МИКРОСХЕМЫ ИНТЕГРАЛЬНЫЕ

1874ВЕ10Т, 1874ВЕ10АТ

Руководство пользователя

2017

Содержание

Введение

В настоящем техническом описании КФДЛ.431295.059ТО приведено описание архитектуры, функционального построения, системы команд и особенностей применения микросхем 1874ВЕ10Т, 1874ВЕ10АТ. Микросхемы представляют собой СБИС спецстойкого однокристального 16-разрядного микроконтроллера с тактовой частотой до 66 МГц, с двумя регистровыми ОЗУ общим объемом 32 Кбайт, внутренним ОЗУ команд объемом 4 Кбайт, модулем отладки с блоком JTAG и сторожевым таймером. Доступное адресное пространство 4 Гбайт.

Периферия микроконтроллеров представлена блоками ШИМ (шесть каналов), импульсными квадратурными декодерами, последовательными портами ввода-вывода типа UART, блоком высокоскоростного ввода-вывода HSIO, контроллерами интерфейсов SPI, I2C, SpaceWire, ARINC 429, МПИ (магистральный последоватльный интерфейс, реализованный по ГОСТ Р 52070-2003), 16-канальным 14-разрядным АЦП с возможностью работы с дифференциальными сигналами и восемью 8-разрядными параллельными портами ввода-вывода для взаимодействия с внешними устройствами.

В состав микроконтроллеров также входит блок вычислений с плавающей точкой. Для связи с внешней памятью реализована динамически конфигурируемая (8-/16-/32-разрядная) шина адреса/данных с защитой кодом Хэмминга.

Изделия служат цели развития отечественной элементной базы для применения в различных системах управлениях, радиосвязи, радиолокации и других изделиях, требующих точных аналогово-цифровых преобразований в условиях радиационного воздействия.

Настоящее техническое описание может служить практическим руководством по применению микроконтроллера для разработчиков систем на основе ИС 1874ВЕ10Т, 1874ВЕ10АТ.

1 Особенности микроконтроллеров и области применения

В настоящее время отечественной промышленностью выпускается несколько 16-разрядных микроконтроллеров серии 1874 на базе процессорного ядра с архитектурой MCS-96. Данные микроконтроллеры имеют хорошие электрические параметры и обеспечивают высокие требования по устойчивости к механическим и климатическим воздействиям, однако уровень их радиационной стойкости недостаточен для аппаратуры космического назначения и ядерных объектов.

Высокопроизводительное процессорное ядро микроконтроллеров 1874BE10T, 1874BE10AT реализуют полную поддержку системы команд MCS-96, но с ускоренным выполнением, благодаря наличию RISC подсистемы, и поддержку новых инструкций работы с 32-разрядными данными.

Это позволяет до восьми раз ускорить выполнение арифметических, логических и других операций по сравнению с базовой архитектурой MCS-96. Новый блок интерфейса к внешней памяти позволяет организовывать подключение к микросхемам ОЗУ и ПЗУ по гибконастраиваемой шине адреса/данных с поддержкой механизма исправления ошибок.

Поддержка большого количества современных интерфейсов (в том числе SPI, ГОСТ Р 52070-2003, SpaceWire, ARINC 429), позволяет не только организовывать каналы обмена между различными микроконтроллерами, но и управлять микросхемами внешних периферийных устройств.

Перспективными направлениями использования микросхем являются средства радиосвязи, системы управления, сбора, передачи и обработки данных, в том числе и на объектах атомной энергетики, системы управления бортовой аппаратуры самолетов и вертолетов, обслуживаемых и необслуживаемых космических аппаратов, ракет-носителей.

Микросхемы обеспечат предприятиям-разработчикам современных образцов как гражданской техники, так и вооружения, военной и специальной техники реализацию более высокого уровня тактико-технических характеристик в целом и высокую надежность при эксплуатации в условиях специальных видов воздействий.

Применение микросхем позволит решить вопросы комплектования радиоэлектронной аппаратуры гражданского, военного и специального назначения отечественными изделиями взамен зарубежных аналогичного класса.

Прямые отечественные и зарубежные аналоги микросхем 1874BE10T, 1874BE10AT отсутствуют.

Ближайший зарубежный аналог – UT80C196KDS ф. Aeroflex, США.

.

2 Краткое техническое описание микроконтроллеров

2.1 Функциональные параметры

Структурная схема микроконтроллеров показана на рисунке 2.1.

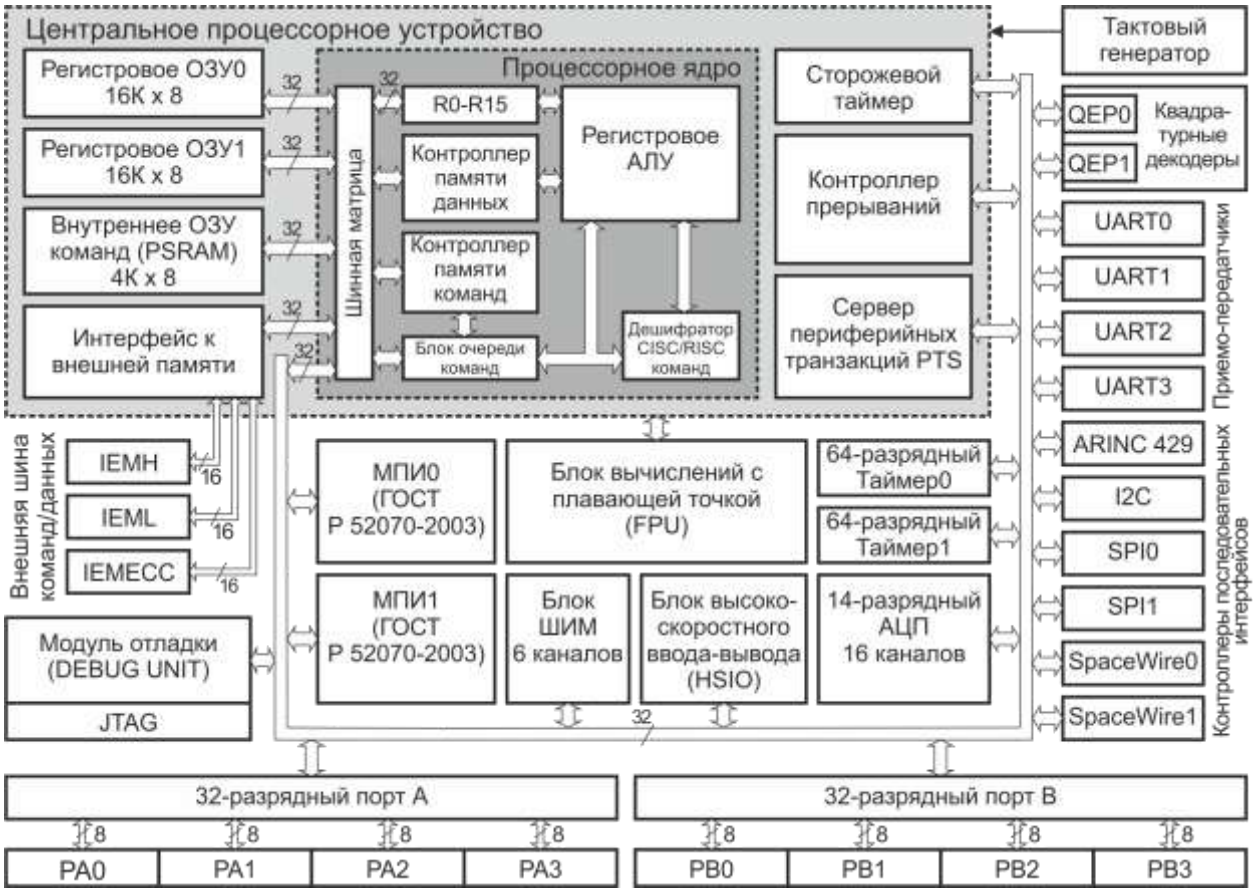


Рисунок 2.1 – Структурная схема микроконтроллеров

В состав микроконтроллеров входят функциональные элементы (см. рисунок 2.1):

- микропроцессорное ядро системы команд MCS-96 с поддержкой однотактных 8-/16-/32-разрядных RISC команд и дополнительных команд для работы с 32-разрядными данными с тактовой частотой до 66 МГц;
- блок вычислений с плавающей точкой (FPU);
- восемь параллельных 8-разрядных портов ввода-вывода (PA0, PA1, PA2, PA3, PB0, PB1, PB2, PB3);
- два регистровых ОЗУ (ОЗУ0, ОЗУ1) объемом по 16 Кбайт каждое;
- внутреннее ОЗУ команд объемом 4 Кбайт (PSRAM);
- доступное адресное пространство 4 Гбайт;
- динамически конфигурируемая 8-/16-/32-разрядная шина данных с защитой кодом Хэмминга (IEMH, IEML, IEMECC);
- контроллер прерываний;
- сервер периферийных транзакций (PTS);
- два 64-разрядных таймера/счетчика (T0, T1);
- 16-канальный 14-разрядный аналого-цифровой преобразователь;
- три двухканальных блока ШИМ (блок ШИМ0, блок ШИМ1, блок ШИМ2);
- блок высокоскоростного ввода-вывода (HSIO);
- два блока импульсных квадратурных декодеров (QEP0, QEP1);

- четыре последовательных приемо-передатчика UART (UART0, UART1, UART2, UART3);
- контроллеры интрефейсов:
  - ARINC 429;
  - I2C;
  - SPI (SPI0, SPI1);
  - SpaceWire (SpaceWire0, SpaceWire1);
  - МПИ (ГОСТ Р 52070-2003) с отдельными входами задания адреса (МПИ0, МПИ1);
- модуль отладки DEBUG UNIT с доступом через JTAG;
- блок JTAG;
- программируемый 32-разрядный сторожевой таймер;
- три режима пониженного энергопотребления (Idle, PowerDown, Slow).

