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

При побайтном обращении физический адрес формируется сложением начального и короткого адресов.

При пословном обращении физический адрес формируется сложением начального и удвоенного короткого адресов, см. таблицу 4.3.

Таблица 4.3 – Режимы короткой адресации

Мнемо-код	Физический адрес	Диапазон короткого адреса	Границы области применения
1	2	3	4
Rw	(CP) + 2×Rw или локальный	Rw = 0 – 15	Область GPR (слово)
Rb	(CP) + 1×Rb или локальный	Rb = 0 – 15	Область GPR (байт)

Окончание таблицы 4.3

1	2	3	4
Reg	00’FE00h + 2×reg	reg = 00h – EFh	Область SFR (слово, младший байт)
	00’F000h + 2×reg		
	(CP) + 2× (reg^0Fh)	reg = F0h – FFh	Область ESFR (слово, младший байт)
	(CP) + 1× (reg^0Fh)		Область GPR (слово, байт)
bitoff	00’FD00h + 2×bitoff	bitoff= 00h – 7Fh	Бит в смещении слова из областей DPRAM, SFR, ESFR, GPR
	00’FF00h + 2× (bitoff^7Fh)	bitoff= 80h – EFh	
	00’F100h + 2× (bitoff^7Fh)	bitoff= 80h – EFh	
	(CP) + 2× (bitoff^0Fh)	bitoff= F0h – FFh	
bitaddr	Смещение как в bitoff. Непосредственная позиция бита (bitpos)	bitoff = 00h – FFh	Любой бит
		bitpos = 0 – 15	

Rw, Rb – определяют прямой доступ к любому регистру GPR в активном (локальном или глобальном) банке регистров. Для Rw и Rb требуется 4 бита в формате команды. Базовый адрес глобального банка регистров задается регистром CP. Rw определяет 4-разрядный адрес слова, а Rb – 4-разрядный адрес байта регистра GPR в пределах локального банка регистров или относительно CP.

reg – определяет прямой доступ к любому регистру SFR или регистру GPR в активном банке. Для reg требуется 8 бит 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. Следует отметить, что, используя режим адресации reg, нельзя обратиться к старшему байту регистра 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 идентично 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 бит 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 бит в формате команды.

Адресация с использованием указателя страницы данных DPP

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

Перемещение по страницам данных осуществляется конкатенацией младших 14 битов прямого или косвенного длинного 16-битного адреса с содержимым регистра DPP, выбираемого двумя старшими битами 16-битного адреса. Содержимое выбранного регистра DPP указывает на одну из 1024 страниц данных. Адрес страницы данных и младшие 14 битов длинного адреса вместе составляют 24-битный физический адрес, см. рисунок 4.2.

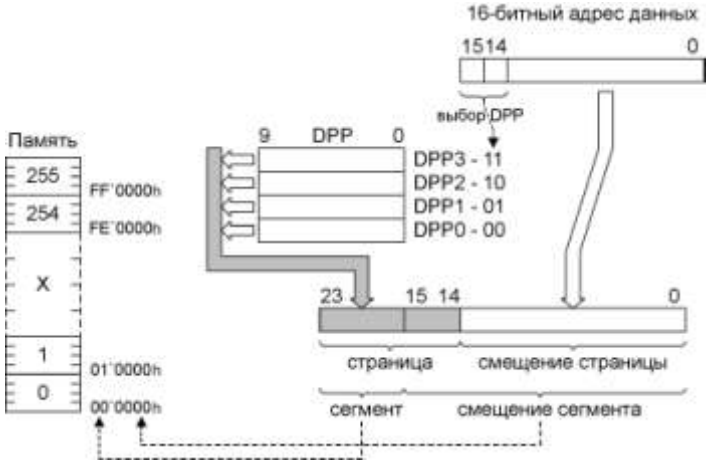


Рисунок 4.2 – Адресация посредством регистров DPP

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.

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), см. рисунок 4.3.

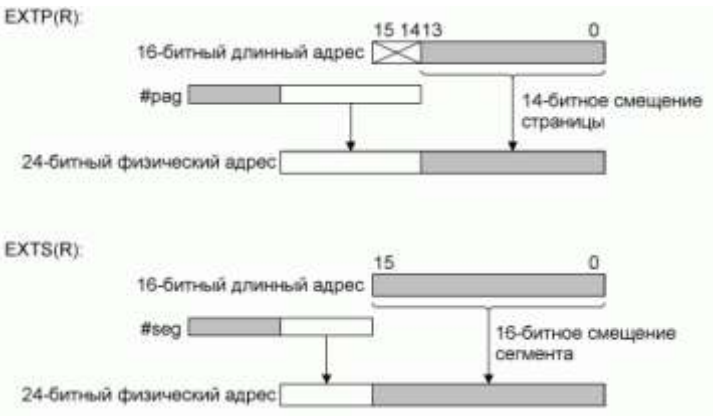


Рисунок 4.3 – Механизм замещения регистра DPP

Режим длинной адресации

Использует 16-разрядное постоянное значение, кодируемое в формат команды, которое определяет смещение страницы данных и выбирает регистр DPP, см. таблицу 4.4.

Таблица 4.4 – 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
	pag|| mem^3FFFh	
	seg|| mem	

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

Режимы косвенной адресации

С помощью этих режимов может быть доступно любое слово и/или байт данных в пределах внутреннего адресного пространства.

Режимы могут быть определены как комбинации режимов короткой и длинной адресаций. Это означает, что длинный 16-битный адрес формируется косвенно содержимым регистра GPR (слово), который в свою очередь выбирается коротким 4-битным адресом. 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, см. рисунок 4.4.

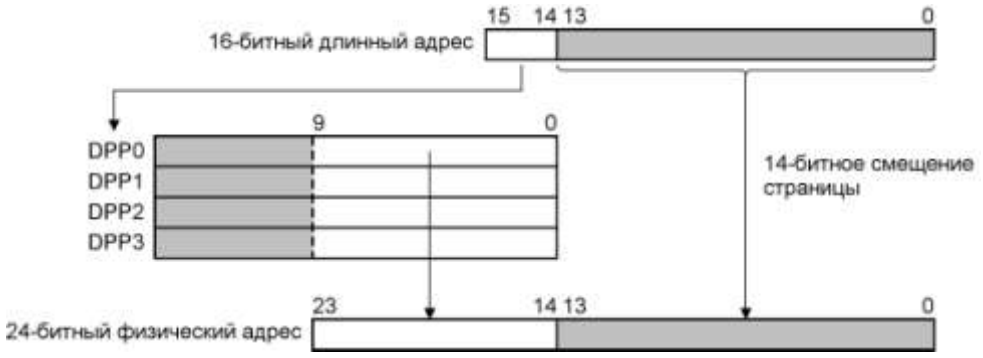


Рисунок 4.4 – Формирование 24-битного адреса

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.

Физический адрес формируется из косвенных указателей адреса по алгоритму:

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

$$\text{Адрес GPR} = (\text{CP}) + 2 \times \text{короткий адрес}.$$

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:

Адрес GPR = (Адрес GPR) – (2, если байт, или 1, если слово).

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

Длинный адрес = (указатель GPR) + постоянное значение.

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

Физический адрес = (DPP) + смещение страницы.

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

Указатель GPR = (Указатель GPR) ± (2, если байт, или 1, если слово).

Поддерживаемые режимы косвенной адресации представлены в таблице 4.5.

Таблица 4.5 – Режимы косвенной адресации

Мнемоника	Описание
[Rw]	Для косвенной адресации большинство команд используют любой из регистров 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

Константы

В дополнение к основным способам адресации микроконтроллер 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 бит. Короткие константы всегда дополняются (расширяются), в то время как длинные константы усекаются в случае необходимости, чтобы соответствовать формату данных, требуемому для определенных действий, см. таблицу 4.6

Таблица 4.6 – 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 – Перед непосредственной константой всегда вначале стоит знак #.		

4.6 Банки регистров GPR

Микроконтроллер оперирует с банками регистров общего назначения. Каждый банк содержит 16 регистров (с R0 по R15), которые могут использоваться как рабочие регистры блока АЛУ, а также как указатели адреса в режимах косвенной адресации. Области банков могут перекрываться; количество банков ограничено только объемом ОЗУ. Базовый адрес активного банка задается регистром контекстного указателя CP. В противоположность системному стеку, нумерация внутри банка регистров возрастает от меньших к большим адресам. Регистры активного банка доступны с помощью 2/4/8-битной адресации и доступны побитно.

Переключение между банками регистров происходит при изменении значения регистра CP. Команда переключения контекста SCXT обеспечивает переключение банков регистров и автоматическое сохранение предыдущего значения CP.

После сброса состояние регистров GPR не определено.

Карта адресов регистров в пределах одного банка представлена в таблице 4.7.

Таблица 4.7 – Карта адресов регистров банка

Адрес	Регистры слов	Регистры байт	
+ 1Eh	R15	–	
+ 1Ch	R14	–	
+ 1Ah	R13	–	
+ 18h	R12	–	
+ 16h	R11	–	
+ 14h	R10	–	
+ 12h	R9	–	
+ 10h	R8	–	
+ 0Eh	R7	RH7	RL7
+ 0Ch	R6	RH6	RL6
+ 0Ah	R5	RH5	RL5
+ 08h	R4	RH4	RL4
+ 06h	R3	RH3	RL3
+ 04h	R2	RH2	RL2
+ 02h	R1	RH1	RL1
CP	R0	RH0	RL0

Короткая 4-битная адресация регистров GPR

Обозначается Rw или Rb. Позволяет использовать относительные адреса памяти, привязанные 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.

Регистры 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 регистры доступны с помощью короткой 2-битной адресации, для которой вычисление физических адресов идентично 4-битной.

Короткая 8-битная адресация регистров GPR

Обозначается 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 4-битного адреса. Для доступа к биту регистра GPR

слово адреса этого регистра вычисляется как описано выше, а позиция бита определяется с помощью независимого дополнительного 4-битного значения, см. рисунок 4.5.

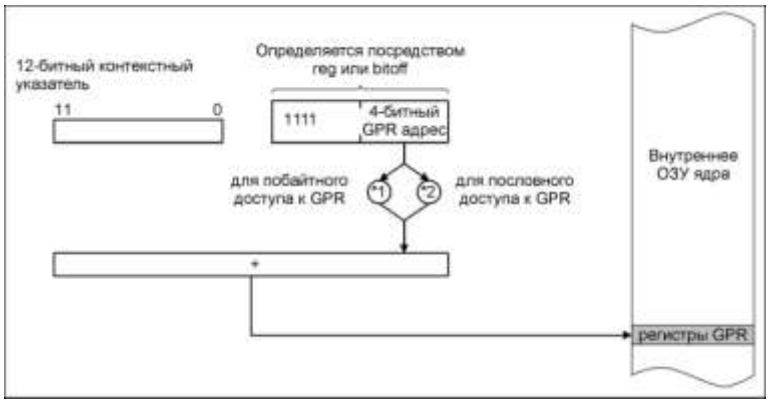


Рисунок 4.5 – Неявное использование регистра CP логикой режима короткой адресации

24-битный режим адресации

24-разрядные адреса в пределах диапазона от +00h до +30h могут использоваться для непосредственного доступа к регистрам GPR. Возможен пословный и побайтный доступ. 24-битный адрес формируется согласно правилам режима формирования длинного и косвенного адреса (см. таблицу 4.8). Первые восемь регистров с R0 по R7 доступны побайтно.

Таблица 4.8 – Режимы адресации для доступа к GPR-байтам

Регистр		Адрес		
слова	байта	физический	8-битный	4-битный
R15	–	+ 30h	FFh	Fh
R14	–	+ 28h	FEh	Eh
R13	–	+ 26h	FDh	Dh
R12	–	+ 24h	FCh	Ch
R11	–	+ 22h	FBh	Bh
R10	–	+ 20h	FAh	Ah
R9	–	+ 18h	F9h	9h
R8	–	+ 16h	F8h	8h
R7	RH7	+ 15h	FFh	Fh
	RL7	+ 14h	FEh	Eh
R6	RH6	+ 13h	FDh	Dh
	RL6	+ 12h	FCh	Ch
R5	RH5	+ 11h	FBh	Bh
	RL5	+ 10h	FAh	Ah
R4	RH4	+ 09h	F9h	9h
	RL4	+ 08h	F8h	8h
R3	RH3	+ 07h	F7h	7h
	RL3	+ 06h	F6h	6h
R2	RH2	+ 05h	F5h	5h
	RL2	+ 04h	F4h	4h
R1	RH1	+ 03h	F3h	3h
	RL1	+ 02h	F2h	2h
R0	RH0	+ 01h	F1h	1h
	RL0	CP	F0h	0h

4.7 Стек

Системный стек

Системный стек расположен в области DPRAM и 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 слов данных (байты должны быть дополнены до слов) 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.

Регистр SP используется для указания четного адреса вершины внутреннего системного стека. 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.

Младший бит регистра SP всегда ноль, а биты с 15 по 12 – всегда единицы, в связи с чем, диапазон доступных адресов F000h – FFFEh, что позволяет получить доступ к физическому стеку внутренней области DPRAM. Виртуальный стек (обычно больший, чем физический) может быть реализован программным способом.

Верхняя и нижняя границы стека определяются регистрами STKOV и STKUN.

Переполнение и опустошение стека (при использовании, например, команд ADD, SUB, PUSH или POP) обрабатывается с помощью специальной подпрограммы обслуживания ловушки. При переполнении 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 в регистр STKOV должно быть записано значение, представляющее сумму адреса желаемой верхней границы стека и смещения, являющегося максимальным размером стека, в регистр STKUN – значение, которое представляет собой верхний адрес основания стека.

Наихудшим случаем, который может случиться, является обнаружение состояния переполнения стека в момент входа в обслуживание прерывания 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.

Механизм ловушек не срабатывает, если значение SP изменяется прямо командой MOV или изменяются значения STKOV и STKUN, в связи с чем, SP находится вне новых границ.

Линейный стек

Формируется в случае использования в качестве системного стека всей области DPRAM (поле STKZ = 111b в регистров SYSCON). 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. Границы стека определяются регистрами STKUN и STKOV. Несмотря на то, что указатель стека SP может покрывать область от F000h до FFFEh, системный (физический) стек не должен выходить за границы области адресов от F600h до FDFEh.

Циркулярный (виртуальный) стек

Реализуются функции автоматического смещения и изменения.

По достижении границы переполнения стека, часть заносимых в стек данных должна быть сохранена во внешней памяти для формирования пространства для последующих сохранений – это механизм автоматического смещения.

По достижении границы опустошения стека, сохраненные ранее во внешней памяти данные должны быть возвращены – это механизм автоматического изменения.

Функции автоматического смещения и изменения используются очень редко поскольку операции наполнения и опустошения стека чередуются. Если использование стека не разрешено для программного обеспечения, то виртуальный стек не может использоваться.

Основной механизм – это аппаратная трансформация адресов виртуального стека, управляемых регистрами SP, STKOV и STKUN для определения области физического стека в пределах DPRAM (диапазон адресов от F000h до FFFEh). Размер области физического стека определяется полем STKSZ.

Адрес виртуального стека аппаратно трансформируется в адрес физического стека конкатенацией значащих битов регистра SP с дополнительными старшими битами адреса верхней границы физического стека FBFEh.

Значения регистров SP, STKOV, STKUN и поля STKSZ после сброса определяют область стека таким образом, что возможно его использование без каких-либо изменений, не выходя за границы 256 слов. Формирование физического адреса стека представлено на рисунке 4.6.

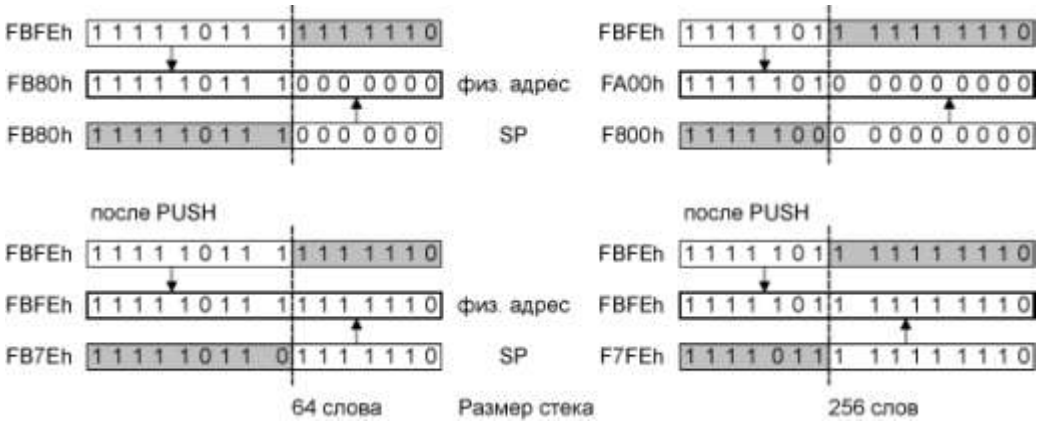


Рисунок 4.6 – Формирование физического адреса стека

Представленный ниже пример показывает реализацию механизма циркулярного стека, который также осуществляет разбиение адресного пространства. Сначала регистр R1 копируется в нижнюю часть физического стека, согласно выбранному максимальному размеру стека, после чего следующая команда заносит 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) 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.

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;

- 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.

5 Блок умножения-накопления

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.

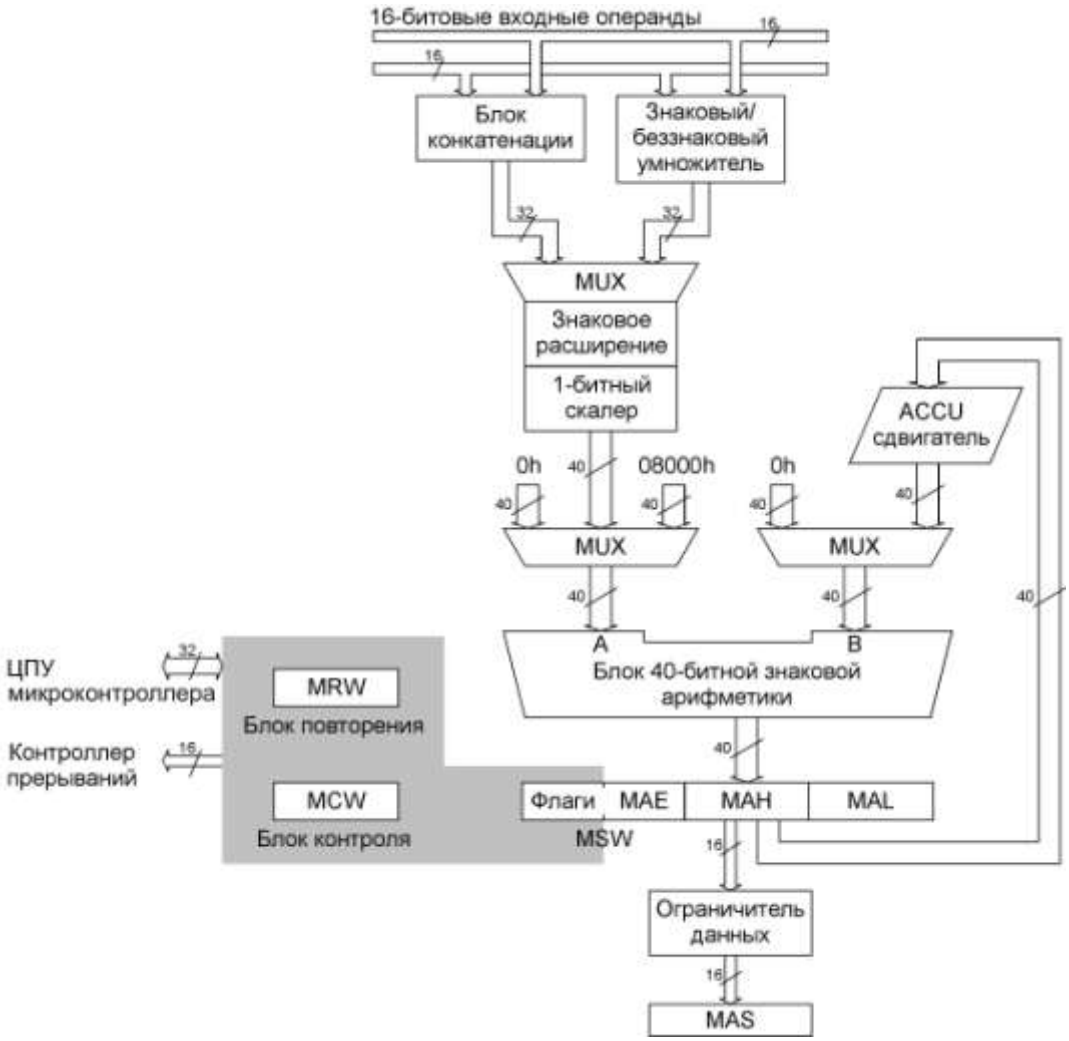


Рисунок 5.1 – Архитектура блока MAC

Блок MAC имеет особенности:

- параллельная пересылка данных, позволяющая пересылать данные одного операнда параллельно с выполнением команд умножения-накопления;
- 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 2^{13} раз с возможностью прерывания цикла;
- генерация прерываний (аппаратные ловушки класса B) при установке соответствующих флагов.

5.1 Операции блока MAC

Генерация адреса

Обработка команд блока MAC производится на 5-ступенчатом конвейере. При выполнении команд используются как режимы адресации с помощью прямого адреса регистра GPR или непосредственной константы #data4, так и косвенная адресация с последующим изменением указателя адреса.

Для двойной косвенной адресации требуется два указателя адреса. Любой регистр GPR может быть использован в качестве одного указателя адреса, в качестве другого указателя могут использоваться один или два регистра SFR – IDX0 или IDX1. Две пары регистров QR0/QR1 или QX0/QX1 связаны с каждым указателем адреса (GPR или IDXi). Указатели адреса регистра 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⊗] и [IDX_i⊗] относятся к данным режимам адресации.

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

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

Параллельная пересылка данных

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

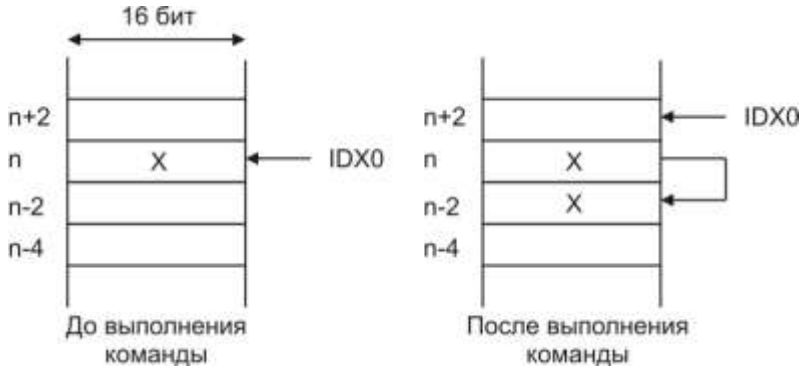


Рисунок 5.2 – Пример параллельной пересылки данных

Режим адресации CoReg

Аккумулятор и регистры управления MAL, MAH, MSW, MCW, MRW блока MAC могут быть адресованы стандартным набором команд, как любой регистр 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. Для 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

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 старшего значащего бита восемь раз. При выполнении команд с двумя числами без знака (например, CoMULu, CoMACu) происходит дополнение нулями результата, независимо от старшего значащего бита.

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 007FFFFFFFh (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 – младшее слово аккумулятора блока MAC;
- MAH – старшее слово аккумулятора блока MAC;
- MAE – расширение аккумулятора (значащий байт аккумулятора).

Доступ к регистрам осуществляется как к младшему байту регистра MSW.

При записи в MAH, записываемое значение 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. Аккумулятор превышает 32-битную границу и изменяет содержимое регистра MAE. В результате этого, в старших 8 битах аккумулятора содержатся значащие биты (без знака). Для обозначения этого расширения устанавливается флаг E, содержащийся в значащем байте статусного регистра MSW.

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. 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 до 2^{13} (8192) раз. Значение в счетчик повтора может устанавливаться как с помощью непосредственной константы (до 31), так и с помощью содержимого счетчика повтора (биты с 12 по 0) регистра MRW. 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 (15-й бит регистра повторения MRW), 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 блока MAC

Система команд блока MAC 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 (максимально 2^{13} раз).

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 блока MAC

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

Окончание таблицы 5.5

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

6 Организация 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. Пространство памяти доступно для данных размером в байт либо слово. Отдельная часть внутренней памяти доступна побитно.

Адресное пространство состоит из 256 сегментов по 64 Кбайт каждый. Каждый сегмент разделен на четыре страницы данных по 16 Кбайт, см. рисунок 6.1.

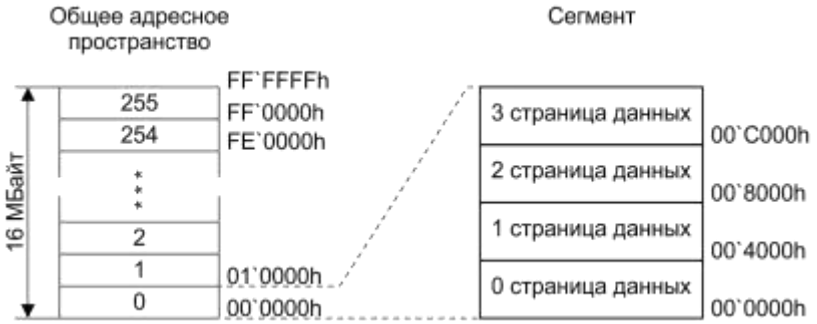


Рисунок 6.1 – Область памяти и адресное пространство

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

Общая карта памяти микроконтроллера показана в таблице 6.1. Запись в зарезервированные области памяти не возможна. Чтение из зарезервированных областей будет возвращать значение FFFFh.

Таблица 6.1 – Распределение адресов памяти

Адресная область	Начальный адрес	Конечный адрес	Объем памяти
Внешняя память	0D’A000h	FF’FFFFh	–
XSFR (ГОСТ 52070-2003)	0D’8000h	0D’9FFFh	8 Кбайт
XRAM	0D’4000h	0D’7FFFh	16 Кбайт
XSFR (CAN)	0D’0000h	0D’3FFFh	16 Кбайт
Зарезервировано	0B’1000h	0C’FFFFh	–
PSRAM	0A’D000h	0B’0FFFh	16 Кбайт
Зарезервировано	09’0000h	0A’CFFFh	–
Флеш-память программ	01’8000h	08’FFFFh	480 Кбайт
Внешняя память	01’0000h	01’7FFFh	32 Кбайта
SFR	00’FE00h	00’FFFFh	512 байт
DPRAM	00’F200h	00’FDFFh	3 Кбайта
ESFR	00’F000h	00’F1FFh	512 байт
XSFR (CAPCOM6)	00’E800h	00’EFFFh	2 Кбайта
Внешняя память	00’8000h	00’E7FFh	26 Кбайт
Флеш-память программ	00’0000h	00’7FFFh	32 Кбайта

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 (не присоединенных). 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, а позиция бита 15 имеет наибольшее значение в байте со следующим нечетным адресом. Битовая адресация поддерживается для регистров GPR и части регистров SFR и DPRAM.

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 информации в памяти

Выбор между внутренней и внешней памятью программ осуществляется посредством бита ROMEN (регистр SYSCON), который устанавливается аппаратно в зависимости от напряжения на входе EA# во время сброса, см. приложение Ж.

6.2 Внутренняя память

Области DPRAM

Память разграничена на внутреннюю память данных DPRAM и внутреннюю память области периферии. Области DPRAM и SFR занимают третью страницу данных нулевого сегмента и обеспечивают быстрый доступ с помощью указателя страницы данных DPP.

Область DPRAM используется для хранения:

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

Старшие 256 байт DPRAM (00’FD00h – 00’FDFFh) и текущий банк регистров области GPR предусматривают хранение единичных битов и являются побитно адресуемыми. 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.

Области SFR и ESFR

Отведены под регистры специальных функций управления ЦПУ, шинным интерфейсом, портами ввода-вывода.

Область XSFR

Области суммарным объемом 26 Кбайт отведены под управляющие регистры и область сообщений, подключенного к шине XBUS контроллера интерфейса CAN, ОЗУ контроллера интерфейса ГОСТ 52070-2003 и регистры блока CAPCOM6.

Область PSRAM

Отведена для хранения программ и может быть использована для хранения данных.

Область XRAM

Предназначена для хранения данных, и может быть использована для организации системного стека. Память XRAM связана с процессором 16-разрядной шиной данных.

Области XRAM и XSFR могут использоваться в качестве внешней памяти.

6.3 Флеш-память программ

Для хранения программного кода и данных используется 512 Кбайт встроенной флеш-памяти, начиная с адреса 00`0000h. Память не требует дополнительного источника напряжения для программирования. 64-битный доступ к программному коду позволяет извлекать две четырехбайтные или четыре двухбайтные команды. Операции стирания и программирования запускаются посредством командных последовательностей, что исключает непредвиденное изменение содержимого флеш-памяти.

Весь объем флеш-памяти делится на 512 равных секторов. Одной командной последовательностью можно стереть всю флеш или один сектор, а также записать одно слово. Во время процесса программирования/стирания чтение флеш-памяти аппаратно запрещено до окончания процесса.

Физическое пространство адресов флеш-памяти расположено в диапазоне 00’0000h - 07’FFFFh. Оно отражается на две области системной памяти: 00’0000h - 00’7FFFh и 01’8000h – 08’FFFFh.

Запись в чистую флеш-память возможна как с помощью программы, расположенной во внешней памяти, так и с помощью интерфейса JTAG. Пустые ячейки флеш-памяти при чтении возвращают значение FFFFh.

Флеш-память имеет две системы защиты (отключаются с помощью пароля):

- защиту от несанкционированной записи/стирания (от записи можно защитить или всю флеш-память, или заданное число секторов 8-го сегмента);
- защиту от внешнего доступа к содержимому, которая включает защиту от чтения данных и защиту от выполнения программного кода (внешний доступ – доступ из программы, запущенной по любым адресам вне флеш-памяти, а также доступ из отладчика OCDS).

Программирование и защита флеш-памяти

Запись и стирание флеш-памяти, а также включение защиты от непредвиденного изменения данных (например, во время включения/выключения питания) и отключение защиты производится посредством командных последовательностей. Командная последовательность состоит из нескольких последовательных записей заранее определенных данных по определенным виртуальным адресам. Если при вводе командной последовательности возникает ошибка данных и/или адреса, то в регистре FSR устанавливается флаг SQERR. Для продолжения работы необходим повторный ввод последовательности с самого начала.

В таблице 6.1 указаны командные последовательности для записи и стирания флеш-памяти.

Таблица 6.1 – Командные последовательности записи и стирания флеш-памяти

Операция		Циклы записи					
		1	2	3	4	5	6
Записать слово	Шина адреса	0555h	0AAAh	0555h	<Адрес>		
	Шина данных	AAh	55h	A0h	<Слово>		
Стереть сектор	Шина адреса	0555h	0AAAh	0555h	0555h	0AAAh	<Адрес>
	Шина данных	AAh	55h	80h	AAh	55h	30h
Полное стирание	Шина адреса	0555h	0AAAh	0555h	0555h	0AAAh	0555h
	Шина данных	AAh	55h	80h	AAh	55h	10h

Пример программы для записи слова:

```
MOV    R1,#05h       ; Сегмент
MOV    R2,#2579h     ; Адрес
MOV    R3,#55AAh     ; Данные
MOV    R4,#55AAh
EXTS   #0Bh,#04h
  MOVB   0555h,RL4
  MOVB   0AAAh,RH4
  MOVB   RL4,#0A0h
  MOVB   0555h,RL4
EXTS   R1,#01h
MOV    [R2],R3       ; Запись значения 55AAh по адресу 05’2579h
```

Диаграмма записи слова представлена на рисунке 6.3.

Рисунок 6.3 – Командная последовательность для записи слова

Пример программы для стирания сектора:

```
MOV    R1,#05h       ; Сегмент
MOV    R2,#2579h     ; Адрес, в котором биты с 15 по 10 являются номером
                     ; стираемого сектора указанного сегмента
MOV    R6,#55AAh
MOV    R7,#8030h
EXTS   #0Bh,#03h
  MOVB   0555h,RL6
  MOVB   0AAAh,RH6
  MOVB   0555h,RH7
EXTS   #0Bh,#02h
  MOVB   0555h,RL6
  MOVB   0AAAh,RH6
EXTS   R1,#01h
MOVB  [R2],RL7       ; Стирание 9 сектора 5 сегмента
```

Пример программы для стирания всей флеш-памяти:

```
MOV    R6,#55AAh
MOV    R7,#8010h
EXTS   #0Bh,#03h
  MOVB   0555h,RL6
  MOVB   0AAAh,RH6
  MOVB   0555h,RH7
EXTS   #0Bh,#03h
```

```
MOVB    0555h,RL6
MOVB    0AAAh,RH6
MOVB    0555h,RL7
```

В таблице 6.2 указаны командные последовательности для включения и отключения систем защиты флеш-памяти.

Таблица 6.2 – Командные последовательности для работы с защитой флеш-памяти

Операция		Циклы записи					
		1	2	3	4	5	6
Удалить защиту	Адрес	0555h	0AAAh	0555h	0555h	0AAAh	1111h
	Данные	AAh	55h	80h	AAh	55h	40h
Записать PROCON	Адрес	0555h	0AAAh	0555h	F000h		
	Данные	AAh	55h	A0h	<Слово>		
Записать PW1	Адрес	0555h	0AAAh	0555h	F008h		
	Данные	AAh	55h	A0h	<PW1>		
Записать PW2	Адрес	0555h	0AAAh	0555h	F010h		
	Данные	AAh	55h	A0h	< PW2 >		
Записать PW3	Адрес	0555h	0AAAh	0555h	F018h		
	Данные	AAh	55h	A0h	< PW2>		
Записать PW4	Адрес	0555h	0AAAh	0555h	F020h		
	Данные	AAh	55h	A0h	< PW4>		
Подтвердить установку защиты	Адрес	0555h	0AAAh	0555h	F400h		
	Данные	AAh	55h	A0h	8AFEh		
Временно снять защиту от записи	Адрес	0555h	0AAAh	0AAAh	0AAAh	0AAAh	0555h
	Данные	3Ch	<PW1>	<PW2>	<PW3>	<PW4>	05h
Временно снять защиту от чтения	Адрес	0555h	0AAAh	0AAAh	0AAAh	0AAAh	0555h
	Данные	3Ch	<PW1>	<PW2>	<PW3>	<PW4>	50h
Возобновить защиту	Адрес	0555h					
	Данные	5Eh					
Обновить FSR	Адрес	0AAAh					
	Данные	F0h					

Конфигурация защиты задается регистром PROCON. При вводе командной последовательности «записать PROCON» в четвертом цикле записи по шине данных передается значение, которое будет записано в регистр.

Для установки (включения) защиты следует ввести ряд командных последовательностей, включающих четыре слова пароля. Порядок ввода командных последовательностей:

- удалить защиту (обязательно!);
- записать PROCON;
- записать PW1 (первое слово пароля);
- записать PW2 (второе слово пароля);
- записать PW3 (третье слово пароля);
- записать PW4 (четвертое слово пароля);
- подтвердить установку защиты.

После этого следует прочитать состояние регистра FSR, чтобы удостовериться в том, что защита установлена. В результате корректного включения защиты должны быть установлены флаги RDPR и/или FTPR и/или WRPR, а бит PRWRN сброшен.

Если бит PRWRN окажется установленным, то рекомендуется повторно ввести командные последовательности включения защиты в следующем порядке:

- временно снять защиту от записи (если установлен бит WRPR и/или значение поля PRSECNUM не равно нулю);
- временно снять защиту от чтения (если установлен бит RDPR и/или бит FTPR);
- удалить защиту;
- записать PROCON;
- записать PW1 (первое слово пароля);
- записать PW2 (второе слово пароля);
- записать PW3 (третье слово пароля);
- записать PW4 (четвертое слово пароля);
- подтвердить установку защиты.

После сброса микроконтроллера и перед началом работы с флеш-памятью желательно проверять состояние памяти, считывая и анализируя значение регистра FSR.

Работа с флеш-памятью

Для получения внешнего доступа к защищенным данным и коду программы, необходимо ввести командную последовательность «временно снять защиту от чтения». Установленный бит DISRDPR будет указывать на то, что защита снята.

Для программирования защищенного участка флеш-памяти необходимо ввести командную последовательность «временно снять защиту от записи». Установленный бит DISWRPR будет указывать на то, что защита снята.

В случае ввода некорректного пароля при попытке снять защиту установится флаг PWERR, и повторить попытку можно будет только после аппаратного сброса.

Для включения защиты необходимо ввести командную последовательность «возобновить защиту».

Для полного отключения защиты предусмотрен следующий порядок ввода командных последовательностей:

- временно снять защиту от записи;
- временно снять защиту от чтения;
- удалить защиту.

Если пароль утерян, то снять защиту можно, только если аппаратно загрузить контроллер с 8-ого сегмента (при этом во время аппаратного сброса на выводах 2 и 3 порта P0 должен быть логический ноль) и ввести командную последовательность «удалить защиту». Следует помнить, что это приведет к полному стиранию содержимого флеш-памяти и последующему отключению защиты.

6.4 Внешняя память

Все адресное пространство, за исключением областей DPRAM и SFR, может использоваться как внешняя память. Обращение к внешней памяти осуществляется посредством контроллера внешней шины.

Внешняя память доступна пословно посредством косвенного или длинного 16-битного режима адресации, с использованием регистров DPPx. Любой доступ к данным осуществляется по четному адресу. Внешняя память не адресуется побитно.

При размещении кода программы на границе внутренней и внешней памяти следует использовать команды безусловного перехода, поскольку последовательный переход через границу не поддерживается и может привести к ошибочным результатам.

7 Система прерываний и ловушек

В микроконтроллере реализовано четыре механизма обслуживания запросов прерываний, формируемых внутренними и внешними источниками прерываний.

Стандартное обслуживание прерываний

ЦПУ временно приостанавливает выполнение текущей программы и переходит к выполнению подпрограммы обработки прерывания. Текущее состояние программы (регистры IP и PSW, а также в режиме сегментации и регистр CSP) сохраняется во внутреннем системном стеке. Обслуживание прерываний выполняется согласно их приоритетам.

Программные и аппаратные ловушки

Функции ловушек активируются в ответ на специальные состояния, которые могут возникать во время выполнения команд. Ловушка также может быть вызвана внешним немаскируемым прерыванием по выводу NMI# микроконтроллера. 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 стандартному программному обслуживанию прерываний. Реагируя на запрос прерывания, блок PEC совершает передачу одного слова или байта между двумя ячейками памяти через один из восьми программируемых каналов. Во время передачи данных выполнение программы приостанавливается. Внутренняя информация состояния программы не сохраняется. При обслуживании прерываний блоком PEC используется та же схема уровней приоритетов, что и для стандартного обслуживания прерываний.

7.1 Структура системы прерываний

В микроконтроллере реализовано 116 векторов прерываний с 16 группами приоритетов. В каждой группе 4 подуровня. Для того чтобы обеспечить возможность модульной и совместимой разработки программ, каждый канал прерывания или запроса PEC имеет специальный регистр xxIC и вектор прерываний. Регистр xxIC содержит флаг IR запроса прерывания, бит IE разрешения прерывания и поля ILVL и GLVL задания индивидуального и группового уровня приоритета прерывания, а также бит GP расширения группового приоритета. Регистры xxIC всех каналов прерываний идентичны, допускают побитную адресацию и доступны на запись и чтение.

Каждый запрос прерывания генерируется источником прерывания в результате возникновения заранее определенного события. В некоторых случаях несколько источников прерываний объединены в узлы для эффективного использования системных ресурсов. В случае возникновения нескольких запросов, ЦПУ обрабатывает прерывание с более высоким уровнем приоритета.

Реализована векторная система прерываний. Для векторов зарезервированы адреса в пространстве памяти.

Запросы прерываний с уровнем приоритета 14 и 15 (т. е. ILVL = 111xb) автоматически направляются для обслуживания в блок PEC за исключением случаев, когда поле COUNT соответствующего регистра PECCx содержит ноль. Младший бит поля ILVL и два бита поля GLVL задают номера канала PEC.

Арбитраж прерываний

В случае генерирования запроса прерывания в соответствующем регистре xxIC устанавливается флаг IR. Запрос на прерывание также может быть сгенерирован программной установкой флага IR. Одновременный опрос состояния всех регистров прерываний производится в начале каждого арбитражного цикла. Определение уровней приоритетов (арбитраж) запросов прерываний происходит в два этапа, см. рисунок 7.1.

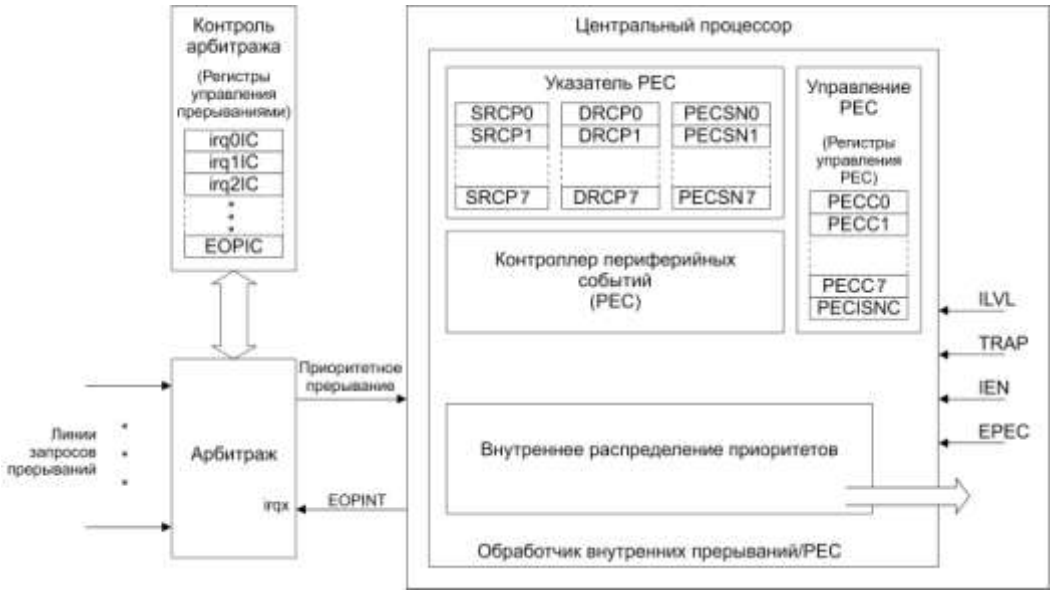


Рисунок 7.1 – Арбитраж прерываний

На первом этапе арбитражного цикла происходит сравнение индивидуальных и групповых уровней приоритетов запросов прерываний.

Примечание – Для источников прерываний, имеющих одинаковый уровень индивидуального приоритета, должны быть заданы различные уровни группового приоритета во избежание генерирования неверных векторов прерываний.

На втором этапе арбитражного цикла уровень приоритетного прерывания сравнивается с приоритетом текущей задачи ЦПУ.

В случае если уровень приоритета ожидающего прерывания выше уровня приоритета текущей задачи (поле ILVL регистра PSW) и установлен бит глобального разрешения прерываний IEN, то прерывание будет обслужено незамедлительно, а иначе поставлено в очередь. Если сброшен бит IEN, никакой запрос прерывания не будет обслужен.

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

Уровень приоритета 0000b устанавливается по умолчанию. Поэтому запрос с уровнем 0h не будет принят ЦПУ. Тем не менее, каждый запрос прерывания (включая и запросы с уровнем 0h) вызовет выход ЦПУ из режима простоя Idle, независимо от состояния бита IEN.

Таблица векторов прерываний

При появлении запроса прерывания ЦПУ совершает переход по вектору, который связан с источником прерывания, что позволяет напрямую идентифицировать источник.

Исключением являются аппаратные ловушки класса B, которые все определены через один вектор прерываний. Для определения источника в этом случае используются флаги регистра TFR.

Зарезервированные адреса векторов собраны в таблицу, которая расположена в адресном пространстве микроконтроллера и содержит команды перехода, передающие управление сервисной программе обслуживания прерываний (может быть расположена в

любом месте адресного пространства). Таблица векторов расположена в нулевом сегменте адресного пространства с начальным адресом 0000h. Каждому вектору отведено два слова, за исключением векторов сброса и аппаратных ловушек, занимающих четыре и восемь слов, соответственно.

Каждый вектор прерывания имеет регистр управления. Все регистры управления программно доступны для записи. Установка флага прерываний в регистре активирует соответствующий запрос прерывания (если он разрешен).

Таблица векторов прерываний представлена в приложении В.

Управление прерываниями посредством регистра PSW

Поле ILVL регистра PSW отражает текущий уровень приоритета исполняемой программы. При входе в подпрограмму обработки прерывания значение этого поля изменяется на значение уровня приоритета обслуживаемого прерывания. Перед этим в стеке сохраняется предыдущее значение регистра PSW. Если уровень приоритета запроса прерывания выше текущего значения ILVL, то прерывание будет обслужено. Любое прерывание с таким же уровнем или ниже останется без ответа. Текущее значение ILVL регистра PSW может быть изменено программно, что дает возможность управления прерываниями с низким приоритетом.

Передачи PEC при обслуживании прерываний не прерывают работу ЦПУ, а только занимают один такт. При этом не меняется значение поля ILVL в регистре PSW.

Во время выполнения подпрограммы обслуживания аппаратной ловушки (имеет максимальный 15 уровень приоритета) никакие прерывания или запросы PEC не обслуживаются до окончания выполнения этой подпрограммы. Следует помнить, что команда TRAP не изменяет уровень приоритета ЦПУ, поэтому программно вызванная подпрограмма обслуживания ловушки может быть прервана запросом с более высоким уровнем приоритета.

Бит IEN глобально разрешает или запрещает запросы на прерывания и операции PEC. После сброса бита IEN будут обработаны только те запросы прерываний, которые находятся в состоянии ожидания; никакие новые запросы приняты не будут.

Примечание – Состояние бита IEN не влияет на обслуживание ловушек, поскольку они являются немаскируемыми.

Сохранение текущего состояния программы

Перед началом обслуживания прерывания состояние текущей задачи автоматически сохраняется в системном стеке и хранится до окончания выполнения подпрограммы обслуживания прерывания.

Рисунок 7.2 – Сохранение состояния текущей задачи в системном стеке

Сначала сохраняется значение регистра PSW, а затем, в зависимости от режима (сегментированой или несегментированой памяти) регистра IP или регистров CSP и IP, см. рисунок 7.2. Эта последовательность оптимизирует использование системного стека при отключенной сегментации. Значение поля ILVL изменяется на значение приоритета обслуживаемого запроса на прерывания, после чего ЦПУ работает с новым уровнем приоритета.

После получения запроса прерывания соответствующий вектор загружается в IP (в режиме сегментированной памяти регистр CSP очищается), и далее первая команда подпрограммы обслуживания прерывания вызывается из адреса вектора. 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 прерванной задачи извлекается из системного стека в обратном порядке. Адрес возврата определяется с помощью регистра IP и в случае использования сегментированного режима памяти – указателя сегмента кода CSP.

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. Микроконтроллер позволяет переключать полный банк ЦПУ регистров (регистры области GPR) за одну команду, таким образом, подпрограмма обслуживания использует собственный независимый банк регистров GPR.

Команда SCXT отсылает содержимое контекстного указателя CP в системный стек и загружает в CP значение базового адреса нового банка регистров. 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 предыдущее значение CP просто извлекается из системного стека.

Примечания

1 Ресурсы, используемые подпрограммой обслуживания прерываний (в том числе DPP и регистры модуля умножения и деления), необходимо сохранять при входе в подпрограмму и необходимо восстанавливать при возврате в основную программу.

2 Первые две команды, следующие после команды SCXT, не должны использовать регистры GPR.

7.2 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 значения PSW, IP и CSP сохраняются во внутреннем системном стеке, после чего 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.

Примечание – Значение ILVL в регистре PSW не изменяется командой TRAP, поэтому подпрограмма обслуживания прерывания выполняется на том же уровне приоритета, что и основная программа, в связи с чем, 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. Аппаратная ловушка может быть вызвана преднамеренно (например, вводом некорректного кода).

При обнаружении ловушки ЦПУ переходит по адресу вектора ловушки. В зависимости от состояния обнаруженной ловушки, вызвавшая ее команда может быть либо завершена, либо не завершена, прежде чем будет запущена подпрограмма

обслуживания прерывания. Аппаратные ловушки не маскируемы и всегда имеют уровень приоритета выше, чем какое-либо другое состояние ЦПУ. В случае обнаружения в одном и том же командном такте нескольких аппаратных ловушек, будет обслужена аппаратная ловушка с наивысшим уровнем приоритета. При активации аппаратной ловушки запускается команда TRAP, в результате выполнения которой содержимое регистров PSW, IP и CSP сохраняется во внутреннем системном стеке, уровень приоритета ЦПУ устанавливается в максимально возможное значение и при этом запрещаются все прерывания, после чего происходит переход по адресу ловушки. В режиме сегментированной памяти в регистре CSP указывается нулевой сегмент. Подпрограмма обслуживания ловушки завершает свою работу по команде RETI.

Микроконтроллер обслуживает девять аппаратных ловушек, подразделяющихся на два класса A и B, см. таблицу 7.1. Ловушки класса A имеют один уровень приоритета, при этом каждая ловушка имеет индивидуальный вектор. Ловушки класса B имеют один уровень приоритета и один адрес вектора.

При возникновении аппаратной ловушки устанавливается соответствующий флаг в регистре TFR (флаги в регистре могут устанавливаться программно). Регистр TFR позволяет подпрограмме прерываний определить тип активной ловушки.

Примечание – Подпрограмма обслуживания ловушки должна сбрасывать соответствующий флаг в регистре TFR, во избежание повторного запуска подпрограммы.

Таблица 7.1 – Параметры ловушек

Ловушка		Параметры ловушки			
		Флаг	Вектор	Номер	Приоритет
Аппаратный сброс Программный сброс Переполнение сторожевого таймера		–	RESET	00h	IV
Ловушка отладки		DEBUG	DEBTAP	08h	III
Класс A	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	Неопределенный программный код	UNDOPC	BTRAP	0Ah	I
	Ошибка защиты	PRTFLT			
	Неверный доступ к операнду слова	ILLOPA			
	Неверный командный доступ	ILLINA			
	Неверный доступ к внешней шине	ILLBUS			

Ловушки аппаратного и программного сброса, а также ловушка переполнения сторожевого таймера активируются при сбросе микроконтроллера и имеют наивысший четвертый приоритет.

Ловушка отладки имеет следующий по уровню приоритет и позволяет отладчику прерывать обслуживание ловушек аппаратных средств и аппаратные прерывания.

Ловушки класса A активируются системными событиями NMI и специальными событиями ЦПУ. Ловушки класса A не используются для указания отказов аппаратных средств. В случае одновременной активации нескольких ловушек класса A, происходит их распределение по приоритетам. После окончания обслуживания ловушки с высшим приоритетом значение IP считывается из стека и начинается обслуживание следующей ловушки.

При обнаружении отрицательного фронта на входе NMI# (немаскируемое прерывание) устанавливается флаг NMI в регистре TFR, и запускается подпрограмма обслуживания ловушки.

Всякий раз, когда значение указателя стека SP становится меньше значения регистра STKOV, устанавливается флаг STKOF. В зависимости от операции, вызвавшей декремент SP, значение IP, помещенное в системный стек, будет являться:

- адресом следующей команды (декремент с помощью команды PUSH или CALL, или прерывания ловушки);
- адресом первой или второй команды после команды вычитания (декремент командой вычитания).

Во избежание переполнения стека, он должен содержать достаточно места для двухкратного сохранения состояния системы. В противном случае должен быть произведен системный сброс.

Всякий раз, когда значение указателя становится больше содержимого регистра STKUF, устанавливается флаг STKUF.

В зависимости от операции вызвавшей инкремент SP значение IP, помещенное в системный стек, будет являться:

- адресом следующей команды (инкремент с помощью команды POP или команды возврата);
- адресом первой или второй команды после команды сложения (инкремент командой сложения).

Ловушки класса B активируются неисправимым отказом аппаратных средств и предполагают немедленный запуск подпрограммы обслуживания отказа центральным процессором. Ловушка класса B может прервать выполнение последовательности команд ATOMIC/EXTEND. После выполнения подпрограммы обработки ловушки прерванная текущая инструкция не может быть восстановлена.

В случае когда ловушки классов A и B активируются одновременно, устанавливаются оба флага ловушек. Если это происходит во время выполнения последовательности EXT-команд или команды ATOMIC, ловушка класса B прерывает выполнение последовательности, и ЦПУ немедленно запускает подпрограмму обслуживания ловушки класса A. После выполнения подпрограммы обработки ловушки значение 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 указателя IP команды при которой возникла ловушка. Все ловушки класса B имеют одинаковый уровень приоритета. При одновременной активизации нескольких ловушек класса B регистре TFR устанавливаются соответствующие флаги, и запускается подпрограмма обработки ловушки (приоритет распределяется программно). Во время выполнения подпрограммы обслуживания ловушки класса A, ни одна ловушка класса B не будет обслужена до тех пор, пока не завершится выполнение подпрограммы. Флаг ловушки класса B сохраняется в TFR, а значение указателя IP команды, при которой возникла ловушка, теряется.

Если при дешифрации текущей команды ЦПУ определяет неверный код команды, устанавливается флаг UNDOPC. 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.

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. Защищенными являются команды DISWDT, EINIT, IDLE, PWRDN, SRST и SRVWDT. Значение IP помещается в системный стек перед выполнением подпрограммы и затем возвращается при выходе из подпрограммы.

Всякий раз, когда при записи или чтении слова ЦПУ обращается к нечетному адресу байта, устанавливается флаг ILLOPA. Значение IP, помещенное в системный стек, является адресом команды после той, которая вызвала активацию ловушки.

Всякий раз, когда переход выполняется к нечетному адресу байта, устанавливается флаг ILLINA. Значение IP, помещенное в системный стек, является неправильным нечетным адресом команды перехода.

Всякий раз при получении команды чтения или записи данных с внешней шины, когда конфигурация шины не была определена, устанавливается флаг ILLBUS. Значение IP, помещенное в системный стек, является адресом команды следующей за той, которая вызвала активацию ловушки.

7.3 Контроллер периферийных событий

Контроллер периферийных событий (PEC) реализует способ обработки запросов прерываний без участия ЦПУ, который заключается в передаче по каналу данных между двумя точками памяти микроконтроллера за один машинный цикл (время, в течение которого ЦПУ приостанавливает выполнение программы без сохранения состояния системы в стеке). Блок PEC управляет восемью каналами.

Передача по каналу PEC – самый быстрый ответ на прерывание при условии, что уровень приоритета этого прерывания выше, чем текущий уровень приоритета ЦПУ. Во многих случаях такой передачи достаточно, чтобы обслужить периферийный запрос на прерывание.

Каждый канал PEC в зависимости от настройки реализует:

- пересылку байта или слова;
- непрерывную пересылку данных;
- автоматический инкремент адреса источника или приемника данных.

В целом блок PEC позволяет организовывать линии связи между двумя любыми каналами, а также генерировать запрос прерывания только от одного выбранного канала после завершения передачи или от всех каналов.

Управление каналами передачи данных

Для управления каналом x (x от 0 до 7) используется регистр PECCx, в котором задаются режим передачи данных, параметры передаваемых данных и указателей адресов, уровень приоритета прерывания канала.

Бит BWT задает формат данных и шаг изменения значения указателя адреса источника или приемника по окончании передачи. Если изменение адреса источника и/или приемника данных не требуется, то это задается значением 00b в поле INC.

24-разрядные указатели адреса источника и приемника определяют местоположения регистров, между которыми должны быть перемещены данные. Старшие 8 бит (номер сегмента) и младшие 16 бит адреса сохраняются в регистрах PECSNx, SRCPx и DSTPx, см. рисунок 7.3. Изменяться может только младшая часть адреса – SRCPx. Изменение номера сегмента запрещено. Если указатель адреса смещения примет значение FFFFh/FFFEh в случае передачи байта/слова, то последующий инкремент значения приведет к переполнению указателя адреса.

Если для канала выбрана передача слова данных, то указатели адресов источника и приемника должны содержать правильный адрес слова во избежание срабатывания соответствующей ловушки.

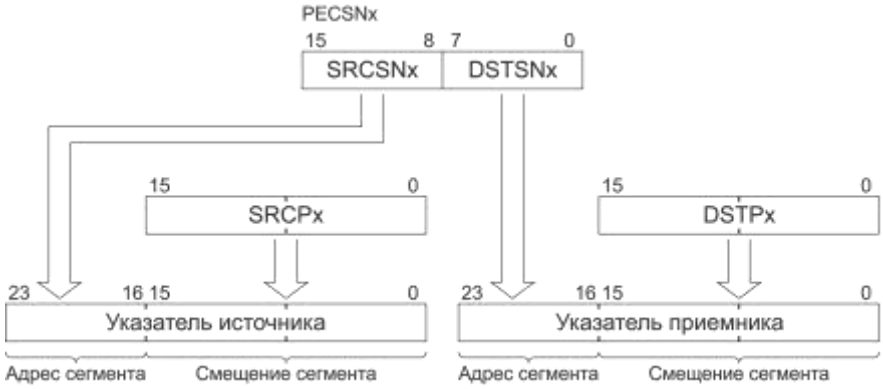


Рисунок 7.3 – Формирование адреса источника и приемника канала x

Блок PEC осуществляет передачу данных в каждом из каналов в двух режимах – короткой и длинной передачи. Выбор режима осуществляется битом PT регистра PECCx.

В режиме короткой передачи контроль осуществляется полем COUNT, показывающим состояние счетчика передач канала. Если задано COUNT = 00h, то передачи PEC запрещены, а если COUNT = FFh, то разрешено неограниченное количество передач. В остальных случаях поле COUNT задает количество запросов, которые должны быть обслужены. По окончании обслуживания каждого запроса значение COUNT декрементируется, и сбрасывается флаг запроса. Когда счетчик передач достигает нуля (т. е. COUNT = 00h), вырабатывается запрос прерывания по окончании передач PEC. В случае если бит EOPINT сброшен, запрос прерывания имеет тот же уровень приоритета, что и передача, или другой уровень приоритета, если бит EOPINT установлен.

Если EOPINT установлен, то в момент переключения счетчика передач из 01h в 00h сбрасывается флаг запроса прерывания. В случае если EOPINT сброшен, то флаг запроса прерывания не сбрасывается, и следующий запрос прерывания будет обслужен с тем же уровнем приоритета.

В режиме длинной передачи контроль осуществляется полем COUNT2 регистра PECXCx, показывающим состояние счетчика передач канала. Состояние бита CL не важно, а значение COUNT должно быть 00h. Если задано COUNT2 = 0000h, то передачи PEC запрещены. В остальных случаях поле COUNT2 задает количество запросов, которые должны быть обслужены. По окончании обслуживания каждого запроса значение COUNT2 декрементируется, и сбрасывается флаг запроса. Когда счетчик передач достигает нуля (т. е. COUNT2 = 0000h), вырабатывается запрос прерывания по окончании передач PEC. В случае если бит EOPINT сброшен, запрос прерывания имеет тот же уровень приоритета, что и передача, или другой уровень приоритета, если бит EOPINT установлен.

Если EOPINT установлен, то в момент переключения счетчика передач из 0001h в 0000h сбрасывается флаг запроса прерывания. В случае если EOPINT сброшен, то флаг запроса прерывания не сбрасывается, и следующий запрос прерывания будет обслужен с тем же уровнем приоритета.

Режим попарной работы каналов

Режим, в котором два канала объединяются в пару. Передача данных в этом случае разделена на отдельно управляемые блочные пересылки и осуществляется каналами пары попеременно друг за другом – по окончании передачи блока данных в одном канале пары автоматически начинается передача следующего блока данных в другом канале.

Режим включается установкой бита CL в регистре PECCx. Соединение каналов в пару разрешено только при условии, что биты CL обоих каналов установлены. По умолчанию аппаратно определены четыре пары каналов – «0 и 1», «2 и 3», «4 и 5», «6 и 7». Передача данных всегда начинается с четного канала пары (этот же канал определяет и приоритет пары).

Как только блок данных полностью передан бит CL ранее активного канала сбрасывается, после чего происходит автоматическое переключение обработки на другой канал пары.

Каждый канал имеет флаг, сигнализирующий ЦПУ об окончании передачи PEC. После завершения передачи для возобновления работы канала требуется установить бит CL. Запрос прерывания по окончании передачи PEC разрешается и индицируется в регистре PECISNC.

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 фиксирует появление прерывания.

Если сброшен бит CL в регистре предыдущего канала и содержимое счетчика передач (COUNT) активного канала равно нулю, то передача данных считается законченной и генерируется запрос прерывания об окончании передачи.

Арбитраж

Программирование уровня приоритета прерываний необходимо для случаев одновременного поступления запросов прерываний по окончании передач PEC. Запросы прерываний для всех каналов объединяются в один узел прерываний, управление которым осуществляет регистр управления EOPIC.

Каждому каналу может быть назначен собственный уровень приоритета в регистре PECCx. Возможные комбинации указаны в таблице 7.3.

Таблица 7.3 – Комбинации значений поля PLEV и уровни приоритета

Поля регистра EOPIC		Выбранный PEC канал			
ILVL	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

7.4 Быстрые внешние прерывания

Внешние прерывания являются альтернативными функциями выводов микроконтроллера, которые включаются посредством соответствующих регистров. С каждым быстрым внешним прерыванием связаны три вывода, для которых реализованы три альтернативные функции EXxIN, EXxINA и EXxINB. События, активирующие прерывания могут комбинироваться, что задается полем, соответствующим прерыванию в регистре EXISEL0 или EXISEL1.

Выборка на выводах быстрых внешних прерываний производится каждый системный такт. Событие на входе микроконтроллера, активирующее прерывание, задается индивидуально, посредством регистра EXICON.

Быстрые внешние прерывания используют вектора прерываний каналов блока CAPCOM1 (с 8 по 15), поэтому не могут использоваться функции захвата/сравнения каналов, связанных с выводами, для которых не запрещены быстрые внешние прерывания. Тем не менее, порт 2 может работать в обычном режиме. Восемь прерываний управляются теми же регистрами, что и прерывания каналов блока CAPCON – регистрами CC8IC – CC15IC.

8 Генератор тактовых сигналов

Блок генератора тактовых сигналов микроконтроллера включает в свой состав основной осциллятор OSC, вспомогательный осциллятор, программируемый генератор частоты PLL и схему распределения синхросигнала.

Блок генератора формирует четыре основных сигнала:

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

Дополнительной схемой формируется независимый медленный тактовый сигнал FRTC, поступающий на блок часов реального времени RTC.

Структурная схема формирования синхросигналов показана на рисунке 8.1.

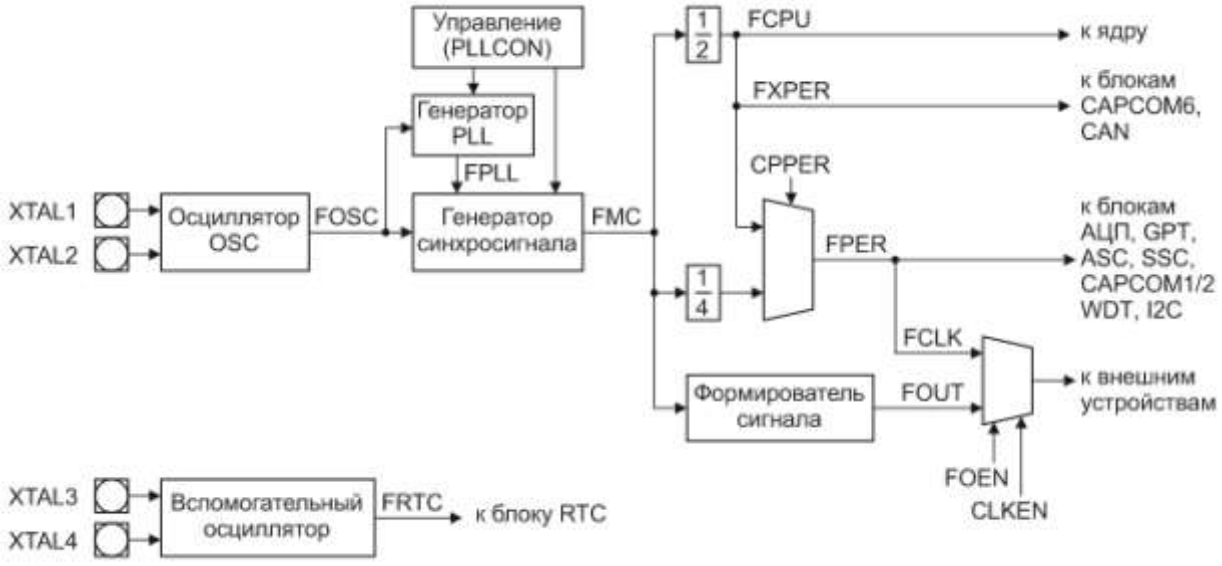


Рисунок 8.1– Схема формирования основных тактовых сигналов

8.1 Осцилляторы

К основному осциллятору OSC может быть подключен внешний кварцевый резонатор или внешний генератор тактового сигнала. При работе от внешнего генератора, входной сигнал подается на вход XTAL1 микроконтроллера (с частотой до 120 МГц), а на выходе осциллятора формируется синхросигнал FOSC с частотой равной входной, деленой на два. Осциллятор OSC запускается во время системного сброса.

Вспомогательный независимый осциллятор используется для формирования медленного тактового сигнала, который синхронизирует работу блока часов реального времени RTC. Вспомогательный осциллятор оптимизирован для работы на частоте 32,768 кГц и может работать от сигнала внешнего генератора подключенного на вход XTAL3. Управление работой вспомогательного осциллятора, а также контроль частоты периферийного тактового сигнала FPER осуществляется посредством регистра GENCON.

При переходе в режим пониженного энергопотребления Power_Down основной осциллятор автоматически выключается, а вспомогательный продолжает работу.

На рисунке 8.2 показаны схемы подключения внешних кварцевых резонаторов основного и вспомогательных осцилляторов

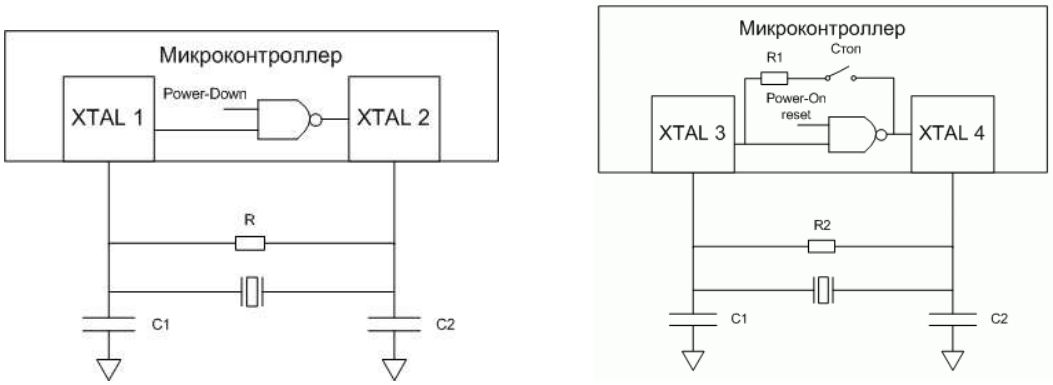


Рисунок 8.2 – Схемы подключения внешних кварцевых резонаторов.

Значения емкостей C1 и C2 выбираются в диапазоне от 10 до 30 пФ, номинал резистора R = (3,5 – 5) МОм. Значения емкостей C3 и C4 выбираются в диапазоне от 10 до 30 пФ, номинал резистора R2 = (3,5 – 5) МОм. Вспомогательный осциллятор содержит внутренний резистор R1 обратной связи, который можно подключать вместо внешнего резистора обратной связи.

8.2 Тактовый генератор PLL

Блок PLL умножает частоту тактового сигнала, поступающего с выхода осциллятора OSC, на программируемый коэффициент, задаваемый в регистре управления. Блок PLL может быть отключен (при этом генератор переходит на синхросигнал FOSC). Управление PLL осуществляется посредством регистра PLLCON.

Блок PLL может функционировать в двух режимах – прямой передачи частоты и синтеза частоты. При переходе в режим пониженного энергопотребления Power_Down блок PLL выключается. При переключении в режим синтеза частоты или изменении коэффициентов деления/умножения требуется.

Режим прямой передачи

Если во время сброса микроконтроллера сигнал на выводе RD# находится в состоянии логического нуля, то блок PLL переключается в режим прямой передачи. Входной сигнал FOSC передается на выход блока без изменений.

Режим синтеза частоты

Если во время сброса микроконтроллера сигнал на выводе RD# находится в состоянии логической единицы, то блок PLL переключается в режим синтеза частоты. Коэффициенты M_VALUE и N_VALUE задаются комбинацией сигналов на выводах P0H.5, P0H.6, P0H.7 микроконтроллера и определяют коэффициент умножения входной частоты сигнала FOSC. Бит P регистра PLLCON включает делитель на четыре. Расчет значения частоты выходного сигнала FPLL генератора PLL производится по формуле:

$$FPLL = FOSC \times \frac{M\_VALUE}{N\_VALUE} \times \frac{1}{(1+P)^2}. \quad (8.1)$$

В таблице 8.1 указаны комбинации состояний выводов микроконтроллера и соответствующие коэффициенты умножения.

При программировании коэффициентов деления следует учитывать ограничения:

2 МГц < fosc < 50 МГц;

12,5 КГц < fpll < 150 МГц.

Таблица 8.1 – Значения выходной частоты блока PLL при выключенном делителе (P= 0b)

Выводы порта			Поля регистра PLLCON		Частота на выходе
P0H.7	P0H.6	P0H.5	N_VALUE	M_VALUE	
0	0	0	02h	03h	fosc × 1,5
0	0	1		04h	fosc × 2
0	1	0		05h	fosc × 2,5
0	1	1		06h	fosc × 3
1	0	0		07h	fosc × 3,5
1	0	1		0Ah	fosc × 5
1	1	0		0Ch	fosc × 6
1	1	1		10h	fosc × 8

Прерывания генератора PLL

При нахождении или потере частоты PLL могут вырабатываться запросы прерываний управляемые регистрами PLLLOCK_IC и PLLUNLOCK_IC, см. приложение В.

8.3 Генерация внешнего тактового сигнала

Для тактирования внешних устройств в микроконтроллере предусмотрена возможность передачи синхросигнала через вывод P3.15. Вывод имеет две альтернативные функции – выход «системный тактовый сигнал» (включается установкой 15 бита регистра ALTSEL0P3) и выход «программируемая частота» (включается установкой 15 бита регистра ALTSEL1P3). В качестве выходных синхросигналов используются системный сигнал FXPER и сигнал с программируемой частотой FOUT.

Сигнал FXPER разрешается установкой бита CLKEN в регистре SYSCON. Для разрешения сигнала FOUT следует сбросить бит CLKEN и установить бит FOEN в регистре FOCON.

Сигнал FOUT формируется на основе сигнала FMC. Программирование осуществляется посредством регистра FOCON. Расчет значения частоты выходного сигнала FOUT производится по формуле:

$$\mathrm{FOUT} = \frac{\mathrm{FMC}}{(\mathrm{FORV}+1) \times 2^{(1-\mathrm{FOSS})}}. \qquad (8.2)$$

9 Блок сторожевого таймера

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 произойдет аппаратный сброс микроконтроллера (WDT-сброс). При этом на выходе RSTOUT# будет установлен уровень логического нуля, что приведет к сбросу подключенных к микроконтроллеру внешних устройств. Если сброс по сигналу сторожевого таймера запрещен, то при переполнении генерируется только прерывание (WDT-прерывание). Это позволяет использовать сторожевой таймер как генератор периодических прерываний.

Сторожевой таймер представляет собой 16-разрядный счетчик. Управление сторожевым таймером осуществляется посредством побитно адресуемого регистра WDTCON (содержимое этого регистра желательно перезаписывать каждый раз перед началом обращения к сторожевому таймеру). Входным синхросигналом является тактовый сигнал FPER. Коэффициент деления частоты входного сигнала задается битами WDTPRE и WDTIN (установлены по умолчанию). При каждом перезагрузке сторожевого таймера происходит перезагрузка значением WDTREL старшего слова счетчика и обнуление младшего.

Режим включенного таймера

После любого сброса сторожевой таймер начинает считать от значения 0000h с частотой fper/256 (если сторожевой таймер не заблокирован командой DISWDT, он считает и в режиме Idle). По достижении значения FFFFh сторожевой таймер устанавливает флаг WDTR, генерирует запрос WDT-прерывания (WDTINT) и активирует WDT-сброс микроконтроллера. Если установлен бит TIMEN, то WDT-сброс запрещен после выполнения команды DISWDT.

Избегать переполнения сторожевого таймера следует программно, при помощи защищенной 32-разрядной команды SRVWDT, которая вызывает перезагрузку счетчика таймера значением WDTREL. Выполнение команды SRVWDT не зависит от выполнения команд EINIT и DISWDT.

Режим выключенного таймера

Если бит TIMEN сброшен и генерирование WDT-сброса остановлено исполнением команды DISWDT, то сторожевой таймер выключен (счетчик остановлен).

Команда DISWDT является защищенной 32-разрядной командой и может быть выполнена только в промежуток времени от сброса до выставления флага EINIT или команды SRVWDT. Любая из этих двух команд блокирует выполнение DISWDT. WDT-сброс не завершит текущий цикл внешней шины до начала внутреннего сброса.

Сброс микроконтроллера

При аппаратном сбросе программный сброс и WDT-сброс блокируются. Флаги SWR и WDTR не выставляются.

При программном сбросе в результате выполнения команды SRST устанавливается флаг SWR.

10 Внутренняя и внешняя шины

10.1 Внутренняя шина XBUS

Для каждого периферийного блока, подключенного к шине XBUS (X-периферия), имеется отдельное адресное окно, управляемое парой регистров XBCONx/XADRSx (подобно регистрам BUSCONx и ADDRSELx). Поскольку регистровая пара управляет объединенной периферией быстрее, чем через внешнее управление, она фиксируется программной маской вместо того, чтобы быть запрограммированной пользователем.

Шина XBUS обеспечивает доступ к X-периферии с разрядностью шины 8 или 16 бит с поддержкой демультиплексного режима.

Разрешение X-периферии

Доступ к X-периферии 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 все устройства, подключенные к шине XBUS, запрещены. X-периферия не может использоваться, если это не разрешено битом XPEN регистра SYSCON (если периферия запрещена, то регистры периферийных блоков не доступны). Регистр XPERCON дополнительно определяет, какая периферия разрешена или запрещена и доступен до выполнения команды EINIT. Регистр XPERCON должен быть запрограммирован до установки бита XPEN. В микроконтроллере используются четыре внутренних сигнала выбора устройств, управляемых битами регистра XPERCON:

- XCS1 для интерфейса CAN;
- XCS2 для 16 Кбайт XRAM;
- XCS3 для 8 Кбайт интерфейса ГОСТ Р 52070-2003;
- XCS4 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

Все доступы к внешней памяти выполняются встроенным контроллером внешней шины EBC. Он может быть запрограммирован на работу без внешней памяти или на работу с внешней памятью в одном из четырех режимов с 16-/18-/20-/24-разрядным адресом:

- 16-разрядные данные, демультиплексная шина;
- 16-разрядные данные, мультиплексная шина;
- 8-разрядные данные, демультиплексная шина;
- 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 сигналов CSx# (четыре сигнала «выбор окна» плюс сигнал, задаваемый по умолчанию) могут быть сформированы для поддержки внешней логики. Внешние модули могут быть прямо подсоединены к общей шине адресов/данных и их отдельным линиям.

Доступ к медленным запоминающим устройствам или модулям с изменяющимися временами доступа поддерживается через специальную функцию «READY». Активный уровень управляющего входного сигнала разрешает выборку.

Быстродействие и интерфейс периферийных устройств может 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 EBC устанавливается регистрами 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).

10.3 Режим работы без внешних устройств

Микроконтроллер начинает работать в режиме без внешних устройств в том случае, если в момент системного сброса на контакте EA# был зафиксирован сигнал уровня логической единицы. При этом в регистр BUSCON0 записывается значение 0000h. Регистры BUSCON1 – BUSCON4 после сброса находятся в состоянии 0000h. Таким образом, контроллер внешней шины EBC отключается.

В этом режиме доступ к внешним устройствам невозможен и используются только внутренние запоминающие устройства микроконтроллера. Поэтому порты P0, P1, P4 и P6 могут использоваться для программного ввода-вывода.

Выполнение программы начинается из внутреннего ПЗУ. Попытка обращения к внешней памяти в этом режиме вызывает аппаратное прерывание ошибочной ситуации ILLBUS.

Обращения по внешней шине можно в дальнейшем разрешить, установив хотя бы один бит BUSACT в значение 1. После этого контроллер EBC включается, и можно осуществлять чтение и запись в пределах разрешенного адресного окна. Остальные параметры шины также необходимо настроить в соответствующем регистре BUSCONx.

10.4 Режимы работы внешней шины

Интерфейс внешней шины микроконтроллера используется в том случае, если хотя бы в одном из регистров BUSCONx установлен бит BUSACT. При обращении к адресному окну режим работы внешней шины устанавливается в зависимости от значения битового поля BTYP.

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 (16 Мбайт адресного пространства). 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 или 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 или A19 – A16, или A17 – 16) 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.

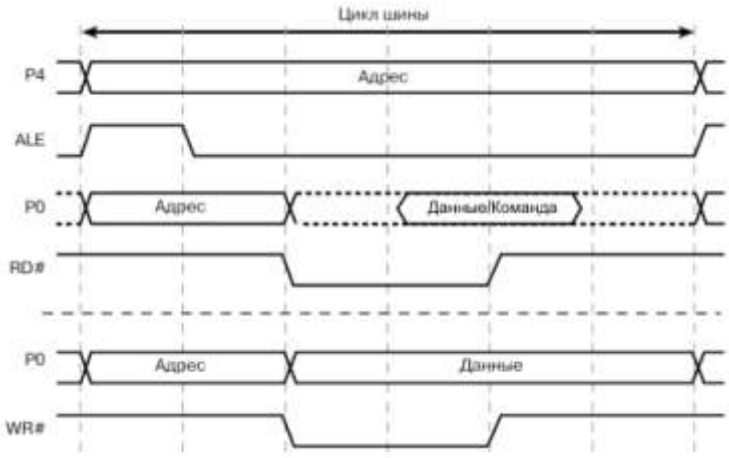


Рисунок 10.1 – Цикл мультиплексной шины

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 или A19 – A16 или A17, A16 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.

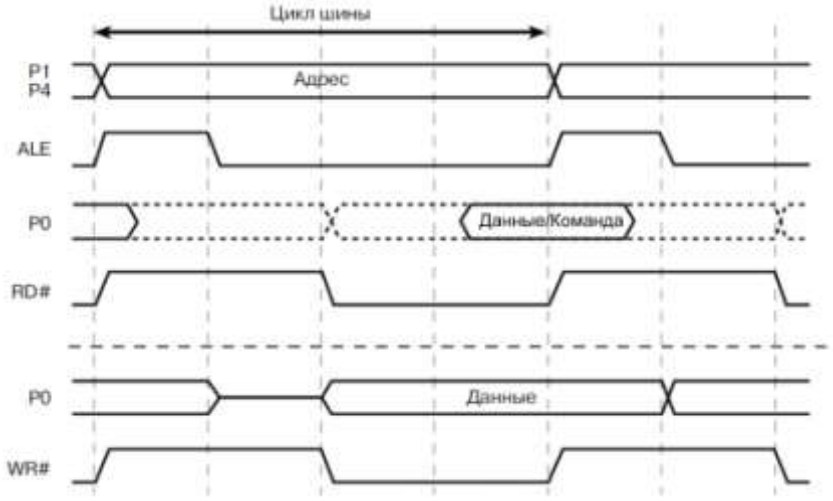


Рисунок 10.2 – Цикл демультиплексной шины

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

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 BSWC в регистре BUSCONx для адресного окна, из которого производится переключение в другое адресное окно. Разрешение вставки не влияет на быстродействие, если производится последовательное обращение к одному и тому же адресному окну.

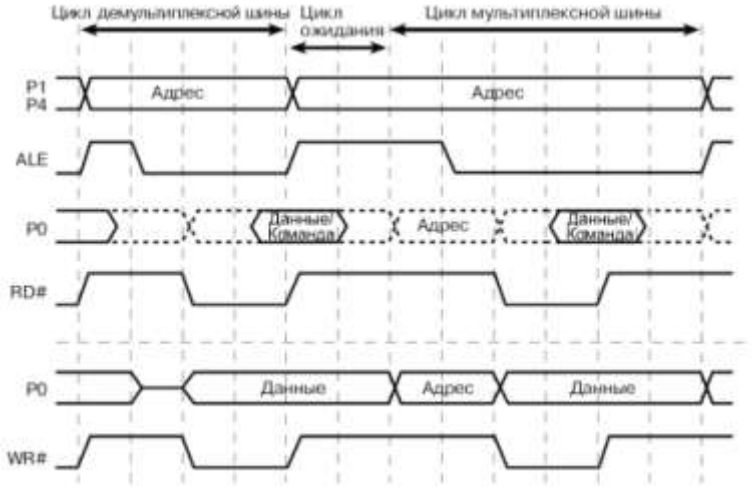


Рисунок 10.3 – Переключение от демультиплексной шины к мультиплексной

10.5 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, однако, быстродействие системы также сокращается.

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:

- старший байт выбирается сигналом BHE#, младший байт выбирается адресным сигналом A0, и оба байта могут пересылаться во внешнее устройство независимо друг от друга или вместе в 16-разрядном режиме.

- контроллер EBC самостоятельно формирует оба сигнала записи, что позволяет сократить внешнюю логику, объединяющую сигнал WR# с сигналами BHE# и A0 (этот 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)

Бит WRCFG регистра SYSCON позволяет определить функции сигналов WR# и BHE#. Если бит WRCFG установлен 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 следует учитывать 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	K = 1,5/3/6, очень низкая	P1H, P1L
8-разрядная демультиплексная	K = 1/2/4, низкая	P0H
16-разрядная мультиплексная	K = 1,5/1,5/3, высокая	P1H, P1L
16-разрядная демультиплексная	K = 1/1/2, очень высокая	–

Использование сигнала BHE#

Сигнал BHE# автоматически разрешается 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 (регистр SYSCON) установлен, и вывод сигнала BHE# запрещается. В этом случае канал P3.12 может использоваться как стандартный вывод. 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 или A19 – A16, или A17, A16 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.

Размер адресуемой внешней памяти может увеличиваться за счет использования нескольких банков, каждый из которых выбирается сигналами CSx# (x от 0 до 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 нуля) 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. Сигналы CSx# остаются неизменными до тех пор, пока не потребуется доступа к другому адресному окну для чтения/записи.

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# в неактивное состояние.

Режим работы каждого из сигналов CSx# устанавливается при помощи битов CSWEN и CSREN 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 состояние логического нуля 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 сброшенном бите CSCFG сигнал CSx# 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 установленном бите CSCFG сигнал CSx# 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, 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 нуля) для сигнала CS0# устанавливается режим формирования для всего цикла шины.

Во время сброса внутренние подтягивающие цепи удерживают сигналы CSx# в состоянии логической единицы. 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 Кбайт, 256 Кбайт, 1 Мбайт или 16 Мбайт. Сигналы выборки внешних устройств CSx# позволяют подключать банки памяти и периферию к микроконтроллеру без дополнительной внешней логики.

Если на внешнюю шину выводится только младшая часть внутреннего адреса, то старшие разряды выполняют функцию дешифратора адресных окон при соответствующей настройке регистров ADDRSELx. Тогда при адресации (выборка инструкций или чтение-запись данных) старшая часть полного внутреннего адреса определяет адресное окно, для которого формируется соответствующий сигнал выборки CSx#.

Например, устанавливая четыре разряда для сегментной части адреса A19 – A16 и используя пять сигналов CSx# для выборки внешних устройств, можно организовать пять банков внешней памяти по 1 Мбайт каждый. Начальные адреса окон необходимо разнести друг от друга не менее чем на 1 Мбайт. Общий размер адресуемой внешней памяти при этом составит 5 Мбайт.

Программируемые временные параметры внешней шины

В микроконтроллере основные временные характеристики (см. рисунок 10.4) могут задаваться программно. Этим достигается возможность работы микроконтроллера с различными типами периферийных устройств и устройств внешней памяти.

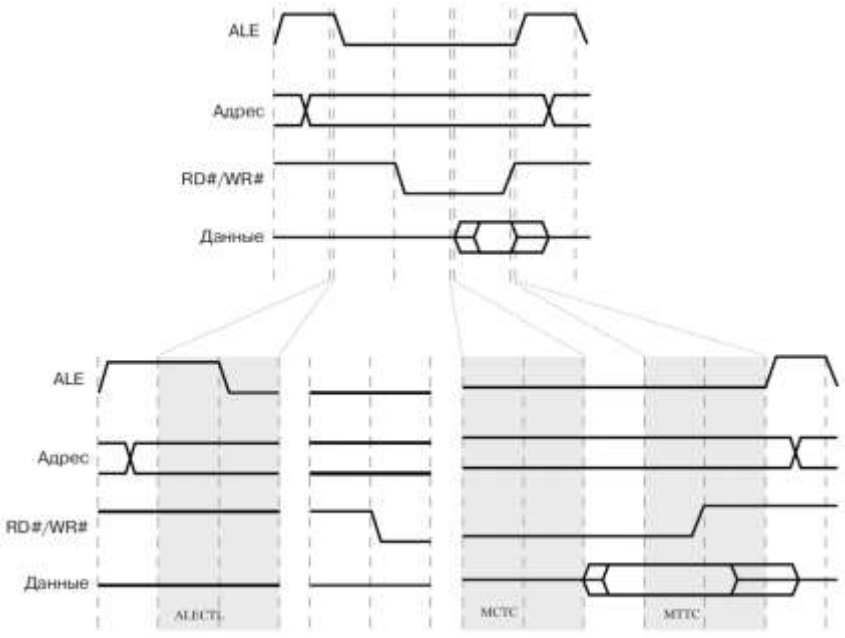


Рисунок 10.4 – Программируемые временные параметры циклов шины

Программируемые параметры цикла внешней шины:

- длительность сигнала ALE и время удержания адреса на шине после отрицательного фронта ALE;
- длительность циклов памяти для задания необходимого времени доступа (задается в TCL от 1 до 15; TCL – время равное половине периода тактового сигнала ЦПУ);
- длительность времени высокоимпедансного состояния на шине (с расширением на один TCL);
- задержка сигналов чтения и записи после отрицательного фронта ALE;
- использование сигнала готовности внешнего устройства READY#.

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 битом ALECTL регистра BUSCONx. Когда бит ALECTL установлен, длительность сигнала ALE увеличивается на половину периода тактового сигнала ЦПУ, т.е. на один TCL. Время удержания адреса для мультиплексной шины при установленном ALECTL увеличивается на один TCL после отрицательного фронта сигнала ALE. Передача данных при этом задерживается на один период системного тактового сигнала CLKOUT (равно двум TCL), чтобы она начиналась по тому же фронту тактового сигнала. После сброса микроконтроллера бит ALECTL0 устанавливается для обеспечения самого медленного цикла шины при обращении к адресному окну 0 (BUSCON0 – CS0#). Регистры ALECTL1 – ALECTL4 после сброса обнуляются.

Сброшенный бита CSCFG конфигурирует контроллер EBC на формирование отрицательных фронтов всех сигналов выборки CSx# – CS4# через один TCL после фронта ALE. 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, см. рисунок 10.5.

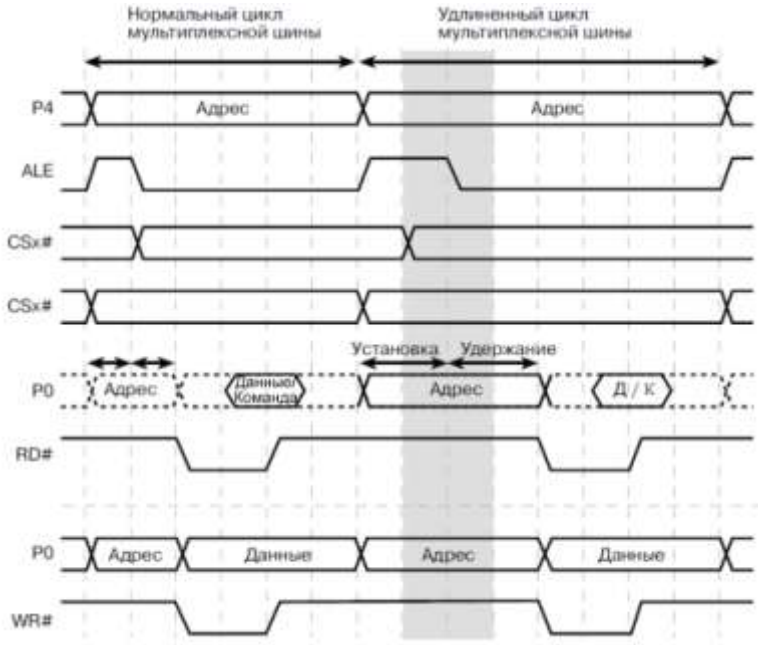


Рисунок 10.5 – Цикл сигнала ALE

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 вычисляется как разность числа 15 и значения MCTC. При MCTC = Fh чтение и запись выполняются без дополнительных задержек. Один такт задержки равен одному периоду тактового сигнала CPU (два TCL). После сброса битовые поля MCTC всех регистров BUSCONx устанавливаются в нулевое значение, что соответствует максимальной задержке.

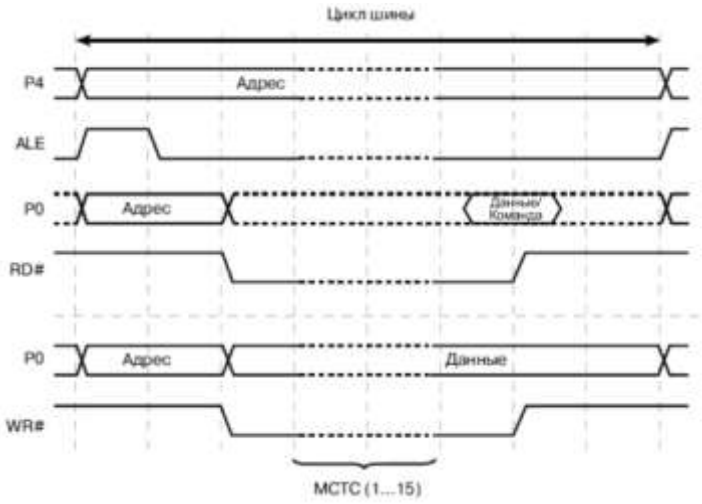


Рисунок 10.6 – Длительность циклов чтения и записи

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 управляется битом MTTC регистра BUSCONx и, по молчанию, устанавливается равным двум TCL, см. рисунок 10.7.

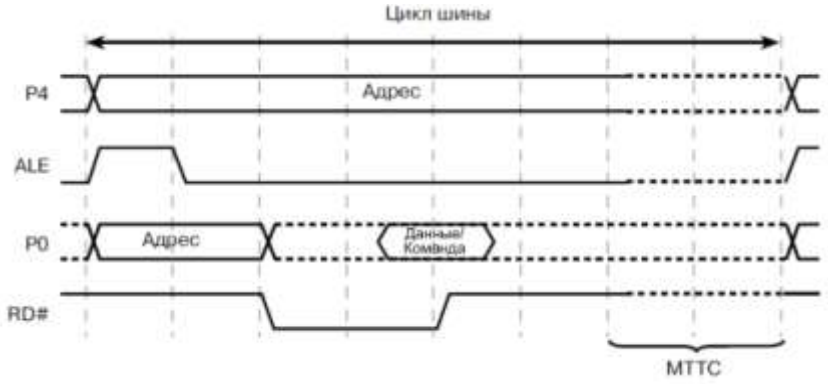


Рисунок 10.7 – Время задержки после чтения

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 MTTC, добавляется еще задержка на один период тактового сигнала ЦПУ.

Задержка сигналов чтения и записи

Задержка сигналов чтения/записи представлена на рисунке 10.8.

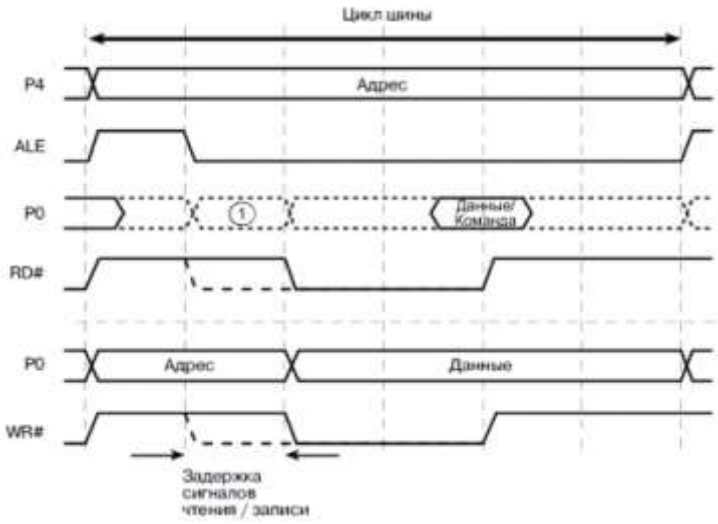


Рисунок 10.8 – Задержка сигналов чтения /записи

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 временные параметры чтения/записи с учетом временных характеристик внешней периферии посредством бита RWDC. 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 (бит RWDC установлен) задний фронт сигналов чтения и записи совпадает с задним фронтом сигнала ALE. При RWDC = 0b устанавливается задержка, равная одному TCL. Данная задержка не увеличивает общее время доступа и не уменьшает быстродействие микроконтроллера. После сброса бит RWDC сброшен. 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 один TCL. Сигнал WR# переключается в активный низкий уровень в стандартном режиме, однако, может быть снят на один TCL раньше, что показано на стандартной временной диаграмме на рисунке 10.9. В этом случае выходной драйвер деактивируется на один TCL 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. Выбором раннего сигнала WR# можно избежать конфликта между микроконтроллером и выходами внешней периферии. Однако надо удостовериться, что ранний сигнал WR# отвечает требованиям внешней периферии.

Деактивация раннего сигнала WR# управляется битом EWEN регистров BUSCONx. Сигнал WR# будет укорочен, если бит EWEN установлен. После сброса микроконтроллера бит EWEN сброшен и сигнал WR# работает в стандартном режиме.

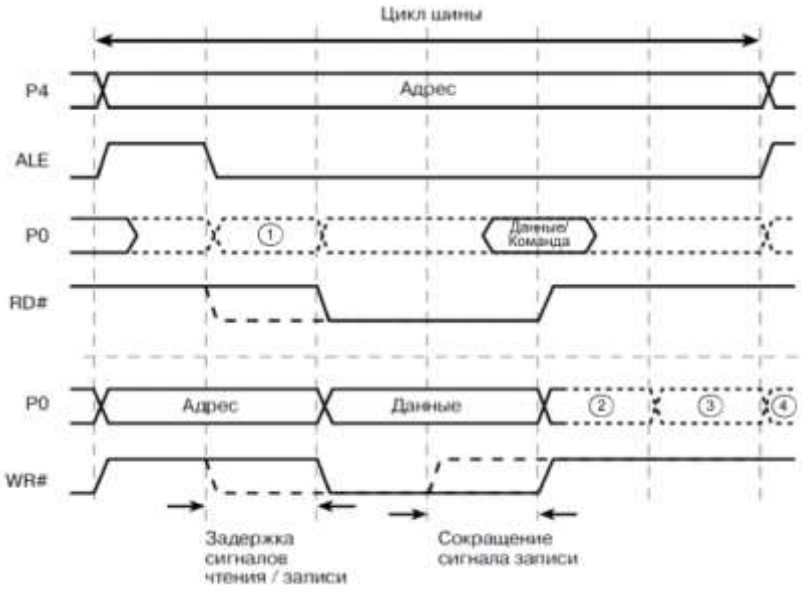


Рисунок 10.9 – Завершение формирования сигнала записи

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 от нуля до семи тактов, а затем отслеживает сигнал READY# для определения необходимости завершения цикла внешней шины. Внешнее устройство должно установить сигнал READY# в состояние логического нуля после того, как данные для чтения выданы на шину или данные для записи были сохранены.

Функция сигнала READY# разрешается битом RDYEN регистра BUSCONx. Когда бит RDYEN установлен, только соответствующие младшие три бита поля 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. Сигнал READY# имеет внутреннюю синхронизацию, и запрограммированные циклы ожидания необходимы для обеспечения параметров шинных циклов. 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#. Обычно внешняя память является более быстрым устройством, чем периферия, поэтому использование сигнала READY# не должно оказывать влияния на скорость выполнения программы.

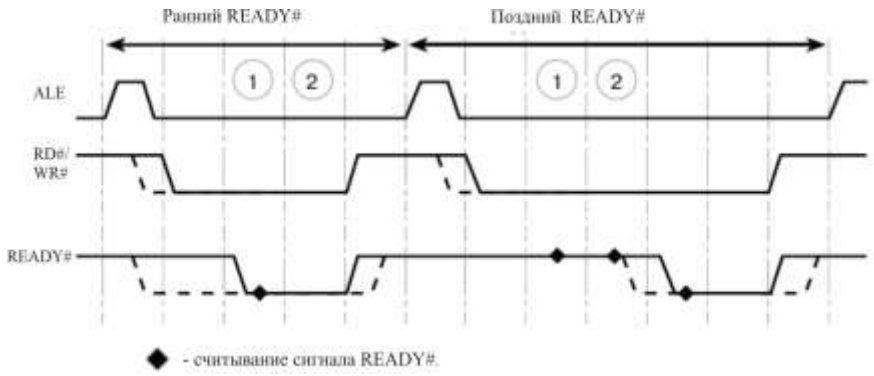


Рисунок 10.10 – Циклы шины с использованием сигнала READY#

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 на рисунке 10.10).

10.6 Настройка контроллера внешней шины

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 регистров BUSCONx идентичны. 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 битов BUSACT, ALECTL и поля BTYP регистра BUSCON0 устанавливаются во время сброса через порт P0. Если во время сброса на входе EA# был уровень логического нуля, то биты BUSACT и ALECTL устанавливаются, а значение BTYP задается состоянием P0L.7 и P0L.6. Остальные битовые поля и биты обнуляются, устанавливая, таким образом, цикл шины с максимальными задержками. Если на входе EA# уровень логической единицы, то регистр BUSCON0 обнуляется. При этом обращение по внешней шине запрещается, т. к. остальные регистры BUSCON1 – BUSCON4 всегда обнуляются после системного сброса. Значения регистров BUSCON0 – BUSCON4 и ADDRSEL1 – ADDRSEL4 могут быть изменены программно.

Следует помнить, что сначала устанавливается значение битов регистра ADDRSELx, а затем – соответствующих битов регистра BUSCONx. Соблюдение этой последовательности обязательно.

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 и приема параллельных данных и одиночных внешних сигналов управления в микроконтроллере имеется 103 линии параллельного ввода и вывода, сгруппированные в восемь портов ввода-вывода и один входной порт: 16-разрядные P0, P1, P3, 8-разрядные P2, P4, P6, P5, 7-разрядный P7 и 16-разрядный P9 (входной).

Линии портов могут использоваться для команд ввода-вывода основного назначения под программным управлением, для интегрированной периферии микроконтроллера и 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 высокоимпедансное состояние при работе на вход. Выводы портов P2, P3, P4, P6, P7 и P9 могут быть сконфигурированы для работы в режиме двухтактных выходов («push/pull») или для работы в режиме с открытым стоком. 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 выполнить запись в порт, настроенный на вход, то 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 микроконтроллера, модифицирует его и записывает назад в выходной триггер.

Если вывод сконфигурирован как выход, то выходной триггер и вывод микроконтроллера соединены (выходной буфер включен). Чтение этого вывода возвращает значения выходного триггера. Действие чтение-изменение-запись читает значение выходного триггера, изменяет его и записывает назад в выходной триггер, изменяя уровень сигнала на выводе микроконтроллера.

Направление работы порта задается соответствующим регистром DPx (x – номер порта).

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 выделены независимые выводы.

Режим с открытым стоком

В режиме с открытым стоком выходной драйвер может устанавливать на выходе только низкий уровень сигнала, см. рисунок 11.1. Эта функция управляется регистром ODPx. При записи единицы в триггер порта n-транзистор закрывается, и выход переходит в состояние высокого импеданса. В этом случае высокий уровень сигнала на линии обеспечивается внешним устройством.

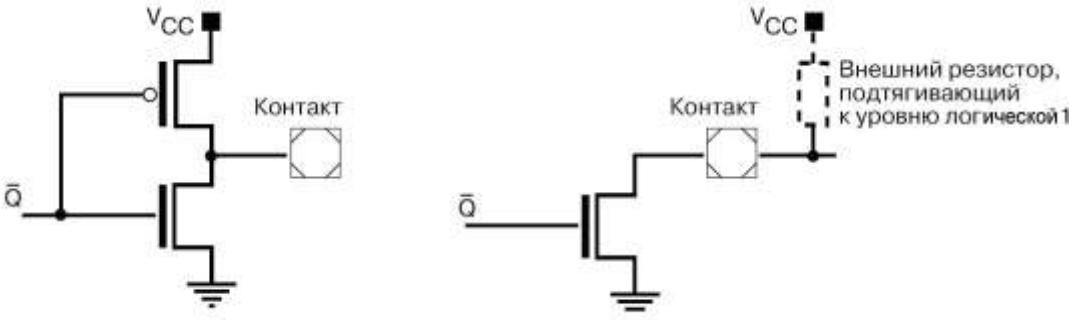


Рисунок 11.1 – Выходные буферы в двухтактном режиме и в режиме с открытым стоком (сигнал $\overline{Q}$ – сигнал, приходящий с выхода триггера порта)

Альтернативные функции выводов портов

При использовании альтернативной функции вывода, он должен быть соответствующим образом сконфигурирован как вход или выход (за исключением автоматически настраиваемых после сброса выводов) посредством регистра DPx. Схемы, реализующие основную и альтернативные функции порта объединены логически по И, поэтому в триггер порта необходимо записывать единицу.

После сброса вывод автоматически настроен как вход и находится в высокоимпедансном состоянии. Если никаких внешних устройств не подключено к выводу порта, то направление работы вывода 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.

Конфигурирование выводов осуществляет программа пользователя. Для некоторых выводов портов направление устанавливается автоматически. Например, в режиме мультиплексной внешней шины порта P0, направление работы выводов переключается несколько раз за один цикл чтения шины, что невозможно сделать посредством команд.

Управление альтернативными функциями порта осуществляется посредством пары регистров ALTSEL0Px и ALTSEL1Px.

Примечание – Порт P0 не имеет регистров альтернативных функций. Порты P1, P2, P4, P6 имеют по одному регистру 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 формирования нежелательных уровней сигналов на выводах.

11.1 Порт P0

16-разрядный порт P0 сформирован из двух 8-разрядных портов P0H и P0L, каждый из которых доступен для изменения независимо от другого. В режиме ввода-вывода основного назначения направление выводов конфигурируется помощью регистров DP0H и DP0L. На рисунке 11.2 показана структура вывода порта P0.

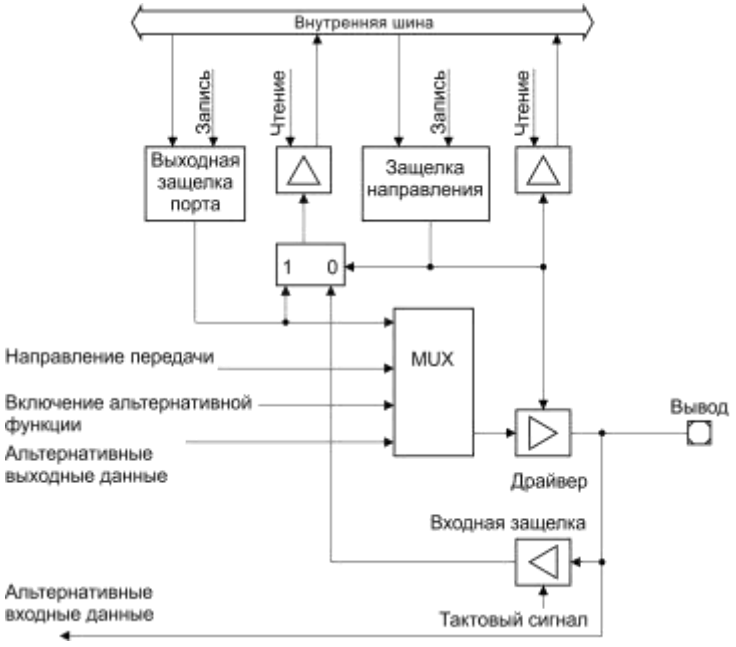


Рисунок 11.2 – Структурная схема вывода порта P0

Альтернативные функции порта P0

При включенной внешней шине порт P0 используется в качестве шины данных или шины адреса и данных. Внешняя 8-разрядная демультиплексная шина использует 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 слова, затем – к старшему.

Порт P0 также используется для выбора конфигурации системы при старте, см. приложение Ж. Во время сброса порт P0 сконфигурирован на вход, и за счет внутренних схем подтяжки (Pull-up) на каждой линии выставлен высокий уровень. Каждая линия может быть индивидуально подключена к нулевому потенциалу, с помощью внешней подтяжки к нулю (Pull-down), что позволяет пользователю изменять состояния выводов.

Альтернативные функции выводов порта P0 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
P0L.0	D0	D0	AD0	AD0
P0L.1	D1	D1	AD1	AD1
P0L.2	D2	D2	AD2	AD2
P0L.3	D3	D3	AD3	AD3
P0L.4	D4	D4	AD4	AD4
P0L.5	D5	D5	AD5	AD5
P0L.6	D6	D6	AD6	AD6
P0L.7	D7	D7	AD7	AD7
P0H.0	Отсутствует	D8	A8	AD8
P0H.1		D9	A9	AD9
P0H.2		D10	A10	AD10
P0H.3		D11	A11	AD11
P0H.4		D12	A12	AD12
P0H.5		D13	A13	AD13
P0H.6		D14	A14	AD14
P0H.7		D15	A15	AD15

Внутренняя схема подтяжки Pull-up подобна внешним Pull-down-резисторам, которые могут использоваться для подачи корректного низкого уровня напряжения. Внешние Pull-down -резисторы могут оставаться подсоединенными к выводам порта P0 во время нормальной работы микроконтроллера, однако, необходимо обращать внимание на то, чтобы их значения не мешали нормальному функционированию порта.

После окончания сброса, конфигурация сигналов с выводов порта P0 записывается в регистр BUSCON.

Когда включен режим внешней шины, направление вывода порта и загрузка данных в выходной триггер порта управляются с помощью 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.

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 высокоимпедансном состоянии. 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

16-разрядный порт P1 сформирован из двух 8-разрядных портов P1H и P1L, каждый из которых доступен для изменения независимо от другого. В режиме ввода-вывода основного назначения направление выводов конфигурируется с помощью регистров DP1H и DP1L. Структурная схема вывода порта P1 представлена на рисунке 11.3.

Для управления альтернативными функциями используются регистры ALTSEL0P1H и ALTSEL0P1L. 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 выводы порта P1 не требуются и могут использоваться как выводы общего назначения.

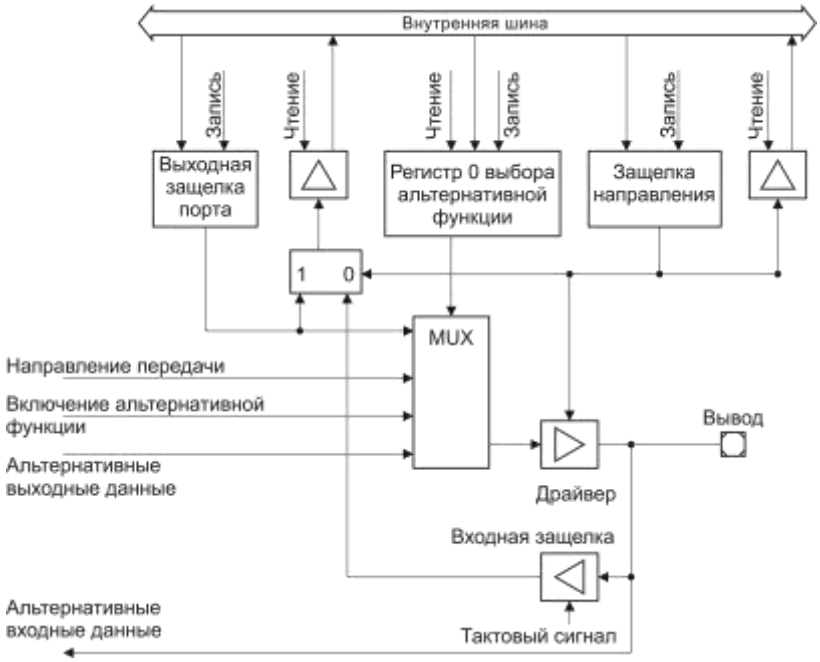


Рисунок 11.3 – Структурная схема вывода порта P1

11.3 Порт P2

8-разрядный порт. Направление выводов конфигурируется с помощью регистра DP2. Каждая линия порта P2 может функционировать в режиме двухтактного выхода или с открытым стоком, что задается регистром ODP2. Структурная схема вывода порта P2 представлена на рисунке 11.4

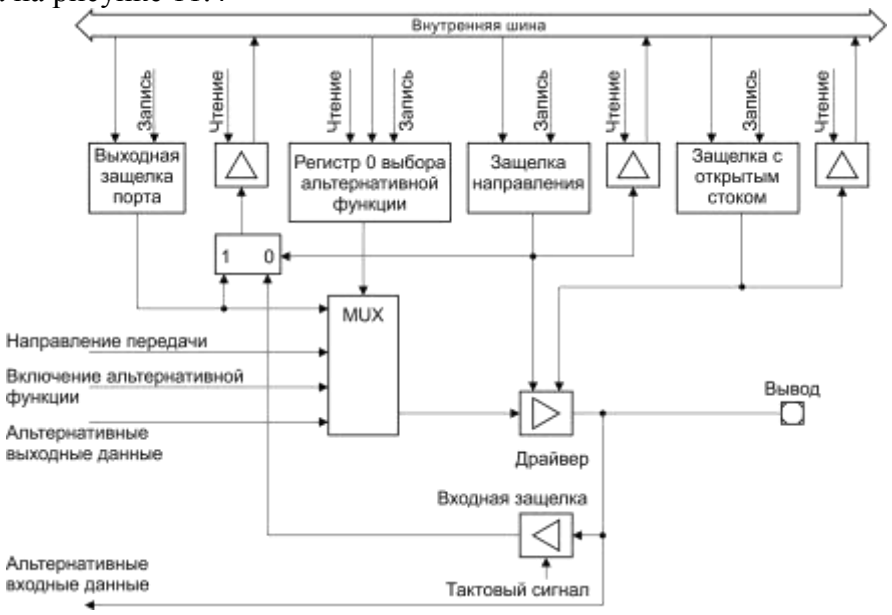


Рисунок 11.4 – Структурная схема вывода порта P2

Альтернативные функции порта P2 управляются регистром ALTSEL0P2. 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 активных устройств, во избежание конфликта сигналов.

Дополнительно все выводы порта являются входами внешних прерываний.

11.4 Порт P3

16-разрядный порт. Направление выводов конфигурируется с помощью регистра DP3. Часть линий порта P3 может функционировать в режиме двухтактного выхода или с открытым стоком, что задается регистром ODP3. Выводы P3.12, P3.14 и P3.15 не поддерживают режим с открытым стоком. Структурная схема вывода порта показана на рисунке 11.5.

Альтернативные функции порта управляются регистрами ALTSEL0P3 и ALTSEL1P3. Выводы порта P3 выполняют различные функций, в том числе передачу сигналов BHE#/WRH# и CLKOUT/FOUT. 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.

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

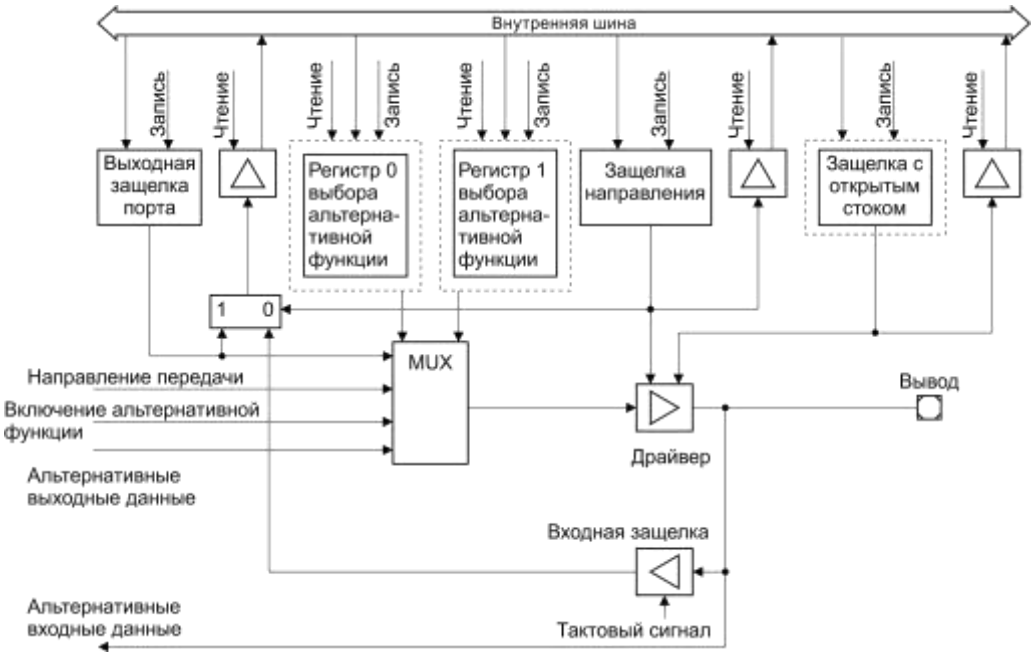


Рисунок 11.5 – Структурная схема вывода порта P3

11.5 Порт P4

8-разрядный порт. Направление выводов конфигурируется с помощью регистра DP4. Если порт используется как внешняя шина, то его работой управляет контроллер EBC. Структурная схема вывода порта P4 представлена на рисунке 11.6

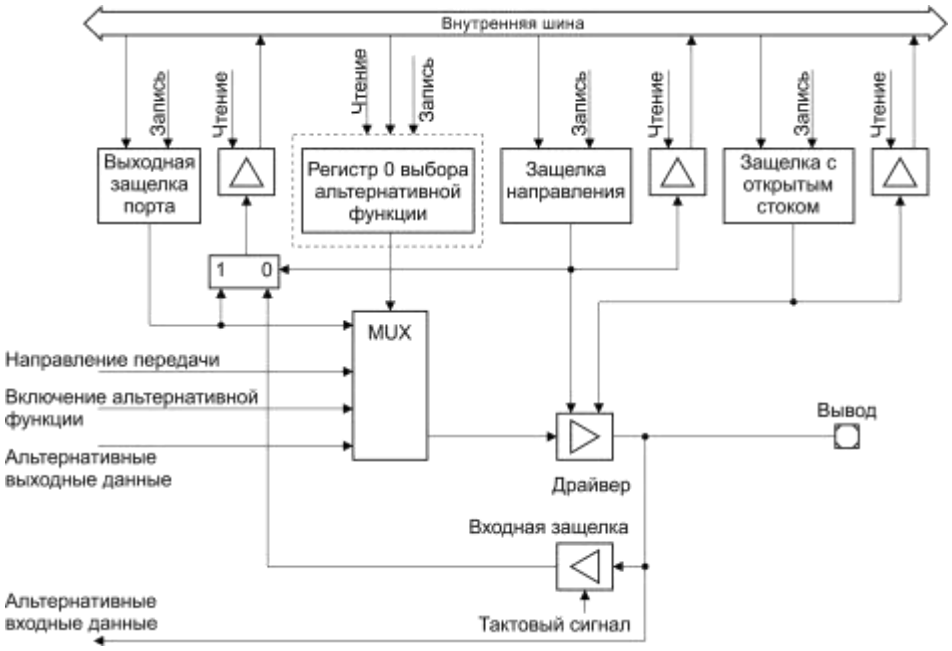


Рисунок 11.6 – Структурная схема вывода порта P4

Альтернативные функции порта управляются регистром ALTSEL0P4. 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.

11.6 Порт P5

16-разрядный входной порт общего назначения. Регистр порта предназначен только для чтения. Структурная схема вывода порта показана на рисунке 11.7.

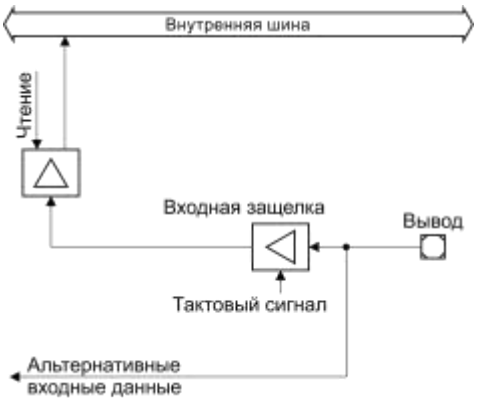


Рисунок 11.7 – Структурная схема вывода порта P5

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.

11.7 Порт P6

8-разрядный порт. Направление выводов конфигурируется с помощью регистра DP6. Каждая линия порта P6 может функционировать в режиме двухтактного выхода или с открытым стоком, что задается регистром ODP6. Структурная схема вывода порта P6 представлена на рисунке 11.8

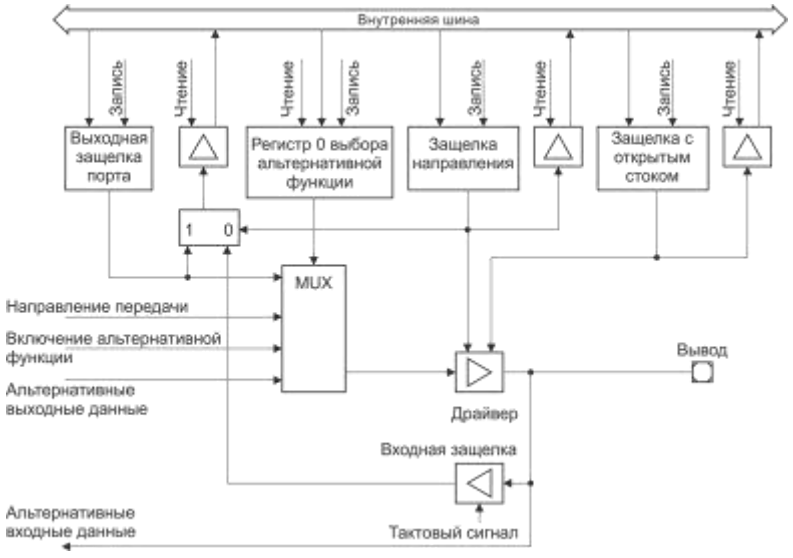


Рисунок 11.8 – Структурная схема вывода порта P6

Альтернативные функции порта управляются регистром ALTSEL0P6 и приведены в таблице 11.8. 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 Pull-up, 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.

11.8 Порт P7

7-разрядный порт. Направление выводов конфигурируется с помощью регистра DP7. Каждая линия порта P7 может функционировать в режиме двухтактного выхода или с открытым стоком, что задается регистром ODP7. Структурная схема вывода порта P7 представлена на рисунке 11.9

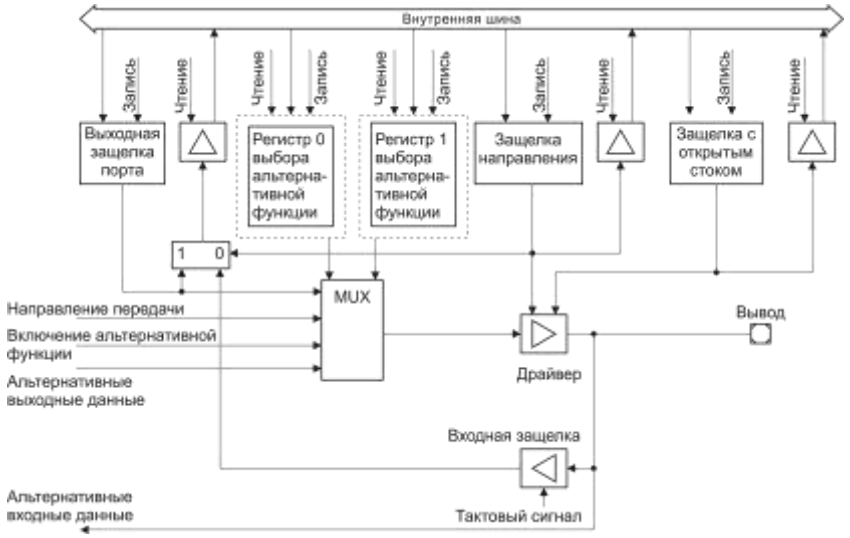


Рисунок 11.9 – Структурная схема вывода порта P7

Альтернативные функции порта управляются регистрами ALTSEL0P7 и ALTSEL1P7.

11.9 Порт P9

8-разрядный порт. Направление выводов конфигурируется с помощью регистра DP8. Каждая линия порта P8 может функционировать в режиме двухтактного выхода или с открытым стоком, что задается регистром ODP8. Структурная схема вывода порта P8 представлена на рисунке 11.10

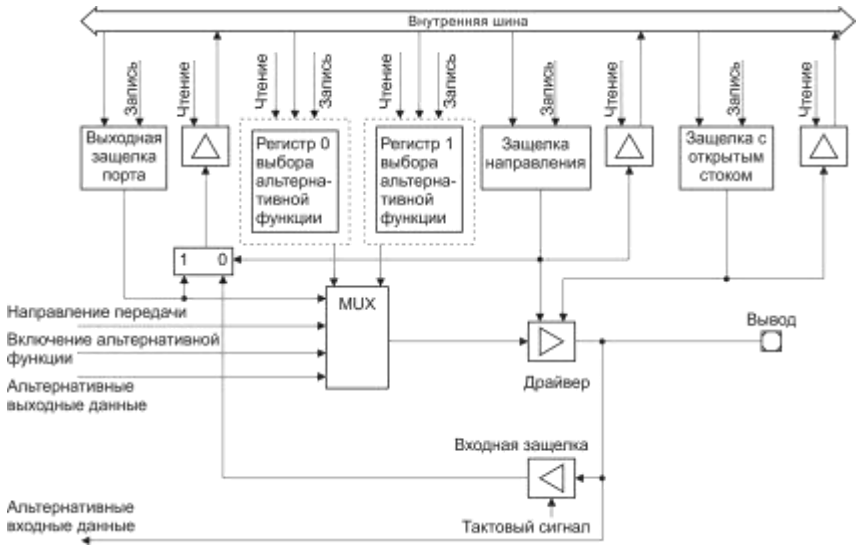


Рисунок 11.10 – Структурная схема вывода порта P9

Альтернативные функции порта управляются регистрами ALTSEL0P9 и ALTSEL1P9

11.10 Специальные выводы

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.

Выводы осциллятора XTAL1 и XTAL2 предназначены для подключения кварцевого резонатора или внешнего генератора к схеме внутреннего тактового генератора. Стандартная схема подключения кварца включает кварцевый резонатор, два конденсатора и резистор, ограничивающий ток через кварцевый резонатор. Основной осциллятор предназначен для генерации основного тактового сигнала контроллера. При работе от внешнего тактового сигнала, например, от внешнего генератора, он подается на вход XTAL1.

Вход осциллятора XTAL3 и выход XTAL4 связывают вспомогательный осциллятор с внешним кварцем. Этот дополнительный осциллятор используется для формирования медленного тактового сигнала, который поступает на тактовый вход модуля часов реального времени RTC, и работает независимо от генератора системного тактового сигнала. При работе от внешнего генератора медленного тактового сигнала, сигнал подают на вход XTAL3.

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 ∩VCC3 и ∩0V3 используются для отдельного питания АЦП, что позволяет уменьшить шум от цифровой части микроконтроллера, передаваемый по линиям питания.

Выводы питания #VCC1, #0V1, #VCC2 и #0V2 предназначены для подключения питания цифровой логики микроконтроллера. Разделительные конденсаторы рекомендуется подключать как можно ближе к выводам питания. Выводы #VCC2 и #0V2 запитывают ядро схемы, в то время как выводы #VCC1, #0V1 обеспечивают питание периферии.

Сигнал на выводе ALE осуществляет управления защелками адреса, обеспечивая стабильный адрес в режиме мультиплексной шины. ALE выдается в каждом цикле внешней шины независимо от выбранного режима, т. е. формируется даже для режима демультиплексной шины. Сигнал ALE не активируется во время внутреннего доступа (при обращении к внутреннему ОЗУ, внутреннему ПЗУ и внутренним регистрам SFR). Во время сигнала сброса внутренняя схема обеспечивает низкий уровень сигнала ALE. Уровень сигнала на выводе ALE во время сброса выбирает режим работы: если ALE = 0 и RD# = 0 – контроллер работает от осциллятора, если ALE = 1 – контроллер запускается в режиме работы от внутреннего генератора PLL.

Сигнал на выводе RD# предназначен для управления выходными буферами внешней памяти или периферии при работе с внешними устройствами. Во время доступа к X-периферии остается неактивным (высокий уровень). Во время сброса внутренняя схема обеспечивает высокий уровень сигнала на выводе RD#. В конце сброса уровень на выводе RD# защелкивается и используется для конфигурации. Уровень сигнала на выводе RD# во время сброса выбирает режим работы: если RD# = 0 и ALE = 0 – контроллер работает от осциллятора, если RD# = 1 – контроллер запускается в режиме работы от генератора PLL.

Сигнал на выводе WR#/WRL# управляет передачей данных во время работы с внешней памятью или периферийными устройствами. Вывод может формировать либо сигнал WR#, общий для записи слов и байтов, либо сигнал WRL# – для записи только младшего байта слова, во время доступа к X-периферии – остается неактивным (высокий уровень). Во время сброса внутренняя схема обеспечивает высокий уровень на выводе WR#/WRL#.

Сигнал на выводе 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 выводе EA# определяет область памяти, из которой начнется выполнение команды. Если во время сигнала сброса EA# = 0, то выборка программы начнется из внешней памяти, если EA# = 1, то контроллер будет сконфигурирован в режиме работы с внутренним ПЗУ.

Вывод 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 1мкс.

Вывод 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

Модуль часов реального времени RTC представляет 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.

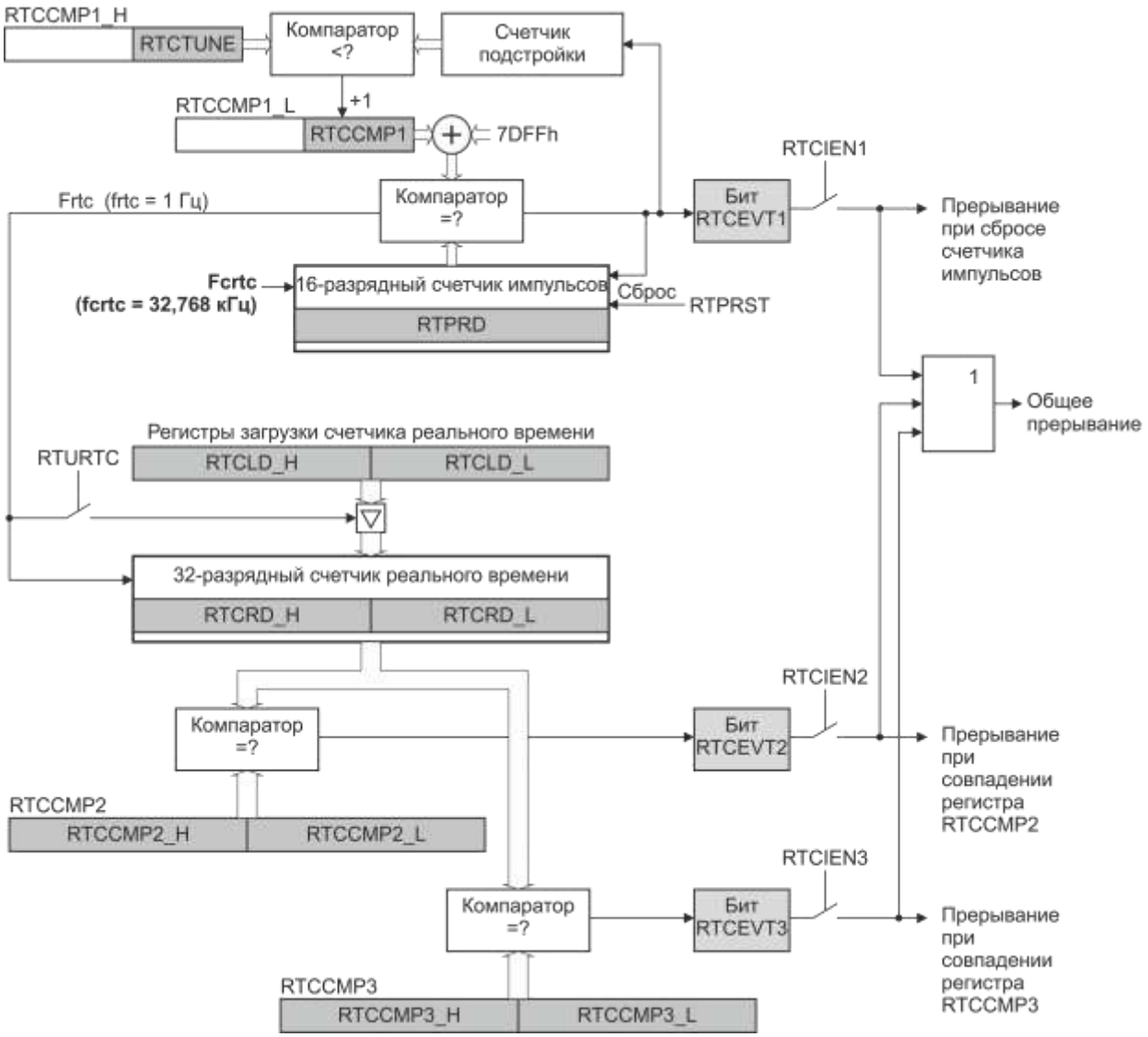


Рисунок 12.1 – Схема модуля RTC

Функционирование

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.

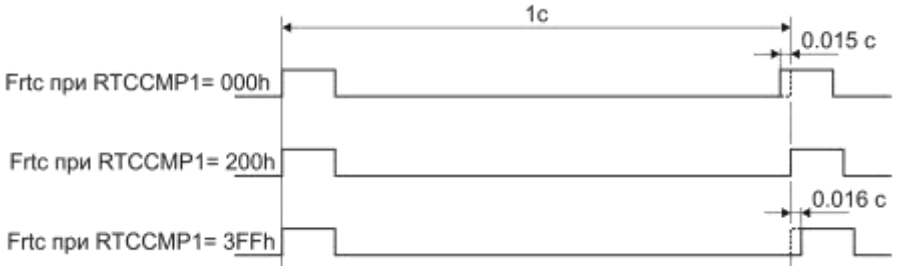


Рисунок 12.2 – Изменение периода сигнала Frtc в зависимости от состояния поля RTCCMP1

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.

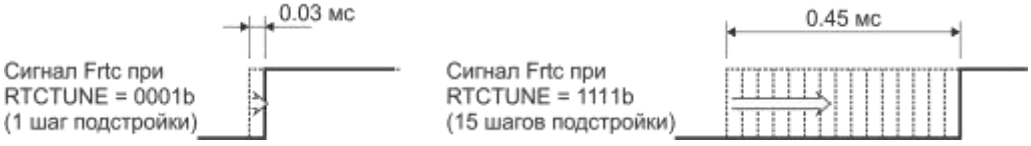


Рисунок 12.3 – Увеличение периода сигнала Frtc

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 представлено, каким образом могут быть проинициализированы регистры и их начальные состояния после инициализации.

Таблица 12.1 – Инициализация регистров

Регистры, которые сбрасываются при системном сбросе		Регистры, которые сбрасываются только при выполнении Power-On-Reset	
Мнемоника	Значение инициализации	Мнемоника	Значение инициализации
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

Блоки таймеров общего назначения используются для измерения времени, подсчета количества событий, измерения длительности импульсов, генерации импульсов, умножения частоты и др.

Основной таймер T3 и вспомогательные T2, T4 входят в состав блока GPT1. Вспомогательные таймеры блока GPT1 могут быть настроены как регистры перезагрузки или захвата для основного таймера. Основной таймер T6 с вспомогательным T5, а также регистр CAPREL (с поддержкой операций захвата и перезагрузки) входят в состав блока GPT2. Любой таймер может работать независимо и может быть объединен с другим таймером. Максимальные частоты переключения счетчиков таймеров блоков GPT1 и GPT2 составляют fosc/4 и fosc/2, соответственно.

Все таймеры являются двунаправленными счетчиками (с возможностью переключения направления счета как программно, так и аппаратно внешним сигналом с входа TxEUD без приостановки) и управляются регистрами TxCON (x – номер таймера). Каждый таймер имеет входную линию синхронизации/управления TxIN (альтернативная функция вывода порта). Текущее состояние счетчика каждого таймера может быть прочитано/изменено посредством регистров T2 – T6 (программная запись в регистр имеет приоритет над аппаратным изменением его содержимого).

Таймеры блока GPT1 поддерживают четыре режима работы: таймера, таймера с внешним управлением, счетчика и счетчика с внешним управлением по двум входам. Сигнал переполнения/опустошения таймера T3 может быть передан на вывод микроконтроллера в качестве альтернативной функции T3OUT. Таймеры T2 и T4 могут быть связаны с таймером T3 через линию T3OTL.

Сигнал переполнения/опустошения таймера T6 может быть передан на выводы микроконтроллера T6OUT и T6OFL. Таймер T6 может быть перезагружен содержимым регистра CAPREL. Имеется возможность объединения таймера T6 с таймером T5, а также с другими таймерами через линию T6OUT. Значение счетчика таймера T5 может захватываться в регистр CAPREL.

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

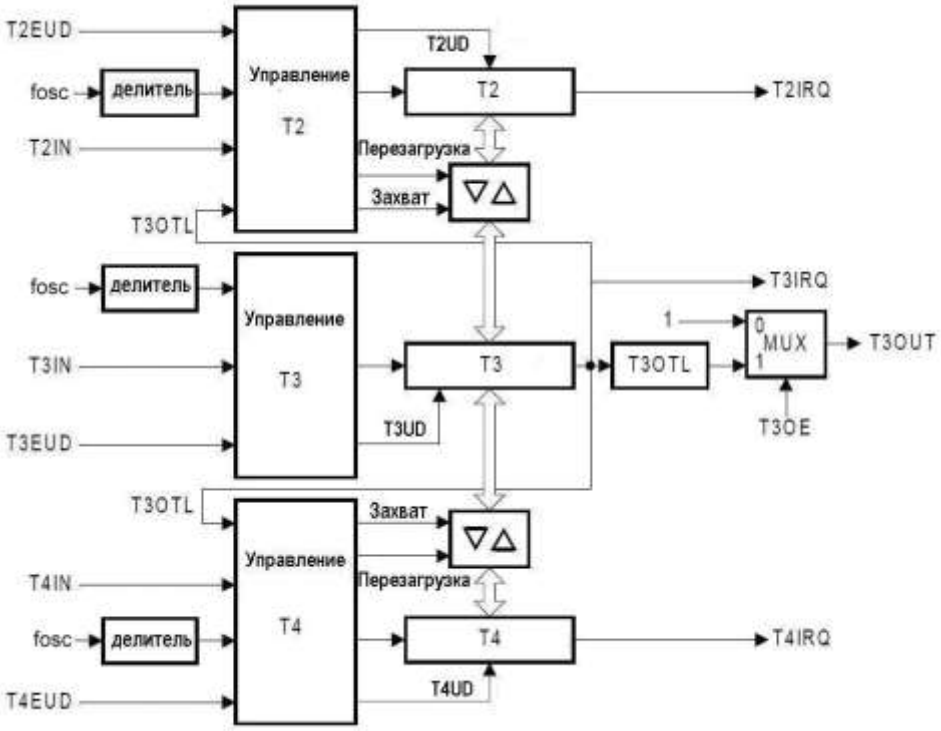


Рисунок 13.1 – Структурная схема блока GPT1

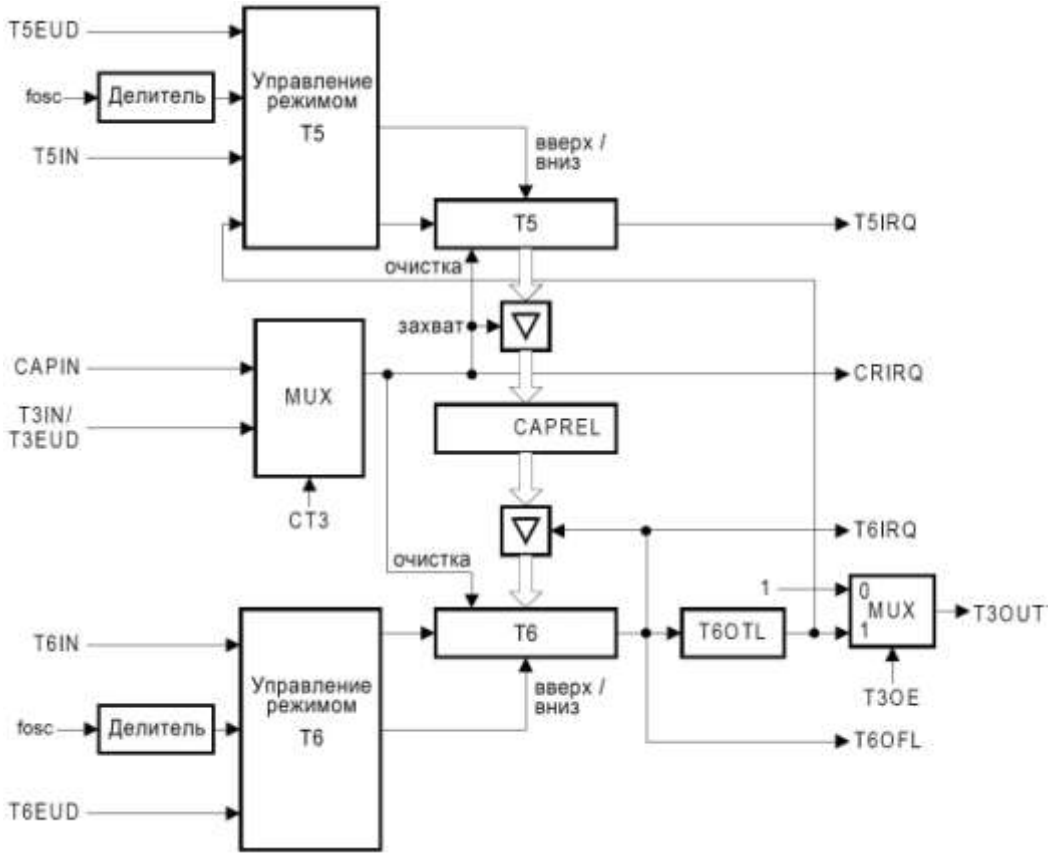


Рисунок 13.2 – Структурная схема блока GPT2

13.1 Таймер T3

Основной таймер блока GPT1. Настраивается и управляется посредством побитно адресуемого регистра T3CON. Регистр T3 доступен только для записи.

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 – Если установлен бит T2RC/T4RC в регистре T2CON/T4CON, бит T3R будет также управлять вспомогательным таймером T2/T4.

Направление счета таймера может контролироваться программно битом T3UD или сигналом на входе P3.4 микроконтроллера (альтернативная функция T3EUD), в зависимости от состояния бита T3UDE. Направление счета можно изменять вне зависимости от того, работает таймер или нет.

Сигнал переполнения/опустошения с выхода таймера T3 подключен к схеме выходного триггера и влияет на состояние бита T3OTL, который также доступен программно. Бит T3OE разрешает передачу сигнала таймера на вывод T3OUT (например, для управления внешними устройствами).

Переключение бита T3OTL может использоваться в качестве синхросигнала или сигнала перезагрузки таймеров T2 и T4. В этом случае передача сигнала на вывод T3OUT не возможна.

Таймер T3 в режиме таймера

Режим выбирается записью значения 000b в поле T3M. Если установлен бит T3R, счетчик таймера инкрементируется/декрементируется с каждым тактом синхросигнала с частотой ft3 . Функциональная схема показана на рисунке 13.3.

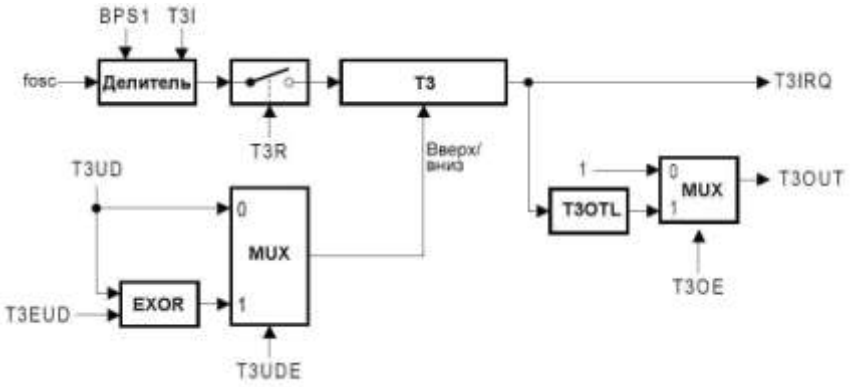


Рисунок 13.3 – Функциональная схема работы Т3 в режиме таймера

Синхросигнал с частотой fosc подается на блок программируемого делителя. В зависимости от значения поля BPS1 частота входного сигнала делится на 4, 8, 16 или 32 и далее на число 2^{T3I}. Полученное значение является значением частоты ft3 переключения таймера Т3. Например, если задано BPS1 = 01b и T3I = 011b, то значение делителя будет равно 32:

$$ft3 = \frac{fosc}{4} \times \frac{1}{2^3} = \frac{fosc}{32}. \quad (13.1)$$

Таймер Т3 в режиме внешнего управления таймером

Режим выбирается записью значения 010b/011b в поле T3M. Функциональная схема показана на рисунке 13.4.

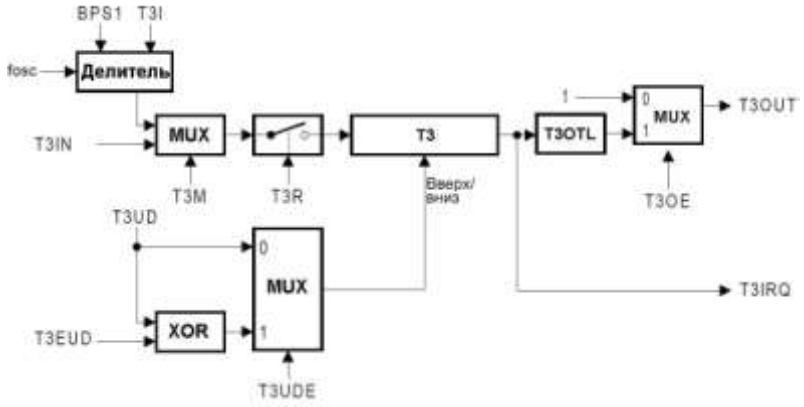


Рисунок 13.4 – Функциональная схема работы таймера Т3 в режиме внешнего управления таймером

Формирование синхросигнала и работа таймера Т3 идентична режиму таймера, с тем отличием, что работа счетчика дополнительно контролируется внешним сигналом на выводе P3.6 (если выбрана альтернативная функция T3IN). Если установлен бит T3R, то счетчик инкрементируется/декрементируется только в том случае, если на входе T3IN установлен выбранный активный уровень сигнала. Так, например, при T3M = 010b (и T3R = 1b) счетчик запущен, если на входе T3IN удерживается низкий уровень сигнала, в противном случае счетчик остановлен.

Примечание – Сигнал внешнего управления на входе T3IN не генерирует запрос прерывания.

Таймер Т3 в режиме счетчика

Режим выбирается записью значения 001b в поле T3M.

Функциональная схема работы T3 в режиме счетчика показана на рисунке 13.5.

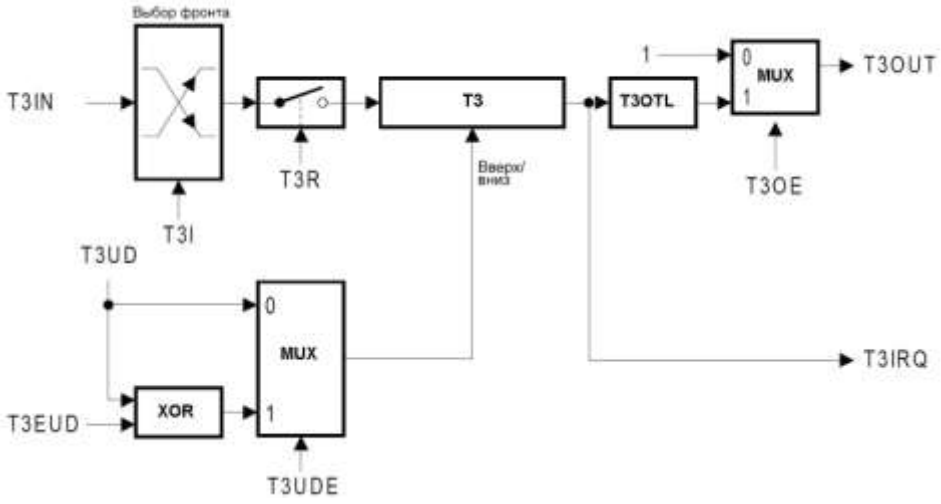


Рисунок 13.5 – Функциональная схема работы таймера T3 в режиме счетчика

Если установлен бит T3R, счетчик таймера инкрементируется/декрементируется по фронту сигнала на выводе P3.6 (альтернативная функция T3IN должна быть выбрана). Активный фронт внешнего сигнала на входе T3IN задается полем T3I.

Максимальная допустимая частота в режиме счетчика составляет fosc/8 (BPS1 = 01b). Время удержания уровня сигнала от одного фронта до другого должно составлять минимум четыре такта fosc.

Таймер T3 в режиме внешнего инкрементирования

Режим выбирается записью значения 110b/111b в поле T3M. Функциональная схема показана на рисунке 13.6.

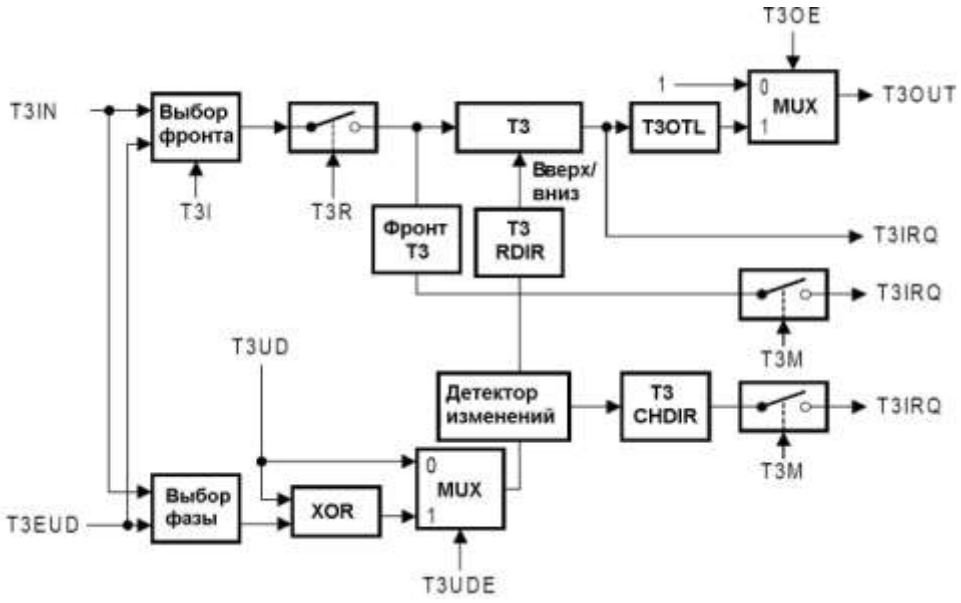


Рисунок 13.6 – Функциональная схема работы таймера T3 в режиме внешнего инкрементирования

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 тактовые импульсы для таймера, и выбирается направление счета. В зависимости от выбранного режима (определения периода или определения фронта), формируется запрос на прерывание T3IRQ. В режиме определения периода прерывание генерируется каждый раз при изменении направления счета таймера T3. В режиме определения фронта прерывание генерируется каждый раз при счете таймера T3. Направление счета, изменение направления счета и запросы на счет контролируются с помощью битов T3RDIR, T3CHDIR, T3EDGE.

Кодер для инкрементирования может быть непосредственно подключен к микроконтроллеру без логики внешнего интерфейса. В стандартной системе используются компараторы, чтобы преобразовать дифференциальные выходы кодера такие как A, A# к цифровым сигналам, таким как «A», см. рисунок 13.7.

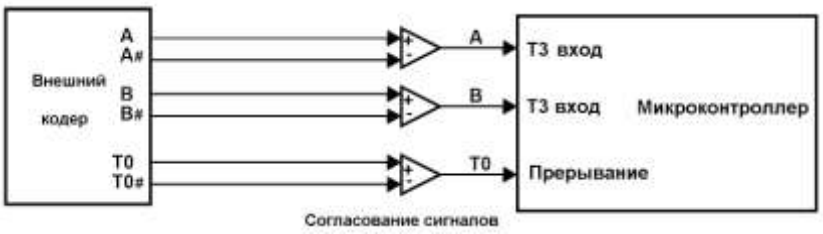


Рисунок 13.7 – Интерфейс кодера и микроконтроллера

Примечание – Третий выход T0 кодера, который указывает начальное положение, может быть подключен к внешнему прерыванию, чтобы вызвать сброс таймера T3.

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 – fosc/8 (BPS1 = 01b). Время удержания уровня сигнала от одного фронта до другого должно составлять минимум четыре такта fosc.

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 T3 – режим внешнего инкрементирования, направления счета

Уровень другого входа	Вход T3IN		Вход T3EUD	
	Положительный фронт	Отрицательный фронт (спад)	Положительный фронт	Отрицательный фронт (спад)
Высокий уровень	Вниз	Вверх	Вверх	Вниз
Низкий уровень	Вверх	Вниз	Вниз	Вверх

На рисунках 13.8 и 13.9 показаны примеры работы таймера T3, 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.

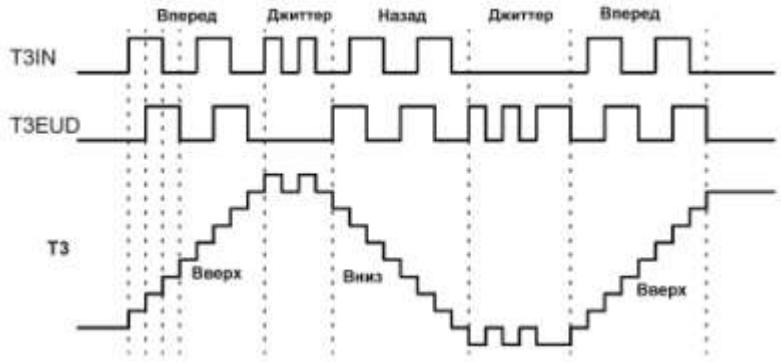


Рисунок 13.8 – Пример работы таймера T3 в режиме внешнего инкрементирования (T3I = 011b)

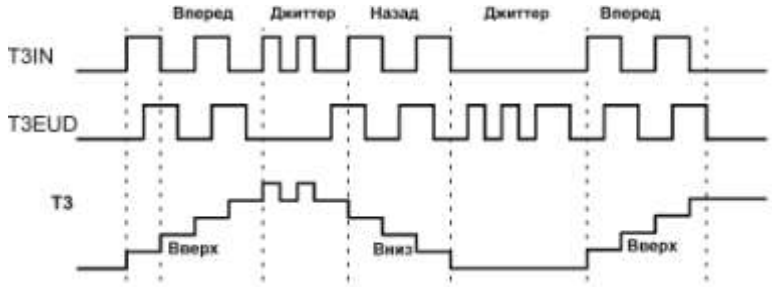


Рисунок 13.9 – Пример работы таймера T3 в режиме внешнего инкрементирования (T3I = 001b)

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 режимах таймера и внешнего управления таймером

Функционирование аналогично работе основного таймера T3 за исключениями:

- для таймеров T2 и T4 отсутствуют выходные сигналы T2OUT и T4OUT;
- отсутствует контроль переполнения/опустошения.

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.

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

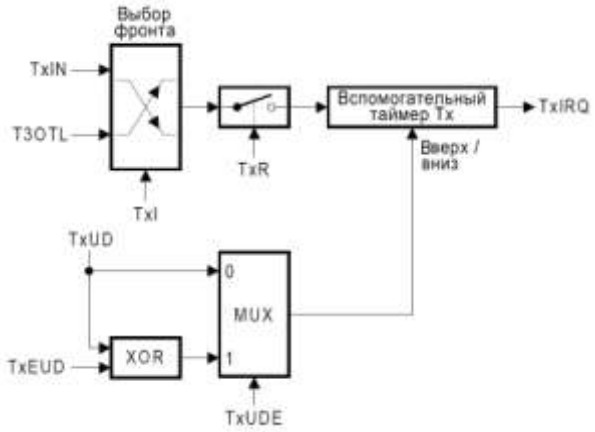


Рисунок 13.10 – Схема вспомогательного таймера в режиме счетчика

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 от одного фронта до другого должно составлять минимум четыре такта fosc.

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.

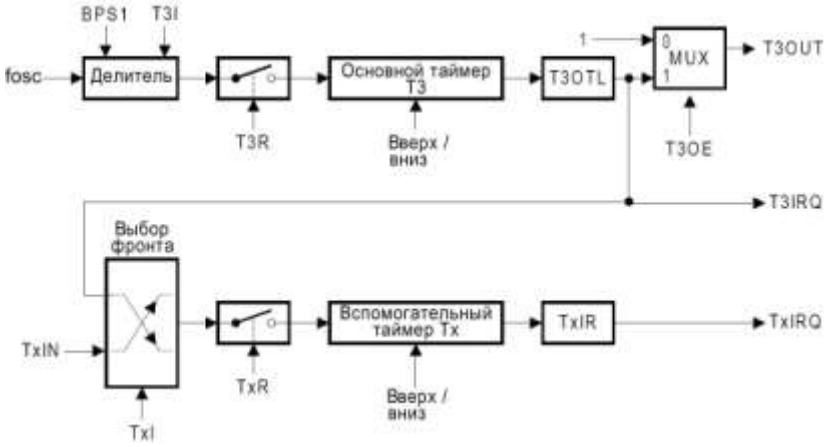


Рисунок 13.11 – Объединение основного таймера T3 и вспомогательного таймера Tx

Если для тактирования вспомогательного таймера выбраны оба фронта входного сигнала (линия T3OTL), тогда этот таймер будет переключаться при каждом переполнении/опустошении основного таймера T3. Таким образом, будет сформирован 32-разрядный таймер.

Если для тактирования вспомогательного таймера выбран один из фронтов (положительный или отрицательный) входного сигнала (линия T3OTL), тогда этот таймер будет переключаться при каждом втором переполнении/опустошении основного таймера T3. И, таким образом, будет сформирован 33-разрядный таймер (16-разрядный основной таймер + T3OTL + 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 T3IRQ оказывает влияние только переполнение/опустошение таймера Т3. Программное изменение состояния бита T3OTL не влияет на состояние линии T3IRQ.

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 так же, как сигнал тактирования в режиме счетчика. В режиме перезагрузки вспомогательный таймер Т2/Т4 может быть остановлен программно, вне зависимости от состояния его флага T2R/T4R.

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

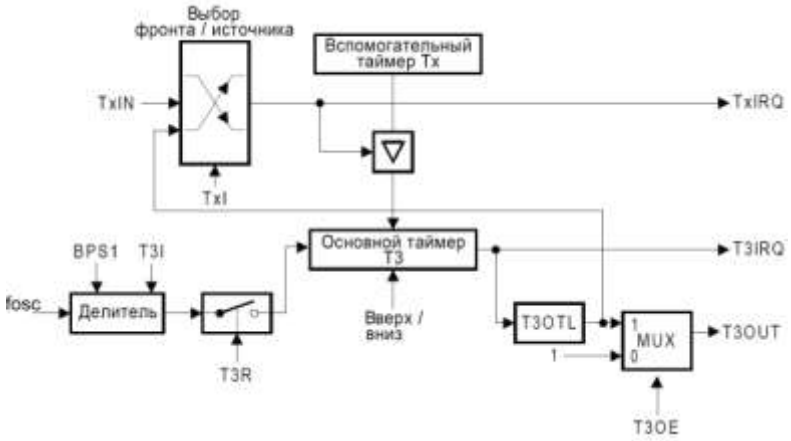


Рисунок 13.12 – Вспомогательный таймер в режиме перезагрузки

Примечание – На изменение уровня сигнала на линии T3IRQ оказывает влияние только переполнение/опустошение таймера Т3. Программное изменение состояния бита T3OTL не влияет на состояние линии T3IRQ.

С появлением сигнала тактирования в таймер Т3 загружается содержимое регистра таймера Т2/Т4 и T2IRQ/T4IRQ переходит в состояние единицы.

Режим перезагрузки с тактированием по входу T3OTL может использоваться в различных конфигурациях. В зависимости от выбранного типа активного фронта тактового сигнала реализуются три режима работы.

1 Стандартный режим перезагрузки. Используются оба фронта сигнала тактирования на входе T3OTL. При переполнении/опустошении основного таймера, в него загружается содержимое вспомогательного таймера.

2 Если используется один из фронтов сигнала тактирования на 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.

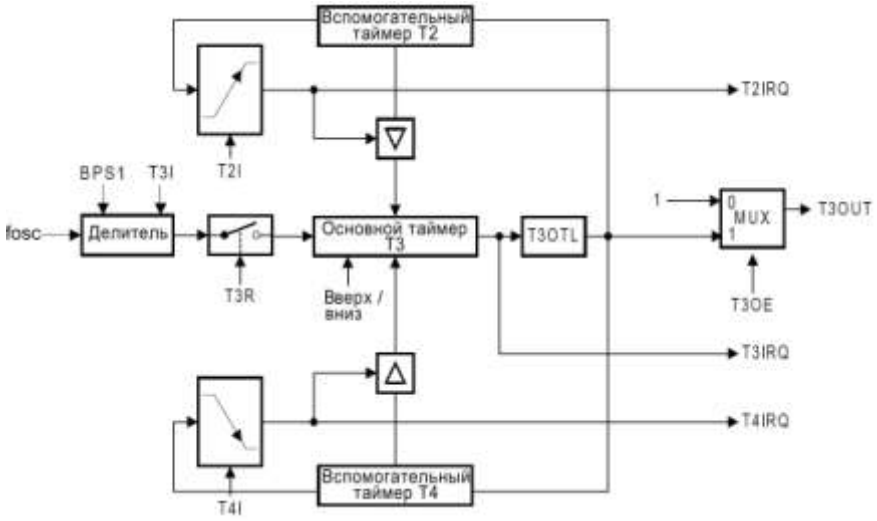


Рисунок 13.13 – Конфигурация блока GPT1 для формирования ШИМ-сигнала

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 T2IN/T4IN, инициирует загрузку содержимого основного таймера T3 в регистр вспомогательного таймера. Функциональная схема работы вспомогательного таймера в режиме захвата показана на рисунке 13.14.

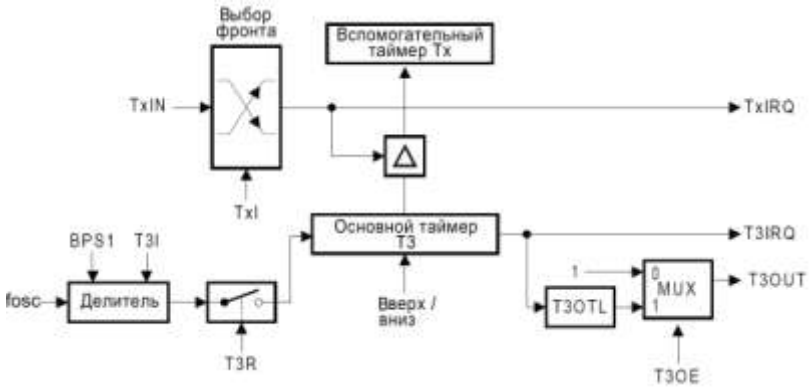


Рисунок 13.14 – Вспомогательный таймер Tx в режиме захвата

В режиме захвата два младших бита битового поля T2I/T4I 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 за исключениями:

- для таймеров T2 и T4 отсутствует выходные сигналы T2OUT и T4OUT;
- отсутствует контроль переполнения/опустошения.

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

В этом режиме тактовый сигнал fclk (в МГц) приходит на программируемый блок делителя, которым управляют битовые поля T6I и BPS2.

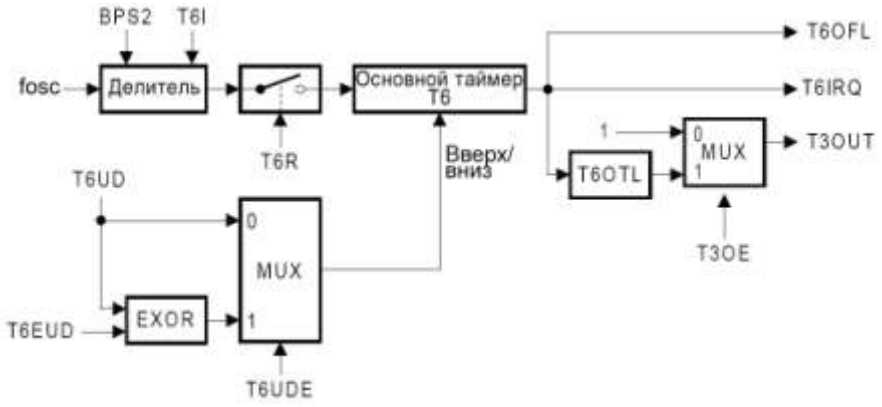


Рисунок 13.15 – Структурная схема работы таймера T6 в режиме таймера

Входная частота таймера ft6 (в МГц) и длительность такта rt6 (в мс) рассчитываются по формулам:

$$ft6 = \frac{fosc}{<BPS2> \times 2^{<T6I>}}, \quad (13.2)$$

$$rt6 = \frac{<BPS2> \times 2^{<T6I>}}{fosc} \quad (13.3)$$

T6 в режиме внешнего управления таймером

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

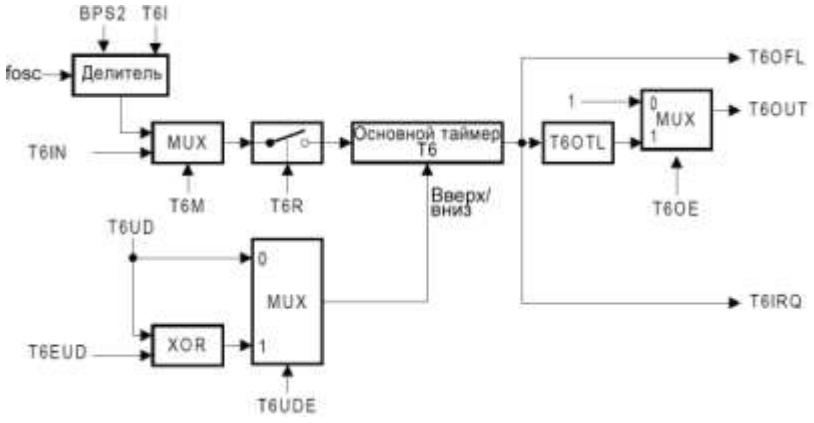


Рисунок 13.16 – Структурная схема работы таймера T6 в режиме внешнего управления таймером

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.

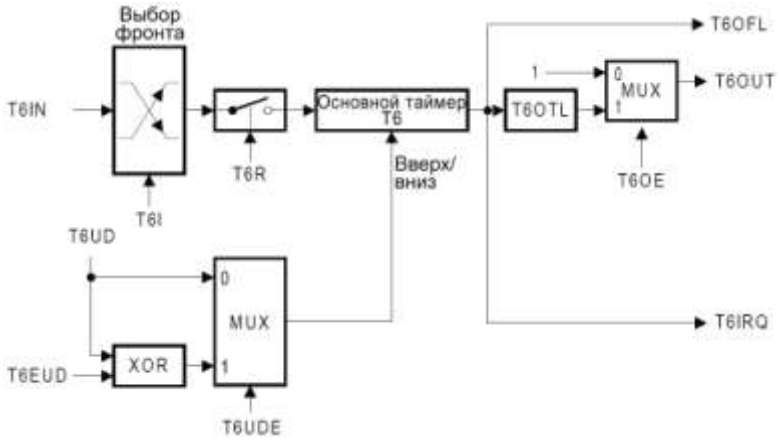


Рисунок 13.17 – Структурная схема работы таймера T6 в режиме счетчика

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.

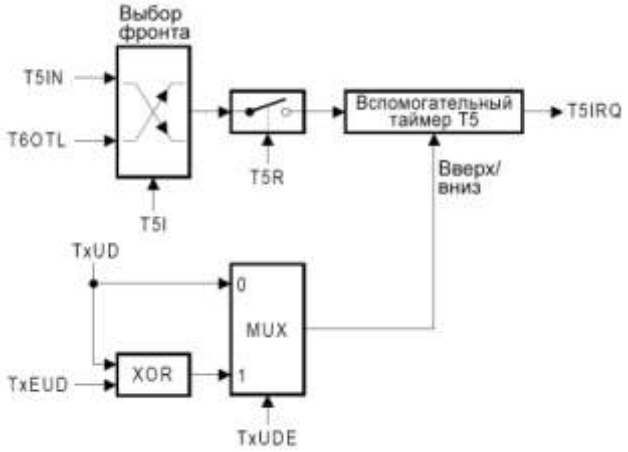


Рисунок 13.18 – Блок-схема вспомогательного таймера в режиме счетчика

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.

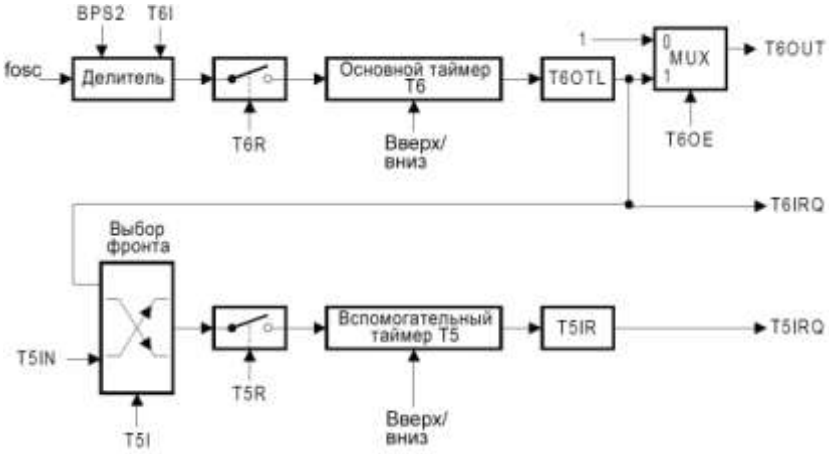


Рисунок 13.19 – Объединение основного таймера T6 и вспомогательного таймера T5

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.

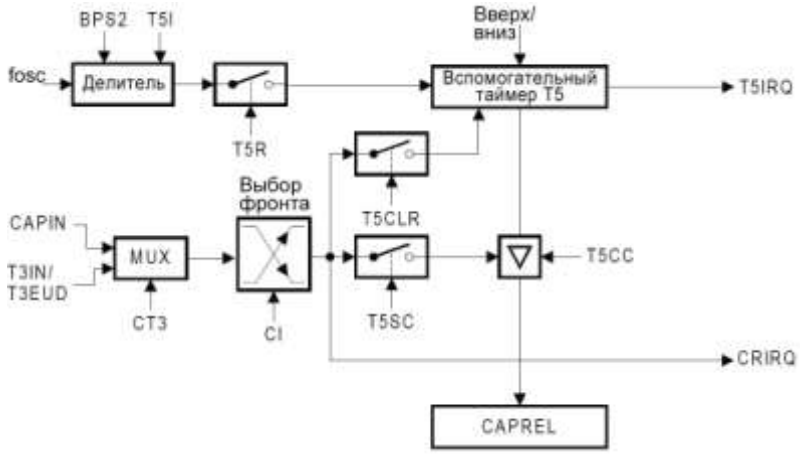


Рисунок 13.20 – Регистр CAPREL блока в режиме захвата таймера T5

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). 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.

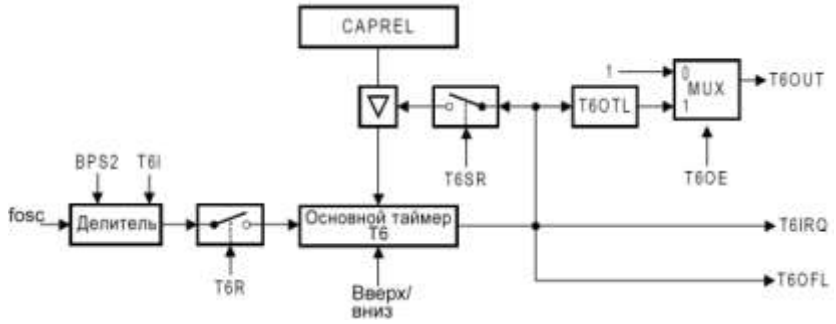


Рисунок 13.21 – Регистр CAPREL блока в режиме перезагрузки таймера T6

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.

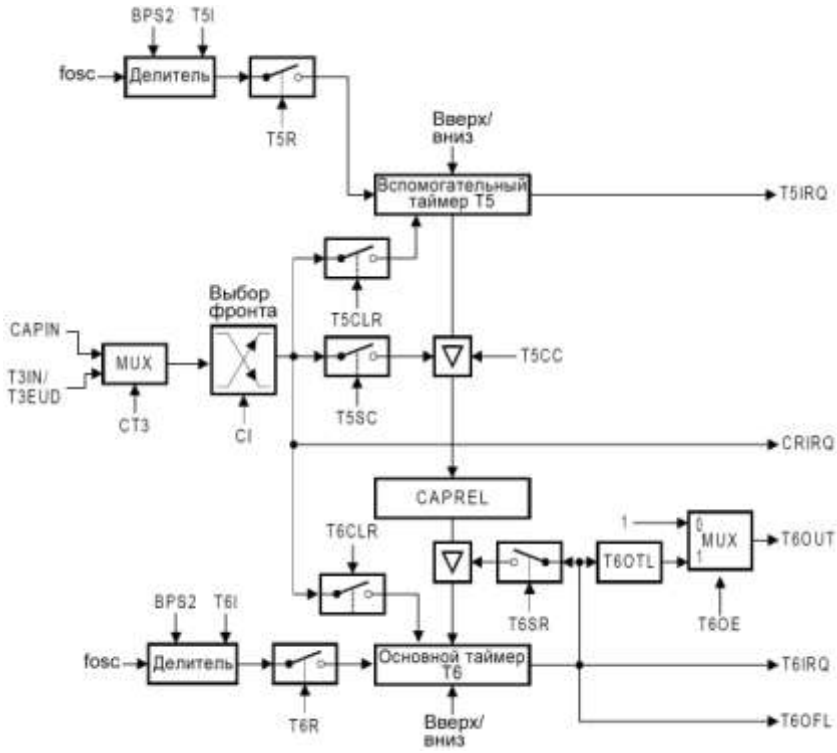


Рисунок 13.22 – Схема работы регистра CAPREL в комбинированном режиме

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 Приемопередатчики ASC0, ASC1, USART2

Асинхронно-синхронный последовательный интерфейс (ASC) обеспечивает передачу данных между микроконтроллером и другими микроконтроллерами и периферией. Микроконтроллер содержит два приемопередатчика ASC0, ASC1 и блок USART2.

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 1,5 Мбод (при частоте fper = 50 МГц);
- полудуплексный 8-битный синхронный режим;
- двойная буферизация при передаче/приеме;
- генерация прерываний;
- определение ошибок четности, фрейма, переполнения.

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. 13-разрядный таймер скорости пересылки данных с универсальной входной схемой делителя тактового генератора формирует 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 происходит 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 доступен только для чтения. Полученный бит четности также может считаться, если это разрешено. Старшие биты регистра SxRBUF будут читаться как нули.

Прием данных осуществляется с двойной буферизацией, поэтому прием очередных 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 включено обнаружение ошибки переполнения, флаги SxCON_OE и 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 и применяется для тестирования подпрограммы последовательной передачи данных. 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 записывать в поле M регистра SxCON зарезервированные комбинаций.

14.1 Асинхронные операции

Асинхронный режим поддерживает полнодуплексную коммуникацию, в которой передатчик и приемник используют одинаковый формат кадра данных и одинаковую скорость пересылки данных. Данные передаются по линии TxDx и принимаются на линии RxDx. На рисунке 14.1 представлена функциональная схема модуля ASCx в асинхронном режиме.

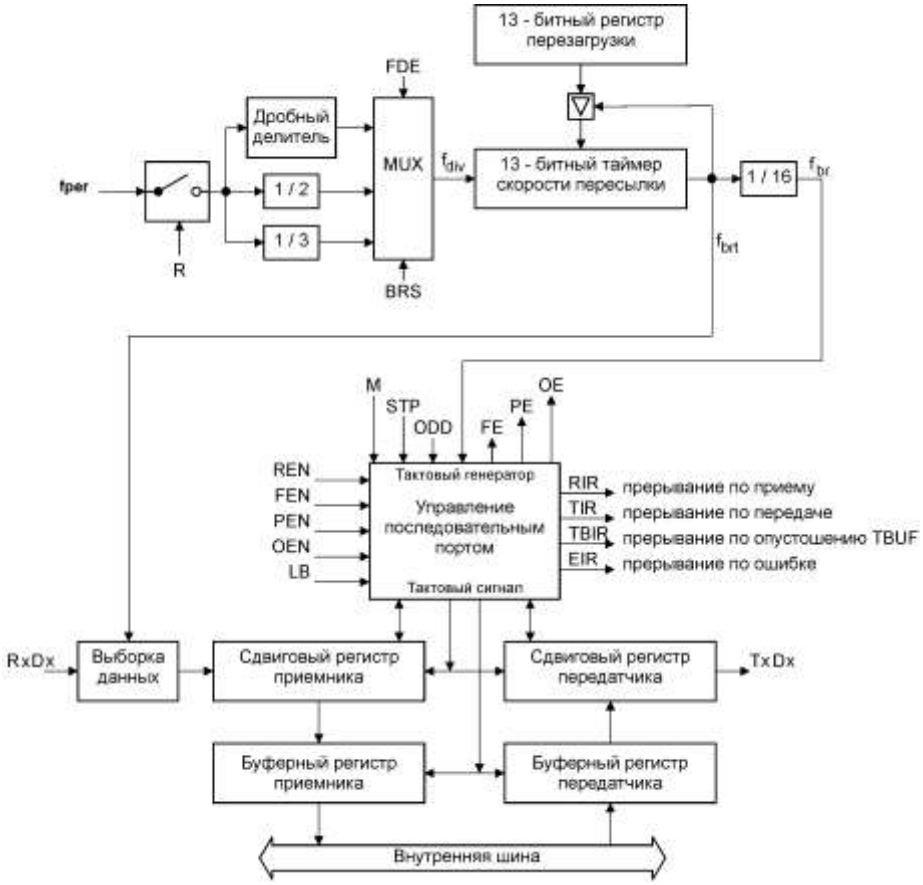


Рисунок 14.1 – Асинхронный режим работы интерфейса ASCx

Асинхронный 8-разрядный фрейм данных

8-разрядный фрейм данных состоит из любых восьми информационных бит данных D7 – D0 (при M = 001b) или из семи бит данных D6 – D0 и автоматически генерируемого бита четности (при M = 011b), см. рисунок 14.2.

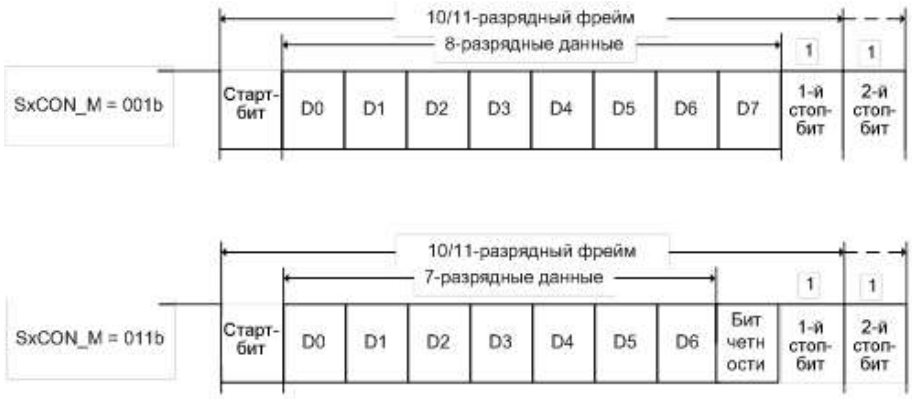


Рисунок 14.2 – Асинхронный 8-разрядный фрейм (младшим битом вперед)

Состояние бита четности зависит 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 сброшен. Проверка четности разрешается установкой бита PEN (при 8-разрядном режиме данных проверка всегда выключена). Флаг ошибки четности PE будет установлен вместе с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит четности будет сохранен в седьмом разряде регистра SxRBUF.

Асинхронный 9-разрядный фрейм данных

9-разрядный фрейм данных состоит из любых девяти информационных бит D8 – D0 (при M = 100b) или из восьми бит данных плюс автоматически генерируемого бита четности (при M = 101b), см. рисунок 14.3.

Рисунок 14.3 – Асинхронный 9-разрядный фрейм (младшим битом вперед)

Состояние бита четности зависит 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 бита PEN (при 9-разрядном режиме данных и режиме пробуждения проверка всегда выключена). Флаг PE будет установлен вместе с флагом запроса прерывания по ошибке, если будет получен неправильный бит четности. Бит четности будет сохранен в седьмом разряде регистра SxRBUF.

Асинхронная передача

Асинхронная передача начинается после того как был установлен бит R и данные были загружены в регистр SxTBUF. Передаваемый фрейм данных состоит из трех элементов:

- стартового бита;
- поля данных (восемь или девять бит, включая бит четности, если выбран);
- одного или двух стоповых битов.

Передаваемые данные, загруженные в буферный регистр передатчика, немедленно перемещаются в передающий сдвиговый регистр, таким образом, освобождая буфер для следующих данных. После очистки буфера генерируется прерывание TBIR «Буфер передатчика ASCx (регистр управления прерыванием SxTBIC) и буферный регистр может приять следующие данные, в то время как идет передача ранее загруженных данных. Перед передачей последнего бита фрейма будет сгенерирован запрос прерывания TIR «Передача ASCx» (регистр управления прерыванием SxTIC) и перед первым (или вторым) стоповым битом данные будут стерты из сдвигового регистра.

Асинхронный прием

Асинхронный прием активируется перепадом уровня сигнала на линии RxDx из единицы в ноль. Биты R и REN при этом должны быть установлены. За время приема одного бита данных выборка значения производится 16 раз, во избежание ошибочного определения значения бита.

Если после начала приема стартовый бит определяется как «1», то дальнейший прием прекращается, и устройство переходит в режим ожидания следующего стартового бита.

Если стартовый бит оказывается действительным, т.е. нулем, то устройство продолжает прием данных.

Когда последний стоповый бит принят, содержимое приемного сдвигового регистра передается в приемный буферный регистр SxRBUF. Одновременно с этим генерируется запрос прерывания RIR «Прием ASCx» (регистр управления прерыванием SxRIC), вне зависимости от правильности принятого значения последнего стопового бита.

После этого приемное устройство переходит в режим ожидания следующего стартового бита.

Асинхронный прием запрещается сбросом бита REN. Если бит сбрасывается во время приема данных, то прием будет продолжается до окончания передачи, по окончании которой будет сгенерировано прерывание (так же и прерывание по ошибке, если таковая будет обнаружена).

14.2 Синхронные операции

В синхронном режиме (M = 000h) осуществляется полудуплексная передача восьми бит данных. Тактовый сигнал, генерируемый внутренним генератором модуля ASCx, передается с выхода TxDx, данные передаются или принимаются через вывод RxDx синхронно импульсам тактового сигнала.

Синхронная передача

Синхронная передача начинается после загрузки данных в буферный регистр передачи SxTBUF, при условии, что установлен бит R, а бит REN сброшен.

Передаваемые данные, загруженные в буферный регистр, немедленно перемещаются в передающий сдвиговый регистр, освобождая буфер для следующих данных. После очистки буфера генерируется прерывание TBIR «Буфер передатчика ASCx и буферный регистр может приять следующие данные, в то время как идет передача ранее загруженных данных. После передачи последнего бита фрейма будет сгенерирован запрос прерывания TIR «Передача ASCx» и данные будут стерты из сдвигового регистра. На линиях RxDx и TxDx будет установлен высокий уровень сигнала.

Синхронный прием

Синхронный прием разрешается установкой бита REN. Если бит R установлен, данные принимаются по линии RxDx в сдвиговый регистр по фронтам тактового сигнала, который передается с выхода TxDx внешнему устройству.

Когда последний бит принят, содержимое приемного сдвигового регистра передается в регистр SxRBUF. Одновременно с этим генерируется запрос прерывания RIR «Прием ASCx» и бит REN аппаратно сбрасывается

Если бит REN сбрасывается во время приема данных, то прием продолжается до окончания передачи, по окончании которой будет сгенерировано прерывание (так же и прерывание по ошибке, если таковая будет обнаружена).

Запись байта в буфер SxTBUF во время приема данных не приведет к началу передачи.

Примечание – В петлевом режиме (установлен бит LB) бит REN должен быть установлен.

На рисунке 14.4 представлена функциональная схема модуля ASCx в синхронном режиме.

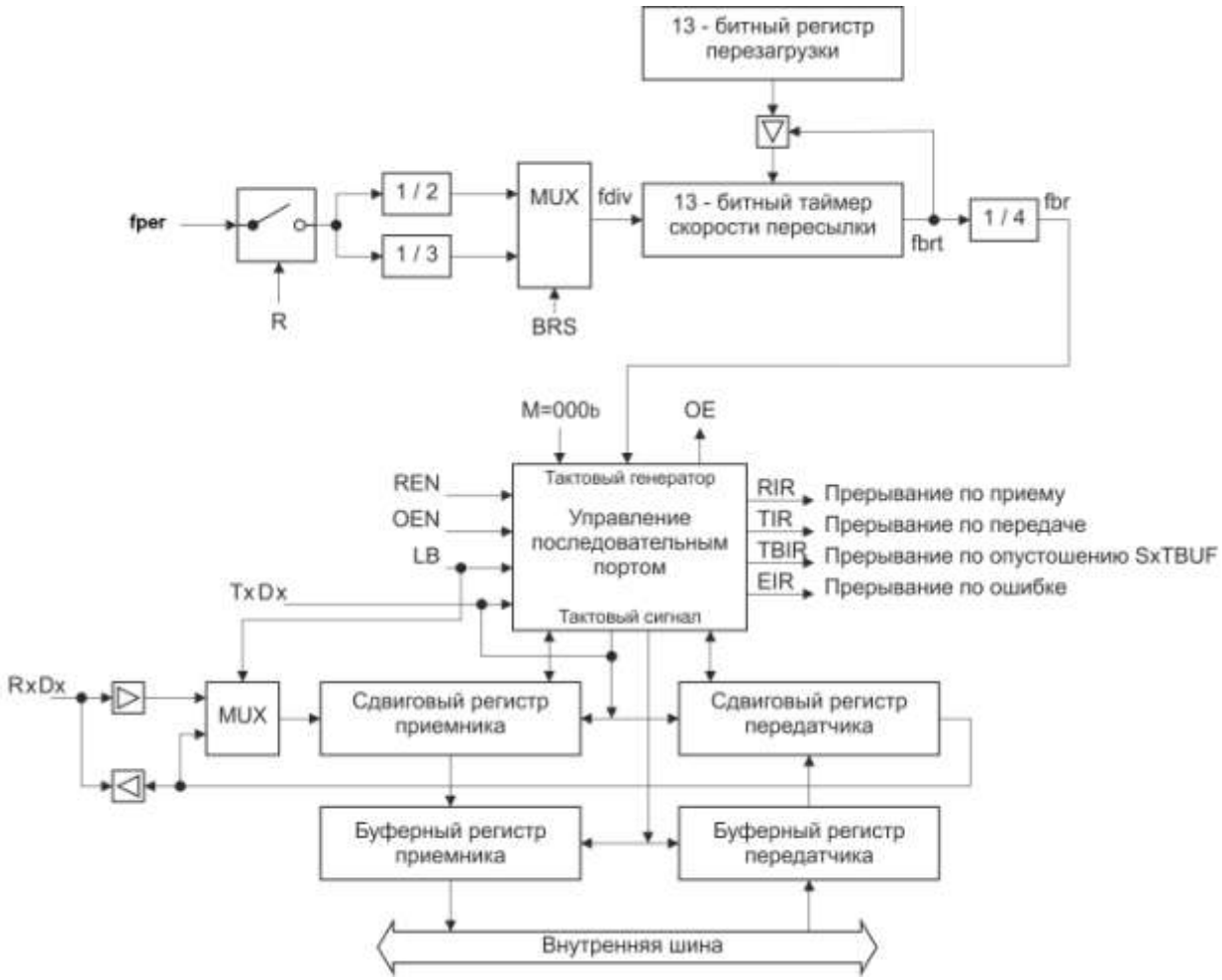


Рисунок 14.4 – Синхронный режим работы модуля ASCx

В неактивном состоянии уровень сигнала на выходе TxDx тактового генератора высокий. С началом синхронной передачи данных биты выставляются на линию RxDx по отрицательному фронту тактового сигнала. В режиме синхронного приема данные считываются с линии RxDx по положительному фронту тактового сигнала.

Между двумя последовательными передачами или приемами данных аппаратно вставляется задержка в один такт синхросигнала.

Временные диаграмма работы в синхронном режиме передачи и приема данных показаны на рисунках 14.5 и 14.6.

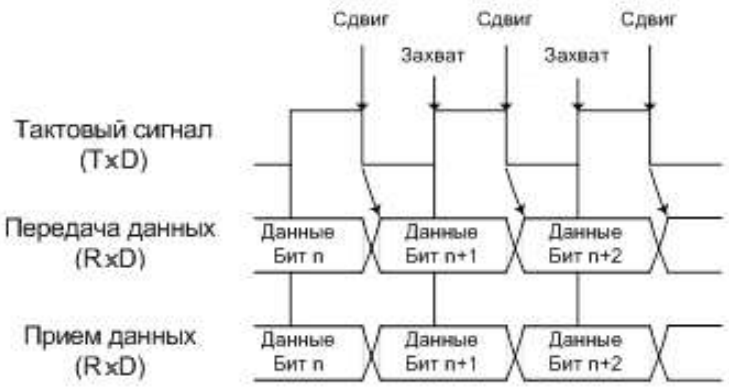


Рисунок 14.5 – Временная диаграмма работы в синхронном режиме

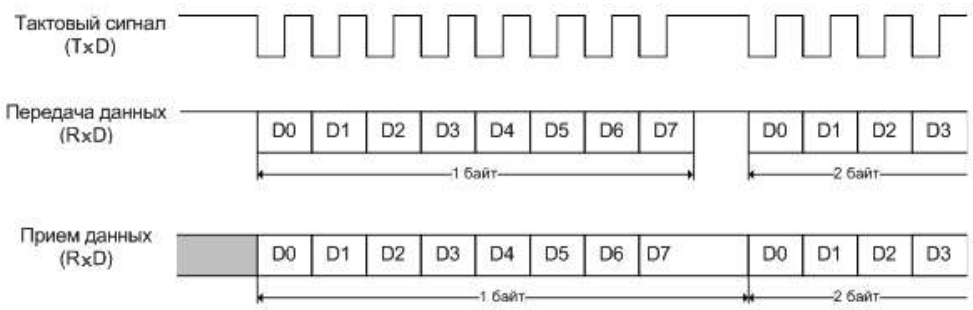


Рисунок 14.6 – Непрерывная передача данных в синхронном режиме

Примечания

1 Для передачи и приема данных должны быть включены альтернативные функции соответствующих выводов микроконтроллера, посредством установки битов ASCx_TXD и ASCx_RXD в регистре выбора альтернативных функций порта P3, и выводы должены быть сконфигурированы соответствующим образом как входы и выходы, посредством регистра DP3.

2 Если к моменту окончания передачи ранее принятые данные не были прочитаны из буферного регистра, то будет сгенерировано прерывание EIR «Ошибка ASCx» (регистр управления прерыванием SxEIC), а также установлен флаг OE (если бит OEN установлен).

14.3 Генератор скорости пересылки данных

Модуль ASCx имеет 13-разрядный обратный счетчик, синхронизируемый сигналом с частотой fdiv с выхода делителя частоты входного сигнала FPER. Счетчик тактирует схему передачи и выборки данных в асинхронных режимах и является генератором синхросигнала скорости передачи в синхронных. Регистр счетчика SxBG программируется (при сброшенном бите R) значением, которое зависит от желаемой скорости передачи данных. Чтение регистра возвращает текущее состояние счетчика.

Примечание – Запись в регистр SxBG при установленном бите R запрещена.

Каждый раз при установке бита R (включение генератора) или опустошении счетчика значение из регистра SxBG загружается в счетчик. Счетчик считает в обратном направлении и, если во время работы бит R сбрасывается, то – выключается.

Частота fbr сигнала, формируемого счетчиком, может быть изменена битами FDE и BRS, которые включают дополнительные делители. Бит FDE включает дробный делитель, позволяющий получать более высокую точность значения скорости передачи. Коэффициент дробного делителя задается регистром SxFDV.

Формулы для расчета значения BG, которое следует записывать в регистр SxBG для получения желаемой скорости передачи baudrate с выключенным дробным делителем, и формула для расчета значения коэффициента дробного делителя для асинхронных режимов передачи данных приведены в таблице 14.1.

Таблица 14.1 – Формулы расчета значений BG и FDV для получения желаемой скорости передачи данных в асинхронных режимах работы

Биты регистра SxCON		Формула расчета	Допустимые значения	
FDE	BRS		для BG	для FDV
0	0	$BG = \frac{fper}{32 \times baudrate} - 1$ (14.1)	0 – 8191	–

Окончание таблицы 14.1

FDE	BRS	Формула расчета	BG	FDV
0	1	$BG = \frac{fper}{48 \times baudrate} - 1$ (14.2)	0 – 8191	–
1	–	$FDV = \frac{baudrate \times 16 \times (BG + 1)}{fper} \times 512$ (14.3)	1 – 8191	1 – 511

Формулы для расчета значения BG, которые следует записывать в регистр SxBG для получения желаемой скорости передачи baudrate для синхронного режима передачи данных, приведены в таблице 14.2.

Таблица 14.2 – Формулы расчета значений BG для получения желаемой скорости передачи данных в синхронном режиме работы

Бит BRS регистра SxCON	Формула расчета	Допустимые значения для BG
0	$BG = \frac{fper}{8 \times baudrate} - 1$ (14.4)	1 – 8191
1	$BG = \frac{fper}{12 \times baudrate} - 1$ (14.5)	

14.4 Ошибки передачи

При обнаружении ошибки по окончании передачи данных модуль ASCx генерирует прерывание EIR «Ошибка ASCx» (регистр управления прерыванием SxEIC) и устанавливает флаг, соответствующий ошибке в регистре SxCON.

Модуль ASCx распознает три вида ошибок:

1 При включенной проверке фрейма (установлен бит FEN, включен асинхронный режим) не обнаружена единица в любом из ожидаемых стоповых битов. Устанавливается флаг FE.

2 При включенной проверке четности (установлен бит PEN, включен асинхронный режим) обнаружен неправильный результат. Устанавливается флаг PE.

3 При включенной проверке переполнения (установлен бит OEN) к моменту окончания приема данных, ранее принятые данные не были прочитаны из приемного буфера. Устанавливается флаг OE.

Примечание – Все три флага сбрасываются только программно.

14.5 Прерывания

Модуль ASCx генерирует четыре прерывания:

- TBIC – буфер для передаваемых данных пуст;
- TIC – окончание передачи данных;
- RIC – окончание приема данных;
- EIR – обнаружена ошибка по окончании передачи.

Если нет необходимости в получении данных, достаточно обслуживать только два типа прерываний – RIC и EIR. В случае одиночных передач достаточно прерывания TIR.

Для реализации непрерывной последовательной передачи данных, очередной байт должен быть записан в передающий буфер SxTBUF до окончания текущей передачи. Для этого следует использовать прерывание TBIR.

На рисунке 14.7 показаны моменты генерирования прерываний во время потока передаваемых данных.

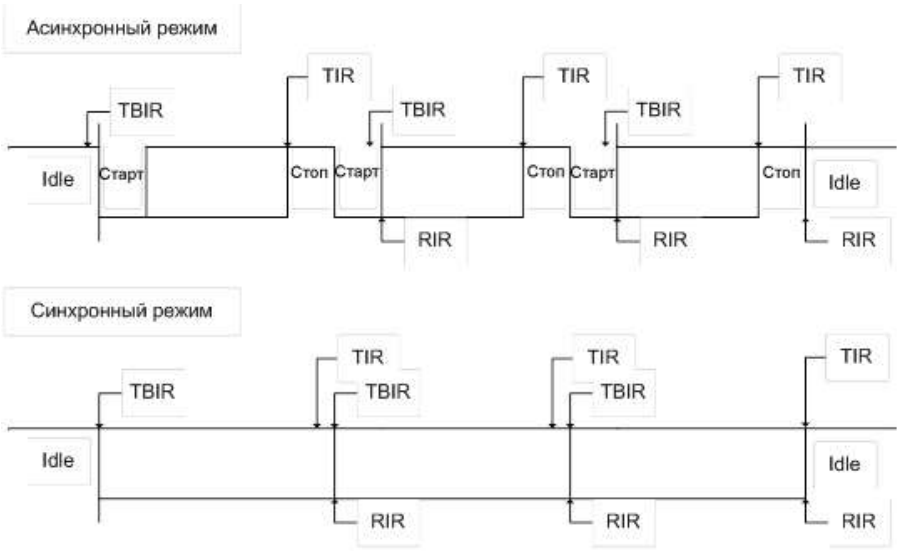


Рисунок 14.7 – Генерирование прерываний во время передачи данных

14.6 Приемопередатчик USART2

Блок USART2 представляет собой приемопередатчик, поддерживающий полнодуплексные асинхронные режимы и полудуплексный 8-битный синхронный режим передачи данных младшим битом вперед. Блок USART2 имеет двойную буферизацию передаваемых и принимаемых данных, систему обнаружения ошибок передачи и систему генерирования прерываний.

Описание функционирования для модулей ASC0 и ASC1 справедливо и для блока USART2. Работа с блоком осуществляется посредством регистров S2CON, S2TBUF, S2RBUF, S2BG, S2FDV, а также регистров прерываний S2EIC, S2RIC, S2TBIC и S2TIC. Вектора прерываний с номерами 72h – 75h, см. приложение B.

15 Синхронные приемопередатчики SSC0, SSC1, SPI

Синхронный последовательный интерфейс (SSC) обеспечивает высокоскоростную передачу данных между микроконтроллером и другими микроконтроллерами и периферией. Микроконтроллер содержит два приемопередатчика SSC0, SSC1 и блок SPI.

Особенности модуля SSCx:

- функционирование в режиме мастера или ведомого;
- полнодуплексный или полудуплексный обмен данными;
- двойная буферизация при передаче и приеме;
- 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.
- генерирование прерываний после передачи-приема данных и при обнаружении ошибок.

На рисунке 15.1 представлена функциональная схема модуля SSCx.

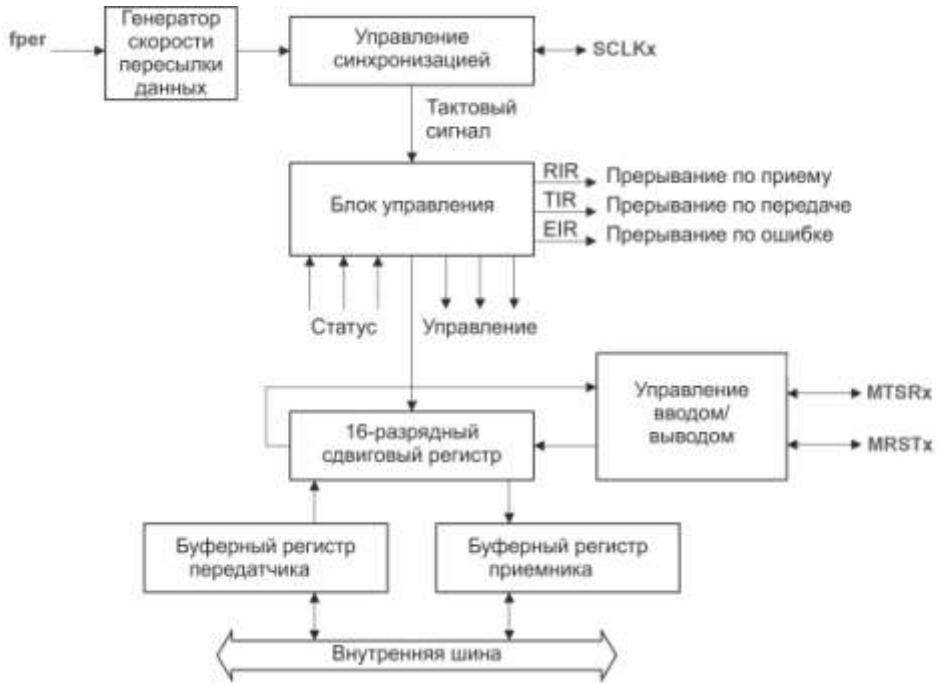


Рисунок 15.1 – Функциональная схема модуля SSCx

Данные передаются и принимаются по линиям MTSRx (мастер передает/ведомый принимает) и MRSTx (мастер принимает/ведомый передает). Синхросигнал передается по линии SCLKx.

Конфигурация модуля SSCx задается посредством регистров SSCxCON и SSCxTB.

Регистр сдвига подключен с помощью логики управления к выводу передачи и к выводу приема. Передача и прием данных синхронизированы, число переданных бит равно числу полученных.

В режиме мастера передача начинается после записи данных в передающий буферный регистр SSCxTB и перемещения их в сдвиговый регистр, если он пуст.

В режиме ведомого после записи в передающий буферный регистр SSCxTB данные сразу выставляются на линию передачи, и далее ведомый ожидает появление тактового сигнала.

Одновременно с началом передачи (появление первого фронта синхросигнала) устанавливается флаг занятости BSY (как в режиме мастера, так и в режиме ведомого) и генерируется прерывание TIR «Передача SSCx» (регистр управления прерыванием SSCxTIC), указывающее на то, что регистр SSCxTB может быть вновь загружен. После передачи запрограммированного числа битов содержимое сдвигового регистра перемещается в приемный буферный регистр SSCxRB, и генерируется прерывание RIR «Прием SSCx» (регистр управления прерыванием SSCxRIC). Если к этому моменту буфер SSCxTB пуст, то бит BSY сбрасывается.

Примечание – Флаг BSY устанавливается и сбрасывается только аппаратно.

Если к моменту окончания передачи и генерирования прерывания RIR в буфере SSCxTB находятся данные, то они автоматически загружаются в передающий сдвиговый регистр и передача продолжается. При этом не возникает пауза на линиях тактового сигнала и данных, т.е. передача двух байт выглядит как передача слова. В режиме непрерывной передачи может быть передано любое количество данных.

Независимо от направления передачи данных (задается битом HB) в регистрах SSCxTB и SSCxRB данные всегда расположены одинаково – самый старший бит слева, младший – справа.

Управление тактовым сигналом осуществляется посредством битов PO и PH. Один из фронтов тактового сигнала используется для выставления данных на линию передачи, а другой – для захвата данных. Бит PO задает уровень сигнала на линии SCLKx во время отсутствия передачи (в режиме ожидания), см. рисунок 15.2.

Примечание – Состояние битов PO и PH мастера и ведомых должно совпадать.

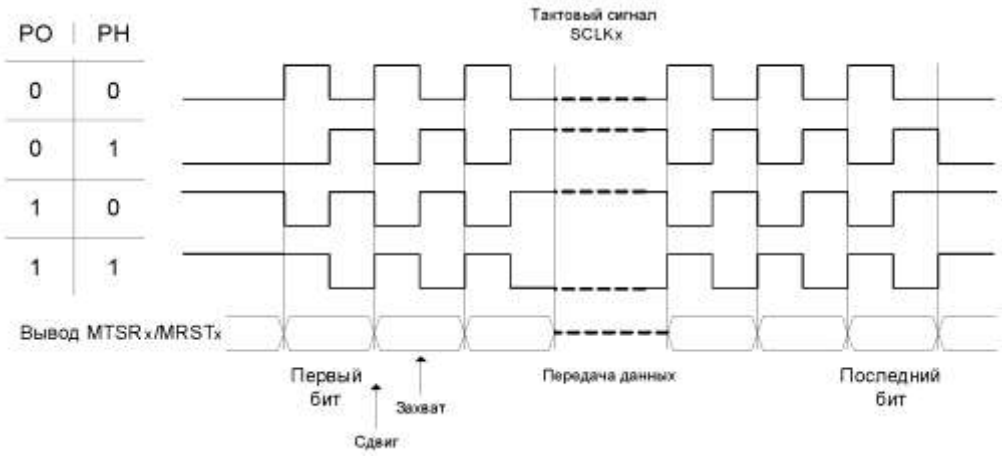


Рисунок 15.2 – Полярность и фаза тактового сигнала

Дуплексный режим

Обмен информацией осуществляется по линиям MTSRx (передача данных от мастерка к ведомым) и MRSTx (прием данных от ведомого). Тактирование передач осуществляется только мастером по линии SCLKx. Все ведомые устройства только принимают тактовый сигнал. Мастер и ведомый передают (сдвигают) и принимают (захватывают) данные синхронно по фронтам тактового сигнала, см. рисунок 15.2.

Все выходные линии MRSTx ведомых устройств должны быть объединены по принципу монтажного И с внешней подтяжкой до единицы и подключены к линии приема данных мастера с таким же названием. Все невыбранные ведомые удерживают свои выходы в состоянии единицы. Мастер выбирает ведомого, от которого ожидает передачу данных либо с помощью независимой линии выбора, либо с помощью специальной команды, отправленной ведомому. Только одно выбранное ведомое устройство включает

драйвер передачи линии MRSTx. По окончании передачи мастер может деактивировать ведомого. Схема интерфейса для дуплексного режима показана на рисунке 15.3.

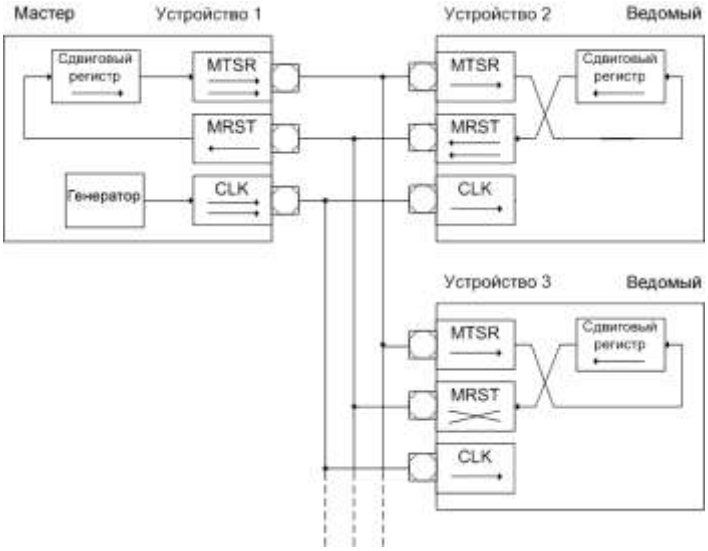


Рисунок 15.3 – Модули SSCx в дуплексном режиме работы

Полудуплексный режим

В этом режиме для приема и для передачи данных используется только одна линия – объединенные MTSRx и MRSTx каждого устройства. Тактирование передач осуществляется по линии SCLKx. Схема интерфейса для полудуплексного режима показана на рисунке 15.4

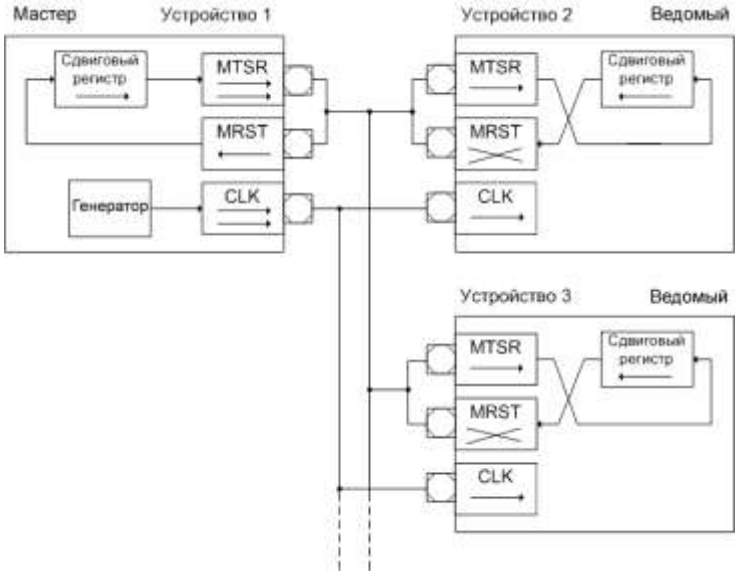


Рисунок 15.4 – Модули SSCx в полудуплексном режиме работы

Все выходные линии MRST–MRSTx устройств должны быть объединены по принципу монтажного И. Данные могут передаваться между любыми устройствами. Разрешение на передачу дается только одному устройству, которое на время передачи является мастером и источником синхросигнала. Вследствие объединения выводов MTSRx и MRSTx, передаваемые устройством данные одновременно принимаются им, что дает возможность детектирования искажений данных.

15.1 Управление скоростью передачи

Модуль SSCx имеет 16-разрядный обратный счетчик, синхронизируемый сигналом FPER с частотой fper. Счетчик является генератором синхросигнала передачи данных. Регистр счетчика SSCxBR программируется (в режиме программирования модуля, EN = 0) значением, которое зависит от желаемой скорости передачи данных. Чтение регистра (при EN = 0) возвращает текущее состояние счетчика.

Каждый раз при включении модуля или опустошении счетчика значение BR из регистра SSCxBR загружается в счетчик. Счетчик считает в обратном направлении и, если во время работы бит EN сбрасывается, то – выключается по окончании текущей передачи.

Для расчета значения BR, соответствующего желаемой скорости передачи baudrate, следует пользоваться формулой:

$$BR = \frac{fper}{2 \times baudrate} - 1 \qquad (15.1)$$

Примечание – В режиме мастера (при BR = 0000h) максимальное значение частоты генерируемого синхросигнала составит fper/2. В режиме ведомого максимальное значение частоты принимаемого синхросигнала не должно превышать fper/4.

15.2 Механизм определения ошибок

При обнаружении ошибки во время передачи данных модуль SSCx генерирует прерывание EIR «Ошибка SSCx» (регистр управления прерыванием SSCxEIC) и устанавливает флаг, соответствующий ошибке в ргистре SSCxCON. Программа обслуживания прерывания должна считывать состояние флагов для определения источника прерывания по ошибке

Модуль SSCx распознает четыре вида ошибок:

1 Ошибка приема. При включенном контроле приема (установлен бит REN) к моменту окончания передачи ранее принятые данные не были прочитаны из приемного буфера SSCxRB. Устанавливается флаг RE. После этого новые данные записываются поверх предыдущих.

2 Ошибка фазы. При включенной проверке фазы (установлен бит PEN) сигнал на входе MRSTx (в режиме мастера) или на входе MTSR (в режиме ведомого) изменяет свое значение в интервале, начинающемся за один такт сигнала FPER до прихода фронта тактового сигнала передачи и заканчивающемся через два такта после прихода фронта. Устанавливается флаг PE.

3 Ошибка скорости передачи (только для режима ведомого). При включенном контроле скорости передачи (установлен бит BEN) входной тактовый сигнал отклоняется от запрограммированного более чем на 100 %, т. е. скорость передачи в два раза больше или меньше ожидаемой. Устанавливается флаг ошибки BE.

Использование этой функции позволяет определять дополнительные ложные или пропущенные такты на линии передачи тактового сигнала. Если на момент возникновения ошибки скорости передачи установлен бит REN, 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.

4 Ошибка передачи (только для режима ведомого). При включенном контроле (установлен бит TEN) к моменту начала передачи (получен синхросигнал от мастера) значение передающего буфера SSCxTB не было изменено после окончания предыдущей передачи. Устанавливается флаг TE.

Ведомый будет передавать данные принятые от мастера во время предыдущей передачи.

Примечания

1 В полудуплексном режиме это может привести к искажению данных на линии, если ведомый не выбран для передачи. В связи с этим все ведомые, не выбранные для передачи, должны содержать в своих буферах SSCxTB значение FFFFh.

2 Все четыре флага сбрасываются только программно.

Прерывание EIR может быть сгенерировано программно, установкой флага в регистре SSCxCON. В связи с этим программа обслуживания прерывания должна сбрасывать соответствующий флаг, чтобы избежать повторного генерирования прерывания.

15.3 Блок SPI

Блок SPI представляет собой приемопередатчик, поддерживающий дуплексный и полудуплексный режимы передачи данных. Блок SPI имеет двойную буферизацию передаваемых и принимаемых данных, систему обнаружения ошибок передачи и систему генерирования прерываний.

Описание функционирования для модулей SSC0 и SSC1 справедливо и для блока SPI. Работа с блоком осуществляется посредством регистров SSC2CON, SSC2TB, SSC2RB, SSC2BR, а также регистров прерываний SSC2EIC, SSC2RIC и SSC2TIC. Векторы прерываний с номерами 6Fh – 71h см. приложение В.

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, см. рисунок 16.1.

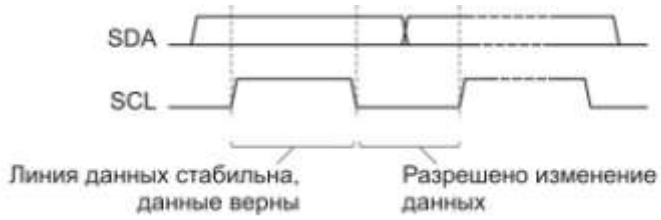


Рисунок 16.1 – Передача бита данных

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.

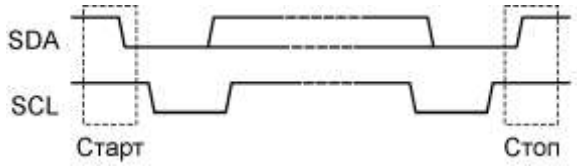


Рисунок 16.2 – Состояния старта и стопа

После того, как сформировано состояние старта, шина считается занятой и другие устройства не должны пытаться управлять ею. 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.

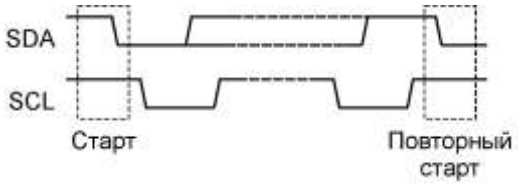


Рисунок 16.3 – Состояние повторного старта

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.

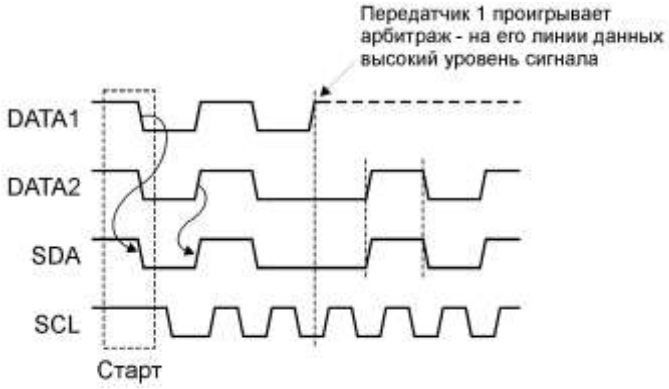


Рисунок 16.4 – Арбитраж на линии SDA

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.

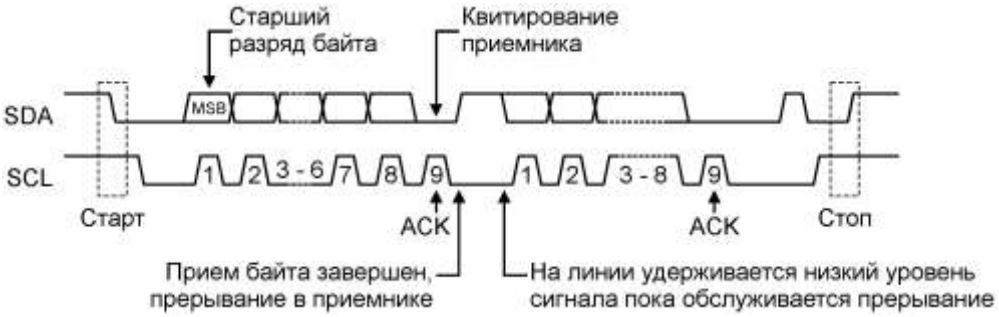


Рисунок 16.6 – Передача данных

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.

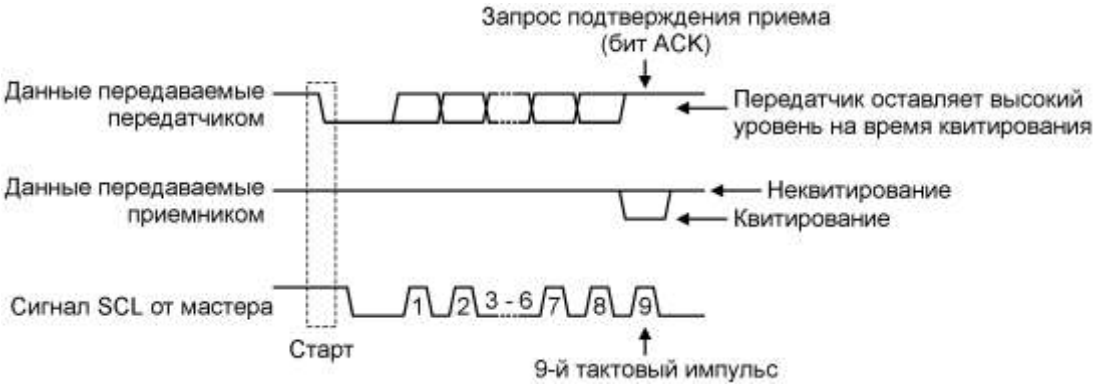


Рисунок 16.7 – Квитирование и неквитирование бита подтверждения ACK

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 монтажного «И». К этой линии также подключаются все устройства. Когда ведомому (или нескольким ведомым) необходимо обратиться к мастеру, он 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.

П р и м е ч а н и е – Описываемый 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 1024 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.

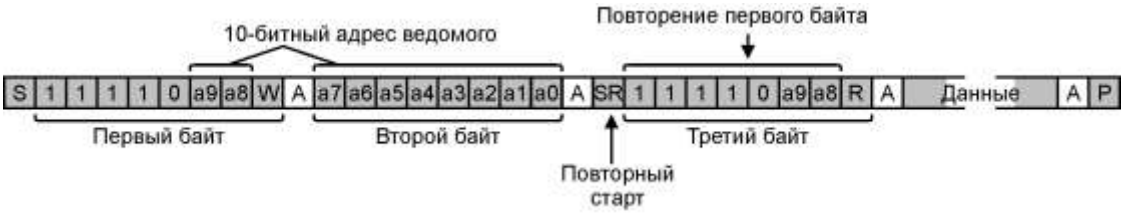


Рисунок 16.11 – Получение данных от ведомого с 10-битным адресом

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

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.

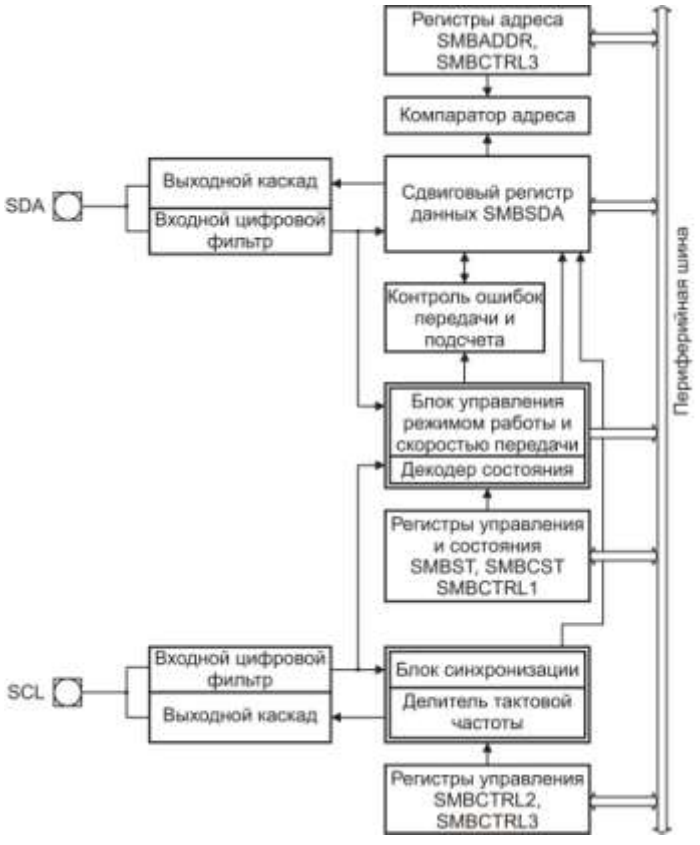


Рисунок 16.12 – Структурная схема модуля I2C

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.

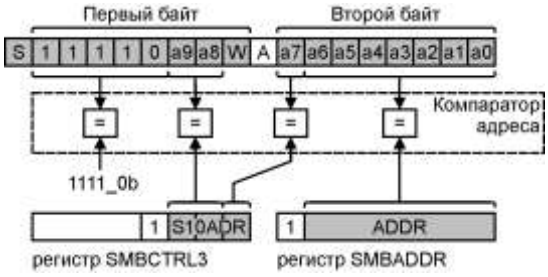


Рисунок 16.13 – Компаратор адреса в режиме 10-битной адресации

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.

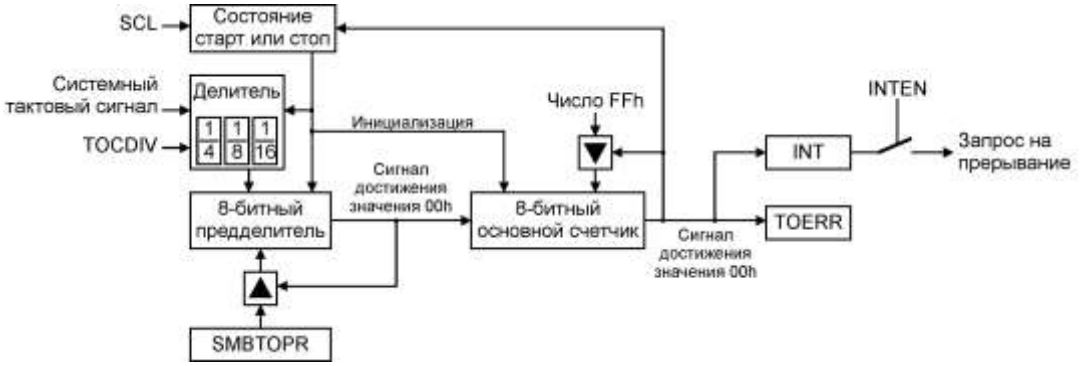


Рисунок 16.14 – Функциональная схема счетчика времени ожидания

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 SMBCTLR2 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.

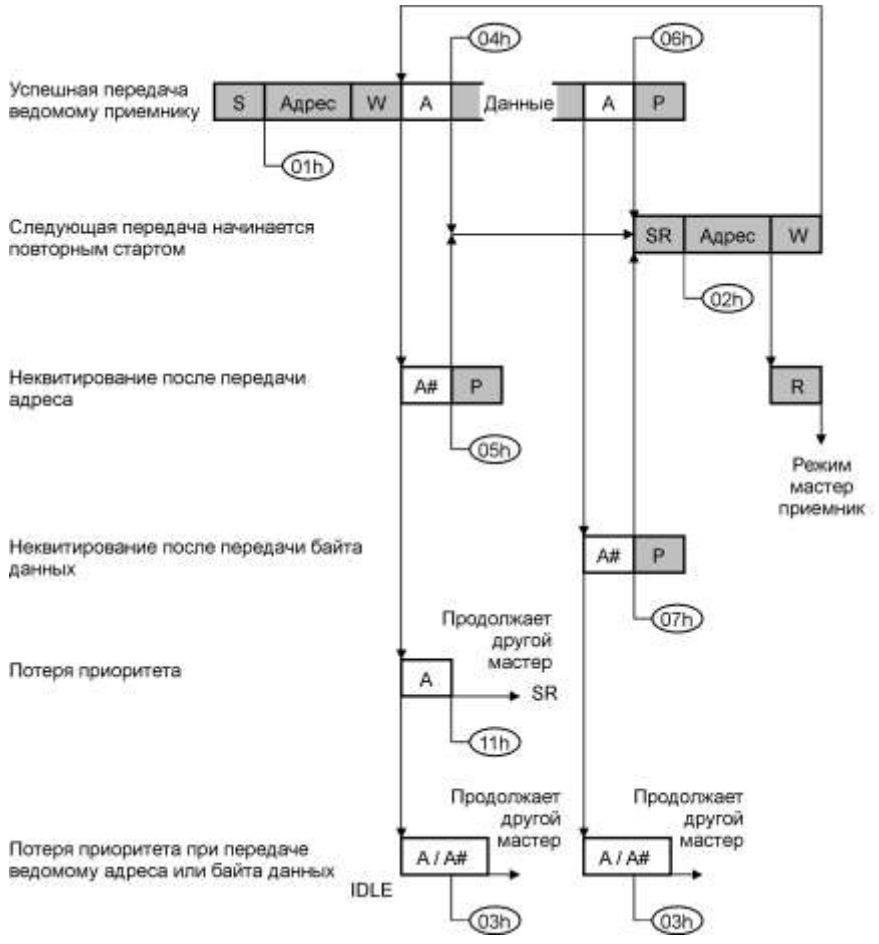


Рисунок 16.16 – Режим FS мастера передатчика

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.

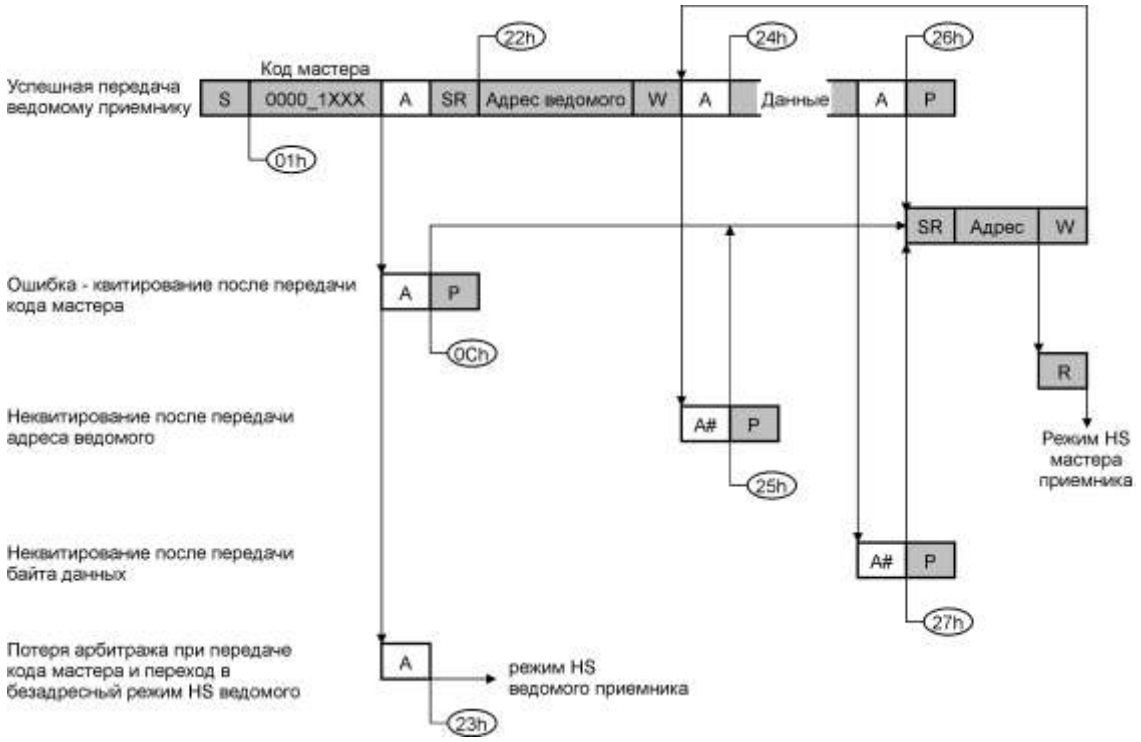


Рисунок 16.17 – Режим HS мастера передатчика

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.

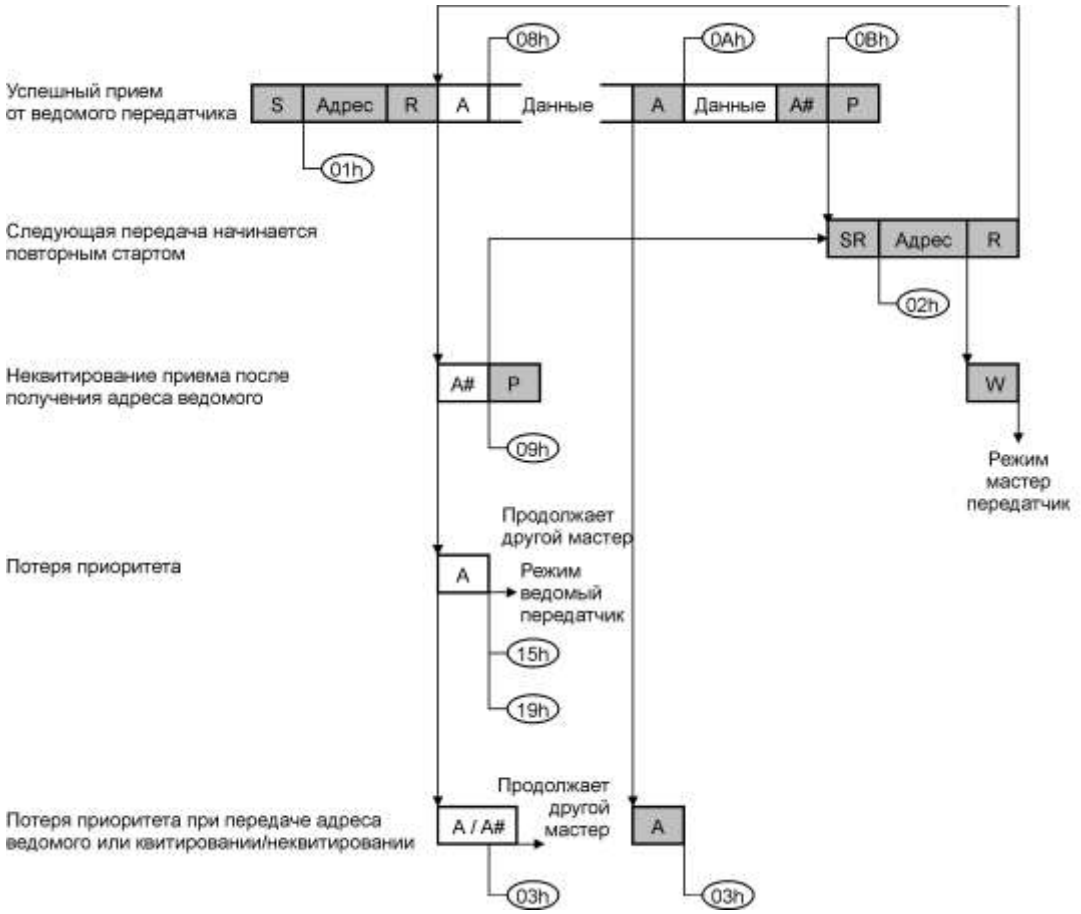


Рисунок 16.18 – Режим FS мастера приемника

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.

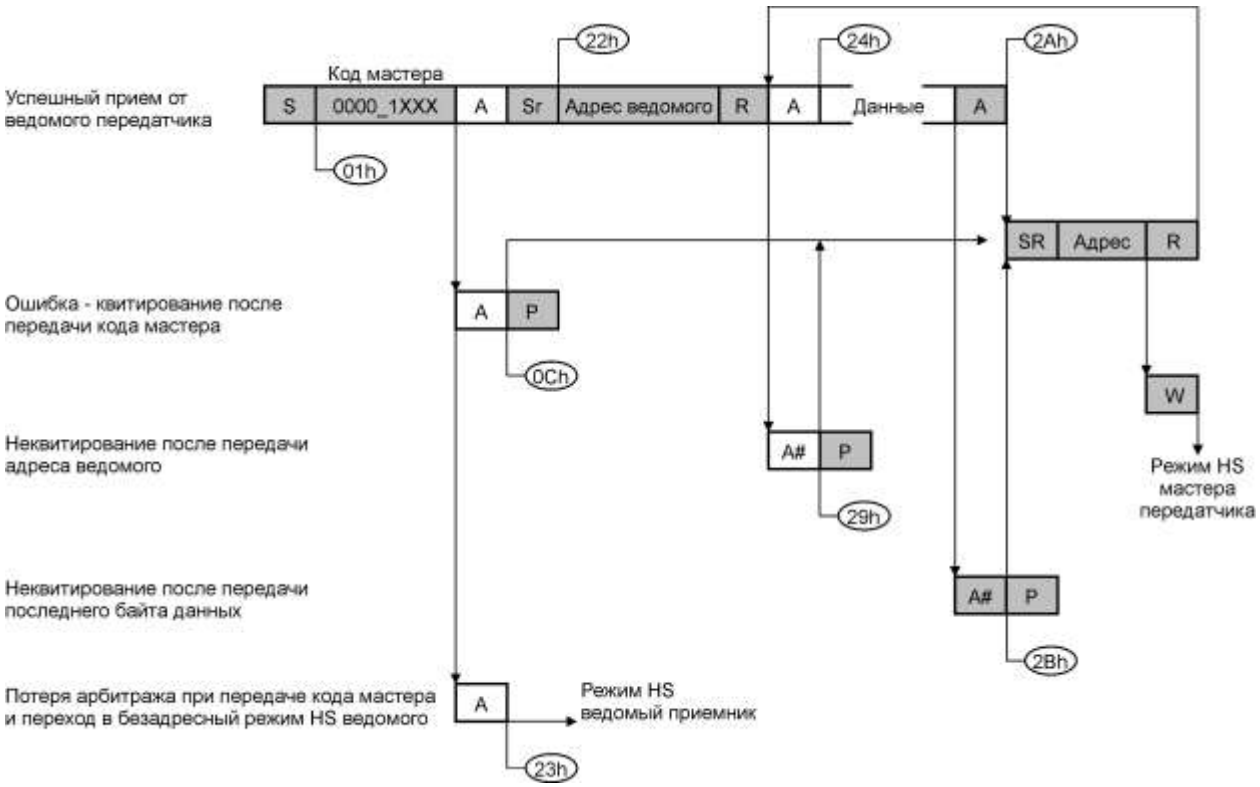


Рисунок 16.19 – Режим HS мастера приемника

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.

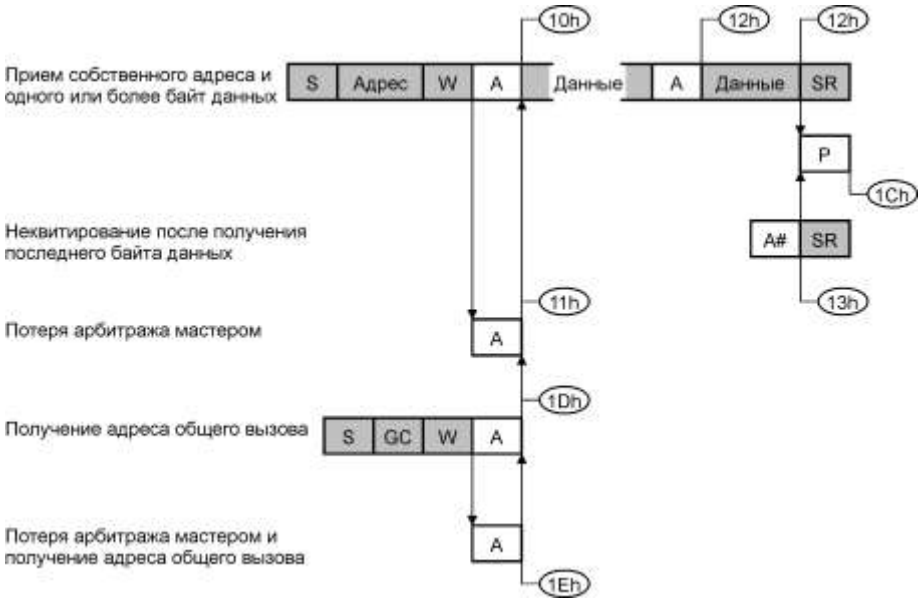


Рисунок 16.20 – Режим FS ведомого приемника

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.

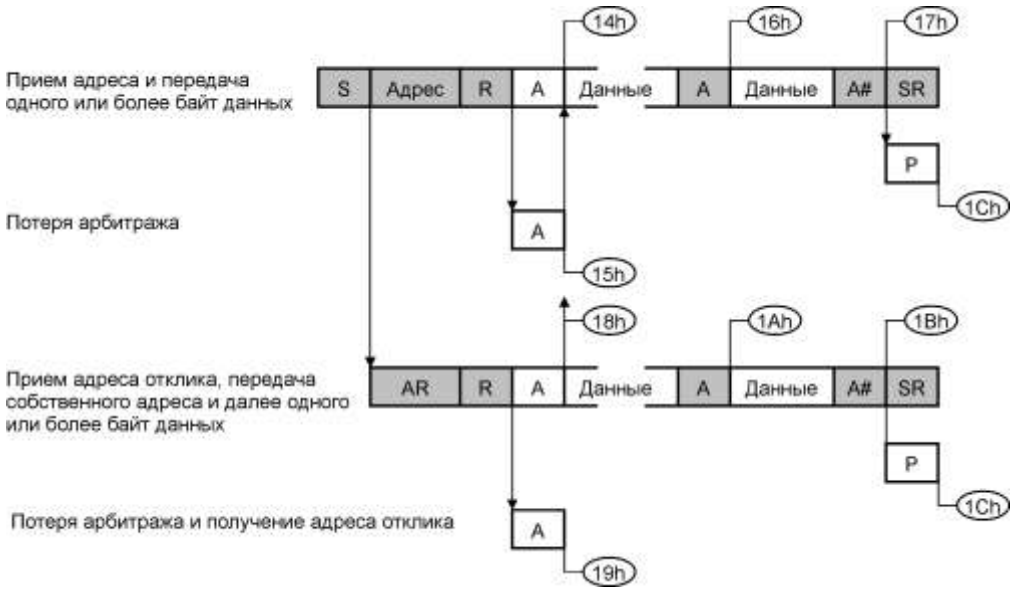


Рисунок 16.22 – Режим FS ведомого передатчика

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.