Условное графическое обозначение микроконтроллеров приведено на рисунке 2.2.

Функциональное назначение выводов указано в таблицах 2.1 и 2.2. В таблице 2.1 в графе «Альтернативная функция вывода» функции указаны согласно их порядковому номеру, т. е. первая, вторая, третья. Выбор альтернативной функции осуществляется независимо для каждого вывода посредством соответствующего поля регистра ALTFSEL (состоит из регистров ALTFSEL0 и ALTFSEL1).

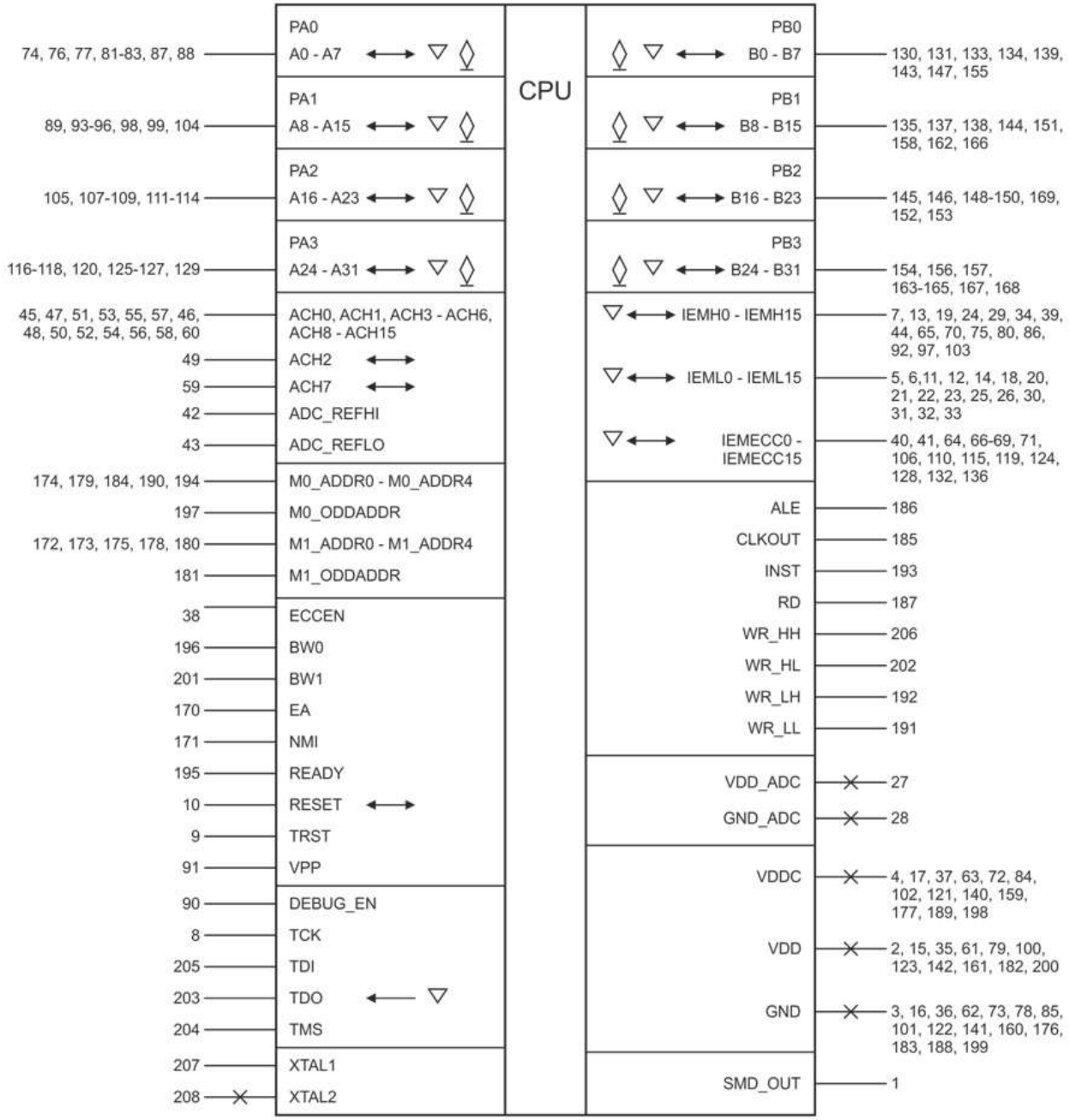
В графе «Номер вывода» указывается номер вывода микроконтроллера.

В графе «Тип вывода» используются обозначения: I – вход, O – выход, I/O – вход/выход, Z – третье состояние.

После сброса микроконтроллера выводы портов PA0, PA1, PA2, PA3, PB0, PB1, PB2, PB3 конфигурируются как выводы общего назначения и находятся в третьем состоянии.

Микросхемы выполнены в металлокерамическом корпусе МК 4250.208-1.

Масса микросхем – не более 20 г.



Примечание – Выводы A0 – A31 и B0 – B31 имеют альтернативные функции, которые указаны в таблице 2.1.

Рисунок 2.2 – Условное графическое обозначение микросхем 1874BE10T, 1874BE10AT

Таблица 2.1 – Функциональное назначение выводов микросхем 1874BE10T и 1874BE10AT, имеющих альтернативные функции

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
A0[1)]		74	I/O/Z/2	Вход/выход порта A0, разряд 0
	CS0		O	Выход нулевого сигнала выбора внешнего устройства
	ARINC_D4		I/O	Вход/выход четвертой линии данных
	HSO0		O/Z	Выход данных, канал 0
A1[1)]		76	I/O/Z/2	Вход/выход порта A0, разряд 1
	CS1		O	Выход первого сигнала выбора внешнего устройства
	ARINC_S4		I/O	Вход/выход четвертой селекторной линии
	HSO1		O/Z	Выход данных, канал 1
A2[1)]		77	I/O/Z/2	Вход/выход порта A0, разряд 2
	CS2		O	Выход второго сигнала выбора внешнего устройства
	ARINC_D5		I/O	Вход/выход пятой линии данных
	HSO2		O/Z	Выход данных, канал 2
A3[1)]		81	I/O/Z/2	Вход/выход порта A0, разряд 3
	CS3		O	Выход третьего сигнала выбора внешнего устройства
	ARINC_S5		I/O	Вход/выход пятой селекторной линии
	HSO3		O/Z	Выход данных, канал 3
A4		82	I/O/Z/2	Вход/выход порта A0, разряд 4
	CS4		O	Выход четвертого сигнала выбора внешнего устройства
	ARINC_D6		I/O	Вход/выход шестой линии данных
	HSO4		O/Z	Выход данных, канал 4
A5		83	I/O/Z/2	Вход/выход порта A0, разряд 5
	CS5		O	Выход пятого сигнала выбора внешнего устройства
	ARINC_S6		I/O	Вход/выход шестой селекторной линии
	HSO5		O/Z	Выход данных, канал 5
A6		87	I/O/Z/2	Вход/выход порта A0, разряд 6
	CS6		O	Выход шестого сигнала выбора внешнего устройства
	ARINC_D7		I/O	Вход/выход седьмой линии данных
	HSO6		O/Z	Выход данных, канал 6

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
A7		88	I/O/Z/2	Вход/выход порта A0, разряд 7
	CS7		O	Выход седьмого сигнала выбора внешнего устройства
	ARINC_S7		I/O	Вход/выход седьмой селекторной линии
	HSO7		O/Z	Выход данных, канал 7
A8		89	I/O/Z/2	Вход/выход порта A1, разряд 0
	UART0_TX		O	Выход данных
	UART0_SIROUT		O	Выход данных в режиме ИК-порта
	HSO8		O/Z	Выход данных, канал 8
A9		93	I/O/Z/2	Вход/выход порта A1, разряд 1
	UART0_RX/ UART0_SIRIN		I	Вход данных/ Вход данных в режиме ИК-порта
	–		–	–
	HSO9		O/Z	Выход данных, канал 9
A10		94	I/O/Z/2	Вход/выход порта A1, разряд 2
	UART0_CTS		I	Вход сигнала готовности к приему
	SPI1_MOSI		I/O	Вход/выход данных
	HSO10		O/Z	Выход данных, канал 10
A11		95	I/O/Z/2	Вход/выход порта A1, разряд 3
	UART0_DSR		I	Вход сигнала готовности источника данных
	SPI1_MISO		I/O	Вход/выход данных
	HSO11		O/Z	Выход данных, канал 11
A12		96	I/O/Z/2	Вход/выход порта A1, разряд 4
	UART0_RTS		O	Выход сигнала запроса на передачу
	SPI1_SCK		I/O	Вход/выход синхронизации
	HSO12		O/Z	Выход данных, канал 12
A13		98	I/O/Z/2	Вход/выход порта A1, разряд 5
	UART0_DTR		O	Выход сигнала готовности приемника данных
	SPI1_SS		I	Вход сигнала выбора ведомого устройства
	HSO13		O/Z	Выход данных, канал 13
A14		99	I/O/Z/2	Вход/выход порта A1, разряд 6
	UART0_DCD		I	Вход отклика при обнаружении информационного сигнала
	UART0_OUT1		O	Выход отклика при обнаружении информационного сигнала
	HSO14		O/Z	Выход данных, канал 14