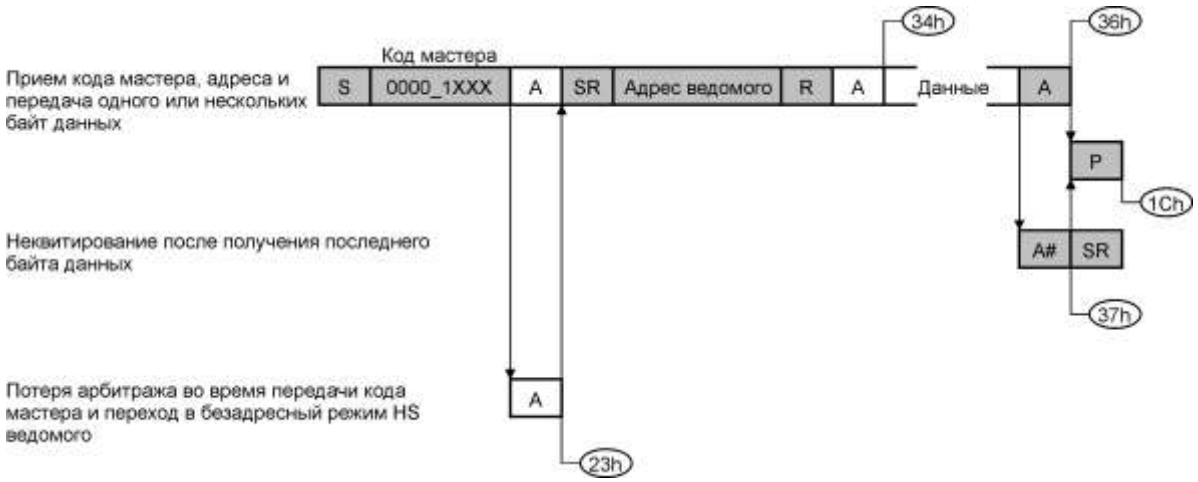


Рисунок 16.23 – Режим HS ведомого передатчика

Дополнительная информация о работе модуля

1 Когда модуль I2C выключен, бит BB регистра SMBCST очищен. Включение 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

17.1 Протокол 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.

Высокая скорость передачи данных (до 1 Мбит/с), хорошая помехозащищенность протокола, защита от неисправности узлов – делают шину CAN подходящей для индустриальных приложений управления типа Device Net.

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.

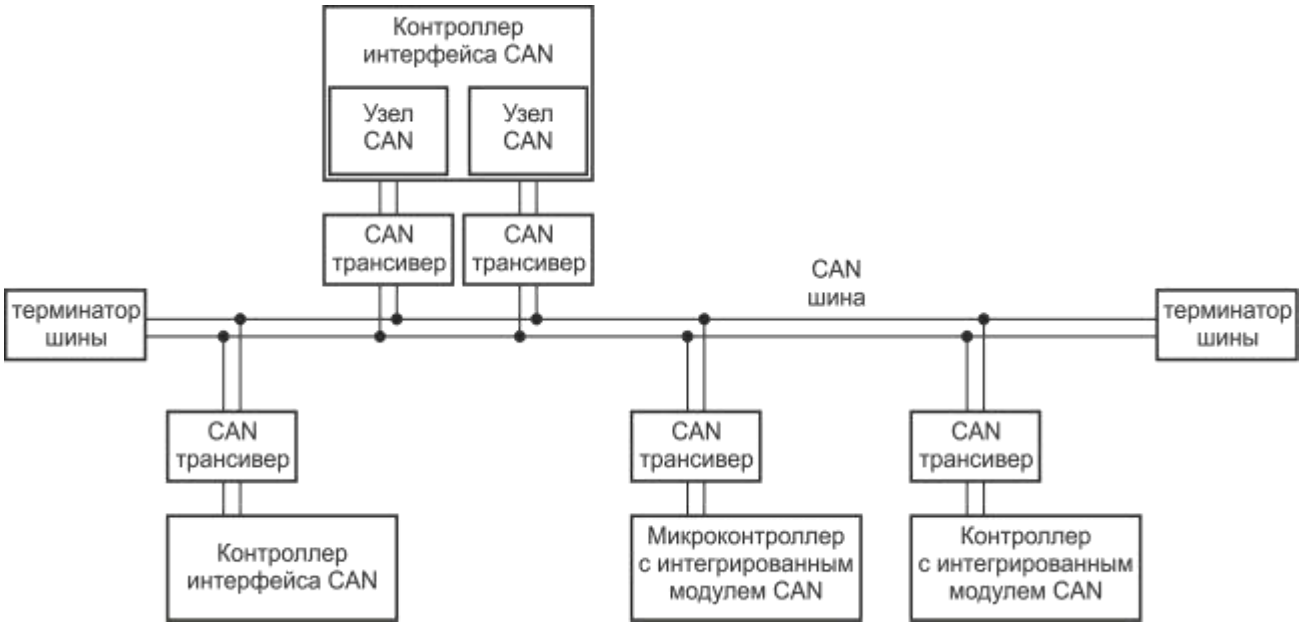


Рисунок 17.1 – Общая структура CAN сети

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.

Типы и структура сообщений CAN

Стандартное сообщение данных

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

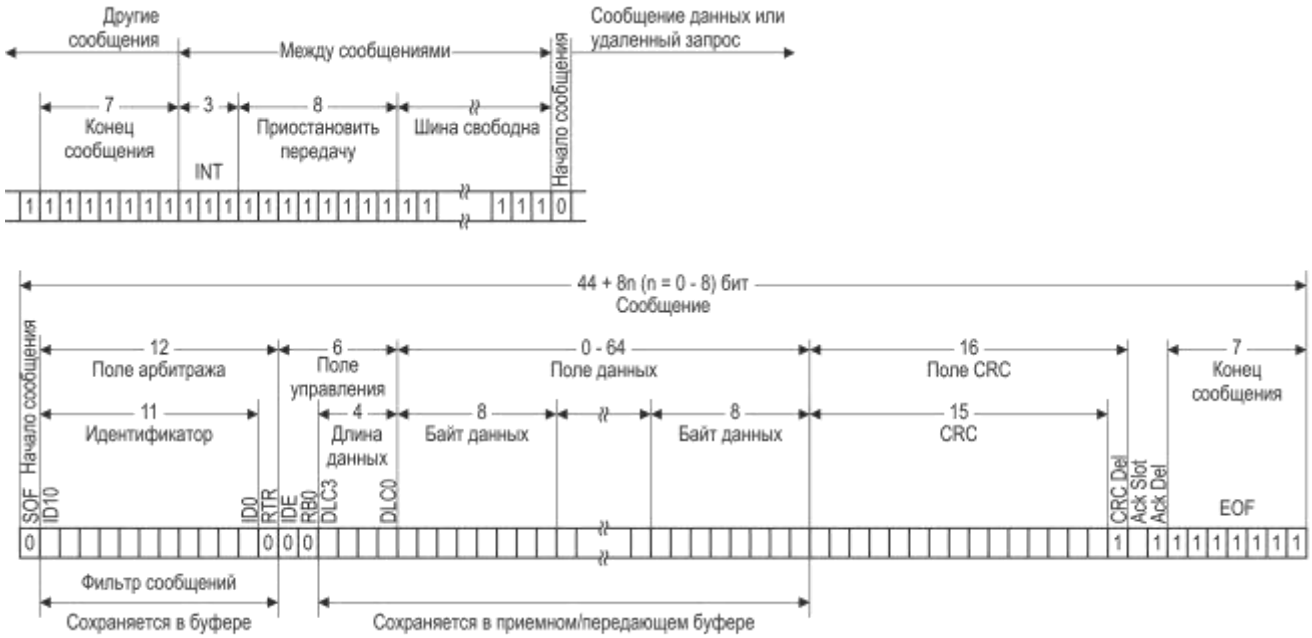


Рисунок 17.2 – Стандартное сообщение данных

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 CRCDel 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 ACKDel 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.

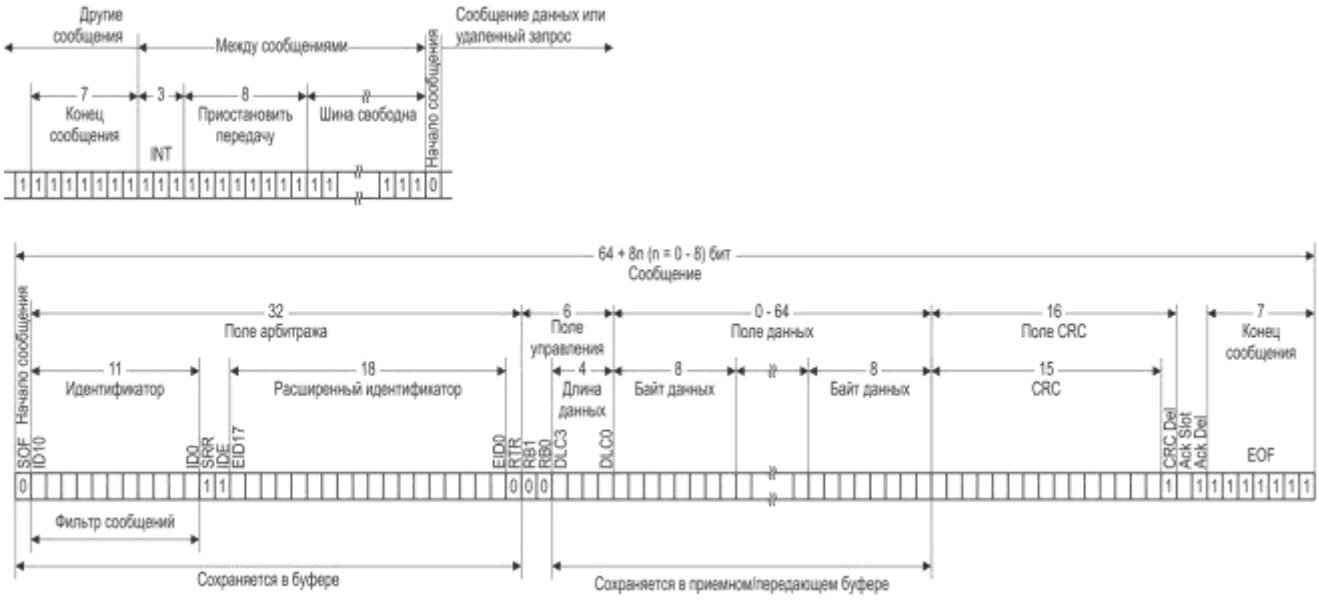


Рисунок 17.3 – Расширенное сообщение данных

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 CRCDel 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 ACKDel 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 рисунке 17.4 представлен вариант удаленного запроса со стандартным идентификатором).

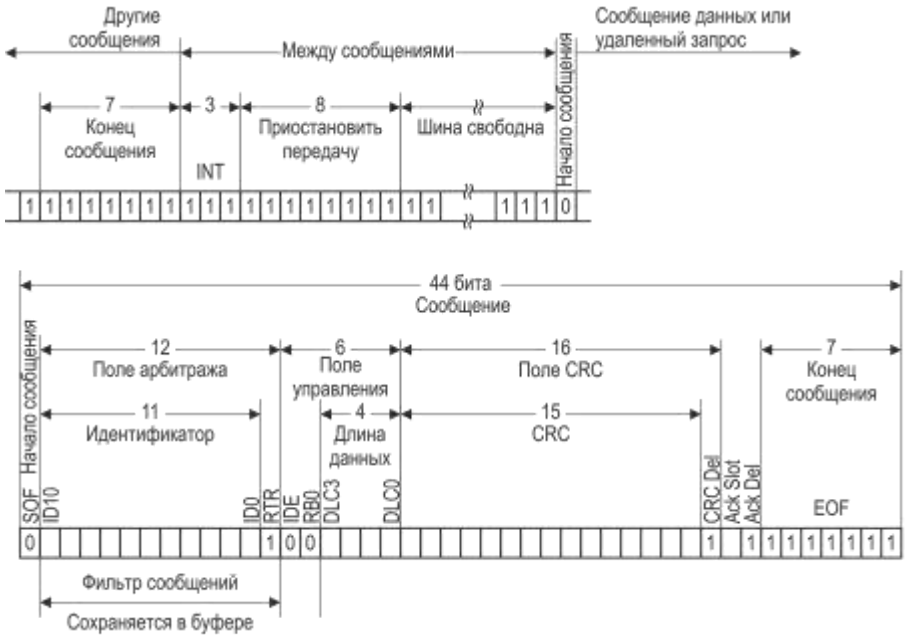


Рисунок 17.4 – Удаленный запрос данных (стандартный формат)

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.

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-стаффинга на шине и не оказывает влияния на передачи других узлов. Исключение составляет узел, который передает данные узлу,

обнаружившему ошибку. В этом случае правила бит-стафинга нарушаются и передача данных прекращается. После передачи пассивной ошибки узел должен ожидать 6 последовательных рецессивных битов для восстановления связи с шиной.

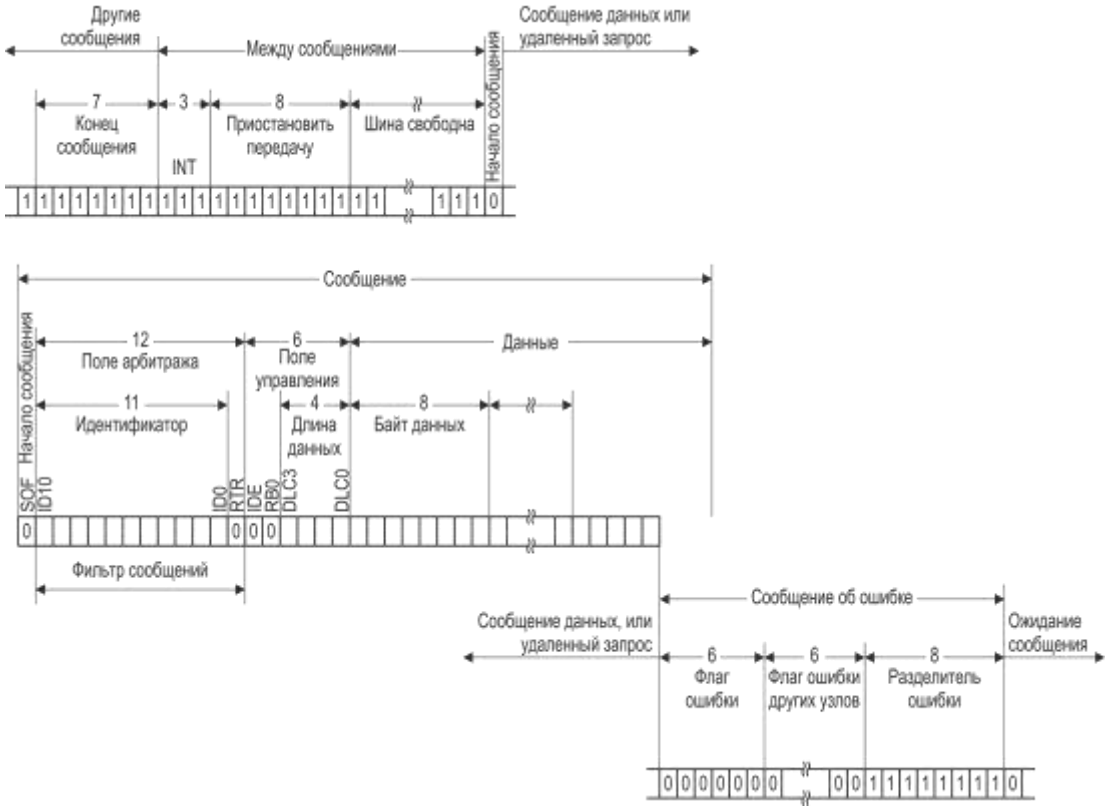


Рисунок 17.5 – Сообщение об ошибке

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 контроллера CAN

В состав контроллера CAN входят четыре идентичных независимых узла CAN0, CAN1, CAN2 и CAN3, ОЗУ для хранения сообщений, которое является общим для узлов, и система управления, см. рисунок 17.7.

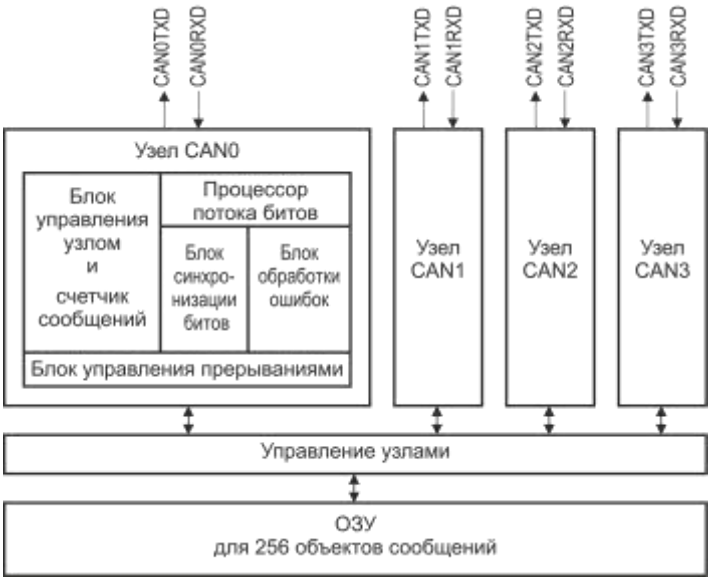


Рисунок 17.7 – Контроллер интерфейса CAN

Каждый узел контроллера CAN имеет следующие функциональные особенности:
- соответствие стандарту ISO 11898;
- функционирование согласно спецификации CAN 2.0b (активная версия);
- отдельные управляющие регистры;
- программируемая скорость передачи информации до 1 Мбит/с;
- гибкий и полный контроль передачи сообщений и обработки ошибок;

Контроллер CAN реализует 16 линий прерываний и 256 объектов сообщений 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 256 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).

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.

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 NCR 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 сообщения MOSTAT. Прерывания после приема генерируются (если это разрешено) для всех принятых сообщений, не содержащих ошибок.

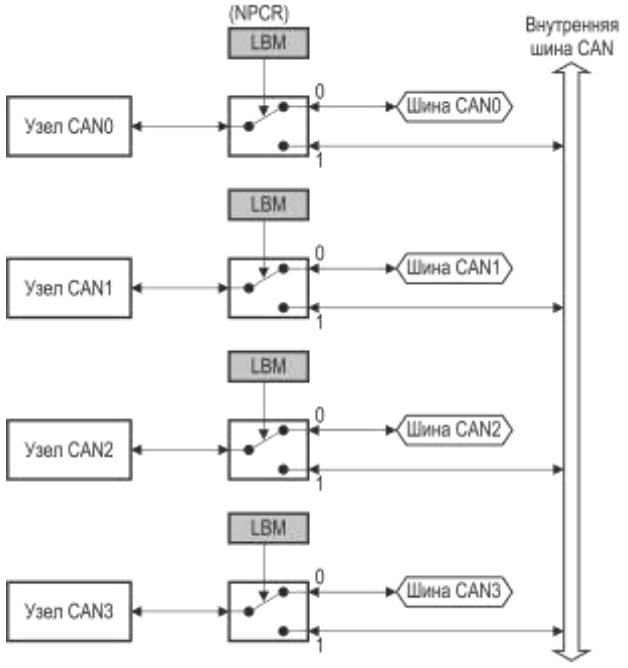


Рисунок 17.8 – Режим внутренней петли

Режим внутренней петли включается установкой бита LBM регистра NPCR 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 узлы 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.

Дробный делитель

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

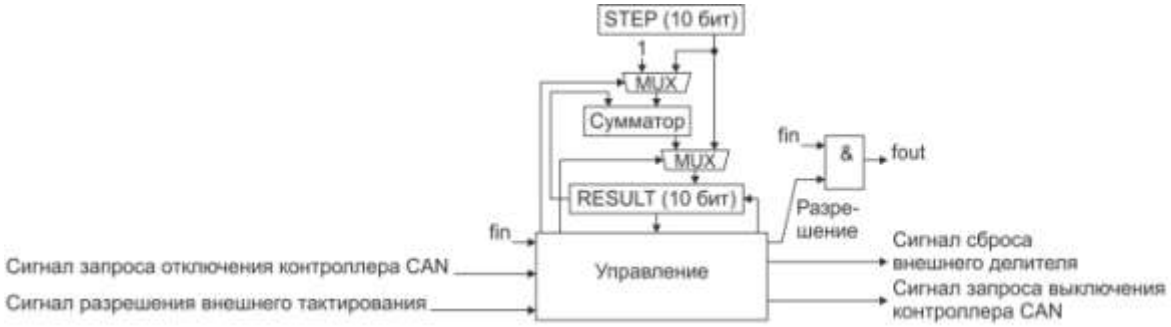


Рисунок 17.9 – Схема дробного делителя

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 finвв МГц следует пользоваться формулой:

$$fin = n \times Ntq, \qquad (17.1)$$

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.

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:

$$fout = fin \times 1/(1024 - STEPd), \qquad (17.2)$$

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).

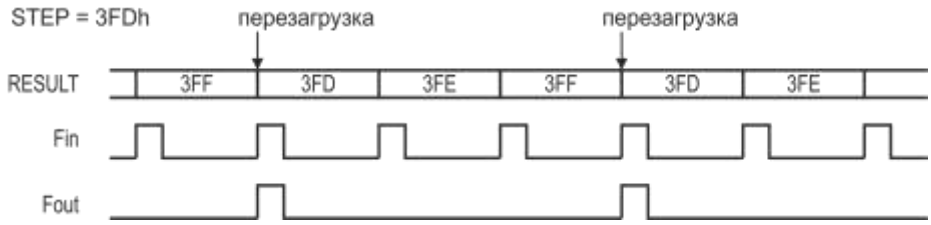


Рисунок 17.10 – Формирование сигнала с частотой fout в нормальном режиме

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:

$$fout = fin \times STEPd / 1024d \qquad (17.3)$$

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).

$$fout = fin \times 564/1024 = 0{,}55 \times fin$$

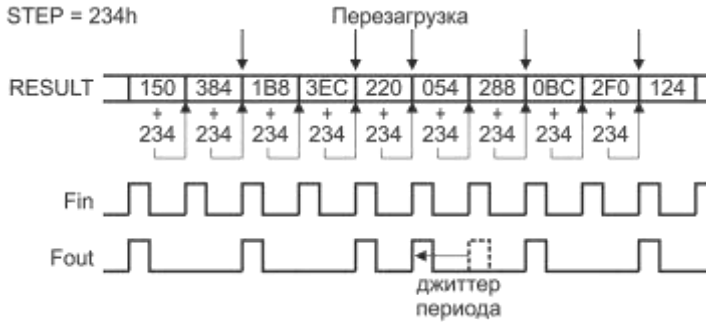


Рисунок 17.11 – Формирование сигнала с частотой fout в режиме дробного деления

Процесс выключения делителя начинается одновременно с возникновением запроса выключения контроллера CAN.

Контроллер сообщений

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

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

Управление прерываниями контроллера CAN

На рисунке 17.12 показана структура формирования запроса на прерывание.

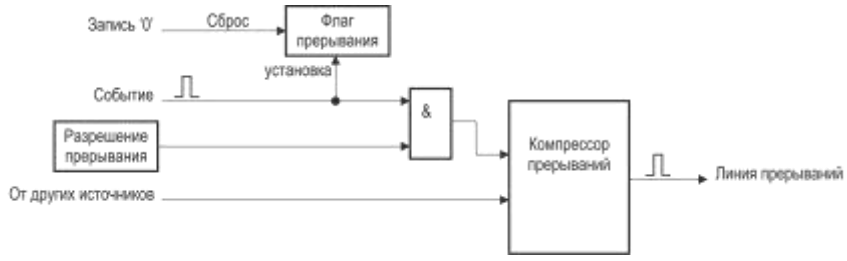


Рисунок 17.12 – Структура формирования запроса на прерывание

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.

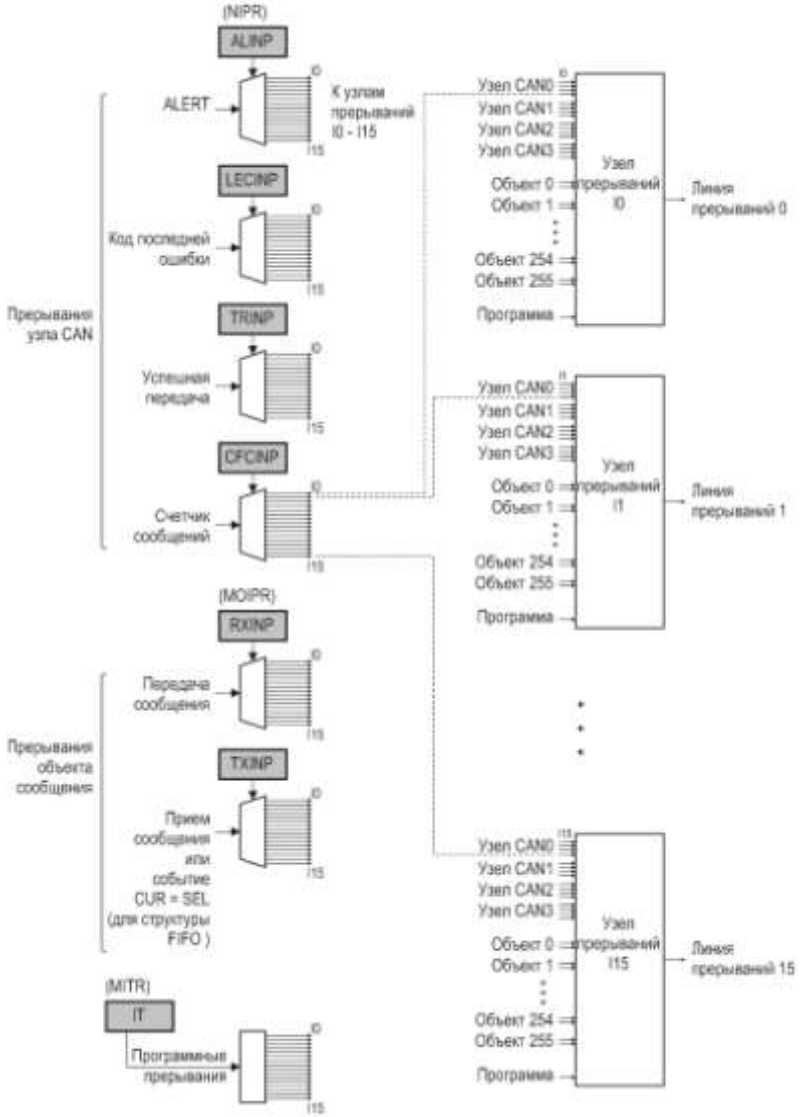


Рисунок 17.13 – Схема коммутации линий прерываний

Источниками прерываний являются:

- узлы CAN (16 источников – по 4 для каждого узла);
- объекты сообщений (512 источников – 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 17.13 представлена схема коммутации линий 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 MOIPR 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 контроллере CAN предусмотрен механизм распределения приоритетов для объектов сообщений.

17.3 Узел контроллера 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 NCR. Режим конфигурации позволяет изменять параметры синхронизации битов и состояния счетчиков ошибок.

Конфигурация прерываний задается битами:

- TRIE – управляет разрешением прерывания после передачи сообщения;
- ALIE – управляет разрешением прерываний по ошибке;
- LECIE – управляет разрешением прерывания по коду последней ошибки.

Регистр NSR 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 17.14. Квант времени – фиксированная единица времени, получаемая из частоты синхронизации и делителя контроллера 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.

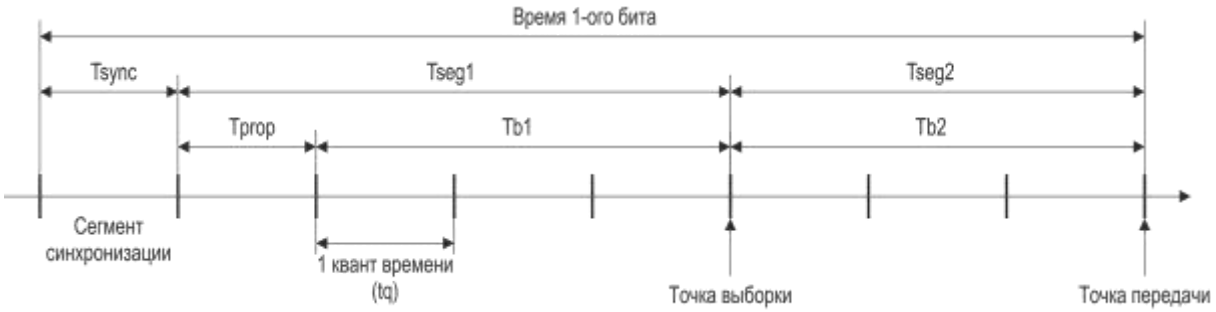


Рисунок 17.14 – Структура одного бита

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 NBTR (может быть записан, только если установлен бит CCE регистра NCR). Согласно стандарту ISO, минимальная длительность сегмента параметра 1 должна составлять три кванта времени.

Сегмент параметра 2 (Tseg2) определяется битовым полем TSEG2 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

$$tq = (BRP + 1) / fout; \quad (17.4)$$

- при DIV8 = 1 значение кванта времени

$$tq = 8 \times (BRP + 1) / fout; \quad (17.5)$$

$$Tsync = 1 \times tq; \quad (17.6)$$

$$Tseg1 = (TSEG1 + 1) \times tq \geq 3tq; \quad (17.7)$$

$$Tseg2 = (TSEG2 + 1) \times tq \geq 2tq; \quad (17.8)$$

$$Tbit = Tsync + Tseg1 + Tseg2 \geq 8tq. \quad (17.9)$$

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:

$$Tsjw = (SJW + 1) \times tq. \quad (17.10)$$

Помимо прочего, должны соблюдаться следующие правила:

$$Tseg1 \geq Tsjw + Tprop \text{ и } Tseg2 \geq Tsjw. \quad (17.11)$$

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

$$-\Delta fout \leq T/2 \times (13 \times Tbit - Tb2), \quad (17.12)$$

$$-\Delta fout \leq Tsjw / 20 \times Tbit, \quad (17.13)$$

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 NBTR (доступен, если установлен бит CCE) до окончания инициализации (до сброса бита INIT регистра NCR), т. е. до начала работы узла CAN.

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 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 NSR.

Корректность получаемых данных проверяется и подтверждается или не подтверждается кодом CRC. В случае отсутствия подтверждения возникает ошибка, генерируется запрос на прерывание и код ошибки выставляется в регистре NSR. 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 предназначен для выявления ошибок в работе устройств узла. В составе блока есть два счетчика: счетчик ошибок приема (поле REC в регистре NECNT) 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 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 NECNT, 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. В этом состоянии узел не может работать с шиной (выходные передатчики отключены). О том, что узел в состоянии отключения от шины сигнализирует флаг BOFF регистра NSR.

Флаг EWRN устанавливается, когда хотя бы один из счетчиков достиг или превысил лимит ошибок, определенный в битовом поле EWRNLVL регистра NECNT. Как только значения обоих счетчиков перестанут превышать лимит ошибок, флаг EWRN сбросится.

Счетчик сообщений

Счетчик сообщений может использоваться получения информации о завершении передачи/приема сообщения соответствующего узла. Подсчет сообщений осуществляется 16 разрядным счетчиком, который управляется регистром NFCR. 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 приема сообщения, содержимое счетчика копируется в битовое поле CFCVAL регистра MOIPR объекта сообщения, участвующего в пересылке данных. После чего счетчик сообщений инкрементируется.

Прерывания узла CAN

Узел может генерировать запросы на прерывания, см. рисунок 17.15, 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).

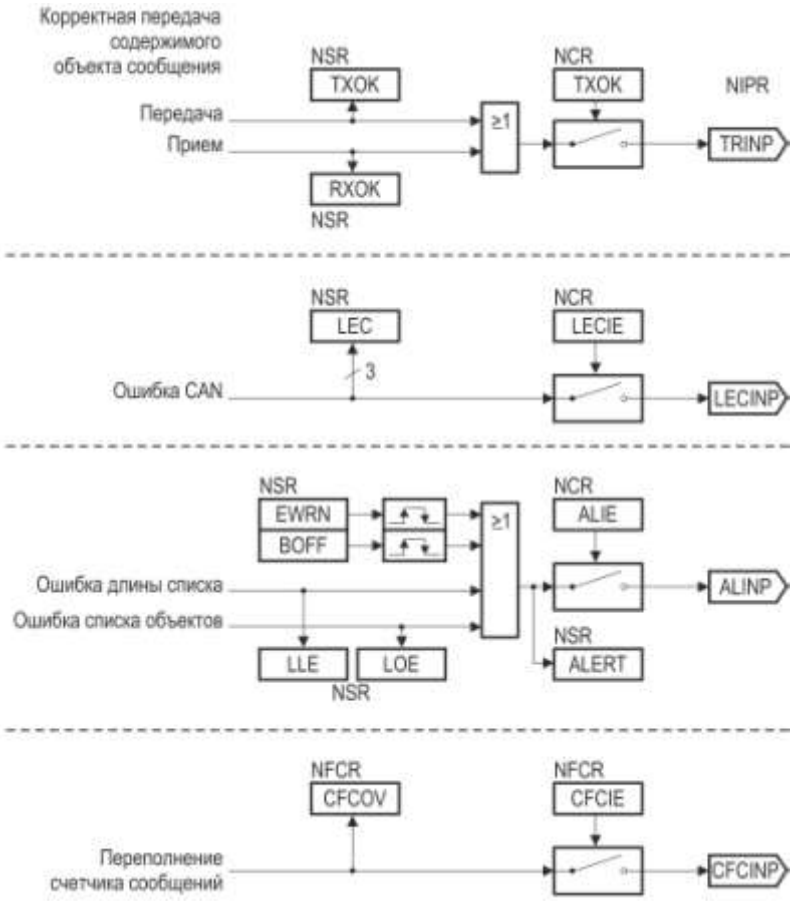


Рисунок 17.15 – Прерывания CAN узла

После каждой успешной передачи или успешного приема сообщения генерируется (если разрешено соответствующими битами TXOK и RXOK) прерывание. Битовое поле TRINP регистра NIPR 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 NFCR. Битовое поле CFCINP задает линию прерывания.

Прерывание ALERT 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 MITR. Запись единицы в i-й разряд поля IT генерирует сигнал запроса прерывания на соответствующей ему i-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

В состав каждого объекта сообщения входят девять регистров: MOAR, MODATAH, MODATAL, MOAMR, MOIPR, MOFGPR, MOFCR, а также расположенные по одному адресу MOCTRn (доступен при записи) и MOSTATn (доступен при чтении). Расположение регистров в памяти микроконтроллера показано на рисунке 17.16.

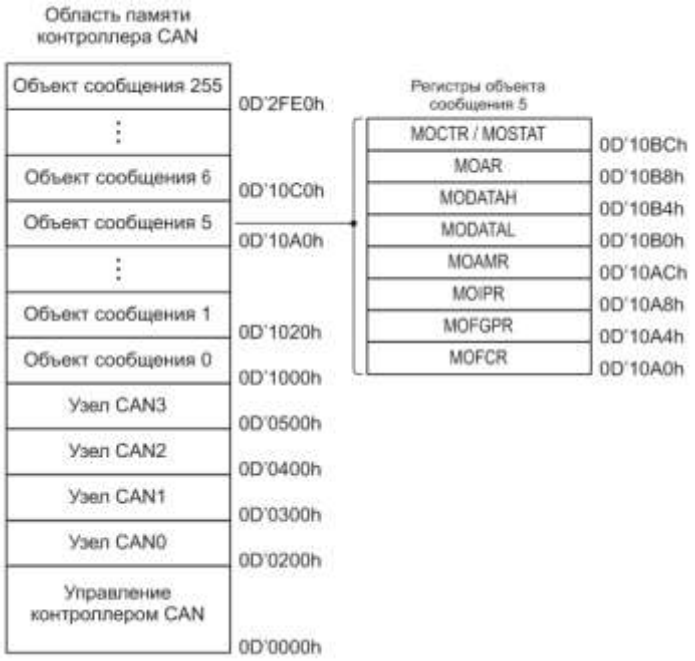


Рисунок 17.16 – Структура объектов сообщений

Объекты сообщений контроллера CAN могут быть организованы в восемь списков, см. рисунок 17.17.

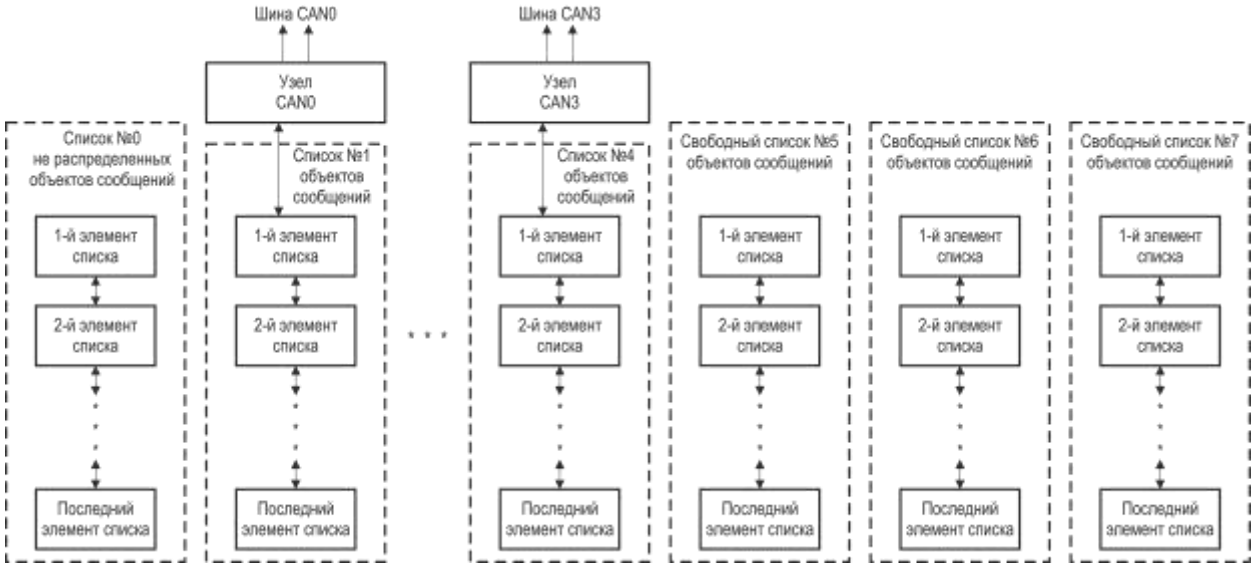


Рисунок 17.17 – Списки контроллера CAN

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, который относится к узлу CAN0, регистр LIST2 – списка №2 узла CAN1, регистр LIST3 – списка №3 узла CAN2 и регистр LIST4 – списка №4 узла CAN3 .

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

Положение объекта сообщения в списке определяется посредством регистра MOSTAT, 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 три списка с номерами 5, 6 и 7 являются свободными (не принадлежат ни одному узлу) и имеют соответствующие регистры LIST5 – LIST7. Объекты сообщений, распределенные в списки 5, 6 и 7, не могут использоваться узлами.

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

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 сообщения содержит номер списка (поле LIST), 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.

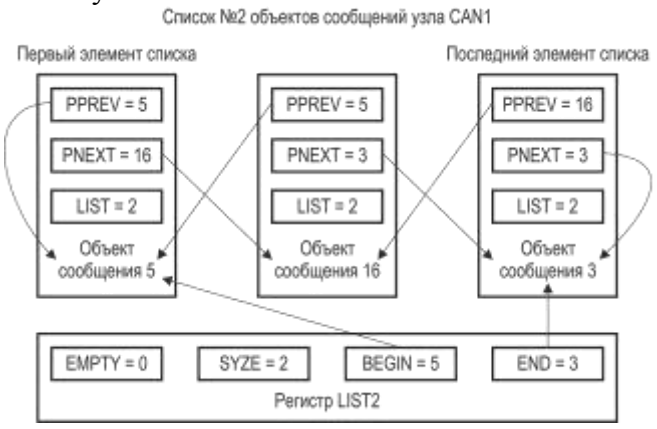


Рисунок 17.18 – Пример списка объектов сообщений

На рисунке 17.18 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 сообщения n – 1 является предыдущим объекта сообщения n, а объект сообщения n + 1 – следующим.

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 с помощью регистра PANCTR.

Панель команд запускается записью соответствующей команды в битовое поле PANCMD. До записи кода команды должны быть записаны соответствующие аргументы команды в 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 объектам сообщений запрещены. По окончании этой операции флаг BUSY сбрасывается, и объекты становятся доступными.

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 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 MOSTAT объекта сообщения должен быть очищен.

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

17.5 Прием и передача сообщений

Прием сообщения

После завершения приема сообщение сохраняется в объекте сообщения в соответствии с установленным алгоритмом, см. рисунок 17.19.

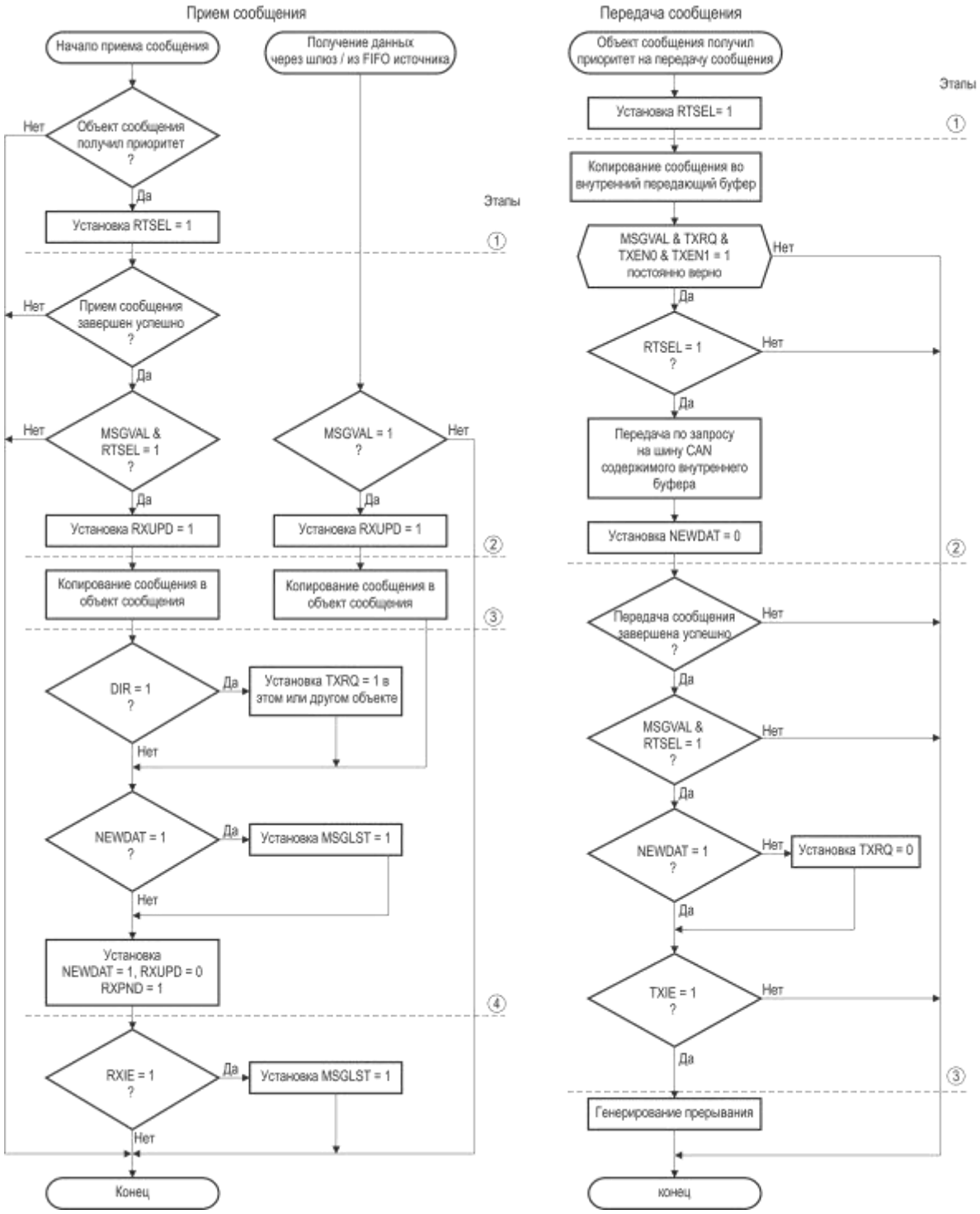


Рисунок 17.19 – Алгоритмы приема и передачи сообщения

Помимо сохранения данных в объекте сообщения, контроллер CAN осуществляет обмен данными с ЦП.

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

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 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.

Реконфигурация объекта сообщения центральным процессором во время работы контроллера 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.

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.

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.

Передача сообщения

Алгоритм передачи сообщений показан на рисунке 17.19. 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.

Сообщение может быть передано только в случае, когда все четыре бита MSGVAL, TXEN0, TXEN1 и TXRQ установлены.

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.

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.

17.6 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.

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 MOAR) 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, то 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 MOAR объекта сообщения, за исключением битов, закрытых маской регистра MOAMR, значение которых не важно. На рисунке 17.20 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 MOAR. 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.

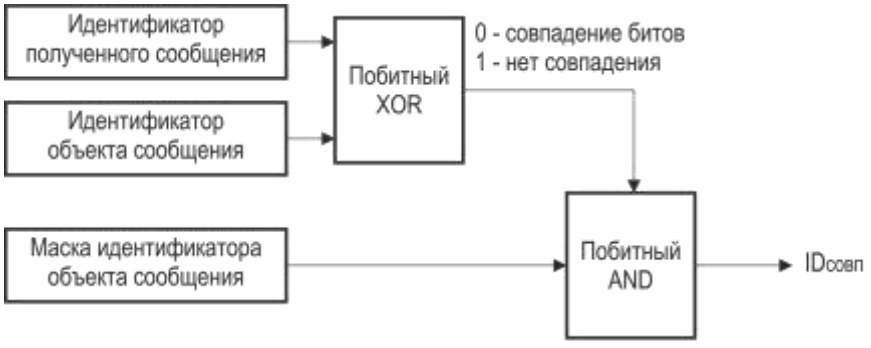


Рисунок 17.20 – Проверка идентификатора полученного сообщения

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 узла CAN;
- 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 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 17.21 показано формирование запроса на передачу объекта сообщения.

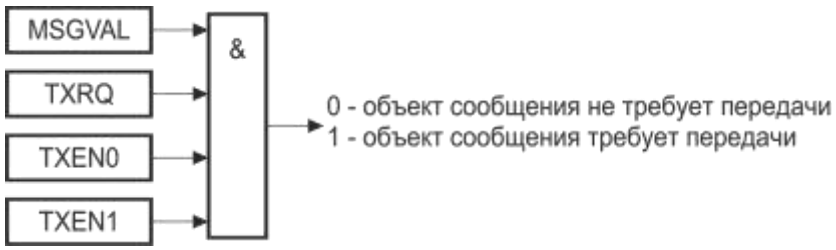


Рисунок 17.21 – Формирование запроса на передачу объекта сообщения

17.7 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.

В зависимости от состояния бита FRREN объекта сообщения, который принял сообщение удаленного запроса, возможны два варианта действий:

– если бит FRREN сброшен, то устанавливается флаг TXRQ этого объекта;

– если бит FRREN установлен, то устанавливается флаг TXRQ того объекта, на который указывает поле CUR объекта, принявшего удаленный запрос. При этом поле CUR не меняет своего значения.

Состояние регистров объекта сообщения, передающего сообщение удаленного запроса

У объекта сообщения, передающего сообщение удаленного запроса, в регистре MOSTAT должен быть сброшен бит DIR (объект передает сообщение данных) и установлены биты TXEN0, TXEN1, MSGVAL и TXRQ. Значение идентификатора в регистре MOAR передающего объекта сообщения, должно быть равно значению идентификатора принимающего объекта сообщения (или совместно с регистром MOAMR обеспечивать успешное прохождение фильтрации), чтобы сообщение удаленного запроса было принято принимающим объектом другого узла. Само сообщение удаленного запроса должно содержать идентификатор принимающего объекта сообщения, поэтому значение регистра MODATAL передающего объекта сообщения должно быть равно значению регистра MOAR принимающего объекта.

Состояние регистров объекта сообщения, принимающего сообщение удаленного запроса при FRREN = 0

У объекта сообщения, принимающего сообщение удаленного запроса, должны быть установлены биты DIR (объект принимает сообщение удаленного запроса), TXEN0 и TXEN1 (если отвечать на запрос будет сам), RXEN и MSGVAL. Регистры MODATAL и MODATAH должны содержать данные, которые будут переданы в ответ на запрос.

Состояние регистров объектов сообщений, принимающего сообщение удаленного запроса (при FRREN = 1) и содержащего данные для ответа на запрос

У объекта сообщения, принимающего сообщение удаленного запроса, должны быть установлены биты DIR, RXEN и MSGVAL. Битовое поле CUR должно указывать на номер объекта сообщения (должен находиться в том же узле, что и объект принявший сообщение удаленного запроса), содержащего данные, предназначенные для передачи в ответ на поступивший удаленный запрос.

В свою очередь у объекта сообщения, хранящего данные для отправки в ответ на запрос, должны быть установлены биты DIR, (объект передает сообщение данных), TXEN0, TXEN1 и MSGVAL. Бит TXRQ 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).

17.8 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.

Режим передачи данных с защитой от повторения

Выбирается установкой бита SDT регистра MOFCR.

После приема сообщения данных и сохранения его в объекте с установленным битом SDT, бит MSGVAL этого объекта аппаратно сбрасывается, чтобы исключить возможность повторного приема и записи в этот объект. Этот режим нельзя использовать для базового объекта FIFO структуры.

В ответ на сообщение удаленного запроса, принятое объектом с установленными битом SDT будут отправлены данные из объекта сообщения, на который указывает поле CUR объекта, принявшего удаленный запрос. После этого бит MSGVAL объекта принявшего сообщение удаленного запроса сбросится.

Примечание – Объект, принявший сообщение удаленного запроса, не может быть источником данных, передаваемых в ответ на запрос. Это означает, что в данном режиме бит FRREN объекта, принявшего удаленный запрос, обязательно должен быть установлен.

Режим однократной пересылки данных

Выбирается установкой бита STT.

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.

17.9 FIFO структура объектов сообщений

Регистр MOFGPR объекта сообщения содержит установки указателей на объекты сообщений, которые используются при операциях FIFO и шлюзовых операциях.

В случае 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 17.22 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) можно присоединять базовый объект к вспомогательным объектам, независимо от того, принадлежат базовый и вспомогательный объекты одному списку или разным спискам.

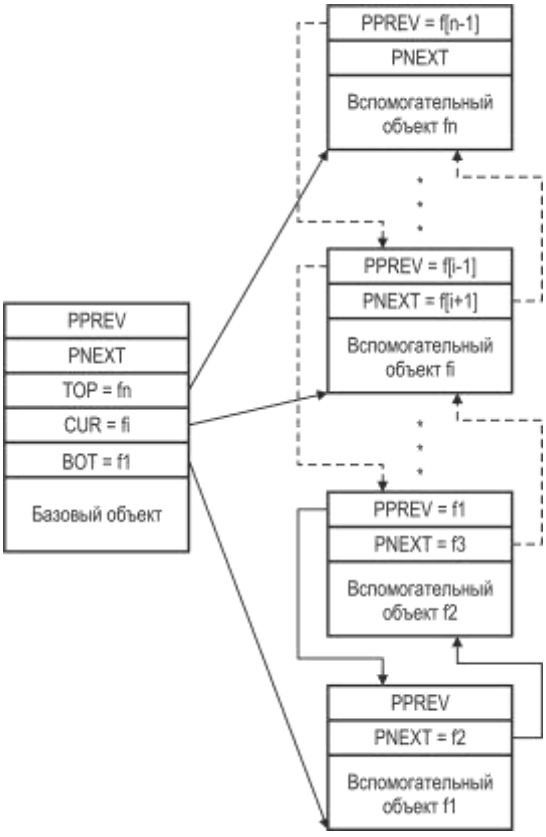


Рисунок 17.22 – FIFO структура с базовым объектом и n вспомогательными объектами

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 256 объектов 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.

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

Используется для буферизации входящих сообщений данных и удаленных запросов.

FIFO структура для приема активируется записью значения 0001b в битовое поле MMC регистра MOFCR 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 MOFCR 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 неважно в случае, когда вспомогательный объект занесен в список, не связанный с узлом CAN.

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 MOFCR 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 MOFCR 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.

Программирование 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 MOAR и MOAMR 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 MOAR передающих вспомогательных объектов должно совпадать (или совместно с регистрами MOAMR обеспечивать успешное прохождение фильтрации) 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 MOAR принимающего базового объекта должно быть равно значению регистров MOAR передающих вспомогательных объектов передачи (или совместно с регистром MOAMR обеспечивать успешное прохождение фильтрации). 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 MOAR 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.

17.10 Режим шлюза

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.

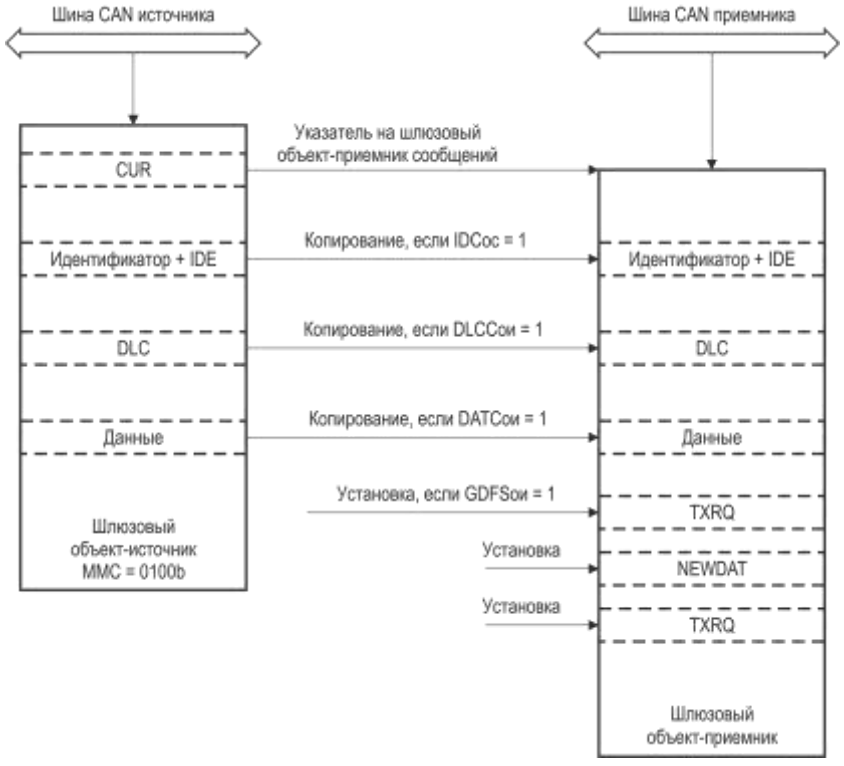


Рисунок 17.23 – Передача через шлюз от источника к приемнику

Режим шлюза активируется записью значения 0100b в битовое поле MMC регистра MOFCR объекта сообщения 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.

Шлюзовый объект-источник, см. рисунок 17.23, функционирует как обычный объект сообщения с тем отличием, что возможны дополнительные действия контроллера CAN при приеме и сохранении сообщения в объекте-приемнике:

1) Если установлен флаг DLCC регистра MOFCR 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 MODATAL и MODATAH 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 MOSTAT объекта-приемника.

6) Если установлен флаг RXIE регистра MOSTAT 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. 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.

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 сообщения 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 MODATAL и MODATAH объекта-источника копируются в MODATAL и MODATAH объекта-приемника (установлен бит DATC регистра MOFCR объекта-источника), вследствие чего автоматически устанавливается бит TXRQ регистра MOCTR 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.

17.11 Прерывания объектов сообщений

После сохранения принятого сообщения в объект сообщения или успешной передачи формируется соответствующее прерывание, которое 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.

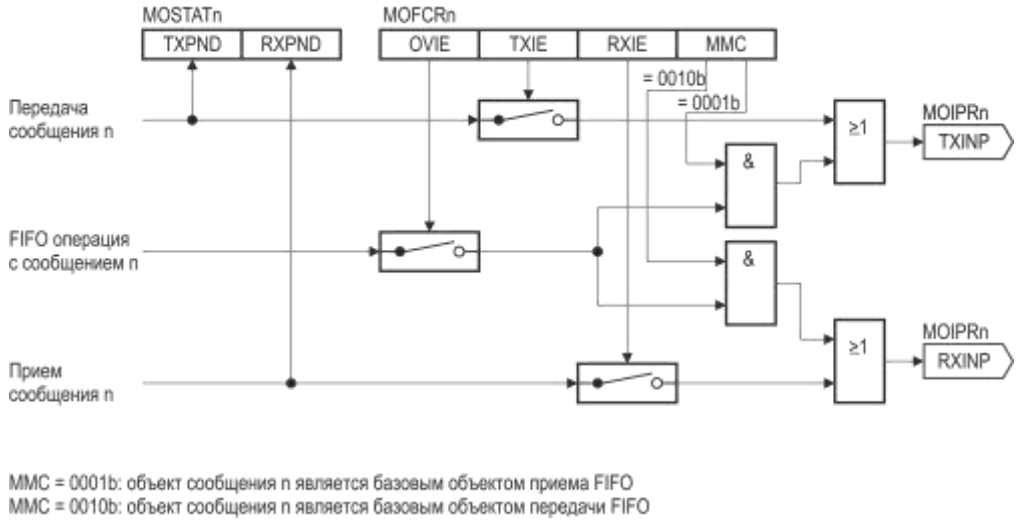


Рисунок 17.24 – Распределение прерываний

Объект сообщения может формировать FIFO прерывания. Если флаг OVIE регистра MOFCR установлен, то формирование FIFO прерывания будет зависеть от типа объекта сообщений (см. рисунок 17.24):

- 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 восьми регистров ждущих прерываний MSPNDx (x от 0 до 7) выставляется флаг ждущего сообщения. Восемь регистров образуют область из 32 × 8 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, см. рисунки 17.25 и 17.25.

В зависимости от значения поля MPSEL регистра MCR, реализуется один из двух режимов выбора и установки флагов, ждущих сообщения:

- 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:

- 7, 6 и 5 биты поля MPN выбирают регистр MSPNDx, в котором будет установлен флаг ждущего сообщения;

- пять младших бит поля MPN (на рисунке 17.25 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:

- 3, 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 17.26 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.

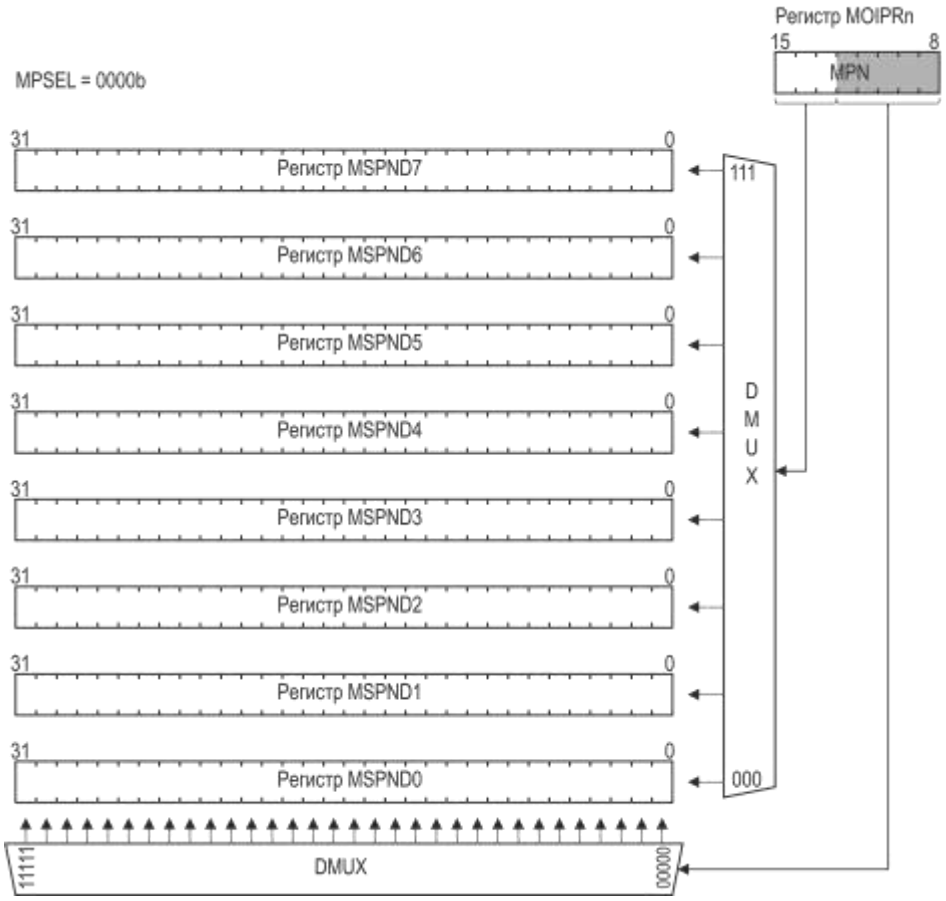


Рисунок 17.25 – Режим выбора и установки флагов при MPSEL = 0h

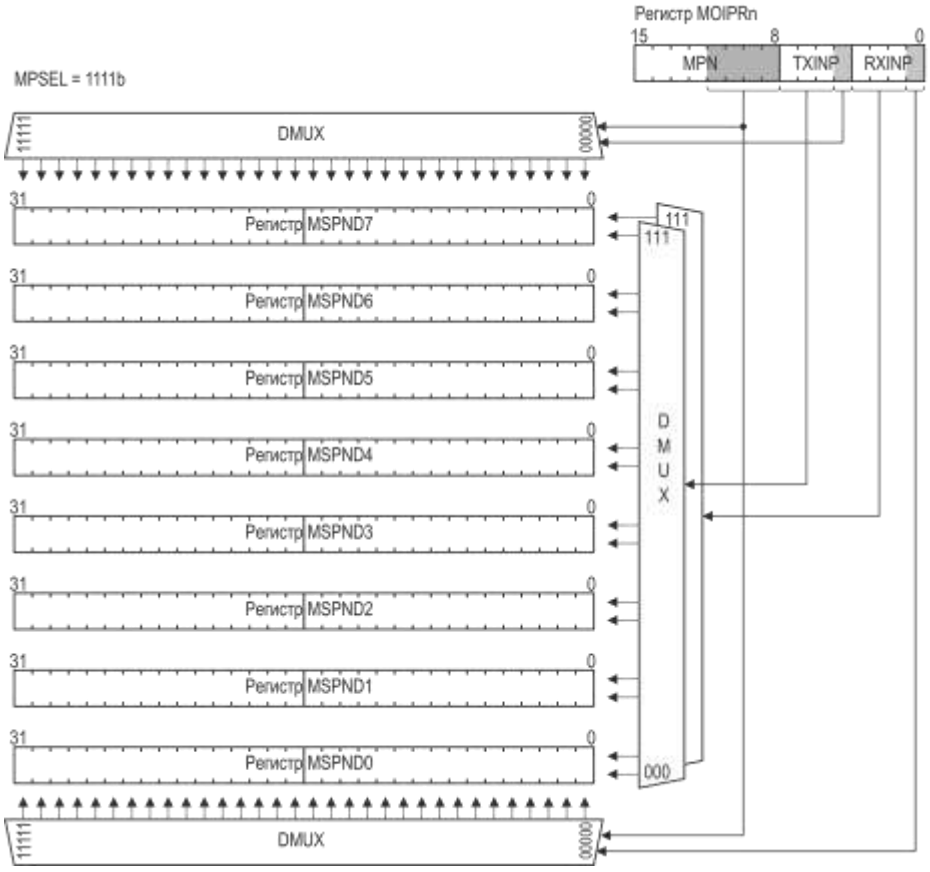


Рисунок 17.26 – Режим выбора и установки флагов при MPSEL = Fh

17.12 Программирование контроллера CAN

Для корректной работы контроллера CAN следует соблюдать порядок программирования регистров.

Для запуска контроллера:

- записать регистр CLC;
- проверить, что сброшен бит DISR и регистр PANCTR = 00000000h и после этого записать регистр FDR.

Далее для конфигурирования узла CAN с номером x (x от 0 до 3) выполнить:

- в регистре узла 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;
- распределить объекты сообщений в списки посредством регистра PANCTR.

Для корректной работы объектов сообщений регистры каждого из них должны быть проинициализированы. Для объектов, использование которых не предусматривается достаточно записать ноль в бит MSGVAL регистра MOCTR.

Рекомендуемый порядок инициализации регистров объекта сообщения:

- установить бит DIR в регистре MOSTAT 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 MOCTR;
- записать регистр MOFCR;
- записать регистр MOAR;
- записать регистр MOAMR (если необходимо);
- записать регистр MOFGPR (если будут использоваться FIFO структуры);
- записать регистр MOIPR;
- записать регистры MODATAL и MODATAH;
- установить бит MSGVAL корректности объекта сообщения в регистре MOCTR (для неиспользуемых объектов этот бит должен быть сброшен);
- для активирования передачи установить бит TXRQ регистра MOCTR.

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

Модуль CAPCOM1	Модуль CAPCOM2
Регистры управления	
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

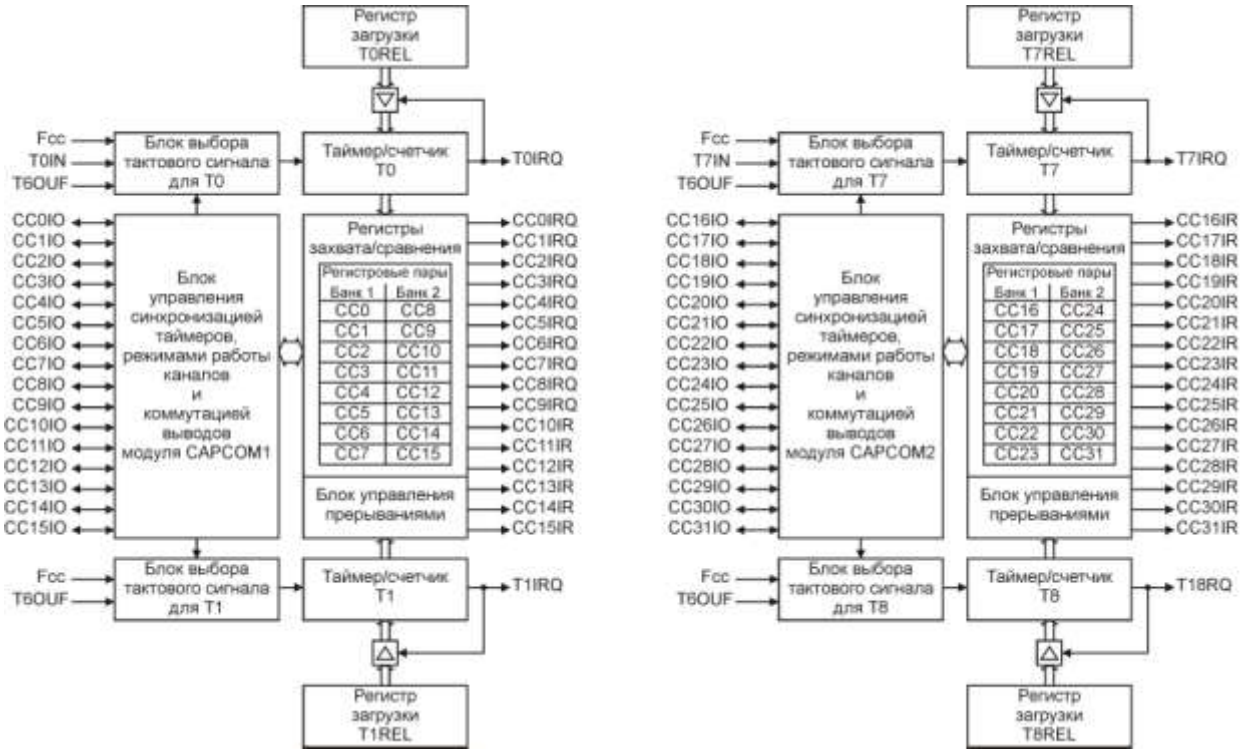


Рисунок 18.2 – Структурные схемы модулей CAPCOM1 и CAPCOM2

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.

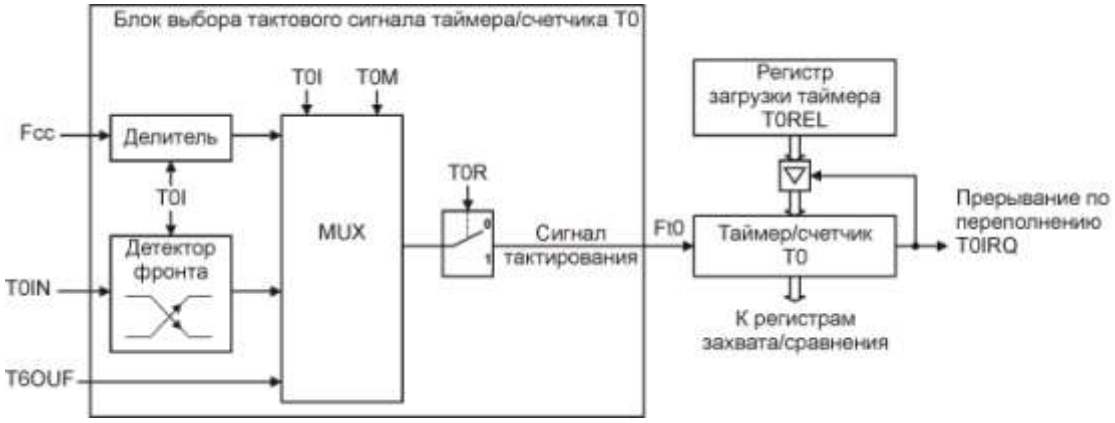


Рисунок 18.3 – Таймер/счетчик T0 и его схема тактирования

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:

$$Pt0 = \frac{(2^{16} - \langle T0REL \rangle) \times 2^{(\langle T0I \rangle + 3)}}{fcc} \qquad (18.1)$$

- для нормального режима:

$$Pt0 = \frac{(2^{16} - \langle T0REL \rangle) \times 2^{\langle T0I \rangle}}{fcc} \qquad (18.2)$$

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.

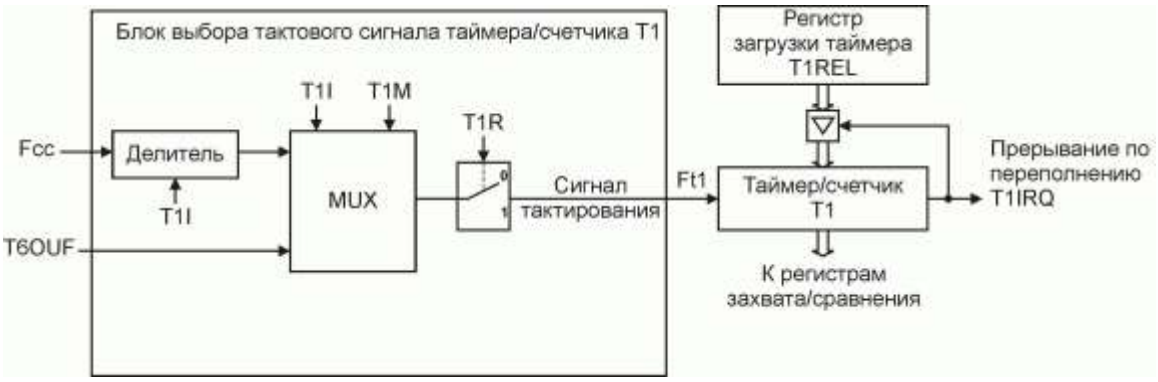


Рисунок 18.4 – Таймер/счетчик T0 и его схема тактирования

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.

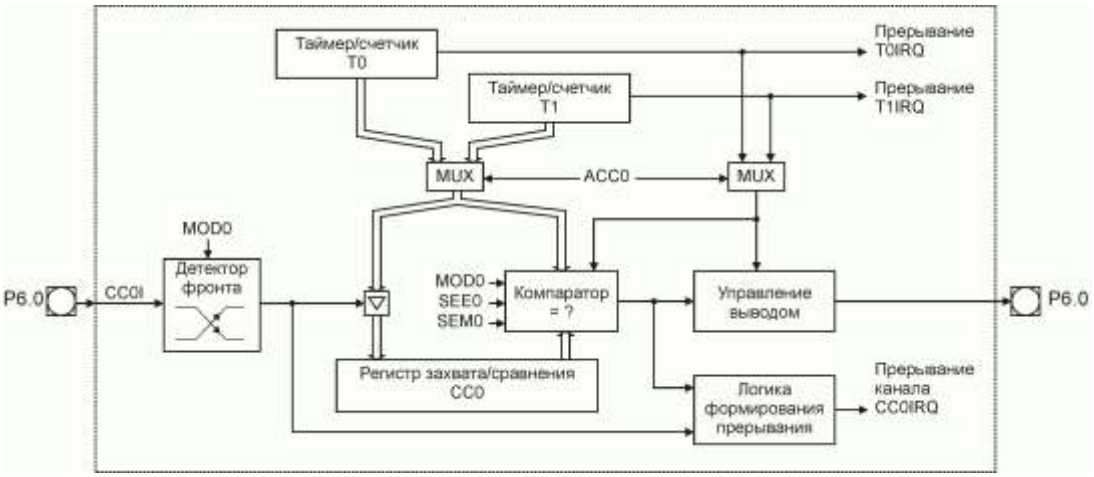


Рисунок 18.5 – Функциональная схема нулевого канала

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.

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.

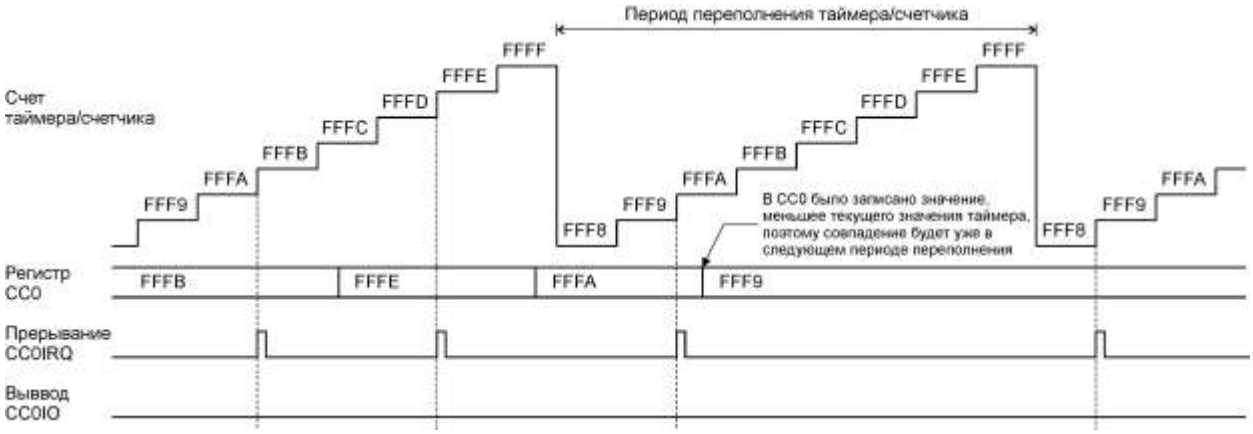


Рисунок 18.6 – Режим сравнения 0, значение регистра загрузки – FFF8h

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.

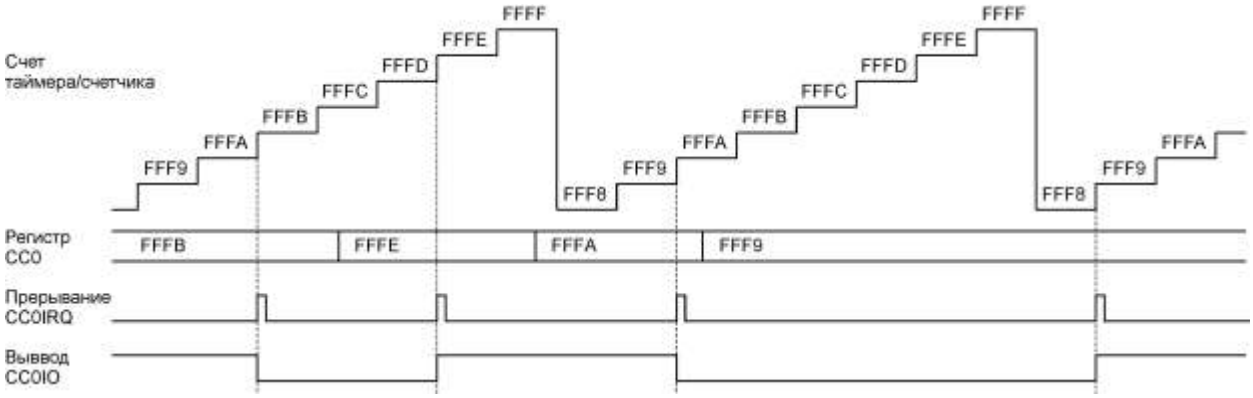


Рисунок 18.7 – Режим сравнения 1, значение регистра загрузки – FFF8h

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.

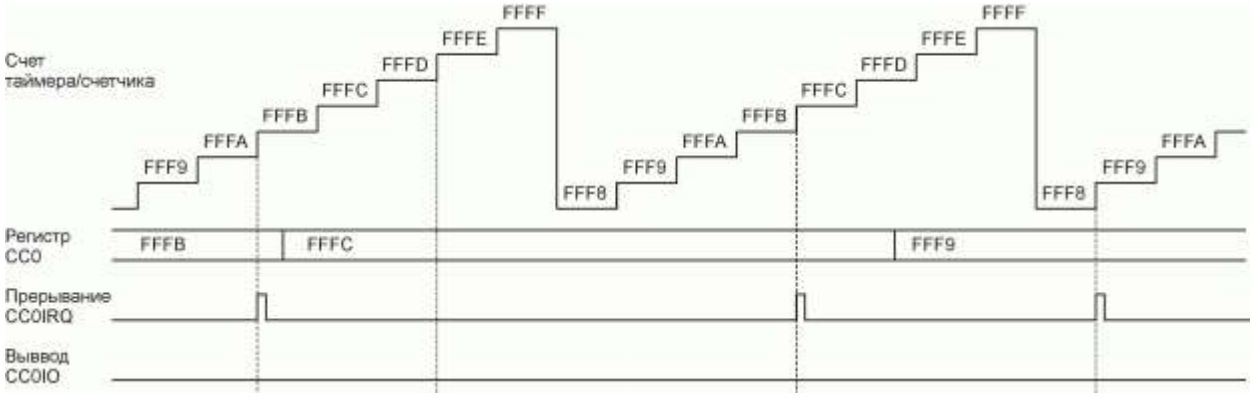


Рисунок 18.8 – Режим сравнения 2, значение регистра загрузки – FFF8h

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.

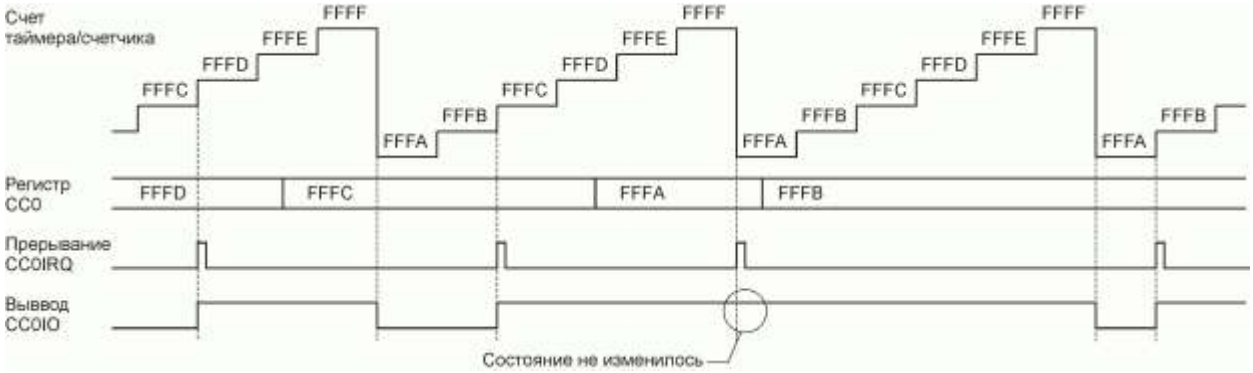


Рисунок 18.9 – Режим сравнения 3, значение регистра загрузки – FFFAh

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 – Банк 1 и Банк 2. Два регистра – один из Банка 1, 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 CAPCOM1				Блок CAPCOM2			
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.

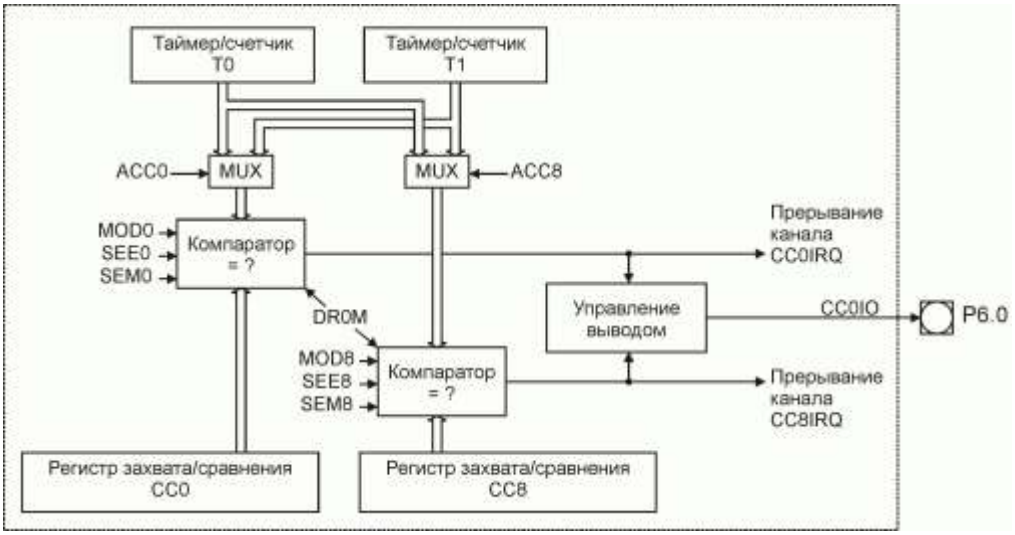


Рисунок 18.10 – Объединение каналов для двухрегистрового режима

Описание функционирования регистров в двухрегистровом режиме здесь приводится на примере регистров CC0 и CC8 модуля CAPCOM1, 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.

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.

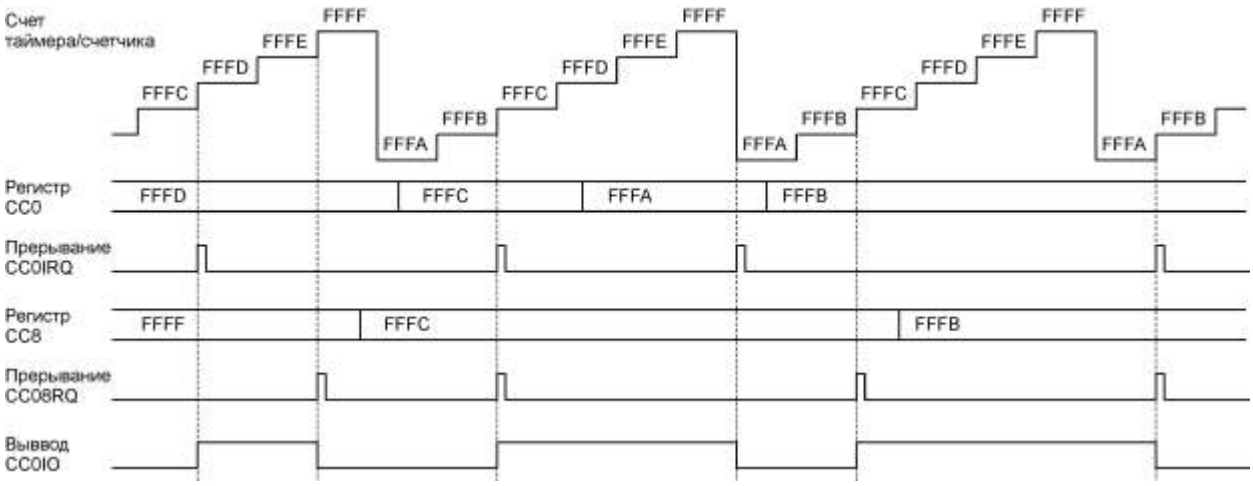


Рисунок 18.11 – Функционирование регистров CC0 и CC8 в двухрегистровом режиме сравнения

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.

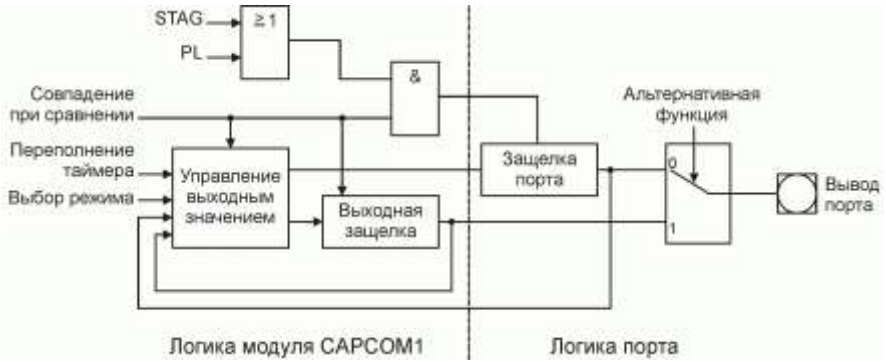


Рисунок 18.12 – Функциональная схема взаимодействия выходной логики канала модуля и логики соответствующего вывода порта

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

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.

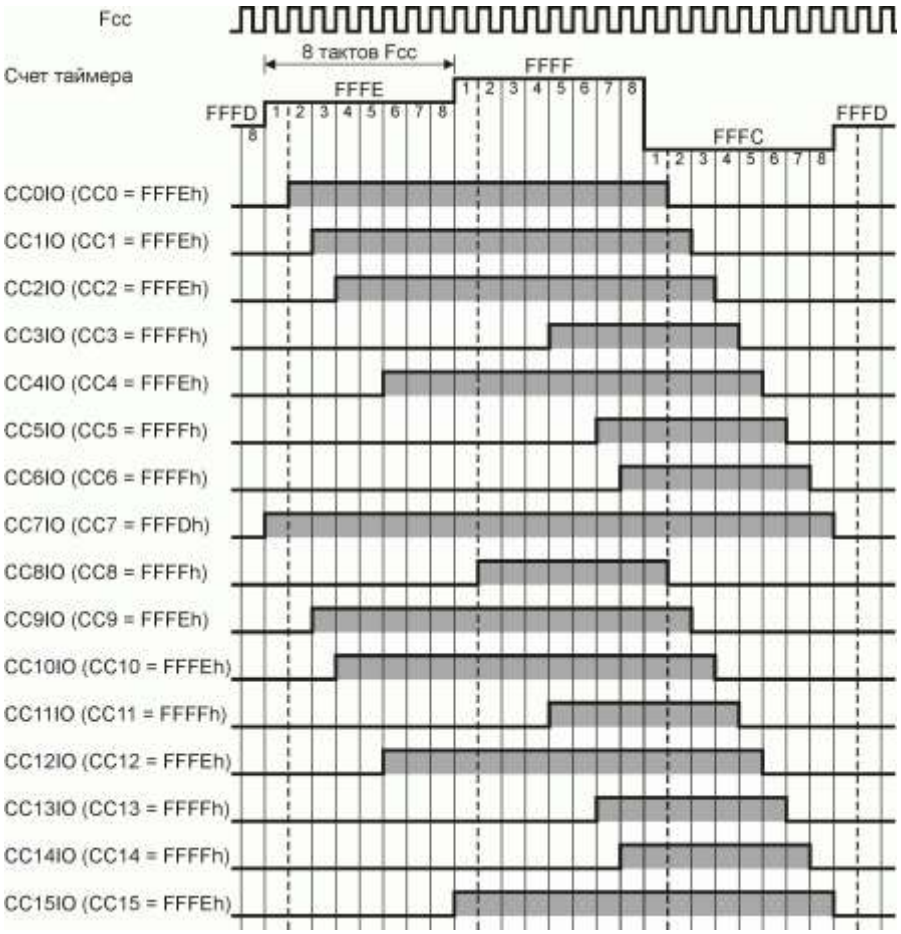


Рисунок 18.13 – Ступенчатый режим. Частота работы таймера fcc/8

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.

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.

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.

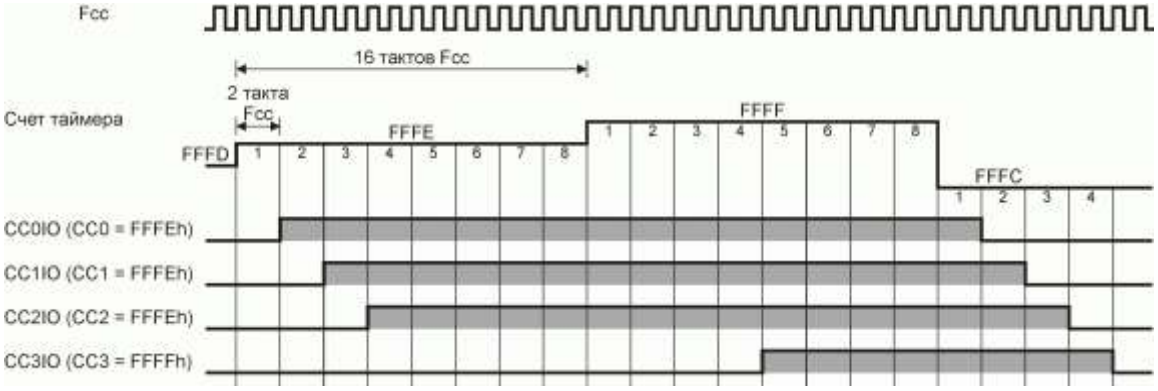


Рисунок 18.14 – Ступенчатый режим. Частота работы таймера fcc/16

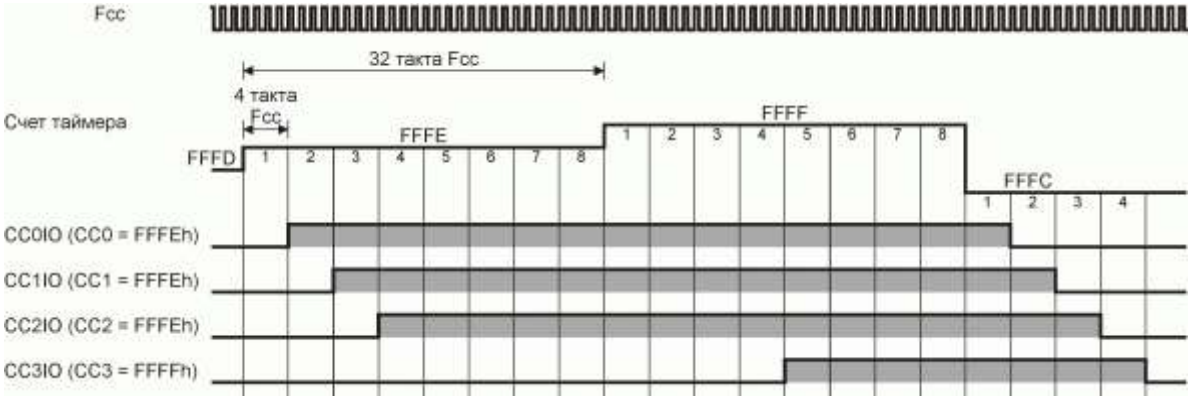


Рисунок 18.15 – Ступенчатый режим. Частота работы таймера fcc/32

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.

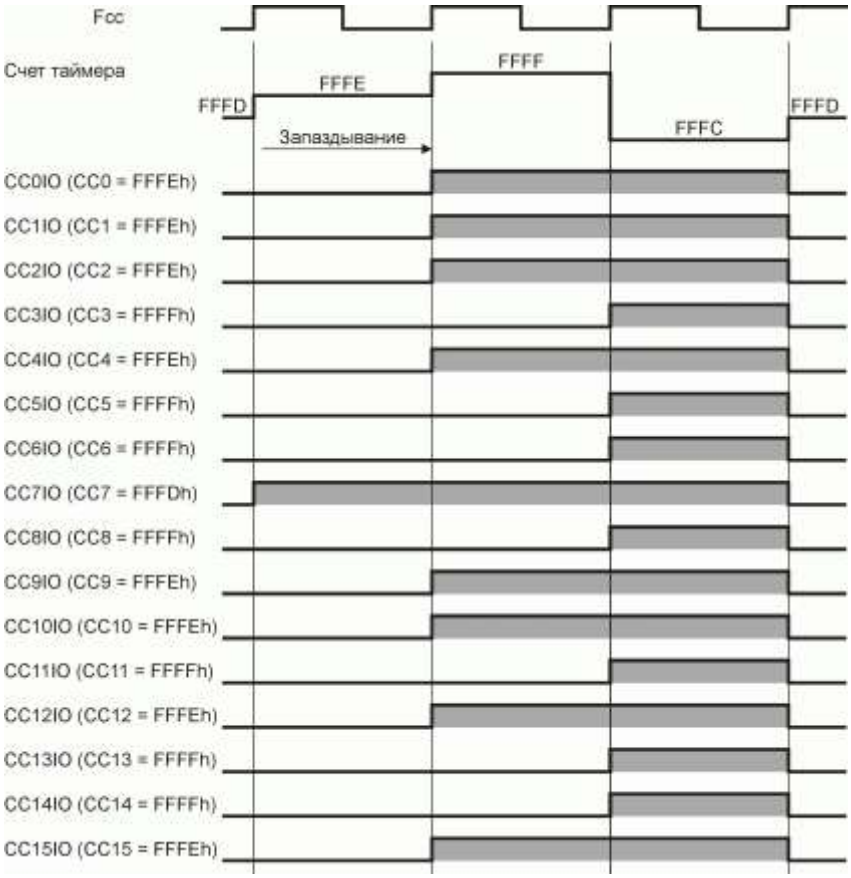


Рисунок 18.16 – Нормальный режим. Частота работы таймера/счетчика fcc

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.

На рисунке 18.17 показан пример для случая, когда таймер работает на частоте меньшей, чем fcc (в данном примере fcc/8).

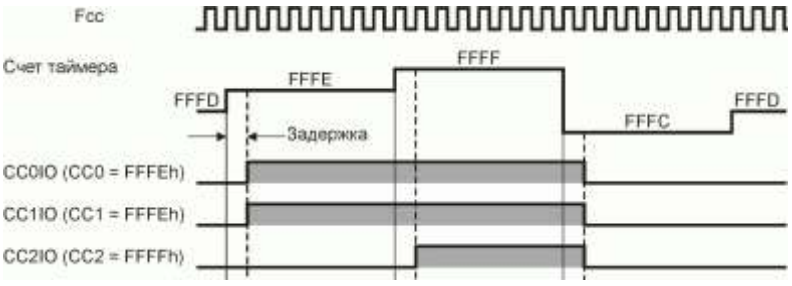


Рисунок 18.17 – Нормальный режим. Частота работы таймера/счетчика fcc/8

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 .

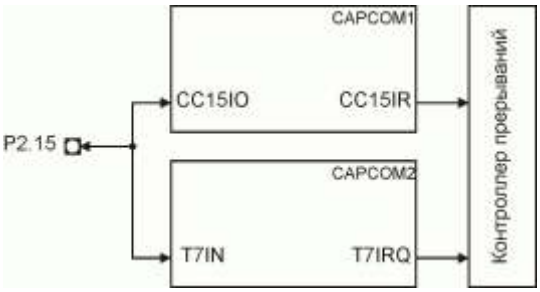


Рисунок 18.18 – Общий вывод модулей захвата/сравнения

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 CAPCOM6

Модуль захвата/сравнения CAPCOM6 (далее модуль CAPCOM6) предназначен для отслеживания заданных внешних (по отношению к микроконтроллеру) и внутренних событий, а также для формирования программируемых последовательностей импульсов (в том числе и ШИМ). Специальные режимы работы позволяют также осуществлять управление бесщеточными ДПТ (с датчиками Холла или контролем обратной ЭДС).

Интерфейс выводов модуля CAPCOM6 и выводов микроконтроллера показан на рисунке 19.1. Структурная схема модуля представлена на рисунке 19.2.

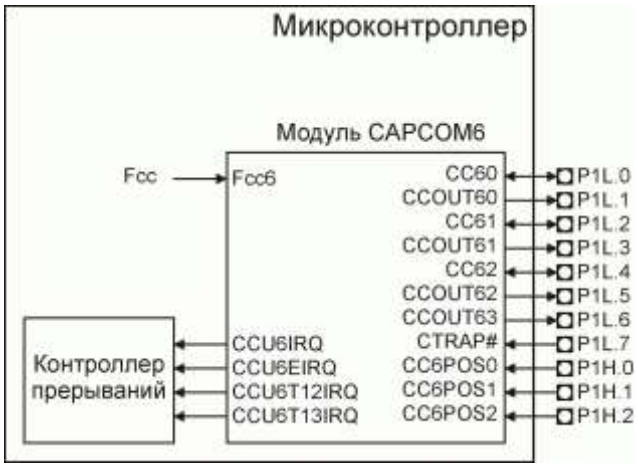


Рисунок 19.1 – Интерфейс модуля CAPCOM6

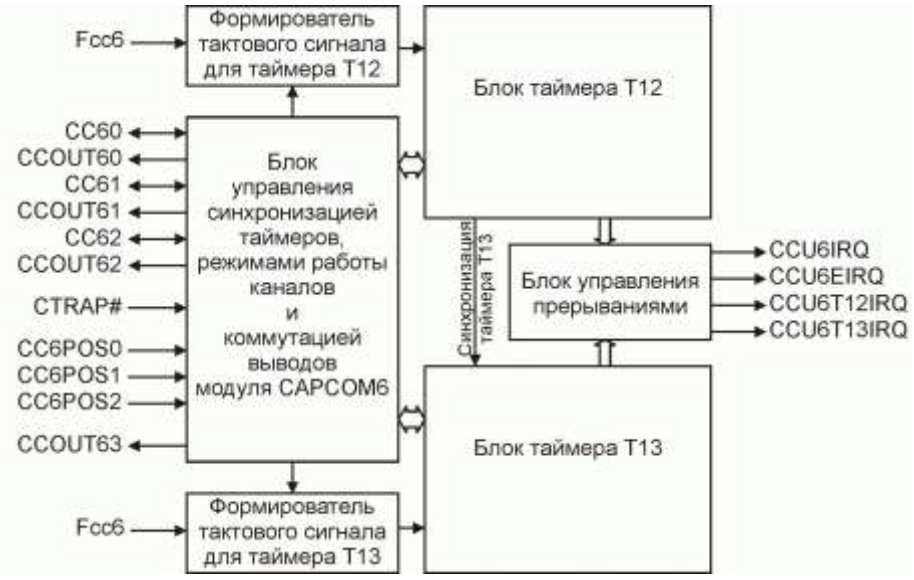


Рисунок 19.2 – Структурная схема модуля CAPCOM6

Модуль CAPCOM6 состоит из блока 16-разрядного таймера T12 и блока 16-разрядного таймера T13, двух формирователей тактовых сигналов, блока управления синхронизацией таймеров, режимами работы каналов и коммутацией выводов и блока управления прерываниями.

Таймер T12 работает с тремя каналами захвата/сравнения, каждый из которых имеет два вывода (один выход, другой вход/выход), что позволяет формировать трехфазный ШИМ-сигнал и аппаратный механизм, позволяющий вводить временную задержку (так называемое время «dead-time») в генерируемый сигнал (например, для предотвращения

короткого замыкания в силовой части, если контроллер работает с двигателем). Максимальная частота переключений счетчика таймера равна внешней тактовой частоте сигнала тактирования Fcc. Таймер T12 может также функционировать в режиме однотактного запуска, а каждый из каналов – в режиме блокирования выходного сигнала.

Таймер T13 работает с одним каналом сравнения, который имеет один выход. Максимальная частота переключений счетчика таймера равна внешней тактовой частоте сигнала тактирования Fcc. Таймер T13 может синхронизироваться с таймером T12. Поддерживается режим однократного запуска.

Сигналы, модулируемые в разных каналах, могут быть смешаны при соответствующем программировании блока коммутации выводов.

Особенности модуля CAPCOM6 позволяют осуществлять коммутацию с бесщеточным ДПТ, отслеживать положение ротора двигателя и фильтровать шумы при работе с датчиками Холла, автоматически измерять скорость вращения. Поддерживается возможность скоростного экстренного прекращения работы без задействования ЦПУ под управлением внешнего сигнала CTRAP#.

19.1 Блок таймера T12

Блок является основным генератором 3-фазной ШИМ. В состав блока входят три канала (каждый со своими входом и выходами), которые взаимодействуют с 16-разрядным таймером. Каждый канал имеет регистр, который через компаратор связан с таймером. При совпадении значений таймера и регистра, на выходе соответствующего канала формируется сигнал.

Таймер T12 представляет собой счетчик (регистр CCU6_T12), который может инкрементироваться и декрементироваться. Тактовый сигнал таймера формируется на основе внешнего тактового сигнала Fcc6. Через компаратор таймер связан с регистром периода CCU6_T12PR (хранит значение, по достижении которого таймер сбрасывается или изменяет направление счета). На рисунке 19.3 показана функциональная схема взаимодействия таймера и регистров периода.

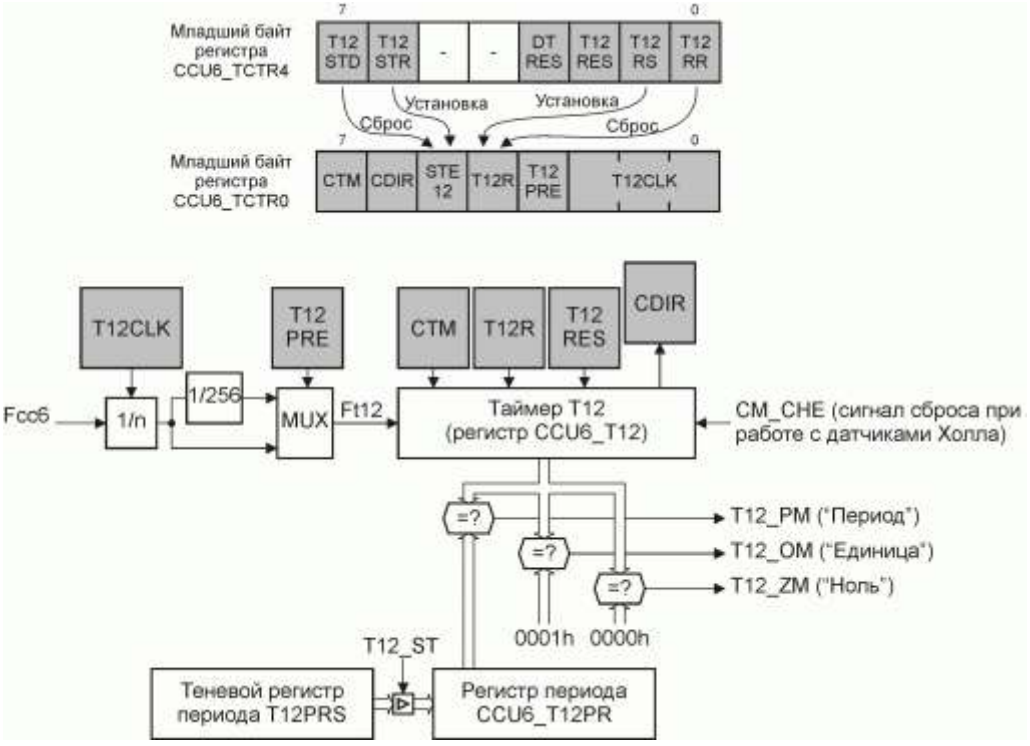


Рисунок 19.3 – Таймер T12 и регистры управления

Примечание – На рисунке 19.3 и далее на рисунках прямоугольники серого цвета указывают на то, что источником сигнала является бит или битовое поле, название которого указано внутри прямоугольника.

Регистр T12PRS выполняет роль буфера для хранения значения, которое должно быть записано в регистр CCU6_T12PR по внутреннему сигналу T12_ST, который формируется в случае программного или аппаратного запроса. Регистр CCU6_T12PR является основным, а регистр T12PRS – теневым. Запись данных из теневого регистра в основной является теневой загрузкой. Если таймер включен, теневая загрузка выполняется только в момент переключения таймера. Это позволяет синхронизировать работу таймера и регистра.

Дополнительные два компаратора позволяют отслеживать моменты, когда значение таймера равно единице или нулю.

Совпадение значения таймера и его регистра периода является событием «Период».

Достижение таймером значения единицы является событием «Единица», достижение нуля – событием «Ноль».

Работой таймера T12 управляют два регистра CCU6_TCTR0 и CCU6_TCTR4.

Пока таймер не запущен, его регистр CCU6_T12 доступен для записи. По умолчанию, начальное состояние таймера 0000h. Запись в регистр таймера во время работы игнорируется. В то же время для чтения таймер доступен всегда.

Для запуска таймера T12 нужно установить бит T12RS в регистре CCU6_TCTR4. После этого аппаратно установится бит T12R в регистре CCU6_TCTR0 и таймер запуститься, см. рисунок 19.3.

В зависимости от состояния бита CTM регистра CCU6_TCTR0 таймер будет работать как однонаправленный или двунаправленный счетчик.

Режим однонаправленного счетчика

После запуска, таймер начинает считать вверх с частотой сигнала Ft12. Каждый раз при инкрементировании значение таймера сравнивается со значением регистра периода CC6U_T12PR, и как только возникает совпадение (событие «Период»), генерируется сигнал T12_PM, и аппаратно устанавливается бит STE12. Со следующим тактом сигнала тактирования Ft12 таймер сбрасывается в ноль, а в регистр периода CC6U_T12PR осуществляется аппаратная теневая загрузка значения из теневого регистра T2PRS. Одновременно с этим возникает событие «Ноль», и генерируется сигнал T12_ZM, а бит STE12 сбрасывается. Таймер продолжает счет. Время работы таймера от сброса до сброса является периодом таймера. Период таймера может оставаться постоянным или изменяться. Для изменения периода следует записать новое значение в регистр периода таймера. Фактически, это значение будет записано в теневой регистр T12PRS и будет сохраняться там до теневой загрузки.

Если новое значение должно вступить в силу по окончании текущего периода таймера, то никаких дальнейших действий не требуется. После события «Период», новое значение будет аппаратно записано в регистр периода одновременно со сбросом таймера.

Если требуется изменить состояние регистра периода, не дожидаясь окончания текущего периода, то надо сгенерировать программную теневую загрузку. Для этого после записи нового значения в регистр периода CC6U_T12PR нужно установить бит T12STR в регистре CCU6_TCTR4. После этого аппаратно будет установлен бит STE12, и логика блока будет ожидать инкрементирование таймера, чтобы выполнить теневую загрузку. Одновременно с переключением таймера новое значение будет записано в регистр CC6U_T12PR, а биты T12STR и STE12 – сброшены.

Рассмотрим работу таймера T12 на примере, представленном на рисунке 19.4.

Таймер T12 считает с частотой сигнала Ft12. В момент времени «1» возникает событие «Период», и устанавливается бит STE12. В следующий такт сигнала Ft12 – момент времени «2» – таймер обнуляется и осуществляется аппаратная теневая загрузка,

но поскольку значения регистров CC6U_T12PR и T12PRS совпадают, состояние регистра периода не меняется.

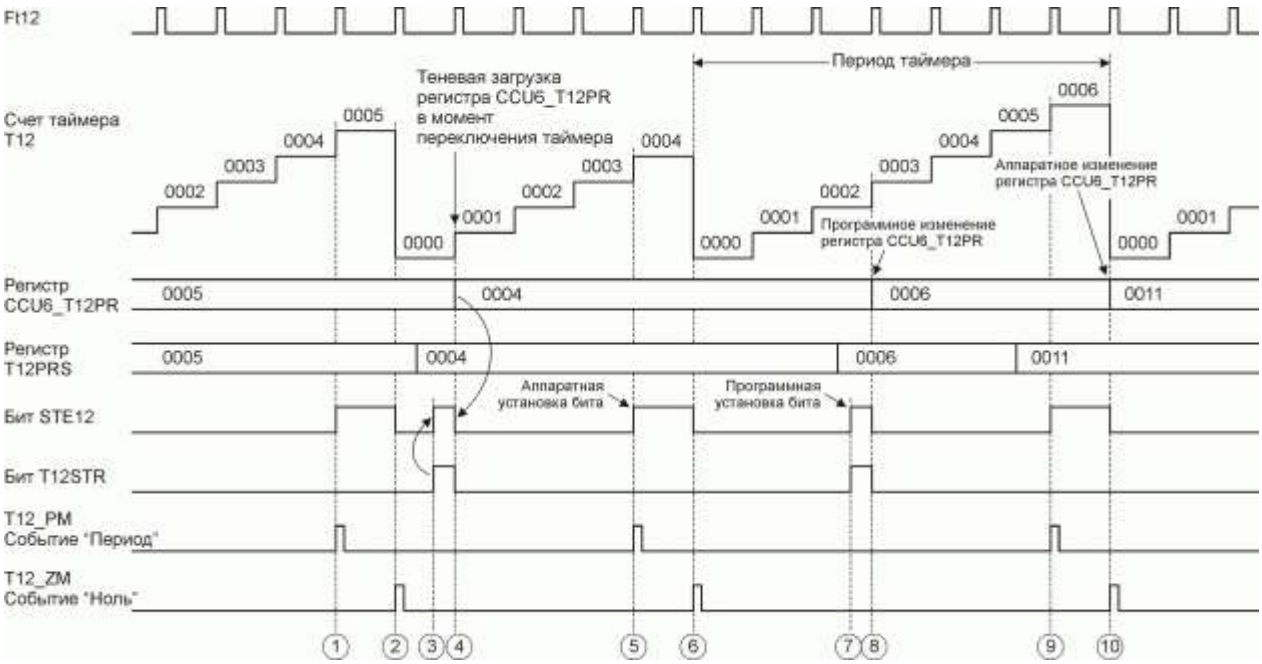


Рисунок 19.4 – Управляющие сигналы блока таймера T12 в режиме однонаправленного счетчика

Допустим, что возникла необходимость изменения длительности периода таймера, причем новые параметры должны вступить в силу в текущем периоде. Для этого следует записать новое значение в регистр CC6U_T12PR. На рисунке 19.4 это изменение содержимого регистра T12PRS с 0005h на h0004h между моментами времени «2» и «3». Далее, установка бита T12STR в момент времени «3» приведет к установке бита STE12. После установки бита STE12 логика блока будет ожидать переключение таймера. В момент инкрементирования таймера происходит запрограммированная теневая загрузка регистра CC6U_T12PR значением 0004h и, как следствие, сброс битов T12STR и STE12. Теперь событие «Период» возникнет в момент времени, когда таймер досчитает до 0004h (момент времени «5»).

Аналогичная ситуация с программной теневой загрузкой значения 0006h отмечена на рисунке цифрами «7» и «8».

Если возникла необходимость изменения длительности периода таймера по окончании текущего периода, то достаточно записать новое значение в регистр CC6U_T12PR (на рисунке 19.4 это изменение состояния регистра T12PRS с 0006h на 0011h). При очередном обнулении таймера – момент времени «10» – новое значение вступит в силу.

После того как бит STE12 установлен, логика блока ожидает переключение таймера. В течение этого времени есть возможность отменить ожидание теневой загрузки. Для этого нужно записать единицу в бит T12STD регистра CCU6_TCTR4. Установка этого бита сбросит бит T12STR и, как следствие, бит STE12. После этого бит T12STD следует сбросить программно, поскольку этот бит не сбрасывается аппаратно. Установленный бит T12STD запрещает выполнение всех аппаратных теневых загрузок.

Следует помнить, что установка бита T12STD отменяет только текущий программный запрос на теневую передачу (если он был), но не запрещает формирование последующих программных запросов.

Для остановки таймера следует установить бит T12RR. После этого бит T12R сбросится, что вызовет остановку таймера T12 и сброс бита T12RR. Остановленный таймер не обнуляется, а сохраняет свое состояние и при следующем запуске продолжает счет.

Для обнуления таймера следует программно записать значение 0000h в регистр CCU6_T12, если таймер остановлен, или воспользоваться битом T12RES регистра CCU6_TCTR4. Этот бит позволяет обнулять таймер в любой момент времени, не оказывая влияния на его работу.

Режим двунаправленного счетчика

Пример работы таймера T12 в режиме двунаправленного счетчика представлен на рисунке 19.5.

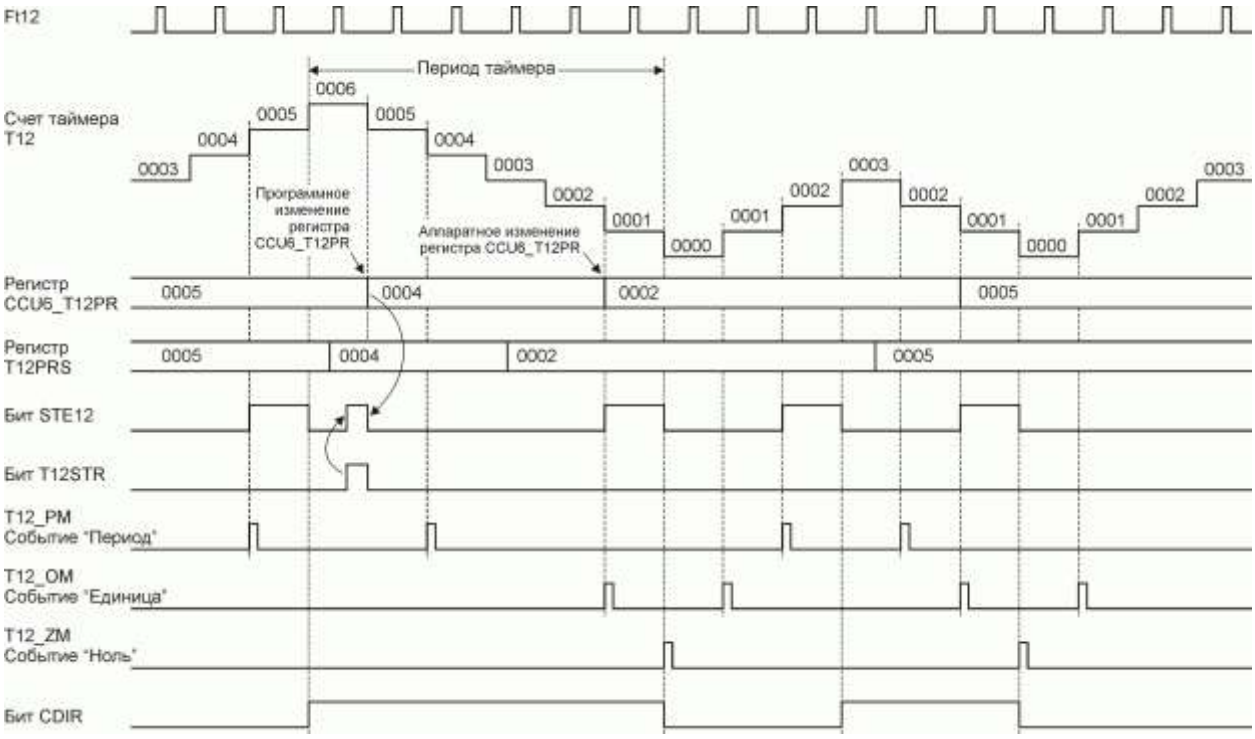


Рисунок 19.5 – Управляющие сигналы блока таймера T12 в режиме двунаправленного счетчика

После запуска, таймер всегда начинает считать вверх с частотой сигнала Ft12. Каждый раз при инкрементировании значение таймера сравнивается со значением регистра периода CCU6_T12PR, и как только возникает событие «Период», генерируется сигнал T12_PM, и аппаратно устанавливается бит STE12. Со следующим тактом сигнала тактирования Ft12 таймер инкрементируется еще раз. Одновременно с этим в регистр периода CCU6_T12PR осуществляется аппаратная теневая загрузка значения из теневого регистра T12PRS, бит STE12 сбрасывается, а бит-индикатор направления счета CDIR регистра CCU6_TCTR0 устанавливается в единицу. Со следующим тактом сигнала тактирования таймер начинает считать вниз.

Далее при каждом декрементировании значение таймера проверяется на совпадение со значением 0001h. Как только возникает событие «Единица», генерируется сигнал T12_OM, и снова аппаратно устанавливается бит STE12. Со следующим тактом сигнала тактирования таймер декрементируется еще раз. Одновременно с этим осуществляется аппаратная теневая загрузка в регистр CC6U_T12PR, возникает событие «Ноль», и генерируется сигнал T12_ZM, бит STE12 сбрасывается, а бит-индикатор направления счета CDIR обнуляется. Со следующим тактом сигнала тактирования Ft12 таймер начинает считать вверх.

Таким образом, после событий «Период» и «Единица», таймер переключается еще раз, прежде чем меняет направление счета.

Время работы таймера между двумя последовательными переключениями направления счета является периодом таймера. Период таймера может оставаться постоянным или изменяться.

Управление таймером и его периодом аналогично управлению в однонаправленном режиме.

Режим однотактного запуска

Режим включается установкой бита T12SSC регистра CCU6_TCTR2 и позволяет запускать таймер на один период. Режим может быть включен в любой момент времени до начала или во время счета таймера T12. После установки бита T12SSC таймер T12 будет считать до окончания текущего периода.

В режиме однонаправленного счетчика таймер остановится после обнуления, вызванного событием «Период». В режиме двунаправленного счетчика таймер остановится по достижении нуля после события «Единица» при счете вниз, смотри рисунок 19.6. Бит не сбрасывается аппаратно, что позволяет запускать таймер на один период желаемое количество раз. Для выключения режима нужно сбросить бит T12SSC.

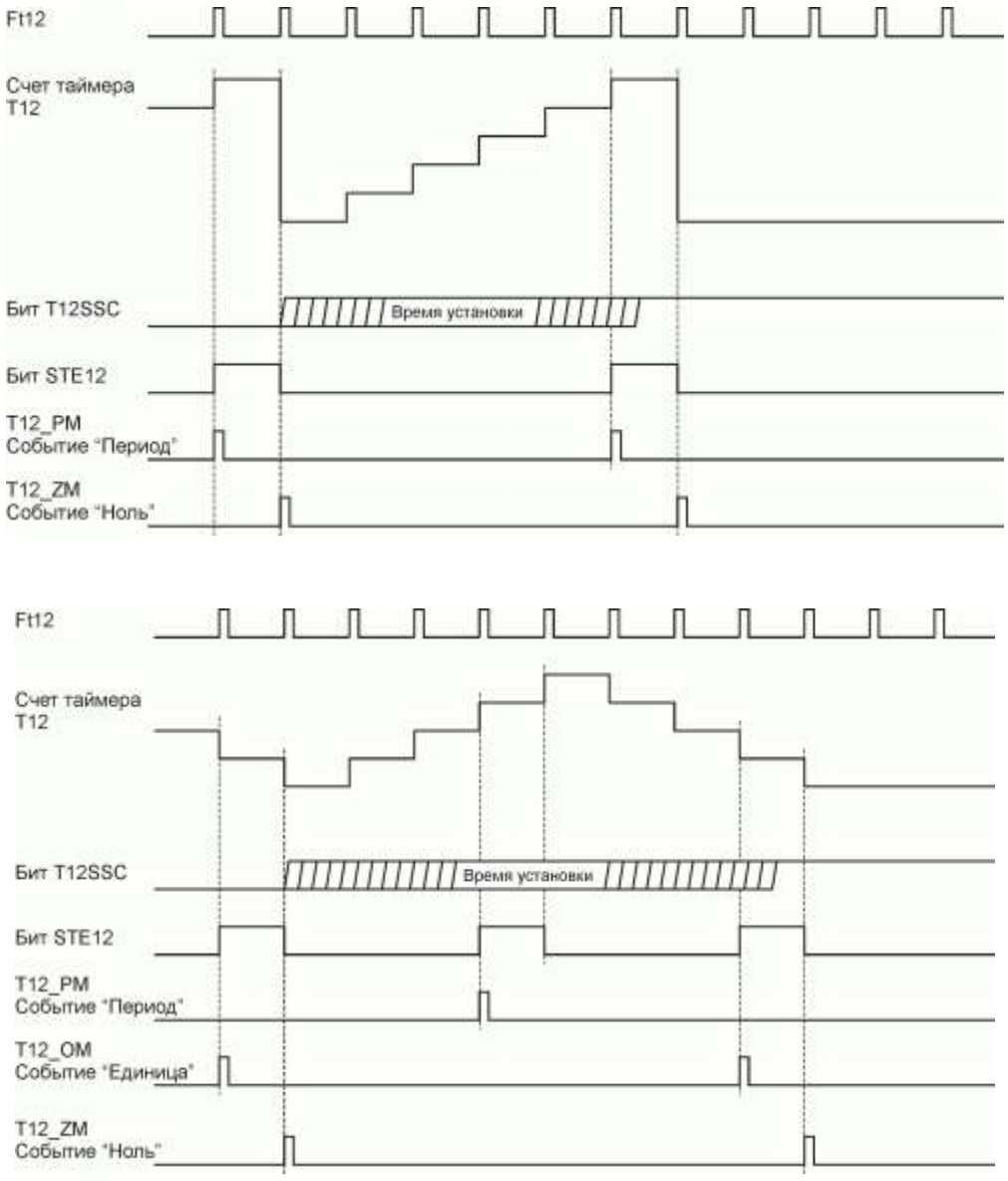


Рисунок 19.6 – Однотактный запуск в режимах однонаправленного и двунаправленного счетчика

На рисунке 19.6 указано «Время установки». Это промежуток времени в пределах одного периода таймера, в течение которого можно установить бит T12SSC, чтобы включить режим однотактного запуска. Если к моменту обнуления таймера бит T12SSC будет установлен, таймер остановится.

Режимы сравнения блока таймера T12

Каждый из трех каналов захвата/сравнения имеет в своем составе 16-разрядные основной регистр и связанный с ним теневой регистр. Основные регистры – CCU6_CC60, CCU6_CC61, CCU6_CC62, теневые – CC60SR, CC61SR, CC62SR.

В каждой паре регистров основной и теневой регистр имеют один адрес. Основной регистр доступен только для чтения. При записи числа в основной регистр, записываемое число размещается в соответствующем теневом регистре. Данные из теневого регистра попадают в основной при теневой загрузке.

Теневая загрузка всех трех основных регистров происходит одновременно по сигналу T12_ST, который формируется при установке бита STE12. Основные регистры через компараторы связаны с таймером T12. При совпадении значений регистра CCU6_CC6xR и таймера возникает событие «Совпадение x», и формируется соответствующий сигнал CM_6x (здесь и далее по тексту под символом «x» подразумевается номер канала). Одновременно с этим логика блока отслеживает направление счета таймера, чтобы при возникновении в канале 1 события «Совпадение 1 при счете вниз» сформировать дополнительный сигнал CM_61_DN, а при возникновении события «Совпадение 1 при счете вверх» – сигнал CM_61_UP. На рисунке 19.7 показано функциональное взаимодействие регистров каналов и таймера T12.

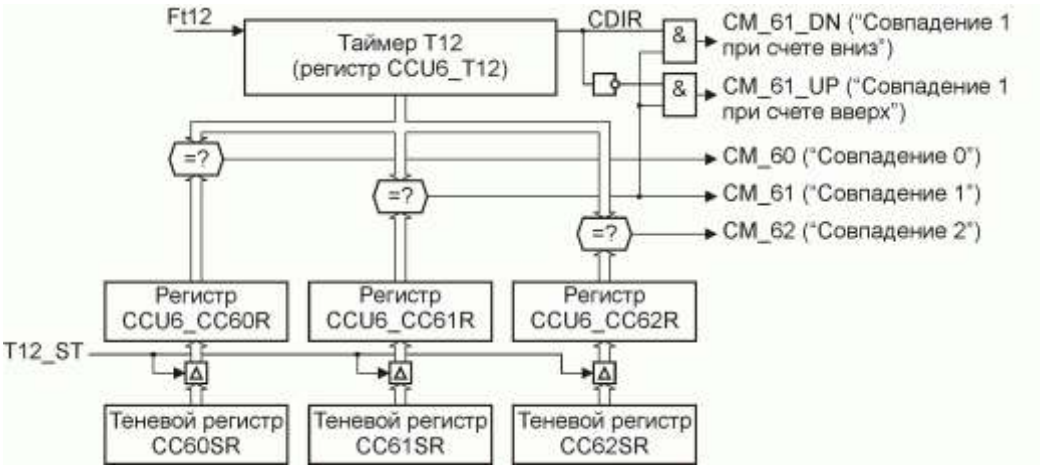


Рисунок 19.7 – Взаимодействие таймера T12 и регистров каналов

Каждый канал захвата/сравнения имеет бит состояния CC6xST, отражающий состояние выходной линии канала. Программно бит доступен в регистре CCU6_CMPSTAT.

Примечание – Если бит является источником управляющего сигнала, то название бита и сигнала могут совпадать. Так, например, бит CC60ST является источником сигнала CC60ST. Установка и сброс бита формируют передний и задний фронты соответствующего сигнала.

Функционированием бита управляет логика сброса/установки бита CC6xST. На рисунке 19.8 показана функциональная схема взаимодействия управляющей логики и бита CC60ST канала 0. На аппаратном уровне на состояние бита CC60ST влияют биты T12R, CDIR, битовое поле режима MSEL60 (регистр CCU6_T12MSEL), а также сигналы CM_60, T12_ZM и сигнал T12_SSEP завершения режима однотактного запуска, генерируемый таймером T12.

Программно управление битом CC60ST осуществляется посредством битов MCC60S и MCC60R регистра CCU6_CMPMODIF.

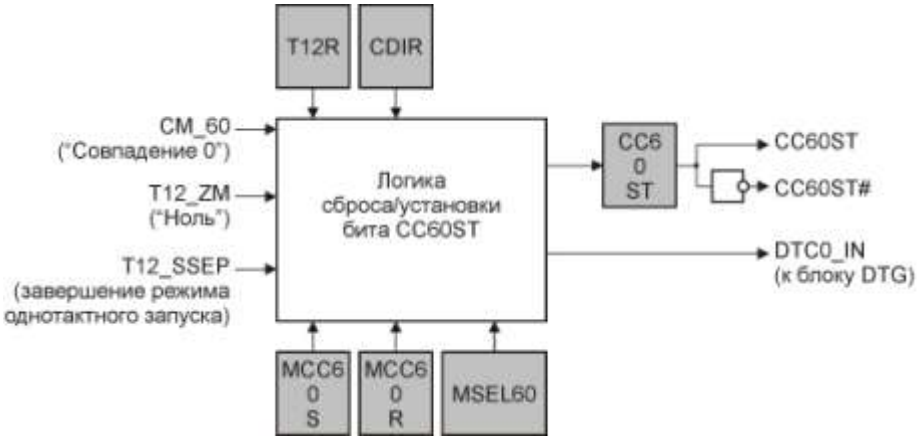


Рисунок 19.8 – Функциональная схема управления битом CC60ST канала 0

Функциональные схемы каналов 0, 1 и 2 в целом аналогичны. Некоторые особенности функционального устройства и работы каналов будут рассмотрены ниже.

Примечание – В режиме Холла (режим работы микроконтроллера с датчиками Холла) на состояние бита CC60ST влияют дополнительные сигналы, не показанные на рисунке 19.8. Эти сигналы будут описаны ниже.

Аппаратное управление битом CC6xST

Аппаратное изменение бита CC6xST возможно только при работающем таймере T12.

Правила поведения бита приведены в таблице 19.1.

Таблица 19.1 – Правила переключения бита CC6xST

Поведение бита CC6xST	Событие, являющееся причиной установки или сброса бита в зависимости от режима работы таймера T12	
	Однонаправленный счетчик	Двунаправленный счетчик
Установка	«Совпадение x» (в том числе при обнулении таймера)	«Совпадение x» при счете вверх (в том числе, если совпадение возникло при смене направления счета таймера в конце периода)
Сброс	«Ноль» (если не возникло «Совпадение x» при обнулении таймера)	«Совпадение x» при счете вниз

Если в момент переключения таймера возникает условие для установки/сброса бита CC6xST, то смена состояния бита произойдет только в момент следующего переключения таймера.

На рисунках 19.9 и 19.10 приведены примеры аппаратного управления битом CC6xST при работе таймера T12 в режиме однонаправленного и двунаправленного счетчика.

Сформированные сигналы CC6xST и инверсные по отношению к нему CC6xST# далее передаются на блок управления выходной модуляцией. На рисунке 19.11 представлена функциональная схема блока канала 0. Функциональные схемы каналов 2 и 3 аналогичны.

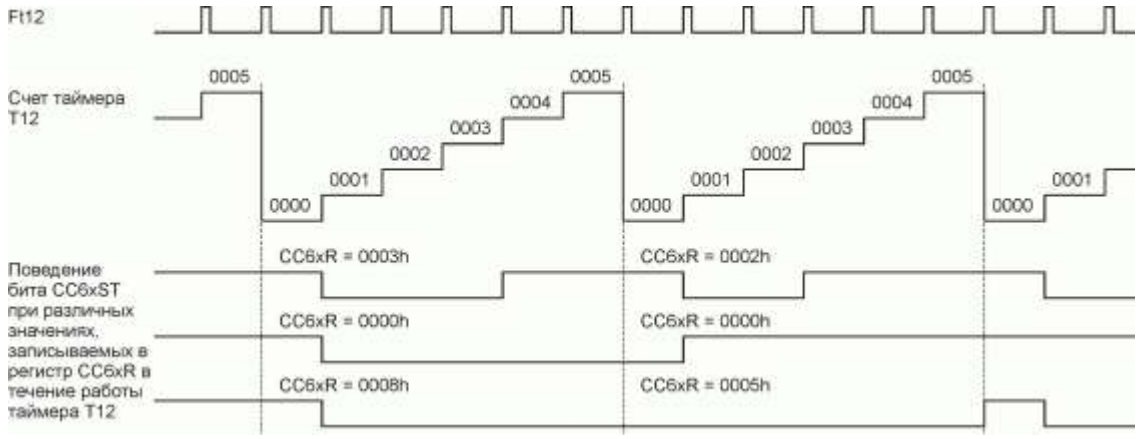


Рисунок 19.9 – Аппаратное управление битом CC6xST
в режиме однонаправленного счетчика

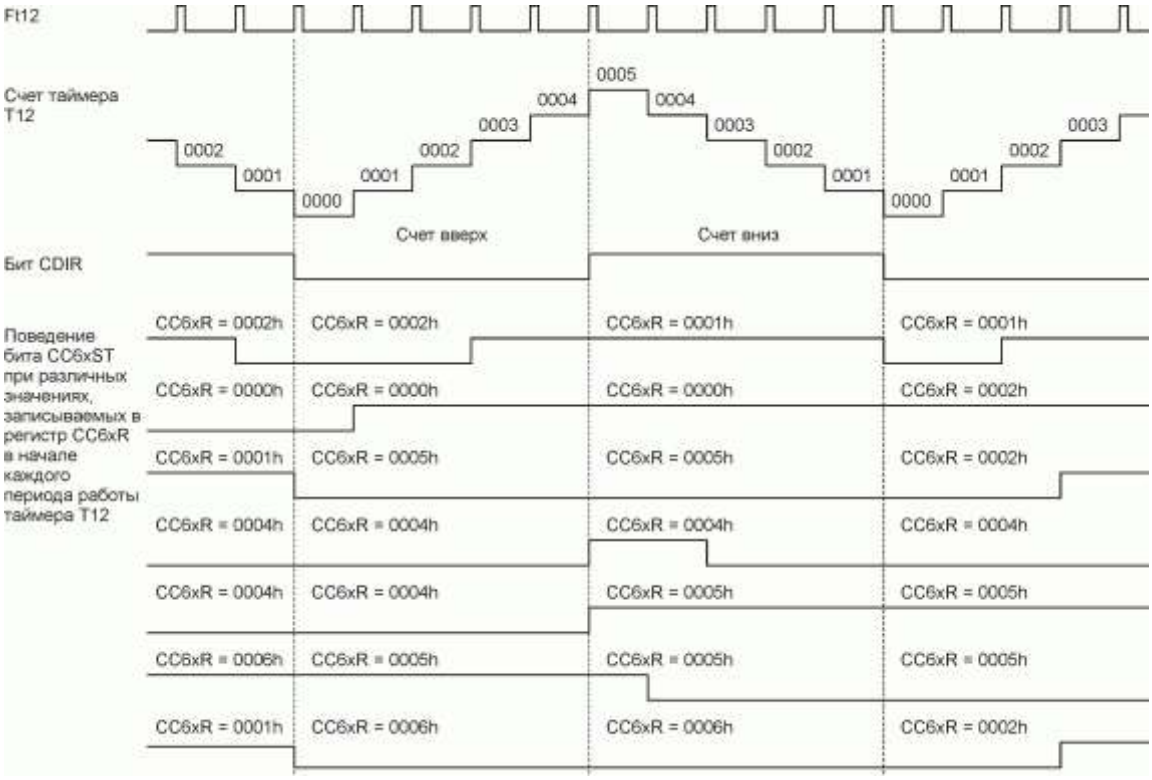


Рисунок 19.10 – Аппаратное управление битом CC6xST в режиме двунаправленного счетчика

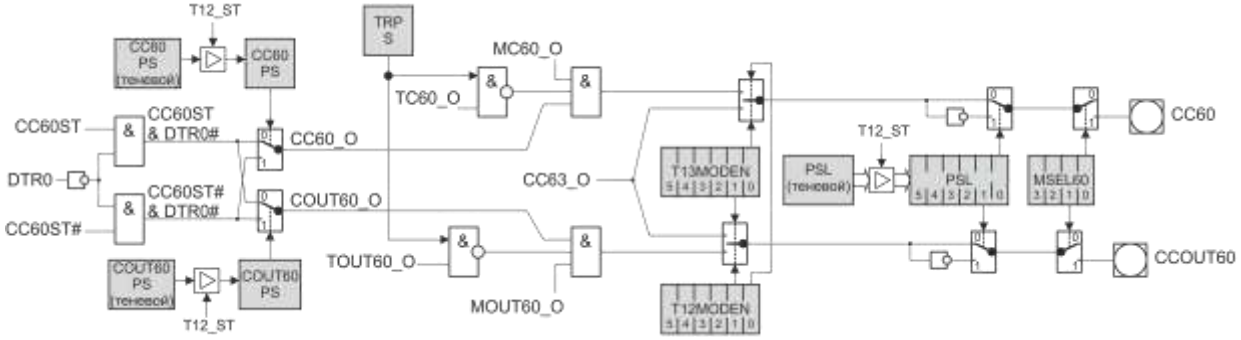


Рисунок 19.11 – Блок управления выходной модуляцией канала 0

Из рисунка 19.11 видно: входные сигналы CC60ST и CC60ST# логически умножаются с сигналом DTR0#.

Сигнал DTR0 формируется в блоке генератора задержки выходного сигнала (далее – блок DTG, который будет подробно рассмотрен ниже). По умолчанию, сигнал DTR0 (как и DTR1 и DTR2 остальных каналов) имеет низкий уровень.

Бит CC60PS указывает, какой из двух сигналов – прямого и инверсного – использовать для дальнейшей модуляции на линии CC60_O, а бит COUT60PS – на линии COUT60_O.

Примечание – Биты CC6xPS и COUT6xPS доступны посредством регистра CCU6_CMPSTAT и имеют теневые биты, загрузка из которых выполняется одновременно с теневой загрузкой регистров CCU6_CC6xR каналов.

Сигналы MC60_O, MOUT60_O, TC60_O, TOUT60_O и бит TRPS используются в мультиканальных режимах и механизме ловушки, которые будут рассмотрены ниже. По умолчанию состояние этих сигналов не препятствует прохождению сигналов CC60_O и COUT60_O, см. рисунок 19.11.

Логика канала позволяет работать с сигналами блока таймера T12 и выходным сигналом CC63_O блока таймера T13. Для выбора источников формирования сигналов следует установить соответствующие биты в поле T12MODEN или (!) T13MODEN. По умолчанию все биты сброшены и источник не выбран, поэтому на линиях удерживается низкий уровень сигнала. Следует помнить, что в случае одновременной установки битов, управляющих одним ключом, на выходе ключа будет сформирован логический ноль.

Битовое поле PSL регистра CCU6_PSLR управляет инвертированием модулируемых сигналов. Это поле имеет теневое поле, значение которого загружается в PSL по сигналу T12_ST теневой передачи таймера T12.

Битовое поле MSEL60 регистра управляет выводом сигналов CC60 и CCOUT60 непосредственно перед передачей их в портовую логику при работе блока таймера T12 в режиме сравнения и в режиме работы с датчиками Холла.

Программное управление битом CC6xST

Бит CC6xST может быть установлен или сброшен программно в любой момент времени, независимо от его текущего состояния. Для этого имеются пары битов MCC6xS – MCC6xR, которые доступны через регистр CCU6_CMPMODIF, см. рисунок 19.12.

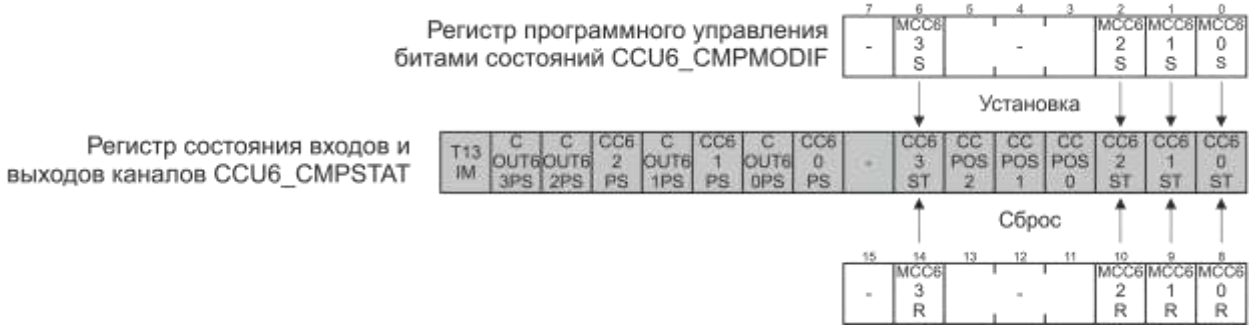


Рисунок 19.12 – Программное управление установкой и сбросом битов состояний

Блок DTG

Микроконтроллер может использоваться как управляющее устройство для силовых ключей (транзисторов) в системе управления двигателем постоянного тока.

Особенностью работы силовых ключей является то, что время их открывания заметно больше времени запирания. Во избежание появления сквозных токов, открывание одних ключей должно быть задержано на некоторое время, достаточное для запирания других. Эта задержка является, так называемым, временем «dead-time».

Для аппаратного формирования и внесения задержек в модулируемые сигналы, а именно затягивания передних (только передних!) фронтов переключения сигналов на

выходных линиях, используется блок DTG. Блок условно состоит из трех подблоков, каждый – для работы с отдельным каналом захвата/сравнения. Структурная схема блока DTG представлена на рисунке 19.13.

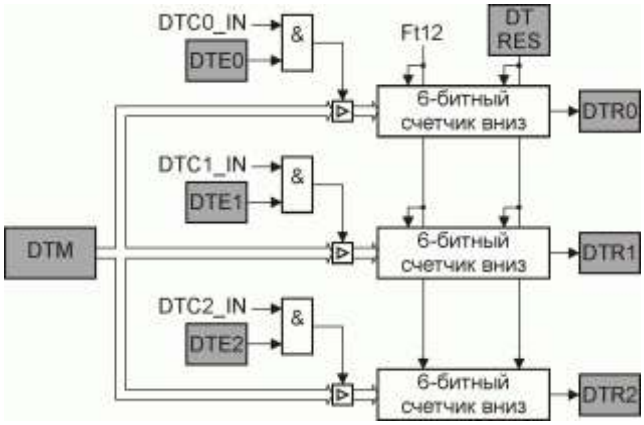


Рисунок 19.13 – Блок DTG

Для включения генератора задержки канала следует установить соответствующий бит разрешения DTEx в регистре CCU6_T12DTC.

Любое переключение бита CC6xST генерирует сигнал DTCx_IN, появление которого инициирует загрузку значения задержки из битового поля DTM в 6-битный счетчик и запускает его. Одновременно с этим устанавливается флаг задержки DTRx и остается до окончания счета. Счетчик тактируется тем же сигналом Ft12 и с той же частотой, что и таймер T12. Счет идет вниз. Как только счетчик достигнет нуля, флаг DTRx сбросится.

Роль флага DTRx можно понять из рисунка 19.11 на примере бита DTR0. Видно, что сигнал DTR0 подключен к блоку выходной модуляции через инвертор. Это указывает на то, что пока флаг DTR0 установлен, передача сигналов CC60ST и CC60ST# заблокирована.

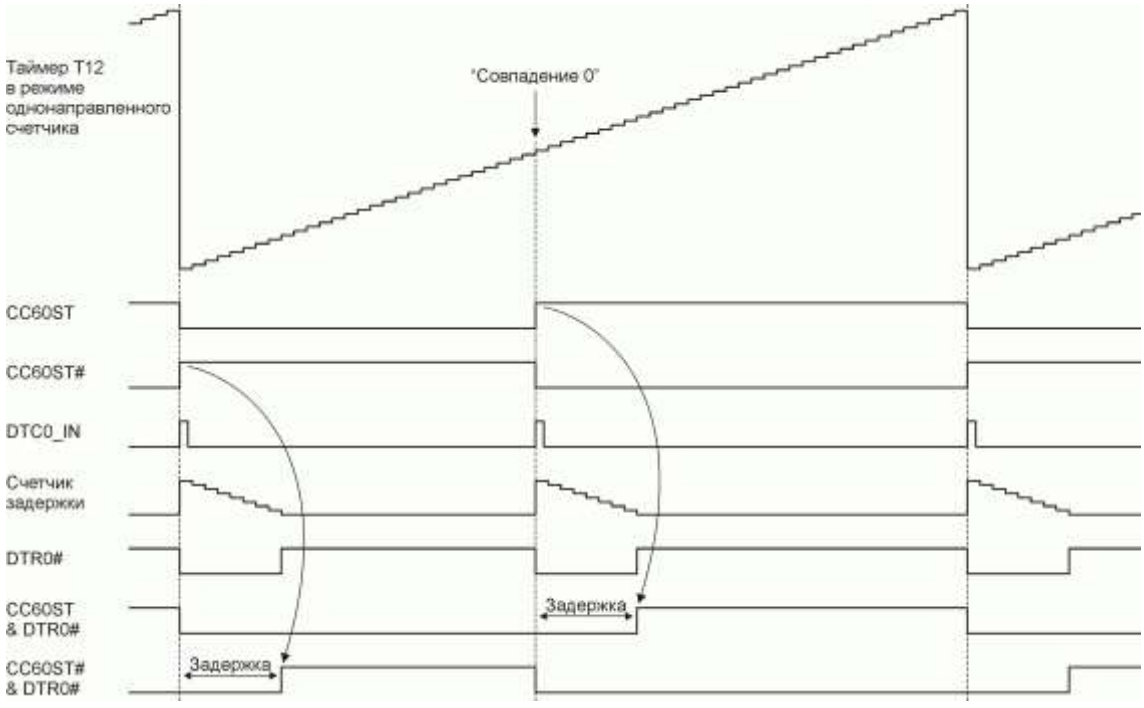


Рисунок 19.14 – Внесение задержки в переключение сигналов

Особенностью счетчиков задержки является то, что после загрузки и запуска и до окончания счета они не могут быть перезагружены. Это позволяет избежать повторного запуска в случае, если состояние бита CC6xST изменится в течение времени задержки. Тем не менее, все счетчики могут быть принудительно обнулены и, соответственно, остановлены установкой бита DTRES.

На рисунке 19.14 показан пример задержки переключения сигналов канала 0. Функционирование каналов 1 и 2 аналогично.

Блок таймера T12 поддерживает четыре основных и пять мультивходовых режимов захвата, а также два специальных режима работы с датчиками Холла и блокирования. Во всех режимах захвата основным действием является захват значения таймера T12 в основной или/и в теневой регистры канала при обнаружении запрограммированного события на входе CC6x канала. Режимы программируются посредством поля MSEL6x регистра CCU6_T12MSEL.

Режимы захвата блока таймера T12

На рисунке 19.15 показана функциональная схема работы канала 0 в режиме захвата 1. Когда на входе CC60 обнаруживается передний фронт сигнала, детектор фронта генерирует соответствующий сигнал, по которому текущее состояние таймера T12 захватывается в теневой регистр CC60SR. В случае обнаружения на входе CC60 заднего фронта сигнала, текущее состояние таймера T12 захватывается в основной регистр CCU6_CC60R. Одновременно с захватом значения таймера происходит установка бита CC60ST. Сбрасывать бит нужно программно.

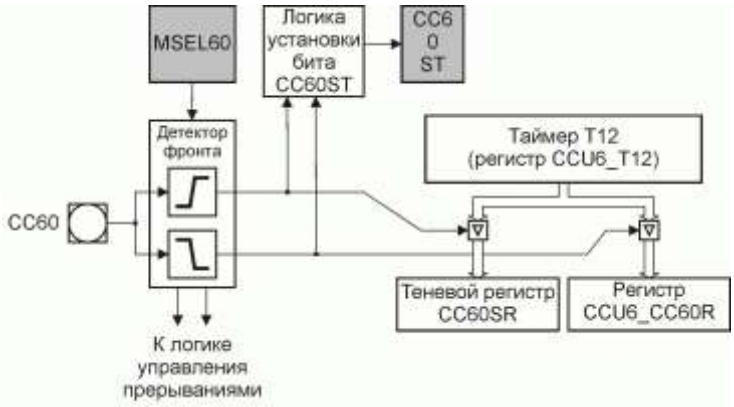


Рисунок 19.15 – Функциональная схема канала 0 в режиме захвата 1

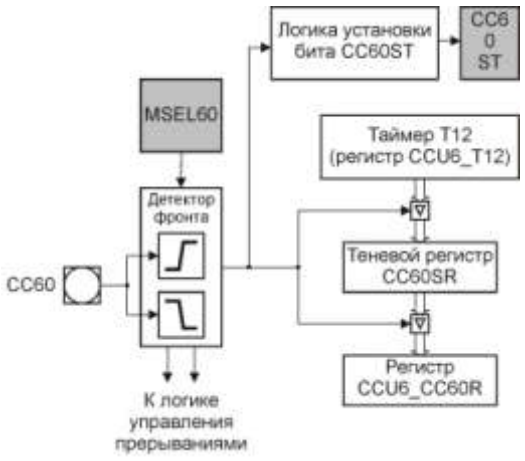


Рисунок 19.16 – Функциональная схема канала 0 в режимах захвата 2, 3 и 4

На рисунке 19.16 показана функциональная схема работы канала 0 для режимов захвата 2, 3 и 4. Когда на входе CC60 обнаруживается ожидаемый фронт сигнала, текущее состояние теневого регистра захватывается в основной регистр, а в теневой регистр записывается текущее состояние таймера T12. Одновременно с захватом происходит установка бита CC60ST. Сбрасывать бит нужно программно.

Функциональная схема работы в мультиканальном режиме захвата показана на рисунке 19.17.

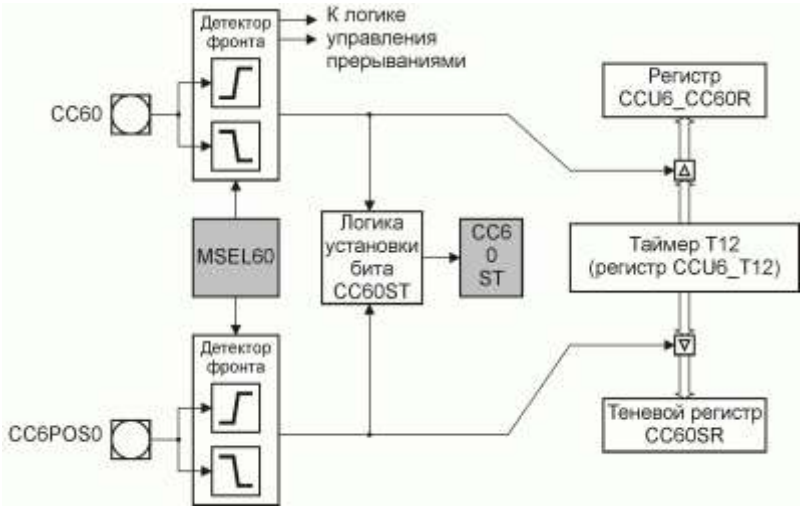


Рисунок 19.17 – Функциональная схема канала 0 для мультиканального режима

Когда на входе CC60 обнаруживается запрограммированный фронт сигнала, детектор фронта генерирует соответствующий сигнал, по которому текущее состояние таймера T12 захватывается в основной регистр CCU6_CC60R.

В случае обнаружения запрограммированного фронта на входе CC6POS0, текущее состояние таймера T12 захватывается в теневой регистр CC60SR.

Одновременно с каждым захватом значения таймера происходит установка бита CC60ST. Сбрасывать бит нужно программно.

Во всех режимах захвата работа каналов 2 и 3 аналогична работе канала 0.

Регистры CCU6_CC6xR и CC6xSR всегда доступны для чтения. Детектор фронта (независимо от выбранного режима захвата), при обнаружении фронта сигнала на входе CC6x, генерирует соответствующий запрос на прерывание, поступающий далее на блок управления прерываниями модуля CAPCOM6.

Примечание – Детекторы фронта, работающие с входами CC6POSx, не генерируют запросов на прерывания.

Режим блокирования

Режим (MSEL = 1001b) позволяет запрещать модулирование сигналов канала, посредством удержания бита CC6xST в нулевом состоянии и блокирования его установки под управлением сигнала, приходящего на вход CC6POSx. Функциональная схема работы канала 0 в режиме блокирования показана на рисунке 19.18.

Рисунок 19.18 – Функциональная схема для режима блокирования канала 0

Появление низкого уровня сигнала на входе CC6POS0 сбрасывает бит CC60ST и удерживает его в нулевом состоянии.

После появления высокого уровня сигнала на входе CC6POS0, блокирование бита CC60ST снимается, и его состояние переходит под управление логики сброса/установки.

Особенностью режима блокирования является то, что он в любой момент может быть включен параллельно с любым из режимов сравнения. Если в поле MSEL6x записать значение 1001b, то это не внесет изменений в работу соответствующего канала, но дополнительно будет подключен вход CC6POSx, состояние которого будет влиять на функционирование бита CC6xST. Для выключения режима блокирования нужно записать в поле MSEL6x значение, соответствующее режиму сравнения.

19.2 Блок таймера T13

В состав блока входит один канал с двумя выходами, взаимодействующий с 16-разрядным таймером и поддерживающий только режим сравнения. Канал имеет регистр, который через компаратор связан с таймером. При совпадении значений таймера и регистра на выходе канала формируется сигнал.

Таймер T13 представляет собой счетчик (регистр CCU6_T13), который может только инкрементироваться. Тактовый сигнал таймера формируется на основе внешнего тактового сигнала Fcc6. Кроме того работа таймера T13 может синхронизироваться событиями таймера T12. Через компаратор таймер T13 связан с регистром периода CCU6_T13PR (хранит значение, по достижении которого таймер сбрасывается). На рисунке 19.19 показана функциональная схема взаимодействия таймера и регистров периода.

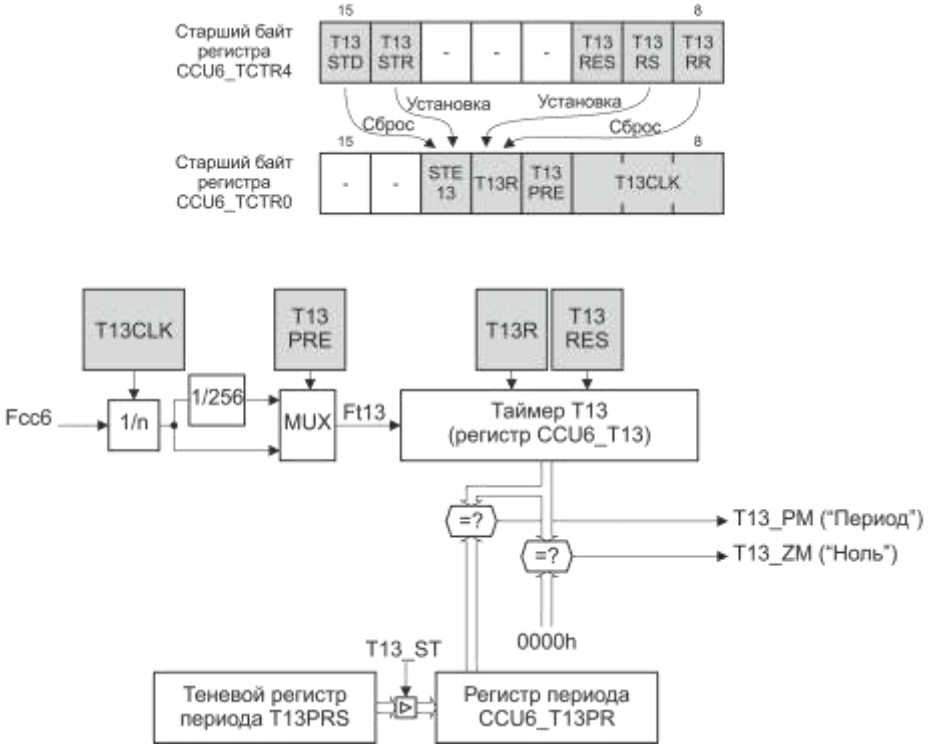


Рисунок 19.19 – Таймер T12 и регистры управления

Регистр T13PRS выполняет роль буфера для хранения значения, которое должно быть записано в регистр CCU6_T13PR по внутреннему сигналу T13_ST, формируемому в случае программного или аппаратного запроса. Регистр CCU6_T13PR является основным,

а регистр T13PRS – теневым. Если таймер включен, теневая загрузка выполняется только в момент переключения таймера.

Дополнительный компаратор позволяет отслеживать моменты, когда значение таймера равно нулю.

Совпадение значения таймера и его регистра периода является событием «Период», достижение нуля – событием «Ноль».

Работой таймера T13 (также как и работой T12) управляют два регистра CCU6_TCTR0 и CCU6_TCTR4.

Пока таймер не запущен, его регистр CCU6_T13 доступен для записи. По умолчанию, начальное состояние таймера 0000h. Запись в регистр таймера во время работы игнорируется. В то же время для чтения таймер доступен всегда.

Для запуска таймера T13 нужно установить бит T13RS в регистре CCU6_TCTR4. После этого аппаратно установится бит T13R в регистре CCU6_TCTR0 и таймер запустится.

Таймер T13 может функционировать только в режиме однонаправленного счетчика с возможностью однотактного запуска. Эти режимы полностью аналогичны соответствующим режимам таймера T12.

Синхронизация таймеров T12 и T13

Запуск таймера T13 может быть синхронизирован событиями блока таймера T12. Для этого следует задать ожидаемое событие и направление счета таймера T12, при котором указанное событие будет отслеживаться. Параметры записываются в поля T13TEC и T13TED регистра CCU6_TCTR2. При обнаружении запрограммированного события бит T13R аппаратно устанавливается и таймер T13 запускается.

Примечание – Частоты таймеров T12 и T13 могут не совпадать, поскольку имеется возможность их независимого программирования.

На рисунке 19.20 показан пример запуска таймера T13 при обнаружении совпадения значений таймера T12 и регистра CCU6_CC62, в котором находится значение 0002h при счете вверх таймера T12 (T13TED = 01b, T13TEC = 011b).

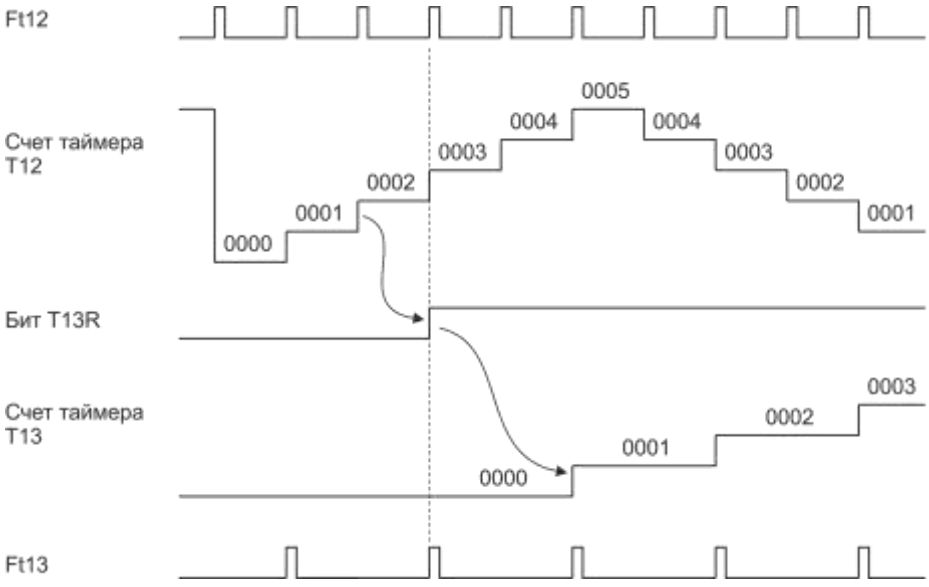


Рисунок 19.20 – Синхронизация запуска таймера T13 по таймеру T12

Режимы сравнения блока таймера T13

Канал блока таймера T13 имеет в своем составе 16-разрядные основной регистр CCU6_CC63 и связанный с ним теневой регистр CC63SR, см. рисунок 19.21.

Основной и теневой регистры имеют один адрес. Основной регистр доступен только для чтения. При записи числа в основной регистр записываемое число размещается в

соответствующем теневом регистре. Данные из теневого регистра попадают в основной при теневой загрузке по сигналу T13_ST, который формируется при установке бита STE13.

Основной регистр через компаратор связан с таймером T13. При совпадении значений регистра CCU6_CC63R и таймера возникает событие «Совпадение 3» и формируется сигнал CM_63.

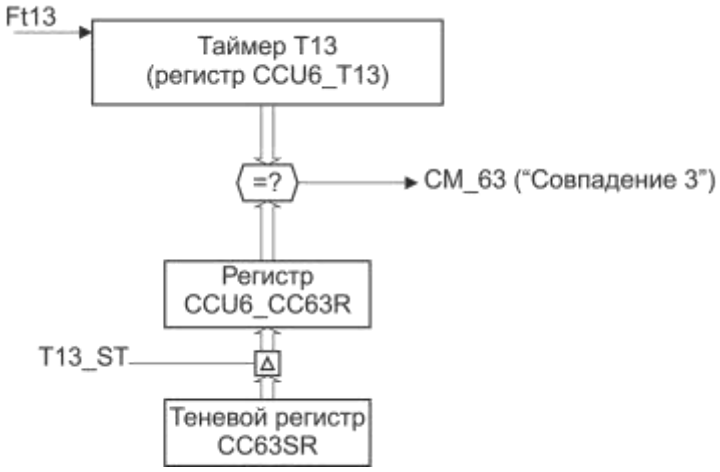


Рисунок 19.21 – Взаимодействие таймера T13 и регистра канала

Канал имеет бит состояния CC63ST, отражающий состояние выходной линии канала. Программно бит доступен в регистре CCU6_CMPSTAT. Установка и сброс бита формируют передний и задний фронты сигнала CC63ST.

На рисунке 19.22 показана функциональная схема взаимодействия управляющей логики и бита CC63ST. На аппаратном уровне на состояние бита CC63ST влияют бит T13R, сигналы CM_63, T13_ZM и сигнал T13_SSEP завершения режима однотактного запуска, генерируемый таймером T13.

На программном уровне управление битом CC63ST осуществляется посредством битов MCC63S и MCC63R регистра CCU6_CMPMODIF.

Рисунок 19.22 – Функциональная схема управления битом CC63ST

Аппаратное управление битом CC63ST

Аппаратное изменение бита CC63ST возможно только при работающем таймере T13. Правила поведения бита CC63ST приведены в таблице 19.2.

Таблица 19.2 – Правила переключения бита CC63ST

Поведение бита CC63ST	Событие, являющееся причиной установки или сброса бита
Установка	«Совпадение 3» (в том числе при обнулении таймера)
Сброс	«Ноль» (если не возникло «Совпадение 3» при обнулении таймера)

Если в момент переключения таймера возникает условие для установки/сброса бита CC63ST, то смена состояния бита произойдет только в момент следующего переключения таймера. Функционирование бита CC63ST полностью аналогично функционированию битов CC6xST блока таймера T12.

Сформированный сигнал CC63ST и инверсный по отношению к нему CC63ST# далее передаются на блок управления выходной модуляцией канала 3, см. рисунок 19.23.

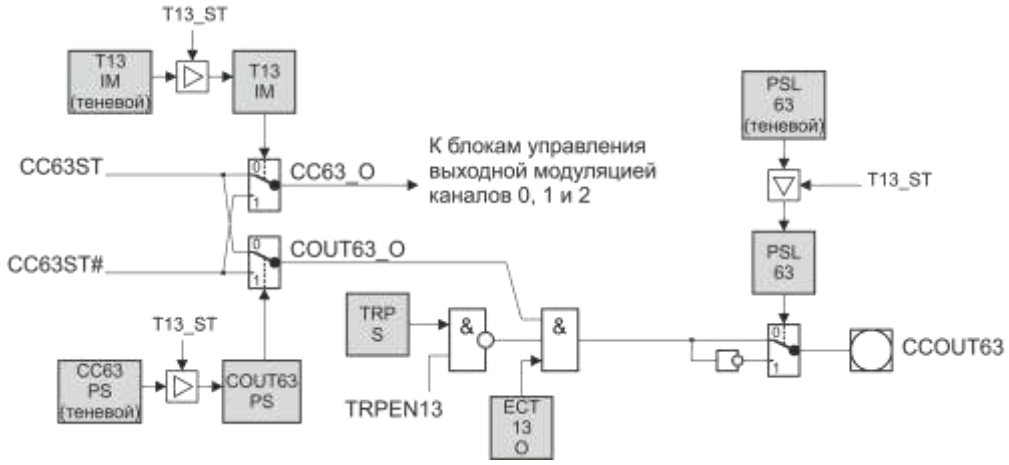


Рисунок 19.23 – Блок управления выходной модуляцией канала 3

Бит CC63PS указывает, какой из двух сигналов – прямой или инверсный – использовать для дальнейшей модуляции на линии COUT63_O. Бит COUT63PS доступен посредством регистра CCU6_CMPSTAT и имеет теневой бит, загрузка из которого выполняется одновременно с теневой загрузкой регистров CCU6_CC63R.

Блок таймера T13 позволяет генерировать ШИМ-сигнал на выходе CCOUT63. Дополнительными битами управления модуляцией сигнала являются бит ECT13O регистра CCU6_MODCTR и бит PSL63 регистра CCU6_PSLR. Бит TRPS и сигнал TRPEN используются механизмом ловушки и будут рассмотрены ниже. По умолчанию, состояние этих сигналов не препятствует прохождению сигнала COUT63_O, см. рисунок 19.23.

Сигнал CC63_O используется в блоках управления выходной модуляцией каналов таймера T12, как показано на рисунке 19.11. Дополнительным битом управления является бит T13IM регистра CCU6_CMPSTAT.

Программное управление битом CC63ST

Бит CC63ST может быть установлен или сброшен программно в любой момент времени, независимо от его текущего состояния. Для этого имеются пары битов MCC63S-MCC63R, которые доступны через регистр CCU6_CMPMODIF.

19.3 Мультиканальный режим работы блоков таймеров

Режим включается установкой бита MCMEN регистра MODCTR и позволяет управлять модуляцией всех шести выходных сигналов таймера T12.

Каждый канал таймера T12 имеет две линии вывода сигнала к выходам CC6x и COUT6x. Каждая линия имеет два источника – сигнал CC6x_O и CC63_O, смотри рисунок 19.11. Выбор источника осуществляется посредством полей T13MODEN и T12MODEN, которые управляют ключами. После сброса содержимое обоих полей равно нулю, вследствие чего сигналы на выходе ключей имеют низкий уровень.

Поля T13MODEN и T12MODEN должны быть обязательно запрограммированы независимо от состояния бита MCMEN, и их значения не должны побитно совпадать.

В мультиканальном режиме прохождение сигналов в каналах дополнительно управляется сигналами MC6x_O и MOUT6x_O, см. рисунок 19.11. Эти сигналы формируются полем MCMP регистра CCU6_MCMOUT. Прохождение сигнала по линии возможно лишь в том случае, когда соответствующий бит установлен, т. е. линия открыта. Закрытые линии удерживаются в нулевом состоянии.

Поле MCMP имеет теневое поле MCMPS в регистре CCU6_MCMOUTS. Передача значения из MCMPS в MCMP может быть инициирована двумя способами:

- программно – сигнал теневой передачи MCM_ST формируется в любой момент, когда это необходимо, установкой бита STRMCM (асинхронно по отношению к работе таймеров);

- аппаратно – сигнал теневой передачи MCM_ST формируется синхронно с событиями таймеров T12 и T13, что позволяет избежать появления нежелательного импульса из-за возможной асинхронности работы таймеров. Индикатором ожидания синхронной передачи является флаг R.

Примечание – Аппаратное генерирование сигнала теневой передачи возможно лишь в случае, если бит MCMEN установлен.

На рисунке 19.24 показана схема формирования сигнала теневой передачи MCM_ST в мультиканальном режиме, схема обработки сигнала – на рисунке 19.25.

Поле SWEL выбирает событие, с которым требуется синхронизация теневой передачи. После обнаружения события устанавливается флаг ожидания R. После того как выбранный полем SWSYN таймер обнулится, и соответствующий сигнал поступит на логику синхронизации загрузки, будет сгенерирован сигнал MCM_ST и содержимое теневого поля MCMPS загрузится в MCMP, флаг R аппаратно сбросится.

Примечание – Если SWSEL = 000b (т. е. событие не задано, поскольку не требуется аппаратная синхронизация), то сигнал теневой передачи будет генерироваться сразу при появлении сигнала заданного полем SWSYN. В случае если SWSEL ≠ 000b, то обязательным условием теневой передачи является установленный флаг R.

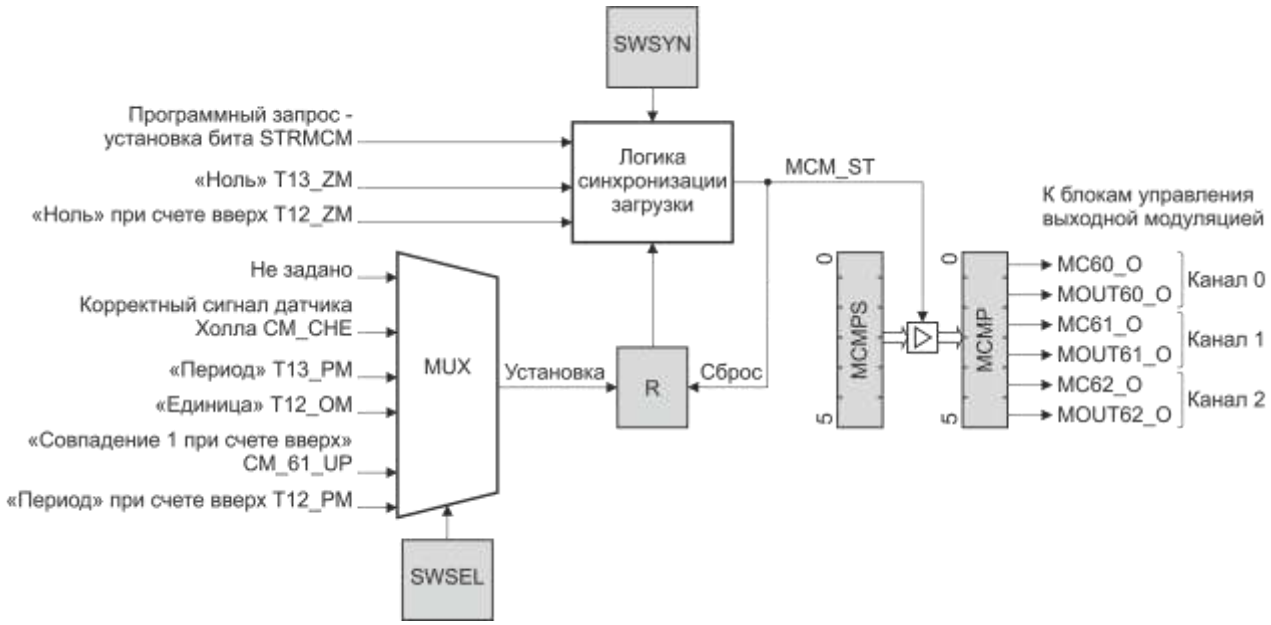


Рисунок 19.24 – Схема формирования сигнала теневой передачи MCM_ST

Поле MCMP аппаратно очищается, если устанавливается флаг IDLE (устанавливается аппаратно сигналом CM_CHE или программно – посредством бита SIDLE). После сброса флага IDLE для возврата к нормальной работе следует программно установить биты STRMCM и STRHP, т. е. сгенерировать два сигнала теневой передачи для загрузки полей MCMP, CURH и EXPH, см. рисунок 19.27.

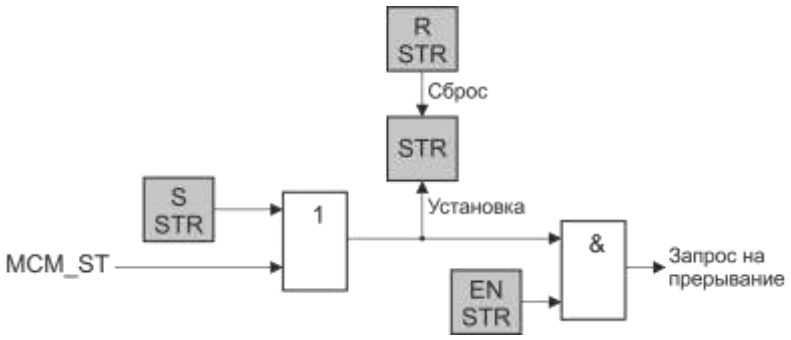


Рисунок 19.25 – Схема обработки сигнала MCM_ST и генерирования прерываний

Примечание – Установка флага STR и генерирование соответствующего прерывания, если разрешено битом ENSTR, происходит параллельно. В связи с этим, нет необходимости обязательного программного сброса флага – прерывания будут генерироваться даже в том случае, если флаг уже установлен.

19.4 Режим работы с датчиками Холла

Бесщеточные двигатели постоянного тока (БДПТ) состоят из статора с многофазной обмоткой и ротора в виде постоянного магнита. Для контроля положения ротора используются различные датчики, среди которых наиболее широкое распространение получили датчики, работающие на основе эффекта Холла. Их размещают, как правило, на статоре так, чтобы на них воздействовали магниты ротора. Конструктивное расположение датчиков зависит от количества полюсов ротора. При использовании трех датчиков Холла

с дискретными выходами угол между ними должен быть 120 «электрических» градусов. В этом случае положение ротора кодируется шестью двоичными комбинациями от 001 до 110. Комбинации 000 и 111 считаются ошибочными.

Работа с датчиками Холла

Для примера, на рисунке 19.26 показан постоянный четырехполюсный магнит ротора, схема расположения датчиков Холла Д0, Д1, Д2 и их подключение к выводам микроконтроллера.

Между выходным ШИМ-сигналом микроконтроллера и позицией ротора существует жесткая связь. Как только обнаруживается ожидаемая комбинация состояний датчиков, что указывает на то, что ротор достиг желаемого положения, на выходы микроконтроллера выставляется очередная комбинация сигналов управления.

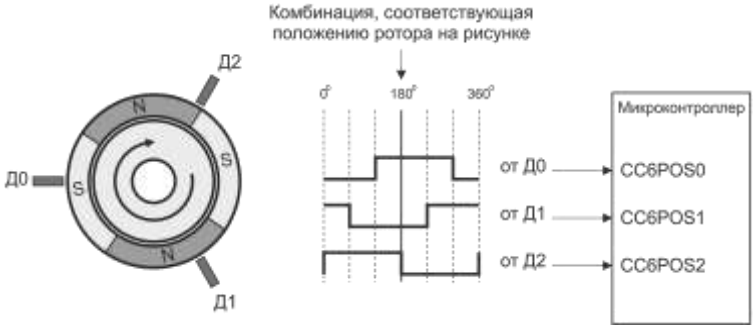


Рисунок 19.26 – Схема расположения датчиков Холла и комбинации их состояний

Текущее состояние датчиков хранится в поле CURH, а ожидаемое – в поле EXPH регистра CCU6_MCMOUT. Текущее состояние сигналов управления, т. е. состояние выводов находится под управлением поля MCMP, см. рисунок 19.24. Все каналы блока таймера T12 должны быть запрограммированы на работу в режиме датчиков Холла, т. е. MSEL60 = MSEL61 = MSEL62 = 1000b.

Информация от датчиков Д1, Д2 и Д3 поступает на входы микроконтроллера CC6POS0, CC6POS1 и CC6POS2 и далее к модулю CAPCOM6. Функциональная схема обработки сигналов от датчиков Холла показана на рисунке 19.27.

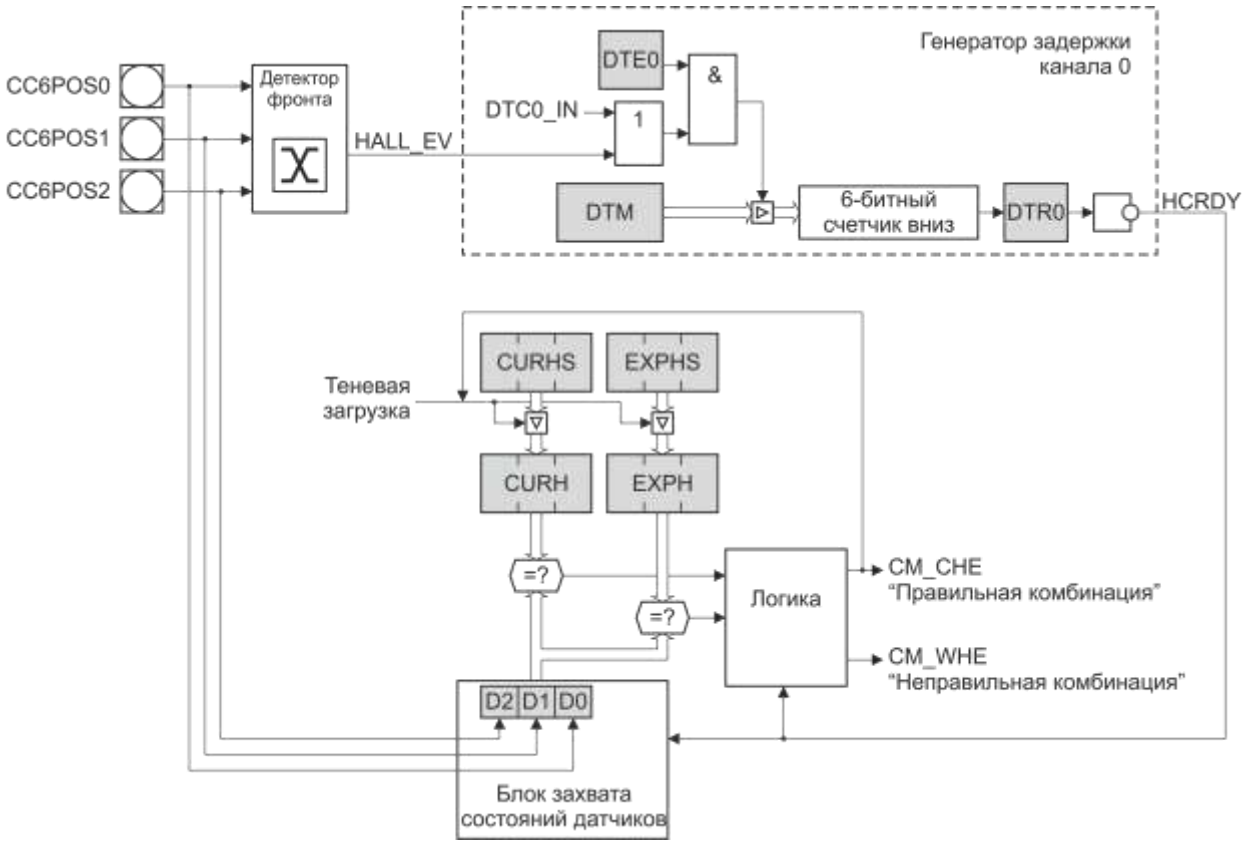


Рисунок 19.27 – Функциональная схема обработки сигналов от датчиков Холла

Только детектор фронта канала 0 имеет возможность одновременно мониторить все три входа CC6POS0, CC6POS1 и CC6POS2. Так же в процессе обработки информации от датчиков участвует генератор задержки канала 0 (блок DTG), блок захвата состояний датчиков и регистры комбинаций состояний со своей логикой.

Когда на любом из входов CC6POS0, CC6POS1, CC6POS2 обнаруживается перепад сигнала (опрос происходит каждый такт синхросигнала Fcc6), детектор фронта генерирует сигнал HALL_EV, который поступает на генератор задержки и далее – на блок захвата состояний датчиков.

Если задержка «dead-time» не используется, то сигнал захвата состояний датчиков HCRDY генерируется незамедлительно, но в этом случае фильтрация входных шумов, которые могут поступать от датчиков, отсутствует, и считываемые комбинации могут не соответствовать реальному положению ротора. Во избежание этого от момента обнаружения изменений на входах CC6POS0, CC6POS1, CC6POS2 и до их считывания желательно вносить программируемую задержку с помощью блока DTG, который в этом режиме будет включаться по сигналу HALL_EV и формировать сигнал HCRDY по достижении обратным счетчиком единицы.

Захваченная комбинация посредством компараторов сравнивается с комбинациями, находящимися в полях CURH и EXPH. При совпадении с полем текущего значения CURH комбинация считается ошибочной, а обнаруженный перепад сигнала – помехой. В результате логика формирует сигнал CM_WHE – «Неправильная комбинация» и схема переходит в режим ожидания следующего перепада сигнала.

При совпадении с полем ожидаемого значения EXPH комбинация считается правильной и логика формирует сигнал CM_CHE «Правильной комбинации», который в свою очередь инициирует теневую передачу. Поля CURH, EXPH и MCMP обновляются значениями из своих теневых полей CURHS, EXPHS и MCMPS соответственно.

Своевременное обновление теневых полей значениями из предопределенных таблиц должно контролироваться программно.

В случае необходимости теневая загрузка полей может быть инициирована программно установкой битов STRMCM и STRHP регистра CCU6_MCMOUT.

Управление БДПТ

Если включен режим работы с датчиками Холла, то блок таймера T12 автоматически конфигурируется, как показано на рисунке 19.28. Канал 0 функционирует в режиме захвата, а каналы 1 и 2 – в режиме сравнения.

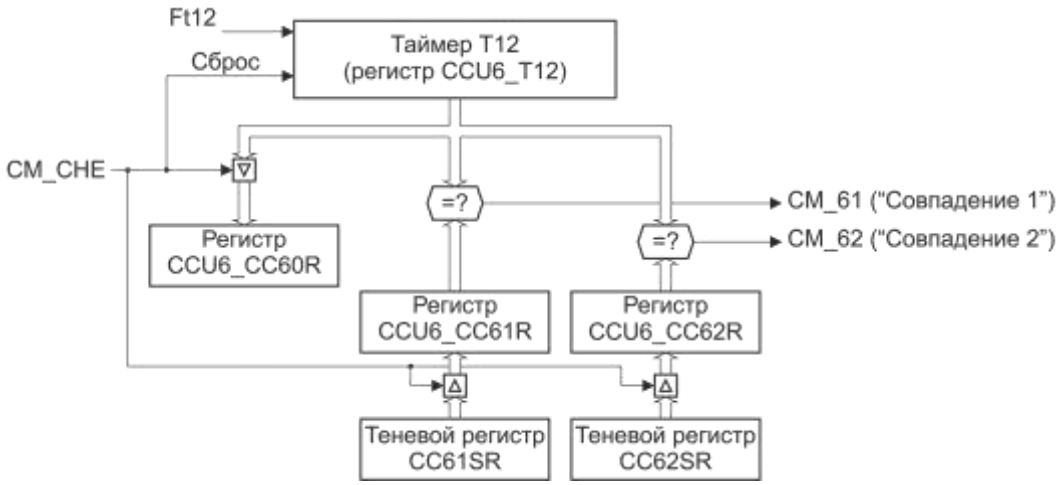


Рисунок 19.28 – Блок таймера T12 в режиме работы с датчиками Холла и управления БДПТ

После того как ожидаемая комбинация состояния датчиков обнаруживается на входах микроконтроллера, формируется сигнал CM_CHE «Правильной комбинации». По этому сигналу содержимое таймера (фактически это показатель скорости вращения ротора БДПТ) захватывается регистром CCU6_CC60R и таймер сбрасывается, регистр периода таймера загружается значением из теневого регистра периода.

Далее таймер инкрементируется и в момент достижения значения регистра CCU6_CC61R формируется сигнал «Совпадение 1 при счете вверх» CM_61_UP, по которому (если SWSEL = 100b, см. рисунок 19.24) управляющие выходы микроконтроллера переключаются в очередное состояние, т. е. выполняется теневая загрузка поля MCMP. Это событие может комбинироваться с другими событиями, которые могут использоваться для фильтрации шумов на линиях датчиков и для синхронизации переключения управляющих выходов.

Канал 2 можно использовать для отслеживания таймаута. Если таймер достигнет значения регистра CCU6_CC61R, то сформируется сигнал «Совпадение 2», что будет информировать о том, что текущая скорость вращения ротора гораздо ниже желаемой, что может быть вызвано превышением нагрузки на валу двигателя. В этом случае генерирование ШИМ-сигнала должно быть приостановлено обнулением битов поля T12MODEN. Пример управления БДПТ с фазовой задержкой переключения управляющих выводов CC6x и COUT6x и уходом в таймаут показан на рисунке 19.29.

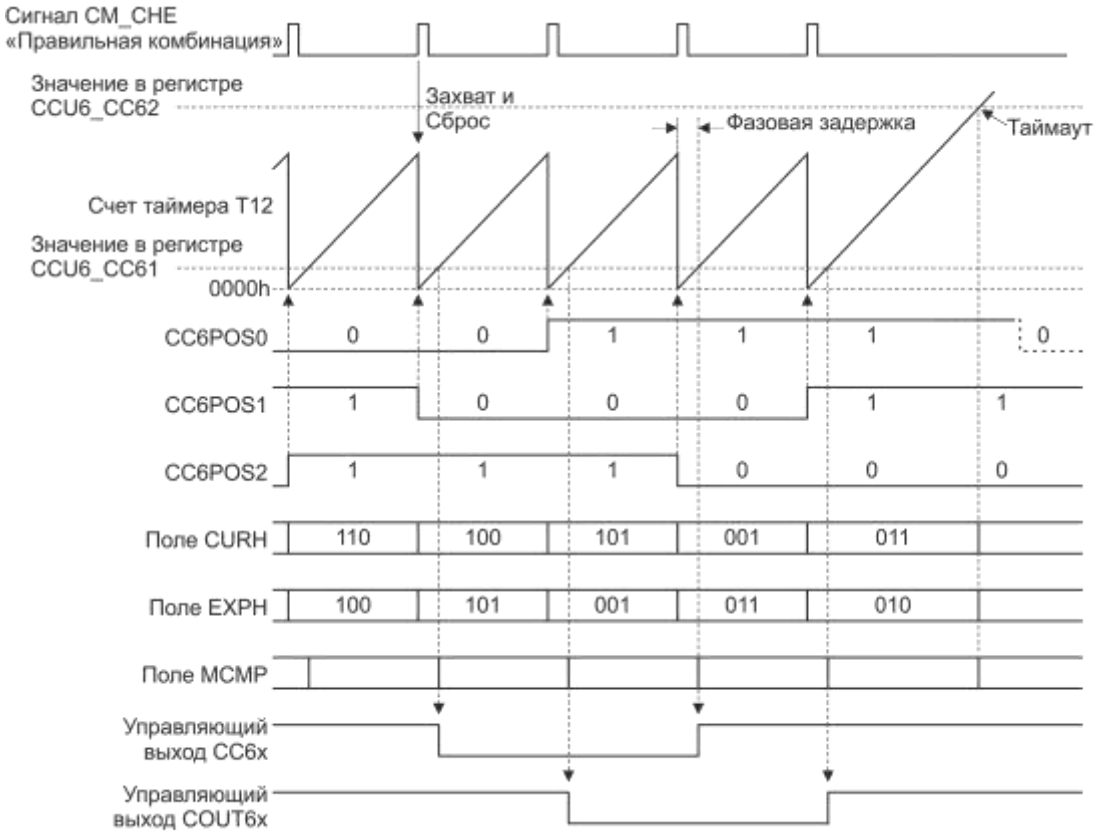


Рисунок 19.29 – Пример управления БДПТ

Примечание – В режиме работы с датчиками Холла аппаратно не генерируется сигнал теневой передачи T12ST, поэтому основные регистры, которым требуется этот сигнал, должны загружаться программно в то время, когда таймер T12 остановлен.

Прерывания в режиме работы с датчиками Холла

Сигналы CM_CHE и CM_WHE могут генерировать прерывания, если это разрешено битами ENCHE и ENWHE регистра CCU6_IEN соответственно, см. рисунки 19.30 и 19.31. Дополнительно, если разрешено битом ENIDLE, сигнал CM_WHE может устанавливать бит IDLE регистра CCU6_IS, что в свою очередь приводит к обнулению поля MCMP, которое является элементом управления выходными сигналами. До тех пор пока бит IDLE остается установленным, значение поля MCMP равно нулю (выходы в нулевом состоянии). Для возврата к нормальной работе флаг IDLE следует сбросить посредством бита RIDLE.

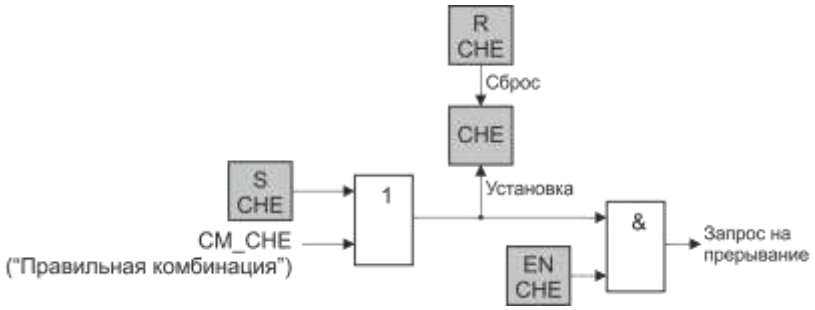


Рисунок 19.30 – Схемы обработки сигнала CM_CHE и генерирования прерывания

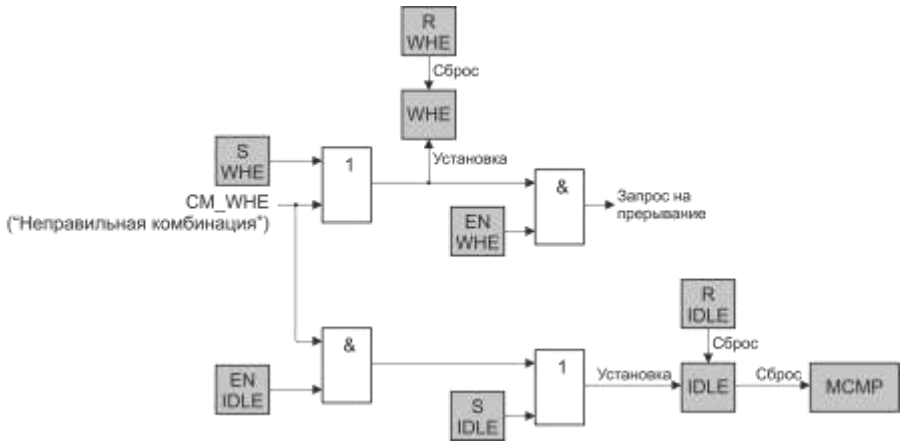


Рисунок 19.31 – Схемы обработки сигнала CM_WHE и генерирования прерывания

Примечание – Следует помнить, что установка флагов CHE и WHE и генерирование соответствующих прерываний происходит параллельно. В связи с этим, нет необходимости программного сброса флагов, поскольку прерывания будут генерироваться даже в том случае, если флаги уже установлены.

19.5 Ловушка

Ловушка представляет собой механизм аппаратного и программного блокирования сигналов в каналах блоков таймеров T12 и T13 в случае возникновения необходимости экстренной остановки работы, см. рисунок 19.32.

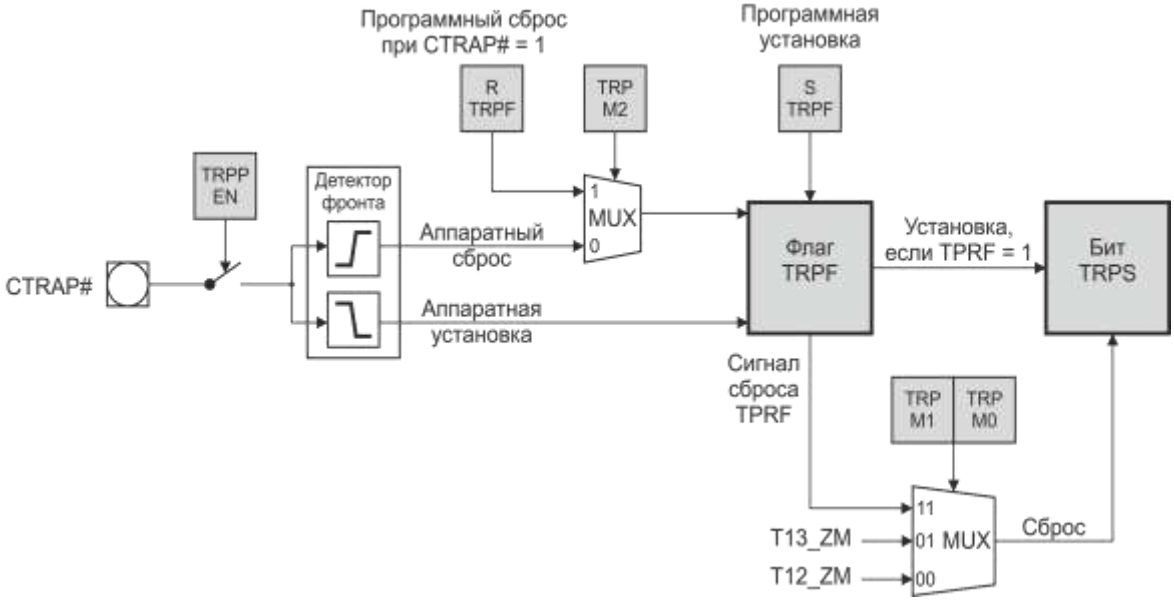


Рисунок 19.32 – Схема установки и сброса флага TRPF и бита TPRS

Аппаратная ловушка управляется битом разрешения TRPPEN регистра CCU6_TRPCTR и сигналом на входе CTRAP# микроконтроллера. Если бит TRPPEN установлен, то появление низкого уровня сигнала на входе CTRAP# устанавливает флаг TPRF и бит TRPS в регистре CCU6_IS, что приводит к незамедлительному попаданию модуля CAPCOM6 в состояние ловушки.

Для программного включения ловушки следует установить флаг TPRF посредством установки бита STRPF регистра CCU6_ISS. После этого автоматически установится бит TPRS, и ловушка включится.

В состоянии ловушки блокирование сигналов CC6x_O и COUT6x_O в каналах блока таймера T12 управляется сигналами TC6x_O и TOUT6x_O, см. рисунок 19.11. Эти сигналы формируются полем TRPEN регистра CCU6_TRPCTR. Каждый бит поля контролирует одну из шести линий, см. рисунок 19.33. В случае попадания в ловушку, блокируются те линии, для которых биты установлены. Заблокированные линии удерживаются в состоянии логического нуля.

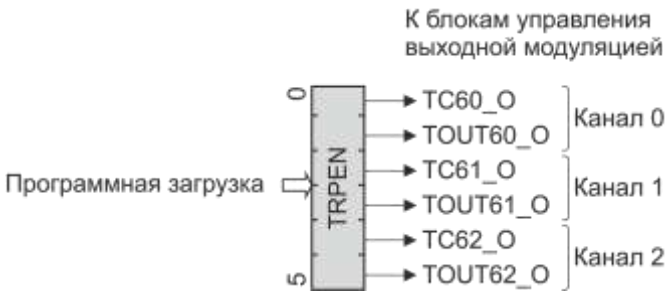


Рисунок 19.33 – Соответствие битов поля TRPEN и линий каналов

Блокирование сигнала COUT63_O в канале блока таймера T13 управляется сигналом TRPEN13, см. рисунок 19.23.

До тех пор, пока на входе CTRAP# будет удерживаться низкий уровень сигнала или биты TRPF и TRPS будут оставаться установленными, модуль CAPCOM6 будет находиться в состоянии ловушки. Для выхода из этого состояния сначала нужно подать высокий уровень сигнала на вход CTRAP# (обязательное условие!).

Сбросом флага TRPF управляет бит TRPM2 регистра CCU6_TRPCTR, который задает тип сброса – программный или аппаратный. Если бит TRPM2 сброшен, то сразу после появления неактивного уровня на входе CTRAP# флаг TRPF будет сброшен. Если бит TRPM2 установлен, то сбросить флаг TRPF можно программно – установкой бита RTPRF регистра CCU6_ISR, но при условии, что сигнал CTRAP# неактивен.

После сброса флага TRPF следует очистить бит TRPS, после чего модуль CAPCOM6 выйдет из состояния ловушки. Сбросом бита TRPS управляют два бита TRPM1 и TRPM0, которые определяют сигнал сброса – обнуление выбранного таймера (T12_ZM или T13_ZM) или сброс флага TPRF, см. рисунок 19.32. Привязка к одному из таймеров является синхронным выходом из ловушки, и потому может пройти некоторое время до сброса бита TRPS. Третий вариант сброса является асинхронным и при условии, что бит TRPM2 сброшен, может обнулить бит TRPS за два такта сигнала Fcc6 после появления на входе CTRAP# логической единицы.

19.6 Прерывания

Модуль CAPCOM6 поддерживает 14 источников аппаратных прерываний. Сигналы источников сгруппированы попарно и подаются на семь узлов прерываний, см. рисунок 19.34. Каждый узел имеет четыре выходные линии, каждая из которых подключена к одному из четырех логических элементов «ИЛИ». Таким образом, каждый из 14 сигналов запросов прерываний может быть выведен на одну из четырех линий I0, I1, I2 и I3, которые соответствуют прерываниям CCU6IRQ, CCU6EIRQ, CCU6T12IRQ, CCU6T13IRQ, см. рисунок 19.34.

Примечание – На рисунке 19.34 дополнительно жирным курсивом отмечены названия линий прерываний, которые являются активными «по умолчанию» после сброса микроконтроллера.

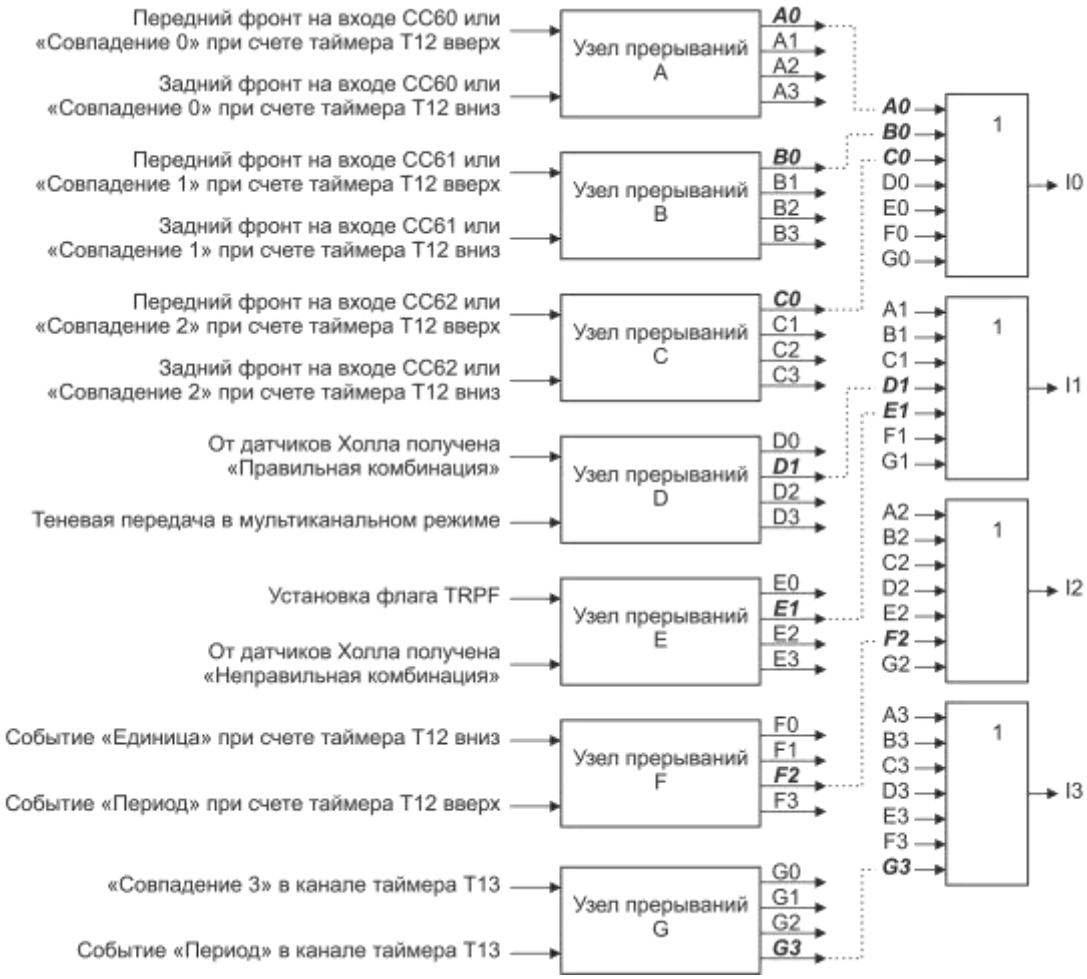


Рисунок 19.34 – Узлы прерываний и обслуживаемые сигналы

Все узлы прерываний идентичны. Функциональная схема узла прерываний показана на рисунке 19.35.

Аппаратное событие устанавливает соответствующий флаг (не сбрасывается аппаратно) и, если разрешено, генерирует прерывание. На выходе узла прерываний расположен демультиплексор, который подключает сигнал прерывания к одному из выходов I0 – I3. Демультиплексор каждого узла управляется полем указателя, расположенного в регистре CCU6_INP.

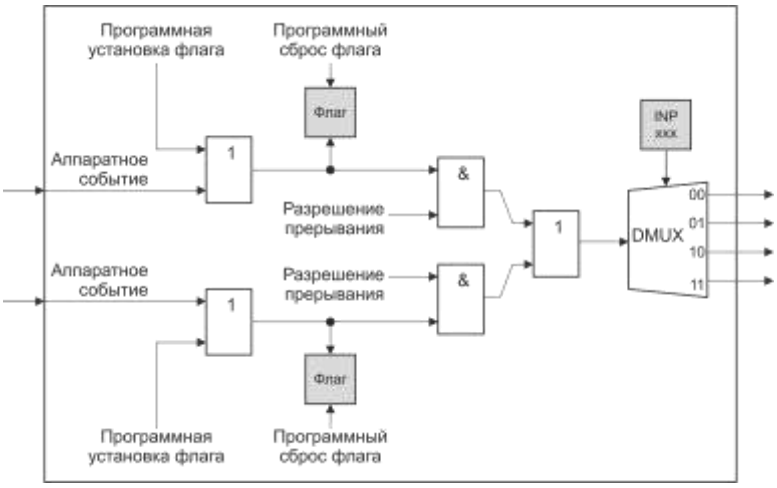


Рисунок 19.35 – Структура узла прерываний

Флаги всех 14 источников прерываний собраны в регистре CCU6_IS, который доступен только для чтения. Биты разрешения прерываний находятся в регистре CCU6_IEN, см. рисунок 19.36.

Помимо аппаратного события флаг может быть установлен программно. Для этого следует установить соответствующий бит в регистре CCU6_ISS. Сбросить флаг можно только программно – установкой соответствующего бита регистра CCU6_ISR. Биты регистров CCU6_ISS и CCU6_ISR сбрасываются аппаратно сразу же после установки/сброса флага. Допускается одновременная установка нескольких флагов. Чтение регистров возвращает нули.

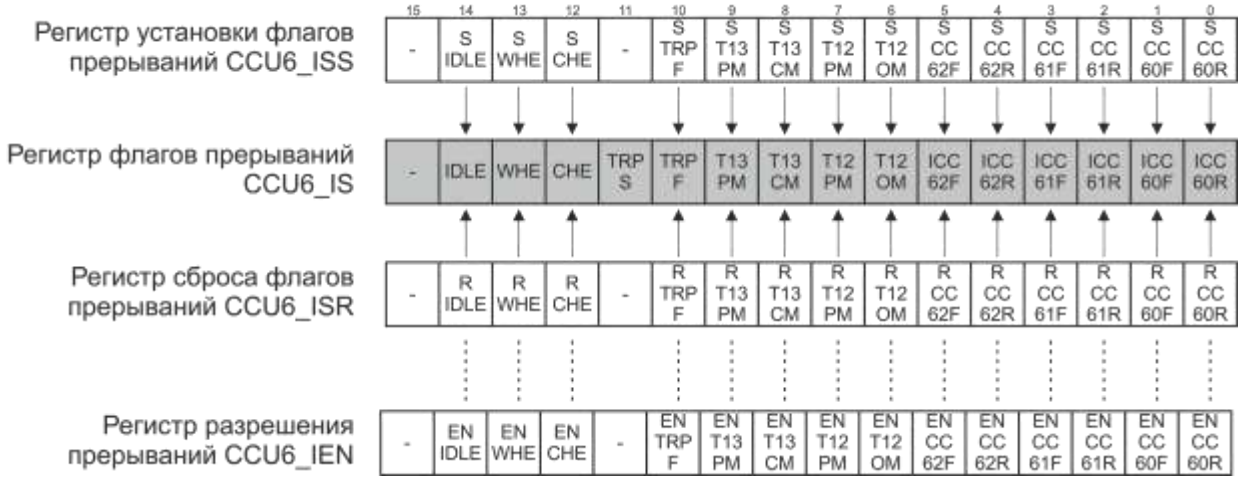


Рисунок 19.36 – Программное управление флагами прерываний

Как при аппаратном событии, так и при появлении сигнала программной установки флага генерируется прерывание. Это прерывание генерируется вне зависимости от состояния флага (т. е. и в том случае, если флаг не был заранее сброшен).

Примечание – В режиме сравнения и режиме работы с датчиками Холла аппаратные прерывания таймера могут генерироваться, только если таймер T12 запущен. В режиме захвата аппаратные прерывания могут генерироваться также и во время, когда таймер T12 остановлен.

В таблице 19.3 указаны прерывания модуля CAPCOM6 и соответствующие управляющие регистры.

Таблица 19.3 – Соответствие прерываний и управляющих регистров

Выход	Прерывание	Управляющий регистр прерывания
I0	CCU6IRQ	CCU6_IC
I1	CCU6EIRQ	CCU6_EIC
I2	CCU6T12IRQ	CCU6_T12IC
I3	CCU6T13IRQ	CCU6_T13IC

19.7 Справочная информация о регистрах и теневых передачах

Основной регистр – регистр, всегда доступный только для чтения, в который данные записываются по сигналу теневой передачи.

Теневой регистр – регистр, всегда доступный только для записи, в который данные записываются при записи по адресу основного регистра.

Наличие теневых регистров позволяет синхронизировать работу всех узлов блоков таймеров, генерировать ШИМ сигнал и параллельно программно задавать новые параметры модуляции.

По сигналу теневой передачи информация из теневого регистра загружается в основной. Используются несколько сигналов теневых передач. Генератором сигнала/сигналов могут быть как аппаратные события в модуле CAPCOM6, так и программные – установка управляющих битов. Суммарная справочная информация о регистрах и источниках теневых передач находится в таблице 19.4.

Таблица 19.4 – Общая информация об основных и теневых регистрах

<table>
<tr><td rowspan="2">Основной регистр</td><td colspan="2">Способ активации теневой передачи</td><td rowspan="2">Теневой регистр/биты</td></tr>
<tr><td>Аппаратное генерирование сигнала</td><td>Программная установка бита</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td></tr>
<tr><td>CCU6_T12PR</td><td>T12_ST</td><td rowspan="6">T12STE (посредством записи единицы в бит T12STR регистра CCU6_TCTR4)</td><td>T12PRS</td></tr>
<tr><td>CCU6_CC60R</td><td rowspan="3">T12_ST, CM_CHE, сигналов в режимах захвата</td><td>CC60SR</td></tr>
<tr><td>CCU6_CC61R</td><td>CC61SR</td></tr>
<tr><td>CCU6_CC62R</td><td>CC62SR</td></tr>
<tr><td>CCU6_CMPSTAT (биты COUT62PS, COUT61PS, COUT60PS, CC62PS, CC61PS, CC60PS)</td><td rowspan="2">T12_ST</td><td>Биты COUT62PS, COUT61PS, COUT60PS, CC62PS, CC61PS, CC60PS</td></tr>
<tr><td>CCU6_PSLR (поле PSL)</td><td>Поле PSL</td></tr>
<tr><td>CCU6_CMPSTAT (биты COUT63PS и T13IM)</td><td rowspan="4">T13_ST</td><td rowspan="4">T13STE (посредством записи единицы в бит T13STR регистра CCU6_TCTR4)</td><td>Биты COUT63PS и T13IM</td></tr>
<tr><td>CCU6_T13PR</td><td>T13RPS</td></tr>
<tr><td>CCU6_CC63R</td><td>CC63SR</td></tr>
<tr><td>CCU6_PSLR (бит PSL63)</td><td>Бит PSL63</td></tr>
<tr><td>CCU6_MCMOUT (поля CURH, EXPH)</td><td>CM_CHE</td><td>STRHP регистра CCU6_MCMOUTS</td><td>CCU6_MCMOUTS (поля CURHS и EXPHS)</td></tr>
<tr><td>CCU6_MCMOUT (поле MCMP)</td><td>MCM_ST</td><td>STRMCM регистра CCU6_MCMOUTS</td><td>CCU6_MCMOUTS (поле MCMPS)</td></tr>
</table>

Сигнал теневой передачи T12_ST

Сигнал T12_ST генерируется автоматически, как только устанавливается бит STE12 регистра CCU6_TCTR0. Этот бит может устанавливаться как аппаратно, так и программно.

Если сброшен бит T12STD (разрешение теневой передачи) регистра CCU6_TCTR4, то аппаратная установка бита STE12 происходит в случае:

- достижения счетчиком таймера T12 значения регистра периода CCU6_T12PR при счете вверх (событие «Период»);
- достижения счетчиком таймера T12 значения 0001h при счете вниз (событие «Единица»).

Теневая передача возможна также и в случае, если счетчик таймера T12 остановлен; например, при записи в регистр периода таймера значения, которое совпадет с текущим значением счетчика.

После выполнения теневой передачи бит STE12 сбрасывается автоматически.

Примечание – В режиме работы с датчиками Холла установки бита STE12 не происходит независимо от состояния бита T12STD.

Программно бит STE12 всегда может быть установлен (независимо от состояния бита T12STD) посредством записи единицы в бит T12STR регистра CCU6_TCTR4. Если на момент установки бита STE12 счетчик таймера T12 включен, то аппаратная часть будет ожидать очередного переключения счетчика, чтобы выполнить синхронную теневую передачу. Если счетчик таймера остановлен, то теневая передача будет выполнена сразу же. После выполнения теневой передачи бит STE12 сбросится автоматически.

Установленный бит T12STD запрещает аппаратную установку бита STE12 при возникновении событий «Период» и «Единица» и, как следствие, аппаратную теневую передачу. В то же время он не препятствует программной установке бита STE12, но если во время ожидания теневой передачи записать единицу в бит T12STD (даже если он уже установлен), то это приведет к сбросу бита STE12 и отмене теневой передачи.

Примечание – Запись единицы в бит T12STD всегда сбрасывает бит STE12.

Сигнал теневой передачи T13_ST

Сигнал T13_ST генерируется автоматически, как только аппаратно или программно устанавливается бит STE13.

Если сброшен бит T13STD, то аппаратная установка бита STE13 происходит в случае достижения счетчиком таймера T13 значения регистра периода CCU6_T13PR (событие «Период»).

Теневая передача возможна также и в случае, если счетчик таймера T13 остановлен; например, при записи в регистр периода таймера значения, которое совпадет с текущим значением счетчика.

После выполнения теневой передачи бит STE13 сбрасывается автоматически.

Программно бит STE13 всегда может быть установлен (независимо от состояния бита T13STD) посредством записи единицы в бит T13STR. Если на момент установки бита STE13 счетчик таймера T13 включен, то аппаратная часть будет ожидать очередного переключения счетчика, чтобы выполнить синхронную теневую передачу. Если счетчик таймера остановлен, то теневая передача будет выполнена сразу же. После выполнения теневой передачи бит STE13 сбросится автоматически.

Установленный бит T13STD запрещает аппаратную установку бита STE13. В то же время он не препятствует программной установке бита STE13, но если во время ожидания теневой передачи записать единицу в бит T13STD (даже если он уже установлен), то это приведет к сбросу бита STE13 и отмене теневой передачи.

Примечание – Запись единицы в бит T13STD всегда сбрасывает бит STE13.

20 Контроллер интерфейса ГОСТ Р 52070-2003

Модуль представляет собой устройство, поддерживающее обмен данными с другими устройствами (контроллерами) посредством магистрального последовательного интерфейса (ГОСТ Р 52070-2003), совместимого со стандартом MIL-STD-1553B на частоте 1 МГц.

На физическом уровне интерфейс представляет собой последовательную шину данных (экранированная витая пара), к которой подключены устройства. Допустимыми устройствами являются: контроллер шины, монитор шины и удаленные терминалы (оконечные устройства).

Контроллер шины является ведущим устройством. Он единственный инициирует любой обмен информацией и контролирует работу сети. Контроллер шины может обращаться к любому из удаленных терминалов (максимальное количество 31), каждому из которых присвоен уникальный 5-битный адрес. Монитор шины – пассивное устройство, подключенное к шине данных и занимающееся только отслеживанием и записью передаваемой по шине информации.

Для получения более подробной информации о протоколе следует обратиться к ГОСТ Р 52070–2003.

В микроконтроллере реализованы два идентичных блока поддержки магистрального последовательного интерфейса: нулевой M0 и первый M1, см. рисунок 20.1.

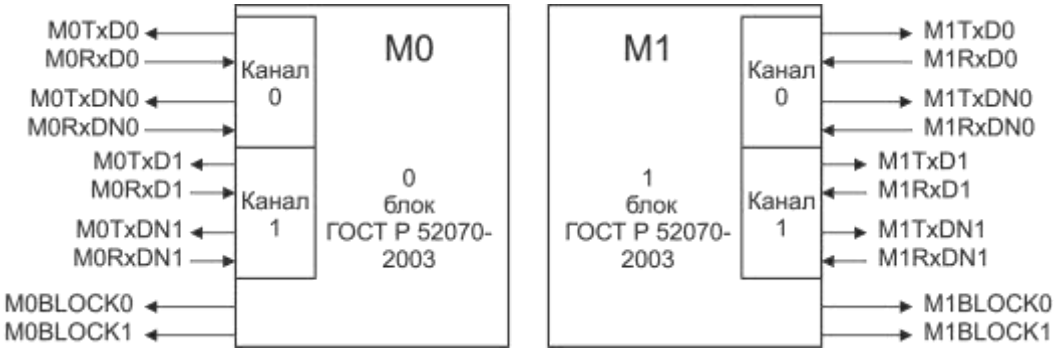


Рисунок 20.1 – Блоки ГОСТ Р 52070-2003

Блок Mn имеет канал 0 и канал 1 одинаковой функциональности с соответствующими прямыми выводами приема-передачи MnTxD0, MnRxD0, MnTxD1, MnRxD1 и инверсными MnTxDN0, MnRxDN0, MnTxDN1, MnRxDN1, а также два выхода блокировки каналов MnBLOCK0 и MnBLOCK1 (n в обозначении – номер блока 0 или 1). Блок может функционировать в одном из трех режимов:

- контроллер шины (КШ);
- удаленный терминал (оконечное устройство);
- монитор шины (МШ).

Установка режима, а также других дополнительных параметров работы осуществляется посредством регистра конфигурации BSICONFIG.

20.1 Контроллер шины

Контроллер шины инициирует любой обмен информацией в сети и контролирует работу сети. Контроллер шины может обращаться к любому из удаленных терминалов по его адресу, посылать и принимать как одиночные сообщения, так и последовательности сообщений.

Для организации последовательности сообщений используются области ОЗУ с адресами от 0D’8000h до 0D’8FFFh (для блока M0) и от 0D’9000h до 0D’9FFFh (для

блока М1), в которых формируются блоки данных для каждого сообщения. Каждый блок данных состоит из управляющих слов и слов данных, количество и порядок размещения которых зависит от формата сообщения, см. таблицу 20.6.

Типы слов:

- управляющее слово (записывается пользователем и содержит параметры управления сообщением);
- слово указатель (записывается пользователем и содержит адрес управляющего слова следующего блока);
- статус окончания сообщения (записывается аппаратно по окончании сообщения);
- ответное слово (записывается аппаратно после принятия второго ответного слова в формате 3);
- командное слово (записывается пользователем);
- слова данных (слова данных для передачи записываются пользователем, принятые слова данных записываются аппаратно).

Форматы и назначение битов слов приведены в таблицах 20.1 – 20.4.

Пример размещения двух блоков данных разных форматов в ОЗУ блока М0 приведен на рисунке 20.2.

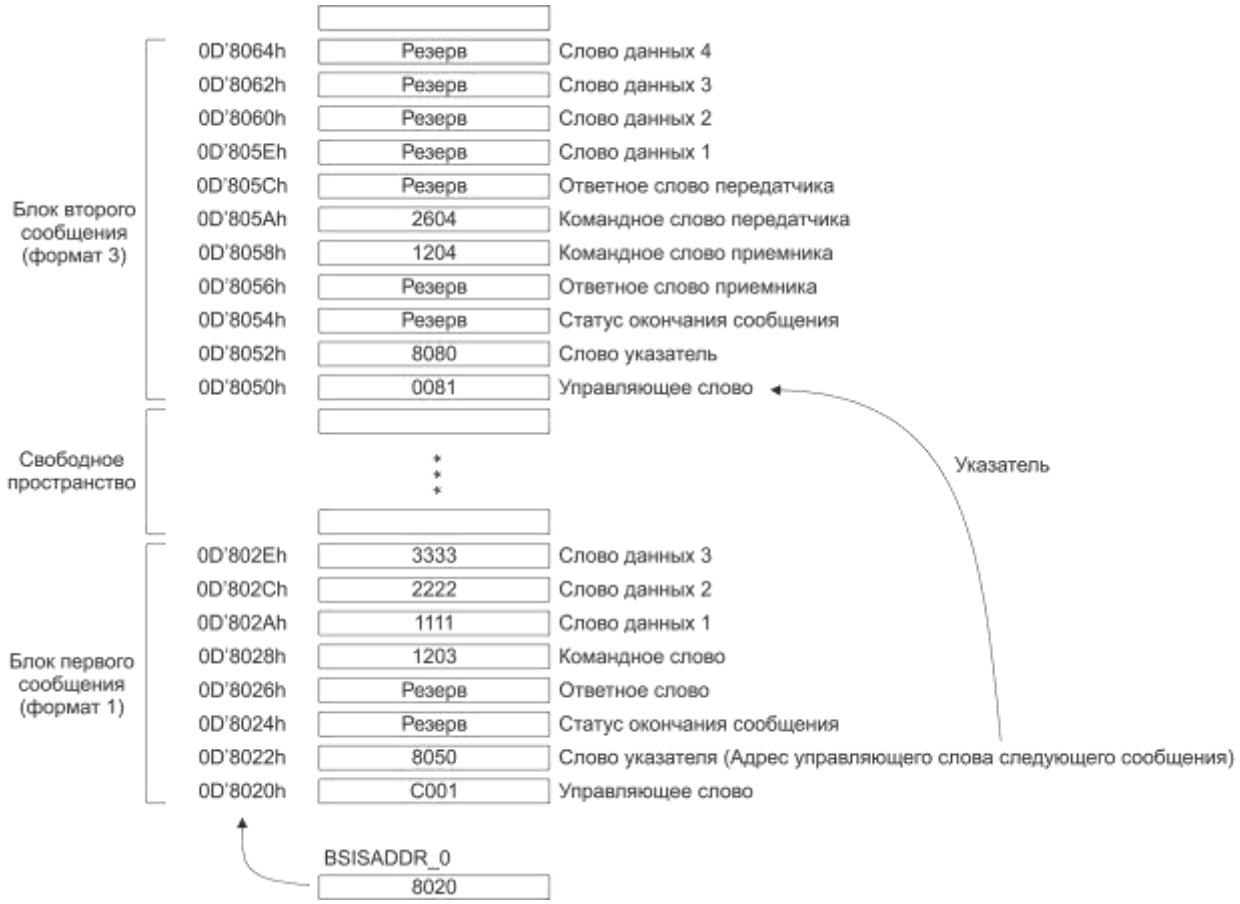
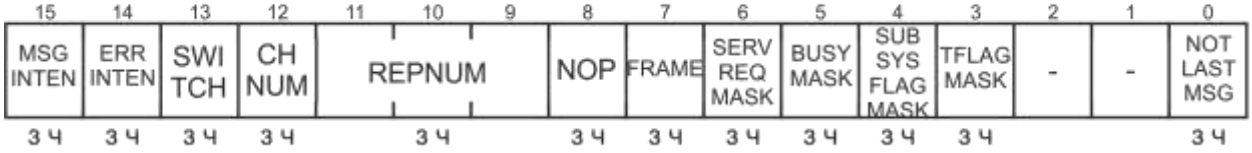


Рисунок 20.2 – Размещения двух блоков данных в ОЗУ

Порядок и последовательность размещения блоков сообщений в пределах памяти не важны, поскольку все они связаны посредством указателей.

Таблица 20.1 – Назначение битов управляющего слова

Управляющее слово (УС)

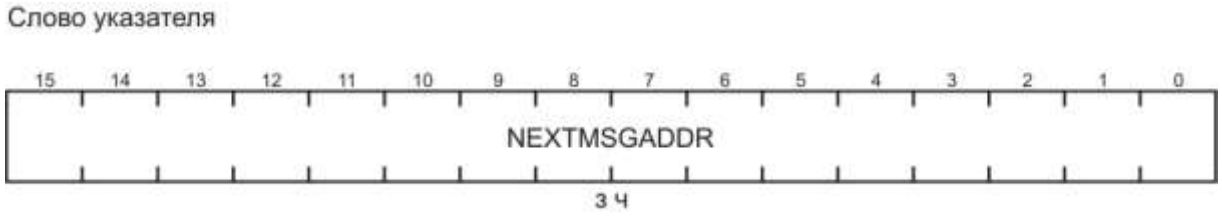


Поле	Бит	Описание		
1	2	3		
MSGINTEN	15	Бит разрешения прерывания. Если бит установлен, то после безошибочного окончания сообщения будет сформирован запрос на прерывание, если только установлен бит USUAL_INT_EN в регистре BSICON		
ERRINTEN	14	Бит разрешения прерывания по ошибке. Если бит установлен, то после окончания сообщения по ошибке будет сформирован запрос на прерывание, если только установлен бит ERR_INT_EN в регистре BSICON		
SWITCH	13	Бит переключения канала при ошибке		
		0	Канал передачи/приема не меняется при повторной передаче	
		1	Канал передачи/приема меняется после каждого повторения (если установлен бит JUMPEN в регистре BSICON)	
CHNUM	12	Выбор канала для передачи/приема		
		0	Основной	
		1	Резервный	
REPNUM	11-9	Поле количества повторений. Если во время передачи была обнаружена ошибка, сообщение может быть передано повторно. Поле REPNUM устанавливает, следует ли повторить попытку передачи или нет, а также количество повторов		
		000	Повтор запрещен	Разрешено, если установлен бит REPEN регистра BSICON
		001	1	
		010	2	
		011	3	
		100	4	
		101	5	
		110	6	
		111	Повтор передачи до тех пор, пока сообщение не будет передано без ошибок (не будет ошибок в словах данных и установленных битов ошибок в ответном слове)	
NOP	8	Нет операции. Сообщение, в управляющем слове которого обнаружен установленный бит NOP, не обслуживается, а указатель адреса переключается на адрес, по которому расположено управляющее слово следующего сообщения		

Окончание таблицы 20.1

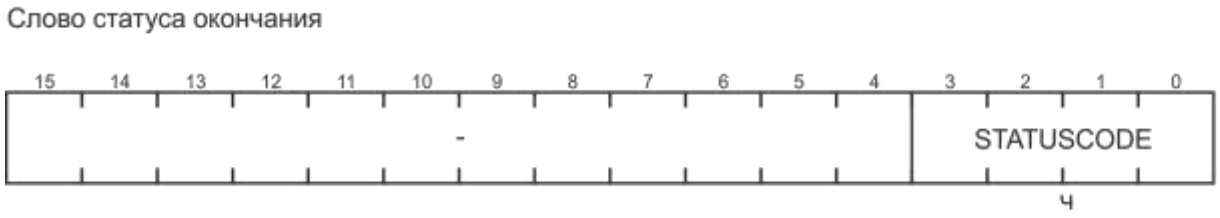
1	2	3	
FRAME	7	Указатель формата сообщения	
		0	Используется любой формат, за исключением форматов 3 и 8
		1	Используется формат 3 или формат 8
SERVREQMASK	6	Маска бита SERVREQ ответного слова	
BUSYMASK	5	Маска бита BUSY ответного слова	
SUBSYSFLAG MASK	4	Маска бита SUBSYSFLAG ответного слова	
TFLAGMASK	3	Маска бита TFLAG ответного слова	
NOTLASTMSG	0	Индикатор последнего сообщения. Сообщение, в управляющем слове которого этот бит не установлен, является последним в цепочке сообщений	
–	2, 1	Зарезервировано	
Примечание – Если бит маски установлен, то маска считается включенной. Бит ответного слова, закрытый соответствующей маской, считается сброшенным.			

Таблица 20.2 – Назначение битов слова указателя



Поле	Бит	Описание
NEXTMSGADDR	15-0	Адрес управляющего слова следующего сообщения

Таблица 20.3 – Назначение битов слова статуса окончания сообщения

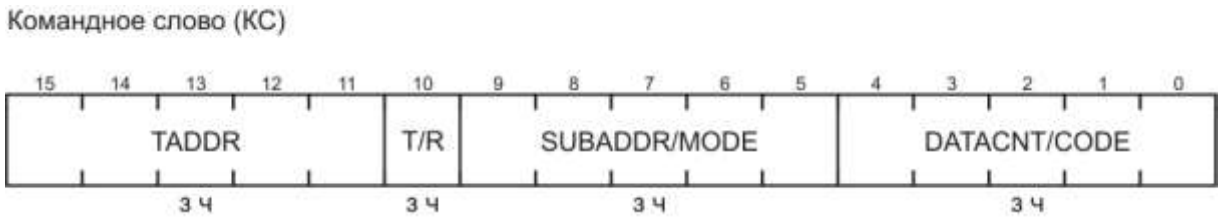


Поле	Бит	Описание	
1	2	3	
STATUS CODE	3-0	Код статуса окончания сообщения	
		0000	Безошибочное окончание последовательности
		0001	Безошибочное окончание одного сообщения
		0010	Генерация в канале 0
		0011	Генерация в канале 1
		0100	Ошибка контроля передачи (принято не то, что передано)
		0101	Неправильный адрес в ответном слове
		0110	Неправильный синхроимпульс в ответном слове

Окончание таблицы 20.3

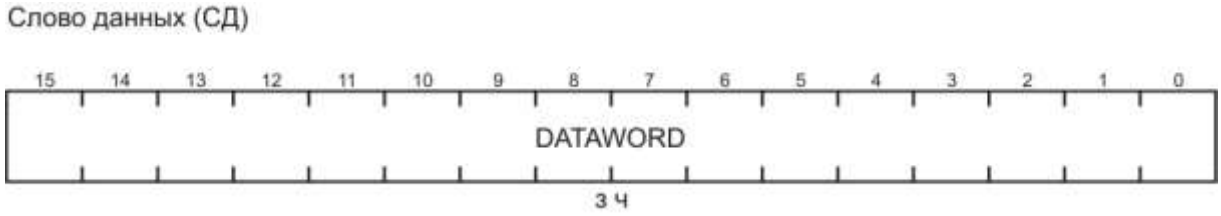
1	2	3	
STATUS CODE	3-0	0111	Ненулевое ответное слово
		1000	Ошибка манчестерского кода в ответном слове
		1001	Отсутствие ответного слова
		1010	Неправильный синхроимпульс в слове данных
		1011	Ошибка манчестерского кода в слове данных
		1100	Отсутствие слова данных или нарушение непрерывности при приеме слова данных
		1101	-
		1110	-
		1111	Неверное командное слово (недопустимая комбинация бит)
–	15-4	Зарезервировано	

Таблица 20.4 – Назначение битов командного слова



Поле	Бит	Описание	
TADDR	15-11	Адрес оконечного устройства. В поле записывается адрес оконечного устройства, к которому обращается контроллер шины. Значение 1Fh является групповым адресом и служит для обращения ко всем оконечным устройствам одновременно.	
T/R	10	Направление передачи данных	
		1	Передача
		0	Прием
SUBADDR/ MODE	9-5	Подадрес/Управление	
		00h или 1Fh	Режим «Управление». Если в поле записано 00h или 1Fh , то значение в поле DATACNT/CODE является кодом команды управления
		01h–1Eh	Режим «Подадрес». В поле находится адрес подчиненного устройства (абонента), подключенного непосредственно к выбранному удаленному терминалу. В этом случае в поле DATACNT/CODE указывается количество слов данных для передачи/приема
DATACNT/ CODE	4-0	Количество данных/Код команды управления. Режим работы битового поля устанавливается полем SUBADDR/ MODE	

Таблица 20.5 – Назначение битов слова данных



Поле	Бит	Описание
DATAWORD	15-0	Данные

Инициализация

Для запуска последовательности сообщений необходимо:

- записать 01b в поле MODE (режим КШ);
- сформировать блоки сообщений и занести адрес управляющего слова первого блока в регистр BSISADDR (в примере на рисунке 20.2 стартовый адрес – 08E8h);
- в регистре BSISPACE установить паузу между сообщениями в микросекундах (по умолчанию 4 мкс);
- инициализировать передачу установкой бита START в регистре BSICON.

После этого запись в регистр BSICONFIG будет заблокирована и на аппаратном уровне выполнятся следующие действия:

- чтение управляющего слова сообщения, на которое указывает регистр BSISADDR;
- проверка состояния бита NOP управляющего слова и переход к управляющему слову следующего сообщения (если бит NOP установлен) или чтение командного слова сообщения и передача его в выбранный канал, заданный битом CHNUM (если бит NOP сброшен);
- передача/получение всех остальных слов сообщения в порядке, установленном форматом сообщения.

После успешного окончания передачи одного сообщения проверяется состояние бита NOTLASTMSG управляющего слова сообщения и:

- если бит установлен, то выполняется переход к управляющему слову следующего сообщения цепочки и далее по описанному выше алгоритму;
- если бит не установлен, то передача сообщений завершается, бит START сбрасывается и модуль переходит в режим ожидания.

В случае обнаружения ошибки проверяются состояния бита REPEN регистра BSICON. Если бит установлен, и значение поля REPNUM управляющего слова больше 0, осуществляется повтор передачи сообщения указанное количество раз или до тех пор, пока не будет выполнена передача без ошибок. При этом если установлены бит SWITCHEN в регистре BSICON и бит SWITCH в управляющем слове, при каждом повторе сообщения будет происходить переключение текущего канала на другой.

Прерывания

Управление прерываниями осуществляется с помощью битов USUAL_INT_EN и ERR_INT_EN регистра BSICON, а также MSGINTEN и ERRINTEN в управляющем слове каждого сообщения.

Если установлен бит USUAL_INT_EN, то прерывания будут происходить в конце последовательности сообщений, а также после безошибочного окончания сообщений, в управляющем слове которых установлен бит MSGINTEN.

Если установлен бит ERR_INT_EN регистра BSICON, то прерывания будут происходить после сообщений, завершенных с ошибкой, и в управляющем слове которых установлен бит ERRINTEN. На прерывание в конце последовательности сообщений этот бит не влияет.

При сброшенных битах USUAL_INT_EN и ERR_INT_EN прерываний не будет.

Для определения причины прерывания используется регистр BSISTAT, где сохраняется код причины прерывания и данные о последней переданной команде. Также о результате каждого сообщения можно узнать из слова статуса окончания сообщения в блоке сообщения.

Форматы сообщений

Форматы делятся на две группы – форматы основных сообщений и форматы групповых сообщений (по ГОСТ Р 52070–2003). Ниже представлены 10 допустимых форматов сообщений, с указанием порядка размещения слов и резервирования пространства в памяти при формировании блоков сообщений, см. таблицы 20.6 и 20.7.

Форматы основных сообщений

Применяются для передачи информации, предназначенной одному из оконечных устройств с получением от него соответствующего ответа. Описания форматов приведено в таблице 20.6.

Таблица 20.6 – Форматы основных сообщений

Название формата	Алгоритм передачи	Порядок размещения слов в блоке сообщения
1	2	3
Формат 1	Передача данных от контроллера шины к оконечному устройству	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова. 5 Командное слово. 6-37 Последовательно все слова данных (не более 32 слов).
Формат 2	Передача данных от оконечного устройства к контроллеру шины	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова. 5 Командное слово. 6-37 Резерв для слов данных (согласно запросу).
Формат 3	Передача данных от оконечного устройства к оконечному устройству	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова от приемника. 5 Командное слово для приемника. 6 Командное слово для передатчика. 7 Резерв для ответного слова от передатчика. 8-39 Резерв для слов данных (согласно запросу).
Формат 4	Передача команды управления от контроллера шины к оконечному устройству	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова. 5 Командное слово.
Формат 5	Передача команды управления от контроллера шины к оконечному устройству и получение от него слова данных	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова. 5 Командное слово. 6 Резерв для слова данных.

Окончание таблицы 20.6

1	2	3
Формат 6	Передача команды управления и одного слова данных от контроллера шины к оконечному устройству	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Резерв для ответного слова. 5 Командное слово. 6 Слово данных.

Форматы групповых сообщений

Применяются для передачи информации, предназначенной нескольким оконечным устройствам без получения от них ответных слов, см. таблицу 20.7. Сообщение является групповым, если в нем адрес удаленного терминала равен 11111b. Каждый удаленный терминал, который принял команду общего вызова, после ее обнаружения устанавливает соответствующий флаг в ответном слове, но само ответное слово не передает.

Таблица 20.7 – Форматы групповых сообщений

Название формата	Алгоритм передачи	Порядок размещения слов в блоке сообщения
Формат 7	Передача данных (в групповом сообщении) от контроллера шины к оконечным устройствам	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Не используется. 5 Командное слово. 6-37 Последовательно все слова данных (не более 32 слов).
Формат 8	Передача данных (в групповом сообщении) от оконечного устройства к оконечным устройствам	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Не используется. 5 Командное слово для приемника. 6 Командное слово для передатчика. 7 Резерв для ответного слова от передатчика. 8-39 Резерв для слов данных (согласно запросу).
Формат 9	Передача групповой команды управления от контроллера шины к оконечным устройствам	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Не используется. 5 Командное слово.
Формат 10	Передача групповой команды управления и одного слова данных от контроллера шины к оконечным устройствам	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Резерв для слова статуса окончания сообщения. 4 Не используется. 5 Командное слово. 6 Слово данных.

Команды управления

Командами управления являются командные слова с кодами 00000b или 11111b в поле SUBADDR/MODE. Поле DATACNT/CODE команды управления является кодом команды. Команды управления применяются только для управления удаленными терминалами (не для обмена данными!). Наличие команд управления позволяет

контроллеру шины не только контролировать работу сети, но и обрабатывать, и исправлять обнаруженные ошибки.

Команды управления с кодами 00h – 08h применяются без слов данных, а с кодами 10h – 15h – с одним словом данных.

Возможность использования той или иной команды управления в групповых сообщениях и состояние бита T/R указаны в таблице 20.8.

Таблица 20.8 – Команды управления, передаваемые в командном слове

Код команды	Название команды управления	Бит T/R	Возможность применения		Прерывание*
			в групповых сообщениях	со словом данных	
01h	Синхронизация	1	Да	Нет	Да
02h	Передать ответное слово	1	Нет		Нет
03h	Начать самоконтроль	1	Да		Да
04h	Блокировать передатчик	1	Да		Да
05h	Разблокировать передатчик	1	Да		Да
06h	Блокировать признак неисправности удаленного терминала	1	Да		Нет
07h	Разблокировать признак неисправности удаленного терминала	1	Да		Нет
08h	Установить удаленный терминал в исходное состояние	1	Да		Нет
10h	Передать векторное слово	1	Нет	Да	Да
11h	Синхронизация со словом данных	0	Да		Да
12h	Передать последнюю команду	1	Нет		Нет
13h	Передать слово встроенной системы контроля (ВСО) удаленного терминала к контроллеру шины	1	Нет		Да
14h	Блокировать выбранный передатчик	0	Да		Да
15h	Разблокировать выбранный передатчик	0	Да		Да
00h, 09h-0Fh, 16h-1Fh	Зарезервировано. Не использовать!				