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
A15		104	I/O/Z/2	Вход/выход порта A1, разряд 7
	UART0_RI		I	Вход сигнала-индикатора вызова
	UART0_OUT2		O	Выход сигнала-индикатора вызова
	HSO15		O/Z	Выход данных, канал 15
A16		105	I/O/Z/2	Вход/выход порта A2, разряд 0
	UART1_TX		O	Выход данных
	UART1_SIROUT		O	Выход данных в режиме ИК-порта
	HSI0		I	Вход данных, канал 0
A17		107	I/O/Z/2	Вход/выход порта A2, разряд 1
	UART1_RX/ UART1_SIRIN		I	Вход данных/ Вход данных в режиме ИК-порта
	–		–	–
	HSI1		I	Вход данных, канал 1
A18		108	I/O/Z/2	Вход/выход порта A2, разряд 2
	UART2_TX		O	Выход данных
	UART2_SIROUT		O	Выход данных в режиме ИК-порта
	HSI2		I	Вход данных, канал 2
A19		109	I/O/Z/2	Вход/выход порта A2, разряд 3
	UART2_RX/ UART2_SIRIN		I	Вход данных/ Вход данных в режиме ИК-порта
	–		–	–
	HSI3		I	Вход данных, канал 3
A20		111	I/O/Z/2	Вход/выход порта A2, разряд 4
	UART3_TX		O	Выход данных
	UART3_SIROUT		O	Выход данных в режиме ИК-порта
	HSI4		I	Вход данных, канал 4
A21		112	I/O/Z/2	Вход/выход порта A2, разряд 5
	UART3_RX/ UART3_SIRIN		I	Вход данных/ Вход данных в режиме ИК-порта
	–		–	–
	HSI5		I	Вход данных, канал 5
A22		113	I/O/Z/2	Вход/выход порта A2, разряд 6
	SPI0_MOSI		I/O	Вход/выход данных
	QEP1_A		I	Вход синхросигнала QEP1
	HSI6		I	Вход данных, канал 6

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
A23		114	I/O/Z/2	Вход/выход порта A2, разряд 7
	SPI0_MISO		I/O	Вход/выход данных
	QEP1_B		I	Вход сигнала направления вращения ротора QEP1
	HSI7		I	Вход данных, канал 7
A24		116	I/O/Z/2	Вход/выход порта A3, разряд 0
	SPI0_SCK		I/O	Вход/выход синхронизации
	QEP1_I		I	Индексный вход QEP1
	HSI8		I	Вход данных, канал 8
A25		117	I/O/Z/2	Вход/выход порта A3, разряд 1
	SPI0_SS		I	Вход сигнала выбора ведомого устройства
	QEP1_S		I	Вход стробирования QEP1
	HSI9		I	Вход данных, канал 9
A26		118	I/O/Z/2	Вход/выход порта A3, разряд 2
	SW0_SIN		I	Вход стробирующего сигнала SpaceWire0
	–		–	–
	HSI10		I	Вход данных, канал 10
A27		120	I/O/Z/2	Вход/выход порта A3, разряд 3
	SW0_DIN		I	Вход последовательных данных SpaceWire0
	–		–	–
	HSI11		I	Вход данных, канал 11
A28		125	I/O/Z/2	Вход/выход порта A3, разряд 4
	SW0_SOUT		O	Выход стробирующего сигнала SpaceWire0
	–		–	–
	HSI12		I	Вход данных, канал 12
A29		126	I/O/Z/2	Вход/выход порта A3, разряд 5
	SW0_DOUT		O	Выход последовательных данных SpaceWire0
	–		–	–
	HSI13		I	Вход данных, канал 13
A30		127	I/O/Z/2	Вход/выход порта A3, разряд 6
	I2C_SDA		I/O	Вход/выход данных
	–		–	–
	HSI14		I	Вход данных, канал 14
A31		129	I/O/Z/2	Вход/выход порта A3, разряд 7
	I2C_SCL		I/O	Вход/выход синхронизации
	–		–	–
	HSI15		I	Вход данных, канал 15

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
B0		130	I/O/Z/2	Вход/выход порта B0, разряд 0
	M1_TXD0		O	Прямой выход канала 0 блока МПИ1
	PWM0_A		O	Выход линии A
	–		–	–
B1		131	I/O/Z/2	Вход/выход порта B0, разряд 1
	M1_TXDN0		O	Инверсный выход канала 0 блока МПИ1
	PWM0_B		O	Выход линии B
	–		–	–
B2		133	I/O/Z/2	Вход/выход порта B0, разряд 2
	M1_TXD1		O	Прямой выход канала 1 блока МПИ1
	PWM1_A		O	Выход линии A
	–		–	–
B3		134	I/O/Z/2	Вход/выход порта B0, разряд 3
	M1_TXDN1		O	Инверсный выход канала 1 блока МПИ1
	PWM1_B		O	Выход линии B
	–		–	–
B4		139	I/O/Z/2	Вход/выход порта B0, разряд 4
	M0_TXD0		O	Прямой выход канала 0 блока МПИ0
	–		–	–
	–		–	–
B5		143	I/O/Z/2	Вход/выход порта B0, разряд 5
	M0_TXDN0		O	Инверсный выход канала 0 блока МПИ0
	–		–	–
	–		–	–
B6		147	I/O/Z/2	Вход/выход порта B0, разряд 6
	M0_TXD1		O	Прямой выход канала 1 блока МПИ0
	–		–	–
	–		–	–
B7		155	I/O/Z/2	Вход/выход порта B0, разряд 7
	M0_TXDN1		O	Инверсный выход канала 1 блока МПИ0
	–		–	–
	–		–	–
B8		135	I/O/Z/2	Вход/выход порта B1, разряд 0
	M1_RXD0		I	Прямой вход канала 0 блока МПИ1
	PWM2_A		O	Выход линии A
	–		–	–

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
B9		137	I/O/Z/2	Вход/выход порта B1, разряд 1
	M1_RXDN0		I	Инверсный вход канала 0 блока МПИ1
	PWM2_B		O	Выход линии B
	–		–	–
B10		138	I/O/Z/2	Вход/выход порта B1, разряд 2
	M1_RXD1		I	Прямой вход канала 1 блока МПИ1
	PWM_TZ0		I	Вход нулевого сигнала аварии
	–		–	–
B11		144	I/O/Z/2	Вход/выход порта B1, разряд 3
	M1_RXDN1		I	Инверсный вход канала 1 блока МПИ1
	PWM_TZ1		I	Вход первого сигнала аварии
	–		–	–
B12		151	I/O/Z/2	Вход/выход порта B1, разряд 4
	M0_RXD0		I	Прямой вход канала 0 блока МПИ0
	–		–	–
	–		–	–
B13		158	I/O/Z/2	Вход/выход порта B1, разряд 5
	M0_RXDN0		I	Инверсный вход канала 0 блока МПИ0
	–		–	–
	–		–	–
B14		162	I/O/Z/2	Вход/выход порта B1, разряд 6
	M0_RXD1		I	Прямой вход канала 1 блока МПИ0
	–		–	–
	–		–	–
B15		166	I/O/Z/2	Вход/выход порта B1, разряд 7
	M0_RXDN1		I	Инверсный вход канала 1 блока МПИ0
	–		–	–
	–		–	–
B16		145	I/O/Z/2	Вход/выход порта B2, разряд 0
	M1_BLOCK0		O	Выход блокировки канала 0 блока МПИ1
	PWM_TZ2		I	Вход второго сигнала аварии
	–		–	–
B17		146	I/O/Z/2	Вход/выход порта B2, разряд 1
	M1_BLOCK1		O	Выход блокировки канала 1 блока МПИ1
	–		–	–
	–		–	–

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
B18		148	I/O/Z/2	Вход/выход порта B2, разряд 2
	M0_BLOCK0		O	Выход блокировки канала 0 блока МПИ0
	–		–	–
	–		–	–
B19		149	I/O/Z/2	Вход/выход порта B2, разряд 3
	M0_BLOCK1		O	Выход блокировки канала 1 блока МПИ0
	–		–	–
	–		–	–
B20		150	I/O/Z/2	Вход/выход порта B2, разряд 4
	M1_EXTCLK		I	Вход внешнего синхросигнала блока МПИ1
	–		–	–
	–		–	–
B21		169	I/O/Z/2	Вход/выход порта B2, разряд 5
	M0_EXTCLK		I	Вход внешнего синхросигнала блока МПИ0
	–		–	–
	–		–	–
B22		152	I/O/Z/2	Вход/выход порта B2, разряд 6
	ARINC_D0		I/O	Вход/выход нулевой линии данных
	QEP0_A		I	Вход тактового сигнала QEP0
	–		–	–
B23		153	I/O/Z/2	Вход/выход порта B2, разряд 7
	ARINC_S0		I/O	Вход/выход нулевой селекторной линии
	QEP0_B		I	Вход сигнала направления вращения ротора QEP0
	–		–	–
B24		154	I/O/Z/2	Вход/выход порта B3, разряд 0
	ARINC_D1		I/O	Вход/выход первой линии данных
	QEP0_I		I	Индексный вход QEP0
	–		–	–
B25		156	I/O/Z/2	Вход/выход порта B3, разряд 1
	ARINC_S1		I/O	Вход/выход первой селекторной линии
	QEP0_S		I	Вход стробирования QEP0
	–		–	–
B26		157	I/O/Z/2	Вход/выход порта B3, разряд 2
	ARINC_D2		I/O	Вход/выход второй линии данных
	SW1_SIN		I	Вход стробирующего сигнала SpaceWire1
	–		–	–