* В случае получения управляющей команды удаленный терминал не формирует запрос на прерывание, а только аппаратно выполняет предписанное командой действие.

20.2 Удаленный терминал

Удаленный терминал (оконечное устройство) выполняет команды КШ. Каждому удаленному терминалу присваивается адрес в диапазоне от 00h до 1Eh.

Объем области ОЗУ микроконтроллера, которая выделяется каждому блоку магистрального последовательного интерфейса во время его работы в режиме удаленного терминала, составляет 4 Кбайт. Области расположены в диапазоне адресов от 0D’8000h до 0D’8FFFh (для блока М0) и от 0D’9000h до 0D’9FFFh (для блока М1), и имеют условные названия – память сообщений М0 и память сообщений М1. Если один из блоков не используется, его область сообщений может использоваться как обыкновенное ОЗУ.

В отличие от режима контроллера шины, в режиме удаленного терминала память сообщений имеет четкую структуру, и все ее пространство разделено на 64 блока данных, каждый из которых охватывает 32 слова.

Половины областей сообщений М0 и М1 с адресами 0D’8000h – 0D’87FFh и 0D’9000h – 0D’97FFh выделены для записи и хранения принятых слов данных, а вторые половины – для хранения слов данных, записанных пользователем и предназначенных для передачи. Структура области показана на рисунке 20.3.

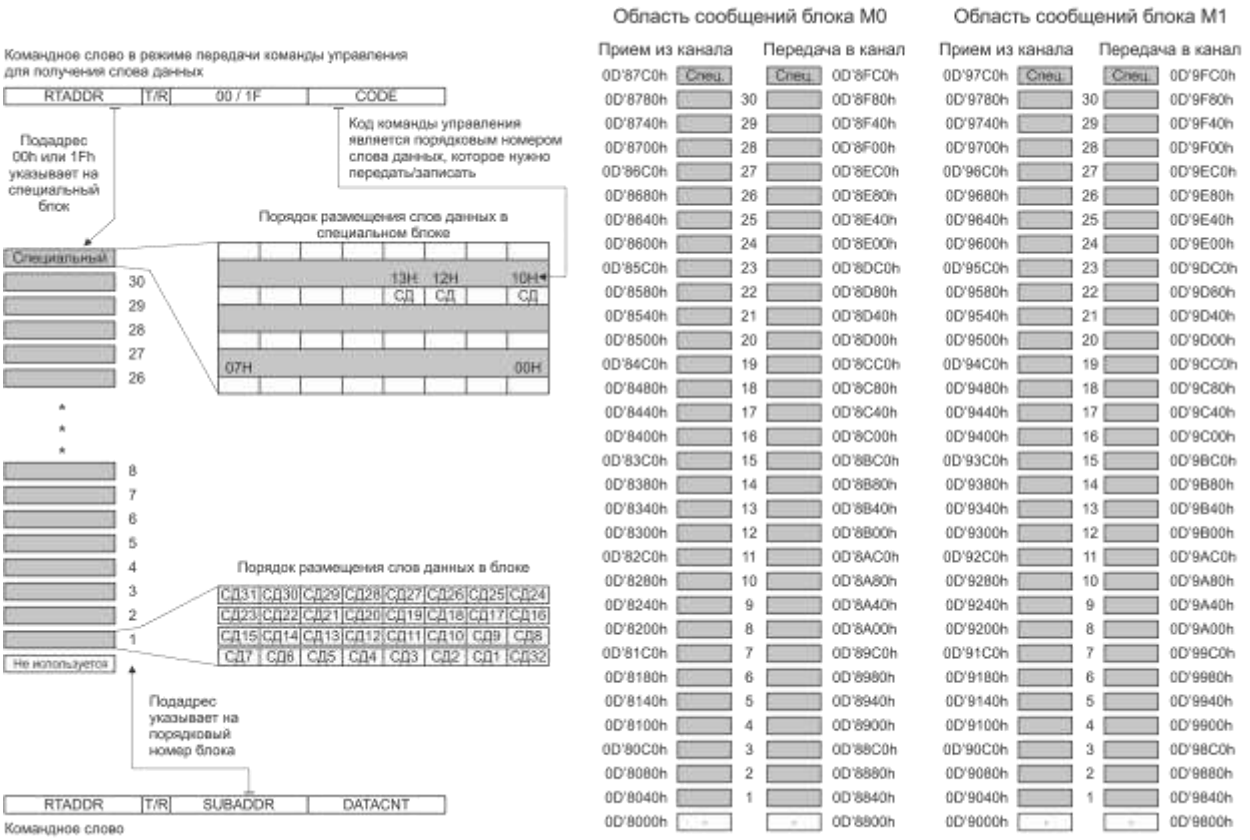


Рисунок 20.3 – Структура области памяти сообщений и разделение ОЗУ на области

Пояснение к рисунку 20.3

Область памяти сообщений поделена на 32 блока, каждый со своим порядковым номером от 0 до 31. Нулевой блок не используется. 31-й блок данных является специальным. Остальные 30 блоков могут заполняться данными. Структура всех блоков идентична. В пределах блока располагаются 32 слова данных.

Прием данных

Получив командное слово, удаленный терминал анализирует его.

T/R = 0 указывает на то, что нужно принять данные.

SUBADDR указывает на номер блока, в который следует записать слова данных.

DATACNT указывает, какое количество данных будет передано. DATACNT = 00h означает, что будут переданы 32 слова данных.

Принимаемые слова данных записываются в блок последовательно, начиная со второй ячейки, условно обозначенной СД1, а после – в СД2, СД3 и т. д. 32-е слово данных записывается в первую ячейку блока, обозначенную СД32. Если значение поля SUBADDR/MODE 00h или 1Fh, то это указывает на то, что в поле DATACNT/CODE находится управляющая команда. В этом случае вместе с командным словом может быть получено и слово данных, которое должно быть записано в память сообщений. Для таких слов данных резервируется один блок, называемый специальным. В пределах этого блока каждая ячейка имеет свой порядковый номер от 00h до 1Fh, на который указывает код полученной команды управления. Слово данных, принятое после командного слова, записывается в указанную ячейку, см. таблицу 20.9.

Таблица 20.9 – Соответствие кодов команд управления и адресов ОЗУ

Код команды управления, с которой передается одно слово данных	Физический адрес ячейки памяти, в которую записывается принятое слово данных
11h	0FE2h
14h	0FE8h
15h	0FEAh

Передача данных

Первым передается слово из ячейки СД1, а после из СД2, СД3 и т. д. Из ячейки СД32 слово данных будет передано последним.

Получив командное слово, удаленный терминал анализирует его.

T/R = 1 указывает на то, что нужно передать данные.

SUBADDR указывает на номер блока, из которого следует прочитать передаваемые данные.

DATACNT указывает, какое количество данных требуется прочитать. Так DATACNT = 0h означает, что требуется прочитать все 32 слова данных.

Если значение поля SUBADDR/MODE 00h или 1Fh, то это указывает на то, что в поле DATACNT/CODE находится управляющая команда. В этом случае командное слово может являться запросом, по которому должно быть отправлено слово данных. Для таких слов данных резервируется один блок, называемый специальным. В пределах этого блока каждая ячейка имеет свой порядковый номер от 00h до 1Fh. Код полученной команды управления и будет указывать на ячейку, данные из которой следует передать.

Специальные данные, предназначенные для передачи, должны быть предварительно записаны в память так же, как обыкновенные слова данных, см. таблицу 20.10.

Таблица 20.10 – Соответствие кодов команд управления и адресов ОЗУ

Код команды управления, запрашивающей специальное слово данных	Физический адрес ячейки памяти, из которой будет прочитано слово данных
10h	17E0h
12h	17E4h
13h	17E6h

В режиме удаленного терминала можно включить функцию распознавания адреса общего вызова и механизм работы с групповыми сообщениями. Для этого следует установить бит BCMSGEN в регистре BSICON.

Также можно включить функцию распознавания командного и ответного слов по десятому биту, установив бит B10EN в регистре BSICON. Это необходимо для нормальной работы монитора шины, если он имеется в общей сети.

В удаленном терминале аппаратно реализован циклический возврат данных. Если удаленный терминал получает команду на прием данных в подадрес 11110, а сразу за этим команду на передачу такого же количества слов из подадреса 11110, то будут переданы данные, полученные в предыдущем сообщении. Если же количество слов, принятых в подадрес 11110 будет отличаться от количества запрошенных из подадреса 11110 слов, то переданы будут данные из 30-го блока данных, начиная с адреса 1782h (обычный режим работы).

Инициализация

Для включения удаленного терминала необходимо:

- установить режим удаленного терминала, записав 10b в поле MODE регистра BSICONFIG;

- записать собственный адрес в поле TADDR регистра BSICON;

- при необходимости разрешить работу с групповым адресом, установив бит BCMSGEN;

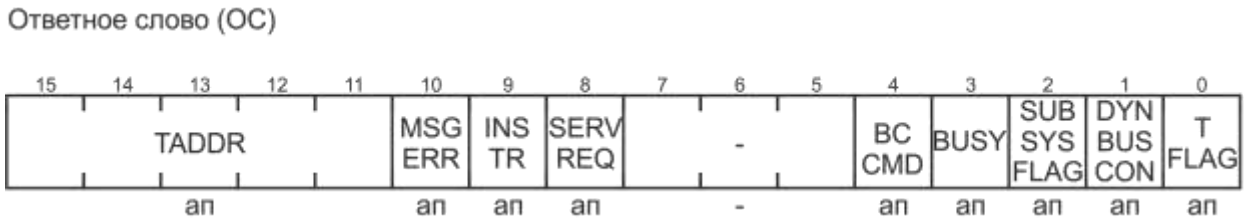
- запустить удаленный терминал, установив бит START.

При этом блокируются любые изменения регистра BSICONFIG и поля TADDR регистра BSICON.

После установки бита START удаленный терминал переходит в режим ожидания командного слова.

Для контроля работы удаленного терминала со стороны контроллера шины используется ответное слово, которое отправляется контроллеру в конце передачи каждого сообщения или по прямому запросу. Описание ответного слова приведено в таблице 20.11.

Таблица 20.11 – Формат и назначение битов ответного слова ОС



Поле	Бит	Описание
TADDR	15-11	Собственный адрес удаленного терминала.
MSGERR	10	Ошибка в сообщении. Установленный бит указывает на то, что ОУ не смог осуществить корректный прием.
INSTR	9	Бит распознавания (всегда ноль).
SERV REQ	8	Запрос на обслуживание Устанавливается посредством бита SERVREQ регистра BSICON.
BCCMD	4	Принято групповое сообщение. Устанавливается, если принято групповое сообщение, и установлен бит BCMSGEN в регистре BSICON.
BUSY	3	Абонент занят. Устанавливается посредством бита BUSY регистра BSICON. Указывает на то, что подчиненное устройство (абонент) занят.
SUBSYS FLAG	2	Неисправность абонента. Устанавливается посредством бита SUBSYSFLAG регистра BSICON. Указывает на ошибки в работе подчиненного устройства (абонента).
DYNBUS CON	1	Бит подтверждения принятия управления. Устанавливается аппаратно в случае, если удаленный терминал получил команду управления «Принять управление интерфейсом» с кодом 00h и установлен бит DYNBUSCON в регистре BSICON
TFLAG	0	Неисправность удаленного терминала. Устанавливается посредством бита TFLAG регистра BSICON. Указывает на ошибки в работе оконечного устройства.
–	7-5	Зарезервировано

Примечание – Ответное слово не передается после приема группового сообщения и в случае обнаружения ошибки в принятых данных.

Прерывания

Управление прерываниями осуществляется с помощью битов USUAL_INT_EN и ERR_INT_EN регистра BSICON.

Если бит USUAL_INT_EN установлен, то прерывания будут происходить после безошибочного окончания сообщения.

Если установлен бит ERR_INT_EN регистра BSICON, то прерывания будут происходить после сообщений, завершенных с ошибкой.

При сброшенных битах USUAL_INT_EN и ERR_INT_EN прерываний не будет.

Прерывания также никогда не формируются после выполнения следующих команд управления:

- передать ответное слово (00010);
- блокировать признак неисправности удаленного терминала (00110);
- разблокировать признак неисправности удаленного терминала (00111);
- передать последнюю команду (10010).

Для определения причины прерывания используется регистр BSISTAT, где сохраняется код причины прерывания и данные о последней переданной команде.

Блокировка передатчика

При приеме команды «Блокировать передатчик» (код 04h) по линии 0/1 удаленный терминал устанавливает на выходе MnBLOCK0/MnBLOCK1 высокий уровень сигнала и будет удерживать его до прихода команды «Разблокировать передатчик» (код 05h) по соответствующей линии или сброса микроконтроллера.

20.3 Монитор шины

Монитор шины (МШ) непрерывно «слушает» оба канала и сохраняет в памяти результаты мониторинга. Монитор шины имеет собственный 32-разрядный таймер, который запускается при включении режима МШ и отсчитывает интервалы времени по 0,5 мкс. Информация о каждом сообщении в линии сохраняется в информационном блоке, который формируется в ОЗУ в диапазоне адресов от 0D’8000h до 0D’8FFFh (для блока М0) и от 0D’9000h до 0D’9FFFh (для блока М1). Формат информационного слова в таблице 20.12.

Длина каждого информационного блока составляет от 5 до 40 16-разрядных слов в зависимости от сообщения. Информационный блок сохраняется в ОЗУ от младшего адреса к старшему и имеет следующий формат:

1 Адрес следующего информационного блока. Если адрес равен 0, значит текущее сообщение полностью еще не принято.

2 Старшее слово таймера на момент принятия достоверного командного слова.

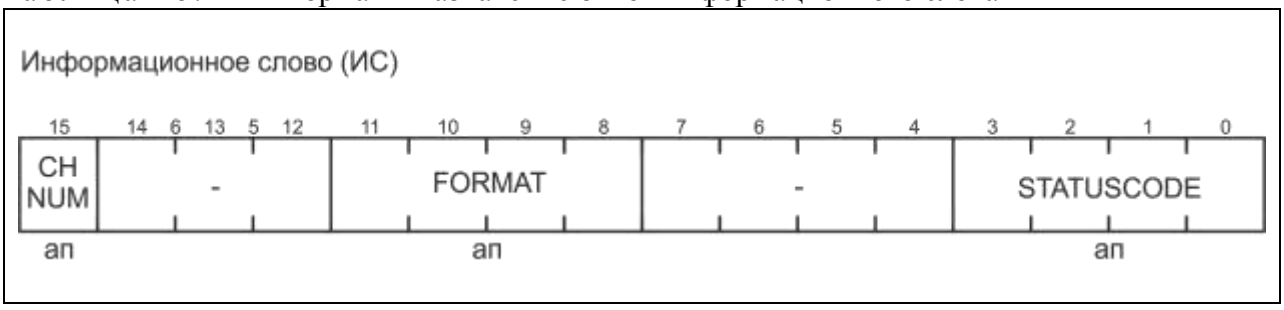
3 Младшее слово таймера на момент принятия достоверного командного слова.

4 Информационное слово.

5 – 40 Все принятые слова сообщения, начиная с командного, по порядку принятия.

Монитор шины отличает командное слово от ответного по десятому биту. Это необходимо учитывать при проектировании всей системы.

Таблица 20.12 – Формат и назначение битов информационного слова



Окончание таблицы 20.12

Поле	Бит	Описание	
CHNUM	15	Обнаруженный канал	
		0	Основной
		1	Резервный
FORMAT	11-8	Формат сообщения	
STATUS CODE	3-0	Код статуса окончания сообщения	
		0h	
		1h	Безошибочное окончание одного сообщения
		2h	Генерация в канале 0
		3h	Генерация в канале 1
		4h	Ошибка контроля передачи (принято не то, что передано)
		5h	Неправильный адрес в ответном слове передатчика (форматы 3 и 8)
		6h	Неправильный синхроимпульс в ответном слове передатчика (форматы 3 и 8)
		7h	Ненулевое ответное слово передатчика (форматы 3 и 8)
		8h	Ошибка манчестерского кода в ответном слове передатчика (форматы 3 и 8)
		9h	Отсутствие ответного слова передатчика (форматы 3 и 8)
		Ah	Неправильный синхроимпульс в слове данных
		Bh	Ошибка манчестерского кода в слове данных
		Ch	Отсутствие слова данных или нарушение непрерывности при приеме слова данных
		Dh	Ошибка манчестерского кода в командном слове передатчика (форматы 3 и 8)
		Eh	Отсутствие 2-го командного слова или нарушение непрерывности при приеме 2-го командного слова (форматы 3 и 8)
		Fh	Неверное командное слово (недопустимая комбинация бит) передатчика (форматы 3 и 8)
–	14-12, 7-4	–	Зарезервировано

Пример заполнения памяти представлен на рисунке 20.5.

Рисунок 20.5 – Пример заполнения ОЗУ в режиме монитора шины

В регистре BSICON имеется возможность установить режим выборки сообщений. В зависимости от значения поля MONMODE можно отслеживать все подряд сообщения, все сообщения с ошибками, сообщения для удаленного терминала с определенным адресом, сообщения для определенного подадреса любого удаленного терминала, а также сообщения для конкретного подадреса конкретного удаленного терминала.

Инициализация

Для включения монитора шины необходимо:

- установить режим монитора шины, записав 11b в поле MODE регистра BSICONFIG;
- установить требуемый режим выборки в поле MONMODE регистра BSICON, а также, если это необходимо, контролируемые адрес и подадрес;
- задать в поле INTNUM регистра BSICON частоту прерываний;
- запустить монитор шины, установив бит START регистра BSICON. При этом блокируются любые изменения регистра BSICONFIG и полей ADDR, SUBADDR и MONMODE регистра BSICON.

После установки бита START монитор шины переходит в режим ожидания командного слова, соответствующему режиму выборки.

Приняв достоверное, соответствующее выбранному режиму командное слово, МШ фиксирует и сохраняет в информационном блоке по адресу, указанному в регистре BSISADDR, значение таймера на момент принятия команды и само командное слово. Затем сохраняет каждое принятое слово сообщения. После окончания сообщения в информационный блок записывается слово статуса и реальный адрес следующего информационного блока. Этот же адрес сохраняется в регистре BSISADDR. По этому же адресу в первое слово следующего информационного блока записывается 0000h. Далее все повторяется.

Прерывания

В поле INTNUM регистра BSICON можно задать количество информационных блоков, по истечении записи которых генерируется прерывание. При этом в регистре BSISTAT хранится адрес первого еще непрочитанного информационного блока, а регистре BSISADDR – адрес информационного блока, который будет заполнен после принятия следующего сообщения. После чтения регистра BSISTAT, обработчик прерывания должен прочитать обязательно все непрочитанные информационные блоки, так как после чтения в этот регистр будет записано другое значение и данные могут быть потеряны.

21 Аналого-цифровой преобразователь

В состав контроллера входит 12-разрядный блок АЦП, включающий схему выборки и хранения, а также мультиплексор, выбирающий один аналоговый канал из 16 (входы порта P5 микроконтроллера).

Блок АЦП реализует несколько режимов преобразования:

- однократное преобразование фиксированного канала;
- непрерывное преобразование фиксированного канала;
- автоматическое однократное сканирование, т. е. последовательное переключение каналов с выполнением однократного преобразования каждого;
- автоматическое непрерывное сканирование;
- режим ожидания чтения результата, т. е. включение преобразования только после того, как предыдущий результат будет прочитан.

Выводы ∩VCC3 и ∩0V3 микроконтроллера являются выводами отдельного питания блока АЦП. Опорное напряжение должно быть стабильным во время любого преобразования (±100 мкВ) для достижения максимальной точности.

21.1 Функционирование блока АЦП

Функциями блока АЦП управляет побитно адресуемый регистр ADC_CON.

Битовое поле ADCH управляет входной логикой мультиплексора каналов. В однократных режимах поле ADCH задает входной канал, который должен быть преобразован, а в режимах автоматического сканирования – номер самого старшего канала, который будет преобразован в цикле сканирования.

Режим преобразования задается полем ADM.

Поля ADCH и ADM могут быть изменены во время преобразования. В режимах фиксированного канала новое значение будет использоваться после завершения текущего преобразования, а в режиме автоматического сканирования – после завершения текущего цикла сканирования.

Флаг занятости ADBSY устанавливается, если преобразование началось (т. е. установлен бит ADST), и остается установленным, пока преобразование выполняется. В состоянии простоя (преобразование не выполняется) бит ADBSY сброшен.

Однократное преобразование или последовательность преобразований начинается после установки бита ADST. После установки флага ADBSY преобразователь выбирает канал, который указан в поле ADCH. Напряжение на входе канала будет захвачено внутренней схемой сэмплирования после запуска преобразователя. По окончании преобразования результат вместе с номером преобразованного канала заносится в регистр ADC_DAT, и генерируется запрос на прерывание.

Бит ADST остается установленным, пока не будет сброшен программно или аппаратно. Аппаратно бит ADST сбрасывается:

- после завершения преобразования в режиме однократного преобразования фиксированного канала;
- после завершения преобразования нулевого канала в режиме автоматического однократного сканирования.

В режимах непрерывного преобразования бит ADST может быть сброшен только программно.

В зависимости от режима работы блока АЦП его реакция на сброс бита ADST может быть следующей:

- в режиме однократного преобразования фиксированного канала блок АЦП завершает преобразование, сбрасывает флаг ADBSY и затем останавливается (независимо от программного сброса бита ADST в момент выполнения текущего преобразования);

- в режиме непрерывного преобразования фиксированного канала блок АЦП завершает текущее преобразование, сбрасывает флаг ADBSY и затем останавливается;

- в режимах автоматического однократного сканирования и автоматического непрерывного сканирования АЦП завершает преобразование текущего канала, сбрасывает флаг ADBSY и затем останавливается.

21.2 Режимы работы

Режим преобразования фиксированных каналов

Преобразование канала, заданного в битовом поле ADCH начинается после установки бита ADST и автоматической установки флага ADBSY. После завершения преобразования будет установлен флаг ADCIR в регистре управления прерыванием ADC_CIC (см. описание регистра xxIC).

В режиме однократного преобразования преобразователь автоматически останавливается по окончании преобразования. В режиме непрерывного преобразования каждое новое преобразование начинается автоматически. Флаг ADCIR устанавливается после каждого преобразования.

Режимы преобразования с автоматическим сканированием

В режиме автоматического сканирования преобразователь автоматически выполняет преобразование каналов, по очереди, начиная с указанного в поле ADCH и заканчивая нулевым каналом. Преобразование начинается после установки бита ADST и флага ADBSY. После завершения преобразования одного канала устанавливается флаг ADCIR и автоматически начинается преобразование следующего канала. После преобразования нулевого канала текущая последовательность сканирования считается законченной.

Далее, в случае режима однократного сканирования преобразователь останавливается и сбрасывает биты ADST и ADBSY. В случае режима непрерывного сканирования преобразователь автоматически начинает новую последовательность с канала, указанного в поле ADCH.

На рисунке 21.1 показан пример работы блока АЦП в режиме непрерывного автоматического сканирования каналов.

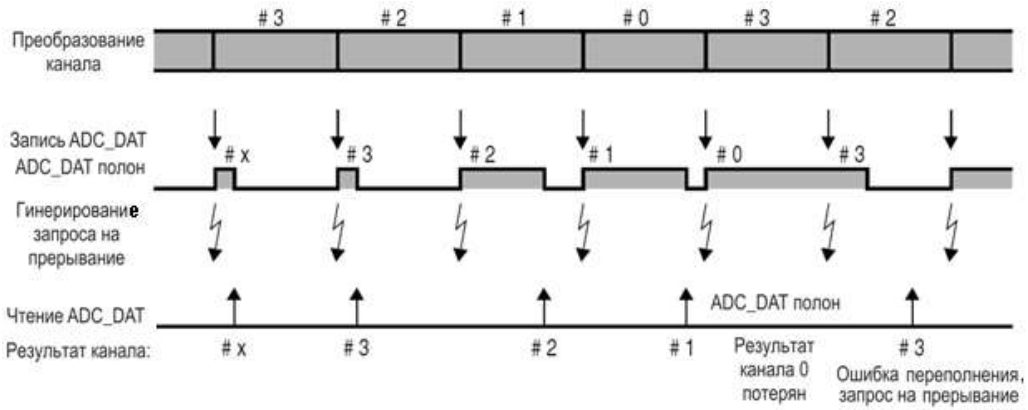


Рисунок 21.1 – Работа в режиме непрерывного автоматического сканирования

Режим ожидания чтения

Если значение результата предыдущего преобразования не было прочитано из регистра ADC_DAT до завершения текущего преобразования, то оно будет потеряно, и при этом будет установлен флаг ADEIR в регистре управления прерыванием ADC_EIC (см. описание регистра xxIC).

Во избежание потерь результатов преобразований и генерирования запросов прерываний по ошибке следует переключить блок АЦП в режим ожидания чтения регистра результата установкой бита ADWR.

В режиме ожидания чтения если к моменту окончания преобразования в регистре ADC_DAT находится непрочитанное значение, то результат преобразования сохраняется во временном буфере и дальнейшие преобразования не выполняются. Значения ADST и ADBSY остаются неизменными, и запрос прерывания по окончании преобразования не генерируется. После чтения регистра ADC_DAT содержимое временного буфера переносится в регистр ADC_DAT, при этом генерируется запрос на прерывание.

Этот механизм работает как в режиме однократного преобразования, так и в режиме многократного преобразования. На рисунке 21.2 показан пример работы блока АЦП в режиме ожидания чтения.

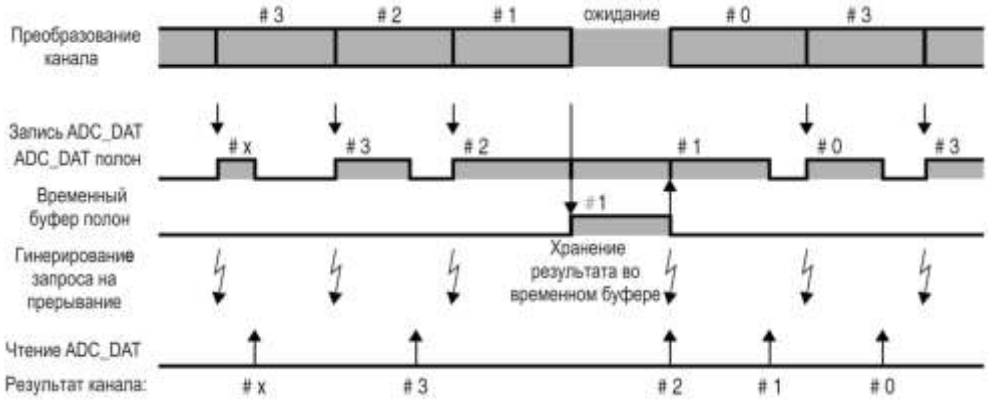


Рисунок 21.2 – Работа в режиме ожидания чтения

21.3 Синхронизация работы блока

На тактовый вход блока АЦП подается системный синхросигнал FPER. Внутри блока АЦП частота этого сигнала дополнительно делится. Коэффициент деления задается полем ADCTC регистра ADC_CON. Полученный на выходе делителя сигнал ADC_CLK используется в качестве основного тактового сигнала. На рисунке 21.3 представлена временная диаграмма работы блока АЦП.

Рисунок 21.3 – Временная диаграмма АЦП

После запуска преобразования внутренний сигнал SAMPCLK определяет текущий режим работы: выборку данных или преобразование. Сэмплирование данных производится в то время, когда SAMPCLK=1. Длительность семплирования составляет 6 тактов сигнала ADC_CLK, во время выполнения выборки входной сигнал должен оставаться стабильным. После завершения семплирования внутренний сигнал SAMPCLK автоматически переключается в состояние низкого логического уровня, тем самым запуская преобразование напряжения, зафиксированного на внутренней схеме выборки. Преобразование длится 13 тактов тактового сигнала ADC_CLK. Если АЦП работает в режимах непрерывного преобразования и автоматического сканирования, то после завершения текущего преобразования сразу же начинается новый цикл преобразования. По истечении указанного времени необходимо еще 1/2 такта сигнала ADC_CLK для того, чтобы результат преобразования был помещен в регистр результата ADC_DAT, выработался флаг окончания преобразования EOC и сформировался флаг запроса прерывания по окончании преобразования. Для корректной оцифровки входного сигнала последний должен оставаться стабильным во время выборки (SAMPCLK=1), однако может произвольно изменяться в процессе выполнения преобразования (SAMPCLK=0). Таким образом, общее время преобразования АЦП от начала выборки сигнала до записи результата в регистр результата занимает 19 тактов тактового сигнала ADC_CLK. Время преобразования может быть определено по формуле:

$$T_{ADC} = \frac{K}{f_{ADC} * 19}, \tag{21.1}$$

где K – коэффициент деления частоты входного сигнала и определяется значением поля ADCTC.

Передаточная характеристика АЦП

Диапазон преобразования АЦП U_{FSR} определяется напряжением опорного источника, подаваемым на вывод REFIO микроконтроллера, т. е. $U_{FSR} = U_{REFIO}$.

Значение младшего разряда определяется по формуле:

$$U_{LSB} = \frac{U_{FSR}}{2^N} = \frac{U_{FSR}}{4096} = \frac{U_{REFIO}}{4096}, \tag{21.2}$$

где N = 12 (разрядность АЦП).

Код на выходе АЦП определяется соотношением:

$$D = \frac{U_{IN}}{U_{LSB}} = \frac{U_{IN} * 4096}{U_{FSR}} \tag{21.3}$$

Передаточная характеристика АЦП показана на рисунке 21.4

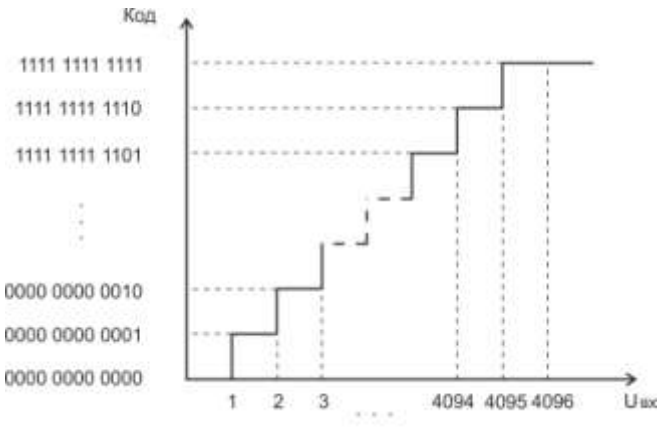


Рисунок 21.4 – Передаточная характеристика АЦП

21.4 Прерывания

В конце каждого преобразования устанавливается флаг запроса прерывания ADCIR в регистре ADC_CIC. Этот флаг может использоваться для входа в стандартное прерывание по вектору ADCINT или обслуживание блоком PEC. Эти действия необходимы для записи результата преобразования в ОЗУ.

Если к моменту окончания текущего преобразования результат предыдущего преобразования не был прочитан из регистра ADC_DAT, то устанавливается флаг запроса прерывания ADCIR в регистре ADC_EIC. Этот флаг может использоваться для входа в стандартное прерывание по вектору ADEINT или обслуживание блоком PEC.

22 Блок кодирования по ГОСТ 28147-89

Блок кодирования/декодирования (далее – блок кодирования) представляет собой аппаратную реализацию методов кодирования данных на основе алгоритмов ГОСТ 28147–89.

Блок кодирования позволяет производить кодирование/декодирование данных:

- гаммированием;
- гаммированием с обратной связью.

Важным дополнением является возможность блока кодирования формировать на основе исходных данных контрольную комбинацию, так называемую, имитовставку (аналог контрольной суммы), которая позволяет создавать дополнительную защиту данных.

Функциональное описание

В состав блока входят регистры управления и состояния CDCON, CDDATNUM, CDSTATE, основные и дополнительные регистры для кодирования исходных данных CDUW, CDAUTH, CDKEY и CDSBSTN, а также регистр исходных/полученных данных CDDATA.

Посредством регистра управления CDCON осуществляется конфигурирование блока, а также запуск и остановка (при необходимости) кодирования.

Дополнительным регистром для задания числа блоков данных, подлежащих обработке, является регистр CDDATNUM. Фактически, регистр CDDATNUM является счетчиком и находящееся в нем значение уменьшается на единицу каждый раз, по окончании обработки очередного блока данных. Достижение регистром нулевого состояния указывает на то, что все данные обработаны. Одновременно с этим в регистре состояния CDSTATE устанавливается флаг DONE.

При использовании регистра CDDATNUM, его состояние должно быть задано до начала процесса кодирования/декодирования (до установки бита START).

В случае, если к моменту установки бита START состояние регистра равно нулю, то окончание преобразования всего потока данных должно быть указано своевременной установкой бита STOP.

Блок оперирует с 8-байтными блоками исходных данных. Таким образом, если имеется массив данных, он должен быть разбит на блоки по четыре слова. Если последний блок данных меньше четырех слов, то он должен быть дополнен любым набором бит до размера четырех полных слов.

Для загрузки одного блока данных используется регистр данных CDDATA. Регистр является 64-битным. Запись в регистр происходит пословно по адресу, указанному в таблице 22.1. Если данные поступают в порядке D0, D1, D2, D3, то их размещение в регистре CDDATA будет соответствовать указанному на рисунке 22.1.

Рисунок 22.1 – Структура регистра CDDATA

В пределах регистра данные разбиваются на два 32-разрядных подблока с названиями DATA_H и DATA_L.

Помимо источника данных для преобразования, регистр CDDATA также является приемником обработанных данных, для получения которых требуется четырехкратное использование команды чтения. Порядок получения данных при чтении – D0, D1, D2, D3. Для записи и чтения регистра CDDATA используется один адрес.

В состав блока кодирования входят ключевое устройство с регистром CDKEY и блок замены с регистром CDSBSTN.

Регистр CDKEY является 256-битным регистром ключа. Запись в регистр происходит пословно. Если данные поступают в порядке K0, K1, K2 и т. д., то их размещение в регистре CDKEY будет соответствовать рисунку 22.2. Регистр CDKEY доступен только для записи.

Рисунок 22.2 – Структура регистра CDKEY

В пределах регистра данные разбиваются на восемь подблоков с названиями KEY0, KEY1, KEY2, KEY3, KEY4, KEY5, KEY6, KEY7. Каждый из этих подблоков также является ключом и используется в процессе кодирования/декодирования. Выбор ключа осуществляет ключевое устройство под управлением логики.

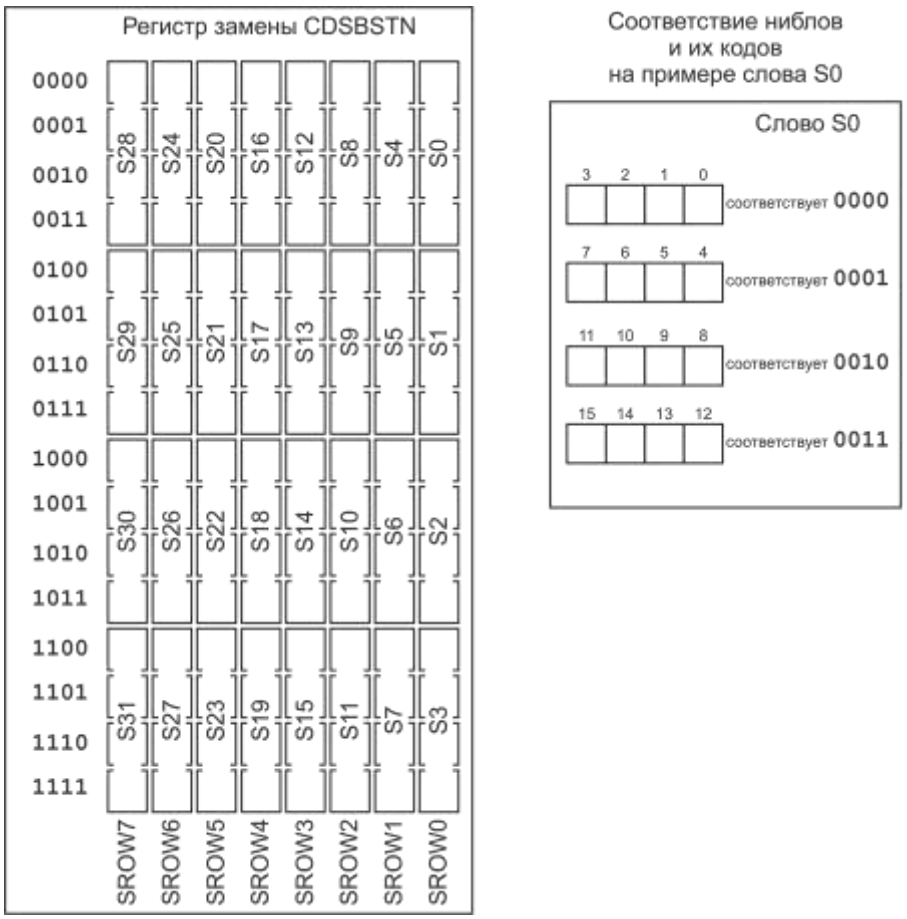


Рисунок 22.3 – Структура регистра CDSBSTN

Регистр CDSBSTN является 512-битным регистром замены. Запись в регистр происходит пословно. Если данные поступают в порядке S0, S1, S2 и т. д., то их размещение в регистре CDSBSTN будет соответствовать указанному на рисунке 22.3. Регистр CDSBSTN доступен только для записи.

В пределах регистра данные разбиваются на восемь подблоков с названиями SROW0, SROW1, SROW2, SROW3, SROW4, SROW5, SROW6, SROW7. Структура регистра CDSBSTN, показанная на рисунке 22.3, соответствует таблице подстановки, применяемой в процессе кодирования/декодирования, с восемью столбцами и 16 строками. Каждому ниблу соответствует код (двоичный порядковый номер).

Согласно алгоритму кодирования/декодирования, двойное слово обрабатываемых данных разбивается на восемь полубайт, каждому из которых соответствует один из столбцов, см. рисунок 22.4. Значение нибла является кодом выбора полубайта из регистра замены. Выбранные полубайты в свою очередь образуют двойное слово результата подстановки.

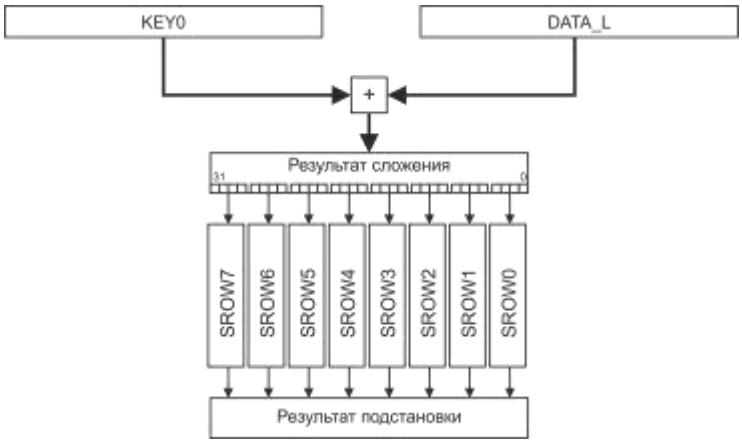


Рисунок 22.4 – Частичная функциональная схема алгоритма режима простой замены

Помимо рассмотренных ключа и таблицы подстановки, для кодирования/ декодирования данных, в некоторых случаях требуются дополнительные исходные данные, называемые синхропосылкой. Для хранения синхропосылки используется регистр CDUW. Регистр является 64-битным. Запись в регистр происходит пословно. Если данные поступают в порядке UW0, UW1, UW2, UW3, то их размещение в регистре CDUW будет соответствовать указанному на рисунке 22.5. Регистр CDUW доступен только для записи.

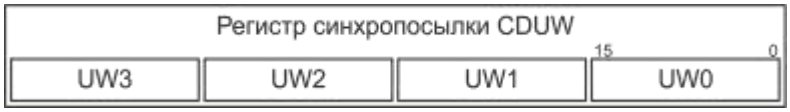


Рисунок 22.5 – Структура регистра CDUW

Для обеспечения имитозащиты данных применяется имитовставка, которая вычисляется параллельно с процессом кодирования/декодирования. Результат вычисления помещается в регистр CDAUTH. Размещение данных в регистре CDAUTH будет соответствовать рисунку 22.6.

Рисунок 22.6 – Структура регистра CDAUTH

Регистр CDAUTH является 64-битным и доступен только для чтения. Для получения имитовставки требуется четырехкратное использование команды чтения. Порядок получения данных при чтении – AU0, AU1, AU2, AU3.

В процессе работы возможны ситуации некорректного задания исходных данных (недозагрузка регистра) или неполного прочтения вычисленных.

В зависимости от ситуации, логика блока установит соответствующий флаг ошибки или установит флаг предупреждения в регистре CDSTATE. Регистр CDSTATE является регистром состояния блока и доступен только для чтения.

Примечание – Блок кодирования не генерирует прерывания, поскольку время, затрачиваемое на кодирование/декодирование, во много меньше времени выполнения одной (любой) команды из системы команд микроконтроллера. Это означает, что после запуска преобразования можно следующей командой уже считывать результат.

Таблица 22.1 – Адреса доступа к 64-/256-/512-битным регистрам блока кодирования

Мнемоническое название регистра	Адрес доступа к регистру	Обязательное количество последовательных циклов обращения к регистру	Состояние после сброса
CDDATA	F1A8h	4 (запись/чтение)	0000h
CDKEY	F1ACh	16 (только запись)	0000h
CDSBSTN	F1AEh	32 (только запись)	0000h
CDUW	F1B0h	4 (только запись)	0000h
CDAUTH	F1B6h	4 (только чтение)	0000h

Режимы работы

В зависимости от состояния битового поля MODE регистра CDCON, блок может функционировать в одном из трех режимов кодирования/декодирования данных.

Режим простой замены

Режим кодирования блоков данных, опирающийся на алгоритм, в котором блоки открытых данных кодируются с помощью ключа и таблицы замены с получением блока закодированных данных. Декодирование данных происходит в обратном, относительно кодирования, порядке с теми же ключом и таблицей замены.

Режим гаммирования и гаммирования с обратной связью

Режимы кодирования блоков данных, опирающиеся на алгоритм, в которых открытые данные кодируются с помощью гамм-кодов, получаемых на основе закодированных с помощью ключа и таблицы замены синхровставок с получением блоков закодированных данных. Декодирование данных происходит в обратном, относительно кодирования, порядке с теми же синхровставкой, ключом и таблицей замены.

Вычисление имитовставки (доступно для всех режимов)

Вычисление имитовставки может происходить параллельно с выполнением основного алгоритма выбранного режима. Механизм вычисление имитовставки единообразен для всех режимов и включается установкой бита AUTH регистра CDCON.

Вычисление опирается на алгоритм, в котором данные преобразуются с помощью ключа и таблицы замены с получением 64-битной имитовставки. По окончании преобразования всех данных имитовставка может быть считана из регистра CDAUTH.

Имитовставка, полученная вместе с закодированными данными, отделяется от них. Данные декодируются, и параллельно с этим вычисляется имитовставка. По окончании декодирования всех данных вычисленная имитовставка считывается из регистра CDAUTH и сравнивается с полученной. При несовпадении декодированные данные считаются ошибочными.

Для более подробного ознакомления с алгоритмами, лежащими в основе вычислений блока, следует обратиться к ГОСТ 28147–89.

Порядок работы с блоком

1 Запуск преобразования:

- в регистр ключа CDKEY записываются 32 байта данных;
- в регистр замены CDSBSTN записываются 64 байта данных;
- если предполагается использование одного из режимов гаммирования, то в регистр синхропосылки CDUW записываются восемь байт данных;
- в регистр данных CDDATA записываются восемь байт данных, подлежащих кодированию/декодированию;
- если известно количество блоков данных и если предполагается использование регистра CDDATNUM, то соответствующее значение должно быть записано в регистр;
- в регистре управления CDCON программируются биты, отвечающие за алгоритм преобразования данных, необходимость вычисления имитовставки, режим работы, и устанавливается бит START.

2 Получение данных и дальнейшая работа:

- по окончании кодирования следует прочитать регистр CDSTATE и проверить состояние флага DONE, указывающего на то, что преобразование завершилось успешно;
- если флаг DONE установлен, то преобразованные данные могут быть прочитаны из регистра CDDATA.
- если обработанный блок данных был не последний (согласно состоянию регистра CDDATNUM и/или нулевому состоянию бита STOP), то в регистр CDDATA записывается очередной блок данных, после чего процесс преобразования запускается автоматически (устанавливать бит START не нужно);
- если обработанный блок данных был последним и имитовставка не вычислялась, цикл работы с блоками данных считается завершенным;
- если был установлен бит AUTH и, соответственно, была вычислена имитовставка, она может быть прочитана из регистра CDAUTH, после чего цикл работы с блоками данных считается завершенным.

Особенности работы блока

Для обработки только одного блока загруженных данных нужно записать значение 01h в регистр CDDATNUM и установить бит START или одновременно установить биты START и STOP.

Количество циклов записи/чтения 32-/256-/512-разрядных регистров должно точно соответствовать указанному в таблице 22.1. Если число записанных/прочитанных слов окажется меньше или больше указанного, регистр будет считаться незаполненным/непрочитанным.

Если на момент запуска преобразования в регистре CDDATA и/или регистре CDAUTH находятся непрочитанные данные, то это не влияет на запуск, а логика устанавливает флаг/флаги предупреждения в регистре CDSTATE. Следует помнить, что непрочитанные данные будут утеряны, вследствие записи поверх них новых вычисленных значений.

До перезаписи по окончании преобразования, данные в регистре CDDATA сохраняются и могут быть прочитаны.

Независимо от состояния регистра CDDATNUM, установка бита STOP перед записью очередного блока данных будет означать, что этот блок данных последний. Следует помнить, что регистр CDDATNUM не сбрасывается аппаратно в случае преждевременного завершения обработки блоков данных. Поэтому следует контролировать состояние этого регистра и, при необходимости, перепрограммировать его перед каждой установкой бита START.

В любой момент все регистры (кроме CDCON) и конечный автомат преобразования блока кодирования могут быть сброшены установкой бита RST. При необходимости сброса только конечного автомата преобразования с сохранением состояния всех регистров нужно записать значение 00h в поле MODE.

23 Блок квадратурного декодера

Квадратурный декодер преобразует цифровой сигнал с датчика положения вала двигателя, позволяя вычислять скорость, ускорение, направление вращения, а также текущее положение вала.

Квадратурный декодер обрабатывает три входных сигнала:

- квадратурные A_IN и B_IN, сдвинутые по фазе на 90 градусов, используемые для определения скорости, ускорения и направления вращения ротора;
- индексный I_IN, сигнализирующий о полном обороте ротора и позволяющий осуществлять контроль счетчика позиции.

Сигналы A_IN, B_IN и I_IN подаются с выводов микроконтроллера P5.9, P5.8 и P5.7, соответственно, при включенных альтернативных функциях QDI, QDB и QDA. Общее относительное расположение сигналов во времени показано на рисунке 23.1.

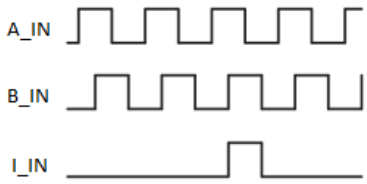


Рисунок 23.1 – Входные сигналы квадратурного декодера

Блок квадратурного декодера включает в свой состав входную логику, счетчик позиций и измерительный модуль. Управление и контроль осуществляются посредством регистров QD1CON и QD2CON.

Входная логика

Все входные сигналы могут быть независимо проинвертированы. Включение инверсии осуществляется битами INVA, INVB и INVI. Установка бита SWAPAB позволяет включить обмен сигналов, т. е. сигнал с вывода QDA микроконтроллера будет считываться как B_IN, а сигнал с вывода QDB – как A_IN. Кроме того реализована возможность выбора активного фронта индексного сигнала (по умолчанию используется положительный фронт) для инициирования одного из действий:

- пропуск фронта;
- сброс счетчика позиции;
- захват значения счетчика позиции;
- захват значения счетчика позиции и затем его сброс.

Определение направления вращения ротора происходит автоматически, исходя из последовательности изменения входных сигналов, и влияет на состояние бита QDIR.

Примечание – При отсутствии как минимум одного из двух квадратурных сигналов на входах состояние бита QDIR неопределенно.

При смене направления генерируется прерывание блока квадратурного декодера CHDIR_INT.

В таблице 23.1 приведены возможные состояния входных сигналов и соответствующие им направления переключения счетчика позиции

Таблица 23.1 – Направление переключения счетчика позиции в зависимости от события

Уровень другого сигнала	Событие:			
	фронт сигнала A_IN		фронт сигнала B_IN	
	Положительный	Отрицательный	Положительный	Отрицательный
Высокий	Вниз	Вверх	Вверх	Вниз
Низкий	Вверх	Вниз	Вниз	Вверх

Реакция счетчика позиции на каждое из событий, приводящих к его переключению, управляется битами APEDGE, ANEDGE, BPEDGE и BNEDGE.

При обнаружении каждого из разрешенных событий генерируется прерывание QEVENT_INT и устанавливается соответствующий флаг в регистре QD2CON. Флаг удерживается до обнаружения следующего события, после чего сбрасывается и устанавливается новый флаг.

Счетчик позиции

Счетчик позиций используется для подсчета количества квадратурных событий, что позволяет контролировать текущее положение вала.

Счетчик представляет собой 16-разрядный регистр QPOS. Направление счета контролируется полем DIRCON, которое позволяет выбирать один из четырех режимов работы.

1 Квадратурный счет (включен по умолчанию). Направление переключения задается событиями на квадратурных входах и определяется согласно таблице 23.1.

2 Сигнал на входе QDB задает направление счета. Подсчитываются только события на входе QDA.

3 Счет только вверх (независимо от состояния бита QDIR) при обнаружении события.

4 Счет только вниз (независимо от состояния бита QDIR) при обнаружении события.

Верхняя граница счета определяется регистром QPOSMAX (по умолчанию, значение FFFFh). По достижении верхней границы счетчик сбрасывается в состояние 0000h. в случае обратного счета, по достижении значения 0000h в регистр счетчика будет загружено значение из регистра QPOSMAX. Каждый раз, когда значение счетчика совпадает с 0000h генерируется внутреннее прерывание OVF и выставляется флаг OVF (сбрасывается программно).

Значение счетчика постоянно сравнивается с значением регистра компаратора QPOSCOMP. При возникновении совпадения генерируется внутреннее прерывание COMP и выставляется флаг COMP (сбрасывается программно). Если установлен бит COMPRST, то после обнаружения совпадения при следующем событии в счетчик будет загружено значение QPOSMAX.

Поле INDCON задает действие, которое будет выполняться при обнаружении события на индексном входе QDI. По умолчанию, события игнорируются и никаких действий не производится. Как только действие задано, т. е. INDCON ≠ 00b, и обнаружено событие, запрограммированное действие будет выполнено одновременно с появлением следующего события.

Варианты действий

1 Пропуск события. Никаких действий не производится.

2 Захват значения счетчика позиции. Состояние счетчика сохраняется в регистре QPOSCAP. При этом генерируется внутреннее прерывание CAP и выставляется флаг CAPF (сбрасывается программно).

3 Захват значения счетчика позиции и его сброс. Действия выполняются аналогично пункту 3, а затем в счетчик загружается значение из регистра QPOSMAX.

4 Сброс счетчика позиции. В счетчик загружается значение из регистра QPOSMAX.

Сигналы внутренних прерываний OVF, COMP и CAP объединены логически по ИЛИ и при возникновении любого из прерываний генерируется прерывание блока квадратурного декодера QPOS_INT.

Измерительный модуль

Позволяет измерять в количестве тактов переключения счетчика временные отрезки между двумя квадратурными событиями: последовательными положительными фронтами

(измерение периода) или между двумя соседними фронтами (измерение длины импульса) сигнала A_IN.

Управление измерительным модулем осуществляется посредством регистра QD2CON. Измерение начинается по ближайшему положительному фронту входного сигнала A_IN после установки бита QMBUSY. Измерение может выполняться в режиме непрерывного счета или в режиме одного измерения.

Режим непрерывного счета задается установкой бита QMMODE. При обнаружении положительного фронта входного сигнала счетчик начинает инкрементироваться с частотой, полученной в результате деления частоты входного сигнала синхронизации FPER. Делитель частоты задается полем QTDIV. Счетчик переключается до тех пор, пока не будет обнаружен следующий положительный фронт. В этот момент значение счетчика сохраняется в регистре QMRES, а сам он сбрасывается в ноль и генерируется прерывание QMRES_INT. Далее счетчик продолжает инкрементироваться до появления следующего положительного фронта входного сигнала.

Примечание – Регистр QMRES не имеет дополнительного буфера, поэтому каждое новое значение записывается поверх предыдущего.

Режим одного измерения включен по умолчанию. После установки бита QMBUSY при обнаружении положительного фронта входного сигнала счетчик начинает инкрементироваться. С появлением очередного положительного фронта значение счетчика сохраняется в регистре QMRES и генерируется прерывание QMRES_INT. Счетчик сбрасывается в ноль и одновременно с этим сбрасывается бит QMBUSY, что выключает дальнейшее измерение.

На рисунке 23.2 показаны примеры измерения периода входного сигнала. При условии, что QTDIV = 000b (счетчик переключается на частоте fper) значение периода оказалось равным 00FFh. Это значение было помещено в регистр QMRES. В случае, если бы в поле QTDIV было записано значение 001b (частота переключения счетчика равна fper/2), то значение периода было бы равно 007Fh.

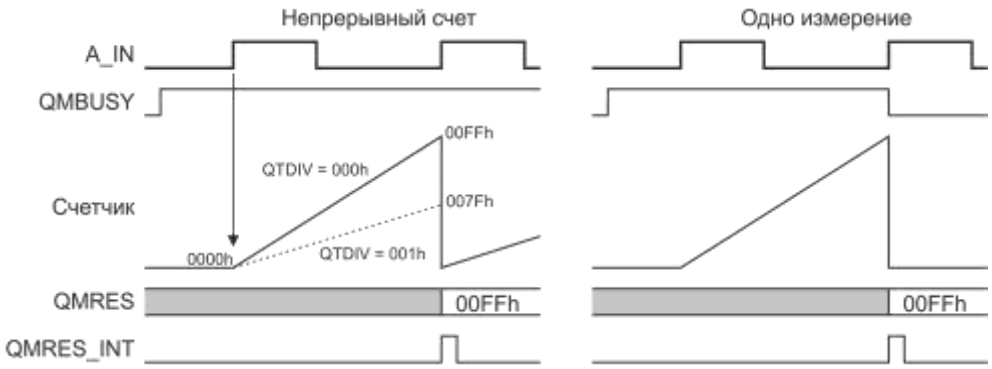


Рисунок 23.2 – Измерение периода квадратурного сигнала

Для измерения длины импульса следует установить бит QMLEN.

В режиме непрерывного счета после включения счетчик начинает инкрементирование с появлением первого положительного фронта входного сигнала. При обнаружении следующего фронта (отрицательного) сигнала значение счетчика записывается в регистр QMRES и генерируется прерывание QMRES_INT. После этого счетчик сбрасывается в ноль и продолжает инкрементирование до появления следующего фронта сигнала. Таким образом, каждый фронт сигнала будет вызывать сохранение значение счетчика в регистре QMRES, его сброс в ноль и генерирование прерывания.

В режиме одного измерения после включения счетчик запускается с появлением положительного фронта входного сигнала. При обнаружении отрицательного фронта значение счетчика записывается в регистр QMRES и генерируется прерывание

QMRES_INT. После этого счетчик сбрасывается в ноль и продолжает инкрементирование до появления следующего фронта сигнала. Появление очередного положительного фронта приведет к сохранению значения счетчика, генерированию прерывания, сбросу бита QMBUSY и прекращению дальнейших измерений.

На рисунке 23.3 показаны примеры измерения длины импульса входного сигнала. Значение длины импульса оказалось равным 006Fh, а значение периода – сумме значений 006Fh и 0090h. Таким образом, коэффициент заполнения будет (006Fh/00FFh)%.

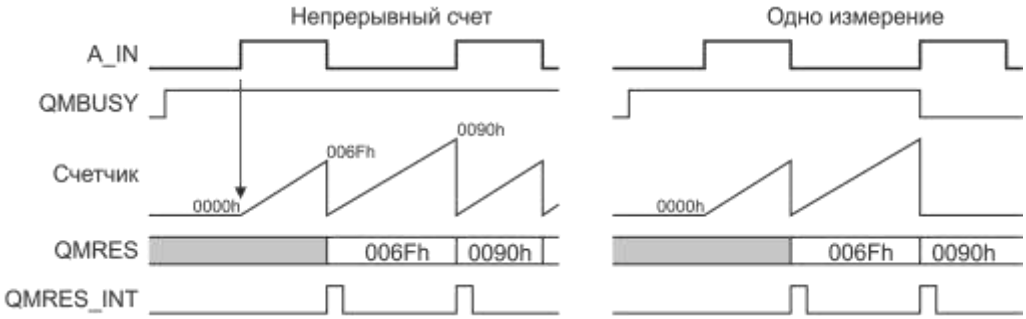


Рисунок 23.3 – Измерение длины импульса квадратурного сигнала

Параметры измерительного модуля накладывают ограничения на частоту следования импульсов входного квадратурного сигнала и их длину. При обслуживании прерывания QMRES_INT следует всегда проверять состояние флагов FAST и SLOW. Если флаги сброшены, то результат измерения, записанный в регистре QMRES можно считать достоверным. В противном случае, результат является не достоверным и нужно провести корректировку параметров измерительного модуля.

Следует помнить, что прерывание QMRES генерируется также и при установке флагов FAST и SLOW.

Установленный флаг FAST указывает на то, что частота переключения счетчика слишком высокая и он достиг значения FFFFh до того как был обнаружен ожидаемый фронт сигнала. Следует уменьшить частоту переключения счетчика за счет смены значения делителя полем QTDIV.

Установленный флаг SLOW указывает на то, что частота инкрементирования счетчика слишком мала и к моменту обнаружения ожидаемого фронта сигнала, он не сделал ни одного переключения. При этом в регистр QMRES будет записано значение 0000h. Следует увеличить частоту переключения счетчика за счет смены значения делителя полем QTDIV.

Флаги FAST и SLOW сбрасываются только программно.

Управление прерываниями

Управление прерываниями квадратурного декодера осуществляется посредством регистров QEVENT_IC (прерывание по событию), CHDIR_IC (прерывание при изменении направления), QPOSIT_IC (прерывание счетчика позиций) и QMRES_IC (прерывание по окончании измерения).

24 Средства отладки

24.1 Программно-аппаратные средства отладки

Для освоения и изучения 16-разрядных микроконтроллеров 1887ВЕ9Т, макетирования и отладки систем пользователя, а так же разработки и отладки программ используется программно-аппаратный комплекс, состоящий из аппаратной и программной частей.

Аппаратная часть представляет собой макетно-отладочную плату, которая позволяет осуществлять оценку работы прикладных программ в режиме реального времени. Плата имеет COM-порт и разъем OCDS для подключения средств отладки и программирования.

Программная часть – это стандартные средства отладки IDE Keil со средой «KeilμVision» с использованием аппаратного отладчика «Keil Ulink2» и загрузчик программного кода.

Описание программно-аппаратного комплекса приведено в КФДЛ.424939.014ТО.

24.2 Отладочная система OCDS

Устройство обеспечения отладки на кристалле позволяет выполнять эмуляцию аппаратного окружения. Отладочная система управляется непосредственно внешним устройством через выводы отладочного интерфейса. Структурная схема отладочной системы микроконтроллера приведена на рисунке 24.1.

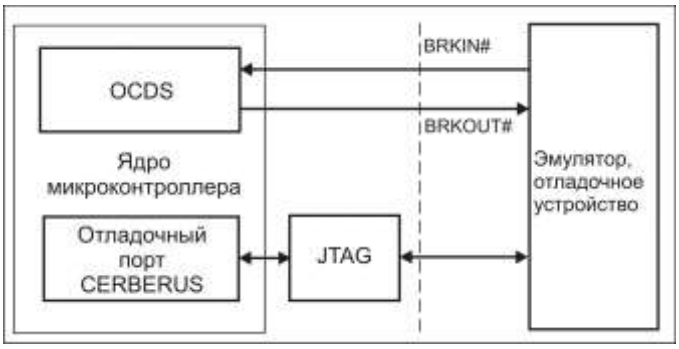


Рисунок 24.1 – Отладочная система микроконтроллера

Отладочная система микроконтроллера включает в свой состав блоки OCDS и JTAG, а также отладочный порт CERBERUS.

Характеристики блока OCDS:

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

Блок OCDS позволяет вести отладку программного обеспечения, запускаемого пользователем при помощи внешнего отладочного устройства через независимый 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 интерфейса JTAG 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 осуществлять 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 для ускорения вызова и поддержки нескольких отладочных приложений и для разрешения многозадачного совместного использования блока JTAG.

Основное предназначение блока CERBERUS это подключение интерфейса JTAG 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 прямой интерфейс для стандартных отладочных устройств и производят арбитраж доступа к JTAG интерфейсу прозрачным способом.

24.3 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 24.2.

Рисунок 24.2 – Структурная схема блока OCDS

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

Действия на отладочные события:

- останов ЦПУ (Halt);
- вызов монитора (Monitor);
- передача данных DPEC, выполняемая блоком CERBERUS;
- активизация внешнего выхода по выводу BRKOUT#.

Разрешение и запрещение работы блока OCDS

По умолчанию, работа блока OCDS запрещена для защиты системы в течение нормальной работы. Генерация событий может быть только тогда, когда блок OCDS разрешен. Сигнал разрешения соединен с внутренним сигналом сброса JTAG. Это означает, что блок 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 случайного программного разрешения, необходимо, чтобы истинными были следующие условия:

- OCDS запрещен;
- MUX_E = 10b (регистр DTREVT);
- SELECT_E ≠ 00b (регистр DTREVT);
- для регистра DCMP0 сравнение соответствует (независимо от SELECT_E);
- установлен бит DEBUG_ENABLED (регистр DBGSR).

Таким образом, монитор должен выполнить следующее:

- записать F0FCh (адрес регистра DBGSR) в регистр DCMP0;
- записать 2200h в регистр DTREVT;
- записать 0001h в регистр DBGSR.

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

Сброс с переходом в режим останова Halt

Микроконтроллер может быть переведен в режим Halt сразу после сброса. Это управляется битом RST_HLT регистра CCONF блока JTAG. Для сброса с переходом в режим Halt необходимо выполнить:

- установить бит RST_HLT до выполнения сброса микроконтроллера;
- задать DEBUG_STATE = 10h (регистр DBGSR) для перевода в режим Halt после выполнения сброса;
- сбросить бит RST_HLT.

Для вывода микроконтроллера из режима Halt необходимо задать DEBUG_STATE = 00h.

Источники отладочных событий

Аппаратные источники отладочных событий указаны в таблице 24.1.

Таблица 24.1 – Аппаратные источники событий запуска отладки

Источник запуска	Размер, бит	Назначение
TASKID	16	TASKID в регистре DTIDR
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)

Значение TASKID используется в расширенных системах операций реального времени для запоминания ID текущей задачи. Все источники запуска сравниваются и комбинируются в аппаратном устройстве генерации запуска отладочного события, которое программируется записью в управляющий регистр отладочных событий. Оно

состоит из двух частей. Одна часть служит для одного диапазона сравнения, а нижняя часть служит для трех сравнений на равенство.

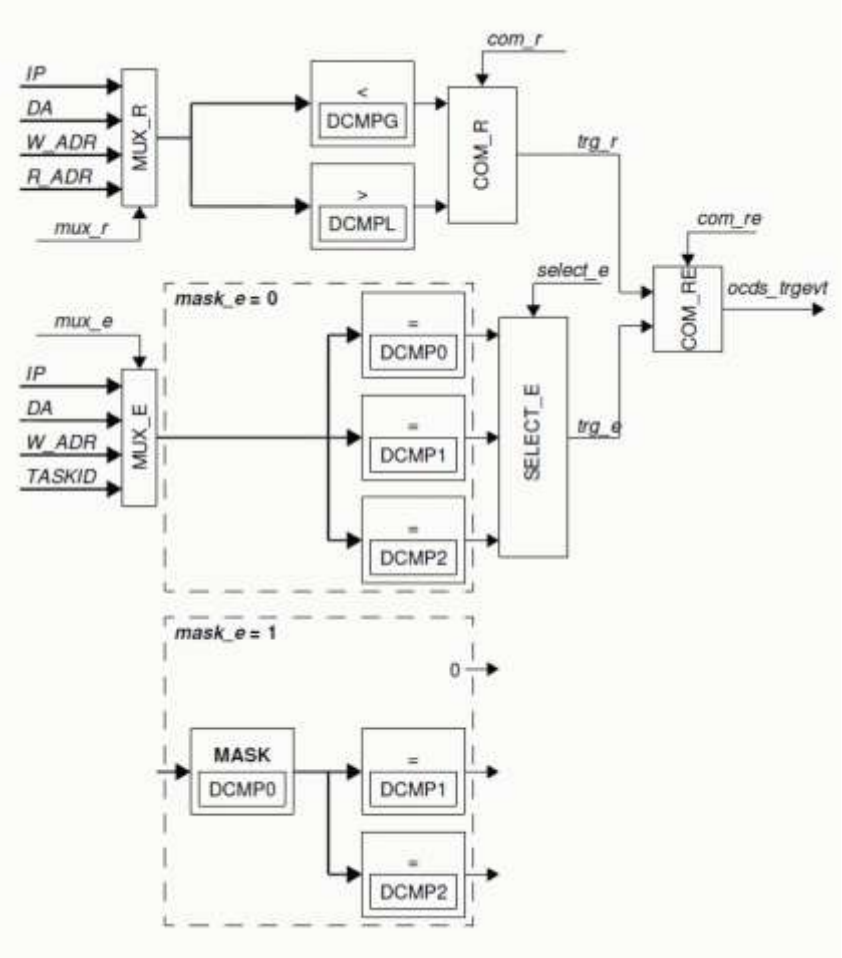


Рисунок 24.3 – Структурная схема аппаратного устройства генерации сигнала запуска отладочного события

На структурной схеме, приведенной на рисунке 24.3, 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 ОЗУ 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 команда NOP.

Вывод отладочного прерывания BRKIN#

Вывод микроконтроллера BRKIN# (вход отладочного прерывания) позволяет отладчику выполнять асинхронное прерывание микроконтроллера после переключения входного сигнала на выводе в низкий уровень и удержания его в таком состоянии не менее двух периодов системного тактового сигнала. Действие по прерыванию определяется регистром DEXEVT.

Приоритеты событий

При одновременном возникновении двух отладочных событий, первым обрабатывается событие с более высоким приоритетом. События и приоритеты приведены в таблице 24.2.

Таблица 24.2 – Приоритеты отладочных событий

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

Действия в ответ на отладочные события

Когда блок OCDS разрешен и возникает отладочное событие, выполняется одно из действий, указанных в таблице 24.3.

Таблица 24.3 – 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	–	–	

В ответ на отладочное событие возможны два вида реакций: включение блока CERBERUS и вызов монитора.

Включение блока CERBERUS для выполнения ожидаемой передачи DPEC 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.

Короткий вход в 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.

Примечание – Вызов монитора выполняется специальной аппаратной отладочной ловушкой trap 8 с адресом вектора 20h. Эта ловушка имеет наивысший приоритет, однако, подпрограмма монитора может ограничить собственный уровень приоритета при переустановке бита отладочного флага DEBTRAP в регистре TFR и записать поле ILVL в регистре PSW.

Модель отладки приведена на рисунке 24.4.

Рисунок 24.4 – Простая и расширенная модели отладки

Структура непрерываемой подпрограммы монитора:
- запуск процесса (непрерываемого);
- запись 0000h в регистр DBGSR;
- сброс бита DEBTRAP регистра TFR;
- возврат в пользовательскую программу командой RETI.

Структура прерываемой подпрограммы монитора:
- запись 00h в поле DEBUG_STATE регистра DBGSR (пользовательский режим);
- сброс бита DEBTRAP;
- уменьшение уровня прерывания ILVL в регистре PSW;
- запуск процесса;
- запись 0000h в регистр DBGSR;
- команда RETI.

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

Режим останова Halt

В режиме Halt система приостанавливает 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 возобновит пользовательский режим, когда внешнее аппаратное отладочное устройство сбросит поле DEBUG_STATE в пользовательский режим. Также нужно сбросить биты OCDS_P_SUSPEND и EVENT_SOURCE.

Активация внешнего вывода

Изменение уровня сигнала на внешнем выводе может быть определено как отладочное событие. Это может использоваться в подпрограммах, в которых работа системы не может быть прервана для сообщения внешним устройствам о возникновении события, а также для синхронизации внутренних и внешних аппаратных средств.

Пошаговая отладка в режиме Halt

Условия срабатывания должны оставаться истинными (например: включение диапазона IP с DCMPL = 000000h, DCMPG = 000001h и COM_R = 11b и COM_RE = 0b) и установлен бит BREAK_AFTER_MAKE в регистре DTREVT. После каждого ре-старта ЦПУ будет снова остановлен после выполнения очередной команды.

Пошаговая отладка с отладочным монитором

Преимуществом этого типа пошаговой отладки является то, что система может обслуживать высокоприоритетные запросы прерывания.

Режим имеет особенности:

- реакцией на отладочное событие является вызов монитора.

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

Рекомендуется корректировка адресного пространства IP запуска отладочного события для текущей (C-) функции пользовательской программы. Это приводит к пошаговому 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. Регистры управления отладочными событиями имеют одинаковую структуру для всех определяемых 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 режиме 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.

Поле SELECT_E разрешает включать сравнение на равенство в генерацию сигнала ocds_trgevt и выбирает режимы, см. таблицу 24.4. Следует помнить, что для маскируемого сравнения поле SELECT_E должно быть установлено в 10b или 11b.

Таблица 24.4 – 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 нулю, также устанавливаются в ноль до сравнения. Сравниваемые значения в DCMP1 и DCMP2 должны быть равны нулю, когда маска DCMP0 равна нулю, иначе при сравнении не будет соответствия.

Поле COM_R позволяет включать сравнение диапазона в формирование сигнала ocds_trgevt, см. таблицу 24.5. Для сравнений в диапазоне регистр DCMPG использует

верхнюю границу, а регистр DCMPL нижнюю границу диапазона. При сравнении за пределами диапазона все наоборот.

Таблица 24.5 – 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

Биты EVENT_SOURCE устанавливаются независимо от поля EVENT_ACTION, за исключением значения 000b. Эти биты должны быть перезаписаны 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, то соответствующая периферия приостанавливается. Этот бит устанавливается в ходе отладочного события в соответствии с битом PERIPHERALS_STOP. Этот бит должен быть перезаписан 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.

Регистры указания команд DIP и DIPX

Регистры предусмотрены для того, чтобы сделать видимым регистр указатель команд IP, когда CPU находится в режиме Halt.

Регистры аппаратного генератора сигнала запуска отладочного события

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

Обращение к регистрам OCDS

Функциями OCDS в основном управляет регистр DBGSR. Для корректного исполнения любого шага отладки необходимо чтобы соответствующее поле было запрограммировано 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:

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

2 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.

Общие приемы исключения проблем программного обеспечения с OCDS

Принципиальное решение 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 вступили в силу.

Использование некритических команд

После записи в регистр 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 – вставка команд NOP 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 fpdbus/fcpu = = 1/8, для меньшей пропорции необходимо больше команд NOP.

Операции чтения после записи

Немедленно после записи в OCDS может следовать операция чтения 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, гарантирующий 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. 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.

24.4 Блок JTAG

Блок JTAG является мостом между выводами JTAG (TDO, TRST#, TMS, TCK и TDI) и блоком CERBERUS и 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 флеш-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. На рисунке 24.5 показана структурная схема блока JTAG.

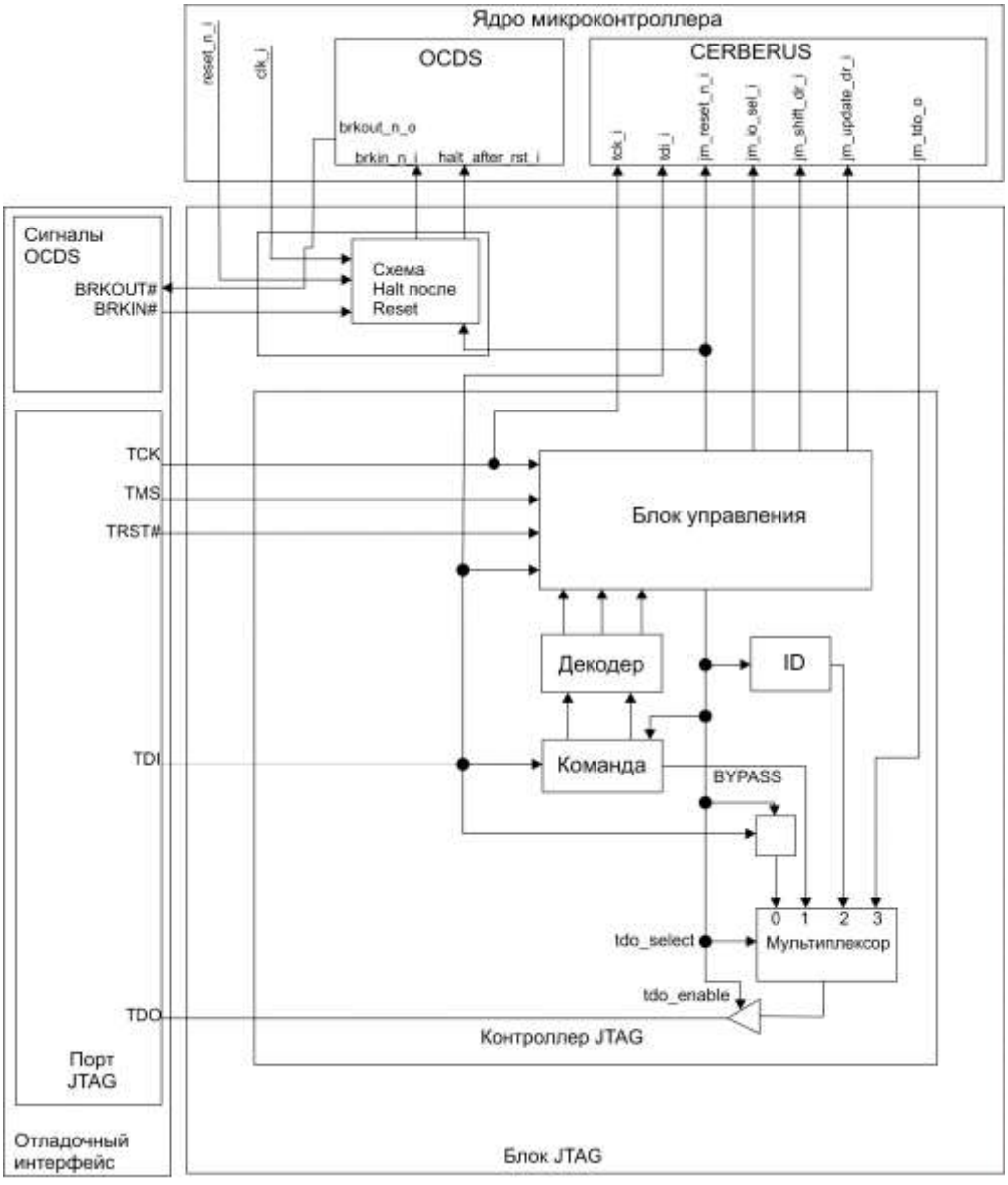


Рисунок 24.5 – Структурная схема блока JTAG

Схема состояний контроллера JTAG

Схема состояний контроллера JTAG является основной составляющей блока. Все переключения состояний происходят по положительному фронту, управляемому выводом TMS. После сброса TRST# схема состояний переходит в состояние сброса тестовой логики. При низком уровне сигнала на выводе TMS и положительном фронте сигнала на выводе TCK схема переводится в тестовое состояние холостого хода. 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 24.6 приведена схема состояний контроллера JTAG.

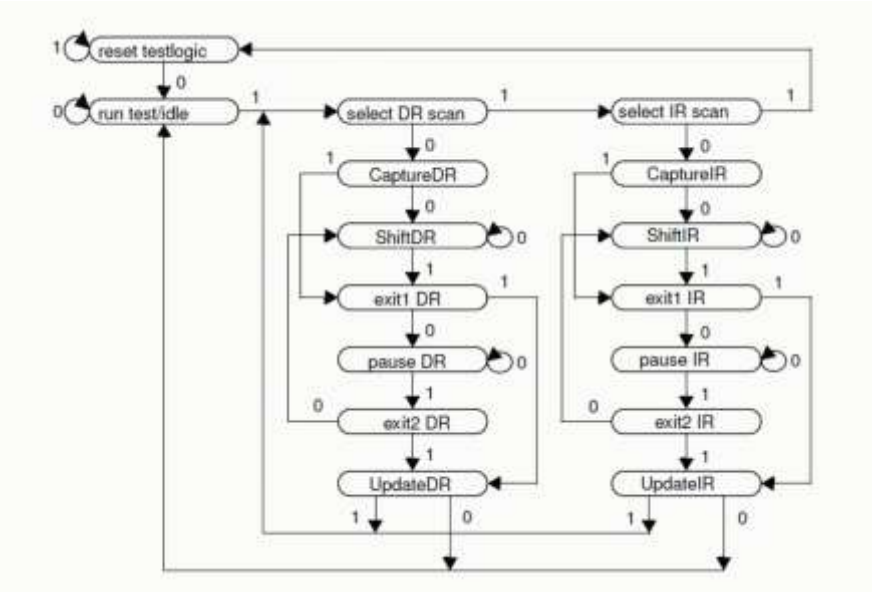


Рисунок 24.6 – Схема состояний контроллера JTAG

Все сигналы, приведенные на рисунке 24.6, 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 останова Halt микроконтроллера. Ввести этот режим возможно, когда микроконтроллер находится в состоянии сброса. 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 сигнал сброса 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 24.6, 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 24.6, 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 24.6.

Таблица 24.6 – 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

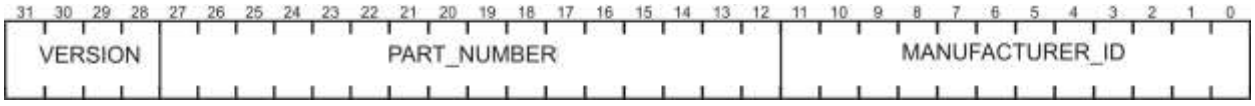
Это однобитный JTAG регистр. Если происходит выбор, то сигнал на выходе TDO равен сигналу на входе TDI, задержанный на один цикл сигнала TCK.

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 24.7.

Таблица 24.7 – 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 24.7, 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», а не «0», как при BYPASS. Это различие важно в случае ошибочного бита, когда команда JTAG помещается 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.

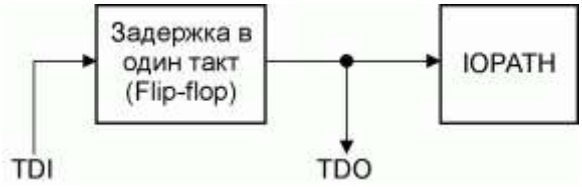


Рисунок 24.7 – Схема загрузки регистра IOPATH

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 задан 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. 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.

Инициализация блока CERBERUS

Используемые в микроконтроллере коды команд, приведенные в таблице 24.8, заносятся в регистр INSTRUCTION, когда JTAG модуль находится в состоянии «IR scan».

Таблица 24.8 – 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.

24.5 Блок CERBERUS

Блок CERBERUS является универсальным средством для подключения интерфейса JTAG. Ядро блока содержит сдвигатель, который управляется сигналами JTAG, поэтому является асинхронным к другим частям блока CERBERUS. Структурная схема блока CERBERUS приведена на рисунке 24.8.

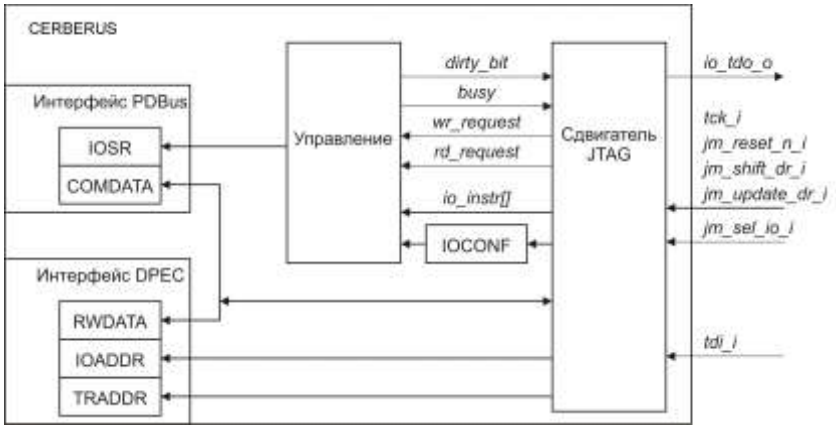


Рисунок 24.8 – Структурная схема блока CERBERUS

Режимы

Блок CERBERUS может использоваться для:

- чтения и записи ячейки памяти (режим чтения-записи),
- обмена данными с программой монитором, запущенной ЦПУ (режим коммуникации).

Защита от ошибок

Стандарт JTAG не включает в себя какую-либо защиту для последовательной передачи по выводам TDI, TDO и управления TMS. Тем не менее, защита от ошибки для входных данных по выводу TDI может быть реализована задержкой выходных данных на один цикл TCK на выводе TDO. Выходные данные могут быть перемещены дважды (также многократно) и затем сравнены для максимальной защиты от ошибок. Блок CERBERUS считается занятым, если запрошенные операции чтения или записи не завершены.

Все данные и адреса вводятся и выводятся, начиная с младшего бита. Внешнее отладочное устройство является мастером при всех передачах, инициализируя передачи в обоих направлениях.

Последовательная передача битов по выводам TDI и TDO

Когда блок CERBERUS выбран, 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 имеются во всех командах чтения-записи за исключением IO_CONFIG, когда предыдущая операция еще не окончилась.

Если тип команды «запись», то следующий TDI бит, после параллельного стартового TDO бита, используется, как первый бит данных. Далее следует передача данных. На

рисунке 24.9 приведена последовательность битового потока для выводов TDI и TDO в состоянии ShiftDR.

Если тип команды «чтение», то все TDI биты после команды игнорируются. После стартового бита на TDO начнут выдаваться биты. Если команда не определена или не выполняема, выдается неопределенное число битов занятости.

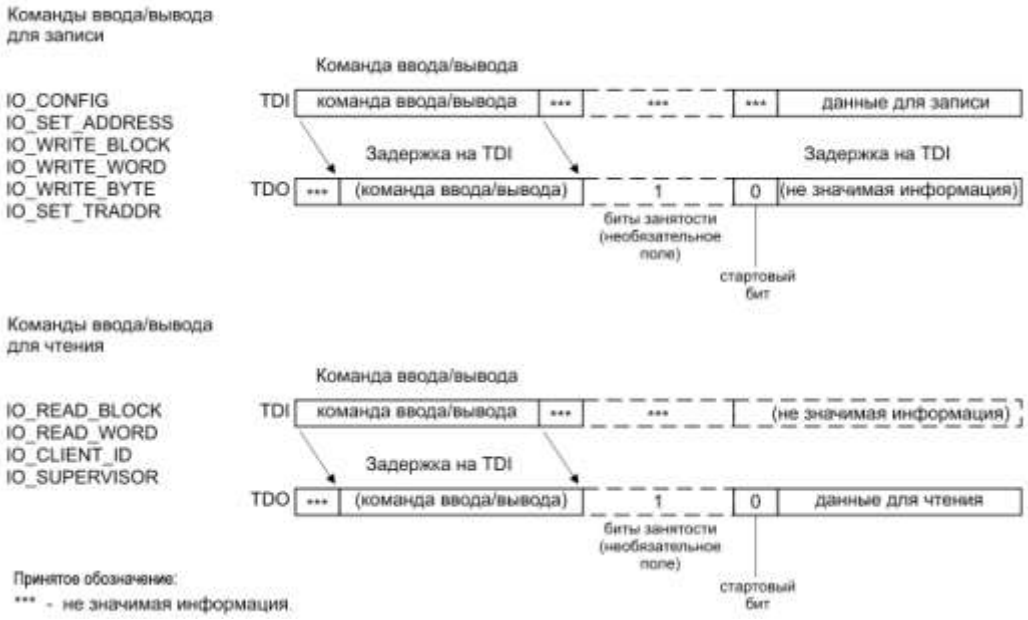


Рисунок 24.9 – Синтаксис последовательной передачи для выводов TDI и TDO в состоянии ShiftDR

Команды чтения-записи блока CERBERUS

В таблице 24.9 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 24.9 – 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 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. 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 24.10 показана взаимосвязь выводов TDI, TDO и содержания сдвигового регистра блока CERBERUS после ввода клиентской команды.

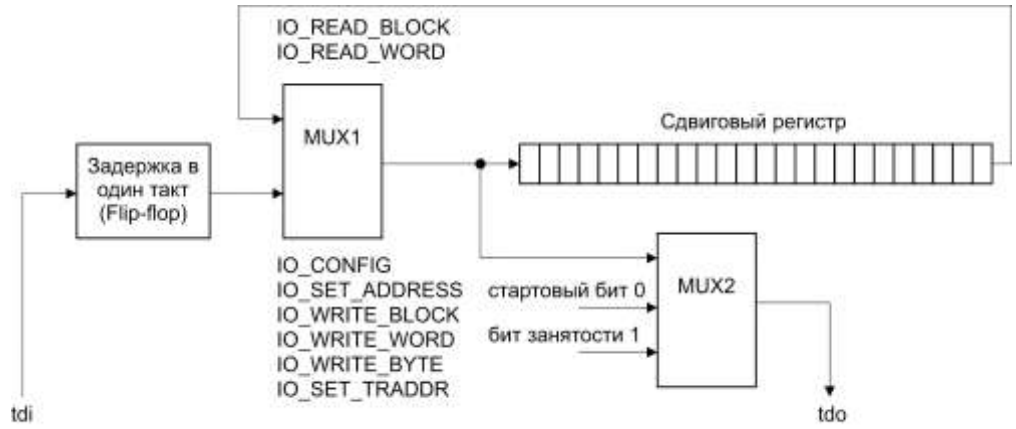


Рисунок 24.10 – Сдвиговый регистр в состоянии ShiftDR

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, 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. Результаты в обоих случаях одинаковые.

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.

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
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**1111111111111111111111111111111111111111111111111111111111
TDO: 0**1010**1110**0010110001001000**xxxxxxxx**0010110001001000**xxxxxxxx.

Регистр CLIENT_ID

Регистр позволяет внешнему отладочному устройству проверять аппаратные средства в автоконфигурационном режиме.

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).

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. 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 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. Регистр COMDATA является эквивалентом для режима коммуникации.

Регистр IOSR

Регистр используется в режиме коммуникации для запрещения работы блока CERBERUS со стороны ЦПУ из соображений безопасности и выполнения монитором трассировки. Регистр IOSR доступен только со стороны ЦПУ

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.

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.

Бит DBG_ON показывает, присутствует ли внешний отладчик. Он непосредственно управляется внутренним сигналом сброса JTAG.

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 трассировкой памяти. Необходимо отметить, что это можно использовать, только если внешний отладчик управляет блоком CERBERUS через интерфейс JTAG.

24.6 Режимы работы

Режим чтения-записи

Режим чтения-записи используется внешним интерфейсом (хостом) для чтения или записи ячеек памяти. В этом режиме 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 происходит, когда сброшен бит RW_ENABLED в регистре IOSR и внешний интерфейс устанавливает бит MODE регистра IOCONF.

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 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 ЦПУ, 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 ЦПУ. 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 ЦПУ для сообщения монитору, что внешний интерфейс желает передать IO_WRITE_WORD или принять IO_READ_WORD. Монитор опрашивает IOSR. 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 бита: CRSYNC, CWSYNC, CW_ACK и COM_SYNC. Биты CRSYNC, CWSYNC и COM_SYNC устанавливаются и сбрасываются аппаратно, однако, могут читаться монитором (ЦПУ). 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 обслуживаются 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.

Режим коммуникации является режимом по умолчанию после сброса микроконтроллера. Если блок CERBERUS находится в режиме чтения-записи, вход в

режим коммуникации осуществляется записью внешним интерфейсом нуля в бит MODE регистра IOCONF.

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).

Бит CRSYNC сообщает монитору ЦПУ, 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 ЦПУ 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.

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 ЦПУ 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 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.

Если монитор ЦПУ не обслуживает запрос чтения или записи регистра COMDATA, биты CWSYNC или CRSYNC могут быть сброшены битом COM_MODE_RST в регистре IOCONF.

Чтобы улучшить надежность канала коммуникации, следует 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. 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 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 единице, 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 ЦПУ к определенным областям памяти;

TINSTR – время цикла команды ЦПУ, 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 24.10.

Таблица 24.10 – 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 через порт JTAG, однако, напрямую записываются по адресам внешней шины. Данные захватываются отладчиком из внешней шины. Этот вид трассировки может быть разрешен только в режиме коммуникации и может применяться параллельно ему.

Для вызова передачи следует задать: MODE = 0b, TRIGGER_ENABLE = = 1b, EX_BUS_TRACE = 1b. Адрес по внешней шине имеет формат, приведенный на рисунке 24.11.

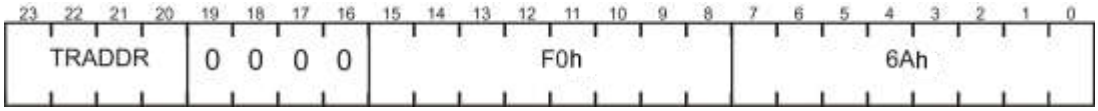


Рисунок 24.11 – Адрес по внешней шине

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.

Управляемая монитором трассировка является эквивалентом вызванных передач, однако, управляется монитором, запущенным ЦПУ. 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 кода на языке C 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 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.

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 ЦПУ, 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 запускает 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. Внешний интерфейс должен подтвердить это состояние командой IO_SUPERVISOR. Это делается для уведомления внешнего интерфейса о том, что, возможно, произошло не ожидаемое событие, и необходимо проверить канал связи с монитором.

Сброс JTAG всегда требует последующий сброс ЦПУ для гарантии того, что сдвигатель JTAG и управляющая часть блока CERBERUS находятся в определенных состояниях при всех условиях.

25 Система команд

Таблицы 25.1 и 25.2 дают краткую информацию о командах и кодах команд микроконтроллера 1887ВЕ9Т.

Таблица 25.1 – 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 25.2 – 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 25.3 – 25.6.

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 25.3 – 25.6:

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 25.3 – 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 25.3

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 25.4 – 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 25.4

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 25.5 – 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 25.6 – 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	

Команда ATOMIC и EXT-команды (расширенные)

Команда ATOMIC и EXT-команды (EXTR, EXTP, EXTS, EXTPR, EXTSR) запрещают стандартные прерывания класса A, 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 класса B, 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.

Подробное описание команд микроконтроллера 1887ВЕ9Т приводится в приложении Г настоящего руководства.

26 Режимы энергосбережения

Режимы пониженного энергопотребления Idle и PowerDown реализуются блоком управления энергосбережением. Включение режимов осуществляется командами IDLE и PWRDN при условии, что на входе NMI# микроконтроллера удерживается низкий уровень сигнала.

В режиме пониженного энергопотребления поддерживается высокий уровень готовности системы к возвращению в режим нормальной работы. Внешние сигналы и события могут сканироваться (с пониженной скоростью) при периодической активности ЦПУ и выбранных периферийных устройств, которые возвращаются к экономичному режиму энергопотребления после кратковременного режима нормальной работы.

Режимы Idle и PowerDown могут быть прерваны сигналом внешнего сброса микроконтроллера и сигналами внешних прерываний.

Заключение

В настоящем руководстве КФДЛ.431295.052 был представлен обзор архитектуры и функционального построения 16-разрядного RISC микроконтроллера 1887ВЕ9Т. Приведена система команд и указаны особенности применения.

Значения параметров, приведенные в настоящем руководстве, являются справочными. Значения электрических параметров ИС приведены в техническом условии АЕЯР.431280.904 на изделие.

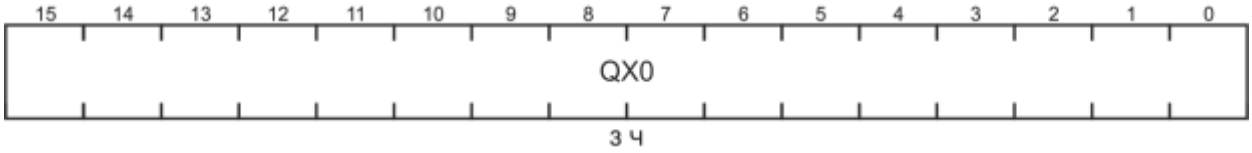
Руководство может служить практическим пособием по применению микроконтроллеров 1887ВЕ9Т для разработчиков систем на их основе и программистов.

Приложение А (обязательное) **Регистры микроконтроллера**

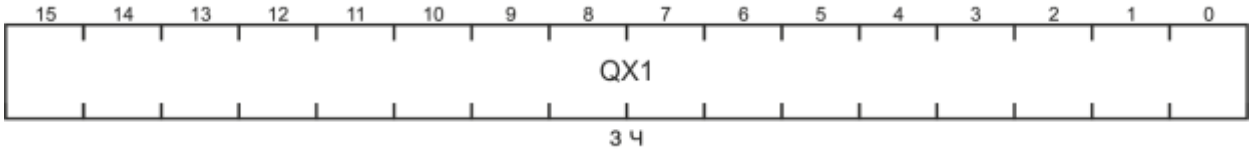
А.1 Регистры ЦПУ

Таблица А.1.1 – Регистры смещения для нулевого и первого указателей адреса

QX0



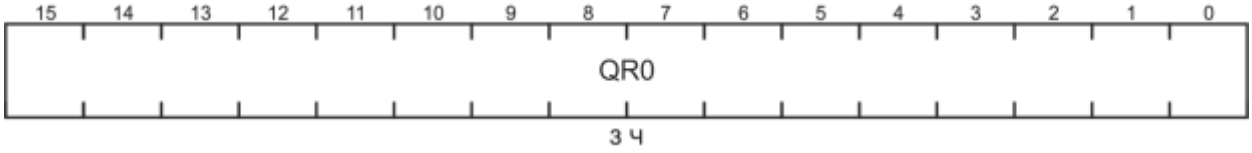
QX1



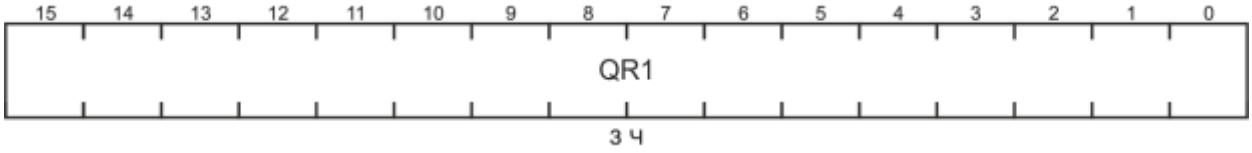
Поле	Бит	Описание
QX0/QX1	15–0	16-разрядное смещение для указателя адреса IDX0/IDX1
Примечание – Поскольку поля QR0 и QR1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю		

Таблица А.1.2 – Регистры смещения для нулевого и первого указателей адреса

QR0



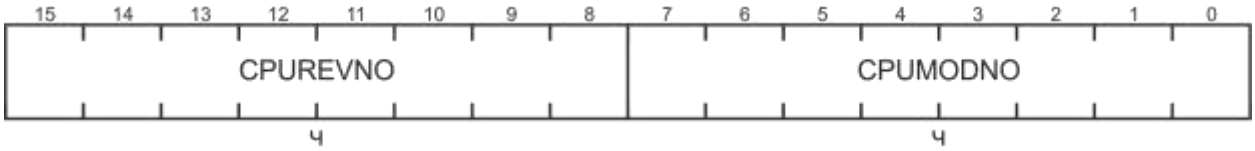
QR1



Поле	Бит	Описание
QR0	15–0	16-разрядное смещение для указателя адреса GPR
Примечание – Поскольку поля QR0 и QR1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю		

Таблица А.1.3 – Регистр идентификации ЦПУ

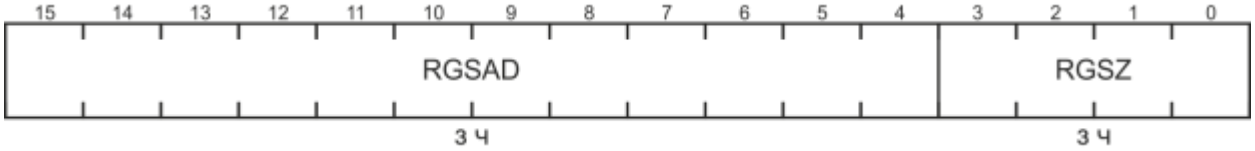
CPUID



Поле	Биты	Описание
CPUREVNO	15–8	Номер микроконтроллера
CPUMODNO	7–0	Версия микроконтроллера

Таблица А.1.4 – Регистр выбора адреса шины XBUS (x – номер окна от 1 до 6)

XADRSx



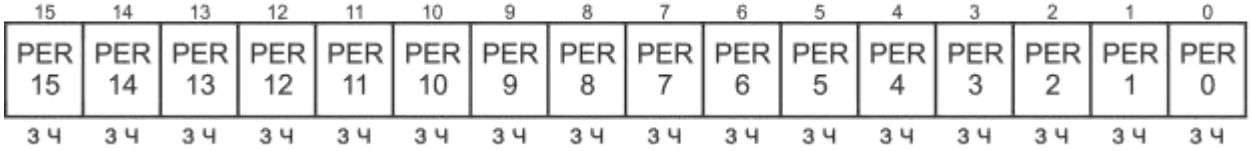
Поле	Бит	Описание
RGSAD	15–4	Выбор стартового адреса адресного окна x
RGSZ	3–0	Выбор размера адресного окна x (см. в таблице А.1.5)

Таблица А.1.5 – 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
Ch-Fh	Зарезервировано			

Таблица А.1.6 – Регистр управления X-периферией

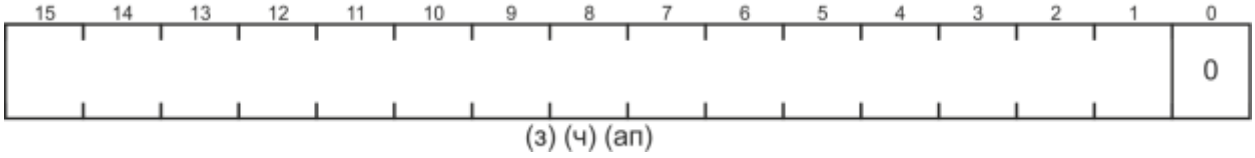
XPERCON



Поле	Биты	Описание
PER3	3	Бит включения сигнала XCS4 для блока ШИМ (CAPCOM6)
PER2	2	Бит включения сигнала XCS3 для интерфейса ГОСТ Р 52070-2003 8 Кбайт
PER1	1	Бит включения сигнала XCS2 для XRAM 16 Кбайт
PER0	0	Бит включения сигнала XCS1 для контроллера CAN
PER15–PER4	15–4	Не используются
Примечание – Для всех битов: 0 – не активен, 1 – активен.		

Таблица А.1.7 – Указатель команд

IP сброс: 0000h



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.1.8 – Регистр начальной конфигурации внешней шины

RP0H

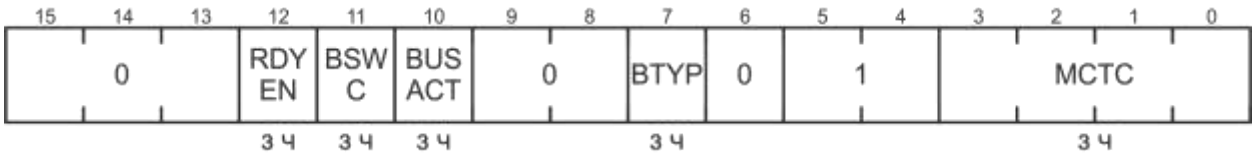
Поле	Биты	Описание	
USERCONF	7–5	Код пользователя. Нефункциональные биты	
SALSEL	4–3	Разрядность внешней шины. Поле показывает количество разрядов старшей части адреса	
		00	Сегментная часть адреса A19–A16. Адресное пространство – 1 Мбайт
		01	Адресное пространство – 64 Мбайт
		10	Сегментная часть адреса A23–A16. Адресное пространство – 16 Мбайт
		11	Сегментная часть адреса A17, A16. Адресное пространство – 256 Кбайт

Окончание таблицы А.1.8

Поле	Биты	Описание	
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.1.9 – Регистр управления шиной XBUS (x от 1 до 6)

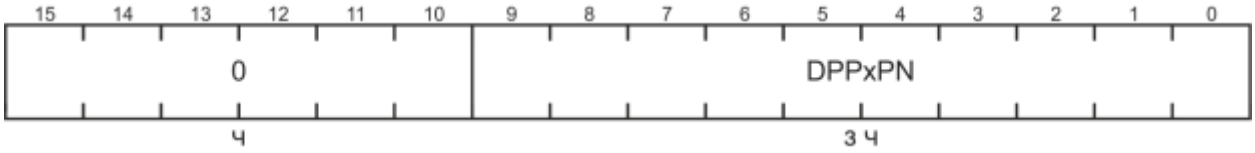
XBCONx



Поле	Бит	Описание	
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
BUSACT	10	Управление активностью шины 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 разрядности демультиплексной шины XBUS	
		0	8-разрядная
		1	16-разрядная
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.1.10 – Указатель страницы данных (x от 0 до 3)

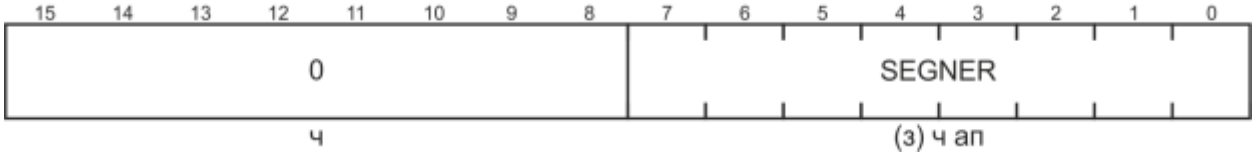
DPPx



Поле	Биты	Описание
DPPxPN	9–0	Номер страницы данных в пределах выбранного регистра DPPx
0	15–10	Зарезервировано. Всегда возвращает нули

Таблица А.1.11 – 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.1.12 – Старший и младший регистры умножения/деления

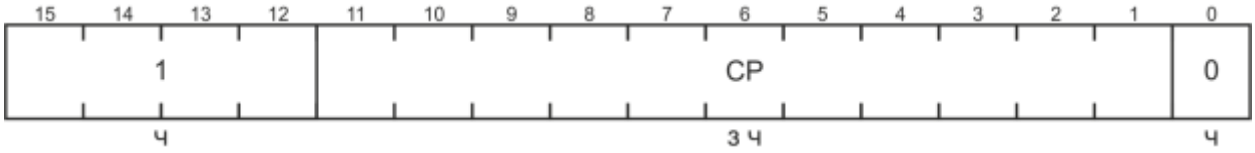
MDH

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.1.13 – Контекстный указатель

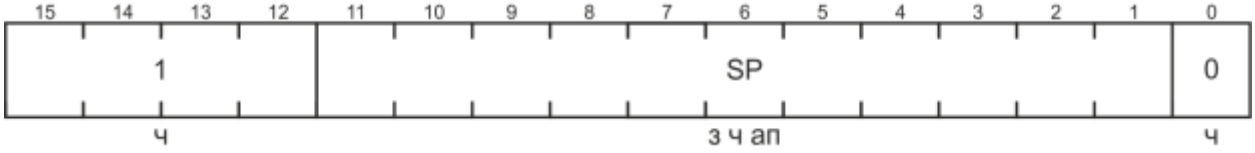
CP



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.1.14 – Указатель стека

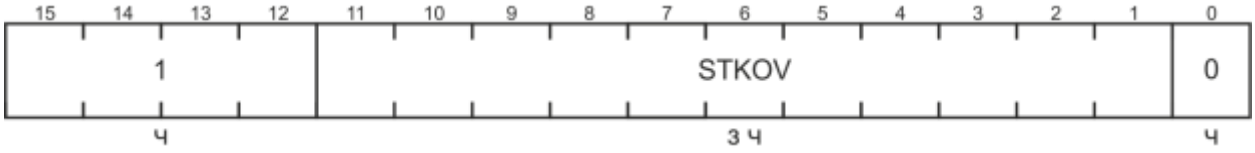
SP



Поле	Биты	Описание
SP	11–1	Указатель вершины системного стека. Выровнен пословно.
1	15–12	Зарезервировано. При чтении всегда возвращает значение единицы

Таблица А.1.15 – Указатель переполнения стека

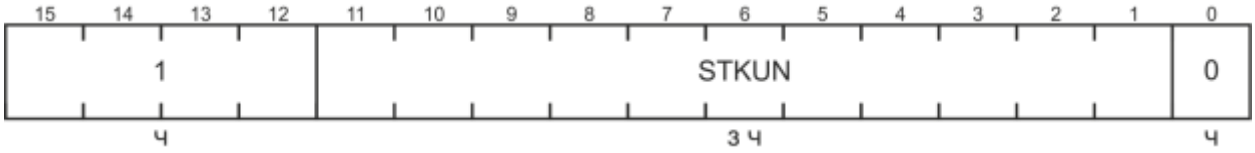
STKOV



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.1.16 – Указатель опустошения стека

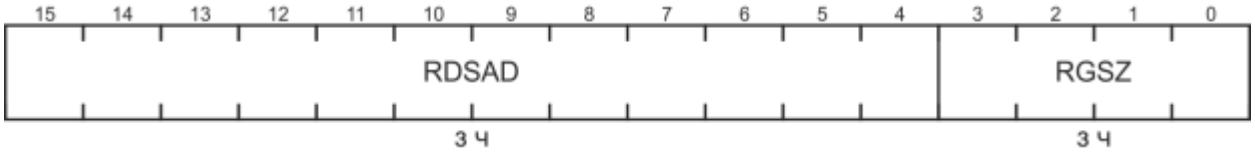
STKUN



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.1.17 – Регистр выбора адреса (x от 1 до 4)

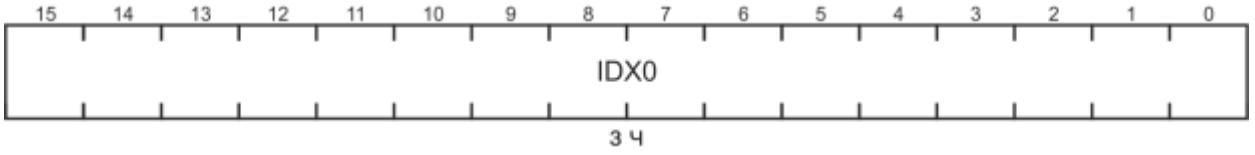
ADDRSELx



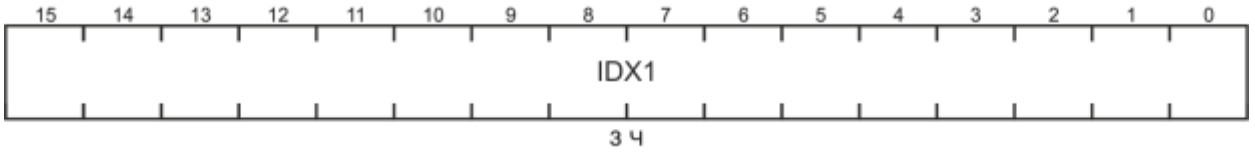
Поле	Бит	Описание
RDSAD	15–4	Биты разрядов A23 – A12 начального адреса окна
RGSZ	3–0	Биты управления размером адресного окна, управляемого парой регистров BUSCONx – ADDRSELx

Таблица А.1.18 – Регистры нулевого и первого указателей адреса

IDX0



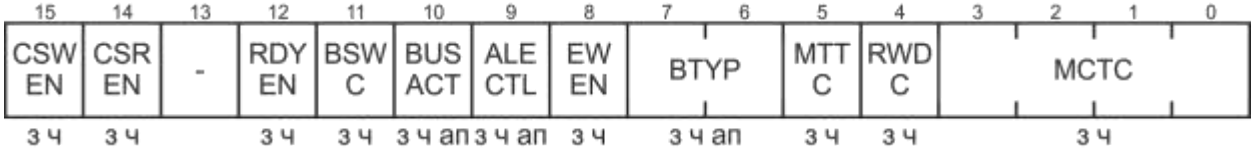
IDX1



Поле	Бит	Описание
IDX0/ IDX1	15–0	Изменяемая часть указателя адреса
Примечание – Поскольку поля IDX0 и IDX1 могут содержать только четные числа, то нулевые биты регистров всегда равны нулю.		