Продолжение таблицы 2.1

Обозначение вывода	Альтернативная функция вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
B27		163	I/O/Z/2	Вход/выход порта B3, разряд 3
	ARINC_S2		I/O	Вход/выход второй селекторной линии
	SW1_DIN		I	Вход последовательных данных SpaceWire1
	–		–	–
B28		164	I/O/Z/2	Вход/выход порта B3, разряд 4
	ARINC_D3		I/O	Вход/выход третьей линии данных
	SW1_SOUT		O	Выход стробирующего сигнала SpaceWire1
	–		–	–
B29		165	I/O/Z/2	Вход/выход порта B3, разряд 5
	ARINC_S3		I/O	Вход/выход третьей селекторной линии
	SW1_DOUT		O	Выход последовательных данных SpaceWire1
	–		–	–
B30		167	I/O/Z/2	Вход/выход порта B3, разряд 6
	WDT_EXTCLK		I	Вход внешнего синхросигнала
	–		–	–
	–		–	–
B31		168	I/O/Z/2	Вход/выход порта B3, разряд 7
	–		–	–
	–		–	–
	–		–	–

[1)] Выводы A0–A3 во время сброса являются конфигурационными входами:
- A0 – разрешение режима PowerDown;
- A1 – разрешение использования кэша;
- A2, A3 – выбора количества сигналов CS.

Для получения более подробной информации следует обратиться к разделу 5 «Конфигурация, старт, сброс, тактирование микроконтроллера».

Таблица 2.2 – Функциональное назначение выводов микросхем 1874BE10T и 1874BE10AT, не имеющих альтернативных функций

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
ACH0	45	I	Вход АЦП, канал 0
ACH1	47	I	Вход АЦП, канал 1
ACH2	49	I I/O	Вход АЦП, канал 2/ Вход/выход напряжения средней точки
ACH3	51	I	Вход АЦП, канал 3
ACH4	53	I	Вход АЦП, канал 4
ACH5	55	I	Вход АЦП, канал 5
ACH6	57	I	Вход АЦП, канал 6
ACH7	59	I I/O	Вход АЦП, канал 7/ Вход/выход напряжения средней точки

Продолжение таблицы 2.2

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
ACH8	46	I	Вход АЦП, канал 8
ACH9	48	I	Вход АЦП, канал 9
ACH10	50	I	Вход АЦП, канал 10
ACH11	52	I	Вход АЦП, канал 11
ACH12	54	I	Вход АЦП, канал 12
ACH13	56	I	Вход АЦП, канал 13
ACH14	58	I	Вход АЦП, канал 14
ACH15	60	I	Вход АЦП, канал 15
ADC_REFHI	42	I	Верхнее опорное напряжение АЦП
ADC_REFLO	43	I	Нижнее опорное напряжение АЦП
ALE	186	O	Выход сигнала разрешения записи адреса ALE
BW0	196	I	Вход конфигурации для внешней шины (младший бит)
BW1	201	I	Вход конфигурации для внешней шины (старший бит)
CLKOUT	185	O	Выход системного тактового сигнала
DEBUG_EN	90	I	Вход разрешения порта JTAG
EA	170	I	Вход конфигурации для выбора внутренней памяти программ (следует притягивать к земле)
ECCEN	38	I	Вход конфигурации для включения коррекции ошибок
IEMECC0	40	I/O/Z	Вход/выход контрольного бита 0 младшего байта адреса/данных
IEMECC1	41	I/O/Z	Вход/выход контрольного бита 1 младшего байта адреса/данных
IEMECC2	64	I/O/Z	Вход/выход контрольного бита 2 младшего байта адреса/данных
IEMECC3	66	I/O/Z	Вход/выход контрольного бита 3 младшего байта адреса/данных
IEMECC4	67	I/O/Z	Вход/выход контрольного бита 0 первого байта адреса/данных
IEMECC5	68	I/O/Z	Вход/выход контрольного бита 1 первого байта адреса/данных
IEMECC6	69	I/O/Z	Вход/выход контрольного бита 2 первого байта адреса/данных
IEMECC7	71	I/O/Z	Вход/выход контрольного бита 3 первого байта адреса/данных
IEMECC8	106	I/O/Z	Вход/выход контрольного бита 0 второго байта адреса/данных
IEMECC9	110	I/O/Z	Вход/выход контрольного бита 1 второго байта адреса/данных
IEMECC10	115	I/O/Z	Вход/выход контрольного бита 2 второго байта адреса/данных
IEMECC11	119	I/O/Z	Вход/выход контрольного бита 3 второго байта адреса/данных
IEMECC12	124	I/O/Z	Вход/выход контрольного бита 0 старшего байта адреса/данных
IEMECC13	128	I/O/Z	Вход/выход контрольного бита 1 старшего байта адреса/данных

Продолжение таблицы 2.2

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
IEMECC14	132	I/O/Z	Вход/выход контрольного бита 2 старшего байта адреса/данных
IEMECC15	136	I/O/Z	Вход/выход контрольного бита 3 старшего байта адреса/данных
IEMH0	7	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 0
IEMH1	13	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 1
IEMH2	19	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 2
IEMH3	24	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 3
IEMH4	29	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 4
IEMH5	34	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 5
IEMH6	39	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 6
IEMH7	44	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 7
IEMH8	65	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 8
IEMH9	70	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 9
IEMH10	75	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 10
IEMH11	80	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 11
IEMH12	86	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 12
IEMH13	92	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 13
IEMH14	97	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 14
IEMH15	103	I/O/Z	Вход/выход старшего слова адреса/данных шины внешней памяти, разряд 15
IEML0	5	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 0
IEML1	6	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 1
IEML2	11	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 2
IEML3	12	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 3

Продолжение таблицы 2.2

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
IEML4	14	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 4
IEML5	18	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 5
IEML6	20	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 6
IEML7	21	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 7
IEML8	22	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 8
IEML9	23	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 9
IEML10	25	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 10
IEML11	26	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 11
IEML12	30	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 12
IEML13	31	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 13
IEML14	32	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 14
IEML15	33	I/O/Z	Вход/выход младшего слова адреса/данных шины внешней памяти, разряд 15
INST	193	O	Выход сигнала чтения команды
M0_ADDR0	174	I	Входы адреса удаленного терминала для блока МПИ0
M0_ADDR1	179	I	
M0_ADDR2	184	I	
M0_ADDR3	190	I	
M0_ADDR4	194	I	
M0_ODDADDR	197	I	Вход контрольного бита блока МПИ0
M1_ADDR0	172	I	Входы адреса удаленного терминала для блока МПИ1
M1_ADDR1	173	I	
M1_ADDR2	175	I	
M1_ADDR3	178	I	
M1_ADDR4	180	I	
M1_ODDADDR	181	I	Вход контрольного бита блока МПИ1
NMI	171	I	Вход немаскируемого прерывания
RD	187	O	Выход сигнала чтения
READY	195	I	Вход сигнала готовности
RESET	10	I/O	Вход/выход сигнала сброса
TCK	8	I	Вход тактового сигнала порта JTAG
TDI	205	I	Вход данных порта JTAG
TDO	203	O/Z	Выход данных порта JTAG
TMS	204	I	Вход режима порта JTAG
TRST	9	I	Вход сброса порта JTAG

Окончание таблицы 2.2

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
VPP	91	I	Вход конфигурации для старта с альтернативного адреса/ Вход сигнала возврата из режима пониженного энергопотребления
WR_HH	206	O	Выход сигнала записи старшего байта адреса/данных
WR_HL	202	O	Выход сигнала записи второго байта адреса/данных
WR_LH	192	O	Выход сигнала записи первого байта адреса/данных
WR_LL	191	O	Выход сигнала записи младшего байта адреса/данных
XTAL1	207	– I	Вывод подключения кварцевого резонатора/ Вход тактового сигнала
XTAL2	208	–	Вывод подключения кварцевого резонатора
VDD_ADC	27	–	Питание АЦП
GND_ADC	28	–	Земля АЦП
VDDC	4, 17, 37, 63, 72, 84, 102, 121, 140, 159, 177, 189, 198	–	Питание ядра
VDD	2, 15, 35, 61, 79, 100, 123, 142, 161, 182, 200	–	Питание периферии
GND	3, 16, 36, 62, 73, 78, 85, 101, 122, 141, 160, 176, 183, 188, 199	–	Цифровая земля
SMD_OUT	1	–	Не используется. **Оставлять неподключенным!**

2.2 Электрические параметры

Номинальное значение напряжения питания по выводам VDD должно быть 3,3 В ± 5 %.

Электрические параметры микросхем при приемке и поставке соответствуют нормам, приведенным в таблице 2.3. Электрические параметры микросхем во время и после воздействия специальных факторов приведены в таблице 2.4. Предельно допустимые и предельные значения параметров электрических режимов эксплуатации микросхем приведены в таблице 2.5.