Таблица А.1.19 – Регистры управления шиной (x от 0 до 4)

BUSCONx



Поле	Бит	Описание	
CSWEN	15	Бит разрешения сигнала записи WR#, WRL#, WRH# (см. таблицу А.1.20)	
		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.1.20)	
		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
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

Окончание таблицы А.1.19

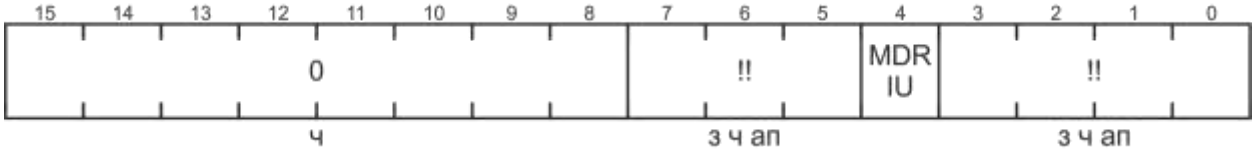
Поле	Биты	Описание	
MTTC	5	Бит управления третьим состоянием, то есть управления временем, необходимым для освобождения шины внешним устройством при чтении:	
		0	Формирование одного цикла ожидания (2TCL)
		1	Нет дополнительного цикла ожидания
RWDC	4	Бит управления задержкой сигналов RD# и WR#:	
		0	Задержка 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.1.20 – 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.1.21 – Регистр управления умножением/делением

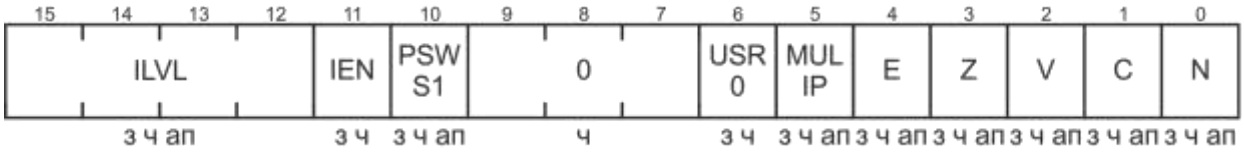
MDC



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.1.22 – Слово состояния процессора

PSW



Поле	Бит	Описание	
ILVL	15–12	Уровень приоритета ЦПУ. Изменяется аппаратно при входе в подпрограмму обслуживания прерываний. Для запрещения прерываний ILVL может быть программно изменено. 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	
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	

Продолжение таблицы А.1.22

Поле	Бит	Описание
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
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

Окончание таблицы А.1.22

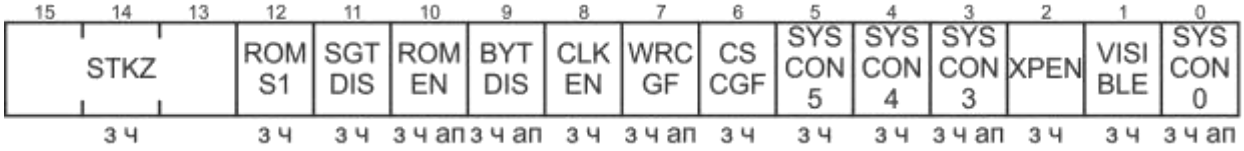
Поле	Бит	Описание
C	1	Флаг переноса. После операции сложения он показывает наличие переноса из старшего значащего бита в вычисляемом байте или слове. После вычитания или сравнения C-флаг показывает заимствованный бит, который представляет собой отрицательную логическую величину. Это означает, что C-флаг 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
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.1.23 – Регистр управления системы

SYSCON



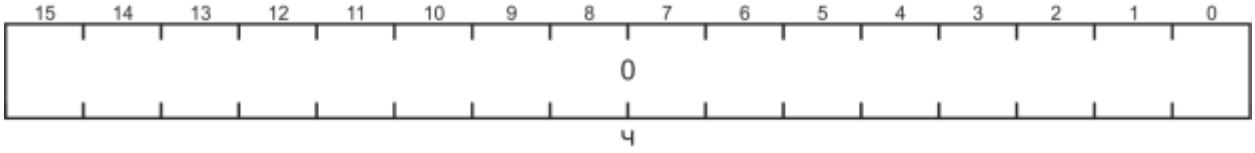
Поле	Биты	Описание	
STKZ	15–13	Биты выбора размера системного стека в DPRAM	
		000	256 слов (FBFEh…FA00h)
		001	128 слов (FBFEh…FB00h)
		010	64 слова (FBFEh…FB80h)
		011	32 слова (FBFEh…FBC0h)
		100	512 слов (FBFEh…F800h)
		101	Зарезервировано
		110	Зарезервировано
		111	1536 слов (FDFEh …F200h)
ROMS1	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#
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

Окончание таблицы А.1.23

Поле	Биты	Описание	
SYSCON5–SYSCON3	5–3	Зарезервировано	
XPEN	2	Бит разрешения доступа к шине XBUS (X-периферии)	
		0	Запрещен
		1	Разрешен
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.1.24 – Регистр постоянного нуля

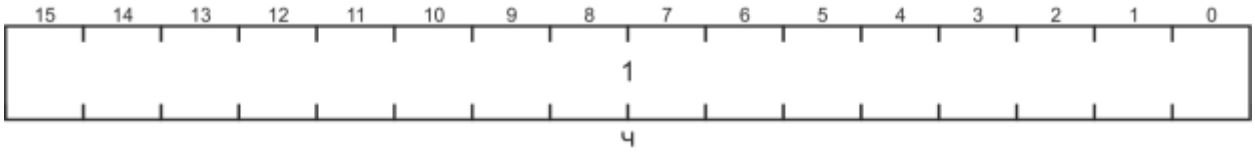
ZEROS



Поле	Биты	Описание
0	15–0	Всегда возвращает нули

Таблица А.1.25 – Регистр постоянной единицы

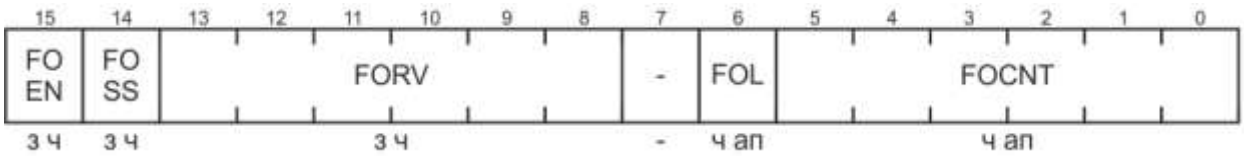
ONES



Поле	Биты	Описание
1	15–0	Всегда возвращает единицы

Таблица А.1.26 – Регистр управления выходным тактовым сигналом

FOCON



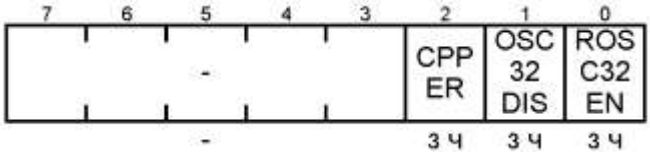
Поле	Бит	Описание
FOEN	15	Разрешение вывода сигнала программируемой частоты

Окончание таблицы А.1.26

Поле	Бит	Описание	
FOSS	14	Бит включения дополнительного делителя на два	
		0	Включен
		1	Выключен
FORV	13–8	Значение коэффициента деления частоты опорного сигнала FMC	
FOL	6	Индикатор обнуления счетчика частоты. Переключается при каждом обнулении.	
FOCNT	5–0	Значение счетчика выходной частоты	
–	7	Зарезервировано	

Таблица А.1.27– Регистр управления вспомогательным генератором

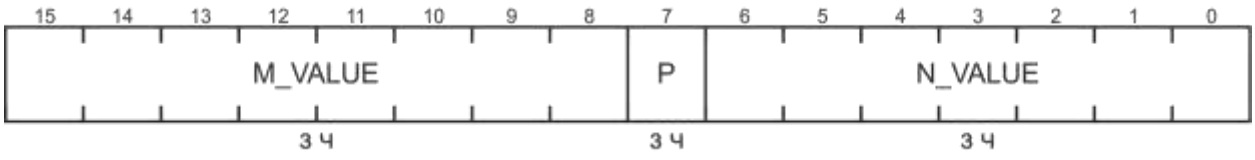
GENCON



Поле	Бит	Описание	
CPPER	2	Коэффициент деления частоты синхросигнала FMC для формирования синхросигнала периферии	
		0	1/2
		1	1/4
OSC32DIS	1	Бит отключения вспомогательного осциллятора	
		0	Включен
		1	Выключен
ROSC32EN	0	Бит подключения внутреннего резистора в цепи обратной связи схемы вспомогательного осциллятора	
		0	Отключен
		1	Подключен
–	7-3	Зарезервировано	

Таблица А.1.28– Регистр управления вспомогательным генератором

PLLCON

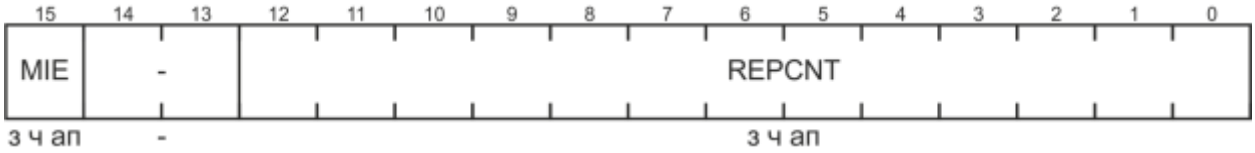


Поле	Бит	Описание	
M_VALUE	15-8	Коэффициент деления входного каскада	
P	7	Коэффициент деления выходного каскада	
		0	Делитель выключен
		1	1/4
N_VALUE	6-0	Коэффициент деления входного каскада	

A.2 Регистры блока MAC

Таблица А.2.1 – Регистр повторения блока MAC

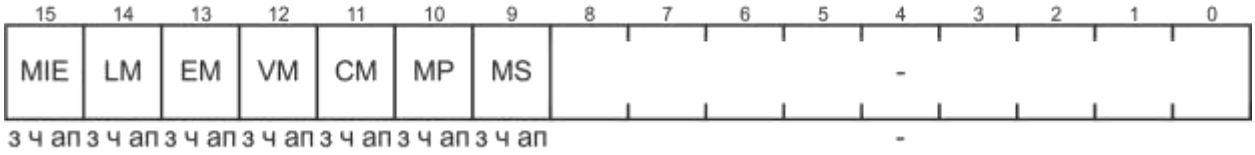
MRW



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.2.2 – Регистр управления блока MAC

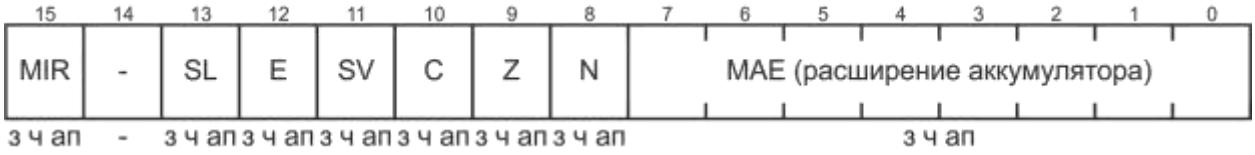
MCW



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.2.3 – Статусный регистр блока MAC

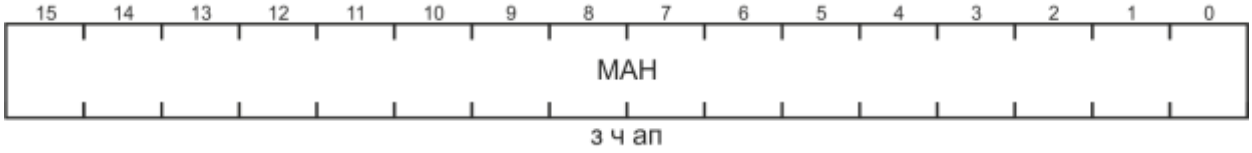
MSW



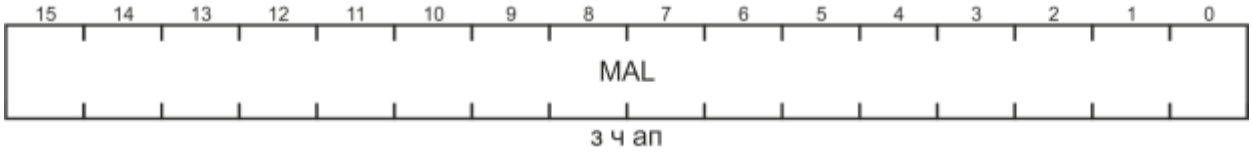
Поле	Бит	Описание	
MIR	15	Запрос на прерывание блока MAC	
		0	Нет запроса на прерывание блока MAC
		1	Запрос на прерывание блока MAC
SL	13	Флаг ограничения	
		0	Не произошло автоматического 32-битного ограничения
		1	Произошло автоматическое 32-битное ограничение
E	12	Флаг расширения	
		0	MAE не содержит 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.2.4 – Регистры старшего и младшего байт аккумулятора

MAH



MAL

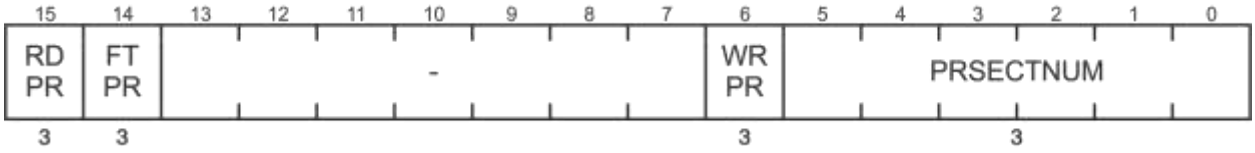


Окончание таблицы А.2.4

Поле	Биты	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.2.5 – Регистр конфигурации защиты флеш-памяти

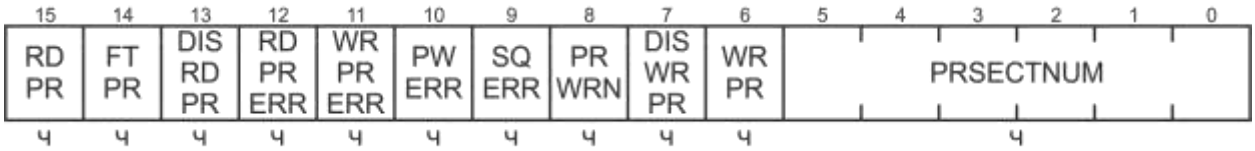
PROCON



Поле	Бит	Описание
PDPR	15	Запрета чтения данных из флеш-памяти внешней программой
FTPR	14	Запрет выполнения кода из флеш-памяти, вызванного внешней программой
WRPR	6	Запрет записи и стирания всей флеш-памяти
RESECT NUM	5-0	Число секторов 8-го сегмента, защищенных от записи и стирания (начиная с 1-ого)
–	13–7	Зарезервированы

Таблица А.2.6 – Регистр состояния флеш-памяти

FSR

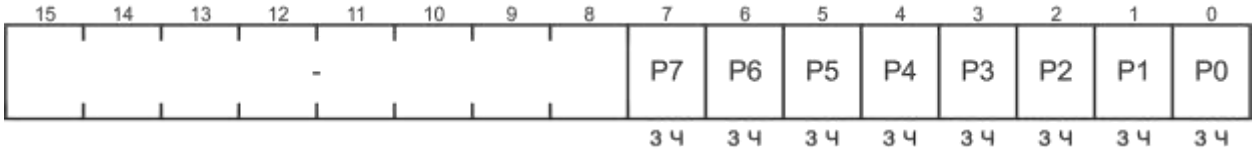


Поле	Бит	Описание
RDPR	15	Флаг включенной защиты от чтения данных
FTPR	14	Флаг включенной защита от выборки команд
DISRDPR	13	Флаг приостановки защиты от чтения данных и выборки команд
RDPRERR	12	Флаг попытки прочитать защищенные данные или команды. Сбрасывается после чтения.
WRPRERR	11	Флаг попытки запрограммировать или стереть защищенную область. Сброс после чтения.
PWERR	10	Флаг введения неверного пароля. Сброс при аппаратном сбросе.
SQERR	9	Флаг ошибки в командной последовательности. Сброс после чтения.
PRWRN	8	Флаг неравенства как минимум одной копии PROCON и PW1-PW4
DISWRPR	7	Флаг отключения защиты от записи/стирания.
WRPR	6	Флаг включенной защиты от записи/стирания всей флеш-памяти.
PRSECT NUM	5-0	Число секторов 8-го сегмента, защищенных от записи /стирания (начиная с 1-ого)

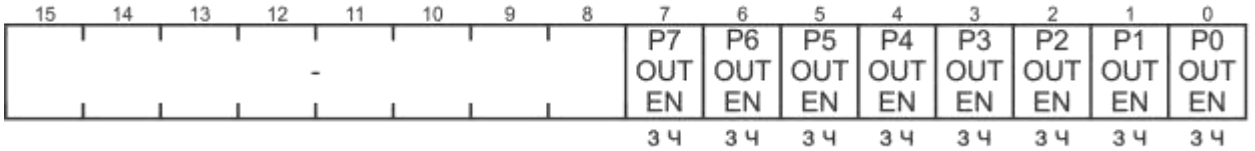
A.3 Регистры портов

Таблица А.3.1 – Регистры порта P0

P0H / P0L



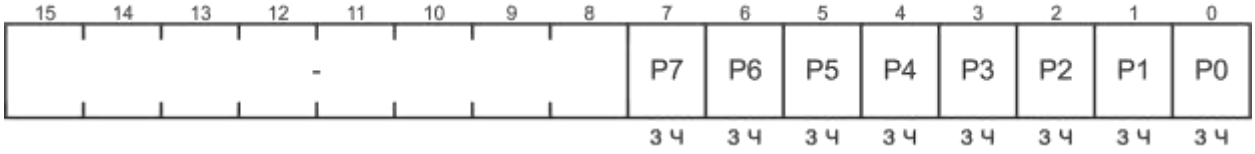
DP0H / DP0L



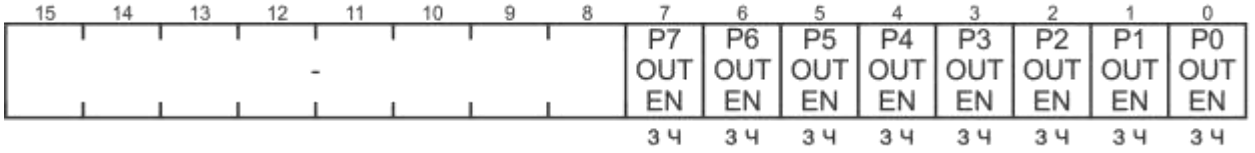
1 Биты P7, P6, …, P0 являются битами состояния выводов порта.
2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то соответствующий вывод работает как вход, иначе – как выход. Биты с 15 по 8 регистров не используются.

Таблица А.3.2 – Регистры порта P1

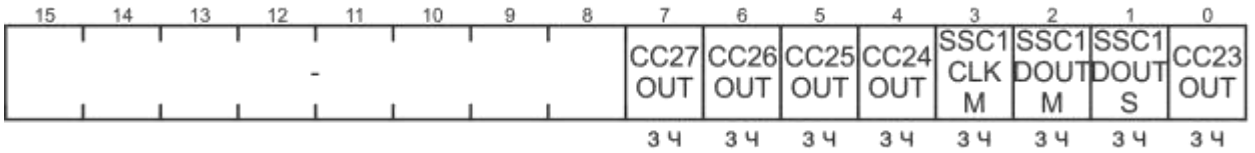
P1H / P1L



DP1H / DP1L

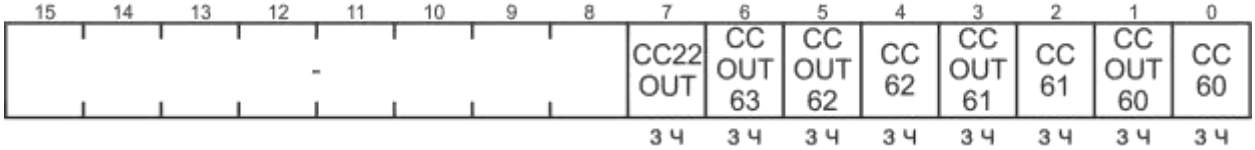


ALTSEL0P1H



Окончание таблицы А.3.2

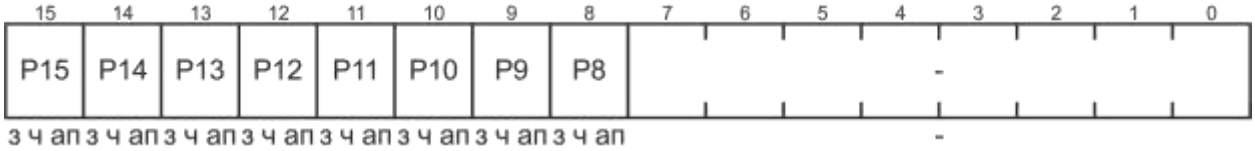
ALTSEL0P1L



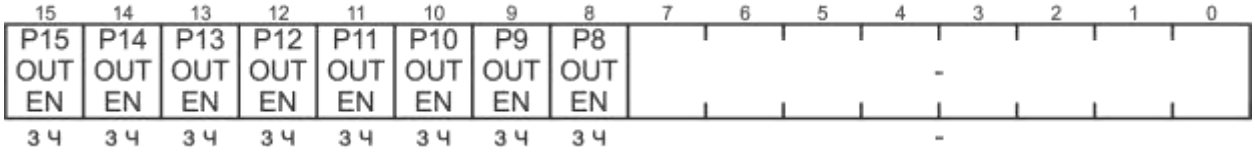
1 Биты P7, P6, …, P0 являются битами состояния выводов порта.
2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.
3 Биты с 7 по 0 регистра ALTSEL0P1H/L являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит. Биты с 15 по 8 регистров не используются.
Назначение битов:
- CC27OUT – CC22OUT подключают выходы блока CAPCOM2;
- CCOUT63 – CCOUT60 и CC62 – CC60 подключают выходы блока CAPCOM6;
- SSC1CLKM, SSC1DOUTM и SSC1DOUTS подключают выводы тактового сигнала, передаваемых данных мастера и ведомого блока SSC1.

Таблица А.3.3 – Регистры порта P2

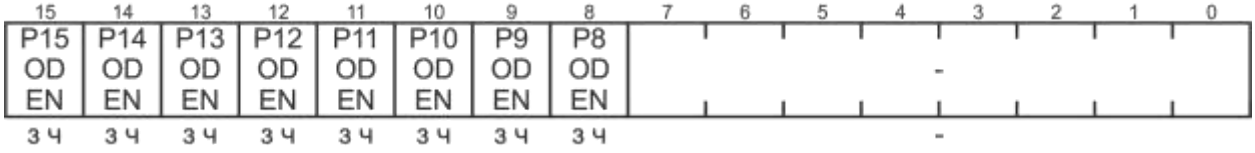
P2



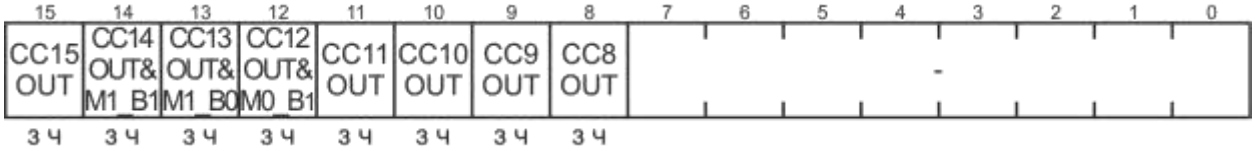
DP2



ODP2



ALTSEL0P2



Окончание таблицы А.3.3

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. Биты с 7 по 0 регистров не используются.
Назначение битов:
- CC15OUT, CC11OUT – CC8OUT подключают выходы блока CAPCOM1;
- CC14OUT&M1_B1, CC13OUT&M1_B0 и CC12OUT&M0_B1 подключают, объединенные логически по И, выходы блока CAPCOM1 и выходы блокировки нулевого и первого каналов контроллера интерфейса по ГОСТ Р 52070.

Таблица А.3.4 – Регистры порта P3

P3

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

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK OUT	-	SSC0 CLK M	BHE#	ASC0 RXD	ASC0 TXD	SSC0 DOUT M	SSC0 DOUT S	-	M1_ TXDN 1	M1_ TXD1	ASC2 _TXD	T3_ OVFL TGL	ASC2 _RXD	ASC1 _RXD	ASC1 _TXD
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

ALTSEL1P3

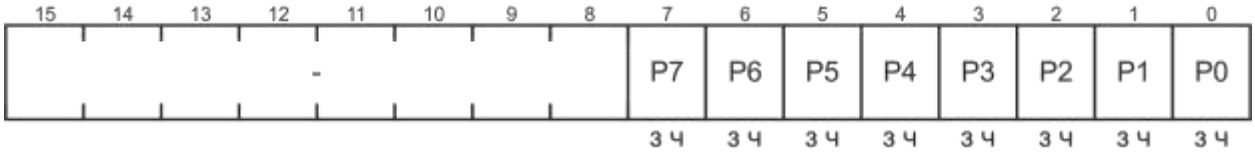
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FOUT	SSC2 CLK M	-	WRH#	-				SSC2 DOUT M	SSC2 DOUT S	ASC1 TXD	M1_ TXDN 0	M1_ TXD0	-	T6_ OVFL TGL	-
з ч	з ч		з ч					з ч	з ч	з ч	з ч	з ч		з ч	

Окончание таблицы А.3.4

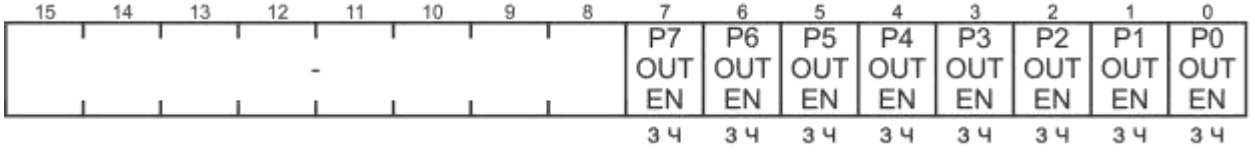
1 Биты P15, P14, …, P0 являются битами состояния выводов порта.
2 Биты P15OUTEN, P14OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.
3 Биты 13 и с 11 по 0 регистра ODP3 являются битами включения режима открытого стока выводов. 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. Биты, отмеченные символом «-», не используются.
Назначение битов:
- CLKOUT, FOUT подключают выводы блока генератора частоты;
- BHE#, WRH# подключают выводы контроллера внешней шины.
- SSC0CLKM, SSC0DOUTM и SSC0DOUTS подключают выводы тактового сигнала, передаваемых данных мастера и ведомого блока SSC0.
- SSC2CLKM, SSC2DOUTM и SSC2DOUTS подключают выводы блока SSC2.
- ASC0RXD, ASC0TXD, ASC1RXD, ASC1TXD, ASC2RXD и ASC2TXD подключают выводы блоков ASC0, ASC1 и ASC2.
- M1_TXDN1, M1_TXD1, M1_TXDN0 и M1_TXD0 подключают прямые и инверсные выходы данных канала 1 контроллера интерфейса по ГОСТ Р 52070;
- T3_OVFLTGL и T6_OVFLTGL подключают выходы сигналов переполнения таймеров общего назначения T3 и T6.

Таблица А.3.5 – Регистры порта P4

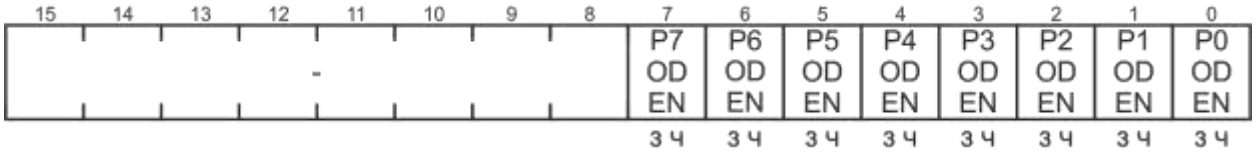
P4



DP4



ODP4



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. Биты, отмеченные символом «-», не используются.

Таблица А.3.6– Регистр порта P5

P5

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч

Примечание – Биты P15, P14, …, P0 являются битами состояния выводов порта.

Таблица А.3.7 – Регистры порта P6

P6

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

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

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

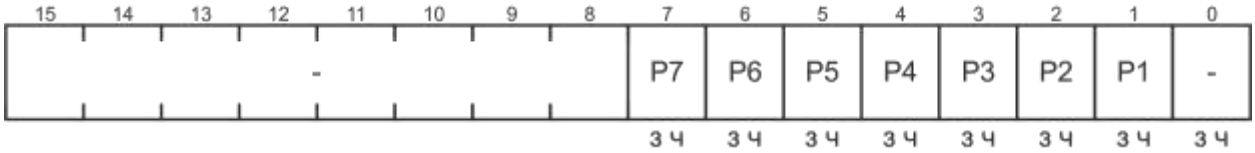
ALTSEL0P6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								CC7 OUT	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

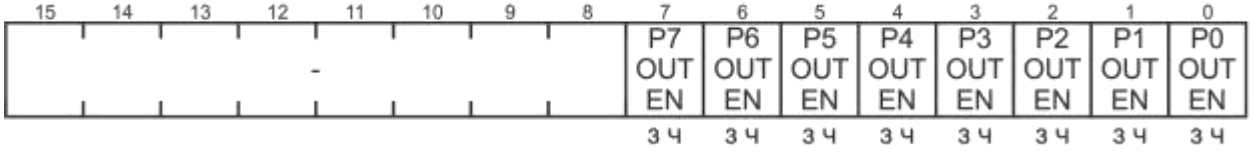
1 Биты P7, P6, …, P0 являются битами состояния выводов порта.
2 Биты P7OUTEN, P6OUTEN, …, P0OUTEN являются битами задания направления выводов. Если бит сброшен, то вывод работает как вход, иначе – как выход.
3 Биты с 7 по 0 регистра ODP6 являются битами включения режима открытого стока выводов. Для включения 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. Биты, отмеченные символом «-», не используются.
Назначение битов:
- CC7OUT – CC0OUT подключают выходы блока CAPCOM1.

Таблица А.3.8 – Регистры порта P7

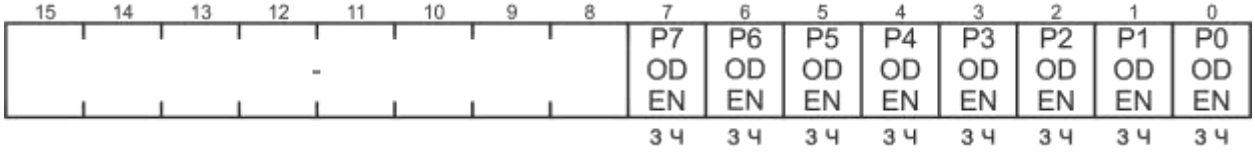
P7



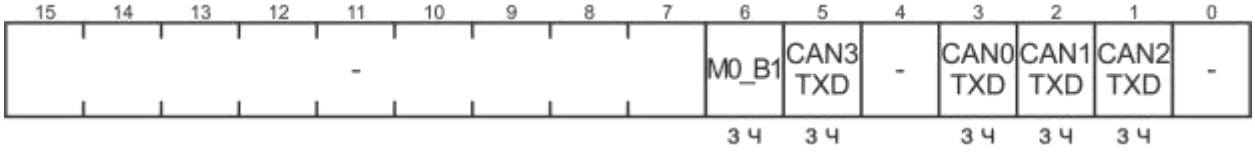
DP7



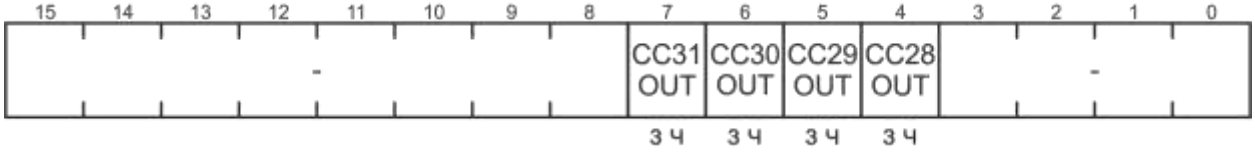
ODP7



ALTSEL0P7



ALTSEL1P7



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 ODP7 являются битами включения режима открытого стока выводов. Для включения режима вывода следует установить соответствующий бит.

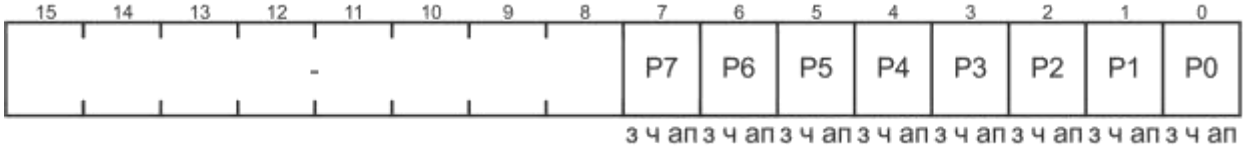
4 Биты регистров ALTSEL0P7 и ALTSEL1P7 являются битами включения альтернативных функций выводов. Для включения альтернативной функции следует установить соответствующий бит. Биты, отмеченные символом «-», не используются.

Назначение битов:

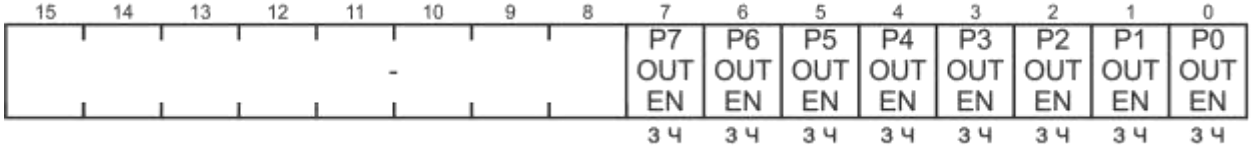
- M0_B1 подключает первый выход блокировки канала 0 контроллера интерфейса по ГОСТ Р 52070;
- CAN3TXD – CAN0TXD подключают выходы данных узлов контроллера интерфейса CAN;
- CC31OUT – CC28OUT подключают выходы блока CAPCOM2.

Таблица А.3.9– Регистры порта P8

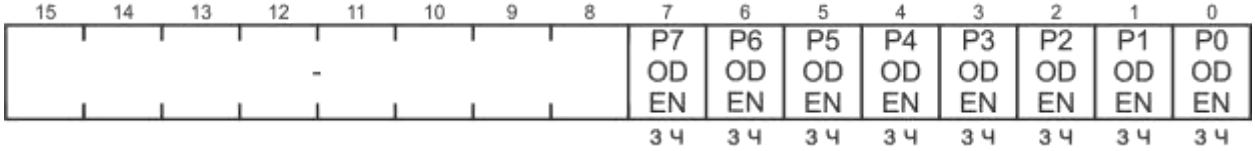
P9



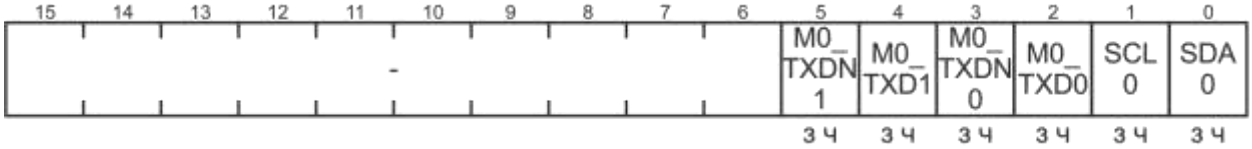
DP9



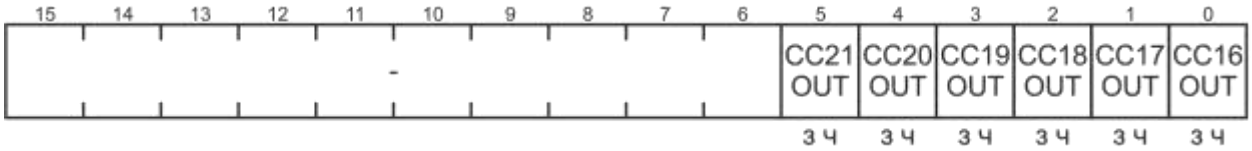
ODP9



ALTSEL0P9



ALTSEL1P9



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. Биты, отмеченные символом «-», не используются.

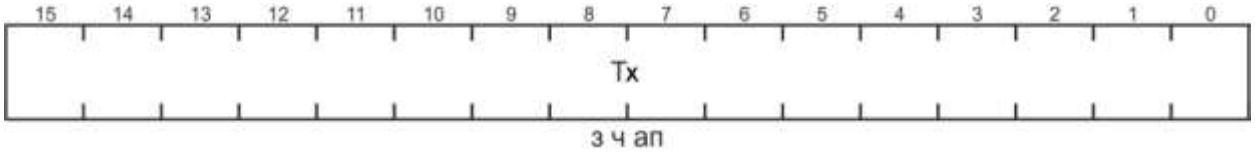
Назначение битов:

- M0_TXDN1, M0_TXD1, M0_TXDN0 и M0_TXD0 подключают прямые и инверсные выходы данных канала 0 контроллера интерфейса по ГОСТ Р 52070;
- SCL0 и SDA0 подключают выводы тактового сигнала и данных блока I2C;
- CC21OUT – CC16OUT подключают выходы блока CAPCOM2.

A.4 Регистры блоков таймеров GPT1 и GPT2

Таблица А.4.1 – Регистр таймера Tx (x от 2 до 6)

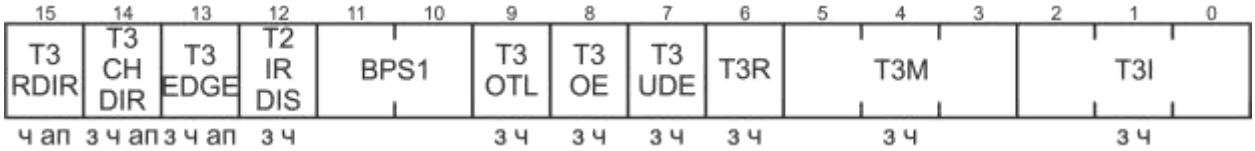
Tx



Поле	Бит	Описание
Tx	15–0	Текущее значение таймера Tx

Таблица А.4.2 – Регистр управления таймера T3

T3CON



Поле	Бит	Описание	
T3RDIR	15	Индикатор направления счета таймера	
		0	Вверх (инкрементирование)
		1	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 не обнаружен
		1	Фронт обнаружен
		Бит должен очищаться программно	
BPS1	12–11	Делитель частоты.	
		00	fosc/8
		01	fosc/4
		10	fosc/32
		11	fosc/16
T3OTL	10	Бит выходной защелки таймера. Этот бит переключается каждый раз при переполнении/ опустошении таймера T3. Бит может быть установлен/сброшен программно	

Окончание таблицы А.4.2

Поле	Бит	Описание	
T3OE	9	Бит разрешения вывода сигнала о переполнении/опустошении таймера	
		0	Не детектируется
		1	Детектируется по сигналу на линии T3OUT
T3UDE	8	Бит выбора способа контроля направления счета таймера	
		0	Программно
		1	Посредством линии T3EUD
T3UD	7	Бит установки направления счета таймера	
		0	Вверх
		1	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. Коды для режимов таймера и внешнего управления в таблице А.4.3, для режима счетчика – в таблице А.4.4, для режима внешнего инкрементирования – в таблице А.4.5	

Таблица А.4.3 – 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.4.4 – Режим счетчика Tx (x = 2, 3, 4)

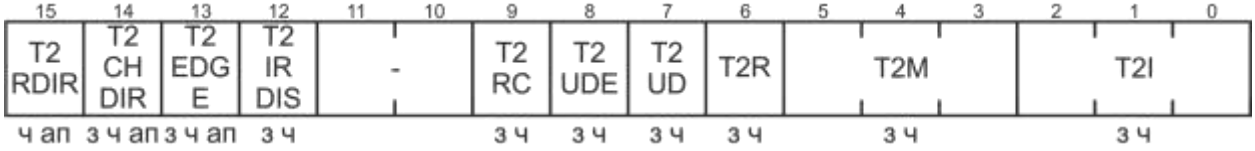
TxI	Ожидаемый фронт входного сигнала для инициирования переключения таймера
X00b	Счетчик Tx отключен
001b	Положительный фронт на входе TxIN
010b	Отрицательный фронт на входе TxIN
011b	Положительный и отрицательный фронты на входе TxIN
101b	Положительный фронт на входе T3OTL
110b	Отрицательный фронт на входе T3OTL
111b	Положительный и отрицательный фронты на входе T3OTL

Таблица А.4.5 – Режим внешнего инкрементирования Т3

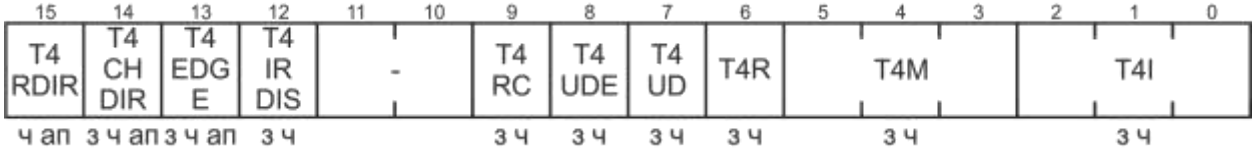
T3I	Выбор фронта для положительного или отрицательного счета
000b	Счетчик не считает
001b	Любой фронт (положительный или отрицательный) на T3IN
010b	Любой фронт (положительный или отрицательный) на T2EUD
011b	Любой фронт (положительный или отрицательный) на любом из входов T3IN и T3EUD
1XXb	Зарезервировано. Не использовать

Таблица А.4.6 – Регистры управления таймерами Т2 и Т4

T2CON



T4CON



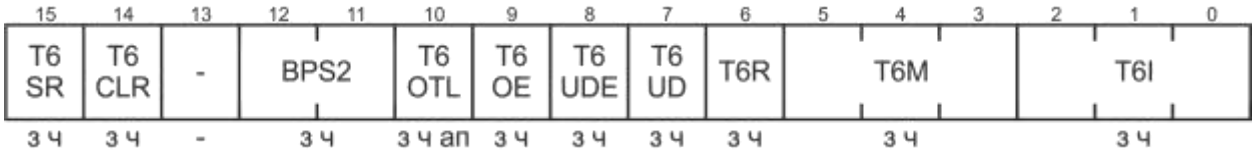
Поле	Бит	Описание	
T2RDIR T4RDIR	15	Индикатор направления счета таймера	
		0	Вверх
		1	Вниз
T2CHDIR T4CHDIR	14	Индикатор изменения направления счета таймера. Этот бит устанавливается каждый раз при изменении направления счета таймера	
		0	Нет изменения направления счета
		1	Направления счета изменилось
		Бит должен очищаться программно	
T2EDGE T4EDGE	13	Индикатор обнаружения фронта входного сигнала таймера	
		0	Не обнаружен
		1	Обнаружен
		Бит должен очищаться программно	
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	Включается собственным битом работы T2R/T4R
		1	Включается битом основного таймера Т3

Окончание таблицы А.4.6

Поле	Биты	Описание	
T2UDE T4UDE	8	Бит выбора способа контроля направления счета таймера	
		0	Программно
		1	Посредством линии T2EUD/T4EUD
T2UD T4UD	7	Бит установки направления счета таймера (при T2UDE/T4UDE = 0)	
		0	Вверх
		1	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. Коды для режима таймера в таблице А.4.3, для режима внешнего управления таймером – в таблице А.4.4, для режима счетчика – в таблице А.4.4, для режима внешнего инкрементирования – в таблице А.4.5	
–	11–10	Зарезервировано. Не использовать	

Таблица А.4.7 – Регистр управления таймером Т6

T6CON



Поле	Бит	Описание	
T6SR	15	Бит разрешения загрузи таймера Т6 значением из регистра CAPREL	
		0	Запрещена
		1	Разрешена
T6CLR	14	Бит очистки таймера Т6	
		0	Не очищается при захвате
		1	Очищается после захвата

Окончание таблицы А.4.7

Поле	Биты	Описание	
BPS2	12–11	Делитель блока таймеров GPT2. Задает входную частоту сигнала тактирования. Дополнительно частота входного сигнала тактирования может быть изменена (см. T6I) для режима таймера, режима счетчика и режима внешнего управления	
		00	fosc/4
		01	fosc/2
		10	fosc/16
		11	fosc/8
T6OTL	10	Бит выходной защелки таймера T6. 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	Не детектируется
		1	Детектируется по сигналу на линии T6OUT
T6UDE	8	Бит выбора способа контроля направления счета таймера	
		0	Программно
		1	Посредством линии T6EUD
T6UD	7	Бит установки направления счета таймера T6 (если T6UDE = 0b)	
		0	Вверх
		1	Вниз
T6R	6	Бит работы таймера T6	
		0	Остановлен
		1	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
T6I	2–0	Выбор режима тактирования таймера T6. Для режима таймера комбинации в таблице А.4.8, для режима внешнего управления таймером – в таблице А.4.8, для режима счетчика – в таблице А.4.9.	
–	13	Зарезервировано	

Таблица А.4.8 – Режимы таймера и внешнего управления таймером Tx (x = 5, 6)

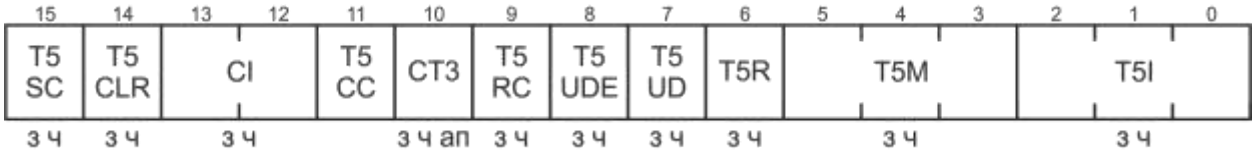
Tx	Делитель для fosc			
	BPS2= 00b	BPS2 = 01b	BPS2 = 10b	BPS2 = 11b
000b	4	2	16	8
001b	8	4	32	16
010b	16	8	64	32
011b	32	16	128	64
100b	64	32	256	128
101b	128	64	512	256
110b	256	128	1024	512
111b	512	256	2048	1024

Таблица А.4.9 – 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.4.10 – Регистр управления таймером T5

T5CON



Поле	Бит	Описание	
T5SC	15	Бит разрешения захвата содержимого таймера T5 в регистр CAPREL	
		0	Запрещено
		1	Разрешено
T5CLR	14	Бит разрешения сброса таймера T5 после захвата его содержимого	
		0	Не сбрасывается
		1	Сбрасывается
CI	13–12	Выбор события на линии таймера T3 для инициации захвата содержимого таймера T5 (в зависимости от бита CT3)	
		00	Захват запрещен
		01	Передний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3IN
		10	Задний фронт сигнала на входе CAPIN или любой фронт сигнала на входе T3EUD
		11	Любой фронт сигнала на входе CAPIN или любой фронт сигнала на входах T3IN или T3EUD
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, по сигналу которой выполняется захват содержимого таймера T3	
		0	Линия CAPIN
		1	Линия T3IN и/или T3EUD
T5RC	9	Бит удаленного контроля таймером Tx	
		0	Управляется собственным битом работы T5R
		1	Включается битом основного таймера T3
T5UDE	8	Бит выбора способа контроля направления счета таймера	
		0	Программно
		1	Посредством линии T5EUD

Окончание таблицы А.4.10

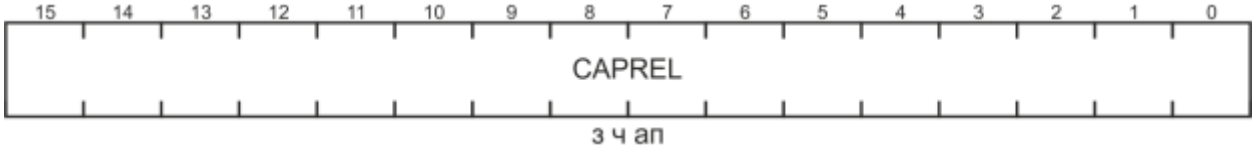
Поле	Биты	Описание	
T5UD	7	Бит установки направления счета таймера T5 (когда T5UDE = 0b):	
		0	Вверх
		1	Вниз
T5R	6	Бит работы таймера T5	
		0	Остановлен
		1	Работает
T5M	5–3	Выбор режима таймера T5 (основной режим работы):	
		000	Режим таймера
		001	Режим счетчика
		010	Режим внешнего управления активным низким уровнем
		011	Режим внешнего управления активным высоким уровнем
		1XX	Зарезервировано. Не использовать
T5I	2–0	Выбор режима тактирования таймера T5. Для режима счетчика комбинации в таблице А.4.11.	

Таблица А.4.11 – 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.4.12 – Регистр захвата/перезагрузки

CAPREL



Поле	Бит	Описание
CAPREL	15–0	Значение регистра захвата/перезагрузки

A.5 Регистры прерываний

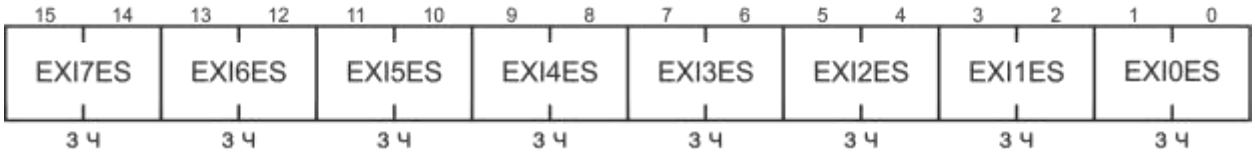
Таблица А.5.1 – Регистр управления прерываниями

xxIC / EOPIC

Поле	Бит	Описание
GP	8	Бит расширения группового приоритета. Определяет значение старшего разряда уровня группы. Примечание – Не следует устанавливать бит GP, если задействовано меньше 64 узлов прерываний и меньше восьми каналов PEC
EOP/ IR	7	Флаг запроса прерываний
		0 Ожидание запроса
		1 Источник выставил запрос на прерывание
		Этот бит поддерживает бит-защиту
EOP/ IE	6	Бит разрешения прерываний. Индивидуальное разрешение/запрет прерывания определенного источника
		0 Запрещено
		1 Разрешено
ILVL	5–2	Уровень приоритета прерывания. Значение Fh соответствует наивысшему уровню, значение 0h – низшему
GLVL	1–0	Уровень группового приоритета. Определяет внутренний приоритет в случае одновременного выставления запроса на прерывание с одинаковым приоритетом Значение 11b соответствует наивысшему уровню, значение 00b – низшему
0	15–9	Зарезервировано

Таблица А.5.2 – Регистр управления внешними прерываниями

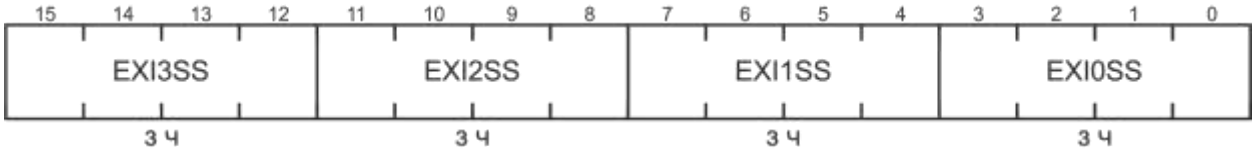
EXICON



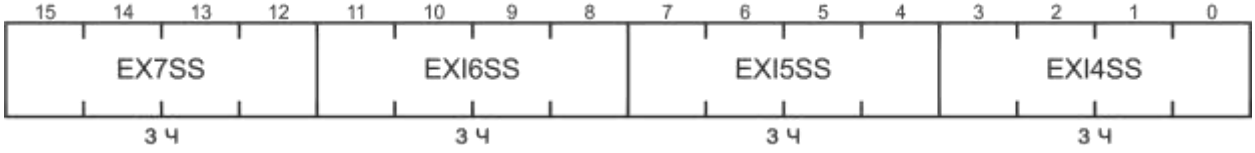
Поле	Бит	Описание
EXIxES	15–14, 13–12, 11–10, 9–8, 7–6, 5–4, 3–2, 1–0	Поле выбора события на входе микроконтроллера для активации быстрого внешнего прерывания x (от 0 до 7)
		00 Стандартный режим. Прерывания запрещены
		01 Положительный фронт сигнала
		10 Отрицательный фронт сигнала
		11 Любой фронт сигнала

Таблица А.5.3– Регистр выбора событий внешних прерываний

EXISEL0



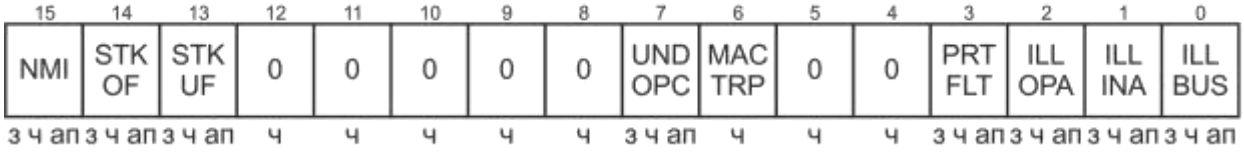
EXISEL1



Поле	Бит	Описание	
EXIxSS	15–12, 11–8, 7–4, 3–0	Источник события для быстрого внешнего прерывания x (от 0 до 7)	
		0000	EXxIN
		0001	EXxINA
		0010	EXxINB
		0011	EXxIN (ИЛИ) EXxINA
		0100	EXxIN (И) EXxINA
		0101	EXxINA (ИЛИ) EXxINB
		0110	EXxINA (И) EXxINB
		0111	EXxIN (ИЛИ) EXxINA (ИЛИ) EXxINB
		Остальные комбинации зарезервированы	

Таблица А.5.4 – Регистр флагов ловушек

TFR



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

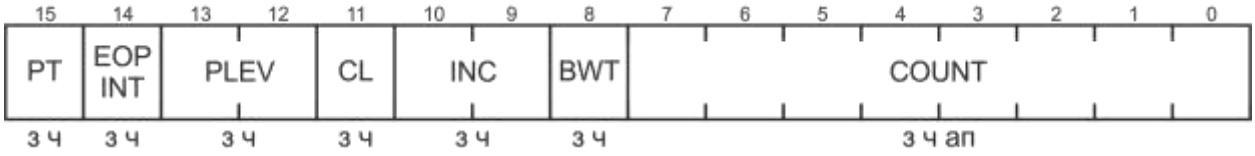
Окончание таблицы А.5.4

Поле	Бит	Описание	
UNDOPC	7	Флаг неопределенного программного кода	
		0	Нет ошибочного программного кода
		1	Обнаружен неверный программный код, текущая декодируемая команда не является командой контроллера
MACTRP	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	

А.6 Регистры контроллера PEC

Таблица А.6.1 – Регистр управления каналом x

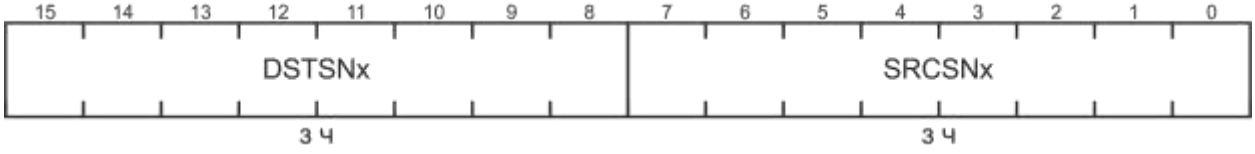
PECCx



Поле	Бит	Описание	
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)
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.6.2 – Регистр указателей сегмента адресов канала x

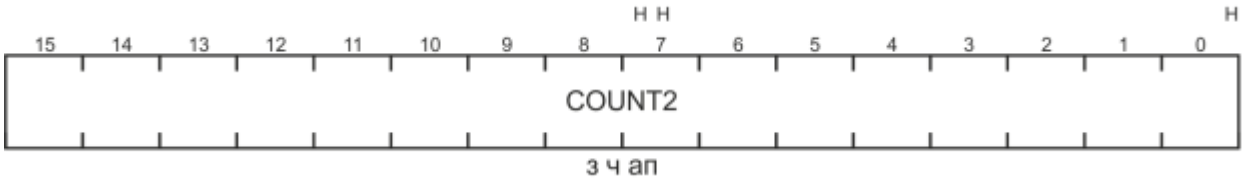
PECSNx



Поле	Бит	Описание
DSTSNx	15–8	Указатель сегмента адреса приемника канала x
SRCSNx	7–0	Указатель сегмента адреса источника канала x

Таблица А.6.3 – Регистры расширенного управления каналами 0 и 2

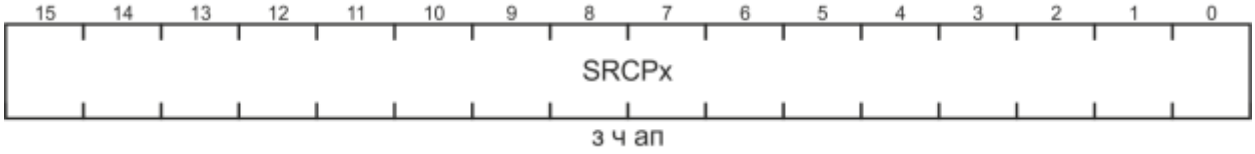
PECXCx



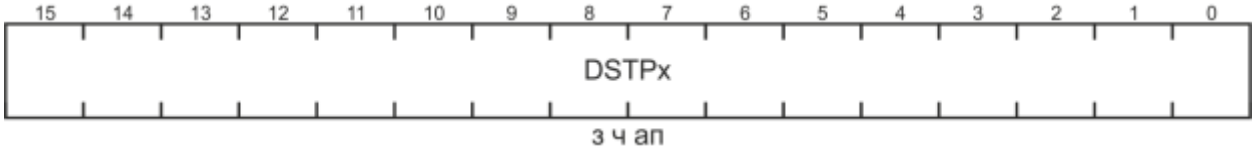
Поле	Бит	Описание
COUNT2	15–0	Счетчик расширенного управления каналом PEC

Таблица А.6.4 – Регистры адреса источника и приемника канала x

SRCPx



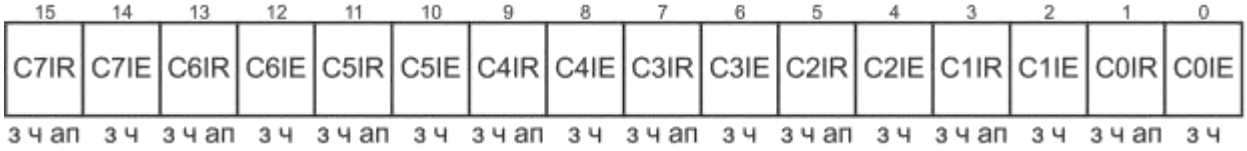
DSTPx



Поле	Бит	Описание
SRCPx	15–0	Адрес источника канала x
DSTPx	15–0	Адрес приемника канала x

Таблица А.6.5 – Регистр управления прерываниями каналов

PECISNC



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

Окончание таблицы А.6.5

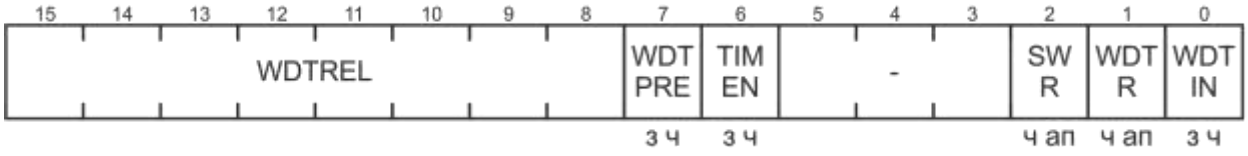
Примечание – Флаги запросов прерываний не очищаются аппаратно и должны быть очищены подпрограммой обработки прерываний.

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.7 Регистр сторожевого таймера

Таблица А.7.1 – Регистр управления сторожевым таймером

WDTCON



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.7.2)
TIMEN	6	Бит разрешения таймера. Если бит установлен, то WDT-сброс запрещается после выполнения команды DISWDT. При переполнении сторожевого таймера генерируется запрос на WDT-прерывание.
SWR	2	Флаг программного сброса (см. таблицу А.7.3)
WDTR	1	Флаг WDT-сброса (см. таблицу А.7.3)
WDTIN	0	Бит выбор делителя входной частоты. Функционирует в паре с битом WDTPRE (см. таблицу А.7.2)
–	5–3	Зарезервированы

Таблица А.7.2 – Комбинации битов выбора делителя входной частоты

WDTPRE	WDTIN	Делитель
0	0	2
1	0	4
0	1	128
1	1	256

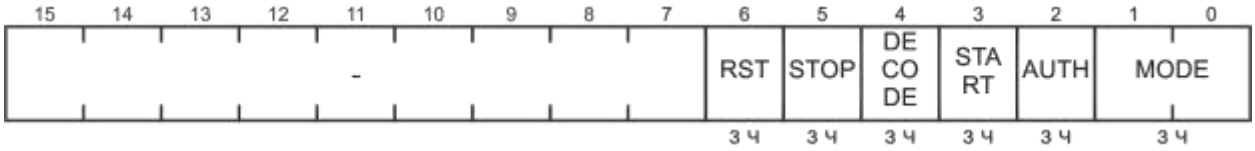
Таблица А.7.3 – Функционирование флагов программного сброса и WDT-сброса

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

А.8 Регистры блока кодирования по ГОСТ 28147–89

Таблица А.8.1 – Регистр управления

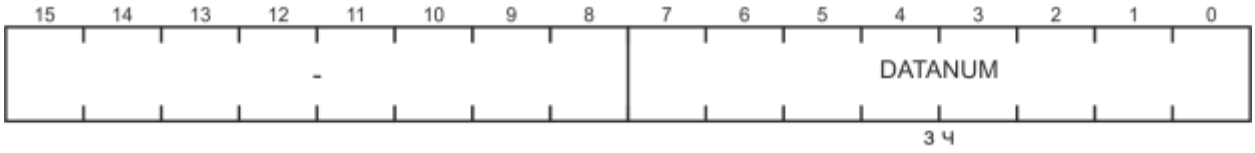
CDCON



Поле	Бит	Описание	
RST	6	Сброс блока кодирования. Установка этого бита приводит к сбросу всех регистров, за исключением регистра CDCON. Бит сбрасывается аппаратно	
STOP	5	Флаг последнего блока данных. Установка этого бита перед записью очередного блока данных в регистр CDDATA будет означать, что записываемый блок данных является последним и по окончании его обработки процесс кодирования/ декодирования следует завершить (независимо от значения, хранящегося в регистре CDDATNUM). Бит сбрасывается аппаратно после начала обработки загруженного блока данных	
DECODE	4	Бит выбора направления преобразования данных	
		0	Кодирование
		1	Декодирование
START	3	Бит запуска. Установка этого бита после записи первого блока данных запускает процесс кодирования/декодирования. Для обработки одного или более блоков данных бит следует установить только один раз. Бит сбрасывается аппаратно	
AUTH	2	Бит включения вычисления имитовставки (контрольной суммы)	
		0	Имитовставка не вычисляется
		1	Параллельно с процессом кодирования/декодирования идет вычисление имитовставки
		Следует помнить, что вычисление имитовставки замедляет процесс обработки блока данных в 1,5 раза	
MODE	1–0	Поле выбора режим работы	
		00	Режим бездействия. Запись значения 00h во время процесса кодирования/ декодирования приводит к остановке работы и сбросу логики блока кодирования. В отличие от результата установки бита RST, все регистры сохраняют свое состояние
		01	Простая замена
		10	Гаммирование
		11	Гаммирование с обратной связью
–	15-7	Зарезервировано	

Таблица А.8.2 – Регистр числа блоков данных

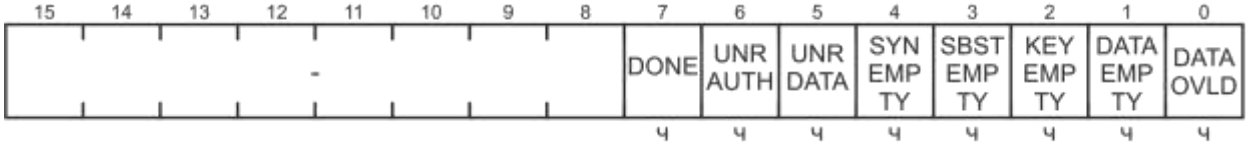
CDDATNUM



Поле	Бит	Описание
DATANUM	7–0	Число 64-разрядных блоков открытых данных, которые подлежат кодированию/декодированию
–	15-0	Зарезервировано

Таблица А.8.3 – Регистр состояния блока кодирования

CDSTATE



Поле	Бит	Описание
DONE	7	Флаг завершения преобразования
UNRAUTH	6	Предупреждение «Непрочитанная имитовставка». Флаг устанавливается в случае, если к моменту запуска преобразования (установка бита START) в регистре имитовставки CDAUTH находятся полностью или частично непрочитанные данные
UNRDATA	5	Предупреждение «Непрочитанные данные». Флаг устанавливается в случае, если к моменту запуска преобразования в регистре данных CDDATA находятся полностью или частично непрочитанные данные
SYNEMPTY	4	Ошибка синхровставки. Флаг устанавливается в случае, если к моменту запуска преобразования в режиме гаммирования (MODE = 10/11b) полностью или частично не заполнен регистр синхровставки CDUW
SBSTEMPTY	3	Ошибка регистра замены. Флаг устанавливается в случае, если к моменту запуска преобразования полностью или частично не заполнен регистр замены CDSBSTN
KEYEMPTY	2	Ошибка ключа. Флаг устанавливается в случае, если к моменту запуска преобразования полностью или частично не заполнен регистр ключа CDKEY
DATAEMPTY	1	Ошибка данных. Флаг устанавливается в случае, если к моменту запуска преобразования полностью или частично не заполнен регистр данных CDDATA

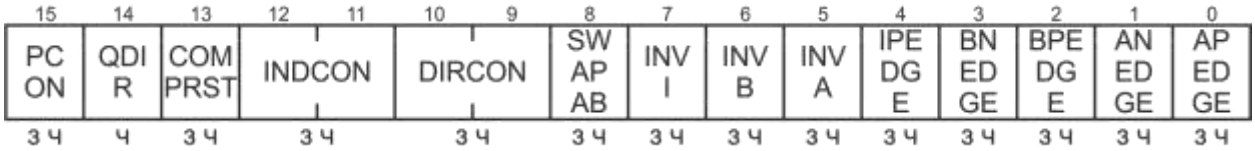
Окончание таблицы А.8.3

Поле	Бит	Описание
DATAOVLD	0	Предупреждение «Перегрузка данных». Флаг устанавливается: - если в буфер данных CDDATA записывается 9-й байт данных до начала преобразования (до установки бита START); - если в буфер данных записывается байт данных во время преобразования блока данных
–	15-8	Зарезервировано

A.9 Регистры квадратурного декодера

Таблица А.9.1 – Регистр управления 1

QD1CON



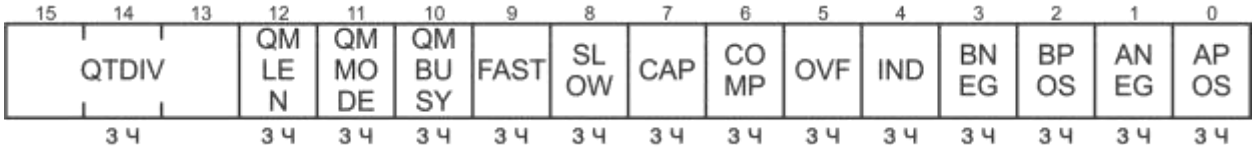
Поле	Бит	Описание	
PCON	15	Бит запуска счетчика позиции	
		0	Выключен
		1	Включен
QDIR	14	Флаг направления квадратурного счета	
		0	Вверх
		1	Вниз
COMPRST	13	Бит управления сбросом счетчика по компаратору	
		0	Нет действий
		1	При совпадении со значением регистра QPOSCOMP в счетчик будет загружено значение из регистра QPOSMAX
INDCON	12-11	Выбор действия по событию на индексном входе	
		00	Пропуск события
		01	Сброс счетчика
		10	Захват значения счетчика позиции
		11	Захват значения счетчика и последующий сброс
DIRCON	10-9	Режим счета	
		00	Квадратурный счет (в зависимости от бита QDIR)
		01	Режим счета/направления
		10	Счет вверх
		11	Счет вниз
SWAPAB	8	Бит включения обмена сигналов A_IN и B_IN	
		0	Выключен
		1	Включен
INVI	7	Бит включения инверсии индексного сигнала	
		0	Выключена
		1	Включена
INVB	6	Бит включения инверсии сигнала A_IN	
		0	Выключена
		1	Включена
INVA	5	Бит включения инверсии сигнала B_IN	
		0	Выключена
		1	Включена
IPEDGE	4	Бит выбора фронта индексного сигнала	
		0	Положительный
		1	Отрицательный
BNEDGE	3	Бит выключения счета по отрицательному фронту сигнала B_IN	
		0	Включен
		1	Выключен

Окончание таблицы А.9.1

Поле	Бит	Описание
BPEDGE	2	Бит выключения счета по положительному фронту сигнала B_IN
		0 Включен
		1 Выключен
ANEDGE	1	Бит выключения счета по отрицательному фронту сигнала A_IN
		0 Включен
		1 Выключен
APEDGE	0	Бит выключения счета по положительному фронту сигнала A_IN
		0 Включен
		1 Выключен

Таблица А.9.2 – Регистр управления 2

QD2CON



Поле	Бит	Описание
QTDIV	15-13	Делитель частоты синхросигнала счетчика
		000 Выключен
		001 1/2
		010 1/4
		011 1/8
		100 1/16
		101 1/32
		110 1/64
		111 1/128
QMLEN	12	Вид измерения
		0 Измерение периода сигнала
		1 Измерение длины импульса сигнала
QMMODE	11	Режим измерения
		0 Одно измерение
		1 Непрерывный счет
QMBUSY	10	Бит включения измерения
		0 Выключено
		1 Включено
FAST	9	Флаг ошибки измерения
		0 Флаг сброшен или ошибка не обнаружена
		1 Высокая частота переключения счетчика
SLOW	8	Флаг ошибки измерения
		0 Флаг сброшен или ошибка не обнаружена
		1 Низкая частота переключения счетчика
CAP	7	Флаг захвата содержимого счетчика позиции
		0 Флаг сброшен или захват еще не произошел
		1 Выполнен захват

Окончание таблицы А.9.2

Поле	Бит	Описание	
COMP	6	Флаг компаратора	
		0	Флаг сброшен или совпадение еще не произошло
		1	Совпадение произошло
OVF	5	Флаг переполнения	
		0	Флаг сброшен или переполнение еще не произошло
		1	Счетчик достиг максимального значения
IND	4	Флаг квадратурного события – активный фронт индекса	
		0	Событие не обнаружено
		1	Событие обнаружено
BNEG	3	Флаг квадратурного события – отрицательный фронт сигнала B_IN	
		0	Событие не обнаружено
		1	Событие обнаружено
BPOS	2	Флаг квадратурного события – положительный фронт сигнала B_IN	
		0	Событие не обнаружено
		1	Событие обнаружено
ANEG	1	Флаг квадратурного события – отрицательный фронт сигнала A_IN	
		0	Событие не обнаружено
		1	Событие обнаружено
APOS	0	Флаг квадратурного события – положительный фронт сигнала A_IN	
		0	Событие не обнаружено
		1	Событие обнаружено

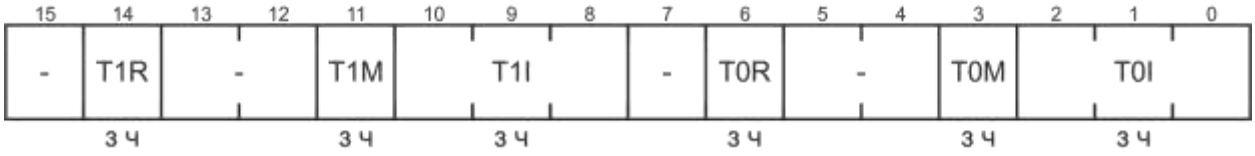
Таблица А.9.3– 16-разрядные регистры

Мнемоника	Назначение регистра	Сброс
QPOS	Регистр, содержащий текущее значение счетчика позиции	0000h
QPOSMAX	Регистр, содержащий значение верхней границы счетчика позиции, которое загружается в регистр QPOS по достижении им нуля при счете вниз. При счете вверх по достижении значения QPOSMAX счетчик QPOS сбрасывается в ноль.	FFFFh
QPOSCAP	Регистр захваченного значения QPOS	0000h
QPOSCOMP	Регистр компаратора, содержащий значение, которое сравнивается с текущим значением QPOS	0000h
QMRES	Регистр результата измерения периода или длительности импульса квадратурного сигнала A_IN. Доступен только для чтения	0000h
Примечание – все регистры, за исключением QMRES доступны для записи и чтения		

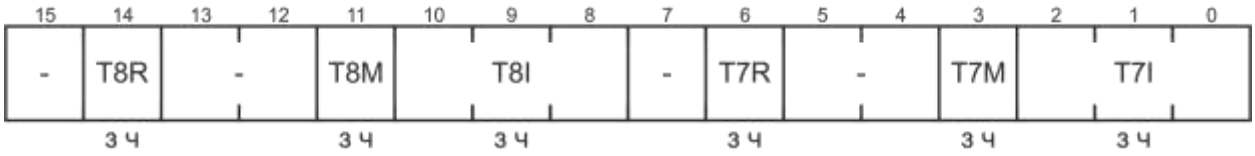
А.10 Регистры блоков CAPCOM1 и CAPCOM2

Таблица А.10.1 – Регистр управления таймерами/счетчиками T0 и T1

T01CON



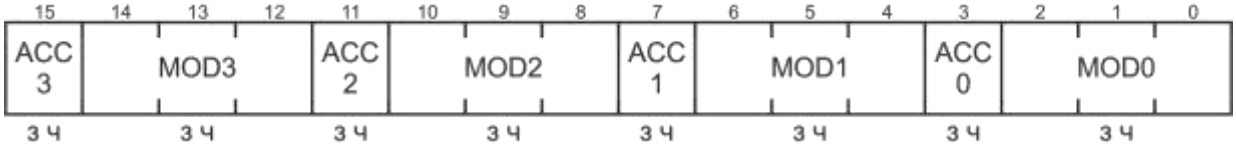
T78CON



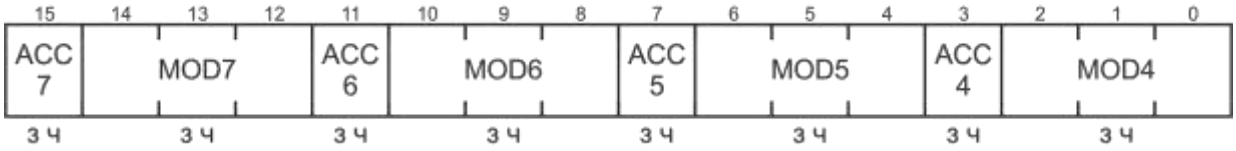
Поле	Бит	Описание	
T1R/T8R T0R/T7R	14 6	Бит запуска таймера	
		0	Выключен
		1	Включен
T1M/T8M T0M/T7M	11 3	Бит управления режимом таймера	
		0	Таймер
		1	Счетчик
T1I/T8I T0I/T7I	10–8 2–0	В режиме таймера поле задает коэффициента деления входной частоты fcc для получения частоты ftx (x – 0, 1, 7, 8) сигнала тактирования. Для ступенчатого режима $$ftx = \frac{fcc}{2 \times (TxI + 3)}, \quad (A.10.1)$$ для нормального режима $$ftx = \frac{fcc}{2TxI}. \quad (A.10.2)$$ Примечание – В ступенчатом режиме частота сигнала тактирования таймера дополнительно делится на 8.	
		В режиме счетчика: 1 В поле T1I/T8I может быть записано только значение 000b. Источником сигнала тактирования T6OUF является таймер T6 блока GPT2. Счетчик T1/T8 инкрементируется каждый раз, когда переполняется таймер T6. 2 Поле T0I/T7I выбирает источник, сигнала тактирования счетчика T0/T7.	
		000	Переполнение таймера T6 блока GPT2
		001	Передний фронт сигнала на входе T0IN/T8IN
		010	Задний фронт сигнала на входе T0IN/T8IN
		011	Передний/задний фронт сигнала на входе T0IN/T8IN
		Остальные значения зарезервированы	
–	15, 13, 12, 7, 5, 4	Зарезервировано	

Таблица А.10.2 – Регистры управления режимами

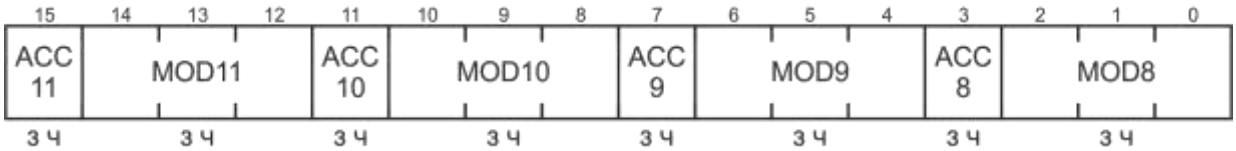
CC1_M0



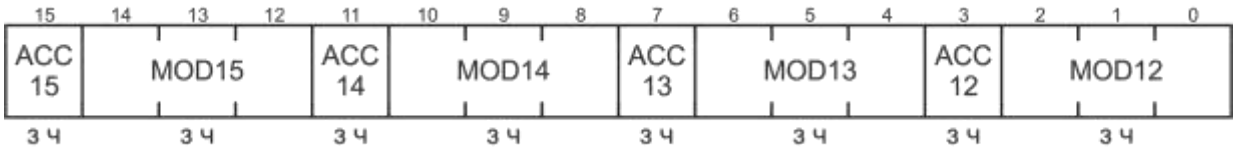
CC1_M1



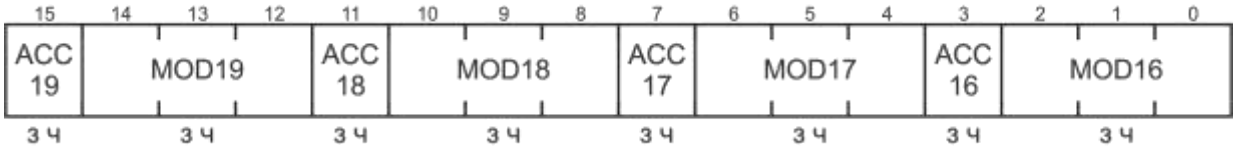
CC1_M2



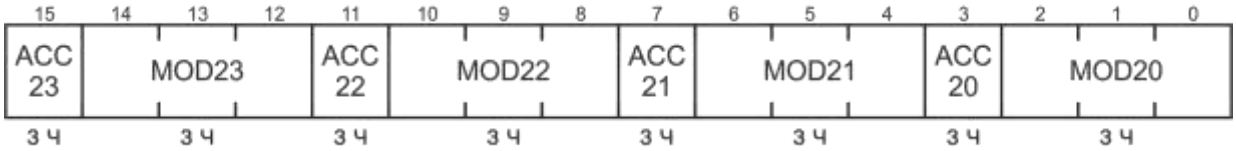
CC1_M3



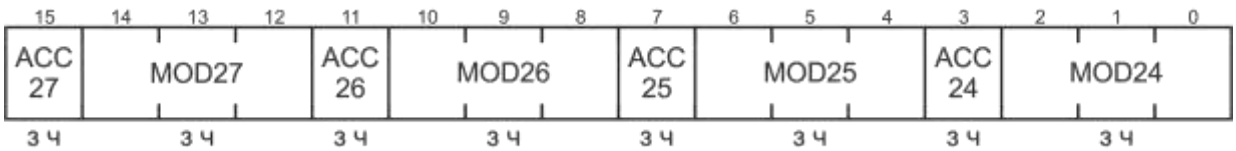
CC2_M4



CC2_M5

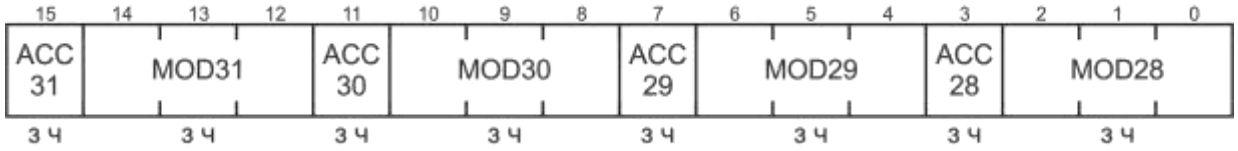


CC2_M6



Окончание таблицы А.10.2

CC2_M7



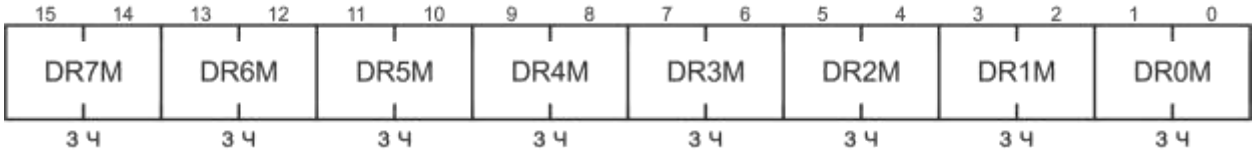
Поле	Бит	Описание	
ACCy	15, 11, 7, 3	Бит выбора таймера/счетчика для регистра CCy	
		0	T0
		1	T1
MODy	14 – 12, 10 – 8, 6 – 4, 2 – 0	Поле задания режима работы канала y (см. таблицу А.10.3)	

Таблица А.10.3 – Режимы работы канала y, задаваемые полем MODy

Код режима	Описание режима
000b	Регистр CCy используется как регистр общего назначения микроконтроллера
001b	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении переднего фронта сигнала на входе CCyIO и генерирование прерывания
010b	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении заднего фронта сигнала на входе CCyIO и генерирование прерывания
011b	Захват значения выбранного таймера/счетчика в регистр CCy при обнаружении переднего/заднего фронта сигнала на входе CCyIO и генерирование прерывания
100b	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
101b	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
110b	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
111b	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.10.4 – Регистр двухрегистрового режима сравнения

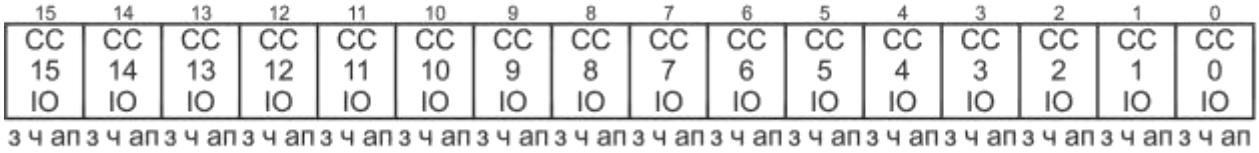
CC1_DRM / CC2_DRM



Поле	Бит	Описание	
DRxM (x от 0 до 7)	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	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.10.5 – Регистр выходных сигналов

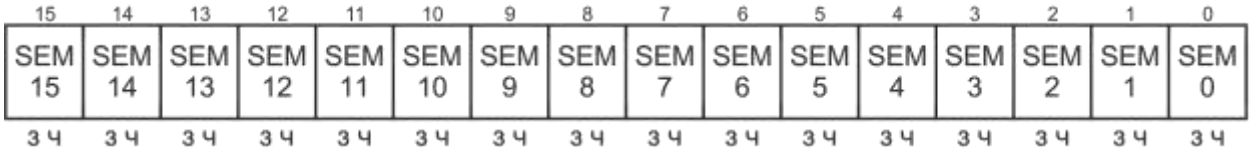
CC1_OUT / CC2_OUT



Поле	Бит	Описание
CCxIO (x от 0 до 15)	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Выходная защелка канала

Таблица А.10.6 – Регистр режима однократного срабатывания

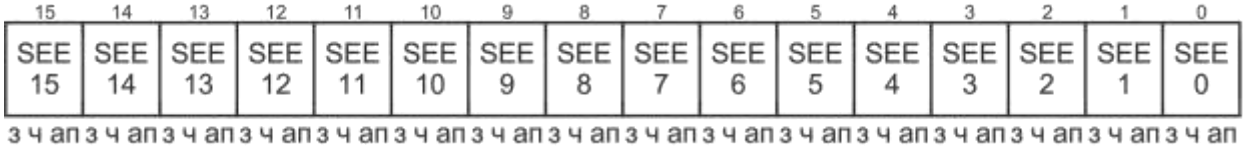
CC1_SEM / CC2_SEM



Поле	Бит	Описание	
SEMx (x от 0 до 15)	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Бит включения режима однократного срабатывания	
		0	Выключен
		1	Включен

Таблица А.10.7 – Регистр однократного срабатывания

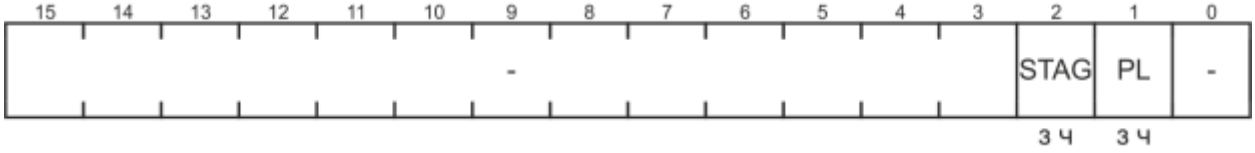
CC1_SEE / CC2_SEE



Поле	Бит	Описание	
SEEx (x от 0 до 15)	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0	Флаг ожидания события	
		0	Ожидание запрограммированного события не запущено или событие уже произошло
		1	Ожидается событие. Бит может быть установлен только, если установлен соответствующий ему бит в регистре CC1_SEM/ CC2_SEM

Таблица А.10.8– Регистры управления вводом-выводом

CC1_IOC / CC2_IOC

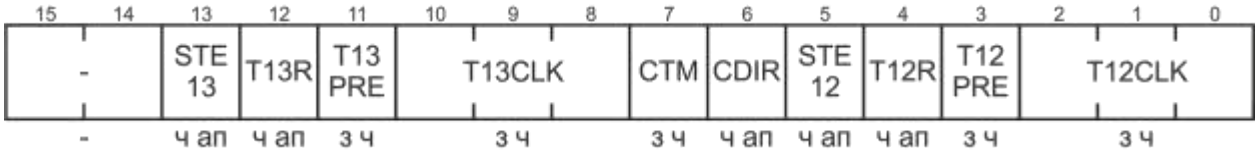


Поле	Бит	Описание	
STAG	2	Бит выключения ступенчатого режима	
		0	Ступенчатый режим
		1	Нормальный режим
PL	1	Бит управления подключением выходов модуля к выводам микроконтроллера	
		0	Все выходы подключены к соответствующим выводам
		1	Выходы отключены и не оказывают влияния на выводы
–	15–3, 0	Зарезервировано	

A.11 Регистры блока CAPCOM6

Таблица А.11.1 – Регистр управления таймерами

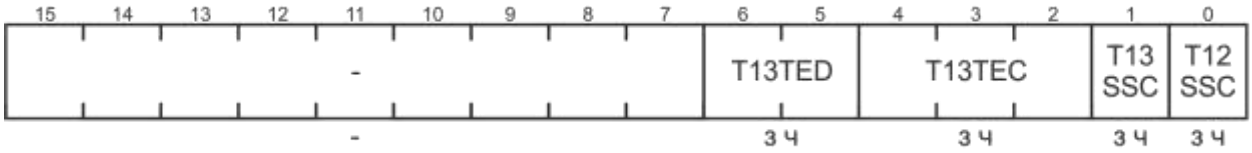
CCU6_TCTR0



Поле	Бит	Описание		
STE13 STE12	13 5	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	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	Остановлен	
		1	Запущен	
T13PRE T12PRE	11 3	Бит включения дополнительного делителя входной частоты		
		0	Выключен	
		1	Включен делитель 1/256	
T13CLK T12CLK	10–8, 2–0	Поле задания коэффициента деления частоты fosc внешнего сигнала		
		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
CTM	7	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	Индикатор текущего направления счета таймера T12		
		0	Вверх	
		1	Вниз	
–	15, 14	Зарезервировано		

Таблица А.11.2 – Регистр управления таймерами

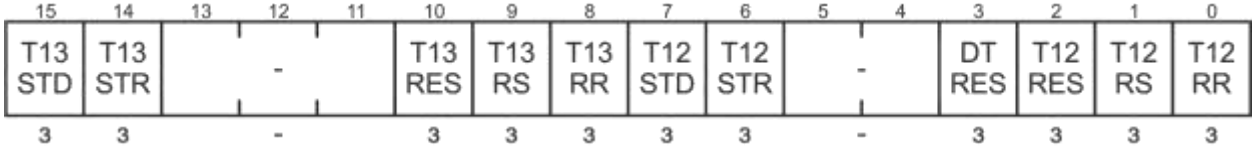
CCU6_TCTR2



Поле	Бит	Описание	
T13TED	6, 5	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	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	Бит включения режима однотактного запуска таймера T13/T12	
		0	Выключен
		1	Включен
–	15–7	Зарезервировано	

Таблица А.11.3 – Регистр управления таймерами

CCU6_TCTR4



Поле	Бит	Описание	
T13STD T12STD	15 7	Бит управления аппаратной теневой загрузкой регистра T13PR/T12PR	
		0	Теневая загрузка таймера T13 происходит аппаратно каждый раз, когда для таймера возникает событие «Период»
			Теневая загрузка таймера T12 происходит аппаратно каждый раз, когда для таймера возникает событие «Период» при счете вверх и «Единица» при счете вниз

Окончание таблицы А.11.3

Поле	Бит	Описание	
T13STD T12STD	15 7	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	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	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	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
T13RR T12RR	8 0	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	Бит сброса и остановки счетчиков задержки. Сбрасывается аппаратно	
		0	Нет действий
		1	Счетчики задержки всех каналов обнуляются и останавливаются
–	13–11, 5, 4	Зарезервировано	

Таблица А.11.4 – Регистры блока таймера T12

Название регистра	Адрес	Назначение
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

Окончание таблицы А.11.4

Название регистра	Адрес	Назначение
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
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.11.5 – Регистр состояния входов и выходов каналов

CCU6_CMPSTAT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13IM	CCOUT63PS	CCOUT62PS	CC62PS	CCOUT61PS	CC61PS	CCOUT60PS	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

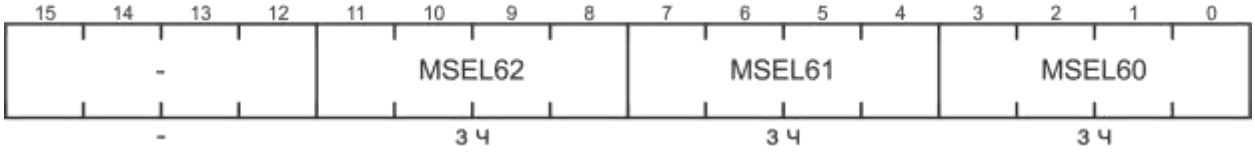
Поле	Бит	Описание	
T13IM	15	Бит выбора сигнала для дальнейшей модуляции в канале 3	
		0	CC63ST
		1	CC63ST#

Окончание таблицы А.11.5

Поле	Бит	Описание	
COUT63PS COUT62PS COUT61PS COUT60PS	14 13 11 9	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	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	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	Бит состояния входа, к которому подключается датчик Холла. Доступен только в режиме работы с датчиками Холла	
–	7	Зарезервировано	

Таблица А.11.6 – Регистр выбора режима захвата/сравнения T12

CCU6_T12MSEL



Поле	Бит	Описание	
MSEL62 MSEL61 MSEL60	11–8 7–4 3–0	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

Окончание таблицы А.11.6

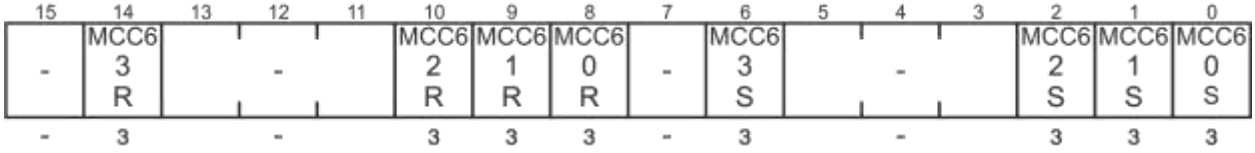
Поле	Бит	Описание	
MSEL62 MSEL61 MSEL60	11–8 7–4 3–0	Двухрегистровые и мультивходовые режимы захвата	
		0100	Режим 1
		0101	Режим 2
		0110	Режим 3
		0111	Режим 4
		1010	Режим 5
		1011	Режим 6
		1100	Режим 7
		1101	Режим 8
		1110	Режим 9
		Специальные режимы	
		1000	Режим работы с датчиками Холла
		1001	Режим блокирования
		1111	Зарезервировано
–	15–12	Зарезервировано	

Таблица А.11.7 – 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 регистра CC6xSR переносится в	Содержимое T12 переносится в
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 (действия захвата/сравнения не производятся)			
Примечание – Режим датчиков Холла должен программироваться одновременно для трех каналов таймера T12.					

Таблица А.11.8 – Регистр управления битами состояния

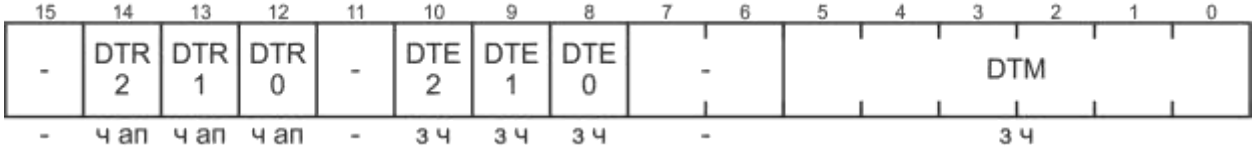
CCU6_CMPMODIF



Поле	Бит	Описание	
MCC63R MCC62R MCC61R MCC60R	14 10 9 8	Бит программного сброса бита состояния CC6xST	
		0	Нет действий
		1	Сбросить
MCC63S MCC62S MCC61S MCC60S	6 2 1 0	Бит программной установки бита состояния CC6xST	
		0	Нет действий
		1	Установить
–	15, 13–11, 7, 5–3	Зарезервировано	

Таблица А.11.9 – Регистр управления задержкой

CCU6_T12DTC



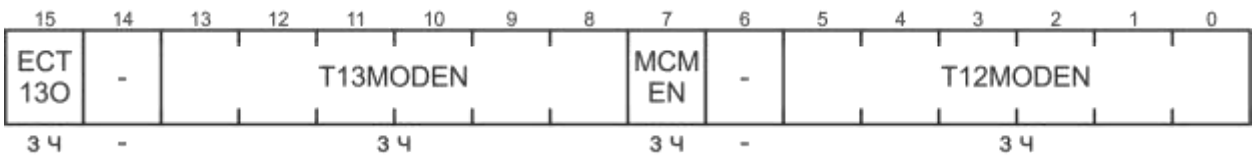
Поле	Бит	Описание	
DTR2 DTR1 DTR0	14 13 12	Флаг задержки	
		0	Счетчик задержки остановлен
		1	Счетчик задержки работает
DTE2 DTE1 DTE0	10 9 8	Бит разрешения генерирования задержки	
		0	Запрещено
		1	Разрешено
DTM	5–0	Битовое поле программируемого значения задержки. Определяет программируемую задержку «dead-time» между переключением канала из пассивного состояния в активное (определяет значение, с которого начинает счет счетчик задержки)	
–	15, 11, 7, 6	Зарезервировано	

Таблица А.11.10 – 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.11.11 – Регистр управления модуляцией

CCU6_MODCTR

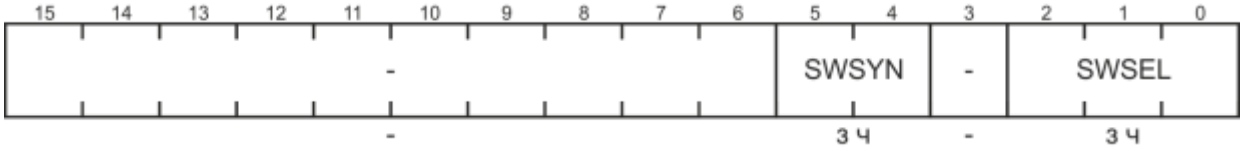


Окончание таблицы А.11.11

Поле	Биты	Описание	
ECT13O	15	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	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	Включение мультиканального режима. Бит разрешения управления модуляцией выходных сигналов посредством поля MCMOUT	
		0	Выключен
		1	Включен
–	6, 14	Зарезервировано	

Таблица А.11.12 – Регистр управления мультиканальным режимом

CCU6_MCMCTR



Поле	Биты	Описание
SWSYN	5–4	Поле выбора сигнала синхронизации. Появление выбранного полем SWSYN сигнала активирует сигнал теневой передачи MCM_ST в случае, если бит установлен флаг R регистра MCMOUT
SWSEL	2–0	Поле выбора события для теневой передачи. Поле SWSEL определяет сигнал, по которому будет установлен флаг R
–	15–6, 3	Зарезервировано

Таблица А.11.13 – Выбор события запуска теневой передачи в мультиканальном режиме

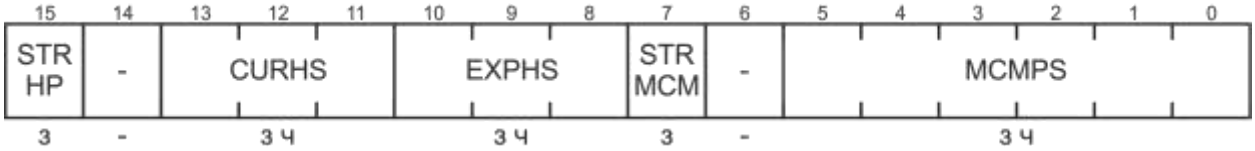
SWSEL	Выбранное событие
000b	Событие не выбрано
001b	Корректный сигнал датчика Холла CM_CHE
010b	Совпадение по периоду таймера T13 (T13_PM)
011b	Совпадение счетчика таймера T12 с единицей (T12_OM) при счете «вниз»
100b	Совпадение по значению для канала 1 при счете таймера T12 «вверх» (CM_61_UP)
101b	Совпадение по периоду таймера T12 (T12_PM) при счете «вверх»
11xb	Зарезервировано

Таблица А.11.14 – 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.11.15 – Теневой регистр управления выходами в мультиканальном режиме

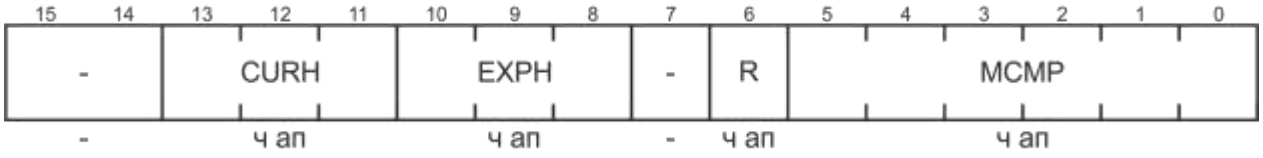
CCU6_MCMOUTS



Поле	Биты	Описание
STRHP	15	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	Теневое поле текущей комбинации сигналов датчиков. Содержимое этого поля записывается в поле CURH по сигналу теневой передачи
EXPHS	10–8	Теневое поле ожидаемой комбинации сигналов датчиков. Содержимое этого поля записывается в поле EXPH по сигналу теневой передачи
STRMCM	7	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	Теневое поле комбинации значений, управляющих выходной модуляцией битов. Содержимое этого поля записывается в поле MCMP по сигналу теневой передачи
–	14, 6	Зарезервировано

Таблица А.11.16 – Регистр управления выходами в мультиканальном режиме

CCU6_MCMOUT

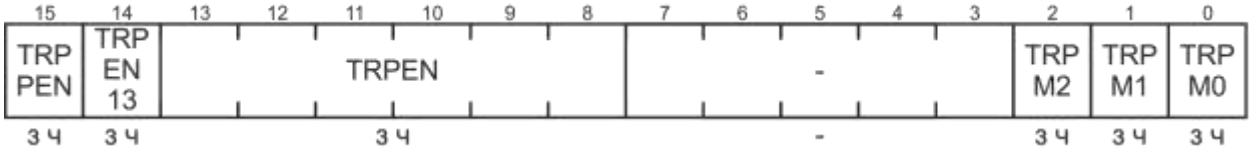


Поле	Биты	Описание
CURH	13–11	Поле текущей комбинации сигналов датчиков. Это поле получает значение из теневого поля CURHS по сигналу теневой передачи и хранит значение текущей комбинации сигналов, приходящих с датчиков Холла
EXPH	10–8	Поле ожидаемой комбинации сигналов датчиков. Это поле получает значение из теневого поля EXPHS по сигналу теневой передачи и хранит значение ожидаемой комбинации сигналов, приходящих с датчиков Холла
R	6	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
MCMP	5–0	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. MCMP.5 – MCMP.0 соответствуют (побитно слева направо) COUT62, CC62, COUT61, CC61, COUT60, CC60
		0 – На соответствующей выходной линии поддерживается низкий уровень
		1 – Соответствующая выходная линия активна
		Поле MCMP очищается в то время, как установлен бит IDLE регистра IS
–	15–14, 7	Зарезервировано

Примечание – Биты в полях EXPH и CURH соответствуют сигналам с датчиков Холла на входах CCPOSx (x – от 0 до 2) EXPH.2, EXPH.1, EXPH.0 и CURH.2, CURH.1, CURH.0 соответствуют побитно слева направо CCPOS2, CCPOS1, CCPOS0

Таблица А.11.17 – Регистр управления ловушками

CCU6_TRPCTR

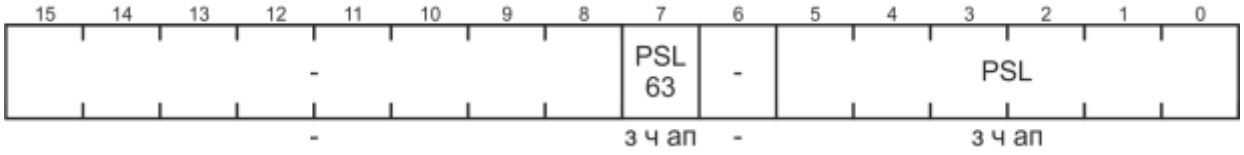


Окончание таблицы А.11.17

Поле	Биты	Описание	
TRPPEN	15	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	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 установлен бит TRPS
TRPEN	13–8	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. TRPEN.5 – TRPEN.0 соответствует побитно слева направо COUT62, CC62, COUT61, CC61, COUT60, CC60.	
		0	Запрет попадания в ловушку. Линия функционирует независимо от бита TRPS
		1	Разрешение попадания в ловушку. На линии поддерживается низкий уровень сигнала, пока установлен бит TRPS
TRPM2	2	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 единице
		1	Флаг TRPF должен быть сброшен программно после того, как CTRAP# перейдет в неактивное состояние
TRPM1, TRPM0	1, 0	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.11.18 – Регистр уровня пассивного состояния

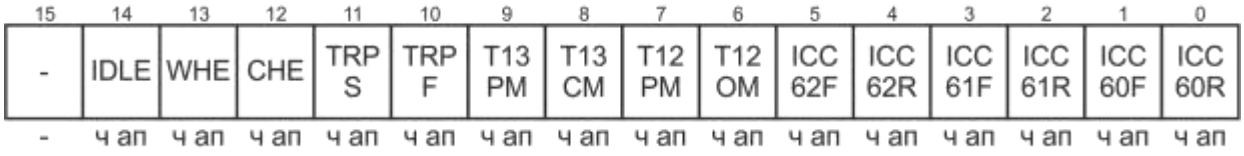
CCU6_PSLR



Поле	Биты	Описание
PSL63	7	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	Поле управления уровнем выходных сигналов каналов таймера T12. Поле побитно определяет, в каком виде будет передан на выход соответствующий модулируемый сигнал. PSLR.5–PSLR.0 соответствуют (побитно слева направо) COUT62, CC62, COUT61, CC61, COUT60, CC60
		0 Модулируемый сигнал передается в прямом виде
		1 Модулируемый сигнал передается в инвертированном виде
–	15–8, 6	Зарезервировано

Таблица А.11.19 – Регистр состояния прерываний

CCU6_IS



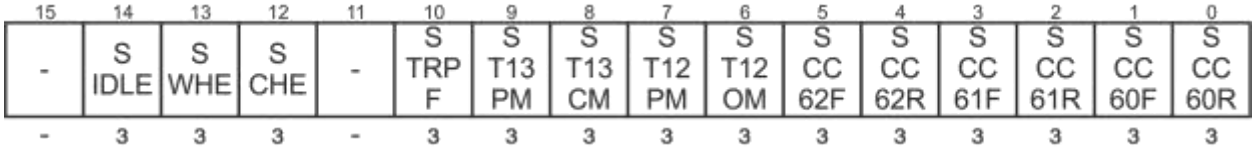
Поле	Биты	Описание
IDLE	14	Флаг режима ожидания Idle. Если разрешено (установлен бит ENIDLE), этот флаг устанавливается совместно с флагом WHE
		0 Бездействие.
		1 Поле MCMP очищается, на выходах каналов таймера T12 поддерживается низкий уровень сигнал
WHE	13	Флаг некорректного сигнала датчика Холла
		0 Бездействие
		1 Обнаружение некорректной комбинации сигналов датчиков
CHE	12	Флаг корректного сигнала датчика Холла
		0 Бездействие
		1 Обнаружение корректной комбинации сигналов датчиков

Окончание таблицы А.11.19

Поле	Биты	Описание	
TRPS	11	Флаг перехода в состояние ловушки	
		0	Попадания в ловушку нет (флаг TRPF не выставлен)
		1	Переход в состояние попадания в ловушку
TRPF	10	Флаг попадания в ловушку	
		0	Аппаратного или программного попадания в ловушку нет
		1	Аппаратное (CTRAP# = 0B) или программное попадание в ловушку
T13PM	9	Флаг совпадения по периоду счетчика таймера T13	
		0	Совпадения по периоду еще не было
		1	Совпадение по периоду произошло
T13CM	8	Флаг совпадения по значению счетчика таймера T13	
		0	Совпадения по значению еще не было
		1	Совпадение по значению произошло
T12PM	7	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	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	Флаг события 1: - появление ожидаемого заднего фронта сигнала в случае режима захвата; - совпадения по значению при счете «вниз» счетчика таймера T12 в случае режима сравнения. В режиме сравнения флаг будет выставлен, когда возникнет совпадение по значению в течение времени декрементирования счетчика таймера T12. В режиме захвата флаг будет выставлен, когда на входе CC6x, x = 0, 1, 2, появится ожидаемый задний фронт (перепад из «1» в «0») входного сигнала.	
		0	Событие не произошло
		1	Ожидаемое событие произошло
ICC62R ICC61R ICC60R	4 2 0	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.11.20– Регистр установки состояния прерываний

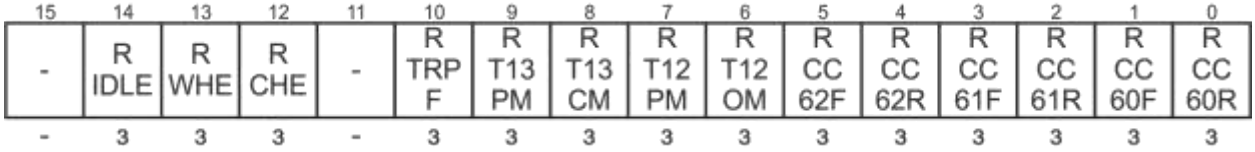
CCU6_ISS



Поле	Биты	Описание
SIDLE	14	Установка флага режима ожидания Idle
SWHE	13	Установка флага некорректного сигнала датчика Холла
SCHE	12	Установка флага корректного сигнала датчика Холла
STRPF	10	Установка флага перехода в состояние ловушки
ST13PM	9	Установка флага совпадения по периоду счетчика таймера T13
ST13CM	8	Установка флага совпадения по значению счетчика таймера T13
ST12PM	7	Установка флага совпадения по периоду при счете «вверх» счетчика таймера T12
ST12OM	6	Установка флага совпадения с единицей при счете «вниз» счетчика таймера T12
SCC62F SCC61F SCC60F	5 3 1	Установка флага события 1
SCC62R SCC61R SCC60R	4 2 0	Установка флага события 2
–	15, 11	Зарезервировано

Таблица А.11.21 – Регистр сброса состояния прерываний

CCU6_ISR



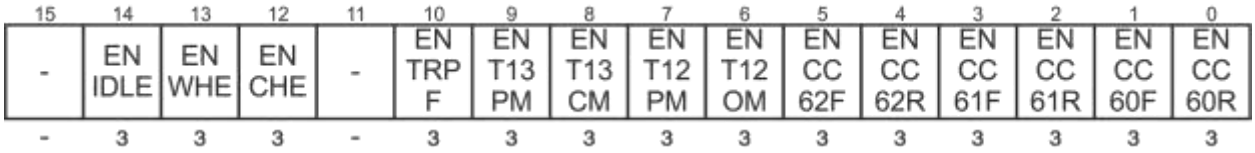
Поле	Биты	Описание
RIDLE	14	Сброс флага режима ожидания Idle
RWHE	13	Сброс флага некорректного сигнала датчика Холла
RCHE	12	Сброс флага корректного сигнала датчика Холла
RTRPF	10	Сброс флага перехода в состояние ловушки
RT13PM	9	Сброс флага совпадения по периоду счетчика таймера T13
RT13CM	8	Сброс флага совпадения по значению счетчика таймера T13
RT12PM	7	Сброс флага совпадения по периоду при счете «вверх» счетчика T12
RT12OM	6	Сброс флага совпадения с единицей при счете «вниз» счетчика T12
RCC62F RCC61F RCC60F	5 3 1	Сброс флага события 1

Окончание таблицы А.11.21

Поле	Биты	Описание
RCC62R RCC61R RCC60R	4 2 0	Сброс флага события 2
–	15, 11	Зарезервировано

Таблица А.11.22 – Регистр разрешения прерываний

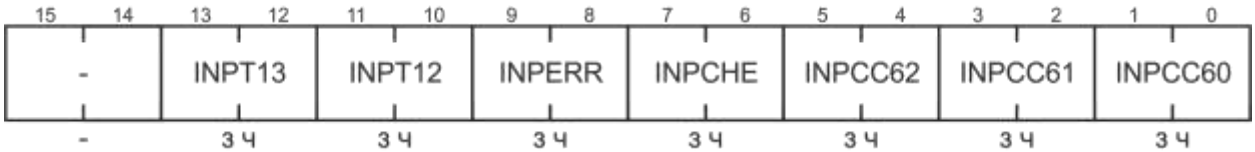
CCU6_IEN



Поле	Биты	Описание
ENIDLE	14	Разрешение установки флага режима ожидания Idle
ENWHE	13	Разрешение формирования запроса на прерывания при появлении сигнала некорректного сигнала датчика Холла
ENCHE	12	Разрешение формирования запроса на прерывания при появлении сигнала корректного сигнала датчика Холла
ENTRPF	10	Разрешение формирования запроса на прерывания при появлении сигнала перехода в состояние ловушки
ENT13PM	9	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду счетчика таймера T13
ENT13CM	8	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по значению счетчика таймера T13
ENT12PM	7	Разрешение формирования запроса на прерывания при появлении сигнала совпадения по периоду при счете «вверх» счетчика таймера T12
ENT12OM	6	Разрешение формирования запроса на прерывания при появлении сигнала совпадения с единицей при счете «вниз» счетчика таймера T12
ENCC62F ENCC61F ENCC60F	5 3 1	Разрешение формирования запроса на прерывания при появлении сигнала события 1
ENCC62R ENCC61R ENCC60R	4 2 0	Разрешение формирования запроса на прерывания при появлении сигнала события 2
–	11, 15	Зарезервировано

Таблица А.11.23 – Регистр указателя узла прерываний

CCU6_INP



Окончание таблицы А.11.23

Поле	Биты	Описание
INPT13	13, 12	Указатель узла прерываний таймера T13. Источники формирования запроса на прерывание: - T13CM; - T13PM
INPT12	11, 10	Указатель узла прерываний таймера T12. Источники формирования запроса на прерывание: - T12OM; - T12PM
INPERR	9, 8	Указатель узла прерываний ошибок. Источники формирования запроса на прерывание: - TRPF; - WHE
INPCHE	7, 6	Указатель узла прерываний теневых передач. Источники формирования запроса на прерывание: - CHE; - MCM_ST
INPCC62 INPCC61 INPCC60	5, 4 3, 2 1, 0	Указатель узла прерываний канала x (x = 0, 1, 2). Источники формирования запроса на прерывание: - CC6xR; - CC6xF
–	15, 14	Зарезервировано
Комбинация в поле INPxx задает выход: 00b – I0; 01b – I1; 10b – I2; 11b – I3.		

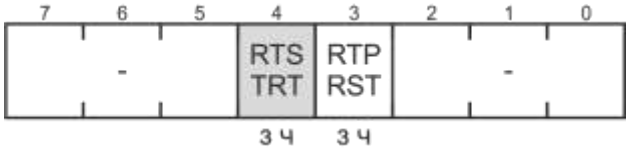
Таблица А.11.24 – Соответствия (по умолчанию) источников запросов на прерывания по узлам (выходам) и регистров

Источник прерывания	Выход	Управляющий регистр
Канал 0	I0	CCU6_IC
Канал 1	I0	
Канал 2	I0	
Корректный сигнал датчика	I1	CCU6_EIC
Возникновение ошибки	I1	
Таймер T12	I2	CCU6_T12IC
Таймер T13	I3	CCU6_T13IC

A.12 Регистры блока RTC

Таблица А.12.1 – Регистр управления и состояния

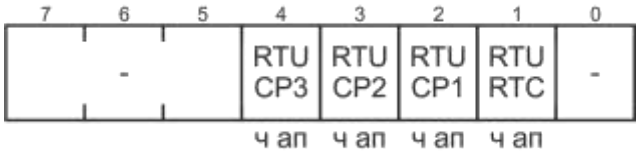
RTCCST



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.12.2 – Регистр ожиданий загрузок

RTUDST



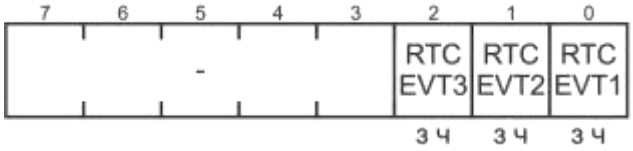
Поле	Биты	Описание	
RTUCP3	4	Бит ожидания загрузки в регистр RTCCMP3. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	Есть данные для загрузки
RTUCP2	3	Бит ожидания загрузки в регистр RTCCMP2. Сбрасывается аппаратно после загрузки данных	
		0	Нет данных для загрузки
		1	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

Окончание таблицы А.12.2

Поле	Биты	Описание
–	7–5, 0	Зарезервировано

Таблица А.12.3 – Регистр совпадений

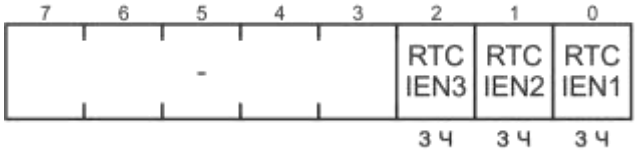
RTCEIST



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.12.4 – Регистр разрешения прерываний

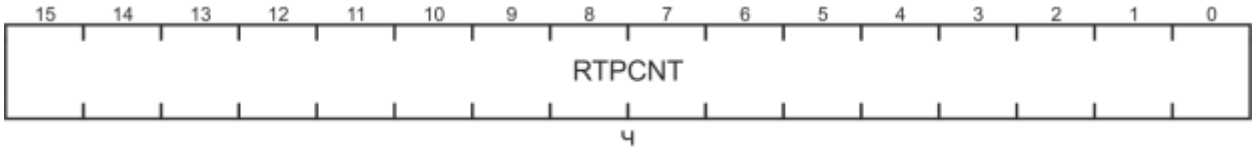
RTCIEN



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

Таблица А.12.5 – Регистр счетчика импульсов

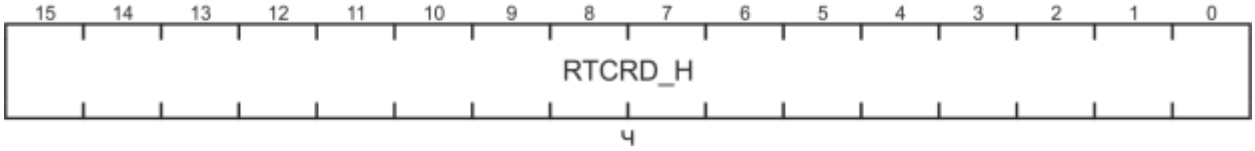
RTPRD



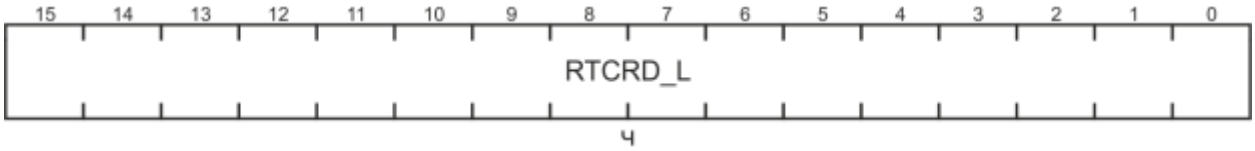
Поле	Биты	Описание
RTPCNT	15–0	Значение 16-разрядного счетчика импульсов

Таблица А.12.6 – Регистры старшего и младшего слов счетчика реального времени

RTCRD_H **(только Power-On-Reset)**



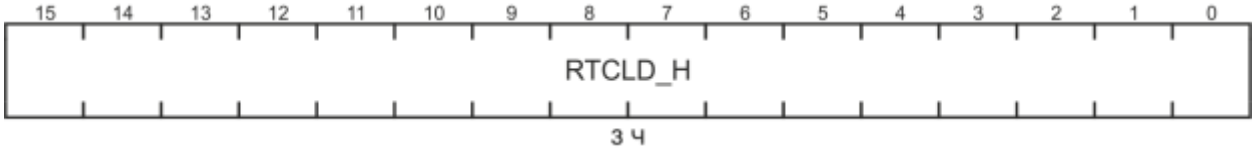
RTCRD_L **(только Power-On-Reset)**



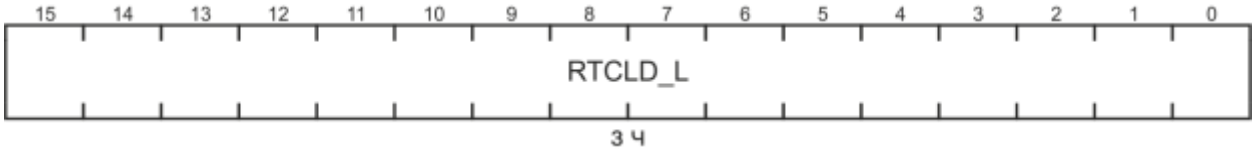
Поле	Биты	Описание
RTCRD_H	15–0	Значение старшего слова 32-разрядного счетчика реального времени
RTCRD_L	15–0	Значение младшего слова 32-разрядного счетчика реального времени

Таблица А.12.7 – Регистры загрузки старшего и младшего слов счетчика реального времени

RTCLD_H



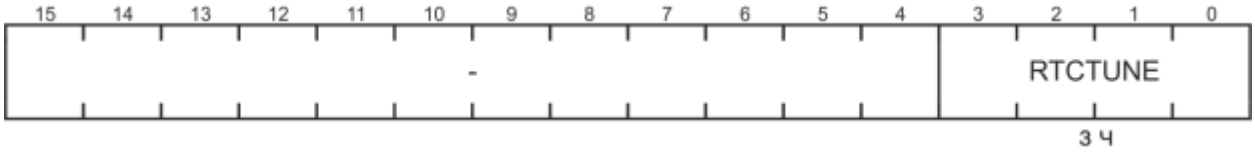
RTCLD_L



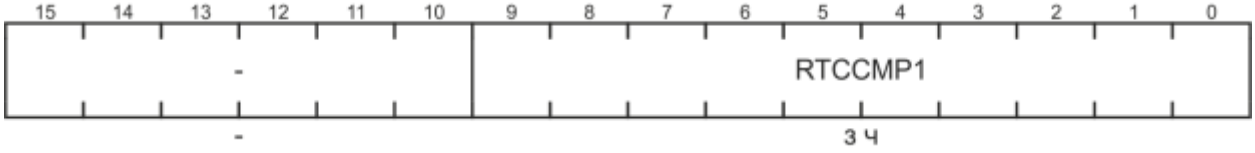
Поле	Биты	Описание
RTCLD_H	15–0	Значение для загрузки в старшее слово счетчика реального времени
RTCLD_L	15–0	Значение для загрузки в младшее слово счетчика реального времени

Таблица А.12.8 – Первый регистр сравнения (старшее и младшее слова)

RTCCMP1_H **(только Power-On-Reset)**



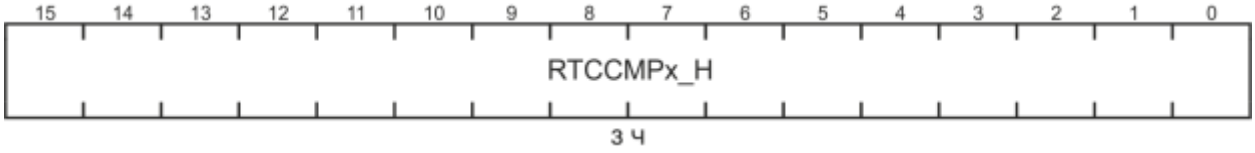
RTCCMP1_L **(только Power-On-Reset)**



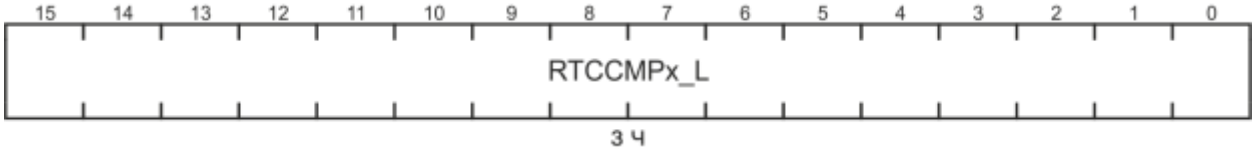
Поле	Биты	Описание
RTCTUNE	3–0	Поле задания количества шагов подстройки
RTCCMP1	9–0	Поле задания значения компенсации при настройке модуля RTC
–	15–10/ 15–4	Зарезервировано

Таблица А.12.9 – Второй и третий регистры сравнения (x – 2 и 3)

RTCCMPx_H **(только Power-On-Reset)**



RTCCMPx_L **(только Power-On-Reset)**

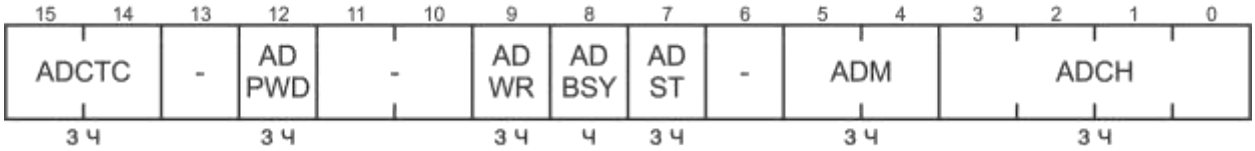


Поле	Биты	Описание
RTCCMPx_H	15–0	Значение для загрузки в старшее слово регистра сравнения RTCCMPx
RTCCMPx_L	15–0	Значение для загрузки в младшее слово регистра сравнения RTCCMPx

А.13 Регистры блока АЦП

Таблица А.13.1 – Регистр управления АЦП

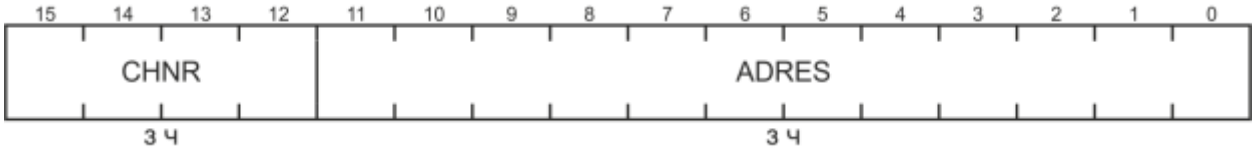
ADC_CON



Поле	Биты	Описание	
ADCTC	15-14	Управление временем преобразования АЦП Поле задает основную частоту преобразования $f_П$	
		00	fп = fadc/8
		01	fп = fadc/4
		10	fп = fadc/32
		11	fп = fadc/16
ADPWD	12	Бит разрешения режима PowerDown	
		0	Запрещено
		1	Разрешено
ADWR	9	Бит включения режима ожидания чтения	
		0	Выключен
		1	Включен
ADBSY	8	Флаг занятости АЦП	
		0	АЦП в состоянии простоя
		1	АЦП активен
ADST	7	Бит запуска преобразования АЦП	
		0	Остановить преобразование
		1	Начать преобразование
ADM	5-4	Режим работы АЦП	
		00	Однократное преобразование фиксированного канала
		01	Непрерывное преобразование фиксированного канала
		10	Автоматическое однократное сканирование
		11	Автоматическое непрерывное сканирование
ADCH	3-0	Выбор канала АЦП	
–	13, 11, 10, 6	Зарезервировано	

Таблица А.13.2 – Регистр управления АЦП

ADC_DAT

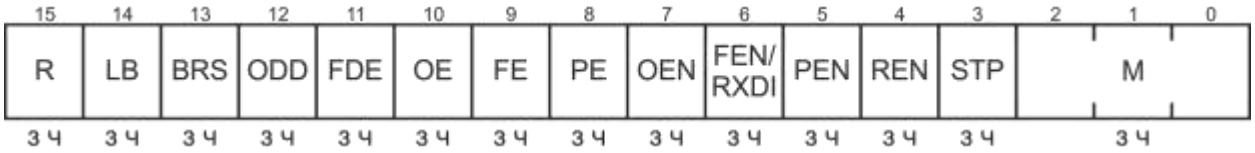


Поле	Биты	Описание
CHNR	15-12	Номер канала преобразования АЦП
ADRES	11-0	Результат последнего преобразования АЦП

A.14 Регистры блоков ASC0, ASC1, USART2

Таблица А.14.1 – Регистр управления ASCx (x – 0, 1 для ASC0, ASC1 и 2 для USART2)

SxCON



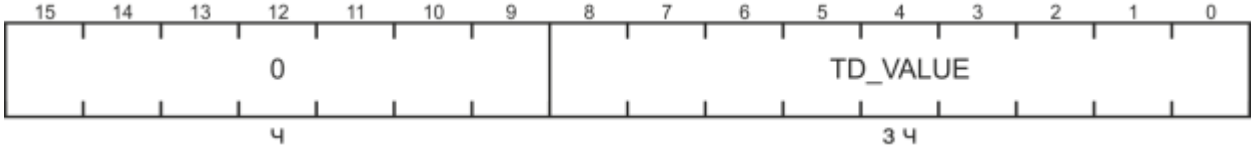
Поле	Бит	Описание	
R	15	Бит включения генератора	
		0	Выключен (модуль ASCx выключен)
		1	Включен
LB	14	Бит разрешения режима петли (замыкания выхода на вход модуля)	
		0	Режим стандартной передачи/получения данных
		1	Режим петли
BRS	13	Выбор скорости пересылки данных	
		0	Деление частоты генератора на сумму запрограммированного значения и постоянной величины (зависит от режима)
		1	Дополнительное уменьшение частоты тактового генератора на 2/3
ODD	12	Бит выбора типа проверки четности	
		0	По четным (проверка на четное количество единиц в данных)
		1	По нечетным (проверка на нечетное количество единиц в данных)
FDE	11	Бит включения дробного делителя	
		0	Выключен
		1	Включен и используется как делитель для генератора скорости передачи
OE	10	Флаг ошибки переполнения. Устанавливается аппаратно, если разрешено битом OEN. Сбрасывается программно	
FE	9	Флаг ошибки фрейма. Устанавливается аппаратно, если разрешено битом FEN. Сбрасывается программно	
PE	8	Флаг ошибки четности. Устанавливается аппаратно, если разрешено битом PEN. Сбрасывается программно	
OEN	7	Бит включения проверки ошибки переполнения	
		0	Проверка не осуществляется
		1	Проверка включена
FEN/ RXDI	6	Бит включения проверки ошибки фрейма (для асинхронных режимов)	
		0	Ошибки фрейма игнорируются
		1	Проверка включена
PEN	5	Бит включения проверки четности (для асинхронных режимов)	
		0	Игнорировать проверку четности
		1	Проверять четность
REN	4	Бит разрешения приема	
		0	Запрещен
		1	Разрешен

Окончание таблицы А.14.1

Поле	Бит	Описание		
STP	3	Выбор количества стоповых бит		
		0	Один	
		1	Два	
M	2–0	Выбор режима работы модуля ASCx		
		000	8-разрядные данные	Синхронный режим
		001	8-разрядные данные	Асинхронные режимы
		010	Зарезервировано	
		011	7-разрядные данные и четность	
		100	9-разрядные данные	
		101	Зарезервировано	
		110	Зарезервировано	
		111	8-разрядные данные и четность	

Таблица А.14.2 – Буферный регистр передатчика

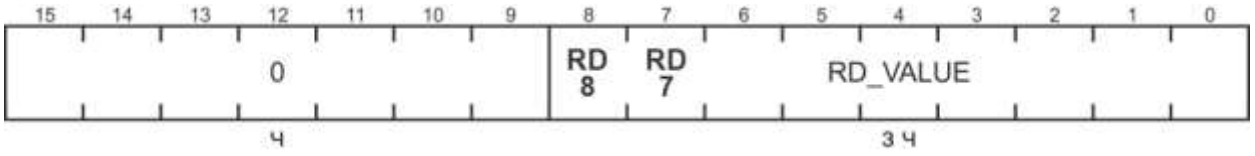
SxTBUF



Поле	Бит	Описание
TD_VALUE	8–0	Значение данных буферного регистра передатчика. Данные для передачи.
0	15–9	Зарезервировано

Таблица А.14.3 – Буферный регистр приемника

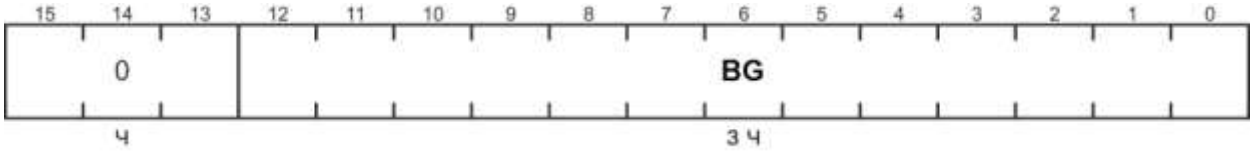
SxRBUF



Поле	Биты	Описание
RD_VALUE	8–0	Значение данных буферного регистра приемника. Полученные данные и бит четности (в зависимости от выбранного режима). В асинхронном режиме полученный бит четности записывается в: - бит RD7 при M = 011b; - бит RD8 при M = 111 b.
0	15–9	Зарезервировано

Таблица А.14.4 – Регистр таймера генератора скорости передачи

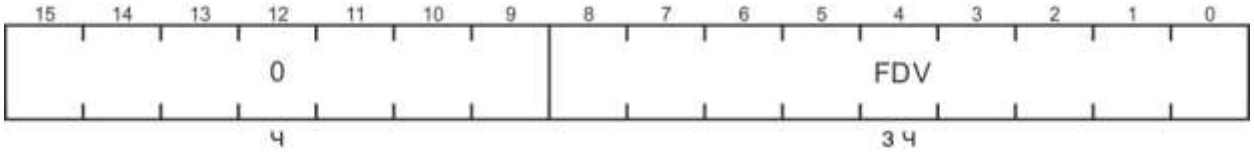
SxBG



Поле	Биты	Описание
BG	12–0	Значение для перезагрузки счетчика. Запись разрешена только при сброшенном бите R регистра SxCON
–	15–13	Зарезервировано

Таблица А.14.5– Регистр дробного делителя

SxFDV

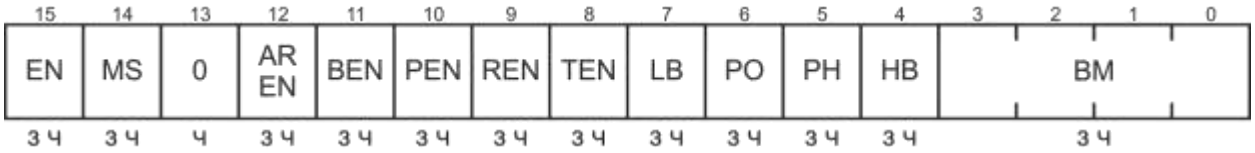


Поле	Биты	Описание
FDV	8–0	Значение коэффициента дробного делителя
–	15–9	Зарезервировано

A.15 Регистры блоков SSC0, SSC1, SPI

Таблица А.15.1 – Регистр управления SSCx (x – 0, 1 для SSC0, SSC1 и 2 для SPI)

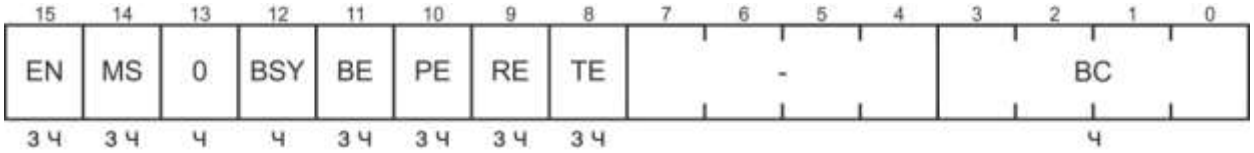
SSCxCON (режим программирования EN = 0)



Поле	Биты	Описание	
EN	15	Бит разрешения работы модуля	
		0	Передача и прием данных отключены, имеется доступ к битам управления
		1	Разрешено
MS	14	Режим функционирования модуля	
		0	Ведомый
		1	Мастер
0	13	Зарезервирован	
AREN	12	Бит разрешения автоматического перезапуска модуля	
		0	Только отслеживание ошибки скорости передачи
		1	Автоматический перезапуск модуля при обнаружении ошибки
BEN	11	Бит включения проверки скорости передачи	
		0	Ошибки скорости передачи игнорируются
		1	Включено
PEN	10	Бит включения проверки фазы	
		0	Ошибки фазы игнорируются
		1	Включено
REN	9	Бит включения контроля приема	
		0	Ошибки приема игнорируются
		1	Включено
TEN	8	Бит включения контроля передачи	
		0	Ошибки передачи игнорируются
		1	Включено
LB	7	Бит управления режимом петли (замыкание выхода на вход модуля)	
		0	Нормальный режим работы
		1	Режим петли (полудуплексный режим)
PO	6	Уровень сигнала тактового сигнала в режиме ожидания	
		0	Ноль. Передний фронт – переход из нуля в единицу
		1	Единица. Передний фронт – переход из единицы в ноль
PH	5	Фаза тактового сигнала	
		0	Установка данных по переднему фронту, захват по заднему фронту
		1	Установка данных по заднему фронту, захват по переднему фронту
HB	4	Бит управления направлением передачи данных	
		0	Младшим битом вперед
		1	Старшим битом вперед
BM	3–0	Выбор длины данных	
		0h	Зарезервировано. Длина данных не задана.
		1h – Fh	Ширина передаваемых данных от 2 до 16 бит

Таблица А.15.2 – Регистр управления SSCx

SSCxCON (операционный режим EN = 1)

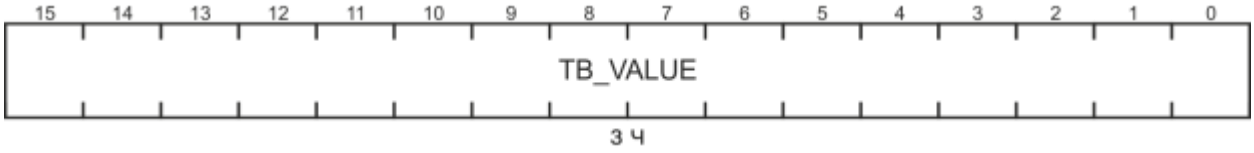


Поле	Биты	Описание	
EN	15	Бит разрешения работы модуля	
		0	Передача и прием данных отключены, имеется доступ к битам управления
		1	Разрешено
MS	14	Режим функционирования модуля	
		0	Ведомый
		1	Мастер
BSY	12	Флаг занятости. Устанавливается аппаратно во время передачи данных. Сбрасывается аппаратно по окончании передачи данных.	
BE	11	Флаг ошибки скорости передачи	
PE	10	Флаг ошибки фазы	
RE	9	Флаг ошибки приема	
TE	8	Флаг ошибки передачи	
BC	3–0	Поле подсчета битов	
–	13, 7–4	Зарезервировано	

Примечание – Для битов BE, PE, RE и TE справедливо: 0b – ошибка не обнаружена, 1b – ошибка обнаружена и генерируется прерывание. Биты сбрасываются только программно.

Таблица А.15.3 – Буферный регистр данных для передачи

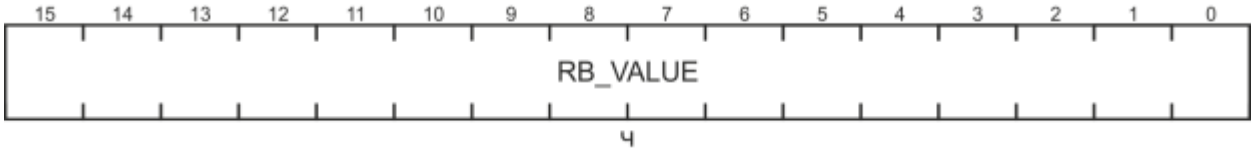
SSCxTB



Поле	Биты	Описание
TB_VALUE	15 – 0	Данные для передачи. Если длина передаваемых данных менее 16 бит, старшие, неиспользуемые биты игнорируются

Таблица А.15.4 – Буферный регистр принятых данных

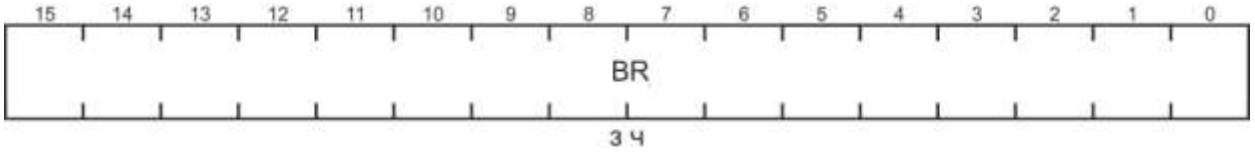
SSCxRB



Поле	Биты	Описание
RB_VALUE	15–0	Принятые данные. Если длина принимаемых данных менее 16 бит, старшие, неиспользуемые биты игнорируются

Таблица А.15.5 – Регистр скорости передачи

SSCxBR



Поле	Биты	Описание
BR	15–0	Значение перезагрузки счетчика. Запись разрешена только при сброшенном бите EN регистра SSCxCON, т. е. в режиме программирования.

A.16 Регистры контроллера I2C

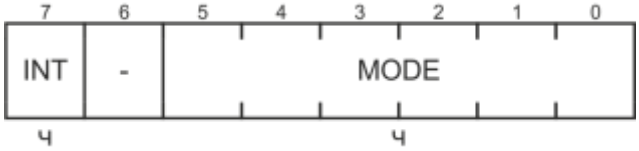
Таблица А.16.1 – Сдвиговый регистр данных

SMBSDA

Поле	Бит	Описание
DATA	7–0	Поле данных

Таблица А.16.2 – Регистр состояния

SMBST



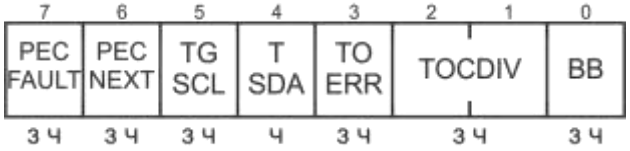
Поле	Бит	Описание
INT	7	Флаг прерывания. Устанавливается после девятого такта сигнала SCL (когда SCL = 0) в любое запрограммированное время. Условия выставления флага INT: - во время приема/передачи как в режиме мастера, так и в режиме ведомого; - при совпадении адреса (адреса ведомого, адреса отклика 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)

Окончание таблицы А.16.2

Поле	Бит	Описание
MODE	5–0	Код состояния. Возникновение того или иного состояния в течение функционирования модуля I2C сопровождается записью соответствующего кода в поле MODE
–	6	Зарезервировано

Таблица А.16.3 – Регистр управления и статуса

SMBCST



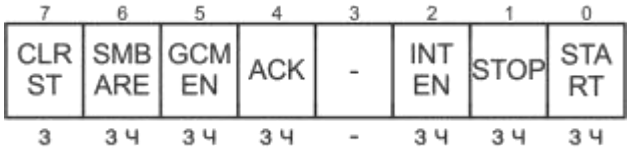
Поле	Бит	Описание
PECFAULT	7	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной суммой, значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, 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
TGSCL	5	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	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

Окончание таблицы А.16.3

Поле	Бит	Описание	
TOERR	3	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	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	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.16.4 – Регистр 1 управления

SMBCTL1



Поле	Бит	Описание	
CLRST	7	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	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	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	

Окончание таблицы А.16.4

Поле	Бит	Описание
ACK	4	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	Бит разрешения прерывания
		0 Запрещено
		1 Разрешено
STOP	1	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	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

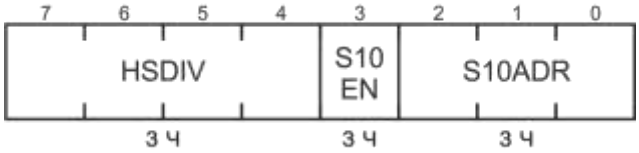
Таблица А.16.5 – Регистр 2 управления

SMBCTL2

Поле	Бит	Описание
SCLFRQ	7–1	Поле выбора частоты fscl сигнала на выводе SCL в режиме мастера. Длительности высокого (Tsclh) и низкого(Tscll) уровней сигнала SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формуле Tsclh = Tscll = 2 × SCLFRQ × (1/fosc). (A.16.1) Таким образом, частота сигнала на выводе SCL равна fscl = 1/(Tsclh + Tscll). (A.16.2) 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	Бит включения модуля I2C
		0 Модуль выключен. Тактирование не осуществляется. Регистры SMBCTRL1, SMBST, SMBCST сброшены
		1 Модуль включен

Таблица А.16.6 – Регистр 3 управления

SMBCTL3



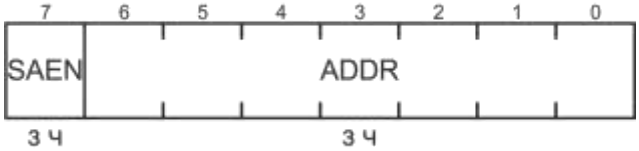
Поле	Бит	Описание
HSDIV	7–4	Поле выбора частоты fscl сигнала на выводе SCL в режиме HS мастера. Длительности высокого(Thsclh) и низкого(Thscll) уровней сигнала на выводе SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формулам $T_{HSCLH} = HSDIV \times (1/fosc)$, (A.16.3) $T_{HSCLL} = 2 \times HSDIV \times (1/fosc)$. (A.16.4) Таким образом, частота сигнала на выводе SCL равна $fscl = 1/(Thsclh + Thscll)$. (A.16.5) 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	Бит разрешения 10-битной адресации ведомого
		0 – Запрещена
		1 – Разрешена при условии, что установлен бит SAEN в регистре SMBADDR
S10ADR	2–0	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.16.7 – Регистр собственного адреса

SMBADDR



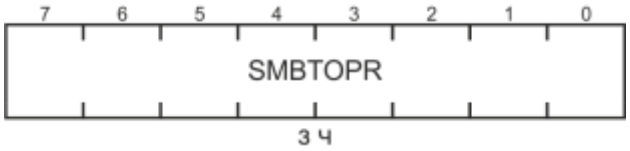
Поле	Бит	Описание
SAEN	7	Бит разрешения распознавания адреса
		0 – Безадресный режим
		1 – Включена функция распознавания принятого адреса

Окончание таблицы А.16.7

Поле	Биты	Описание
ADDR	6–0	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.16.8 – Регистр загрузки предделителя

SMBTOPR

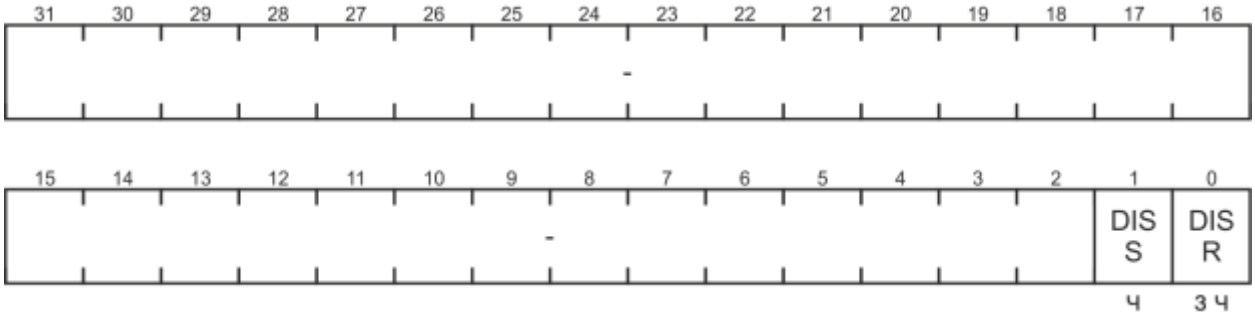


Поле	Бит	Описание
SMBTOPR	7–0	Поле значения перезагрузки предделителя

A.17 Регистры контроллера CAN

Таблица А.17.1 – Регистр управления частотой

CLC

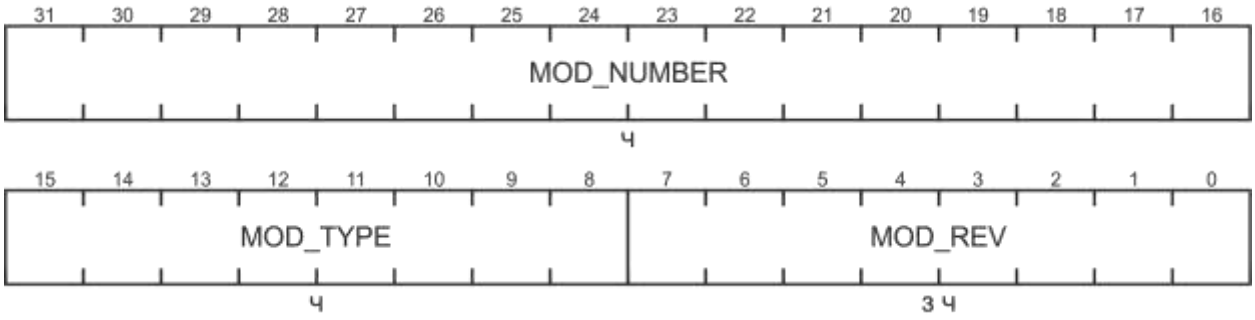


Поле	Бит	Описание	
DISS	1	Бит состояния контроллера CAN	
		0	Включен
		1	Выключен
DISR	0	Бит выключения контроллера CAN	
		0	Нет действий
		1	Запись единицы запускает механизм выключения
–	31-2	Зарезервировано	

Примечание – Когда контроллер CAN находится в выключенном состоянии, только регистр CLC доступен для записи и чтения, доступ к остальным регистрам не возможен.

Таблица А.17.2 – Регистр идентификации

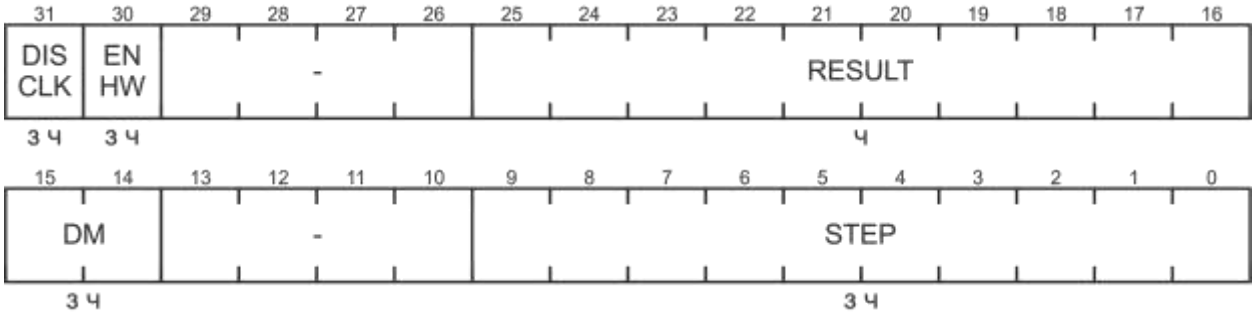
ID



Поле	Биты	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.17.3 – Регистр делителя

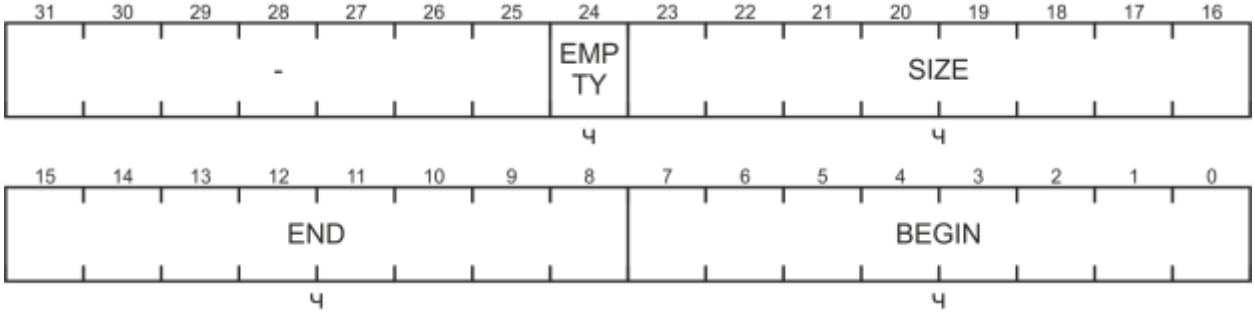
FDR



Поле	Биты	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	
RESULT	25-16	Счетчик делителя частоты	
DM	15-14	Поле задания режима делителя частоты	
		00 11	Счетчик выключен. Синхросигнал FOUT не генерируется. Сигнал сброса внешнего делителя в состоянии логической единицы. Поле RESULT не меняется.
		01	Нормальный режим работы. Сигнал сброса внешнего делителя в состоянии логического нуля. При активации режима поле RESULT загружается значением 3FFh. Далее периодически загружается значением из STEP. Формируется сигнал FOUT.
		10	Режим дробного деления. Сигнал сброса внешнего делителя в состоянии логического нуля. При активации режима поле RESULT загружается значением 3FFh. Далее периодически загружается значением из STEP. Формируется сигнал FOUT.
STEP	9-0	Шаг делителя. Поле хранит значение, которое загружается в RESULT при переполнении счетчика делителя	
–	29-26, 13-10	Зарезервировано	

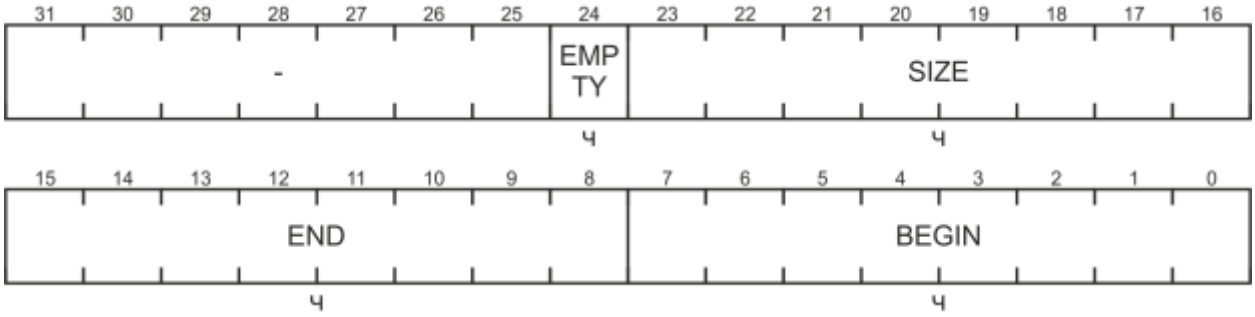
Таблица А.17.4 – Регистр списка №0 и регистр свободного списка x (x от 1 до 7)

LIST0



Окончание таблицы А.17.4

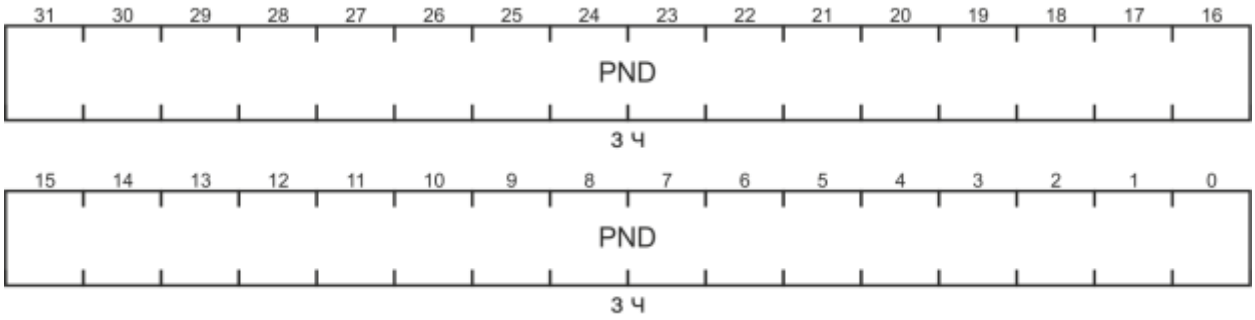
LISTx



Поле	Биты	Описание	
EMPTY	24	Индикатор пустого списка	
		0	В списке есть как минимум один элемент
		1	Список пуст
SIZE	23-16	Размер списка. Количество элементов (объектов сообщений) в списке. Значение поля SIZE всегда на единицу меньше числа элементов. Если список пуст, SIZE = 00h	
END	15-8	Номер объекта сообщения, находящегося последним в списке. Поле может принимать значения от 00h до FFh, согласно количеству объектов сообщений (256)	
BEGIN	7-0	Номер объекта сообщения, находящегося первым в списке. Поле может принимать значения от 00h до FFh, согласно количеству объектов сообщений	
–	31-25	Зарезервировано	

Таблица А.17.5 – Регистр ждущих прерываний (x от 0 до 7)

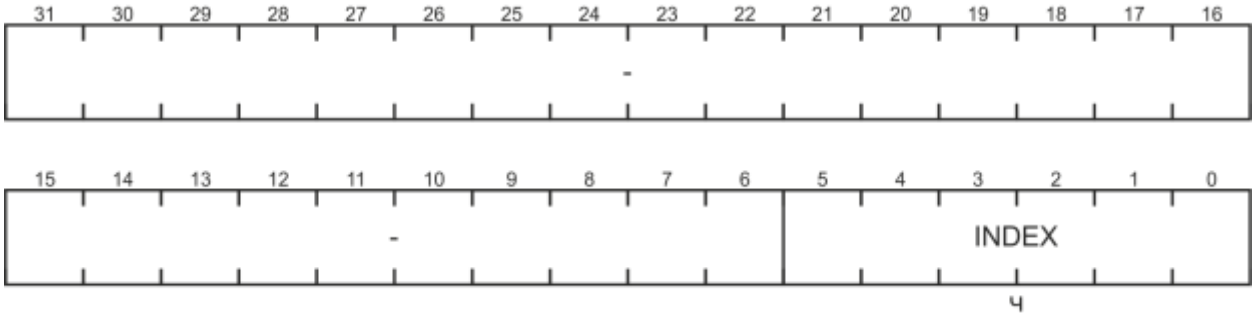
MSPNDx



Поле	Биты	Описание
PND	31-0	Поле ждущих битов сообщений. Каждому объекту сообщения выделяется один бит. Биты устанавливаются только аппаратно. Установленные биты сбрасываются аппаратно по окончании обслуживания запроса прерывания или могут быть сброшены в любой момент программно

Таблица А.17.6 – Регистр индекса сообщения (x – от 0 до 7)

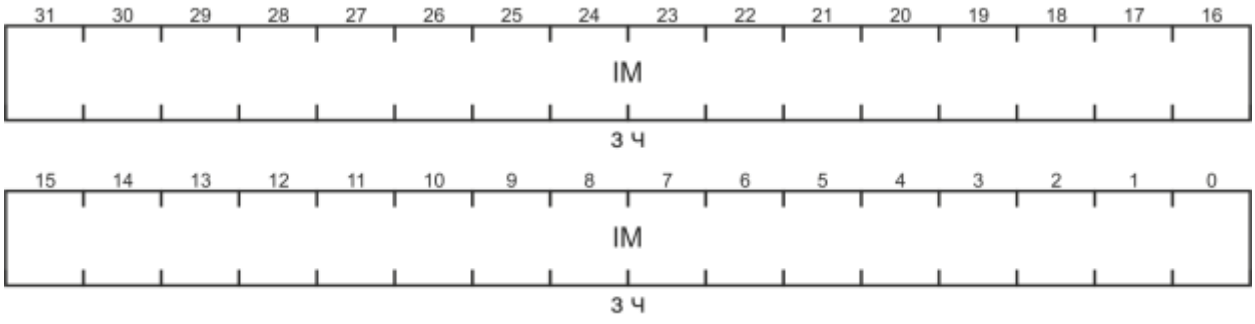
MSIDx



Поле	Биты	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.17.7 – Регистр маски индекса сообщения

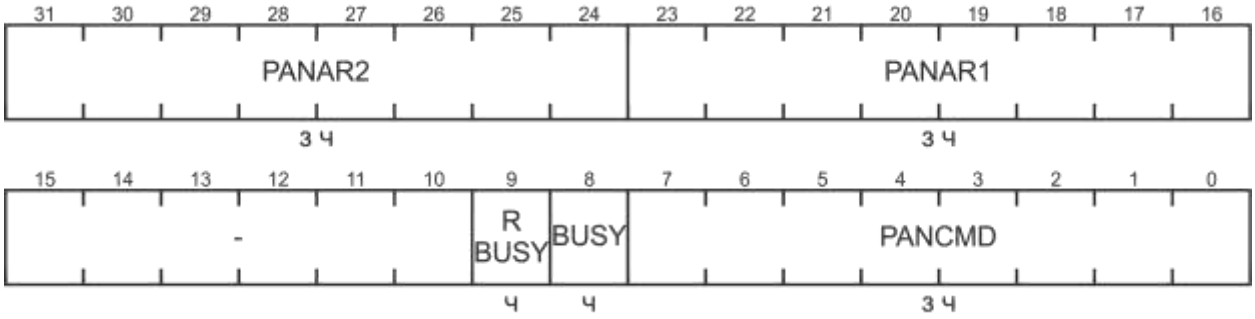
MSIMASK



Поле	Биты	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.17.8 – Регистр панели команд

PANCTR



Поле	Биты	Описание	
PANAR2	31-24	Панель аргумента 2 (см. таблицу А.17.9)	
PANAR1	23-16	Панель аргумента 1 (см. таблицу А.17.9)	
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.17.9). После выполнения команды в это поле записывается 00h	
–	15-10	Зарезервировано	

Таблица А.17.9 – Коды команд работы со списками

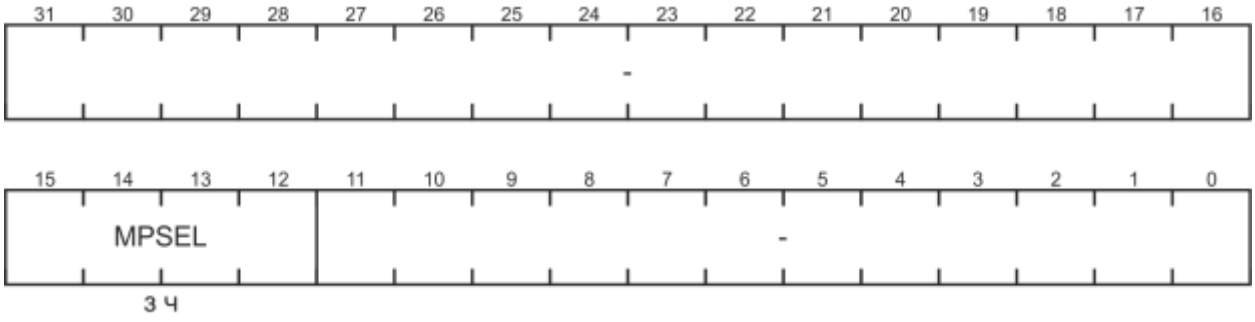
PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
00h	–	–	Нет операции. Никаких действий не выполняется
01h	Результат: бит 7 – ошибка, бит 6 – не определен	–	Инициализация списков. Запуск инициализации для очистки битовых полей CTRL и LIST всех объектов сообщений. Регистры LIST0 – LIST8 устанавливаются 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

Окончание таблицы А.17.9

PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
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)
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 сообщений 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.17.10 – Регистр управления

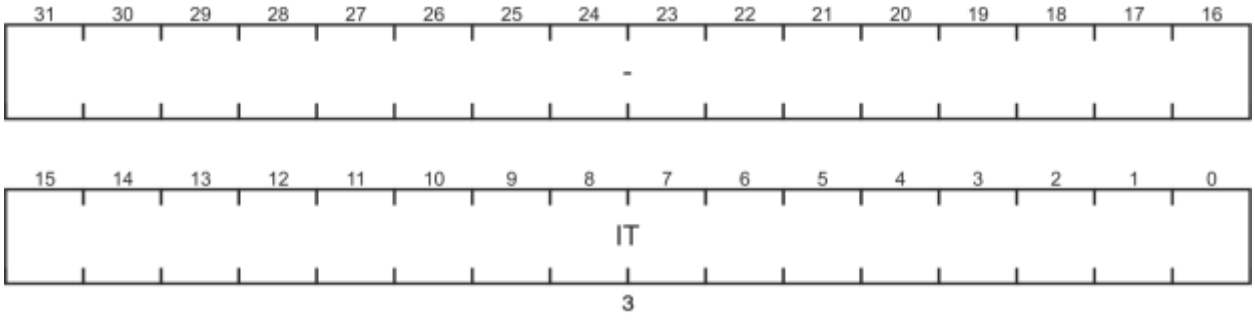
MCR



Поле	Биты	Описание
MPSEL	15-12	Поле задания позиции ждущего бита сообщения после приема/ передачи сообщения
–	31-16, 11-0	Зарезервировано

Таблица А.17.11 – Регистр прерываний

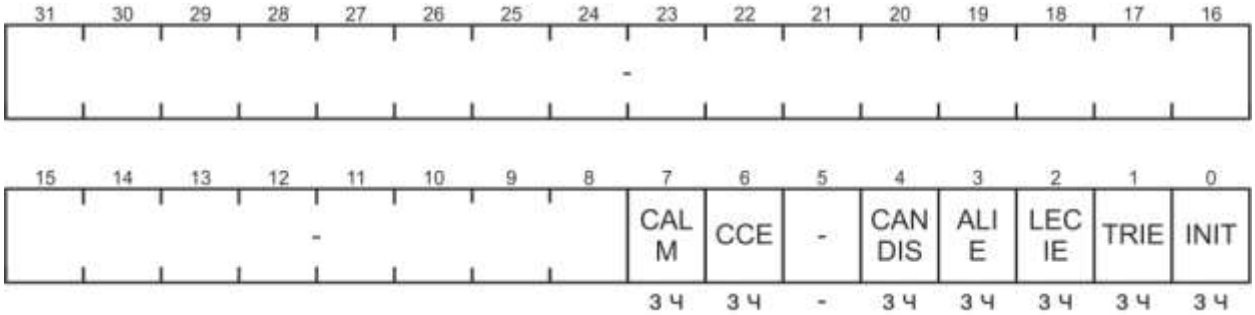
MITR



Поле	Биты	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.17.12 – Регистр управления узла

NCR



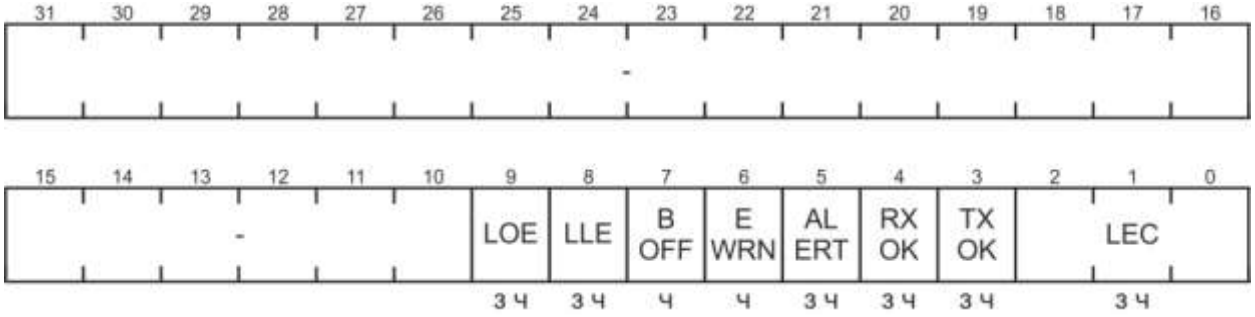
Поле	Бит	Описание	
CALM	7	Бит включения режима анализа узла	
		0	Режим выключен
		1	Установка бита включает режим анализа узла. В этом режиме сообщения могут только приниматься, бит подтверждения не посылается после успешного приема сообщения, флаг активной ошибки посылается рецессивным вместо доминантного. 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 NBTRx, NPCRx и NECNTx	
		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
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

Окончание таблицы А.17.12

Поле	Бит	Описание	
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, 8, 5	Зарезервировано	

Таблица А.17.13 – Регистр состояния узла

NSR



Поле	Биты	Описание	
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 SIZE соответствующего регистра списка
		Бит должен сбрасываться программно записью нуля	

Продолжение таблицы А.17.13

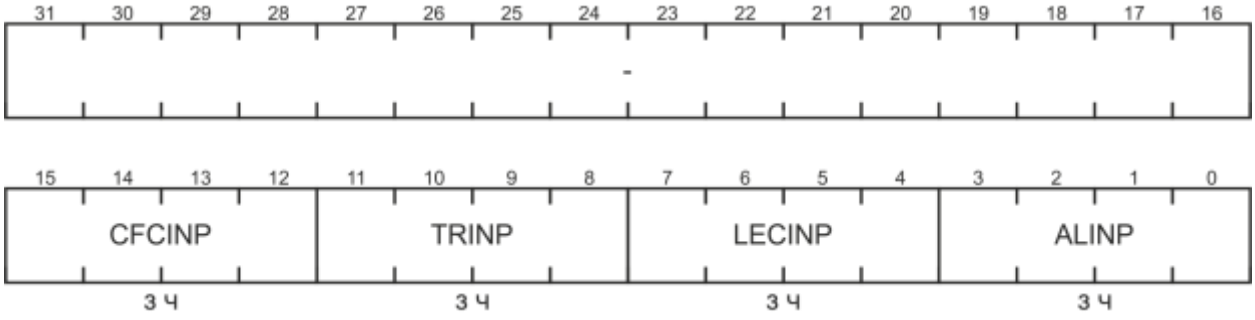
Поле	Биты	Описание	
BOFF	7	Флаг состояния «отключен от шины»	
		0	Узел не находится в состоянии «отключен от шины»
		1	Узел находится в состоянии «отключен от шины»
EWRN	6	Флаг критического количества ошибок	
		0	Лимит ошибок еще не достигнул
		1	По крайней мере, один из счетчиков ошибок (REC, TEC) достиг лимита ошибок, заданного полем EWRNLVL регистра NECNT узла
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»

Окончание таблицы А.17.13

Поле	Биты	Описание	
LEC	2-0	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
		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	
–	31-10	Зарезервировано	

Таблица А.17.14 – Регистр указателя прерываний узла

NIPR



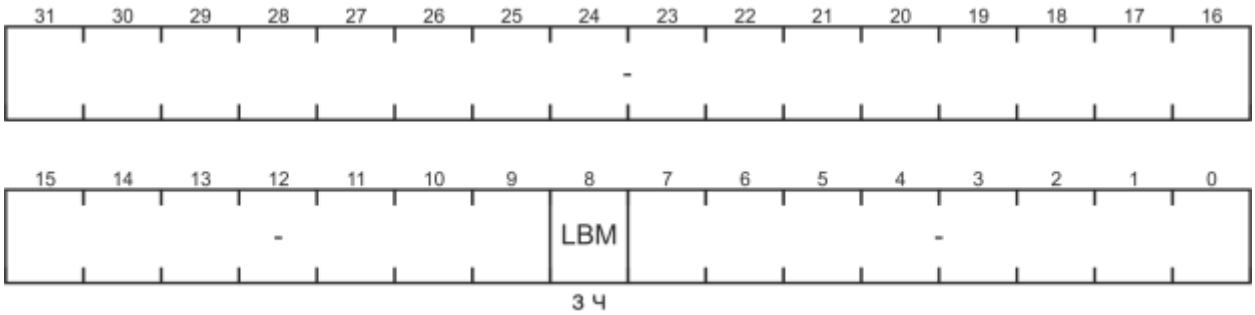
Поле	Биты	Описание
CFCINP	15-12	Указатель линии прерывания для прерывания при переполнении счетчика фреймов узла
TRINP	11-8	Указатель линии прерывания для прерывания по окончании передачи/приема сообщения

Окончание таблицы А.17.14

Поле	Биты	Описание
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
Примечания 1 Каждый из указателей позволяет задать номер одной из 16 линий прерываний для каждого из четырех источников. 2 Значение 00h соответствует нулевой линии прерываний, значение 01h – первой и так далее до значения FFh, которое соответствует линии 15 прерываний		

Таблица А.17.15 – Регистр управления портом узла

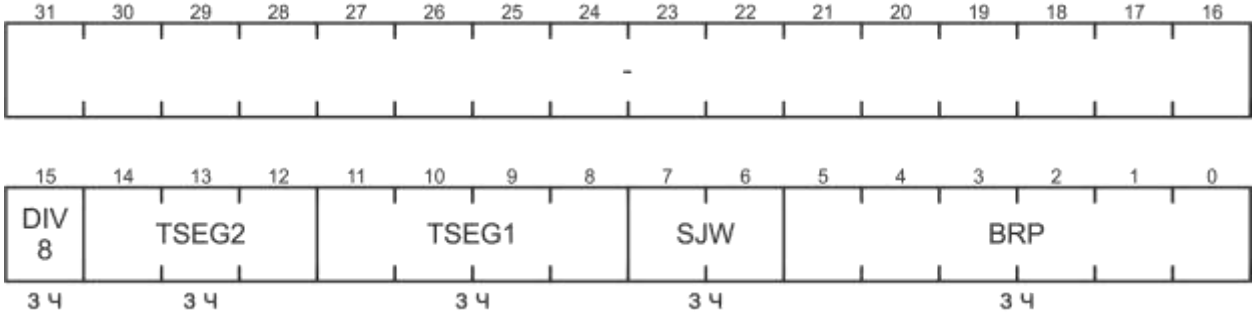
NPCR



Поле	Бит	Описание	
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 не активны
–	31-9, 7-0	Зарезервировано	

Таблица А.17.16 – Регистр синхронизации битов

NBTR

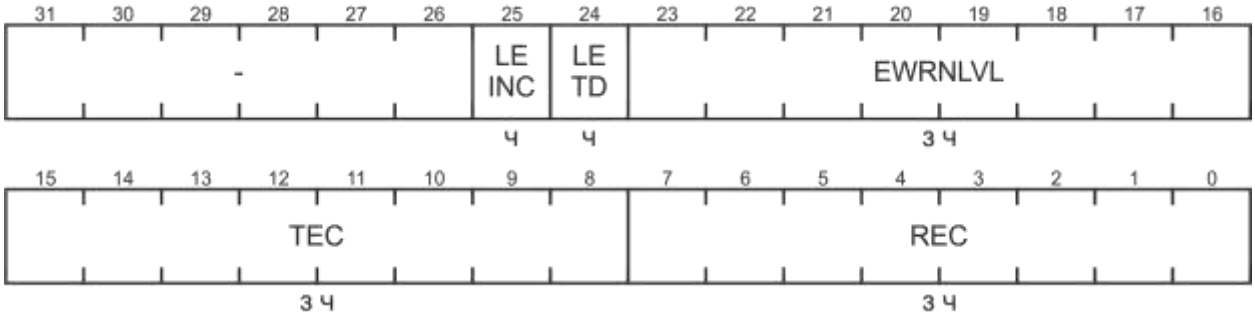


Окончание таблицы А.17.16

Поле	Биты	Описание	
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	
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.17.17 – Регистр счетчика ошибок узла

NECNT



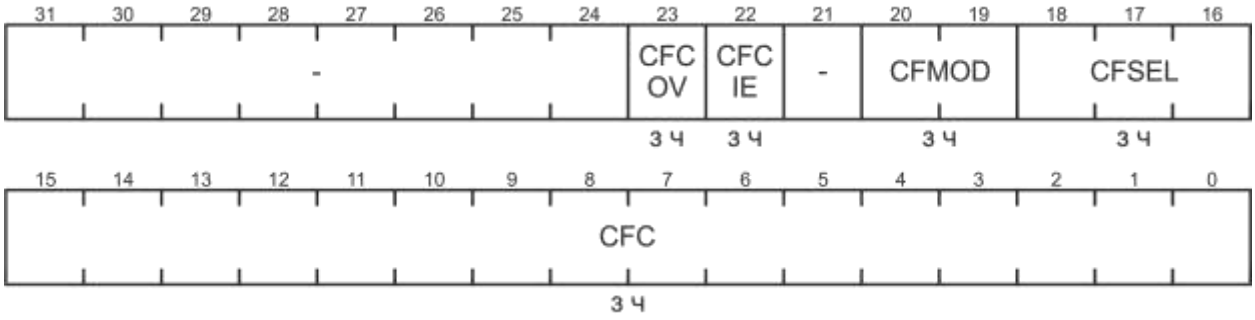
Поле	Биты	Описание	
1	2	3	
LEINC	25	Индикатор инкрементирования при последней ошибке	
		0	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на единицу
		1	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на восемь
LETD	24	Флаг последней ошибки передачи	
		0	При приеме сообщения обнаружена ошибка, и произошло инкрементирование поля REC
		1	При передаче сообщения обнаружена ошибка, и произошло инкрементирование поля TEC

Окончание таблицы А.17.17

1	2	3
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 передачи сообщений
REC	7-0	Поле счетчика ошибок приема сообщений
–	31-26	Зарезервировано

Таблица А.17.18 – Регистр счетчика сообщений узла

NFCR



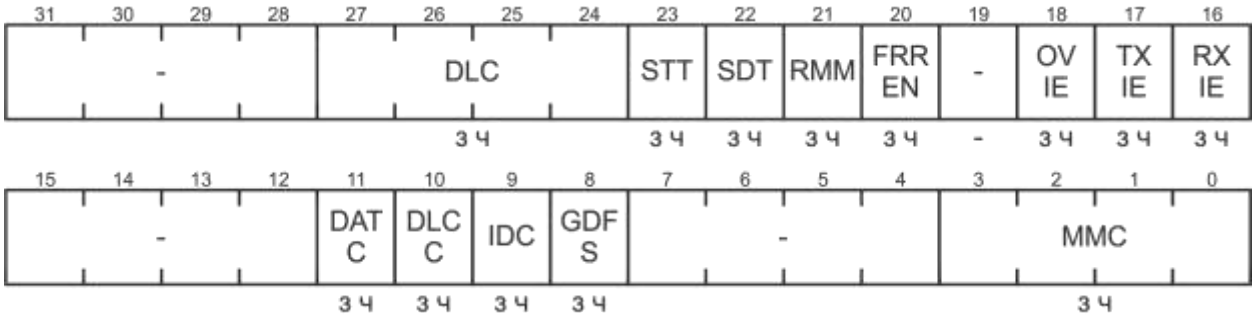
Поле	Биты	Описание	
CFCOV	23	Флаг переполнения счетчика сообщений	
		0	Счетчик не переполнен
		1	Счетчик переполнился. В режиме синхросчетчика этот флаг устанавливается при изменении поля CFC, если установлен бит CFCIE, формируется прерывание
		Бит сбрасывается программно	
CFCIE	22	Бит разрешения прерывания 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
		01-11	Зарезервировано. Не использовать!
CFSEL	18-16	Поле задания параметров выбранного режима счетчика сообщений (см. таблицу А17.19)	
CFC	15-0	Поле счетчика сообщений Хранит значение счетчика сообщений при CFMOD = 00b	
–	31-24, 21	Зарезервировано	

Таблица А.17.19 – Коды задания параметров режима счетчика сообщений

CFSEL			Действия
*	*	1	Счетчик инкрементируется каждый раз при получении сообщения, не имеющего объекта сообщения
*	1	*	Счетчик инкрементируется каждый раз при получении сообщения, имеющего соответствующий объект сообщения
1	*	*	Счетчик инкрементируется каждый раз при успешной отправке сообщения
0	0	0	Зарезервировано. Не использовать!
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)			

Таблица А.17.20 – Регистр управления функционированием объекта сообщения

MOFCR



Поле	Бит	Описание	
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 успешного приема данных.
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	

Продолжение таблицы А.17.20

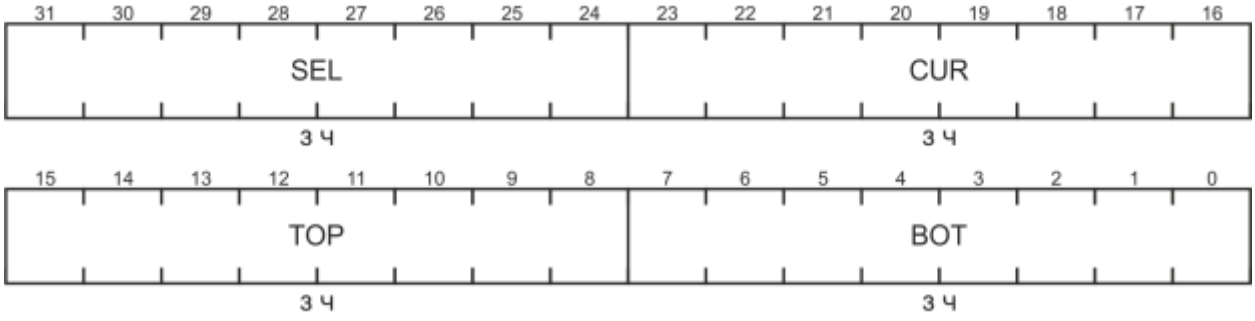
Поле	Бит	Описание	
FRREN	20	Бит разрешения удаленного запроса. Определяет, будет ли устанавливаться бит TXRQ в объекте сообщения n или в другом объекте сообщения, на который указывает CUR	
		0	Бит TXRQ объекта сообщения n устанавливается после получения корректного фрейма удаленного запроса
		1	Бит TXRQ другого объекта сообщения (на который указывает CUR) устанавливается после получения им корректного фрейма удаленного запроса
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	

Окончание таблицы А.17.20

Поле	Биты	Описание	
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	
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	Стандартный объект сообщения
		0001	Базовый объект приемной структуры FIFO
		0010	Базовый объект передающей структуры FIFO
		0011	Вспомогательный объект передающей структуры 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.			

Таблица А.17.21 – Регистр указателя FIFO/шлюза объекта сообщения

MOFGPR



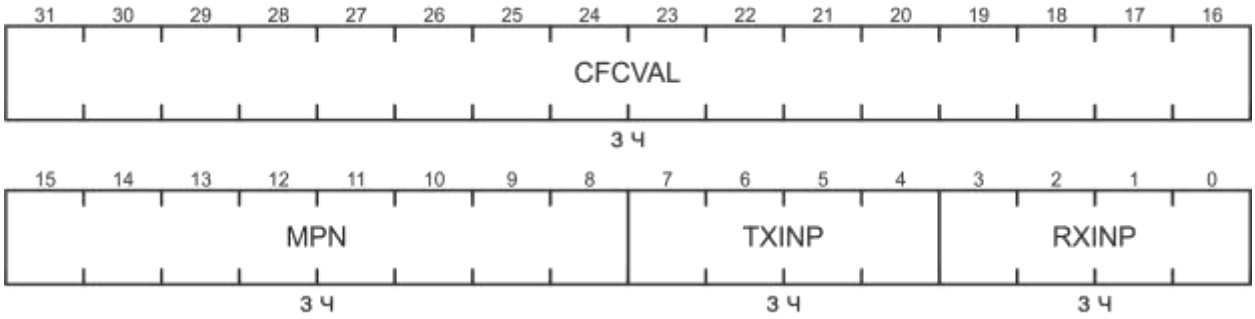
Поле	Биты	Описание
SEL	31-24	Указатель объекта сообщения. Второй (программный) указатель в дополнение к аппаратному указателю CUR при работе с FIFO. Поле SEL используется для общего мониторинга (генерирование прерываний FIFO)

Окончание таблицы А.17.21

Поле	Биты	Описание
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.17.22 – Регистр указателя прерываний объекта сообщения

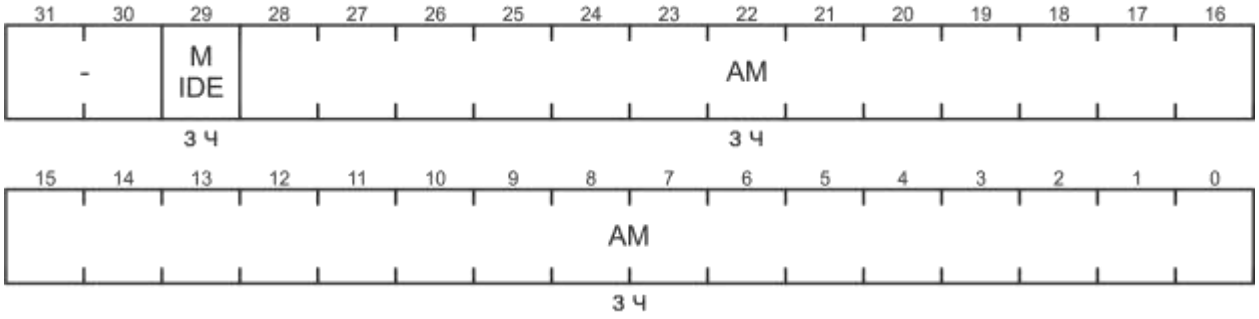
MOIPR



Поле	Биты	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.17.23 – Регистр маски объекта сообщения

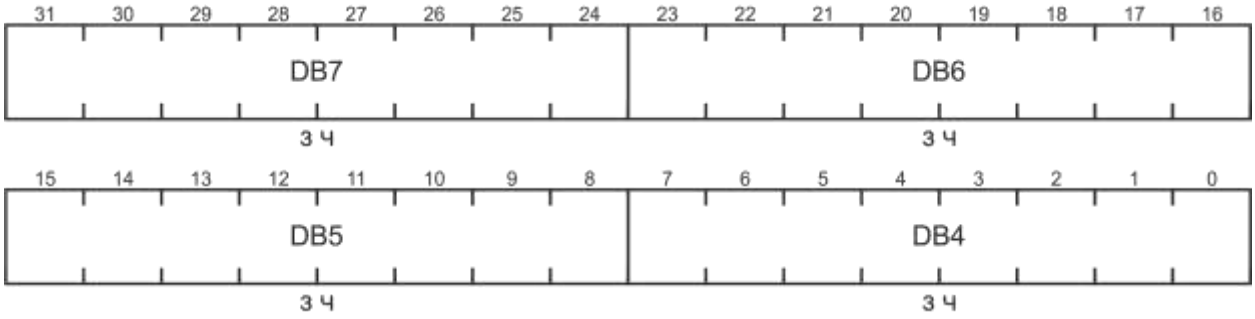
MOAMR



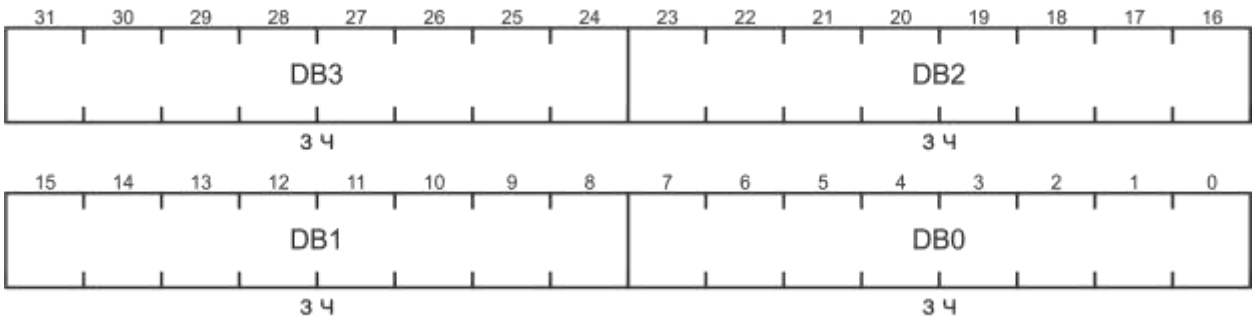
Поле	Биты	Описание	
MIDE	29	Маска бита IDE сообщения	
		0	Объект сообщения n может принимать как стандартные, так и расширенные фреймы
		1	Объект сообщения n может принимать только те фреймы, у которых состояние бита IDE совпадает с его битом IDE
AM	28-0	Маска идентификатора. При приеме расширенного 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.17.24 – Регистры данных объекта сообщения

MODATAH



MODATAL

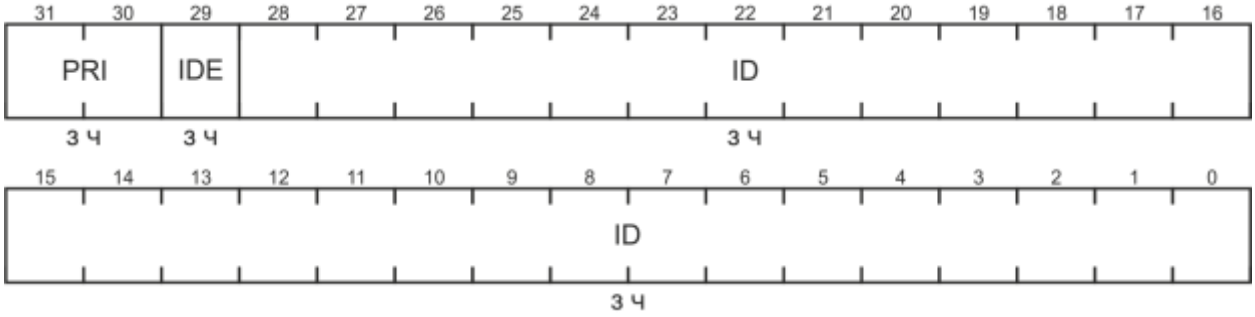


Окончание таблицы А.17.24

Поле	Биты	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.17.25 – Регистр арбитража объекта сообщения

MOAR



Поле	Биты	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 в таблице А3.26)
		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

Окончание таблицы А.17.25

Поле	Биты	Описание
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.17.26 – 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.17.27 – Регистр управления объектом сообщения

MOCTR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-				SET DIR	SET TXEN 1	SET TXEN 0	SET TXRQ	SET RXEN	SET RT SEL	SET MSG VAL	SET MSG LST	SET NEW DAT	SET RXUP D	SET TXPN D	RES RXPN D
				3	3	3	3	3	3	3	3	3	3	3	3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				RES DIR	RES TXEN 1	RES TXEN 0	RES TXRQ	RES RXEN	RES RT SEL	RES MSG VAL	RES MSG LST	RES NEW DAT	RES RXUP D	RES TXPN D	RES RXPN D
				3	3	3	3	3	3	3	3	3	3	3	3

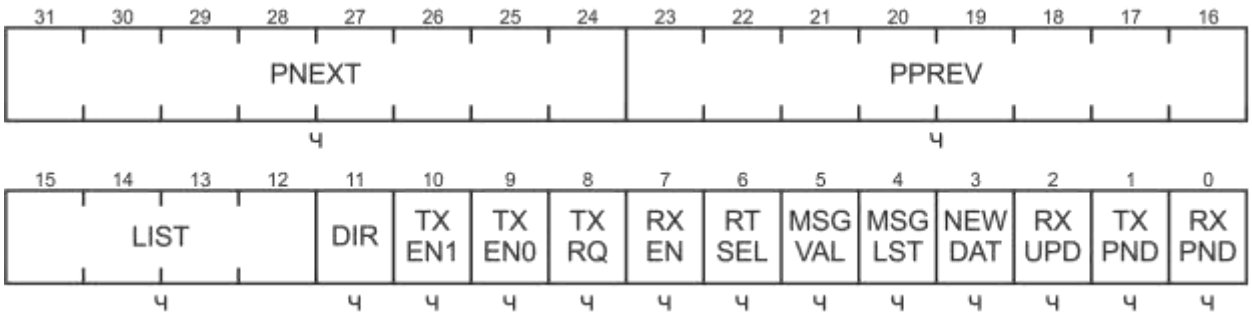
Биты регистра работают попарно. Комбинация состояний бит каждой пары оказывает влияние на один (соответствующий этой паре) бит регистра MOSTATn того же объекта сообщения. Так, пара SETDIR-RESDIR устанавливает и сбрасывает бит DIR, пара SETTXN1-RESTXN1 устанавливает и сбрасывает бит TXEN1 и т. д.

Окончание таблицы А.17.27

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 SETxxx	Бит RESxxx	Действие над битом
0	0	Нет
1	1	
1	0	Установка
0	1	Сброс
Биты 31–28 и 15–12 являются зарезервированными		

Таблица А.17.28 – Регистр состояния объекта сообщения

MOSTAT



Поле	Бит	Описание	
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	
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 сообщения данных. Объект принимает сообщение данных. При установленном бите TXRQ объект формирует сообщение удаленного запроса с идентификатором объекта n, а затем передает его. Полученное в ответ сообщение данных с соответствующим идентификатором сохраняется в объекте сообщения n.
		1	Объект передачи сообщения данных. При установленном бите TXRQ объект формирует, а затем передает сообщение данных. Если объект n получает сообщение удаленного запроса с соответствующим идентификатором, то устанавливается флаг TXRQ его регистра MOSTATn, после чего в ответ передается сообщение данных, содержащихся в объекте n.

Продолжение таблицы А.17.28

Поле	Бит	Описание	
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	
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	

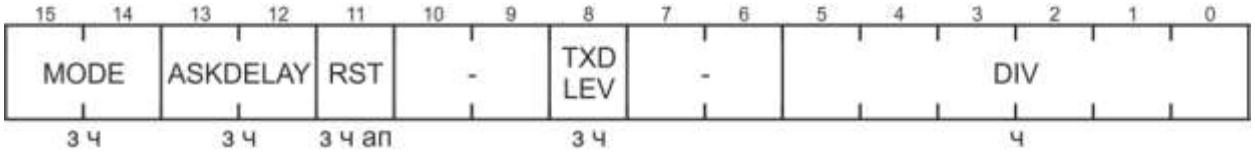
Окончание таблицы А.17.28

Поле	Бит	Описание	
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 сообщения 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 n было успешно передано
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

А.18 Регистры контроллеров ГОСТ Р 52070-2003

Таблица А.18.1 – Регистр конфигурации

BSICONFIG



Поле	Бит	Описание	
MODE	15-14	Режимы работы	
		00	Не используется
		01	Контроллер шины (КШ)
		10	Оконечное устройство (ОУ)
		11	Монитор шины (МШ)
ASK DELAY	13-12	Поле выбора значения времени ожидания ответного слова	
		00	14 мкс
		01	28 мкс
		10	56 мкс
		11	112 мкс
RST	11	Сброс устройства. Установка бита приводит к сбросу модуля и очистки всех его регистров. На данные, хранящиеся в ОЗУ, этот бит не влияет. Бит сбрасывается аппаратно	
TXD LEV	8	Бит выбора уровня сигнала. Бит управляет уровнем сигнала на выводах TxD, TxD_n при их неактивном состоянии	
		0	На выводах удерживается низкий уровень сигнала
		1	На выводах удерживается высокий уровень сигнала
DIV	5-0	Делитель входной частоты. Значение аппаратно зарезервировано и не может быть изменено. При чтении возвращается 05h.	
–	10, 9, 7, 6	Зарезервировано	

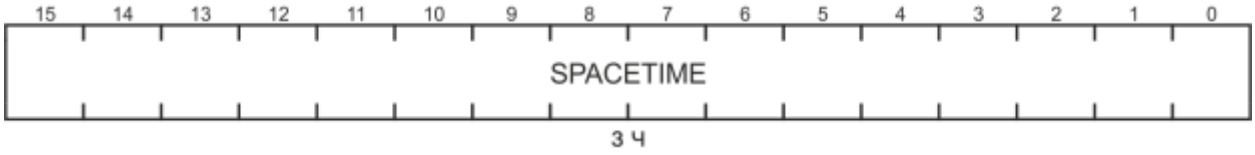
Таблица А.18.2 – Регистр адреса управляющего слова

BSISADDR

Поле	Бит	Описание
STARTADDR	15-0	Стартовый адрес. Адрес в памяти сообщений, по которому расположено управляющее слово первого блока сообщения. По умолчанию, стартовый адрес 0800h – начало памяти сообщений

Таблица А.18.3 – Регистр паузы

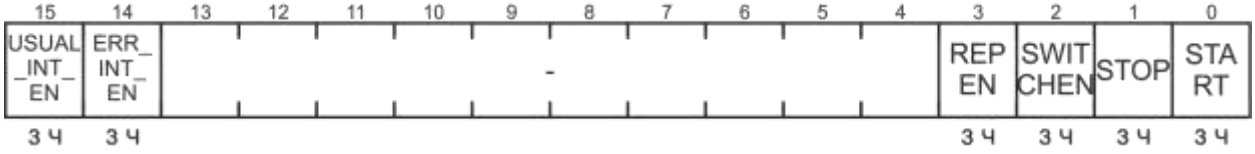
BSISPACE



Поле	Бит	Описание
SPACETIME	15-0	Пауза между сообщениями. Это битовое поле задает временной промежуток, которым контроллер шины разделяет передаваемые сообщения. Единицей измерения является одна микросекунда.

Таблица А.18.4 – Регистр управления

BSICON (в режиме КШ)



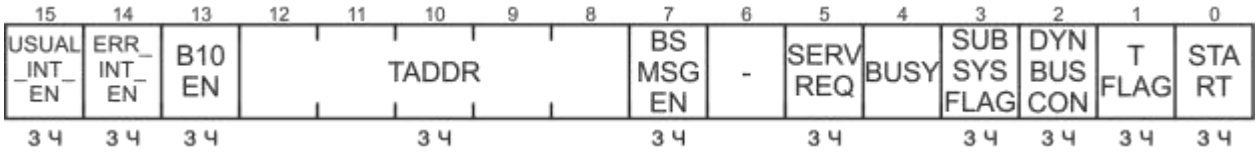
Поле	Бит		Описание
USUAL_INT_EN	15	0	Запрещено прерывание при успешном завершении сообщения
		1	Разрешено прерывание по окончании всей последовательности сообщений, а также после успешного окончания сообщения, в управляющем слове которого установлен бит MSGINTEN.
ERR_INT_EN	14	0	Запрещено прерывание, если сообщение завершено с ошибкой.
		1	Разрешено прерывание, если сообщение завершено с ошибкой и в управляющем слове этого сообщения установлен бит ERRINTEN.
REPEN	3	Бит разрешения повтора передачи. Управляет механизмом повторной отправки сообщения	
		0	Повтор в случае ошибки запрещен
		1	Количеством повторов управляет поле REPNUM управляющего слова
SWITCHEN	2	Бит разрешения смены канала. Управляет механизмом переключения канала передачи при повторных попытках передачи сообщения, вызванных обнаруженными ошибками	
		0	Запрещено
		1	Сменой канала управляет бит SWITCH управляющего слова
STOP	1	Бит останова. Используется для программного прекращения передачи цепочки сообщений. Установка бита во время передачи сообщения не прерывает текущую передачу. По окончании передачи бит сбрасывается аппаратно. После остановки модуль переходит в режим ожидания	

Окончание таблицы А.18.4

Поле	Бит	Описание
START	0	Бит запуска Установка бита запускает последовательность сообщений и запрещает изменения в BSICONFIG. . Сбрасывается аппаратно: - после успешной передачи сообщения, если не был установлен бит NOTLASTMSG управляющего слова; - после установки бита STOP, но только по окончании передачи текущего сообщения.
–	13-4	Зарезервировано

Таблица А.18.5 – Регистр управления

BSICON (в режиме ОУ)



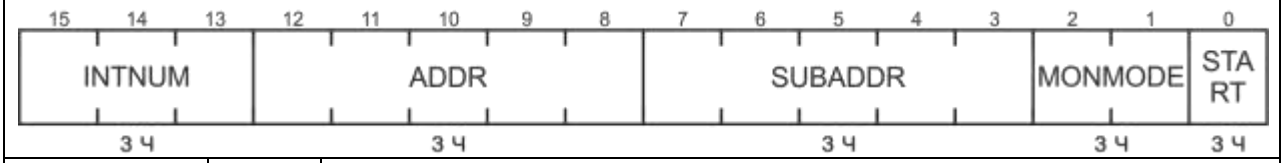
Поле	Бит	Описание	
USUAL_INT_EN	15	0	Запрещено прерывание при успешном завершении сообщения
		1	Разрешено прерывание после успешного окончания сообщения
ERR_INT_EN	14	0	Запрещено прерывание, если сообщение завершено с ошибкой
		1	Разрешено прерывание, если сообщение завершено с ошибкой
B10EN	13	Бит разрешения контроля признака слова	
		0	Запрещено.10-й бит принятого слова не проверяется
		1	Разрешено. Принятое слово считается командным, если имеет соответствующую форму синхронизации и логическую единицу в десятом бите
TADDR	12-8	Собственный адрес ОУ	
BCMSGEN	7	Бит разрешения работы с групповыми сообщениями	
		0	Запрещено. Адрес 1Fh игнорируется
		1	Разрешено. Адрес 1Fh распознается как групповой
SERVREQ	5	Бит запроса на обслуживание. Запись единицы в этот бит установит соответствующий бит в ответном слове	
BUSY	4	Бит занятости абонента. Запись единицы в этот бит установит соответствующий бит в ответном слове	
SUBSYSFLAG	3	Бит неисправности абонента. Запись единицы в этот бит установит соответствующий бит в ответном слове	
DYNBUSCON	2	Принято управление интерфейсом. Запись единицы в этот бит установит соответствующий бит в ответном слове	

Окончание таблицы А.18.5

Поле	Бит	Описание
TFLAG	1	Бит неисправности ОУ. Запись единицы в этот бит установит соответствующий бит в ответном слове
START	0	Бит включения ОУ. Пока этот бит установлен, собственный адрес ОУ и регистр BSICONFIG не могут быть изменены.
-	6	Зарезервировано

Таблица А.18.6 – Регистр управления

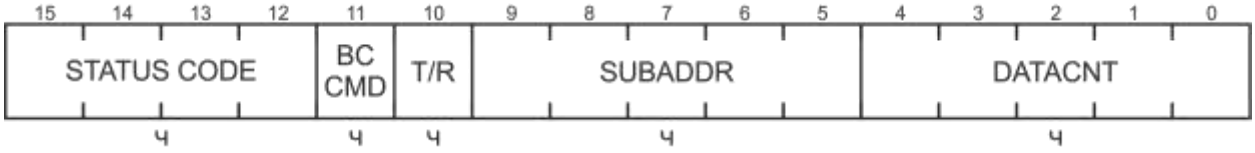
BSICON (в режиме МШ)



Поле	Бит	Описание	
INTNUM	15-13	Количество записанных сообщений, после которого выполняется прерывание	
		000	Нет прерываний
		001	После каждого сообщения
		010	Через 15 сообщений
		011	Через 31 сообщение
		100	Через 63 сообщения
		101	Через 127 сообщений
		110	Через 255 сообщений
		111	Через 511 сообщений
ADDR	12-8	Адрес контролируемого ОУ	
SUBADDR	7-3	Контролируемый подадрес. Для отслеживания команд управления SUBADDR = 11111.	
MONMODE	2-1	Режим выборки сообщений	
		00	Все подряд
		01	Сообщения, закончившиеся ошибкой
		10	По подадресу. Сохраняются сообщения с подадресом, указанным в поле SUBADDR
		11	По адресу и подадресу. Сохраняются сообщения с адресом, указанным в поле ADDR и подадресом, указанным в поле SUBADDR. Если SUBADDR = 00000, то выборка осуществляется только по адресу.
START	0	Бит включения МШ. Пока этот бит установлен, не могут быть изменены поля ADDR, SUBADDR, MONMODE и регистр BSICONFIG	

Таблица А.18.7 – Регистр состояния

BSISTAT (в режимах КШ и ОУ)



Поле	Бит	Описание	
STATUS CODE	15-12	Код статуса окончания сообщения	
		0000	Безошибочное окончание последовательности
		0001	Безошибочное окончание одного сообщения
		0010	Генерация в канале 0
		0011	Генерация в канале 1
		0100	Ошибка контроля передачи (принято не то, что передано)
		0101	Неправильный адрес в ответном слове
		0110	Неправильный синхроимпульс в ответном слове
		0111	Ненулевое ответное слово
		1000	Ошибка манчестерского кода в ответном слове
		1001	Отсутствие ответного слова
		1010	Неправильный синхроимпульс в слове данных
		1011	Ошибка манчестерского кода в слове данных
		1100	Отсутствие слова данных или нарушение непрерывности при приеме слова данных
		1101	Ошибка манчестерского кода в командном слове
		1110	Отсутствие 2-го командного слова или нарушение непрерывности при приеме 2-го командного слова в форматах 3 и 8
		1111	Неверное командное слово (недопустимая комбинация бит)
BCCMD	11	Индикатор общего вызова. Бит устанавливается, если последней отправленной командой была команда с групповым адресом	
T/R	10	Бит направления передачи. Состояние этого бита всегда копирует состояние бита T/R отправленного командного слова	
SUBADDR	9-5	Подадрес. Состояние этого поля всегда копирует состояние поля SUBADDR/MODE отправленного командного слова	
DATACNT	4-0	Длина последней команды. Состояние этого поля всегда копирует состояние поля DATACNT/CODE отправленного командного слова	

Таблица А.18.8 – Регистр состояния

BSISTAT (в режиме МШ)

Поле	Бит	Описание
MSGSTARTADDR	15-0	Адрес начала непрочитанного информационного блока

A.19 Регистры блока OCDS

Таблица 19.1 – Регистры блока JTAG

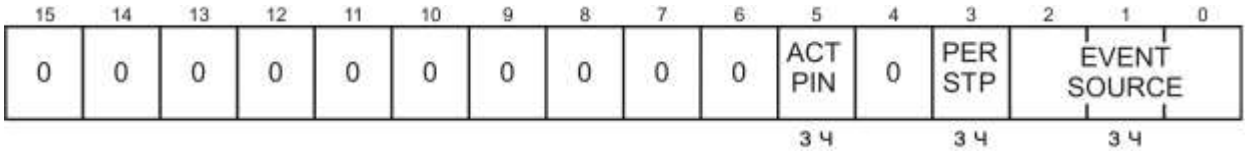
Мнемоника	Разрядность	Назначение	Сброс TRST#
BYPASS	1 бит	Стандартный регистр обхода (bypass) JTAG. Если выбрана команда BYPASS, то сигнал на TDO равен сигналу на TDI, задержанному на один цикл сигнала TCK	Xb
CCONF	16 бит	Регистр конфигурации кристалла	0000h
ID	32 бита	Дополнительный регистр ID стандарта 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 19.2 – Часть регистров блока CERBERUS

Регистр	Разрядность	Описание	Сброс JTAG	Сброс микроконтроллера
IOADDR	24 бита	Адрес для доступа к следующему режиму чтения-записи	000000h	Не изменяется
TRADDR	4 бита	Адрес в режиме трассировки внешней шины	0h	Не изменяется

Таблица А.19.3 – Регистры управления отладочными событиями вывода BREAK и программного обеспечения

DEXEVT / DSWEVT



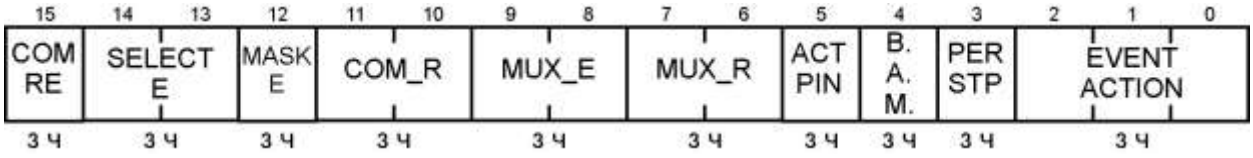
Поле	Биты	Описание	
ACTPIN (ACTIVATE_PIN)	5	Активность внешнего вывода	
		0	Неактивен
		1	Активен в течение отладочного события
PERSTP (PERIPHERALS_ STOP)	3	Бит управления приостановкой периферийных устройств в случае события	
		0	Нет влияния на периферийные устройства
		1	Устройства приостанавливают операции
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

Окончание таблицы А.19.3

Поле	Биты	Описание	
EVENT_SOURCE	2–0	101	Выполнение DPEC
		110	Зарезервировано
		111	Установка бита источника в DBGSR
0	15–6, 4	Зарезервировано	

Таблица А.19.4 – Регистр управления комбинациями аппаратных отладочных событий

DTREVT



Поле	Биты	Описание	
COMRE	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
SELECT_E	14–13	Выбор сравнения на равенство (см. таблицу А.19.5)	
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 А.19.6)	
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
ACTPIN (ACTIVATE_ PIN)	5	Идентично полю ACTIVATE_PIN регистра DEXEVT	
B.A.M. (BREAK_ AFTER_ MAKE)	4	Бит управления остановом	
		0	Остановка до выполнения (только IP)
		1	Остановка после выполнения (только IP)

Окончание таблицы А.19.4

Поле	Биты	Описание
PERSTP (PERIPHERALS_ STOP)	3	Идентично полю PERIPHERALS_STOP регистра DEXEVT
EVENT_ ACTION	2–0	Идентично полю EVENT_SOURCE регистра DEXEVT

Таблица А.19.5 – Формирование сигнала trg_e

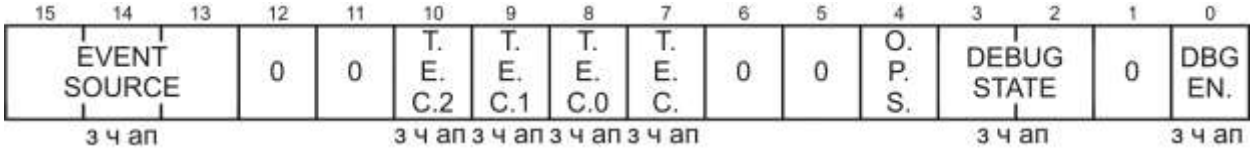
Значение SELECT_E	Сигнал mask_e	Сигнал trg_e
00b	0	0 (не разрешено)
01b		1, если DCMP0 равен, иначе 0
10b		1, если DCMP0 или DCMP1 равны, иначе 0
11b		1, если DCMP0 или DCMP1 или DCMP2 равны, иначе 0
00b	1	0 (не разрешено)
01b		0 (всегда)
10b		1, если DCMP1 равен, иначе 0
11b		1, если DCMP1 или DCMP2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

Таблица А.19.6 – 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

Таблица А.19.7 – Регистр состояния отладки

DBGSR



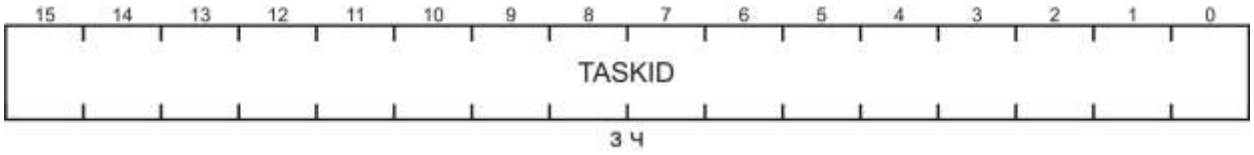
Поле	Биты	Описание	
EVENT_SOURCE	15–13	Источник сообщения о последнем отладочном событии	
		xx1	Внешний вывод приостановки DEXEVT
		x1x	Выполняется команда DEBUG (DSEWT)
		1xx	Комбинация аппаратных источников событий DTREVT
T.E.C.2 (TRGEVT_E_CMP2)	10	Второй флаг сравнения	
		0	Сравнение 2 на равенство не соответствует
		1	Сравнение соответствует для текущего события
T.E.C.1 (TRGEVT_E_CMP1)	9	Первый флаг сравнения	
		0	Сравнение 1 на равенство не соответствует
		1	Сравнение соответствует для текущего события
T.E.C.0 (TRGEVT_E_CMP0)	8	Нулевой флаг сравнения	
		0	Сравнение 0 на равенство не соответствует
		1	Сравнение соответствует для текущего события

Окончание таблицы А.19.7

Поле	Биты	Описание	
T.E.C. (TRGEVT_E_CMP)	7	Флаг сравнения	
		0	Диапазон сравнения не соответствует
		1	Сравнение соответствует для текущего события
O.P.S. (OCDS_P_SUSPEND)	4	Флаг приостановки	
		0	Нет действий
		1	Соответствующая периферия приостанавливает свои операции
DEBUG_STATE	3, 2	Текущее состояние отладки	
		00	Пользовательский режим
		01	Программный отладочный режим
		10	Остановочный отладочный режим (Halt)
		11	Зарезервировано
DBGEN. (DEBUG_ENABLED)	0	Бит разрешения отладки	
		0	Запрещено
		1	Разрешено
0	12, 11, 6, 5, 1	Зарезервировано	

Таблица А.19.8 – Регистр ID задачи

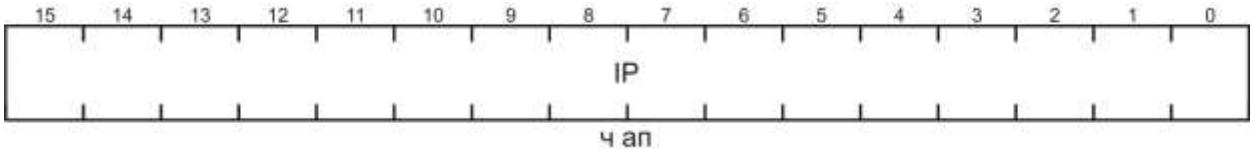
DTIDR



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

Таблица А.19.9 – Регистр указатель команд

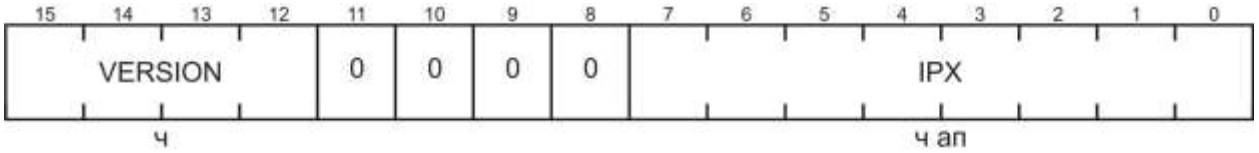
DIP



Поле	Биты	Описание
IP	15–0	Биты 15–0 текущего указателя команд в режиме Halt. Значение верно только в режиме Halt

Таблица А.19.10 – Регистр расширения указателя команд

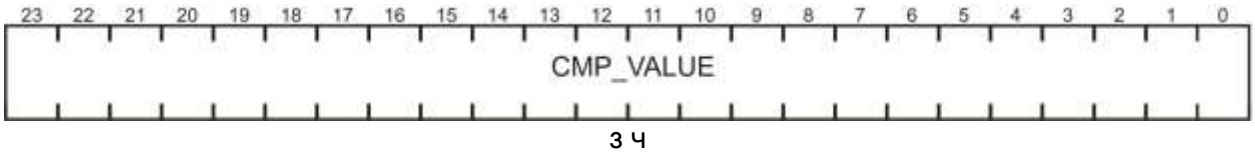
DIPX



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

Таблица А.19.11 – Регистры сравнения аппаратных событий

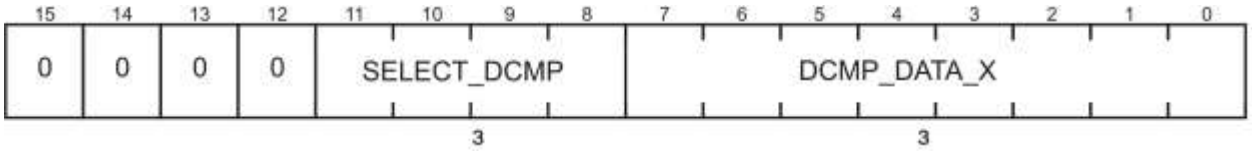
DCMP0, DCMP1, DCMP2, DCMPG, DCMPL



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
Примечание – Регистры DCMP0, DCMP1, DCMP2, DCMPG, DCMPL доступны с помощью регистров DCMPSP и DCMPSD.		

Таблица А.19.12 – Регистр выбора и программирования DCMPx

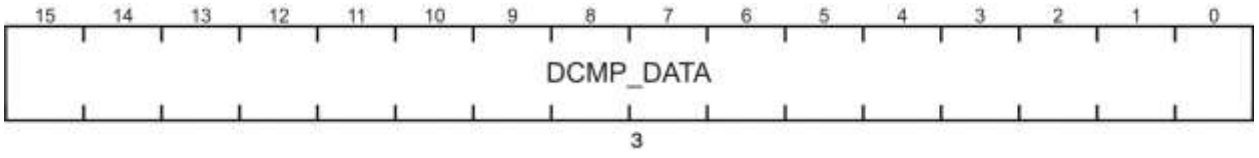
DCMPSP



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 А.19.13 – Регистр программирования данных для DCMPx

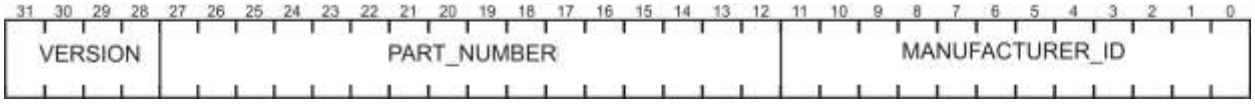
DCMPDP



Поле	Биты	Описание
DCMP_DATA	15–0	Устанавливает биты 15–0 выбранного SELECT_DCMP

Таблица А.19.14 – Регистр идентификации

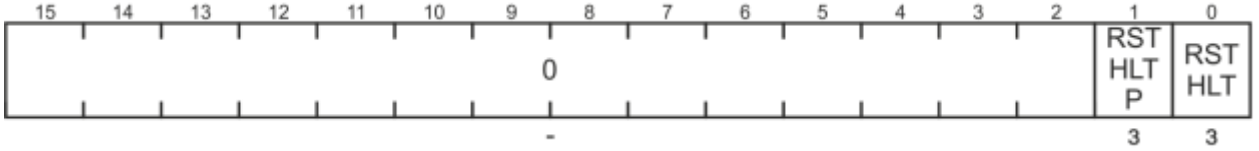
ID (формат команды IDCODE)



Поле	Биты	Описание
VERSION	31–28	Версия кристалла
PART_NUMBER	27–12	Номер кристалла
MANUFACTURER_ID	11–0	Производственный номер
Дополнительный регистр ID стандарта JTAG. Содержание ID выдается наружу, когда поле INSTRUCTION содержит команду IDCODE		

Таблица А.19.15 – Регистр конфигурирования

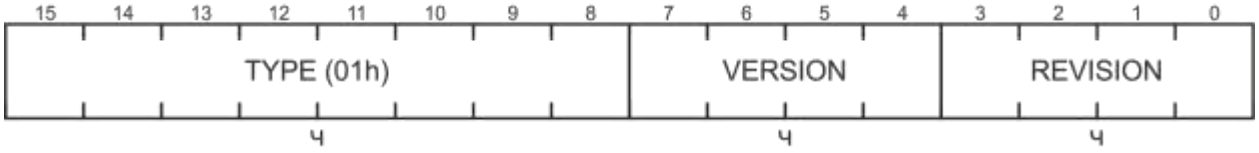
CCONF



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	

Таблица А.19.16 – Регистр идентификации типа клиента

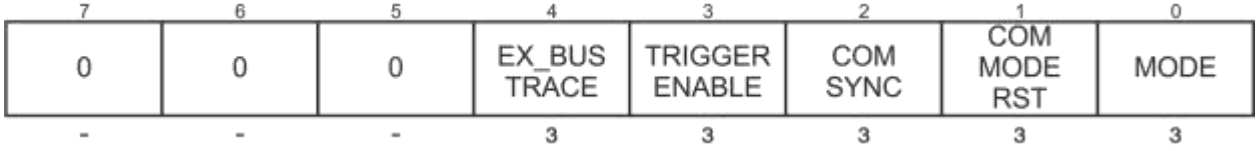
CLIENT_ID (Состояние после сброса аппаратно закодировано)



Поле	Биты	Описание
TYPE	15–8	Тип. Зарезервированное значение – 01h
VERSION	7–4	Версия. Зарезервированное значение – 2h
REVISION	3–0	Ревизия. Зарезервированное значение – 1h

Таблица А.19.17 – Регистр конфигурации

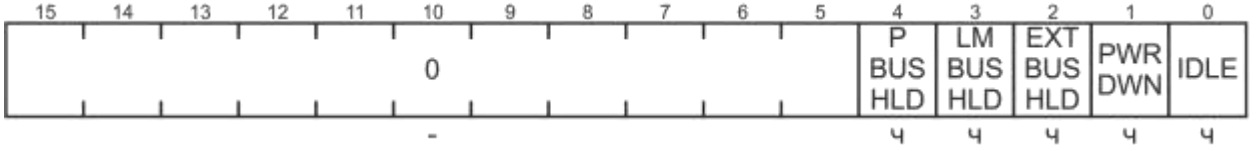
IOCONF (Состояние после сброса JTAG 00h, сброса МК – аппаратно закодировано)



Поле	Биты	Описание
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

Таблица А.19.18 – Регистр состояния для анализа ошибок

IOINFO (Состояние после сброса – спецификация кристалла)



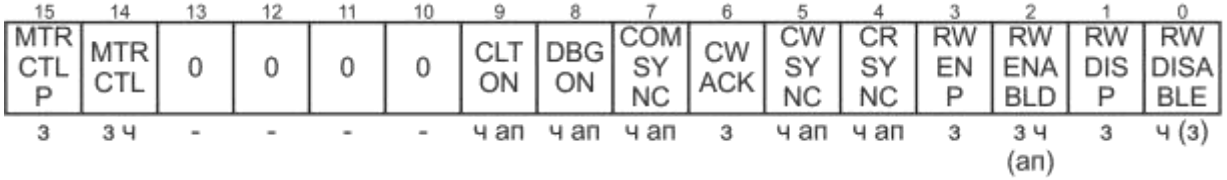
Поле	Биты	Описание
P_BUS_HLD	4	Шина PBus занята
LM_BUS_HLD	3	Не используется

Окончание таблицы А.19.18

Поле	Биты	Описание
EXT_BUS_HLD	2	Внешняя/XBus-шина занята
PWR_DWN	1	Контроллер в режиме PowerDown
IDLE	0	Контроллер в режиме Idle
–	15–5	Зарезервировано

Таблица А.19.19 – Регистр управления и состояния

IOSR (состояние после сброса – см. таблицу А.19.20)



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MTR CTL P	MTR CTL	0	0	0	0	CLT ON	DBG ON	COM SY NC	CW ACK	CW SY NC	CR SY NC	RW EN P	RW ENA BLD	RW DIS P	RW DISA BLE
з	з ч	-	-	-	-	ч ап	ч ап	ч ап	з	ч ап	ч ап	з	з ч (ап)	з	ч (з)

Поле	Биты	Описание	
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
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

Окончание таблицы А.19.19

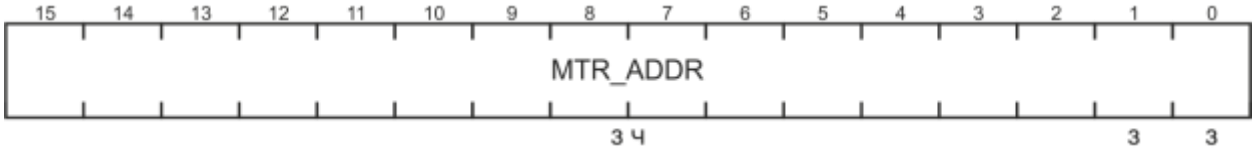
Поле	Биты	Описание	
RW_DISABLE	0	Бит запрета режима чтения-записи	
		0	Режим разрешен
		1	Режим запрещен
–	13–10	Зарезервировано	

Таблица А.19.20 – 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.	

Таблица А.19.21 – Регистры данных режима коммуникации

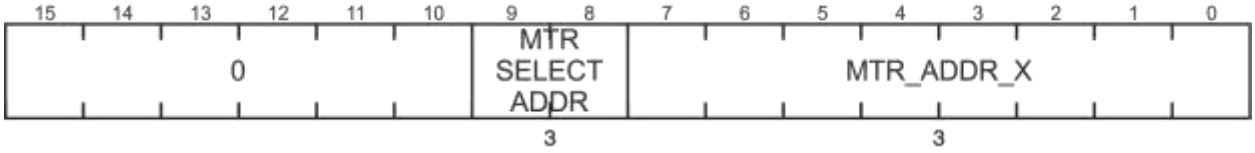
COMDATA (Состояние после сброса JTAG не изменяется, сброса МК – 0000h)



Поле	Биты	Описание
MTR_ADDR	15–0	Данные

Таблица А.19.22 – Регистры данных режима чтения-записи

RWDATA (Состояние после сброса JTAG не изменяется, сброса МК – 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	

Приложение Б (обязательное) Карта памяти регистров микроконтроллера

Карта областей памяти показана в таблице Б.1. Карта регистров областей представлена в таблицах Б.1 – Б.8. Карта регистров 13 сегмента (адреса с 0D’0000h по 0D’3FFCh) представлена в таблицах Б.9 – Б.12.

Таблица Б.1 – Карта областей памяти

Диапазон адресов		Область памяти
00’F000h-00’F0FEh	00h-7Fh	ESFR
00’F100h-00’F1E8h	80h-F4h	ESFR-b (побитно адресуемая)
00’F1EAh-00’F1FEh	F5h-FFh	ESFR
00’F200h-00’FDFEh	–	DPRAM
00’FE00h-00’FEFEh	00h-7Fh	SFR
00’FF00h-00’FFFEh	80h-FFh	SFR-b (побитно адресуемая)
00’E800h-00’EFFEh	–	XSFR

Таблица Б.2 – Область ESFR

Адрес		Мнемоника	Описание	Сброс
F000h	00h	QX0	Регистр смещения 0 для указателя адреса IDX0/1	0000h
F002h	01h	QX1	Регистр смещения 1 для указателя адреса IDX0/1	0000h
F004h	02h	QR0	Регистр смещения 0 для указателя адреса GPRi	0000h
F006h	03h	QR1	Регистр смещения 1 для указателя адреса GPRi	0000h
F008h, F00Ah	04h, 05h	–	Зарезервировано	–
F00Ch	06h	CPUID	Регистр идентификации ЦПУ	0420h
F00Eh–F012h	07h–09h	–	Зарезервировано	–
F014h	0Ah	XADRS1	Регистр 1 выбора адреса шины XBUS	XXXXh
F016h	0Bh	XADRS2	Регистр 2 выбора адреса шины XBUS	XXXXh
F018h	0Ch	XADRS3	Регистр 3 выбора адреса шины XBUS	XXXXh
F01Ah	0Dh	XADRS4	Регистр 4 выбора адреса шины XBUS	XXXXh
F01Ch	0Eh	XADRS5	Регистр 5 выбора адреса шины XBUS	XXXXh
F01Eh	0Fh	XADRS6	Регистр 6 выбора адреса шины XBUS	XXXXh
F020h, F022h	10h, 11h	–	Зарезервировано	–
F024h	12h	XPERCON	Регистр управления X-периферией	0000h
F026h–F02Eh	13h–17h	–	Зарезервировано	–
F030h	18h	RTCCST	Регистр управления и статуса	--00h
F032h	19h	RTUDST	Регистр состояния обновлений	--00h
F034h	1Ah	RTCEIST	Регистр состояния прерываний	--00h
F036h	1Bh	RTCIEN	Регистр разрешения прерываний	--00h
F038h	1Ch	RTPRD	Регистр прескалера	0000h
F03Ah	1Dh	RTCRD_H	Регистр реального времени (старшие разряды)	0000h
F03Ch	1Eh	RTCRD_L	Регистр реального времени (младшие разряды)	0001h

Продолжение таблицы Б.2

Адрес		Мнемоника	Описание	Сброс
F03Eh	1Fh	RTCLD_H	Перезагружаемый регистр (старшие разряды)	0000h
F040h	20h	RTCLD_L	Перезагружаемый регистр (младшие разряды)	0000h
F042h	21h	RTCCMP1_H	Регистр сравнения 1 (старшие разряды)	0000h
F044h	22h	RTCCMP1_L	Регистр сравнения 1 (младшие разряды)	0200h
F046h	23h	RTCCMP2_H	Регистр сравнения 2 (старшие разряды)	FFFFh
F048h	24h	RTCCMP2_L	Регистр сравнения 2 (младшие разряды)	FFFFh
F04Ah	25h	RTCCMP3_H	Регистр сравнения 3 (старшие разряды)	FFFFh
F04Ch	26h	RTCCMP3_L	Регистр сравнения 3 (младшие разряды)	FFFFh
F04Eh	27h	–	Зарезервировано	–
F050h	28h	T7	Регистр таймера T7	0000h
F052h	29h	T8	Регистр таймера T8	0000h
F054h	2Ah	T7REL	Регистр загрузки таймера T7	0000h
F056h	2Bh	T8REL	Регистр загрузки таймера T8	0000h
F058h	2Ch	–	Зарезервировано	–
F05Ah	2Dh	SSC1TB	Буферный регистр передатчика	0000h
F05Ch	2Eh	SSC1RB	Буферный регистр приемника	0000h
F05Eh	2Fh	SSC1BR	Регистр скорости пересылки	0000h
F060h	30h	–	Зарезервировано	–
F062h	31h	CC1_IOC	Регистр управления вводом-выводом	0000h
F064h	32h	–	Зарезервировано	–
F066h	33h	CC2_IOC	Регистр управления вводом-выводом	0000h
F068h	34h	COMDATA	Регистр режима коммуникации Cerberus	0000h
F06Ah	35h	RWDATA	Регистр данных режима RW Cerberus	0000h
F06Ch	36h	IOSR	Регистр статуса Cerberus	0000h
F06Eh–F0AEh	37h–57h	–	Зарезервировано	–
F0B0h	58h	SSC0TB	Буферный регистр передатчика	0000h
F0B2h	59h	SSC0RB	Буферный регистр приемника	0000h
F0B4h	5Ah	SSC0BR	Регистр скорости пересылки	0000h
F0B6h	5Bh	–	Зарезервировано	–
F0B8h	5Ch	SSC2RIC	Регистр контроля прерывания по приему SPI	0000h
F0BAh	5Dh	SSC2TIC	Регистр контроля прерывания по передаче SPI	0000h
F0BCh	5Eh	S2EIC	Регистр управления прерыванием по ошибке USART2	0000h
F0BEh	5Fh	S2RIC	Регистр управления прерыванием по приему USART2	0000h
F0C0h	60h	S2TBIC	Регистр контроля прерывания по опустошению буфера передачи USART2	0000h
F0C2h	61h	S2TIC	Регистр управления прерыванием по передаче USART2	0000h
F0C4h	62h	BSI0_IC	Регистр 0 прерываний магистрального интерфейса	0000h
F0C6h	63h	BSI1_IC	Регистр 1 прерываний магистрального интерфейса	0000h
F0C8h–F0D6h	64h–6Bh	–	Зарезервировано	–
F0D8h	6Ch	DTIDR	Регистр идентификации задачи системы отладки	0000h

Окончание таблицы Б.2

Адрес		Мнемоника	Описание	Сброс
F0DAh	6Dh	SMBSDA	Сдвиговый регистр данных	--XXh
F0DCh	6Eh	SMBST	Регистр статуса	--00h
F0Deh	6Fh	SMBCST	Регистр управления и статуса	--00h
F0E0h	70h	SMBCTL1	Регистр управления 1	--00h
F0E2h	71h	SMBADDR	Регистр собственного адреса	--00h
F0E4h	72h	SMBCTL2	Регистр управления 2	--00h
F0E6h	73h	SMBTOPR	Регистр делителя частоты времени ожидания	--00h
F0E8h	74h	SMBCTL3	Регистр управления 3	--00h
F0Eah	75h	–	Зарезервировано	–
F0Ech	76h	DCMPSP	Регистр выбора и программирования для DCMPx	0000h
F0Eeh	77h	DCMPDP	Регистр данных программирования для DCMPx	0000h
F0F0h	78h	DTREVT	Регистр управления комбинациями аппаратных отладочных событий	0000h
F0F2h	79h	DEXEVT	Регистр управления отладочными событиями вывода BREAK и программного обеспечения	0000h
F0F4h	7Ah	DSWEVT	Регистр управления отладочными событиями вывода BREAK и программного обеспечения	0000h
F0F6h	7Bh	–	Зарезервировано	–
F0F8h	7Ch	DIP	Регистр указателя машинной команды	0000h
F0Fah	7Dh	DIPX	Регистр расширенного указателя инструкций	3000h
F0FCh	7Eh	DBGSR	Регистр состояния отладки	0000h
F0Feh	7Fh	–	Зарезервировано	–

Таблица Б.3 – Область ESFR-b

Адрес		Мнемоника	Описание	Сброс
F100h	80h	DP0L	Регистр выбора направления порта P0L	0000h
F102h	81h	DP0H	Регистр выбора направления порта P0H	0000h
F104h	82h	DP1L	Регистр выбора направления порта P1L	0000h
F106h	83h	DP1H	Регистр выбора направления порта P1H	0000h
F108h	84h	RP0H	Регистр начальной конфигурации внешней шины	--XXh
F10Ah-F112h	85h–89h	–	Зарезервировано	–
F114h	8Ah	XBCON1	Регистр 1 управления шины XBUS	XXXXh
F116h	8Bh	XBCON2	Регистр 2 управления шины XBUS	XXXXh
F118h	8Ch	XBCON3	Регистр 3 управления шины XBUS	XXXXh
F11Ah	8Dh	XBCON4	Регистр 4 управления шины XBUS	XXXXh
F11Ch	8Eh	XBCON5	Регистр 5 управления шины XBUS	XXXXh
F11Eh	8Fh	XBCON6	Регистр 6 управления шины XBUS	XXXXh
F120h	90h	CCU6_IC	Регистр управления прерываниями	--00h
F122h	91h	SSC1TIC	Регистр контроля прерывания по передаче	0000h
F124h	92h	SSC1RIC	Регистр контроля прерывания по приему	0000h
F126h	93h	SSC1EIC	Регистр контроля прерывания по ошибке	0000h
F128h	94h	CAN_IC0	Регистр контроля прерываний линии 0	0000h
F12Ah	95h	CAN_IC1	Регистр контроля прерываний линии 1	0000h
F12Ch	96h	CAN_IC2	Регистр контроля прерываний линии 2	0000h
F12Eh	97h	CAN_IC3	Регистр контроля прерываний линии 3	0000h
F130h	98h	–	–	–

Продолжение таблицы Б.3

Адрес		Мнемоника	Описание	Сброс
F132h	99h	CAN_IC4	Регистр контроля прерываний линии 4	0000h
F134h	9Ah	CAN_IC5	Регистр контроля прерываний линии 5	0000h
F136h	9Bh	CAN_IC6	Регистр контроля прерываний линии 6	0000h
F138h	9Ch	CAN_IC7	Регистр контроля прерываний линии 7	0000h
F13Ah	9Dh	RTC_IC	Регистр контроля прерываний RTC	0000h
F13Ch	9Eh	S1TBIC	Регистр управления прерыванием по опустошению буфера передатчика ASC1	0000h
F13Eh	9Fh	–	–	–
F140h	A0h	CAN_IC8	Регистр контроля прерываний линии 8	0000h
F142h	A1h	CAN_IC9	Регистр контроля прерываний линии 9	0000h
F144h	A2h	CAN_IC10	Регистр контроля прерываний линии 10	0000h
F146h	A3h	CAN_IC11	Регистр контроля прерываний линии 11	0000h
F148h	A4h	CAN_IC12	Регистр контроля прерываний линии 12	0000h
F14Ah	A5h	CAN_IC13	Регистр контроля прерываний линии 13	0000h
F14Ch	A6h	CAN_IC14	Регистр контроля прерываний линии 14	0000h
F14Eh	A7h	CAN_IC15	Регистр контроля прерываний линии 15	0000h
F150h	A8h	RTC_IC1	Регистр контроля прерываний 1	0000h
F152h	A9h	RTC_IC2	Регистр контроля прерываний 2	0000h
F154h	AAh	RTC_IC3	Регистр контроля прерываний 3	0000h
F156h–F15Ch	ABh–AEh	–	Зарезервировано	–
F15Eh	AFh	SSC2EIC	Регистр контроля прерывания по ошибке SPI	0000h
F160h	B0h	CC16IC	Регистр управления прерываниями 16	0000h
F162h	B1h	CC17IC	Регистр управления прерываниями 17	0000h
F164h	B2h	CC18IC	Регистр управления прерываниями 18	0000h
F166h	B3h	CC19IC	Регистр управления прерываниями 19	0000h
F168h	B4h	CC20IC	Регистр управления прерываниями 20	0000h
F16Ah	B5h	CC21IC	Регистр управления прерываниями 21	0000h
F16Ch	B6h	CC22IC	Регистр управления прерываниями 22	0000h
F16Eh	B7h	CC23IC	Регистр управления прерываниями 23	0000h
F170h	B8h	CC24IC	Регистр управления прерываниями 24	0000h
F172h	B9h	CC25IC	Регистр управления прерываниями 25	0000h
F174h	BAh	CC26IC	Регистр управления прерываниями 26	0000h
F176h	BBh	CC27IC	Регистр управления прерываниями 27	0000h
F178h	BCh	CC28IC	Регистр управления прерываниями 28	0000h
F17Ah	BDh	T7IC	Регистр управления прерываниями таймера T7	--00h
F17Ch	BEh	T8IC	Регистр управления прерываниями таймера T8	--00h
F17Eh	BFh	–	Зарезервировано	–
F180h	C0h	EOPIC	Регистр контроля за прерыванием по завершению PEC передачи	0000h
F182h	C1h	S1TIC	Регистр управления прерыванием по передаче ASC1	0000h
F184h	C2h	CC29IC	Регистр управления прерываниями 29	0000h
F186h	C3h	I2C_DIC	Регистр управления прерыванием	--00h
F188h	C4h	CCU6_EIC	Регистр управления прерываниями по ошибке	--00h
F18Ah	C5h	S1RIC	Регистр управления прерыванием по приему ASC1	0000h
F18Ch	C6h	CC30IC	Регистр управления прерываниями 30	0000h
F18Eh	C7h	–	Зарезервировано	–

Окончание таблицы Б.3

Адрес		Мнемоника	Описание	Сброс
F190h	C8h	CCU6_T12IC	Регистр управления прерываниями таймера T12	--00h
F192h	C9h	S1EIC	Регистр управления прерыванием по ошибке ASC1	0000h
F194h	CAh	CC31IC	Регистр управления прерываниями 31	0000h
F196h	CBh	–	Зарезервировано	–
F198h	CCh	CCU6_T13IC	Регистр управления прерываниями таймера T13	--00h
F19Ah	CDh	WDTIC	Регистр управления прерыванием сторожевого таймера	0000h
F19Ch	CEh	S0TBIC	Регистр контроля прерывания по опустошению буфера передачи ASC0	0000h
F19Eh–F1B6h	D0h–DBh	–	Зарезервировано	–
F1B8h	DCh	S1CON	Регистр управления ASC1	0000h
F1BAh	DDh	S2CON	Регистр управления USART2	0000h
F1BCh–F1BEh	DEh–DFh	–	Зарезервировано	–
F1C0h	E0h	EXICON	Регистр контроля внешних прерываний	0000h
F1C2h	E1h	ODP2	Регистр управления режимом с открытым стоком порта P2	0000h
F1C4h	–	–	Зарезервировано	
F1C6h	E3h	ODP3	Регистр управления режимом с открытым стоком порта P3	0000h
F1C8h	E4h	–	Зарезервировано	–
F1CAh	E5h	ODP4	Регистр управления режимом с открытым стоком порта P4	0000h
F1CCh	–	–	Зарезервировано	–
F1CEh	E7h	ODP6	Регистр управления режимом с открытым стоком порта P6	0000h
F1D0h	E8h	PLLCON	Регистр управления PLL	
F1D2h	E9h	ODP7	Регистр управления режимом с открытым стоком порта P7	0000h
F1D4h	EAh	–		–
F1D6h	EBh	ODP9	Регистр управления режимом с открытым стоком порта P9	0000h
F1D8h	ECh	EXISEL1	Регистр 1 выбора источника внешних прерываний	0000h
F1DAh	EDh	EXISEL0	Регистр 0 выбора источника внешних прерываний	0000h
F1DCh, F1DEh	EEh, EFh	–	Зарезервировано	–
F1E0h	F0h	S2TBUF	Буферный регистр передатчика USART2	0000h
F1E2h	F1h	S2RBUF	Буферный регистр приемника USART2	0000h
F1E4h	F2h	S2BG	Регистр скорости пересылки USART2	0000h
F1E6h	F3h	S2FDV	Регистр дробного делителя USART2	0000h
F1E8h	F4h	–	Зарезервировано	–

Таблица Б.4 – Область ESFR

Адрес		Мнемоника	Описание	Сброс
F1EAh	F5h	ALTSEL0P1L	Регистр управления альтернативными функциями порта P1L	0000h
F1ECh	F6h	ALTSEL0P1H	Регистр управления альтернативными функциями порта P1H	0000h
F1EEh	F7h	ALTSEL0P2	Регистр управления альтернативными функциями порта P2	0000h
F1F0h	F8h	ALTSEL0P3	Регистр 0 управления альтернативными функциями порта P3	0000h
F1F2h	F9h	ALTSEL1P3	Регистр 1 управления альтернативными функциями порта P3	0000h
F1F4h	FAh	ALTSEL0P4	Регистр управления альтернативными функциями порта P4	0000h
F1F6h	FBh	ALTSEL0P6	Регистр управления альтернативными функциями порта P6	0000h
F1F8h	FCh	ALTSEL0P7	Регистр 0 управления альтернативными функциями порта P7	0000h
F1FAh	FDh	ALTSEL1P7	Регистр 1 управления альтернативными функциями порта P7	0000h
F1FCh	FEh	ALTSEL0P9	Регистр 0 управления альтернативными функциями порта P9	0000h
F1FEh	FFh	ALTSEL1P9	Регистр 1 управления альтернативными функциями порта P9	0000h

Таблица Б.5 – Область DPRAM

Адрес	Мнемоника	Описание	Сброс
FCE0h	SRCP0	Регистр адреса источника канала 0	0000h
FCE2h	DSTP0	Регистр адреса назначения канала 0	0000h
FCE4h	SRCP1	Регистр адреса источника канала 1	0000h
FCE6h	DSTP1	Регистр адреса назначения канала1	0000h
FCE8h	SRCP2	Регистр адреса источника канала 2	0000h
FCEAh	DSTP2	Регистр адреса назначения канала 2	0000h
FCECh	SRCP3	Регистр адреса источника канала 3	0000h
FCEEh	DSTP3	Регистр адреса назначения канала 3	0000h
FCF0h	SRCP4	Регистр адреса источника канала 4	0000h
FCF2h	DSTP4	Регистр адреса назначения канала 4	0000h
FCF4h	SRCP5	Регистр адреса источника канала 5	0000h
FCF6h	DSTP5	Регистр адреса назначения канала 5	0000h
FCF8h	SRCP6	Регистр адреса источника канала 6	0000h
FCFAh	DSTP6	Регистр адреса назначения канала 6	0000h
FCFCh	SRCP7	Регистр адреса источника канала 7	0000h
FCFEh	DSTP7	Регистр адреса назначения канала 7	0000h

Таблица Б.6 – Область SFR

Адрес		Мнемоника	Описание	Сброс
FE00h	00h	DPP0	Регистр указателя нулевой страницы данных ЦПУ	0000h
FE02h	01h	DPP1	Регистр указателя первой страницы данных ЦПУ	0001h
FE04h	02h	DPP2	Регистр указателя второй страницы данных ЦПУ	0002h
FE06h	03h	DPP3	Регистр указателя третьей страницы данных ЦПУ	0003h
FE08h	04h	CSP	Регистр указателя сегмента кода ЦПУ	0000h
FE0Ah	05h	–	Зарезервировано	–
FE0Ch	06h	MDH	Старший регистр умножения/деления ЦПУ	0000h
FE0Eh	07h	MDL	Младший регистр умножения/деления ЦПУ	0000h
FE10h	08h	CP	Регистр контекстного указателя ЦПУ	FC00h
FE12h	09h	SP	Регистр указателя стека ЦПУ	FC00h
FE14h	0Ah	STKOV	Регистр указателя переполнения стека ЦПУ	FA00h
FE16h	0Bh	STKUN	Регистр опустошения стека ЦПУ	FC00h
FE18h	0Ch	ADDRSEL1	Регистр 1 выбора адреса	0000h
FE1Ah	0Dh	ADDRSEL2	Регистр 2 выбора адреса	0000h
FE1Ch	0Eh	ADDRSEL3	Регистр 3 выбора адреса	0000h
FE1Eh	0Fh	ADDRSEL4	Регистр 4 выбора адреса	0000h
FE20h–FE26h	10h–13h	–	Зарезервировано	–
FE28h	14h	CC2_SEM	Регистр режима однократного срабатывания	0000h
FE2Ah	15h	CC2_SEE	Регистр однократного срабатывания	0000h
FE2Ch	16h	CC1_SEM	Регистр режима однократного срабатывания	0000h
FE2Eh	17h	CC1_SEE	Регистр однократного срабатывания	0000h
FE30h–FE3Eh	18h–1Fh	–	Зарезервировано	–
FE40h	20h	T2	Регистр таймера T2	0000h
FE42h	21h	T3	Регистр таймера T3	0000h
FE44h	22h	T4	Регистр таймера T4	0000h
FE46h	23h	T5	Регистр таймера T5	0000h
FE48h	24h	T6	Регистр таймера T6	0000h
FE4Ah	25h	CAPREL	Регистр захвата/перезагрузки	0000h
FE4Ch, FE4Eh	26h, 27h	–	Зарезервировано	–
FE50h	28h	T0	Регистр таймера T0	0000h
FE52h	29h	T1	Регистр таймера T1	0000h
FE54h	2Ah	T0REL	Регистр загрузки таймера T0	0000h
FE56h	2Bh	T1REL	Регистр загрузки таймера T1	0000h
FE58h, FE5Ah	2Ch, 2Dh	–	Зарезервировано	–
FE5Ch	2Eh	MAL	Регистр младшего байта аккумулятора	0000h
FE5Eh	2Fh	MAH	Регистр старшего байта аккумулятора	0000h
FE60h	30h	CC16	Регистр захвата/сравнения 16	0000h
FE62h	31h	CC17	Регистр захвата/сравнения 17	0000h
FE64h	32h	CC18	Регистр захвата/сравнения 18	0000h
FE66h	33h	CC19	Регистр захвата/сравнения 19	0000h
FE68h	34h	CC20	Регистр захвата/сравнения 20	0000h
FE6Ah	35h	CC21	Регистр захвата/сравнения 21	0000h
FE6Ch	36h	CC22	Регистр захвата/сравнения 22	0000h
FE6Eh	37h	CC23	Регистр захвата/сравнения 23	0000h
FE70h	38h	CC24	Регистр захвата/сравнения 24	0000h