Таблица 2.3 – Электрические параметры ИС 1874ВЕ10Т, 1874ВЕ10АТ 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 по выводам A0 – A31, B0 – B31, ALE, CLKOUT, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST,	U_{OL}			–60 ± 3 25 ± 10 85 ± 3[2)] 125 ± 5[3)]

RD, TDO, WR_HH, WR_HL, WR_LH, WR_LL, RESET, BW0, BW1, ECCEN[1)], В, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}0$ В, $I_{OL} = 2{,}8$ мА		–	0,4	

Продолжение таблицы 2.3

1	2	3	4	5
2 Выходное напряжение высокого уровня по выводам A0 – A31, B0 – B31, ALE, CLKOUT, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST, RD, TDO, WR_HH, WR_HL, WR_LH, WR_LL, BW0, BW1, ECCEN[1)], В, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}0$ В, $I_{OH} = -3{,}2$ мА	U_{OH}	$U_{CC1}-0{,}4$	–	
3 Ток утечки низкого уровня по входам с отключенной схемой «pull-up» и «pull-down» A0 – A31, B0 – B31, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, M0_ADDR0 – M0_ADDR4, M1_ADDR0 – M1_ADDR4, M0_ODDADDR, M1_ODDADDR, TCK, TDI, TMS, VPP, TRST[1), 4)], мкА, $U_{CC1} = U_{CC2} = 3{,}6$ В, $U_{CC3} = 3{,}3$ В, $U_{IL} = 0$ В	I_{ILL1}	–10	–	
4 Ток утечки высокого уровня по входам с отключенной схемой «pull-up» и «pull-down» A0 – A31, B0 – B31, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, M0_ADDR0 – M0_ADDR4, M1_ADDR0 – M1_ADDR4, M0_ODDADDR, M1_ODDADDR, TCK, TDI, TMS, VPP, TRST[1), 4)], мкА, $U_{CC1} = U_{CC2} = 3{,}6$ В, $U_{CC3} = 3{,}3$ В, $U_{IH} = U_{CC1}$	I_{ILH1}	–	10	
5 Ток утечки низкого уровня по выводам с подключенной схемой «pull-down» A0 – A31, B0 – B31, EA, ECCEN, NMI, BW1[1), 4)], мкА $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IL} = 0$ В	I_{ILL2}	–10	–	
6 Ток утечки высокого уровня по выводам с подключенной схемой «pull-up» A0 – A31, B0 – B31, ALE, BW0, EBUG_EN, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST, RD, VPP, READY, WR_HH, WR_HL, WR_LH, WR_LL, RESET[1), 4)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IH} = U_{CC1}$	I_{ILH2}	–	10	-60 ± 3 25 ± 10 85 ± 3[2)] 125 ± 5[3)]
7 Входной ток низкого уровня по выводам с подключенной схемой «pull-up» A0 – A31, B0 – B31, ALE, BW0, DEBUG_EN, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST, RD, VPP, READY, WR_HH, WR_HL, WR_LH, WR_LL[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IL} = 0$ В	I_{IL1}	–200	–	
8 Входной ток высокого уровня по выводам с подключенной схемой «pull-down» A0 – A31, B0 – B31, EA, ECCEN, BW1[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IH} = U_{CC1}$	I_{IH1}	–	200	
9 Входной ток низкого уровня по выводу тактового сигнала XTAL1[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IL} = 0$ В	I_{IL2}	–40[2)] –60[3)]	– –	
10 Входной ток высокого уровня по выводу тактового сигнала XTAL1[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IH} = U_{CC1}$	I_{IH2}	– –	40[2)] 60[3)]	
11 Входной ток низкого уровня по выводу сброса RESET[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IL} = 0$ В	I_{IL3}	–800	–50	
12 Входной ток высокого уровня по выводу немаскируемого прерывания NMI[1)], мкА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $U_{IH} = 2{,}4$ В	I_{IH3}	10	200	
13 Динамический ток потребления ядра в режиме сброса по выводам VDDC[1), 5)], мА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $f_{CI} = 33$ МГц	I_{OCC1}	–	200	

14 Динамический ток потребления периферии в режиме сброса по выводам VDD[1), 5)], мА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $f_{CI} = 33$ МГц	I_{OCC2}	–	100	

Окончание таблицы 2.3

1	2	3	4	5
15 Общие гармонические искажения канала АЦП, дБ, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	THD	–	–50	
16 Отношение сигнал/(шум + искажения) в канале АЦП, дБ, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	SINAD	50	–	
17 Интегральная нелинейность, МЗР, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	E_L	–5	5	–60 ± 3 25 ± 10 85 ± 3[2)] 125 ± 5[3)]
18 Дифференциальная нелинейность, МЗР, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	E_{LD}	–4	4	
19 Функциональный контроль[6)], $U_{CC1} = U_{CC2} = U_{CC3} = (3{,}0; 3{,}6)$ В, $f_{CI} = (0{,}001; 66)$ МГц	ФК	–	–	

_____

[1)] Погрешность измерения: выходных напряжений ±2 %, входных токов, токов утечки, тока потребления ±3 %.

[2)] Только для ИС 1874ВЕ10Т.

[3)] Только для ИС 1874ВЕ10АТ.

[4)] Параметры I_{ILL1}, I_{ILH1}, I_{ILL2}, I_{ILH2} при температуре минус 60 ºС не измеряются, а гарантируются нормами при температуре (25 ± 10) ºС.

[5)] Параметры I_{OCC1}, I_{OCC2} измеряются при отключенных от нагрузок выходах и подключенных к уровням U_{IL} и U_{IH} входах.

[6)] При проведении функционального контроля АЦП значения напряжений питания U_{CC1}, U_{CC2}, U_{CC3} изменяются синхронно.

Таблица 2.4 – Электрические параметры микросхем во время и после воздействия специальных факторов

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 по выводам A0 – A31, B0 – B31, ALE, CLKOUT, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST, RD, TDO, WR_HH, WR_HL, WR_LH, WR_LL, RESET, BW0, BW1, ECCEN, В, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}0$ В, $I_{OL} = 2{,}8$ мА	U_{OL}	–	0,6	–60 ± 3 25 ± 10 85 ± 3[2)] 125 ± 5[3)]
2 Выходное напряжение высокого уровня по выводам A0 – A31, B0 – B31, ALE, CLKOUT, IEMECC0 – IEMECC15, IEMH0 – IEMH15, IEML0 – IEML15, INST, RD, TDO, WR_HH, WR_HL, WR_LH, WR_LL, BW0, BW1, ECCEN, В, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}0$ В, $I_{OH} = -3{,}2$ мА	U_{OH}	2,0	–	

3 Динамический ток потребления ядра в режиме сброса по выводам VDDC[1)], мА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $f_{CI} = 33$ МГц	I_{OCC1}	–	250	
4 Динамический ток потребления периферии в режиме сброса по выводам VDD[1)], мА, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}6$ В, $f_{CI} = 33$ МГц	I_{OCC2}	–	150	

Окончание таблицы 2.4

1	2	3	4	5
5 Интегральная нелинейность[3)], МЗР, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	E_L	–5[4)]	5[4)]	
		–9[5)]	9[5)]	–60 ± 3
		–49[6)]	49[6)]	25 ± 10
6 Дифференциальная нелинейность[3)], МЗР, $U_{CC1} = U_{CC2} = U_{CC3} = 3{,}3$ В	E_{LD}	–4[4)]	4[4)]	85 ± 3[2)]
		–9[5)]	9[5)]	125 ± 5[3]
		–21[6)]	21[6)]	

[1)] Параметры I_{OCC1}, I_{OCC2} измеряются при отключенных от нагрузок выходах и подключенных к уровням U_{IL} и U_{IH} входах.

[2)] Только для ИС 1874ВЕ10Т.

[3)] Только для ИС 1874ВЕ10АТ.

[4)] В процессе и после воздействия фактора 7.И с характеристикой 7.$И_7$ на уровне $3 \times 1У_с$.

[5)] В процессе и после воздействия фактора 7.И с характеристикой 7.$И_7$ на уровне $2У_с$.

[6)] В процессе и после воздействия фактора 7.И с характеристикой 7.$И_7$ на уровне $5У_с$.

Таблица 2.5 – Предельно допустимые и предельные значения параметров электрических режимов эксплуатации микросхем

Наименование параметра режима, единица измерения		Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
			не менее	не более	не менее	не более
1 Напряжение питания ядра, В	1874ВЕ10Т	U_{CC1}	3,0	3,6	–	5,0
	1874ВЕ10АТ					4,0
2 Напряжение питания периферии, В	1874ВЕ10Т	U_{CC2}	3,0	3,6	–	5,0
	1874ВЕ10АТ					4,0
3 Напряжение питания АЦП, В	1874ВЕ10Т	U_{CC3}	3,0	3,6	–	5,0
	1874ВЕ10АТ					4,0
4 Входное напряжение низкого уровня, В		U_{IL}	–0,5	0,8	–0,6	–
5 Входное напряжение высокого уровня, В		U_{IH}	$0{,}2U_{CC1}+1{,}0$	U_{CC1}	–	$U_{CC1}+0{,}6$
6 Входное напряжение высокого уровня тактового сигнала XTAL1, В		U_{IHCI}	$0{,}7U_{CC1}$	U_{CC1}	–	$U_{CC1}+0{,}6$
7 Входное напряжение высокого уровня сигнала сброса RESET, В		U_{IHRST}	$0{,}6U_{CC1}$	U_{CC1}	–	$U_{CC1}+0{,}6$
8 Выходной ток низкого уровня, мА		I_{OL}	–	2,8	–	10
9 Выходной ток высокого уровня, мА		I_{OH}	–3,2	–	–10	–
10 Частота следования импульсов тактового сигнала XTAL1, МГц		f_{CI}	0,001	66	–	–
11 Емкость нагрузки, пФ		C_L	–	40	–	–

Примечание – Время работы в одном из предельных режимов должно быть не более 5 с (кроме параметров 10, 11).

3 Архитектура изделий

Микроконтроллеры построены по архитектуре Фон Неймана. Он имеет внешнюю системную магистраль для обмена данными с внешней памятью и дополнительными периферийными устройствами. Система команд микроконтроллера поддерживает широкий набор методов адресации, в т. ч. битовую адресацию.

Базовая архитектура микроконтроллеров, называемая архитектурой типа «регистр-регистр», при которой приемник и источники данных для арифметических и прочих команд могут располагаться в любой области карты памяти, дополняется подсистемой RISC однотактных микроопераций, в которой источником и приемником данных являются специальные регистры R0-R15 (расположены в регистровом файле). Это принципиально отличает данную архитектуру от архитектуры микроконтроллеров других серий. Такая архитектура обеспечивает достижение высокой производительности при одновременном упрощении работы с периферией (ячейки периферийных устройств сразу могут быть записаны результатами арифметических операций).

Процессорное ядро

Основные компоненты ядра микроконтроллеров и их взаимосвязи показаны на рисунке 3.1 .

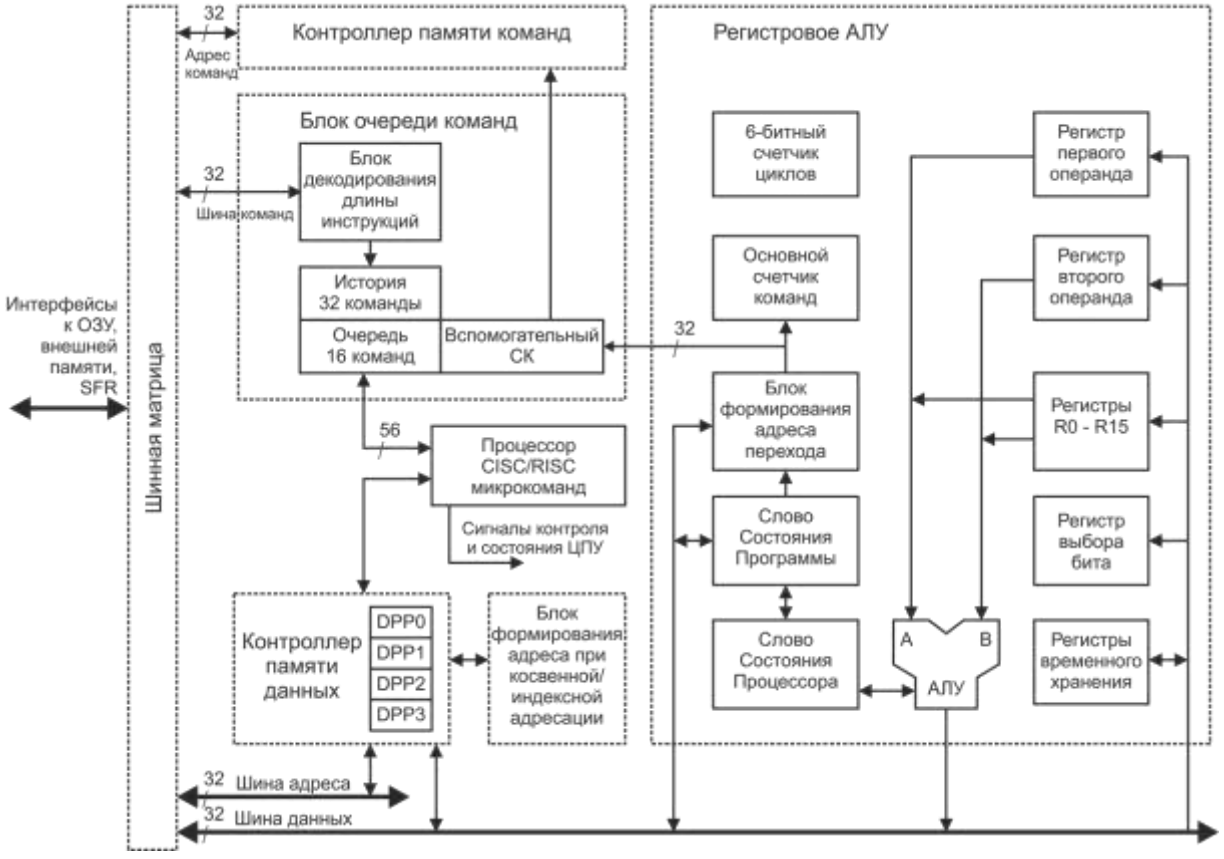


Рисунок 3.1 – Структурная схема ядра микроконтроллеров

Команды поступают в блок очереди команд, который может хранить до 16 команд длиной до 7 байт каждая. Процесс выборки и выполнения команд независимый, что позволяет минимизировать время простоя, как контроллера команд, так и ЦПУ.

Блок формирования адреса при косвенной/индексной адресации позволяет формировать адрес без использования функций АЛУ. В своем составе содержит сумматор/вычитатель.

Контроллер памяти данных выполняет выборку нужных данных из ОЗУ, области регистров периферийных устройств или внешней памяти. Архитектура микроконтроллеров позволяет вести выборку данных из любой области памяти, всегда используя полный 32-разрядный адрес. Максимальная производительность микроконтроллера достигается при выборке команд из внешней памяти, а данных – из области регистрового файла.

ЦПУ выполняет вычисления в АЛУ.

Процессор CISC/RISC микрокоманд подготавливает данные для загрузки в АЛУ, управляет сохранением результата. В случае если операнды и результат находятся в регистрах R0-R15, выполнение команд происходит за один машинный цикл (два такта сигнала XTAL1). В случае если данные находятся в ОЗУ, команда удлиняется на соответствующее количество циклов (на каждое обращение к ОЗУ затрачивается один цикл).

Слова вводятся в АЛУ через входы A и B. АЛУ выполняет все логические и арифметические операции за один машинный цикл (деление – за два), который равен двум тактам сигнала XTAL1. В своем составе АЛУ содержит встроенные аппаратные сумматоры/вычитатели, сдвигатели, умножитель, делитель и др.

Регистр слова состояния программы PSW содержит бит I, который разрешает или запрещает обслуживание всех маскируемых прерываний, бит PSE, разрешающий или запрещающий работу сервера периферийных транзакций (PTS) и шесть флагов, отражающих состояние программы пользователя.

Способы адресации

Система команд микроконтроллеров содержит следующие типы адресации: прямая регистровая, непосредственная, косвенная, косвенная с автоинкрементом, короткая индексная, длинная индексная, а также длинная индексная адресация с использованием регистров счетчика команд PC и указателя стека SP. Каждая команда использует, по крайней мере, один из способов адресации.

Прерывания

Основной функцией микроконтроллеров является обеспечение управления устройствами в реальном времени. Схема контроллера прерываний позволяет событиям управлять выполнением программы. Встроенная периферия, внешний сигнал или команда могут сгенерировать запрос на прерывание.

При возникновении прерывания устанавливается соответствующий бит в регистре ждущих прерываний. Каждое из возможных прерываний маскируется соответствующим битом регистра масок прерываний. Когда событие вызывает прерывание, ядро обслуживает это прерывание перед выполнением следующей команды и возвращается к прерванной программе.

Прерывание может быть обслужено посредством контроллера прерываний или блоком PTS. Для каждого из маскируемых прерываний можно выбрать вариант его обслуживания. Немаскируемое прерывание по выводу NMI может быть обслужено только контроллером прерываний.

4 Распределение адресного пространства памяти

32-разрядная шина адреса и данных позволяет охватывать диапазон адресов объемом до 4 Гбайт. Микроконтроллер может функционировать в одном из двух режимов:

- 16-разрядный с полной поддержкой системы команд MCS-96 для операций над байтами и словами;
- 32-разрядный с частичным использованием системы команд MCS-96 и новых инструкций, позволяющих выполнять операции над двойными словами.

Наличие RISC подсистемы и новых инструкций для работы с 32-разрядными данными позволяет ускорить выполнение арифметических, логических и других операций до восьми раз, по сравнению с базовой архитектурой MCS-96.

На рисунке 4.1 показана структура 32-разрядного регистра для хранения двойного слова данных.

Рисунок 4.1 – Структура регистра

Регистр объединяет в себе четыре байта: байт 1 (младший байт), байт 2, байт 3 и байт 4 (старший байт). Первый и второй байты образуют младшее слово регистра, а третий и четвертый – старшее слово.

Каждый байт доступен программно по собственному адресу при использовании инструкций для работы с байтами.

При обращении к словам адрес должен быть четным (для младшего слова это будет адрес младшего байта, для старшего слова – адрес байта 3).

Обращение к двойным словам осуществляется по адресам их младших байт, т. е. адрес должен быть кратным четырем.

Примечание – При работе со словами и двойными словами использование нечетных адресов не допускается.

4.1 Внутренняя память

Внутренняя память микроконтроллера представлена двумя блоками ОЗУ регистров общего назначения для размещения данных и двумя блоками регистров специальных функций периферии (SFR). Области ОЗУ и SFR представляют собой, так называемый, регистровый файл.

Примечание – Размещение инструкций в регистровом файле не допускается.

ОЗУ

В нулевом блоке с 0000_0000h по 0000_005Fh расположены 16 регистров быстрых команд R0 – R15, нулевой регистр (ZERO_REG) и четыре регистра указателя страниц данных DPP0 – DPP3. Ячейки памяти с адресами от 0000_0060h по 0000_3FFFh составляют ОЗУ0, а от 0001_0000h по 0001_3FFFh – ОЗУ1.