Продолжение таблицы Б.6

Адрес		Мнемоника	Описание	Сброс
FE72h	39h	CC25	Регистр захвата/сравнения 25	0000h
FE74h	3Ah	CC26	Регистр захвата/сравнения 26	0000h
FE76h	3Bh	CC27	Регистр захвата/сравнения 27	0000h
FE78h	3Ch	CC28	Регистр захвата/сравнения 28	0000h
FE7Ah	3Dh	CC29	Регистр захвата/сравнения 29	0000h
FE7Ch	3Eh	CC30	Регистр захвата/сравнения 30	0000h
FE7Eh	3Fh	CC31	Регистр захвата/сравнения 31	0000h
FE80h	40h	CC0	Регистр захвата/сравнения 0	0000h
FE82h	41h	CC1	Регистр захвата/сравнения 1	0000h
FE84h	42h	CC2	Регистр захвата/сравнения 2	0000h
FE86h	43h	CC3	Регистр захвата/сравнения 3	0000h
FE88h	44h	CC4	Регистр захвата/сравнения 4	0000h
FE8Ah	45h	CC5	Регистр захвата/сравнения 5	0000h
FE8Ch	46h	CC6	Регистр захвата/сравнения 6	0000h
FE8Eh	47h	CC7	Регистр захвата/сравнения 7	0000h
FE90h	48h	CC8	Регистр захвата/сравнения 8	0000h
FE92h	49h	CC9	Регистр захвата/сравнения 9	0000h
FE94h	4Ah	CC10	Регистр захвата/сравнения 10	0000h
FE96h	4Bh	CC11	Регистр захвата/сравнения 11	0000h
FE98h	4Ch	CC12	Регистр захвата/сравнения 12	0000h
FE9Ah	4Dh	CC13	Регистр захвата/сравнения 13	0000h
FE9Ch	4Eh	CC14	Регистр захвата/сравнения 14	0000h
FE9Eh	4Fh	CC15	Регистр захвата/сравнения 15	0000h
FEA0h	50h	SSC2TB	Буферный регистр передатчика SPI	0000h
FEA2h	51h	SSC2RB	Буферный регистр приемника SPI	0000h
FEA4h	52h	SSC2BR	Регистр скорости пересылки SPI	0000h
FEA6h	53h	S1TBUF	Буферный регистр передатчика ASC1	0000h
FEA8h	54h	S1RBUF	Буферный регистр приемника ASC1	0000h
FEAAh	55h	S1BG	Регистр скорости пересылки ASC1	0000h
FEACh	56h	S1FDV	Регистр дробного делителя ASC1	0000h
FEAEh	57h	WDTCON	Регистр сторожевого таймера	00000000 10000xx1b
FEB0h	58h	S0TBUF	Буферный регистр передатчика ASC0	0000h
FEB2h	59h	S0RBUF	Буферный регистр приемника ASC0	0000h
FEB4h	5Ah	S0BG	Регистр скорости пересылки ASC0	0000h
FEB6h	5Bh	S0FDV	Регистр дробного делителя ASC0	0000h
FEB8h- FEBEh	5Ch- 5Fh	–	Зарезервировано	–
FEC0h	60h	PECC0	Регистр управления PEC-каналом 0	0000h
FEC2h	61h	PECC1	Регистр управления PEC-каналом 1	0000h
FEC4h	62h	PECC2	Регистр управления PEC-каналом 2	0000h
FEC6h	63h	PECC3	Регистр управления PEC-каналом 3	0000h
FEC8h	64h	PECC4	Регистр управления PEC-каналом 4	0000h
FECAh	65h	PECC5	Регистр управления PEC-каналом 5	0000h
FECCh	66h	PECC6	Регистр управления PEC-каналом 6	0000h
FECEh	67h	PECC7	Регистр управления PEC-каналом 7	0000h
FED0h	68h	PECSN0	Регистр указателя сегмента 0	0000h
FED2h	69h	PECSN1	Регистр указателя сегмента 1	0000h
FED4h	6Ah	PECSN2	Регистр указателя сегмента 2	0000h

Окончание таблицы Б.6

Адрес		Мнемоника	Описание	Сброс
FED6h	6Bh	PECSN3	Регистр указателя сегмента 3	0000h
FED8h	6Ch	PECSN4	Регистр указателя сегмента 4	0000h
FEDAh	6Dh	PECSN5	Регистр указателя сегмента 5	0000h
FEDCh	6Eh	PECSN6	Регистр указателя сегмента 6	0000h
FEDEh	6Fh	PECSN7	Регистр указателя сегмента 7	0000h
FEE0h-FEEEh	70h-77h	–	Зарезервировано	–
FEF0h	78h	PECXC0	Регистр расширенного управления PEC-каналом 0	0000h
FEF2h	79h	PECXC2	Регистр расширенного управления PEC-каналом 2	0000h
FEF4h-FEFEh	7Ch-7Fh	–	Зарезервировано	–

Таблица Б.7 – Область SFR-b

Адрес		Мнемоника	Описание	Сброс
FF00h	80h	P0L	Младший регистр порта P0	00h
FF02h	81h	P0H	Старший регистр порта P0	00h
FF04h	82h	P1L	Младший регистр порта P1	00h
FF06h	83h	P1H	Старший регистр порта P1	00h
FF08h	84h	IDX0	Регистр 0 указателя адреса MAC	0000h
FF0Ah	85h	IDX1	Регистр 1 указателя адреса MAC	0000h
FF0Ch	86h	BUSCON0	Регистр конфигурации шины 0	0000h
FF0Eh	87h	MDC	Регистр управления умножением/делением	0000h
FF10h	88h	PSW	Регистр слова состояния процессора ЦПУ	0000h
FF12h	89h	SYSCON	Регистр конфигурации системы	XXXXh
FF14h	8Ah	BUSCON1	Регистр конфигурации шины 1	0000h
FF16h	8Bh	BUSCON2	Регистр конфигурации шины 2	0000h
FF18h	8Ch	BUSCON3	Регистр конфигурации шины 3	0000h
FF1Ah	8Dh	BUSCON4	Регистр конфигурации шины 4	0000h
FF1Ch	8Eh	ZEROS	Регистр постоянного значения 0	0000h
FF1Eh	8Fh	ONES	Регистр постоянного значения 1	FFFFh
FF20h	90h	T78CON	Регистр управления таймерами T7 и T8	0000h
FF22h	91h	CC2_M4	Регистр 4 управления режимом	0000h
FF24h	92h	CC2_M5	Регистр 5 управления режимом	0000h
FF26h	93h	CC2_M6	Регистр 6 управления режимом	0000h
FF28h	94h	CC2_M7	Регистр 7 управления режимом	0000h
FF2Ah	95h	CC2_DRM	Регистр двухрегистрового режима сравнения	0000h
FF2Ch	96h	CC2_OUT	Регистр выводов сравнения	0000h
FF2Eh–FF3Eh	97h–9Fh	–	Зарезервировано	–
FF40h	A0h	T2CON	Регистр управления таймера T2	0000h
FF42h	A1h	T3CON	Регистр управления таймера T3	0000h
FF44h	A2h	T4CON	Регистр управления таймера T4	0000h
FF46h	A3h	T5CON	Регистр управления таймера T5	0000h
FF48h	A4h	T6CON	Регистр управления таймера T6	0000h
FF4Ah	A5h	ADC_DAT		
FF4Ch–FF4Eh	A5h–A7h	–	Зарезервировано	–

Продолжение таблицы Б.7

Адрес		Мнемоника	Описание	Сброс
FF50h	A8h	T01CON	Регистр управления таймерами T0 и T1	0000h
FF52h	A9h	CC1_M0	Регистр 0 управления режимом	0000h
FF54h	AAh	CC1_M1	Регистр 1 управления режимом	0000h
FF56h	ABh	CC1_M2	Регистр 2 управления режимом	0000h
FF58h	ACh	CC1_M3	Регистр 3 управления режимом	0000h
FF5Ah	ADh	CC1_DRM	Регистр двухрегистрового режима сравнения	0000h
FF5Ch	AEh	CC1_OUT	Регистр выводов сравнения	0000h
FF5Eh	AFh	SSC1CON	Регистр управления	0000h
FF60h	B0h	T2IC	Регистр управления прерываниями таймера T2	0000h
FF62h	B1h	T3IC	Регистр управления прерываниями таймера T3	0000h
FF64h	B2h	T4IC	Регистр управления прерываниями таймера T4	0000h
FF66h	B3h	T5IC	Регистр управления прерываниями таймерами T5	0000h
FF68h	B4h	T6IC	Регистр управления прерываниями таймера T6	0000h
FF6Ah	B5h	CRIC	Регистр управления прерыванием захвата/перезагрузки	0000h
FF6Ch	B6h	S0TIC	Регистр контроля прерывания по передаче ASC0	0000h
FF6Eh	B7h	S0RIC	Регистр контроля прерывания по приему ASC0	0000h
FF70h	B8h	S0EIC	Регистр контроля прерывания по ошибке ASC0	0000h
FF72h	B9h	SSC0TIC	Регистр контроля прерывания по передаче	0000h
FF74h	BAh	SSC0RIC	Регистр контроля прерывания по приему	0000h
FF76h	BBh	SSC0EIC	Регистр контроля прерывания по ошибке	0000h
FF78h	BCh	CC0IC	Регистр управления прерываниями 0	0000h
FF7Ah	BDh	CC1IC	Регистр управления прерываниями 1	0000h
FF7Ch	BEh	CC2IC	Регистр управления прерываниями 2	0000h
FF7Eh	BFh	CC3IC	Регистр управления прерываниями 3	0000h
FF80h	C0h	CC4IC	Регистр управления прерываниями 4	0000h
FF82h	C1h	CC5IC	Регистр управления прерываниями 5	0000h
FF84h	C2h	CC6IC	Регистр управления прерываниями 6	0000h
FF86h	C3h	CC7IC	Регистр управления прерываниями 7	0000h
FF88h	C4h	CC8IC	Регистр управления прерываниями 8	0000h
FF8Ah	C5h	CC9IC	Регистр управления прерываниями 9	0000h
FF8Ch	C6h	CC10IC	Регистр управления прерываниями 10	0000h
FF8Eh	C7h	CC11IC	Регистр управления прерываниями 11	0000h
FF90h	C8h	CC12IC	Регистр управления прерываниями 12	0000h
FF92h	C9h	CC13IC	Регистр управления прерываниями 13	0000h
FF94h	CAh	CC14IC	Регистр управления прерываниями 14	0000h
FF96h	CBh	CC15IC	Регистр управления прерываниями 15	0000h
FF98h	CCh	ADC_CIC	Регистр прерываний АЦП	
FF9Ah	CDh	ADC_EIC	Регистр разрешения прерываний АЦП	
FF9Ch	CEh	T0IC	Регистр управления прерываниями таймера T0	--00h
FF9Eh	CFh	T1IC	Регистр управления прерываниями таймера T1	--00h
FFA0h	D0h	ADC_CON	Регистр управления АЦП	
FFA2h	D1h	P5	Регистр данных порта P5	0000h
FFA4h, FFA6h	D2h, D3h	–	Зарезервировано	–
FFA8h	D4h	PECISNC	Регистр управления узлом прерываний	0000h
FFAAh	D5h	FOCON	Регистр управления выходным тактовым сигналом	0000h
FFACh	D6h	TFR	Регистр флагов ловушек	0000h

Окончание таблицы Б.7

Адрес		Мнемоника	Описание	Сброс
FFAEh	D7h	WDTCON	Регистр управления сторожевым таймером	008Xh
FFB0h	D8h	S0CON	Регистр управления ASC0	0000h
FFB2h	D9h	SSC0CON	Регистр управления	0000h
FFB4h–FFBEh	DAh–DFh	–	Зарезервировано	–
FFC0h	E0h	P2	Регистр данных порта P2	0000h
FFC2h	E1h	DP2	Регистр выбора направления порта P2	0000h
FFC4h	E2h	P3	Регистр данных порта P3	0000h
FFC6h	E3h	DP3	Регистр выбора направления порта P3	0000h
FFC8h	E4h	P4	Регистр данных порта P4	0000h
FFCAh	E5h	DP4	Регистр выбора направления порта P4	0000h
FFCCh	E6h	P6	Регистр данных порта P6	0000h
FFCEh	E7h	DP6	Регистр выбора направления порта P6	0000h
FFD0h	E8h	P7	Регистр данных порта P7	0000h
FFD2h	E9h	DP7	Регистр выбора направления порта P7	0000h
FFD4h	EAh	P9	Регистр данных порта P9	0000h
FFD6h	EBh	DP9	Регистр выбора направления порта P9	0000h
FFD8h	ECh	SSC2CON	Регистр управления SPI	0000h
FFDAh	EDh	MRW	Регистр повторения блока MAC	0000h
FFDCh	EEh	MCW	Регистр управления блока MAC	0000h
FFDEh	EFh	MSW	Статусный регистр блока MAC	0200h
FFE0h	F0h	–	Зарезервировано	–
FFE2h	F1h	ASC0ID	Регистр идентификации ASC01	44XXh
FFE4h	F2h	SSC0ID	Регистр идентификации	45XXh
FFE6h	F3h	GPTID	Регистр идентификации GPT1 и GPT2	58XXh
FFE8h	F4h	BSICONFIG_0	Регистр конфигурации	0005h
FFEAh	F5h	BSICON_0	Регистр управления	0000h
FFECh	F6h	BSISADDR_	Регистр начального адреса	0800h
FFEEh	F7h	BSISPACE_0	Регистр паузы	0004h
FFF0h	F8h	BSISTAT_0	Регистр состояния	0000h
FFF2h	F9h	BSICONFIG_1	Регистр конфигурации	0005h
FFF4h	FAh	BSICON_1	Регистр управления	0000h
FFF6h	FBh	BSISADDR_1	Регистр начального адреса	0800h
FFF8h	FCh	BSISPACE_1	Регистр паузы	0004h
FFFAh	FDh	BSISTAT_1	Регистр состояния	0000h
FFFCh, FFFEh	FEh, FFh	–	Зарезервировано	–

Таблица Б.8 – Область XSFR

Адрес	Мнемоника	Описание	Сброс
E800h–E88Eh	–	Зарезервировано	–
E890h	CCU6_T12	Регистр счета таймера T12	0000h
E892h	CCU6_T12PR	Регистр периода таймера T12	0000h
E894h	CCU6_T12DTC	Регистр контроля задержки «dead-time» таймера T12	0000h
E896h	–	Зарезервировано	–
E898h	CCU6_CC60R	Регистр захвата/сравнения канала 0	0000h
E89Ah	CCU6_CC61R	Регистр захвата/сравнения канала 1	0000h
E89Ch	CCU6_CC62R	Регистр захвата/сравнения канала 2	0000h
E89Eh	–	Зарезервировано	–
E8A0h	CCU6_CC60SR	Теневой регистр канала 0	0000h
E8A2h	CCU6_CC61SR	Теневой регистр канала 1	0000h
E8A4h	CCU6_CC62SR	Теневой регистр канала 2	0000h
E8A6h	CCU6_TCTR4	Регистр управления таймерами 4	0000h
E8A8h	CCU6_CMPSTAT	Регистр состояния	0000h
E8AAh	CCU6_CMPMODIF	Регистр изменения состояния	0000h
E8ACh	CCU6_TCTR0	Регистр управления таймерами 0	0000h
E8AEh	CCU6_TCTR2	Регистр управления таймерами 2	0000h
E8B0h	CCU6_T13	Регистр счета таймера T13	0000h
E8B2h	CCU6_T13PR	Регистр периода таймера T13	0000h
E8B4h	CCU6_CC63R	Регистр захвата/сравнения канала 3	0000h
E8B6h	CCU6_CC63SR	Теневой регистр захвата/сравнения	0000h
E8B8h–E8BEh	–	Зарезервировано	–
E8C0h	CCU6_MODCTR	Регистр управления модуляцией	0000h
E8C2h	CCU6_TRPCTR	Регистр управления ловушками	0000h
E8C4h	CCU6_PSLR	Регистр уровня пассивного состояния	0000h
E8C6h	CCU6_T12MSEL	Регистр выбора режима T12	0000h
E8C8h	–	Зарезервировано	–
E8CAh	CCU6_MCMOUTS	Теневой регистр управления выходами в мультиканальном режиме	0000h
E8CCh	CCU6_MCMOUT	Регистр управления выходами в мультиканальном режиме	0000h
E8CEh	CCU6_MCMCTR	Регистр управления мультиканальным режимом	0000h
E8D0h	CCU6_IS	Регистр состояния прерываний	0000h
E8D2h	CCU6_ISS	Регистр установки состояния прерываний	0000h
E8D4h	CCU6_ISR	Регистр сброса состояния прерываний	0000h
E8D6h	CCU6_INP	Регистр указателя узла прерываний	3940h
E8D8h	CCU6_IEN	Регистр разрешения прерываний	0000h
E8DAh–EFFEh	–	Зарезервировано	–

Таблиц Б.9 – Область XSFR 13 сегмента (регистры контроллера CAN) с адресами 0D’xxxxh

Адрес	Мнемоника	Описание	Сброс
0000h	CLC	Регистр управления частотой	0003h
0002h			0000h
0004h–0006h	–	Зарезервировано	–
0008h	ID	Регистр идентификации	C051h
000Ah			002Bh
000Ch	FDR	Регистр делителя	0000h
000Eh			0000h
0010h–008Eh	–	Зарезервировано	–
0100h	LIST0	Регистр списка 0	7F00h
0102h			007Fh
0104h	LIST1	Регистр свободного списка 1	0000h
0106h			0100h
0108h	LIST2	Регистр свободного списка 2	0000h
010Ah			0100h
010Ch	LIST3	Регистр свободного списка 3	0000h
010Eh			0100h
0110h	LIST4	Регистр свободного списка 4	0000h
0112h			0100h
0114h	LIST5	Регистр свободного списка 5	0000h
0116h			0100h
0118h	LIST6	Регистр свободного списка 6	0000h
011Ah			0100h
011Ch	LIST7	Регистр свободного списка 7	0000h
011Eh			0100h
0120h–013Eh	–	Зарезервировано	–
0140h	MSPND0	Регистр 0 ждущих прерываний	0000h
0142h			0000h
0144h	MSPND1	Регистр 1 ждущих прерываний	0000h
0146h			0000h
0148h	MSPND2	Регистр 2 ждущих прерываний	0000h
014Ah			0000h
014Ch	MSPND3	Регистр 3 ждущих прерываний	0000h
014Eh			0000h
0150h	MSPND4	Регистр 4 ждущих прерываний	0000h
0152h			0000h
0154h	MSPND5	Регистр 5 ждущих прерываний	0000h
0156h			0000h
0158h	MSPND6	Регистр 6 ждущих прерываний	0000h
015Ah			0000h
015Ch	MSPND7	Регистр 7 ждущих прерываний	0000h
015Eh			0000h
0160h–017Eh	–	Зарезервировано	–
0180h	MSID0	Регистр 0 индекса сообщения	0020h
0182h			0000h

Продолжение таблицы Б.9

Адрес	Мнемоника	Описание	Сброс
0184h	MSID1	Регистр 1 индекса сообщения	0020h
0186h			0000h
0188h	MSID2	Регистр 2 индекса сообщения	0020h
018Ah			0000h
018Ch	MSID3	Регистр 3 индекса сообщения	0020h
018Eh			0000h
0190h	MSID4	Регистр 4 индекса сообщения	0020h
0192h			0000h
0194h	MSID5	Регистр 5 индекса сообщения	0020h
0196h			0000h
0198h	MSID6	Регистр 6 индекса сообщения	0020h
019Ah			0000h
019Ch	MSID7	Регистр 7 индекса сообщения	0020h
019Eh			0000h
01A0h–01BEh	–	Зарезервировано	–
01C0h	MSIMASK	Регистр маски индекса сообщения	0000h
01C2h			0000h
01C4h	PANCTR	Регистр панели команд	0301h
01C6h			0000h
01C8h	MCR	Регистр управления	0000h
01CAh			0000h
01CCh	MITR	Регистр прерываний	0000h
01CEh			0000h
01D0h–01FEh	–	Зарезервировано	–
0200h	NCR0	Регистр управления узла 0	0001h
0202h			0000h
0204h	NSR0	Регистр состояния узла 0	0000h
0206h			0000h
0208h	NIPR0	Регистр указателя прерываний узла 0	0000h
020Ah			0000h
020Ch	NPCR0	Регистр управления портом узла 0	0000h
020Eh			0000h
0210h	NBTR0	Регистр синхронизации битов 0	0000h
0212h			0000h
0214h	NECNT0	Регистр счетчика ошибок узла 0	0000h
0216h			0060h
0218h	NFCR0	Регистр счетчика сообщений 0	0000h
021Ah			0000h
021Ch–02FEh	–	Зарезервировано	–
0300h	NCR1	Регистр управления узла 1	0001h
0302h			0000h
0304h	NSR1	Регистр состояния узла 1	0000h
0306h			0000h
0308h	NIPR1	Регистр указателя прерываний узла 1	0000h
030Ah			0000h

Окончание таблицы Б.9

Адрес	Мнемоника	Описание	Сброс
030Ch	NPCR1	Регистр управления портом узла 1	0000h
030Eh			0000h
0310h	NBTR1	Регистр синхронизации битов 1	0000h
0312h			0000h
0314h	NECNT1	Регистр счетчика ошибок узла 1	0000h
0316h			0060h
0318h	NFCR1	Регистр счетчика сообщений 1	0000h
031Ah			0000h
031Ch–03FEh	–	Зарезервировано	–
0400h	NCR2	Регистр управления узла 2	0001h
0402h			0000h
0404h	NSR2	Регистр состояния узла 2	0000h
0406h			0000h
0408h	NIPR2	Регистр указателя прерываний узла 2	0000h
040Ah			0000h
040Ch	NPCR2	Регистр управления портом узла 2	0000h
040Eh			0000h
0410h	NBTR2	Регистр синхронизации битов 2	0000h
0412h			0000h
0414h	NECNT2	Регистр счетчика ошибок узла 2	0000h
0416h			0060h
0418h	NFCR2	Регистр счетчика сообщений 2	0000h
041Ah			0000h
041Ch–04FEh	–	Зарезервировано	–
0500h	NCR3	Регистр управления узла 3	0001h
0502h			0000h
0504h	NSR3	Регистр состояния узла 3	0000h
0506h			0000h
0508h	NIPR3	Регистр указателя прерываний узла 3	0000h
050Ah			0000h
050Ch	NPCR3	Регистр управления портом узла 3	0000h
050Eh			0000h
0510h	NBTR3	Регистр синхронизации битов 3	0000h
0512h			0000h
0514h	NECNT3	Регистр счетчика ошибок узла 3	0000h
0516h			0060h
0518h	NFCR3	Регистр счетчика сообщений 3	0000h
051Ah			0000h
051Ch–0FFEh	–	Зарезервировано	–

Таблица Б.10 – Адреса регистров объектов сообщений (дополнительно см. таблицу Б.11)

Объект сообщения	Регистры объектов сообщений							
	Адрес 0D’xxxx							
	MOFCR	MOFGPR	MOIPR	MOAMR	MO DATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
0	1000h	1004h	1008h	100Ch	1010h	1014h	1018h	101Ch
1	1020h	1024h	1028h	102Ch	1030h	1034h	1038h	103Ch
2	1040h	1044h	1048h	104Ch	1050h	1054h	1058h	105Ch
3	1060h	1064h	1068h	106Ch	1070h	1074h	1078h	107Ch
4	1080h	1084h	1088h	108Ch	1090h	1094h	1098h	109Ch
5	10A0h	10A4h	10A8h	10ACh	10B0h	10B4h	10B8h	10BCh
6	10C0h	10C4h	10C8h	10CCh	10D0h	10D4h	10D8h	10DCh
7	10E0h	10E4h	10E8h	10ECh	10F0h	10F4h	10F8h	10FCh
8	1100h	1104h	1108h	110Ch	1110h	1114h	1118h	111Ch
9	1120h	1124h	1128h	112Ch	1130h	1134h	1138h	113Ch
10	1140h	1144h	1148h	114Ch	1150h	1154h	1158h	115Ch
11	1160h	1164h	1168h	116Ch	1170h	1174h	1178h	117Ch
12	1180h	1184h	1188h	118Ch	1190h	1194h	1198h	119Ch
13	11A0h	11A4h	11A8h	11ACh	11B0h	11B4h	11B8h	11BCh
14	11C0h	11C4h	11C8h	11CCh	11D0h	11D4h	11D8h	11DCh
15	11E0h	11E4h	11E8h	11ECh	11F0h	11F4h	11F8h	11FCh
16	1200h	1204h	1208h	120Ch	1210h	1214h	1218h	121Ch
17	1220h	1224h	1228h	122Ch	1230h	1234h	1238h	123Ch
18	1240h	1244h	1248h	124Ch	1250h	1254h	1258h	125Ch
19	1260h	1264h	1268h	126Ch	1270h	1274h	1278h	127Ch
20	1280h	1284h	1288h	128Ch	1290h	1294h	1298h	129Ch
21	12A0h	12A4h	12A8h	12ACh	12B0h	12B4h	12B8h	12BCh
22	12C0h	12C4h	12C8h	12CCh	12D0h	12D4h	12D8h	12DCh
23	12E0h	12E4h	12E8h	12ECh	12F0h	12F4h	12F8h	12FCh
24	1300h	1304h	1308h	130Ch	1310h	1314h	1318h	131Ch
25	1320h	1324h	1328h	132Ch	1330h	1334h	1338h	133Ch
26	1340h	1344h	1348h	134Ch	1350h	1354h	1358h	135Ch
27	1360h	1364h	1368h	136Ch	1370h	1374h	1378h	137Ch
28	1380h	1384h	1388h	138Ch	1390h	1394h	1398h	139Ch
29	13A0h	13A4h	13A8h	13ACh	13B0h	13B4h	13B8h	13BCh
30	13C0h	13C4h	13C8h	13CCh	13D0h	13D4h	13D8h	13DCh
31	13E0h	13E4h	13E8h	13ECh	13F0h	13F4h	13F8h	13FCh
32	1400h	1404h	1408h	140Ch	1410h	1414h	1418h	141Ch
33	1420h	1424h	1428h	142Ch	1430h	1434h	1438h	143Ch
34	1440h	1444h	1448h	144Ch	1450h	1454h	1458h	145Ch
35	1460h	1464h	1468h	146Ch	1470h	1474h	1478h	147Ch
36	1480h	1484h	1488h	148Ch	1490h	1494h	1498h	149Ch
37	14A0h	14A4h	14A8h	14ACh	14B0h	14B4h	14B8h	14BCh
38	14C0h	14C4h	14C8h	14CCh	14D0h	14D4h	14D8h	14DCh
39	14E0h	14E4h	14E8h	14ECh	14F0h	14F4h	14F8h	14FCh
40	1500h	1504h	1508h	150Ch	1510h	1514h	1518h	151Ch
41	1520h	1524h	1528h	152Ch	1530h	1534h	1538h	153Ch
42	1540h	1544h	1548h	154Ch	1550h	1554h	1558h	155Ch

Продолжение таблицы Б.10

Объект сообще-ния	MOFCR	MOFGPR	MOIPR	MOAMR	MO DATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
43	1560h	1564h	1568h	156Ch	1570h	1574h	1578h	157Ch
44	1580h	1584h	1588h	158Ch	1590h	1594h	1598h	159Ch
45	15A0h	15A4h	15A8h	15ACh	15B0h	15B4h	15B8h	15BCh
46	15C0h	15C4h	15C8h	15CCh	15D0h	15D4h	15D8h	15DCh
47	15E0h	15E4h	15E8h	15ECh	15F0h	15F4h	15F8h	15FCh
48	1600h	1604h	1608h	160Ch	1610h	1614h	1618h	161Ch
49	1620h	1624h	1628h	162Ch	1630h	1634h	1638h	163Ch
50	1640h	1644h	1648h	164Ch	1650h	1654h	1658h	165Ch
51	1660h	1664h	1668h	166Ch	1670h	1674h	1678h	167Ch
52	1680h	1684h	1688h	168Ch	1690h	1694h	1698h	169Ch
53	16A0h	16A4h	16A8h	16ACh	16B0h	16B4h	16B8h	16BCh
54	16C0h	16C4h	16C8h	16CCh	16D0h	16D4h	16D8h	16DCh
55	16E0h	16E4h	16E8h	16ECh	16F0h	16F4h	16F8h	16FCh
56	1700h	1704h	1708h	170Ch	1710h	1714h	1718h	171Ch
57	1720h	1724h	1728h	172Ch	1730h	1734h	1738h	173Ch
58	1740h	1744h	1748h	174Ch	1750h	1754h	1758h	175Ch
59	1760h	1764h	1768h	176Ch	1770h	1774h	1778h	177Ch
60	1780h	1784h	1788h	178Ch	1790h	1794h	1798h	179Ch
61	17A0h	17A4h	17A8h	17ACh	17B0h	17B4h	17B8h	17BCh
62	17C0h	17C4h	17C8h	17CCh	17D0h	17D4h	17D8h	17DCh
63	17E0h	17E4h	17E8h	17ECh	17F0h	17F4h	17F8h	17FCh
64	1800h	1804h	1808h	180Ch	1810h	1814h	1818h	181Ch
65	1820h	1824h	1828h	182Ch	1830h	1834h	1838h	183Ch
66	1840h	1844h	1848h	184Ch	1850h	1854h	1858h	185Ch
67	1860h	1864h	1868h	186Ch	1870h	1874h	1878h	187Ch
68	1880h	1884h	1888h	188Ch	1890h	1894h	1898h	189Ch
69	18A0h	18A4h	18A8h	18ACh	18B0h	18B4h	18B8h	18BCh
70	18C0h	18C4h	18C8h	18CCh	18D0h	18D4h	18D8h	18DCh
71	18E0h	18E4h	18E8h	18ECh	18F0h	18F4h	18F8h	18FCh
72	1900h	1904h	1908h	190Ch	1910h	1914h	1918h	191Ch
73	1920h	1924h	1928h	192Ch	1930h	1934h	1938h	193Ch
74	1940h	1944h	1948h	194Ch	1950h	1954h	1958h	195Ch
75	1960h	1964h	1968h	196Ch	1970h	1974h	1978h	197Ch
76	1980h	1984h	1988h	198Ch	1990h	1994h	1998h	199Ch
77	19A0h	19A4h	19A8h	19ACh	19B0h	19B4h	19B8h	19BCh
78	19C0h	19C4h	19C8h	19CCh	19D0h	19D4h	19D8h	19DCh
79	19E0h	19E4h	19E8h	19ECh	19F0h	19F4h	19F8h	19FCh
80	1A00h	1A04h	1A08h	1A0Ch	1A10h	1A14h	1A18h	1A1Ch
81	1A20h	1A24h	1A28h	1A2Ch	1A30h	1A34h	1A38h	1A3Ch
82	1A40h	1A44h	1A48h	1A4Ch	1A50h	1A54h	1A58h	1A5Ch
83	1A60h	1A64h	1A68h	1A6Ch	1A70h	1A74h	1A78h	1A7Ch
84	1A80h	1A84h	1A88h	1A8Ch	1A90h	1A94h	1A98h	1A9Ch
85	1AA0h	1AA4h	1AA8h	1AACh	1AB0h	1AB4h	1AB8h	1ABCh
86	1AC0h	1AC4h	1AC8h	1ACCh	1AD0h	1AD4h	1AD8h	1ADCh
87	1AE0h	1AE4h	1AE8h	1AECh	1AF0h	1AF4h	1AF8h	1AFCh
88	1B00h	1B04h	1B08h	1B0Ch	1B10h	1B14h	1B18h	1B1Ch

Продолжение таблицы Б.10

Объект сообще-ния	MOFCR	MOFGPR	MOIPR	MOAMR	MODATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
89	1B20h	1B24h	1B28h	1B2Ch	1B30h	1B34h	1B38h	1B3Ch
90	1B40h	1B44h	1B48h	1B4Ch	1B50h	1B54h	1B58h	1B5Ch
91	1B60h	1B64h	1B68h	1B6Ch	1B70h	1B74h	1B78h	1B7Ch
92	1B80h	1B84h	1B88h	1B8Ch	1B90h	1B94h	1B98h	1B9Ch
93	1BA0h	1BA4h	1BA8h	1BACh	1BB0h	1BB4h	1BB8h	1BBCh
94	1BC0h	1BC4h	1BC8h	1BCCh	1BD0h	1BD4h	1BD8h	1BDCh
95	1BE0h	1BE4h	1BE8h	1BECh	1BF0h	1BF4h	1BF8h	1BFCh
96	1C00h	1C04h	1C08h	1C0Ch	1C10h	1C14h	1C18h	1C1Ch
97	1C20h	1C24h	1C28h	1C2Ch	1C30h	1C34h	1C38h	1C3Ch
98	1C40h	1C44h	1C48h	1C4Ch	1C50h	1C54h	1C58h	1C5Ch
99	1C60h	1C64h	1C68h	1C6Ch	1C70h	1C74h	1C78h	1C7Ch
100	1C80h	1C84h	1C88h	1C8Ch	1C90h	1C94h	1C98h	1C9Ch
101	1CA0h	1CA4h	1CA8h	1CACh	1CB0h	1CB4h	1CB8h	1CBCh
102	1CC0h	1CC4h	1CC8h	1CCCh	1CD0h	1CD4h	1CD8h	1CDCh
103	1CE0h	1CE4h	1CE8h	1CECh	1CF0h	1CF4h	1CF8h	1CFCh
104	1D00h	1D04h	1D08h	1D0Ch	1D10h	1D14h	1D18h	1D1Ch
105	1D20h	1D24h	1D28h	1D2Ch	1D30h	1D34h	1D38h	1D3Ch
106	1D40h	1D44h	1D48h	1D4Ch	1D50h	1D54h	1D58h	1D5Ch
107	1D60h	1D64h	1D68h	1D6Ch	1D70h	1D74h	1D78h	1D7Ch
108	1D80h	1D84h	1D88h	1D8Ch	1D90h	1D94h	1D98h	1D9Ch
109	1DA0h	1DA4h	1DA8h	1DACh	1DB0h	1DB4h	1DB8h	1DBCh
110	1DC0h	1DC4h	1DC8h	1DCCh	1DD0h	1DD4h	1DD8h	1DDCh
111	1DE0h	1DE4h	1DE8h	1DECh	1DF0h	1DF4h	1DF8h	1DFCh
112	1E00h	1E04h	1E08h	1E0Ch	1E10h	1E14h	1E18h	1E1Ch
113	1E20h	1E24h	1E28h	1E2Ch	1E30h	1E34h	1E38h	1E3Ch
114	1E40h	1E44h	1E48h	1E4Ch	1E50h	1E54h	1E58h	1E5Ch
115	1E60h	1E64h	1E68h	1E6Ch	1E70h	1E74h	1E78h	1E7Ch
116	1E80h	1E84h	1E88h	1E8Ch	1E90h	1E94h	1E98h	1E9Ch
117	1EA0h	1EA4h	1EA8h	1EACh	1EB0h	1EB4h	1EB8h	1EBCh
118	1EC0h	1EC4h	1EC8h	1ECCh	1ED0h	1ED4h	1ED8h	1EDCh
119	1EE0h	1EE4h	1EE8h	1EECh	1EF0h	1EF4h	1EF8h	1EFCh
120	1F00h	1F04h	1F08h	1F0Ch	1F10h	1F14h	1F18h	1F1Ch
121	1F20h	1F24h	1F28h	1F2Ch	1F30h	1F34h	1F38h	1F3Ch
122	1F40h	1F44h	1F48h	1F4Ch	1F50h	1F54h	1F58h	1F5Ch
123	1F60h	1F64h	1F68h	1F6Ch	1F70h	1F74h	1F78h	1F7Ch
124	1F80h	1F84h	1F88h	1F8Ch	1F90h	1F94h	1F98h	1F9Ch
125	1FA0h	1FA4h	1FA8h	1FACh	1FB0h	1FB4h	1FB8h	1FBCh
126	1FC0h	1FC4h	1FC8h	1FCCh	1FD0h	1FD4h	1FD8h	1FDCh
127	1FE0h	1FE4h	1FE8h	1FECh	1FF0h	1FF4h	1FF8h	1FFCh
128	2000h	2004h	2008h	200Ch	2010h	2014h	2018h	201Ch
129	2020h	2024h	2028h	202Ch	2030h	2034h	2038h	203Ch
130	2040h	2044h	2048h	204Ch	2050h	2054h	2058h	205Ch
131	2060h	2064h	2068h	206Ch	2070h	2074h	2078h	207Ch
132	2080h	2084h	2088h	208Ch	2090h	2094h	2098h	209Ch
133	20A0h	20A4h	20A8h	20ACh	20B0h	20B4h	20B8h	20BCh
134	20C0h	20C4h	20C8h	20CCh	20D0h	20D4h	20D8h	20DCh

Продолжение таблицы Б.10

Объект сообще-ния	MOFCR	MOFGPR	MOIPR	MOAMR	MO DATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
135	20E0h	20E4h	20E8h	20ECh	20F0h	20F4h	20F8h	20FCh
136	2100h	2104h	2108h	210Ch	2110h	2114h	2118h	211Ch
137	2120h	2124h	2128h	212Ch	2130h	2134h	2138h	213Ch
138	2140h	2144h	2148h	214Ch	2150h	2154h	2158h	215Ch
139	2160h	2164h	2168h	216Ch	2170h	2174h	2178h	217Ch
140	2180h	2184h	2188h	218Ch	2190h	2194h	2198h	219Ch
141	21A0h	21A4h	21A8h	21ACh	21B0h	21B4h	21B8h	21BCh
142	21C0h	21C4h	21C8h	21CCh	21D0h	21D4h	21D8h	21DCh
143	21E0h	21E4h	21E8h	21ECh	21F0h	21F4h	21F8h	21FCh
144	2200h	2204h	2208h	220Ch	2210h	2214h	2218h	221Ch
145	2220h	2224h	2228h	222Ch	2230h	2234h	2238h	223Ch
146	2240h	2244h	2248h	224Ch	2250h	2254h	2258h	225Ch
147	2260h	2264h	2268h	226Ch	2270h	2274h	2278h	227Ch
148	2280h	2284h	2288h	228Ch	2290h	2294h	2298h	229Ch
149	22A0h	22A4h	22A8h	22ACh	22B0h	22B4h	22B8h	22BCh
150	22C0h	22C4h	22C8h	22CCh	22D0h	22D4h	22D8h	22DCh
151	22E0h	22E4h	22E8h	22ECh	22F0h	22F4h	22F8h	22FCh
152	2300h	2304h	2308h	230Ch	2310h	2314h	2318h	231Ch
153	2320h	2324h	2328h	232Ch	2330h	2334h	2338h	233Ch
154	2340h	2344h	2348h	234Ch	2350h	2354h	2358h	235Ch
155	2360h	2364h	2368h	236Ch	2370h	2374h	2378h	237Ch
156	2380h	2384h	2388h	238Ch	2390h	2394h	2398h	239Ch
157	23A0h	23A4h	23A8h	23ACh	23B0h	23B4h	23B8h	23BCh
158	23C0h	23C4h	23C8h	23CCh	23D0h	23D4h	23D8h	23DCh
159	23E0h	23E4h	23E8h	23ECh	23F0h	23F4h	23F8h	23FCh
160	2400h	2404h	2408h	240Ch	2410h	2414h	2418h	241Ch
161	2420h	2424h	2428h	242Ch	2430h	2434h	2438h	243Ch
162	2440h	2444h	2448h	244Ch	2450h	2454h	2458h	245Ch
163	2460h	2464h	2468h	246Ch	2470h	2474h	2478h	247Ch
164	2480h	2484h	2488h	248Ch	2490h	2494h	2498h	249Ch
165	24A0h	24A4h	24A8h	24ACh	24B0h	24B4h	24B8h	24BCh
166	24C0h	24C4h	24C8h	24CCh	24D0h	24D4h	24D8h	24DCh
167	24E0h	24E4h	24E8h	24ECh	24F0h	24F4h	24F8h	24FCh
168	2500h	2504h	2508h	250Ch	2510h	2514h	2518h	251Ch
169	2520h	2524h	2528h	252Ch	2530h	2534h	2538h	253Ch
170	2540h	2544h	2548h	254Ch	2550h	2554h	2558h	255Ch
171	2560h	2564h	2568h	256Ch	2570h	2574h	2578h	257Ch
172	2580h	2584h	2588h	258Ch	2590h	2594h	2598h	259Ch
173	25A0h	25A4h	25A8h	25ACh	25B0h	25B4h	25B8h	25BCh
174	25C0h	25C4h	25C8h	25CCh	25D0h	25D4h	25D8h	25DCh
175	25E0h	25E4h	25E8h	25ECh	25F0h	25F4h	25F8h	25FCh
176	2600h	2604h	2608h	260Ch	2610h	2614h	2618h	261Ch
177	2620h	2624h	2628h	262Ch	2630h	2634h	2638h	263Ch
178	2640h	2644h	2648h	264Ch	2650h	2654h	2658h	265Ch
179	2660h	2664h	2668h	266Ch	2670h	2674h	2678h	267Ch

Продолжение таблицы Б.10

Объект сообще-ния	MOFCR	MOFGPR	MOIPR	MOAMR	MO DATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
180	2680h	2684h	2688h	268Ch	2690h	2694h	2698h	269Ch
181	26A0h	26A4h	26A8h	26ACh	26B0h	26B4h	26B8h	26BCh
182	26C0h	26C4h	26C8h	26CCh	26D0h	26D4h	26D8h	26DCh
183	26E0h	26E4h	26E8h	26ECh	26F0h	26F4h	26F8h	26FCh
184	2700h	2704h	2708h	270Ch	2710h	2714h	2718h	271Ch
185	2720h	2724h	2728h	272Ch	2730h	2734h	2738h	273Ch
186	2740h	2744h	2748h	274Ch	2750h	2754h	2758h	275Ch
187	2760h	2764h	2768h	276Ch	2770h	2774h	2778h	277Ch
188	2780h	2784h	2788h	278Ch	2790h	2794h	2798h	279Ch
189	27A0h	27A4h	27A8h	27ACh	27B0h	27B4h	27B8h	27BCh
190	27C0h	27C4h	27C8h	27CCh	27D0h	27D4h	27D8h	27DCh
191	27E0h	27E4h	27E8h	27ECh	27F0h	27F4h	27F8h	27FCh
192	2800h	2804h	2808h	280Ch	2810h	2814h	2818h	281Ch
193	2820h	2824h	2828h	282Ch	2830h	2834h	2838h	283Ch
194	2840h	2844h	2848h	284Ch	2850h	2854h	2858h	285Ch
195	2860h	2864h	2868h	286Ch	2870h	2874h	2878h	287Ch
196	2880h	2884h	2888h	288Ch	2890h	2894h	2898h	289Ch
197	28A0h	28A4h	28A8h	28ACh	28B0h	28B4h	28B8h	28BCh
198	28C0h	28C4h	28C8h	28CCh	28D0h	28D4h	28D8h	28DCh
199	28E0h	28E4h	28E8h	28ECh	28F0h	28F4h	28F8h	28FCh
200	2900h	2904h	2908h	290Ch	2910h	2914h	2918h	291Ch
201	2920h	2924h	2928h	292Ch	2930h	2934h	2938h	293Ch
202	2940h	2944h	2948h	294Ch	2950h	2954h	2958h	295Ch
203	2960h	2964h	2968h	296Ch	2970h	2974h	2978h	297Ch
204	2980h	2984h	2988h	298Ch	2990h	2994h	2998h	299Ch
205	29A0h	29A4h	29A8h	29ACh	29B0h	29B4h	29B8h	29BCh
206	29C0h	29C4h	29C8h	29CCh	29D0h	29D4h	29D8h	29DCh
207	29E0h	29E4h	29E8h	29ECh	29F0h	29F4h	29F8h	29FCh
208	2A00h	2A04h	2A08h	2A0Ch	2A10h	2A14h	2A18h	2A1Ch
209	2A20h	2A24h	2A28h	2A2Ch	2A30h	2A34h	2A38h	2A3Ch
210	2A40h	2A44h	2A48h	2A4Ch	2A50h	2A54h	2A58h	2A5Ch
211	2A60h	2A64h	2A68h	2A6Ch	2A70h	2A74h	2A78h	2A7Ch
212	2A80h	2A84h	2A88h	2A8Ch	2A90h	2A94h	2A98h	2A9Ch
213	2AA0h	2AA4h	2AA8h	2AACh	2AB0h	2AB4h	2AB8h	2ABCh
214	2AC0h	2AC4h	2AC8h	2ACCh	2AD0h	2AD4h	2AD8h	2ADCh
215	2AE0h	2AE4h	2AE8h	2AECh	2AF0h	2AF4h	2AF8h	2AFCh
216	2B00h	2B04h	2B08h	2B0Ch	2B10h	2B14h	2B18h	2B1Ch
217	2B20h	2B24h	2B28h	2B2Ch	2B30h	2B34h	2B38h	2B3Ch
218	2B40h	2B44h	2B48h	2B4Ch	2B50h	2B54h	2B58h	2B5Ch
219	2B60h	2B64h	2B68h	2B6Ch	2B70h	2B74h	2B78h	2B7Ch
220	2B80h	2B84h	2B88h	2B8Ch	2B90h	2B94h	2B98h	2B9Ch
221	2BA0h	2BA4h	2BA8h	2BACh	2BB0h	2BB4h	2BB8h	2BBCh
222	2BC0h	2BC4h	2BC8h	2BCCh	2BD0h	2BD4h	2BD8h	2BDCh
223	2BE0h	2BE4h	2BE8h	2BECh	2BF0h	2BF4h	2BF8h	2BFCh
224	2C00h	2C04h	2C08h	2C0Ch	2C10h	2C14h	2C18h	2C1Ch
225	2C20h	2C24h	2C28h	2C2Ch	2C30h	2C34h	2C38h	2C3Ch

Окончание таблицы Б.10

Объект сообще-ния	MOFCR	MOFGPR	MOIPR	MOAMR	MO DATAL	MO DATAH	MOAR	MOCTR/ MOSTAT
226	2C40h	2C44h	2C48h	2C4Ch	2C50h	2C54h	2C58h	2C5Ch
227	2C60h	2C64h	2C68h	2C6Ch	2C70h	2C74h	2C78h	2C7Ch
228	2C80h	2C84h	2C88h	2C8Ch	2C90h	2C94h	2C98h	2C9Ch
229	2CA0h	2CA4h	2CA8h	2CACh	2CB0h	2CB4h	2CB8h	2CBCh
230	2CC0h	2CC4h	2CC8h	2CCCh	2CD0h	2CD4h	2CD8h	2CDCh
231	2CE0h	2CE4h	2CE8h	2CECh	2CF0h	2CF4h	2CF8h	2CFCh
232	2D00h	2D04h	2D08h	2D0Ch	2D10h	2D14h	2D18h	2D1Ch
233	2D20h	2D24h	2D28h	2D2Ch	2D30h	2D34h	2D38h	2D3Ch
234	2D40h	2D44h	2D48h	2D4Ch	2D50h	2D54h	2D58h	2D5Ch
235	2D60h	2D64h	2D68h	2D6Ch	2D70h	2D74h	2D78h	2D7Ch
236	2D80h	2D84h	2D88h	2D8Ch	2D90h	2D94h	2D98h	2D9Ch
237	2DA0h	2DA4h	2DA8h	2DACh	2DB0h	2DB4h	2DB8h	2DBCh
238	2DC0h	2DC4h	2DC8h	2DCCh	2DD0h	2DD4h	2DD8h	2DDCh
239	2DE0h	2DE4h	2DE8h	2DECh	2DF0h	2DF4h	2DF8h	2DFCh
240	2E00h	2E04h	2E08h	2E0Ch	2E10h	2E14h	2E18h	2E1Ch
241	2E20h	2E24h	2E28h	2E2Ch	2E30h	2E34h	2E38h	2E3Ch
242	2E40h	2E44h	2E48h	2E4Ch	2E50h	2E54h	2E58h	2E5Ch
243	2E60h	2E64h	2E68h	2E6Ch	2E70h	2E74h	2E78h	2E7Ch
244	2E80h	2E84h	2E88h	2E8Ch	2E90h	2E94h	2E98h	2E9Ch
245	2EA0h	2EA4h	2EA8h	2EACh	2EB0h	2EB4h	2EB8h	2EBCh
246	2EC0h	2EC4h	2EC8h	2ECCh	2ED0h	2ED4h	2ED8h	2EDCh
247	2EE0h	2EE4h	2EE8h	2EECh	2EF0h	2EF4h	2EF8h	2EFCh
248	2F00h	2F04h	2F08h	2F0Ch	2F10h	2F14h	2F18h	2F1Ch
249	2F20h	2F24h	2F28h	2F2Ch	2F30h	2F34h	2F38h	2F3Ch
250	2F40h	2F44h	2F48h	2F4Ch	2F50h	2F54h	2F58h	2F5Ch
251	2F60h	2F64h	2F68h	2F6Ch	2F70h	2F74h	2F78h	2F7Ch
252	2F80h	2F84h	2F88h	2F8Ch	2F90h	2F94h	2F98h	2F9Ch
253	2FA0h	2FA4h	2FA8h	2FACh	2FB0h	2FB4h	2FB8h	2FBCh
254	2FC0h	2FC4h	2FC8h	2FCCh	2FD0h	2FD4h	2FD8h	2FDCh
255	2FE0h	2FE4h	2FE8h	2FECh	2FF0h	2FF4h	2FF8h	2FFCh
Адреса 0D’3000h – 0D’3FFCh являются зарезервированными								

Таблица Б.11 – Мнемоника и соответствующие названия регистров объектов сообщений контроллера CAN

Мнемоника	Сброс	Название
MOFCR	00000000h	Регистр управления функционированием объекта сообщения
MOFGPR	00000000h	Регистр указателя FIFO/шлюза объекта сообщения
MOIPR	00000000h	Регистр указателя прерываний объекта сообщения
MOAMR	3FFFFFFFh	Регистр маски объекта сообщения
MODATAL	00000000h	Младший регистр данных объекта сообщения
MODATAH	00000000h	Старший регистр данных объекта сообщения
MOAR	00000000h	Регистр арбитража объекта сообщения
MOCTR/ MOSTAT	См. таблицу Б.12	Регистр управления/состояния объекта сообщения

Таблица Б.12 – Состояние регистров MOCTR/MOSTAT объектов сообщений после сброса (n-порядковый номер объекта сообщения)

n (код)	MOCTR / MOSTAT		
	PNEXT (биты 31–24)	PPREV (биты 23–16)	Биты 15–0
0 (00h)	01h	00h	0000h
1 (01h)	02h	00h	
2 (02h)	03h	01h	
3 (03h)	04h	02h	
…	…	…	
252 (FCh)	FDh	FBh	
253 (FDh)	FEh	FCh	
254 (FEh)	FFh	FDh	
255 (FFh)	FFh	FEh	

Приложение В (обязательное) Таблица векторов прерываний

В таблице В.1 указаны источники прерываний, соответствующие им вектора прерываний и управляющие регистры (адреса в приложении Б).

Таблица В.1 – Векторы прерываний

Номер	Источник прерывания	Адрес вектора	Регистр управления	Примечание
00h	Сброс (RESET)	0000h	–	–
01h	–	0004h	–	–
02h	NMI (немаскируемое прерывание)	0008h	–	–
03h	–	000Ch	–	–
04h	Переполнение стека	0010h	–	–
05h	–	0014h	–	–
06h	Опустошение стека	0018h	–	–
07h	–	001Ch	–	–
08h	Программный останов	0020h	–	–
09h	–	0024h	–	–
0Ah	Прерывание класса B	0028h	–	–
0Bh	–	002Ch	–	–
0Ch	–	0030h	–	–
0Dh	–	0034h	–	–
0Eh	–	0038h	–	–
0Fh	–	003Ch	–	–
10h	Регистр сравнения 0	0040h	CC0IC	CAPCOM1
11h	Регистр сравнения 1	0044h	CC1IC	
12h	Регистр сравнения 2	0048h	CC2IC	
13h	Регистр сравнения 3	004Ch	CC3IC	
14h	Регистр сравнения 4	0050h	CC4IC	
15h	Регистр сравнения 5	0054h	CC5IC	
16h	Регистр сравнения 6	0058h	CC6IC	
17h	Регистр сравнения 7	005Ch	CC7IC	
18h	Регистр сравнения 8	0060h	CC8IC	
19h	Регистр сравнения 9	0064h	CC9IC	
1Ah	Регистр сравнения 10	0068h	CC10IC	
1Bh	Регистр сравнения 11	006Ch	CC11IC	
1Ch	Регистр сравнения 12	0070h	CC12IC	
1Dh	Регистр сравнения 13	0074h	CC13IC	
1Eh	Регистр сравнения 14	0078h	CC14IC	
1Fh	Регистр сравнения 15	007Ch	CC15IC	
20h	Таймер T0	0080h	T0IC	
21h	Таймер T1	0084h	T1IC	
22h	Таймер T2	0088h	T2IC	GPT1, GPT2
23h	Таймер T3	008Ch	T3IC	
24h	Таймер T4	0090h	T4IC	
25h	Таймер T5	0094h	T5IC	
26h	Таймер T6	0098h	T6IC	
27h	Захват/перезагрузка	009Ch	CRIC	
28h	Завершение преобразования АЦП	00A0h	ADC_CIC	–

Продолжение таблицы В.1

Номер	Источник прерывания	Адрес вектора	Регистр управления	Примечание
29h	Ошибка АЦП	00A4h	ADC_EIC	–
2Ah	Передача ASC0	00A8h	S0TIC	–
2Bh	Прием ASC0	00ACh	S0RIC	
2Ch	Ошибка ASC0	00B0h	S0EIC	
2Dh	Передача SSC0	00B4h	SSC0TIC	–
2Eh	Прием SSC0	00B8h	SSC0RIC	
2Fh	Ошибка SSC0	00BCh	SSC0EIC	
30h	Регистр сравнения 16	00C0h	CC16IC	CAPCOM2
31h	Регистр сравнения 17	00C4h	CC17IC	
32h	Регистр сравнения 18	00C8h	CC18IC	
33h	Регистр сравнения 19	00CCh	CC19IC	
34h	Регистр сравнения 20	00D0h	CC20IC	
35h	Регистр сравнения 21	00D4h	CC21IC	
36h	Регистр сравнения 22	00D8h	CC22IC	
37h	Регистр сравнения 23	00DCh	CC23IC	
38h	Регистр сравнения 24	00E0h	CC24IC	
39h	Регистр сравнения 25	00E4h	CC25IC	
3Ah	Регистр сравнения 26	00E8h	CC26IC	
3Bh	Регистр сравнения 27	00ECh	CC27IC	
3Ch	Регистр сравнения 28	00F0h	CC28IC	
3Dh	Таймер T7	00F4h	T7IC	–
3Eh	Таймер T8	00F8h	T8IC	–
3Fh	Программный запрос 15	00FCh	IRQ15	–
40h	I2C	0100h	I2C_DIC	–
41h	–	0104h	–	–
42h	–	0108h	–	–
43h	Нахождение частоты PLL	010Ch	PLLLOCK_IC	
44h	Регистр сравнения 29	0110h	CC29IC	CAPCOM2
45h	Регистр сравнения 30	0114h	CC30IC	
46h	Регистр сравнения 31	0118h	CC31IC	
47h	Буфер передатчика ASC0	011Ch	S0TBIC	–
48h	Передача ASC1	0120h	S1TIC	–
49h	Прием ASC1	0124h	S1RIC	–
4Ah	Ошибка ASC1	0128h	S1EIC	–
4Bh	Сторожевой таймер WDT	012Ch	WDTIC	–
4Ch	Окончание PEC передач	0130h	EOPIC	–
4Dh	Таймер 12	0134h	CCU6_T12IC	CAPCOM6
4Eh	Таймер 13	0138h	CCU6_T13IC	
4Fh	Датчик	013Ch	CCU6_EIC	CAPCOM6
50h	CAPCOM6	0140h	CCU6_IC	–
51h	Передача SSC1	0144h	SSC1TIC	–
52h	Прием SSC1	0148h	SSC1RIC	–
53h	Ошибка SSC1	014Ch	SSC1EIC	–
54h	CAN0	0150h	CAN_0IC	–
55h	CAN1	0154h	CAN_1IC	–
56h	CAN2	0158h	CAN_2IC	–
57h	CAN3	015Ch	CAN_3IC	–

Окончание таблицы В.1

Номер	Источник прерывания	Адрес вектора	Регистр управления	Примечание
58h	Программный запрос 72	0160h	IRQ 72	–
59h	CAN4	0164h	CAN_4IC	–
5Ah	CAN5	0168h	CAN_5IC	–
5Bh	CAN6	016Ch	CAN_6IC	–
5Ch	CAN7	0170h	CAN_7IC	–
5Dh	RTC	0174h	RTC_IC	–
5Eh	Буфер передатчика ASC1	0178h	S1TBIC	–
5Fh	Программный запрос 79	017Ch	IRQ 79	–
60h	CAN8	0180h	CAN_8IC	–
61h	CAN9	0184h	CAN_9IC	–
62h	CAN10	0188h	CAN_10IC	–
63h	CAN11	018Ch	CAN_11IC	–
64h	CAN12	0190h	CAN_12IC	–
65h	CAN13	0194h	CAN_13IC	–
66h	CAN14	0198h	CAN_14IC	–
67h	CAN15	019Ch	CAN_15IC	–
68h	Регистр сравнения RTCCMP1	01A0h	RTC_IC1	–
69h	Регистр сравнения RTCCMP2	01A4h	RTC_IC2	–
6Ah	Регистр сравнения RTCCMP3	01A8h	RTC_IC3	–
6Bh	Потеря частоты PLL	01Ach	PLLUNLOCK_IC	–
6Ch	Программный запрос 92	01B0h	IRQ 92	–
6Dh	Программный запрос 93	01B4h	IRQ 93	–
6Eh	Программный запрос 94	01B8h	IRQ 94	–
6Fh	Ошибка SPI	01BCh	SSC2EIC	–
70h	Прием SPI	01C0h	SSC2RIC	–
71h	Передача SPI	01C4h	SSC2TIC	–
72h	Ошибка USART2	01C8h	S2EIC	–
73h	Прием USART2	01CCh	S2RIC	–
74h	Буфер передатчика USART2	01D0h	S2TBIC	–
75h	Передача USART2	01D4h	S2TIC	–
76h	Программный запрос 102	01D8h	IRQ 102	–
77h	Программный запрос 103	01DCh	IRQ 103	–
78h	Программный запрос 104	01E0h	IRQ 104	–
79h	Программный запрос 105	01E4h	IRQ 105	–
7Ah	Программный запрос 106	01E8h	IRQ 106	–
7Bh	Программный запрос 107	01ECh	IRQ 107	–
7Ch	Программный запрос 108	01F0h	IRQ 108	–
7Dh	Программный запрос 109	01F4h	IRQ 109	–
7Eh	Программный запрос 110	01F8h	IRQ 110	–
7Fh	Программный запрос 111	01FCh	IRQ 111	–

Приложение Г (обязательное) Система команд микроконтроллера

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

Имя команды – уникальный мнемонический код команды.

- Синтаксис

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

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 1887ВЕ9Т – двух- и четырехбайтные.

Формат команды микроконтроллера изображен на рисунке Г.1.

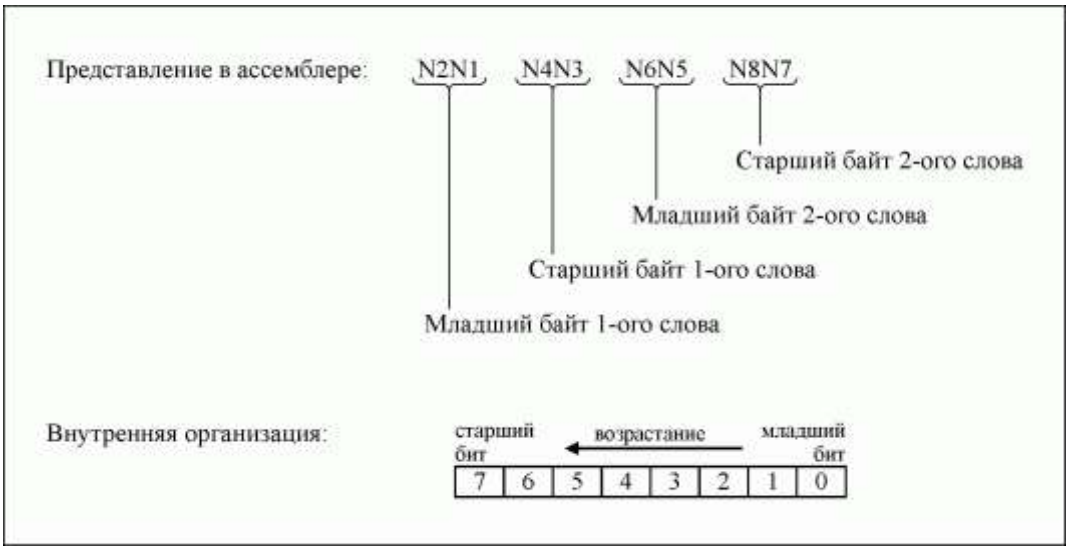


Рисунок Г.1 – Формат команды микроконтроллера 1887ВЕ9Т

ADD **Целочисленное сложение**

Синтаксис ADD op1, op2

Действие (opl) ← (opl) + (op2)

Тип данных WORD

Описание Выполняется сложение содержимого двух операндов, представленных в дополнительном коде – источника op2 и приемника op1. Сумма сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда для указанного типа данных. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADD	Rw_n , Rw_m	00 nm	2
ADD	Rw_n, [Rw_i]	08 n:1 0 i i	2
ADD	Rw_n, [Rw_i +]	08 n:l l i i	2
ADD	Rw_n, #data3	08 n:0 ###	2
ADD	reg, #data16	06 RR ####	4
ADD	reg, mem	02 RR MM MM	4
ADD	mem, reg	04 RR MM MM	4

ADDB **Целочисленное сложение**

Синтаксис ADDB op1, op2

Действие (opl) ← (opl) + (op2)

Тип данных BYTE

Описание Выполняется сложение содержимого двух операндов, представленных в дополнительном коде – источника op2 и приемника op1. Сумма сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда для указанного типа данных. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDB	Rb_n , Rb_m	01 nm	2
ADDB	Rb_n, [Rb_i]	09 n:1 0 i i	2
ADDB	Rb_n, [Rb_i +]	09 n:l l i i	2
ADDB	Rb_n, #data3	09 n:0 ###	2
ADDB	reg, #data16	07 RR ## xx	4
ADDB	reg, mem	03 RR MM MM	4
ADDB	mem, reg	05 RR MM MM	4

ADDC **Целочисленное сложение с переносом**

Синтаксис ADDC op1, op2

Действие (opl) ← (opl) + (op2) + (C)

Тип данных WORD

Описание Выполняется сложение содержимого трех операндов – источника op2, приемника op1 и бита переноса C, где операнды op1 и op2 – числа в дополнительном коде. Сумма сохраняется в op1. Эта инструкция может использоваться для вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю и флаг Z был установлен до выполнения инструкции. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDC	Rw_n , Rw_m	10 nm	2
ADDC	Rw_n, [Rw_i]	18 n:10 i i	2
ADDC	Rw_n, [Rw_i +]	18 n:l l i i	2
ADDC	Rw_n, #data3	18 n:0 # # #	2
ADDC	reg, #data16	16 RR ## ##	4
ADDC	Reg, mem	12 RR MM MM	4
ADDC	mem, reg	14 RR MM MM	4

ADDCB **Целочисленное сложение с переносом**

Синтаксис ADDCB op1, op2

Действие (opl) ← (opl) + (op2) + (C)

Тип данных BYTE

Описание Выполняется сложение содержимого трех операндов – источника op2, приемника op1 и бита переноса C, где операнды op1 и op2 – числа в дополнительном коде. Сумма сохраняется в op1. Эта инструкция может использоваться для вычислений с повышенной точностью.

Флаги состояния

E	Z	V	C	N
*	S	*	*	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю и флаг Z был установлен до выполнения инструкции. В противном случае сбрасывается.

V Устанавливается, если произошло арифметическое переполнение, т. е. результат не может быть представлен указанным типом данных. В противном случае сбрасывается.

C Устанавливается, если произошел перенос из старшего разряда. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ADDCB	Rb_n , Rb_m	11 nm	2
ADDCB	Rb_n, [Rw_i]	19 n:10 i i	2
ADDCB	Rb_n, [Rw_i +]	19 n:l l i i	2
ADDCB	Rb_n, #data3	19 n:0 # # #	2
ADDCB	reg, #data16	17 RR # # xx	4
ADDCB	reg, mem	13 RR MM MM	4
ADDCB	mem, reg	15 RR MM MM	4

AND **Логическое «И»**

Синтаксис AND op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных WORD

Описание Выполняется побитовая операция логического «И» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
AND	Rw_n , Rw_m	60 nm	2
AND	Rw_n, [Rw_i]	68 n:10 i i	2
AND	Rw_n, [Rw_i +]	68 n:l l i i	2
AND	Rw_n, #data3	68 n:0 # # #	2
AND	reg, #data16	66 RR ## ##	4
AND	reg, mem	62 RR MM MM	4
AND	mem, reg	64 RR MM MM	4

ANDB **Логическое «И»**

Синтаксис ANDB op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных BYTE

Описание Выполняется побитовая операция логического «И» над содержимым операндов источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ANDB	Rb_n , Rb_m	61 nm	2
ANDB	Rb_n, [Rw_i]	69 n:10 i i	2
ANDB	Rb_n, [Rw_i +]	69 n:11 i i	2
ANDB	Rb_n, #data3	69 n:0 # # #	2
ANDB	reg, #data8	67 RR ## xx	4
ANDB	reg, mem	63 RR MM MM	4
ANDB	mem, reg	65 RR MM MM	4

ASHR **Арифметический сдвиг в сторону младших адресов**

Синтаксис ASHR op1, op2

Действие

(count) ← (op1)
(V) ← 0, (C) ← 0
DO WHILE ((count) ≠ 0)
(V) ← (C) v (V), (C) ← ($op1_0$)
($op1_n$) ← ($op1_{n+1}$) [n = 0, …, 14]
(count) ← (count) – 1
END WHILE

Тип данных WORD

Описание Арифметически сдвигается содержимое приемника op1 на количество разрядов, определяемое источником op2. Сдвиг производится в сторону младших разрядов. Для сохранения знака операнда op1 старшие разряды результата заполняются нулями, если первоначально старший разряд был 0, или единицами, если старший разряд был 1. Флаг переполнения V используется как флаг округления. Младший значащий разряд сдвигается во флаг переноса C. Допустимые значения сдвига – от 0 до 15 включительно. Если op2 является регистром общего назначения, то используются только 4 младших бита.

Флаги состояния

E	Z	V	C	N
0	*	S	S	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если в любом цикле сдвига происходит переполнение (теряется 1 из флага C). Сбрасывается, если значение содержимого op2 равно 0.

C Принимает значение бита, сдвигаемого из младшего разряда op1. Сбрасывается, если значение содержимого op2 равно 0.

N Устанавливается, если установлен старший разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
ASHR	Rw_n , Rw_m	AC nm	2
ASHR	Rw_n, #data4	BC #n	2

ATOMIC **Временное запрещение обработки запросов на прерывание**

Синтаксис ATOMIC op1

Действие

```
(count) ← opl   [1 ≤ opl ≤ 4]
IRQ–processing = Disable      // запрещение обработки
                              // запросов на
                              // прерывание
DO WHILE ( ( count ) ≠ 0 AND Class_B_IRQ ≠ TRUE)
      ...  // выполнение инструкции
      (count) ← (count) - 1
END WHILE
(count) = 0
IRQ_processing = Enable       //разрешение обработки
                              // запросов на
                              // прерывание
```

Описание Запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC на время выполнения указанного количества инструкций. Действие ATOMIC распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие ATOMIC, определяется операндом op1 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие ATOMIC прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
ATOMIC	#irang2	D1: 00 ## – 0	2

BAND **Битовое логическое «И»**

Синтаксис BAND op1, op2

Действие (opl) ← (opl) ^ (op2)

Тип данных BIT

Описание Выполняется операция логического «И» над содержимым источника op2 и содержимым приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕЕ ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BAND	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	6A QQ ZZ qz	4

BCLR **Очистка бита**

Синтаксис BCLR op1

Действие (opl) ← 0

Тип данных BIT

Описание Обнуляется значение битового операнда op1. Эта инструкция обычно используется для управления системой и периферийными устройствами.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее значение указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
BCLR	$bitaddr_{Q.q}$	qE QQ	2

BCMP **Сравнение битов**

Синтаксис BCMP op1, op2

Действие (op1) ⇔ (op2)

Тип данных BIT

Описание Выполняется сравнение содержимого операнда op1 с содержимым операнда op2. Результат не сохраняется. Изменяются только флаги состояния.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BCMP	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	2A QQ ZZ qz	4

BFLDH **Битовое поле старшего байта**

Синтаксис BFLDH op1, op2, op3

Действие (tmp) ← (op1)
(high byte (tmp)) ← ((high byte (tmp) ^ ¬op2) v op3)
(op1) ← (tmp)

Тип данных WORD

Описание Производится побитная замена старшего байта содержимого приемника op1 значениями источника op3. Маской замены являются значения битов операнда op2. Если значение бита маски равно 1, то производится замена в соответствующем бите приемника, если значение бита маски равно 0, то замена не производится.

Флаги состояния

E	Z	V	C	N
0	*	0	0	*

E Всегда сбрасывается.

Z Устанавливается, если старший и младший байты результата равны нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд старшего байта результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
BFLDH	$bitoff_Q$, #mask8, #data8	1A QQ ## @@	4

BFLDL **Битовое поле младшего байта**

Синтаксис BFLDL op1, op2, op3

Действие (tmp) ← (op1)
(low byte (tmp) ← ((low byte (tmp) ^ ¬op2) v op3)
(op1) ← (tmp)

Тип данных WORD

Описание Производится побитная замена младшего байта содержимого приемника op1 значениями источника op3. Маской замены являются значения битов операнда op2. Если значение бита маски равно 1, то производится замена в соответствующем бите приемника, если значение бита маски равно 0, то замена не производится.

Флаги состояния

E	Z	V	C	N
0	*	0	0	*

E Всегда сбрасывается.

Z Устанавливается, если старший и младший байты результата op1 равны нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд старшего байта результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
BFLDL	$bitoff_Q$, #mask8, #data8	0A QQ @@ ##	4

BMOV **Битовая пересылка**

Синтаксис BMOV op1, op2

Действие (opl) ← (op2)

Тип данных BIT

Описание Содержимое источника op2 пересылается в приемник op1. Флаги устанавливаются соответственно биту источника.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния источника.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние источника.

Формат инструкции

Мнемоника		Формат	Размер
BMOV	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	4A QQ ZZ qz	4

BMOVN **Битовая пересылка с инверсией**

Синтаксис BMOVN op1, op2

Действие (opl) ← ¬ (op2)

Тип данных BIT

Описание Инвертированное значение источника op2 пересылается в приемник op1. Флаги устанавливаются соответственно биту источника.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Не изменяется.

Z Содержит инверсное значение предыдущего состояния источника.

V Не изменяется.

C Не изменяется.

N Содержит предыдущее состояние источника.

Формат инструкции

Мнемоника		Формат	Размер
BMOVN	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	3A QQ ZZ qz	4

BOR **Битовое «ИЛИ»**

Синтаксис BOR op1, op2

Действие (opl) ← (opl) v (op2)

Тип данных BIT

Описание Выполняется операция логического «ИЛИ» над содержимым источника op2 и приемника op1. Результат записывается в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BOR	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	5A QQ ZZ qz	4

BSET **Установка бита**

Синтаксис BSET op1

Действие (opl) ← 1

Тип данных BIT

Описание Содержимое операнда op1 устанавливается в 1. Эта инструкция обычно используется для управления системой и периферийными устройствами.

Флаги состояния

E	Z	V	C	N
0	B#	0	0	B

E Всегда сбрасывается.

Z Содержит инверсное значение предыдущего состояния указанного бита.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Содержит предыдущее состояние указанного бита.

Формат инструкции

Мнемоника		Формат	Размер
BSET	$bitaddr_{Q.q}$	qF QQ	2

BXOR **Битовое «ИСКЛЮЧАЮЩЕЕ ИЛИ»**

Синтаксис BXOR op1, op2

Действие (opl) ← (opl) $\oplus$ (op2)

Тип данных BIT

Описание Выполняется операция «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым источника op2 и приемника op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
0	NOR	OR	AND	XOR

E Всегда сбрасывается.

Z Результат выполнения операции «ИЛИ-НЕ» над содержимым операндов.

V Результат выполнения операции «ИЛИ» над содержимым операндов.

C Результат выполнения операции «И» над содержимым операндов.

N Результат выполнения операции «ИСКЛЮЧАЮЩЕГО ИЛИ» над содержимым операндов.

Формат инструкции

Мнемоника		Формат	Размер
BXOR	$bitaddr_{Z.z}$, $bitaddr_{Q.q}$	7A QQ ZZ qz	4

CALLA **Абсолютный вызов подпрограммы**

Синтаксис CALLA op1, op2

Действие

```
IF  (opl)  THEN
      (SP)  ←  (SP)  -  2
      ( (SP) )  ←  (IP)
      (IP)  ←  (op2)
ELSE
 ...   // выполнение   следующей инструкции
END   IF
```

Описание Если выполняется условие, указанное в op1 (см. коды условий перехода), то осуществляется переход внутри сегмента по адресу, указанному в op2. Значение указателя стека SP уменьшается на 2 и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Если условие не выполняется, никаких действий не производится, и следующая инструкция выполняется как обычно.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLA	cc, caddr	CA c0 MM MM	4

CALLI **Косвенный вызов подпрограммы**

Синтаксис CALLI op1, op2

Действие

```
IF (opl) THEN
      (SP) ← (SP) - 2
      ( (SP) ) ← (IP)
      (IP) ← (op2)
ELSE
  ... // выполнение следующей инструкции
END IF
```

Описание Если выполняется условие, указанное в op1 (см. коды условий перехода), то осуществляется переход внутри сегмента по адресу, равному содержимому op2. Значение указателя стека SP уменьшается на 2 и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Если условие не выполняется, никаких действий не производится, и следующая инструкция выполняется как обычно.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLI	cc, [Rw_n]	AB cn	2

CALLR **Относительный вызов подпрограммы**

Синтаксис CALLR op1

Действие (SP) ← (SP) - 2
((SP)) ← (IP)
(IP) ← (IP) + opl * 2

Описание Осуществляется безусловный переход внутри сегмента по адресу, определяемому суммой указателя инструкций IP и удвоенного смещения op1. Смещение воспринимается как число в дополнительном коде. Значение указателя стека SP уменьшается на 2, и содержимое указателя инструкций IP помещается на вершину системного стека. Поскольку IP всегда указывает на инструкцию, следующую за переходом, то значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы. Значение IR, используемое для вычисления адреса перехода, является адресом инструкции, следующей за CALLR.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLR	rel	BB rr	2

CALLS **Межсегментный вызов подпрограммы**

Синтаксис CALLS op1, op2

Действие
(SP) ← (SP) - 2
((SP)) ← (CSP)
(SP) ← (SP) - 2
((SP)) ← (IP)
(CSP) ← opl
(IP) ← op2

Описание Осуществляется безусловный переход по полному 24-битовому адресу. Номер сегмента определяется операндом op1, а внутрисегментное смещение – операндом op2. Содержимое указателя команд IP и указателя сегмента CSP помещаются на вершину системного стека. Значение указателя стека SP перед каждым занесением на стек уменьшается на 2. Поскольку IP всегда указывает на инструкцию, следующую переходом, значение, сохраненное в стеке, представляет собой адрес возврата для вызываемой подпрограммы.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
CALLS	seg, caddr	DA SS MM MM	4

CMP **Целочисленное сравнение**

Синтаксис CMP op1, op2

Действие (opl) ⇔ (op2)

Тип данных WORD

Описание Сравнивается содержимое операнда op1 с содержимым операнда op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. Флаги устанавливаются по правилам вычитания. Операнды остаются неизменными.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMP	Rw_n , Rw_n	40 nm	2
CMP	Rw_n, [Rw_i]	48 n:10 i i	2
CMP	Rw_n, [Rw_i +]	48 n:l l i i	2
CMP	Rw_n, #data3	48 n:0 # # #	2
CMP	reg, #data16	46 RR ## ##	4
CMP	reg, mem	42 RR MM MM	4

CMPB **Целочисленное сравнение**

Синтаксис CMPB op1, op2

Действие (opl) ⇔ (op2)

Тип данных BYTE

Описание Сравнивается содержимое операнда op1 и операнда op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. Флаги устанавливаются по правилам вычитания. Операнды остаются неизменными.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPB	Rb_n , Rb_n	41 nm	2
CMPB	Rb_n, [Rw_i]	49 n:10 i i	2
CMPB	Rb_n, [Rw_i +]	49 n:ll i i	2
CMPB	Rb_n, #data3	49 n:0 # # #	2
CMPB	reg, #data16	47 RR ## xx	4
CMPB	reg, mem	43 RR MM MM	4

CMPD1 **Целочисленное сравнение с уменьшением на единицу**

Синтаксис CMPD1 op1, op2

Действие (opl) $\Leftrightarrow$ (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) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) - 2

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 уменьшается на 2. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

- E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.
- Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.
- V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.
- C Устанавливается, если произошел заем. В противном случае сбрасывается.
- N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPD2	Rw_n, #data4	B0 #n	2
CMPD2	Rw_n, #data16	B6 Fn ## ##	4
CMPD2	Rw_n, mem	B2 RR MM MM	4

CMPI1 **Целочисленное сравнение с увеличением на единицу**

Синтаксис CMPI1 op1, op2

Действие (opl) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) + 1

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 увеличивается на 1. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т.е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPI1	Rw_n, #data4	80 #n	2
CMPI1	Rw_n, #data16	86 Fn ## ##	4
CMPI1	Rw_n, mem	82 Fn MM MM	4

CMPI2 **Целочисленное сравнение с увеличением на 2**

Синтаксис CMPI2 op1, op2

Действие (opl) $\Leftrightarrow$ (op2)
(opl) $\leftarrow$ (opl) + 2

Тип данных WORD

Описание Содержимое приемника op1 сравнивается с содержимым источника op2 посредством выполнения вычитания в двоичном дополнительном коде op2 из op1. В качестве операнда op1 могут выступать только регистры РОН. После сравнения содержимое приемника op1 увеличивается на 2. Используя флаги и инструкции ветвления, можно реализовывать любые циклы. Эта инструкция используется для уменьшения времени выполнения циклов.

Флаги состояния

E	Z	V	C	N
*	*	*	S	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных. В противном случае сбрасывается.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CMPI2	Rw_n, #data4	90 #n	2
CMPI2	Rw_n, #data16	96 Fn ## ##	4
CMPI2	Rw_n, mem	92 Fn MM MM	4

CPL **Поразрядная инверсия**

Синтаксис CPL op1

Действие (opl) ← ¬ (opl)

Тип данных WORD

Описание Вычисляется поразрядная инверсия содержимого операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CPL	Rw_n	91 n0	2

CPLB **Поразрядная инверсия**

Синтаксис CPLB op1

Действие (opl) ← ¬ (opl)

Тип данных BYTE

Описание Вычисляется поразрядная инверсия содержимого операнда op1. Результат сохраняется в op1.

Флаги состояния

E	Z	V	C	N
*	*	0	0	*

E Устанавливается, если содержимое op1 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Всегда сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
CPLB	Rb_n,	B1 n0	2

DISWDT **Запрещение работы сторожевого таймера WDT**

Синтаксис DISWDT

Действие Запрещение работы сторожевого таймера

Описание Запрещается работа сторожевого таймера WDT. WDT начинает работать после сброса. После сброса эту инструкцию следует выполнить до выполнения инструкции перезапуска SRVWDT сторожевого таймера или инструкции конца инициализации EINIT. После выполнения одной из этих инструкций, DISWDT перестает действовать. Чтобы избежать случайного выполнения, инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
DISWDT	A5 5A A5 A5	4

DIV **Деление со знаком (16 разрядов/16 разрядов)**

Синтаксис DIV op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MDL) mod (opl)

Тип данных WORD

Описание Выполняется деление содержимого 16-разрядного приемника MDL (младшее слово 32-разрядного регистра MD) на содержимое 16-разрядного источника op1. Оба операнда рассматриваются как числа в дополнительном коде. Частное сохраняется в MDL, остаток сохраняется в MDH (старшем слове регистра MD).
Инструкция DIV выполняется 20 машинных циклов. DIV является прерываемой инструкцией. Если во время выполнения DIV произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю.
В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIV	Rw_n	4B nn	2

DIVL **Деление со знаком (32 разряда/16 разрядов)**

Синтаксис DIVL op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MD) mod (opl)

Тип данных WORD, DOUBLE WORD

Описание Выполняется деление содержимого 32-разрядного приемника MD на содержание 16-разрядного источника op1. Оба операнда рассматриваются как числа в дополнительном коде. Частное со знаком сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD). Инструкция DIVL выполняется 20 машинных циклов. DIVL является прерываемой инструкцией. Если во время выполнения DIVL произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVL	Rw_n	6B nn	2

DIVLU **Беззнаковое деление (32 разряда/16 разрядов)**

Синтаксис DIVLU op1

Действие (MDL) ← (MD) / (opl)
(MDH) ← (MD) mod (opl)

Тип данных WORD, DOUBLE WORD

Описание Выполняется беззнаковое деление содержимого 32-разрядного приемника MD на содержимое 16-разрядного источника op1. Частное сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD). Инструкция DIVLU выполняется 20 машинных циклов. DIVLU является прерываемой инструкцией. Если во время выполнения DIVLU произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVLU	Rw_n	7B nn	2

DIVU **Беззнаковое деление (16 разрядов/16разрядов)**

Синтаксис DIVU op1

Действие (MDL) ← (MDL) / (opl)
(MDH) ← (MDL) mod (opl)

Тип данных WORD

Описание Выполняется беззнаковое деление содержимого 16-разрядного приемника (младшее слово регистра MD) на содержимое 16-разрядного источника op1. Частное сохраняется в MDL (младшем слове регистра MD), остаток сохраняется в MDH (старшем слове регистра MD).
Инструкция DIVU выполняется 20 машинных циклов. DIVU является прерываемой инструкцией. Если во время выполнения DIVU произошло прерывание, то устанавливается бит MULIP в регистре PSW, и деление откладывается до выхода из обработчика прерывания.

Флаги состояния

E	Z	V	C	N
0	*	S	0	*

E Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

V Устанавливается, если произошло переполнение, т. е. результат не может быть представлен в указанном типе данных, или если содержимое делителя op1 было равно 0. В противном случае сбрасывается.

C Всегда сбрасывается.

N Устанавливается, если установлен старший значащий разряд результата. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Размер
DIVU	Rw_n	5B nn	2

EINIT **Конец инициализации**

Синтаксис EINIT

Действие Конец инициализации.

Описание После сброса на вывод микроконтроллера RSTOUT# выдается уровень логического 0, который сохраняется до выполнения EINIT, после чего выдается уровень логической 1. Это позволяет программе посылать внешним цепям микроконтроллера сигнал об успешной инициализации. После выполнения EINIT, инструкция запрещения сторожевого таймера DISWDT игнорируется. Инструкция EINIT используется для завершения инициализационной части программы. Чтобы избежать случайного выполнения, инструкция является защищенной.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника	Формат	Размер
EINIT	B5 4A B5 B5	4

EXTR **Переадресация регистров**

Синтаксис EXTR op1

Действие

```
(count) ← opl [ 1 ≤ opl ≤ 4 ]
IRQ_processing = Disable        // запрещение обработки
                                // запросов на прерывание
SFR_range = Extended  // перенаправление 8-битовой
                      // адресации в область ESFR
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
   ... // выполнение следующей инструкции
 (count) ← (count) - 1
END WHILE
(count) = 0
SFR_range = Standard  // отмена перенаправления
IRQ_processing = Enable       // разрешение обработки
                              // запросов на прерывание
```

Описание При использовании адресации "reg", "bitoff", "bitaddr" производится обращение к расширенной области регистров специального назначения. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTR распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOP. Количество инструкций, на которые распространяется действие EXTR, определяется op1 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTR прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется

Z Не изменяется

V Не изменяется

C Не изменяется

N Не изменяется

Формат инструкции

Мнемоника		Формат	Размер
EXTR	#irang2	D1 : 10 ## -0	4

EXTP **Страничная переадресация**

Синтаксис EXTP op1, op2

Действие

```
(count) ← op2 [ 1 ≤ opl ≤ 4 ]
IRQ_processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Page = (opl)
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) - 1
END WHILE
(count) = 0
Data_Page = (DPPx)
IRQ_Processing = Enable      // разрешение обработки
                             // запросов на прерывание
```

Описание Заменяется стандартная схема адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 10-битовый номер страницы (разряды адреса A23, ..., A14) определяется не содержимым соответствующего регистра DPPx, а значением op1.
14-битовое внутристраничное смещение (разряды адреса A13, ..., A0) определяется младшими разрядами адреса, указанного в инструкциях.
На время выполнения указанного количества инструкцией запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTP распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTP, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTP прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.

Z Не изменяется.

V Не изменяется.

C Не изменяется.

N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
EXTP	Rw_m, #irang2	DC : 01 ## -m	2
EXTP	$\#pag_{10}$, #irang2	D7 : 01 ## -0 PP 0:00PP	4

EXTPR **Страничная переадресация и переадресация регистров**

Синтаксис EXTPR op1, op2

Действие

```
(count) ← op2 [ 1 ≤ op2 ≤ 4 ]
IRQ_Processing = Disable      // запрещение обработки
                              // запросов на прерывание
Data_Page = (opl)
SFR_range = Extended      // перенаправление 8-битовой
                          // адресации в область ESFR
DO WHILE ( (count) ≠ 0 AND Class_B_IRQ ≠ TRUE)
     ... // выполнение следующей инструкции
(count) ← (count) - 1
END WHILE
(count) = 0
Data_Page = (DPPx)
SFR_range = Standard   // отмена перенаправления
IRQ_processing = Enable    // разрешение обработки
                           // запросов на прерывание
```

Описание При использовании адресации "reg", "bitoff", "bitaddr" производится обращение к расширенной области регистров специального назначения. Кроме того, заменяется стандартная схема адресации для режимов косвенной ([...]) и 16-битовой непосредственной адресаций (mem). При адресации 10-битовый номер страницы (разряды адреса A23, ..., A14) определяется не содержимым соответствующего регистра DPPx, а значением op1. 14-битовое внутристраничное смещение (разряды адреса A13, ..., A0) определяется младшими разрядами адреса, указанного в инструкциях. На время выполнения указанного количества инструкций запрещается стандартная обработка запросов на прерывания от периферии и запросов класса A, а также обработка запросов от периферии контроллером PEC. Действие EXTPR распространяется уже на следующую инструкцию, поэтому не требуется дополнительных инструкций NOR. Количество инструкций, на которые распространяется действие EXTPR, определяется op2 и принимает значение от 1 до 4 вне зависимости от количества требуемых циклов шины. Если во время выполнения указанного количества инструкций пришел запрос класса B, действие EXTPR прекращается, и осуществляется обработка запроса в соответствии со стандартной процедурой.

Флаги состояния

E	Z	V	C	N
-	-	-	-	-

E Не изменяется.
Z Не изменяется.
V Не изменяется.
C Не изменяется.
N Не изменяется.

Формат инструкции

Мнемоника		Формат	Размер
EXTPR	Rw_m, #irang2	DC : 11 ## -m	2
EXTPR	#pag10, #irang2	D7 : 11 ## -0 PP 0:00PP	4

EXTS **Сегментная переадресация**

Синтаксис EXTS op1

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) Система команд блока 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
((IDXi (- ⊗))) ← ((IDXi))
```

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее суммируется с 40-битным содержимым аккумулятора ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACM	[IDX_i⊗], [Rw_m⊗]	93 Xm D8 rrrr:rqqq	Да

CoMACM- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACM- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
   (tmp) ← ((op1) * (op2)) << 1
   (ACC) ← (ACC) - (tmp)
ELSE
   (tmp) ← (op1) * (op2)
   (ACC) ← (ACC) - (tmp)
END IF
((IDXi (- ⊗))) ← ((IDXi))
```

Описание Умножение двух операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее вычитается из 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACM-	[IDX_i⊗], [Rw_m⊗]	93 Xm E8 rrrr:rqqq	Да

CoMACMR **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMR op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← (((op1)) * ((op2))) << 1
(ACC) ← (tmp) - (ACC) + 0000008000_H
ELSE
(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
END IF
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее из него вычитается с 40-битный аккумулятор ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMR	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	F9	rrrr:rqqq	Да

CoMACMR **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(tmp) ← (((op1)) * ((op2))) << 1
(ACC) ← (tmp) - (ACC)
ELSE
(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
END IF
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMR	[IDX_i⊗], [Rw_m⊗]	93 Xm F8 rrrr:rqqq	Да

CoMACMRsu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRsu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 79 rrrr:rqqq	Да

CoMACMRsu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 78 rrrr:rqqq	Да

CoMACMRu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 39 rrrr:rqqq	Да

CoMACMRu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRu	[IDX_i⊗], [Rw_m⊗]	93 Xm 38 rrrr:rqqq	Да

CoMACMRus **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm B9 rrrr:rqqq	Да

CoMACMRus **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMRus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из него вычитается значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMRus	[IDX_i⊗], [Rw_m⊗]	93 Xm B8 rrrr:rqqq	Да

CoMACMsu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMsu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 59 rrrr:rqqq	Да

CoMACMsu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMsu	[IDX_i⊗], [Rw_m⊗]	93	Xm	58	rrrr:rqqq	Да

CoMACMsu- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается. ACC → 40-битная величина со знаком

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 68 rrrr:rqqq	Да

CoMACMu **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	19	rrrr:rqqq	Да

CoMACMu **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu	[IDX_i⊗], [Rw_m⊗]	93	Xm	18	rrrr:rqqq	Да

CoMACMu- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMu-	[IDX_i⊗], [Rw_m⊗]	93	Xm	28	rrrr:rqqq	Да

CoMACMus **Умножение-накопление, пересылка, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 99 rrrr:rqqq	Да

CoMACMus **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) + (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее к нему добавляется значение 40-битного регистра ACC. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACMus	[IDX_i⊗], [Rw_m⊗]	93	Xm	98	rrrr:rqqq	Да

CoMACMus- **Умножение-накопление, пересылка**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACMus- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

(tmp) ← ((op1)) * ((op2))
(ACC) ← (ACC) - (tmp)
((IDX_i (- ⊗))) ← ((IDX_i))

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком, далее из 40-битного регистра ACC вычитается полученное произведение. Затем полученный результат записывается в регистр ACC. Параллельно с выполнением арифметической команды и двумя параллельными чтениями памяти данные, хранящиеся по адресу, указанному в регистре IDXi, пересылаются в другую ячейку памяти DPRAM. Адрес, по которому пересылаются данные, зависит от значения смещения для указателя адреса IDXi.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACMus-	[IDX_i⊗], [Rw_m⊗]	93 Xm A8 rrrr:rqqq	Да

CoMACR **Умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
  (tmp) ← ((op1) * (op2)) << 1
  (ACC) ← (tmp) - (ACC) + 0000008000_H
ELSE
  (tmp) ← (op1) * (op2)
  (ACC) ← (tmp) - (ACC) + 0000008000_H
END IF
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее, если установлен флаг MP, полученное произведение сдвигается на 1 бит влево, а затем из него вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACR	Rw_n, Rw_m, rnd	A3 nm F1 00	Нет
CoMACR	Rw_n, [Rw_m⊗], rnd	83 nm F1 rrrr:rqqq	Да
CoMACR	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm F1 rrrr:rqqq	Да

CoMACR **Умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
   (tmp) ← ((op1) * (op2)) << 1
   (ACC) ← (tmp) - (ACC)
ELSE
   (tmp) ← (op1) * (op2)
   (ACC) ← (tmp) - (ACC)
END IF
```

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее, если установлен флаг MP, полученное произведение сдвигается на 1 бит влево, а затем из него вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACR	Rw_n, Rw_m	A3	nm	F0	00	Нет
CoMACR	Rw_n, [Rw_m⊗]	83	nm	F0	rrrr:rqqq	Да
CoMACR	[IDX_i⊗], [Rw_m⊗]	93	Xm	F0	rrrr:rqqq	Да

CoMACRsu **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRsu	Rw_n, Rw_m, rnd	A3	nm	71	00	Нет
CoMACRsu	Rw_n, [Rw_m⊗], rnd	83	nm	71	rrrr:rqqq	Да
CoMACRsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	71	rrrr:rqqq	Да

CoMACRsu **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACRsu	Rw_n, Rw_m	A3 nm 70 00	Нет
CoMACRsu	Rw_n, [Rw_m⊗]	83 nm 70 rrrr:rqqq	Да
CoMACRsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 70 rrrr:rqqq	Да

CoMACRu **Умножение-накопление без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRu	Rw_n, Rw_m, rnd	A3	nm	31	00	Нет
CoMACRu	Rw_n, [Rw_m⊗], rnd	83	nm	31	rrrr:rqqq	Да
CoMACRu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	31	rrrr:rqqq	Да

CoMACRu **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACR op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат без знака дополняется нулями. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACRu	Rw_n, Rw_m	A3 nm 30 00	Нет
CoMACRu	Rw_n, [Rw_m⊗]	83 nm 30 rrrr:rqqq	Да
CoMACRu	[IDX_i⊗], [Rw_m⊗]	93 Xm 30 rrrr:rqqq	Да

CoMACRus **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRus	Rw_n, Rw_m, rnd	A3	nm	B1	00	Нет
CoMACRus	Rw_n, [Rw_m⊗], rnd	83	nm	B1	rrrr:rqqq	Да
CoMACRus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	B1	rrrr:rqqq	Да

CoMACRus **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACRus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее из полученного произведения вычитается значение регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Дополнение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACRus	Rw_n, Rw_m, rnd	A3	nm	B0	00	Нет
CoMACRus	Rw_n, [Rw_m⊗], rnd	83	nm	B0	rrrr:rqqq	Да
CoMACRus	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	B0	rrrr:rqqq	Да

CoMACsu **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACsu	Rw_n, Rw_m, rnd	A3 nm 51 00	Нет
CoMACsu	Rw_n, [Rw_m⊗], rnd	83 nm 51 rrrr:rqqq	Да
CoMACsu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 51 rrrr:rqqq	Да

CoMACsu **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACsu	Rw_n, Rw_m	A3 nm 50 00	Нет
CoMACsu	Rw_n, [Rw_m⊗]	83 nm 50 rrrr:rqqq	Да
CoMACsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 50 rrrr:rqqq	Да

CoMACsu- **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACsu-	Rw_n, Rw_m	A3 nm 60 00	Нет
CoMACsu-	Rw_n, [Rw_m⊗]	83 nm 60 rrrr:rqqq	Да
CoMACsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 60 rrrr:rqqq	Да

CoMACu **Умножение-накопление без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACu	Rw_n, Rw_m, rnd	A3 nm 11 00	Нет
CoMACu	Rw_n, [Rw_m⊗], rnd	83 nm 11 rrrr:rqqq	Да
CoMACu	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 11 rrrr:rqqq	Да

CoMACu **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACu	Rw_n, Rw_m, rnd	A3	nm	10	00	Нет
CoMACu	Rw_n, [Rw_m⊗], rnd	83	nm	10	rrrr:rqqq	Да
CoMACu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	10	rrrr:rqqq	Да

CoMACu- **Умножение-накопление без знака**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Полученный 32-битный результат со знаком дополняется нулями. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение есть
*	*	*	*	*	*	

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMACu-	Rw_n, Rw_m, rnd	A3	nm	20	00	Нет
CoMACu-	Rw_n, [Rw_m⊗], rnd	83	nm	20	rrrr:rqqq	Да
CoMACu-	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	20	rrrr:rqqq	Да

CoMACus **Смешанное умножение-накопление, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus	Rw_n, Rw_m, rnd	A3 nm 91 00	Нет
CoMACus	Rw_n, [Rw_m⊗], rnd	83 nm 91 rrrr:rqqq	Да
CoMACus	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm 91 rrrr:rqqq	Да

CoMACus **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) + (tmp)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение суммируется со значением 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое переполнение. В противном случае не изменяется.

C Устанавливается, если произошел перенос. В противном случае не изменяется.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus	Rw_n, Rw_m	A3 nm 90 00	Нет
CoMACus	Rw_n, [Rw_m⊗]	83 nm 90 rrrr:rqqq	Да
CoMACus	[IDX_i⊗], [Rw_m⊗]	93 Xm 90 rrrr:rqqq	Да

CoMACus- **Смешанное умножение-накопление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMACus- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Описание Умножение двух 16-битных операндов источника op1 (без знака) и op2 (со знаком). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Далее полученное произведение вычитается из значения 40-битного регистра ACC. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	*	*	*	*	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Устанавливается, если произошло арифметическое опустошение. В противном случае не изменяется.

C Устанавливается, если произошел заем. В противном случае сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMACus-	Rw_n, Rw_m	A3 nm A0 00	Нет
CoMACus-	Rw_n, [Rw_m⊗]	83 nm A0 rrrr:rqqq	Да
CoMACus-	[IDX_i⊗], [Rw_m⊗]	93 Xm A0 rrrr:rqqq	Да

CoMAX **Максимальное значение**

Группа Команды сравнения

Синтаксис CoMAX op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← max ((ACC) , (tmp))

Описание Сравнение 40-битного операнда со знаком со значением регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Если содержимое регистра ACC меньше, чем 40-битный операнд, то в регистр ACC записывается значение 40-битного операнда. В противном случае содержимое регистра ACC не изменяется. Установка бита MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	нет

SL Устанавливается, если содержимое аккумулятора ACC изменилось. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMAX	Rw_n, Rw_m	A3 nm 3A 00	Нет
CoMAX	Rw_n, [Rw_m⊗]	83 nm 3A rrrr:rqqq	Да
CoMAX	[IDX_i⊗], [Rw_m⊗]	93 Xm 3A rrrr:rqqq	Да

CoMIN **Минимальное значение**

Группа Команды сравнения

Синтаксис CoMIN op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие
(tmp) ← (op2) || (op1)
(ACC) ← min ((ACC) , (tmp))

Описание Сравнение 40-битного операнда со знаком со значением регистра ACC. 40-битный операнд образуется в результате конкатенации двух операндов источника op1 (LSW) и op2 (MSW) и расширения знака образовавшегося числа. Если содержимое регистра ACC больше, чем 40-битный операнд, то в регистр ACC записывается значение 40-битного операнда. В противном случае содержимое регистра ACC не изменяется. Установка бита MS регистра MCW не влияет на результат.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	нет

SL Устанавливается, если содержимое аккумулятора ACC изменилось. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMIN	Rw_n, Rw_m	A3 nm 7A 00	Нет
CoMIN	Rw_n, [Rw_m⊗]	83 nm 7A rrrr:rqqq	Да
CoMIN	[IDX_i⊗], [Rw_m⊗]	93 Xm 7A rrrr:rqqq	Да

CoMOV **Пересылка из области памяти в область памяти**

Группа Команды пересылки данных

Синтаксис CoMOV op1, op2

Операнд(ы) источника op2 → WORD

Операнд(ы) приемника op1 → WORD

Действие (op1) ← (op2)

Описание Пересылка содержимого из области памяти, определяемой операндом источника op2, в область памяти, определяемой операндом приемника op1. Примечательно то, что в отличие от других команд, IDXi может адресовать всю область памяти. Данная команда не влияет на флаги блока MAC, но устанавливает флаги ЦПУ, как любая другая команда пересылки MOV.

Флаги состояния ЦПУ

E	Z	V	C	N
*	*	-	-	*

E Устанавливается, если содержимое op2 равно наименьшему отрицательному числу. В противном случае сбрасывается.

Z Устанавливается, если содержимое op2 равно нулю. В противном случае сбрасывается.

V Не изменяется.

C Не изменяется.

N Устанавливается, если установлен старший значащий разряд операнда источника op2. В противном случае сбрасывается.

Примечание – CoMOV – единственная команда блока MAC, которая изменяет флаги ЦПУ. Флаги MAC не изменяются.

Формат инструкции

Мнемоника		Формат	Повтор
CoMOV	[IDX_i⊗], [Rw_m⊗]	D3 Xm 00 rrrr:rqqq	Да

CoMUL **Умножение со знаком, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

IF (MP = 1) THEN
(ACC) ← ((op1) * (op2)) << + 0000008000_H
ELSE
(ACC) ← (op1) * (op2) + 0000008000_H
END IF
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат в дополнительном коде округляется и записывается в регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Не изменяется, когда биты MP или MS равны нулю. В противном случае устанавливается в случае умножения 8000_H на 8000_H.

E Устанавливается, если бит MP равен единице, а MS равен нулю и в случае умножения 8000_H на 8000_H. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMUL	Rw_n, Rw_m, rnd	A3 nm C1 00	Нет
CoMUL	Rw_n, [Rw_m⊗], rnd	83 nm C1 0:0qqq	Нет
CoMUL	[IDX_i⊗], [Rw_m⊗], rnd	93 Xm C1 0:0qqq	Нет

CoMUL **Умножение со знаком**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (ACC) ← ((op1) * (op2))
ELSE
  (ACC) ← (op1) * (op2)
END IF
```

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат записывается в регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	*	есть

SL Не изменяется, когда биты MP или MS равны нулю. В противном случае устанавливается в случае умножения 8000_H на 8000_H.

E Устанавливается, если бит MP равен единице, а MS равен нулю и в случае умножения 8000_H на 8000_H. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMUL	Rw_n, Rw_m	A3 nm C0 00	Нет
CoMUL	Rw_n, [Rw_m⊗]	83 nm C0 0:0qqq	Нет
CoMUL	[IDX_i⊗], [Rw_m⊗]	93 Xm C0 0:0qqq	Нет

CoMUL- **Умножение со знаком**

Группа Команды умножения/умножения-накопления

Синтаксис CoMUL- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие

```
IF (MP = 1) THEN
  (ACC) ← - ((op1) * (op2))
ELSE
  (ACC) ← - ((op1) * (op2))
END IF
```

Описание Умножение двух 16-битных операндов источника со знаком op1 и op2. Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем, если флаг MP установлен, то произведение сдвигается на 1 бит влево. Затем полученный результат с противоположным знаком записывается в регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMUL-	Rw_n, Rw_m	A3 nm C8 00	Нет
CoMUL-	Rw_n, [Rw_m⊗]	83 nm C8 0:0qqq	Нет
CoMUL-	[IDX_i⊗], [Rw_m⊗]	93 Xm C8 0:0qqq	Нет

CoMULsu **Смешанное умножение, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат в дополнительном коде округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат				Повтор
CoMULsu	Rw_n, Rw_m, rnd	A3	nm	41	00	Нет
CoMULsu	Rw_n, [Rw_m⊗], rnd	83	nm	41	0:0qqq	Нет
CoMULsu	[IDX_i⊗], [Rw_m⊗], rnd	93	Xm	41	0:0qqq	Нет

CoMULsu **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2)

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMULsu	Rw_n, Rw_m	A3 nm 40 00	Нет
CoMULsu	Rw_n, [Rw_m⊗]	83 nm 40 0:0qqq	Нет
CoMULsu	[IDX_i⊗], [Rw_m⊗]	93 Xm 40 0:0qqq	Нет

CoMULsu- **Смешанное умножение**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULsu- op1, op2

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← - ((op1) * (op2))

Описание Умножение двух 16-битных операндов источника op1 (со знаком) и op2 (без знака). Сначала происходит расширение знака получившегося 32-битного числа со знаком. Затем полученный результат с противоположным знаком записывается в 40-битный регистр ACC.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
-	0	-	0	*	*	нет

SL Не изменяется.

E Всегда сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Устанавливается, если значащий бит результата равен единице. В противном случае сбрасывается.

Формат инструкции

Мнемоника		Формат	Повтор
CoMULsu-	Rw_n, Rw_m,	A3 nm 48 00	Нет
CoMULsu-	Rw_n, [Rw_m⊗],	83 nm 48 0:0qqq	Нет
CoMULsu-	[IDX_i⊗], [Rw_m⊗]	93 Xm 48 0:0qqq	Нет

CoMULu **Умножение без знака, округление**

Группа Команды умножения/умножения-накопления

Синтаксис CoMULu op1, op2, rnd

Операнд(ы) источника op1, op2 → WORD

Операнд(ы) приемника ACC → 40-битная величина со знаком

Действие (ACC) ← (op1) * (op2) + 0000008000_H
(MAL) ← 0

Описание Умножение двух 16-битных операндов источника без знака op1 и op2. Сначала происходит дополнение нулями получившегося 32-битного числа без знака. Затем полученный результат округляется и записывается в 40-битный регистр ACC. Регистр MAL обнуляется.

Флаги состояния

SL	E	SV	C	Z	N	Ограничение
*	*	-	0	*	0	есть

SL Устанавливается, если содержимое аккумулятора ACC автоматически ограничено. В противном случае не изменяется.

E Устанавливается, если используется MAE. В противном случае сбрасывается.

SV Не изменяется.

C Всегда сбрасывается.

Z Устанавливается, если результат равен нулю. В противном случае сбрасывается.

N Всегда сбрасывается.

Примечание – Поведение флага E и флага SV изменилось, чтобы гарантировать арифметическую корректность. Если умножаются два больших 16-битных числа без знака, то результат не может быть представлен в 32-битном формате со знаком. В этом случае следует использовать как расширение ACC (автоматическое ограничение запрещено, MS = 0), так и ограничение до 32-битного значения со знаком (автоматическое ограничение разрешено, MS = 1).

Формат инструкции

Мнемоника		Формат	Повтор
CoMULu	Rw_n, Rw_m, rnd	A3 nm 01 00	Нет
CoMULu	Rw_n, [$Rw_m\otimes$], 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, $[Rw_m^{\otimes}]$	A3	nm	0A	00	Нет
CoSUB	Rw_n, $[Rw_m^{\otimes}]$	83	nm	0A	rrrr:rqqq	Да
CoSUB	$[IDX_i^{\otimes}]$, $[Rw_m^{\otimes}]$	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, [$Rw_m^{\otimes}$]	A3	nm	4A	00	Нет
CoSUB2	Rw_n, [$Rw_m^{\otimes}$]	83	nm	4A	rrrr:rqqq	Да
CoSUB2	[$IDX_i^{\otimes}$], [$Rw_m^{\otimes}$]	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, $[Rw_m^{\otimes}]$	A3 nm 52 00	Нет
CoSUB2R	Rw_n, $[Rw_m^{\otimes}]$	83 nm 52 rrrr:rqqq	Да
CoSUB2R	$[IDX_i^{\otimes}]$, $[Rw_m^{\otimes}]$	93 Xm 52 rrrr:rqqq	Да
CoSUB	$[IDX_i^{\otimes}]$, $[Rw_m^{\otimes}]$	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, $[Rw_m^{\otimes}]$	A3	nm	12	00	Нет
CoSUBR	Rw_n, $[Rw_m^{\otimes}]$	83	nm	12	rrrr:rqqq	Да
CoSUBR	$[IDX_i^{\otimes}]$, $[Rw_m^{\otimes}]$	93	Xm	12	rrrr:rqqq	Да
CoSUB	$[IDX_i^{\otimes}]$, $[Rw_m^{\otimes}]$	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 Е.2

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	
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	
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 Е.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 Е.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

Для конфигурирования микроконтроллера 1887BE9T при внешнем сбросе (EA# = 0b) используются выводы порта P0 и выводы ALE, RD# и EA#, состояние которых должно быть установлено во время активного уровня сигнала сброса RSTIN#. После появления положительного фронта сигнала RSTIN# необходимо удерживать комбинацию сигналов на выводах порта P0 не менее 20 периодов XTAL1, чтобы конфигурация прошла успешно.

На рисунке Ж.1 представлены выводы порта P0 и регистры, состояние которых они изменяют.

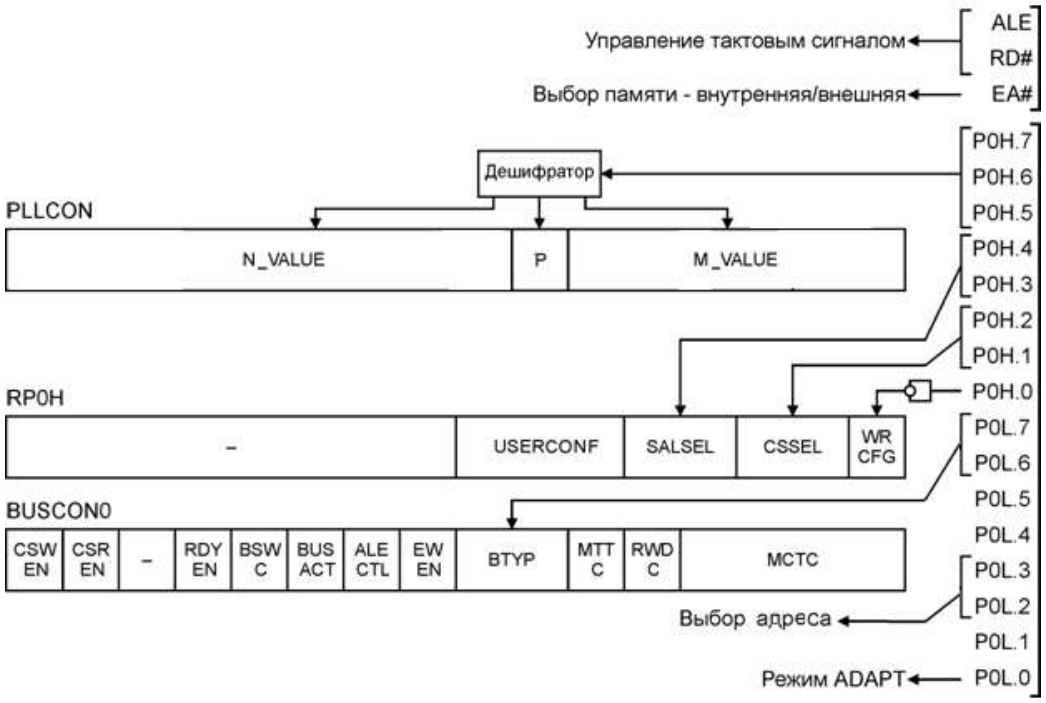


Рисунок Ж.1 – Порт P0 и его взаимодействие с регистрами микроконтроллера при конфигурировании

В таблице Ж.1 указано назначение выводов порта P0 и выводов ALE, RD# и EA# при конфигурировании микроконтроллера.

Таблица Ж.1 – Функциональное назначение выводов микроконтроллера при конфигурировании

<table>
<tr><th>Обозначение вывода</th><th>Функциональное назначение выводов при конфигурации микроконтроллера</th></tr>
<tr><td>P0H.7,
P0H.6,
P0H.5</td><td>Устанавливают коэффициент умножения входной частоты.
В зависимости от комбинации сигналов на выводах P0H.7 – P0H.5 в битовые поля регистра PLLCON записываются значения, соответствующие выбранному коэффициенту умножения входной частоты.
<table>
<tr><th colspan="3">Состояние выводов порта P0</th><th colspan="2">Поля регистра PLLCON</th><th rowspan="2">Частота на выходе блока PLL</th></tr>
<tr><th>P0H.7</th><th>P0H.6</th><th>P0H.5</th><th>N_VALUE</th><th>M_VALUE</th></tr>
<tr><td>1</td><td>1</td><td>1</td><td rowspan="8">02h</td><td>10h</td><td>$f_{OSC} \times 8$</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0Ch</td><td>$f_{OSC} \times 6$</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0Ah</td><td>$f_{OSC} \times 5$</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>07h</td><td>$f_{OSC} \times 3{,}5$</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>06h</td><td>$f_{OSC} \times 3$</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>05h</td><td>$f_{OSC} \times 2{,}5$</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>04h</td><td>$f_{OSC} \times 2$</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>03h</td><td>$f_{OSC} \times 1{,}5$</td></tr>
</table></td></tr>
<tr><td>P0H.4,
P0H.3</td><td>Устанавливают количество разрядов старшей части адреса, используемых для адресации по внешней шине.
С выводов P0H.4, P0H.3 считывается код, определяющий разрядность.
Код записывается в битовое поле SALSEL регистра RP0H.
<table>
<tr><th colspan="2">SALSEL</th><th>Количество разрядов</th><th>Сегментная часть адреса</th><th>Адресное пространство</th></tr>
<tr><td>0</td><td>0</td><td>4</td><td>A19, A18, A17, A16</td><td>1 Мбайт</td></tr>
<tr><td>0</td><td>1</td><td>–</td><td>–</td><td>64 Кбайт</td></tr>
<tr><td>1</td><td>0</td><td>8</td><td>A23, A22, A21, A20, A19, A18, A17, A16</td><td>16 Мбайт</td></tr>
<tr><td>1</td><td>1</td><td>2</td><td>A17, A16</td><td>256 Кбайт</td></tr>
</table></td></tr>
<tr><td>P0H.2,
P0H.1</td><td>Устанавливают количество используемых каналов CS#.
С выводов P0H.2, P0H.1 считывается код, определяющий количество каналов. Код записывается в битовое поле CSSEL регистра RP0H.
<table>
<tr><th colspan="2">CSSEL</th><th>Количество каналов</th><th>Действующие каналы порта P6</th><th>Свободные каналы порта P6</th></tr>
<tr><td>0</td><td>0</td><td>3</td><td>CS0#, CS1#, CS2#</td><td>3, 4, 5, 6, 7</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>CS0#, CS1#</td><td>2, 3, 4, 5, 6, 7</td></tr>
<tr><td>1</td><td>0</td><td>–</td><td>–</td><td>Все</td></tr>
<tr><td>1</td><td>1</td><td>5</td><td>CS0#, CS1#, CS2#, CS3#, CS4#</td><td>5, 6, 7 (без pull-down резисторов)</td></tr>
</table></td></tr>
</table>

Окончание таблицы Ж.1

<table>
<tr><th>Обозначение вывода</th><th>Функциональное назначение выводов при конфигурации микроконтроллера</th></tr>
<tr><td>P0H.0</td><td>Конфигурирует работу сигнала записи.
Значение, считанное с вывода P0H.0, записывается в инверсном виде в бит WRCFG регистра RP0H</td></tr>
<tr><td>P0L.7,
P0L.6</td><td>Конфигурируют внешнюю шину.
С выводов P0L.7, P0L.6 считывается код, определяющий разрядность и тип шины. Код записывается в битовое поле BUSTYP регистра BUSCON0.
<table>
<tr><th colspan="2">BUSTYP</th><th>Разрядность и тип шины</th></tr>
<tr><td>0</td><td>0</td><td>8-разрядная немультиплексная шина</td></tr>
<tr><td>0</td><td>1</td><td>8-разрядная мультиплексная шина</td></tr>
<tr><td>1</td><td>0</td><td>16-разрядная немультиплексная шина</td></tr>
<tr><td>1</td><td>1</td><td>16-разрядная мультиплексная шина</td></tr>
</table></td></tr>
<tr><td>P0L.5,
P0L.4</td><td>Не участвуют в конфигурации</td></tr>
<tr><td>P0L.3,
P0L.2</td><td>Задают стартовый адрес.
С выводов P0L.3, P0L.2 считывается код, определяющий адрес памяти с которого стартует микроконтроллер.
<table>
<tr><th colspan="2">Состояние выводов порта P0</th><th rowspan="2">Стартовый адрес</th></tr>
<tr><th>P0L.3</th><th>P0L.2</th></tr>
<tr><td>0</td><td>0</td><td>08’0000h (8 сегмент)</td></tr>
<tr><td>0</td><td>1</td><td>02’0000h (2 сегмент</td></tr>
<tr><td>1</td><td>0</td><td>01’0000h (1 сегмент)</td></tr>
<tr><td>1</td><td>1</td><td>00’0000h (0 сегмент)</td></tr>
</table></td></tr>
<tr><td>P0L.1</td><td>Не участвует в конфигурации</td></tr>
<tr><td>P0L.0</td><td>Выбирает режим электрической изоляции ADAPT.
Считывание «0» с вывода P0L.0 включает режим ADAPT.
Считывание «1» с вывода P0L.0 выключает режим ADAPT</td></tr>
<tr><td>ALE,
RD#</td><td>Управляют тактовым сигналом микроконтроллера
<table>
<tr><th>ALE</th><th>RD#</th><th>Режим работы микроконтроллера</th></tr>
<tr><td>0</td><td>0</td><td>без PLL</td></tr>
<tr><td>1</td><td>1</td><td>с PLL</td></tr>
</table></td></tr>
<tr><td>ALE,
RD#</td><td></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
	измененных	замененных	новых	аннулированных				
-	-	-	все	-	626			25.12.15
1	1	-	-	-	-			28.12.15
2	-	7	-	-	-			12.10.16
3	-	5, 288	-	-	-			06.04.17