SFR

Ячейки с 0000_4000h по 0000_4FFFh и с 0010_0000h по 0010_4FFFh – регистры специальных функций (SFR). Из них область адресов 0010_0000h – 0010_1FFFh является памятью контроллеров магистрального последовательного интерфейса (МПИ) – по 4 Кбайт на каждый. В этих областях формируются блоки данных сообщений.

PSRAM

Также на кристалле расположена память типа PSRAM, имеющая выделенный для нее из общего адресного пространства диапазон адресов и предназначенная для размещения как команд, так и данных. Если в счетчике команд PC находится значение из диапазона 0000_5000h – 0000_5FFFh, выборка команд или данных всегда осуществляется из внутренней PSRAM. Скорость обращения к PSRAM такая же, как скорость обращения к ОЗУ, что может значительно ускорять выполнение размещенной в ней программы, относительно выполнения этой же программы, но расположенной во внешней памяти.

В отличие от ОЗУ содержимое PSRAM не стирается при сбросе микроконтроллера, а только при отключении питания.

4.2 Внешняя память

Внешняя память микроконтроллера представляет собой память команд и данных, которая может занимать все адресное пространство, за исключением адресов PSRAM, т. е. адреса с 0000_0000 по 0000_4FFFh и с 0000_6000h по FFFF_FFFFh. Две области внешней памяти, адреса которых не пересекаются с адресами областей внутренних ОЗУ и SFR, допускают размещение как команд, так и данных.

На рисунке 4.2 показано распределение адресного пространства между блоками внутренней и внешней памяти.

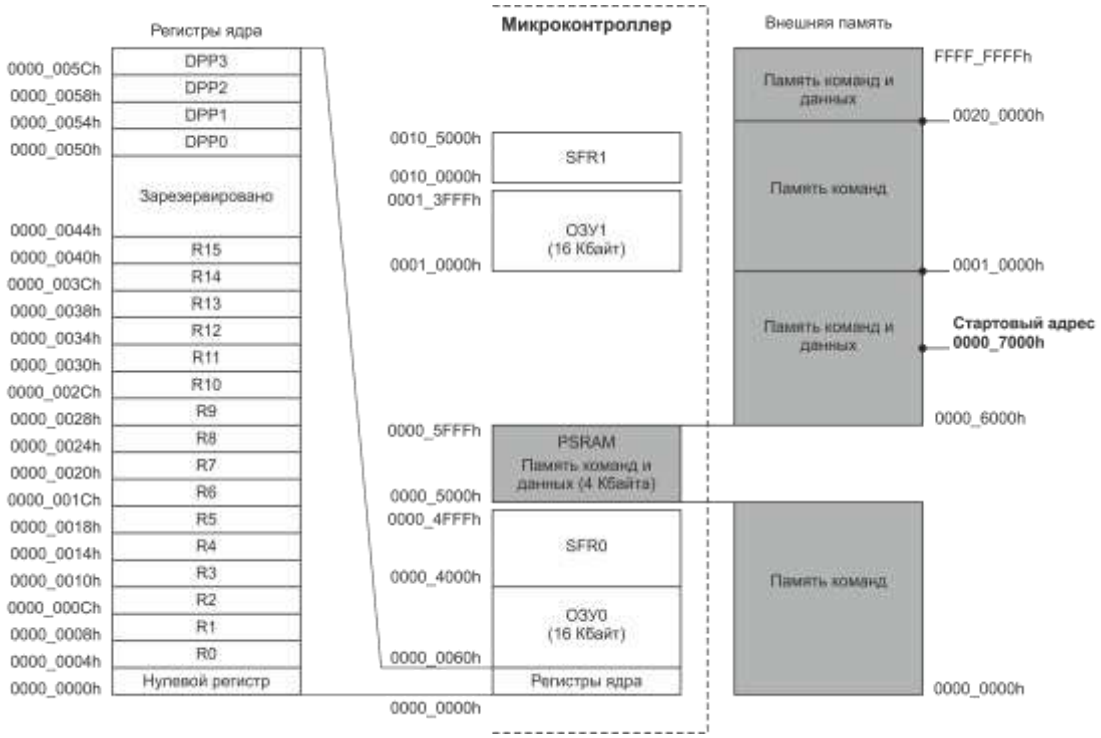


Рисунок 4.2 – Организация памяти

5 Конфигурация, старт, сброс, тактирование микроконтроллеров

Микроконтроллеры позволяют пользователю производить гибкую конфигурацию всей системы, а именно – задавать разрядность внешней шины, включать защиту передаваемой информации кодом Хэмминга, подключать до восьми внешних устройств хранения информации (или других), каждое с индивидуальным управлением.

5.1 Конфигурация микроконтроллеров

Конфигурация микроконтроллеров осуществляется при выходе из состояния сброса (перевод RESET на уровень логической единицы).

1 Если на выводе A0 микроконтроллера находится логическая единица, то это является разрешением для перевода микроконтроллера в режим пониженного энергопотребления PowerDown при его дальнейшей работе по программе командой MOD #2.

2 Если на выводе A1 микроконтроллера находится логическая единица, то это является разрешением для включения механизма кэширования команд в ядре микроконтроллера при выполнении программы.

3 Выводы A2 и A3 задают количество линий из восьми доступных CS0 – CS7 для выбора внешних устройств. Поскольку эти функции являются альтернативными для выводов A0 – A7 микроконтроллера, то для выбранных линий эти функции включаются программно (записью единиц в соответствующие биты регистра ALTFEN порта A). Эти функции могут быть отключены в дальнейшем.

В таблице 5.1 показаны комбинации состояний выводов A2 и A3.

Таблица 5.1 – Комбинации состояний выводов A2 и A3

Выводы микроконтроллера		Выбранные линии
A3	A2	
0	0	Не выбраны
0	1	CS7 и CS6
1	0	CS7 – CS4
1	1	CS7 – CS0

Примечание – В независимости от количества выбранных линий (за исключением случая, когда линии не выбраны), первой всегда активируется CS7 и уровень сигнала на ней устанавливается на уровень логического нуля, на остальных линиях – единицы.

4 Если на входе VPP находится логическая единица, то после выхода микроконтроллера из состояния сброса стартовым адресом будет – 0000_7000h (внешняя память).

Если на входе VPP находится логический ноль, то после выхода микроконтроллера из состояния сброса стартовым адресом будет – 0000_5000h (внутренняя PSRAM), но следует помнить, что в этом случае дальнейшее обращение к внешней памяти будет невозможно.

В таблице 5.2 показано состояние выводов, отвечающих за выбор стартового адреса.

Таблица 5.2 – Выбор стартового адреса

Выводы микроконтроллера			Стартовый адрес
VPP	RESET	ALE	
1	1	1	0000_7000h (внешняя память)
0	1	1	0000_5000h (внутренняя PSRAM)

Примечание – Поскольку микроконтроллер не имеет внутренней ПЗУ программ, вывод EA следует притягивать к земле.

5.2 Конфигурация интерфейса внешней памяти

Интерфейс внешней памяти включает в свой состав конфигурируемую 8-/16-/32-разрядную двунаправленную шину адреса/данных IEM и дополнительную 16-разрядную шину IEMECC, которая используется при защите передаваемой информации кодом Хэмминга, см. рисунок 5.1

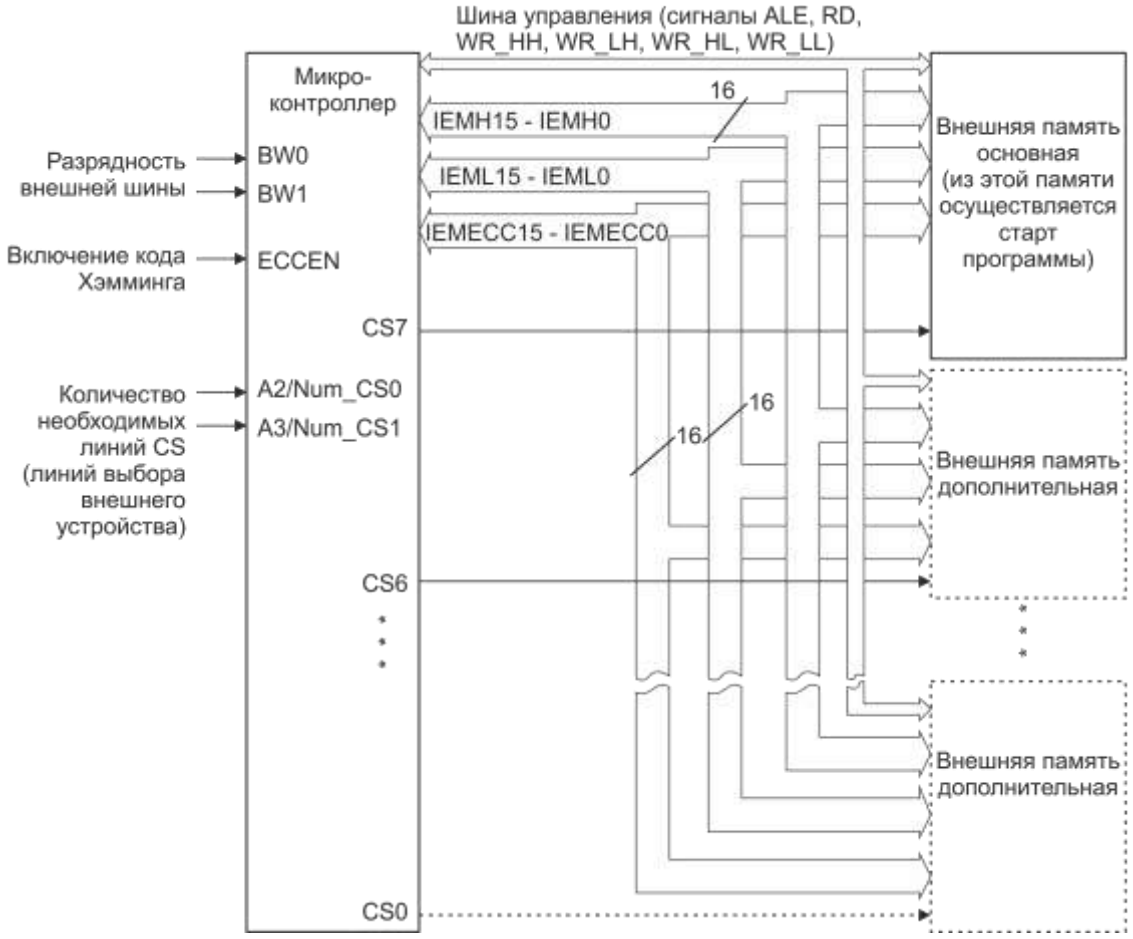


Рисунок 5.1 – Подключение внешней памяти

Конфигурация интерфейса подключения внешней памяти осуществляется посредством регистров BUSCON, см. рисунок 5.2.

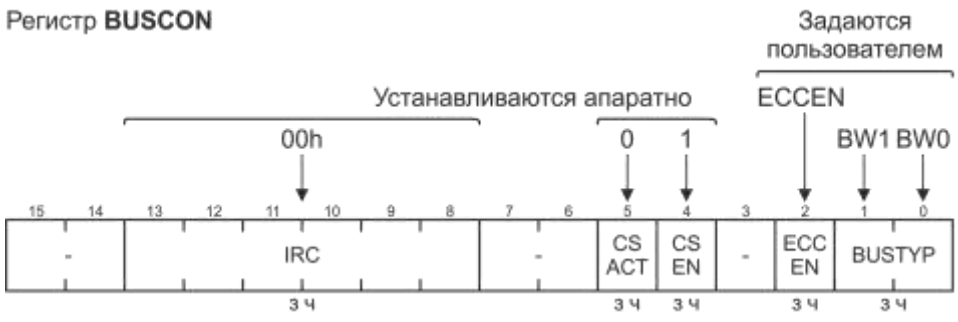


Рисунок 5.2 – Загрузка регистра BUSCON при выходе микроконтроллера из состояния сброса

При выходе из сброса микроконтроллер считывает состояние выводов BW0 и BW1 и записывает обнаруженную комбинацию в регистр BUSCON. Эта комбинация задает разрядность внешней шины (00b – 8-разрядная, 01b – 16-разрядная, 10b – 32-разрядная, 11b – не используется). Также считывается состояние вывода ECCEN и записывается в регистр. Остальные биты регистра заполняются микроконтроллером аппаратно и их состояние может быть перепрограммировано в дальнейшем.

Примечание – Комбинация значений, показанная на рисунке 5.2, идентична для всех восьми регистров BUS0CON – BUS7CON и записывается в них при конфигурировании шины. Бит CSEN, переводящий соответствующий сигнал выбора устройства в активное состояние, установлен во всех выбранных регистрах, но конфликт не возникает, так как на момент старта активен только вывод CS7, поскольку у него самый высокий приоритет.

Адресные окна

Каждому регистру BUSCON соответствует одно адресное окно. Поскольку регистр BUS7CON назначен самым приоритетным, его адресное окно в момент старта охватывает весь адресный диапазон объемом 4 Гбайт. Таким образом, в регистре нижней границы адресного окна BUS7ADDRLO находится значение 0000_0000h, а в регистре верхней границы окна BUS7ADDRHI – FFFF_FFFFh.

На рисунке 5.3 для примера показан случай, когда в системе используются три микросхемы памяти.

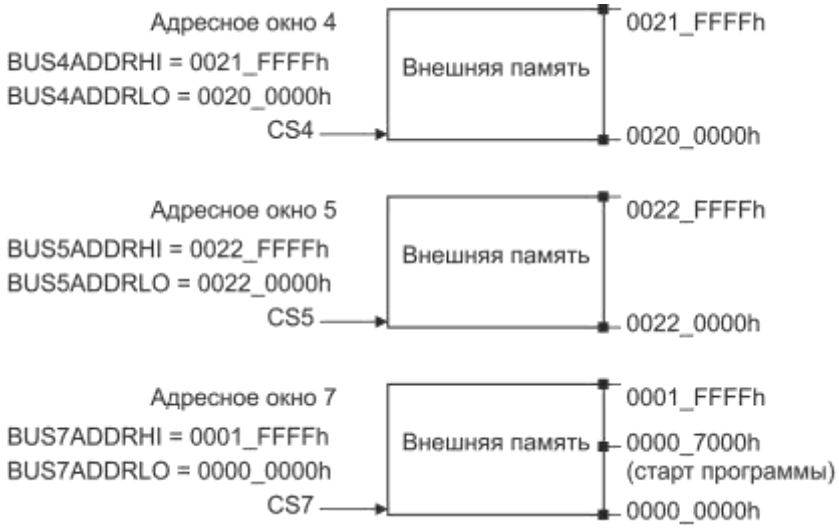


Рисунок 5.3 – Окна в адресном пространстве

При выходе из сброса на конфигурационных выводах микроконтроллера было – A3/Num_CS1 = 1 и A2/Num_CS0 = 0, т. е. были выбраны четыре вывода для выбора внешних устройств, а именно CS7 – CS4 и переведен в активное состояние сигнал CS7. После старта микроконтроллера каждой из микросхем программно было выделено адресное окно и заданы его параметры.

Пока счетчик команд находится в адресном окне 7, активен сигнал CS7. Как только счетчик команд дойдет до значения 0020_0000h, сигнал CS7 станет неактивным, а сигнал CS4 активируется. Таким образом, счетчик команд перейдет в адресное окно 4. Для диапазона адресов 0022_0000h – 0022_FFFFh будет использоваться адресное окно 5. Сигнал CS6 и соответствующее адресное окно 6 не используются.

5.3 Рекомендации по работе с PSRAM

После старта программы из внешней памяти можно переписать часть кода во внутреннюю PSRAM, подать снаружи на вывод VPP логическую единицу и сбросить микроконтроллер внешним сигналом RESET или командой RST. После этого будет произведен старт из PSRAM. Следует помнить, что после этого обращение к внешней памяти будет невозможно. Для прерывания выполнения программы из PSRAM нужно выполнить сброс микроконтроллера.

После старта программы из внешней памяти и переписывания части кода во внутреннюю PSRAM можно не выполнять сброс микроконтроллера, а осуществить косвенный переход во внутреннюю память командой BR или EBR, в зависимости от режима работы, а также можно воспользоваться командой абсолютного перехода VLJMP. В этом случае внешняя память остается доступной.

При достижении верхней границы PSRAM в счетчик команд будет загружен адрес 0000_6000h.

5.4 Сброс

Сброс микроконтроллера осуществляется подачей сигнала низкого уровня на вывод RESET. Микроконтроллер будет находиться в режиме сброса до тех пор, пока на вывод RESET не будет подана логическая единица.

Кроме этого микроконтроллер может быть сброшен встроенным сторожевым таймером, а также инструкцией RST. В этом случае на выводе RESET формируется состояние логического нуля.

5.5 Тактирование микроконтроллера

Источником синхросигнала микроконтроллера является кварцевый резонатор, подключаемый к выводам XTAL1 и XTAL2, или внешний тактовый генератор, подключаемый к выводу XTAL1. Внутри схемы сигнал подается на осциллятор. Выходной сигнал осциллятора с частотой Fosc делится на два. Сигнал на выходе делителя является основным тактовым сигналом микроконтроллера и имеет название SysCLK..

Синхросигнал SysCLK также является сигналом CLKOUT и напрямую непрерывно передается на выход CLKOUT микроконтроллера. Один такт сигнала CLKOUT является машинным тактом (мт). Таким образом

$$SysCLK = CLKOUT = Fosc/2 = 1 \text{ мт.}$$

Управление синхронизацией ЦПУ и периферии осуществляется двумя основными блоками – коммутатором синхросигналов и блоком управления режимами пониженного энергопотребления, см. рисунок 5.4. Дополнительно реализована возможность перевода микроконтроллера в режим Slow (режим пониженной в два раза частоты тактирования) установкой соответствующего бита в регистре CLKEN.

Регистр разрешения тактирования CLKEN входит в состав коммутатора. Каждому периферийному блоку соответствует один бит регистра. Установленный бит разрешает тактирование. По умолчанию тактирование периферии включено. Любой блок всегда можно выключить, сбросив соответствующий ему бит в регистре CLKEN, тем самым уменьшив общее энергопотребление схемы.

Время включения/отключения периферии составляет один машинный такт.

Тактирование ядра и блока вычислений с плавающей запятой (FPU) микроконтроллера постоянное и прекращается только в режиме PowerDown.

Сигналы на включение и выключение режимов пониженного энергопотребления Idle и PowerDown поступают на отдельный блок, который в свою очередь управляет коммутатором синхросигналов.

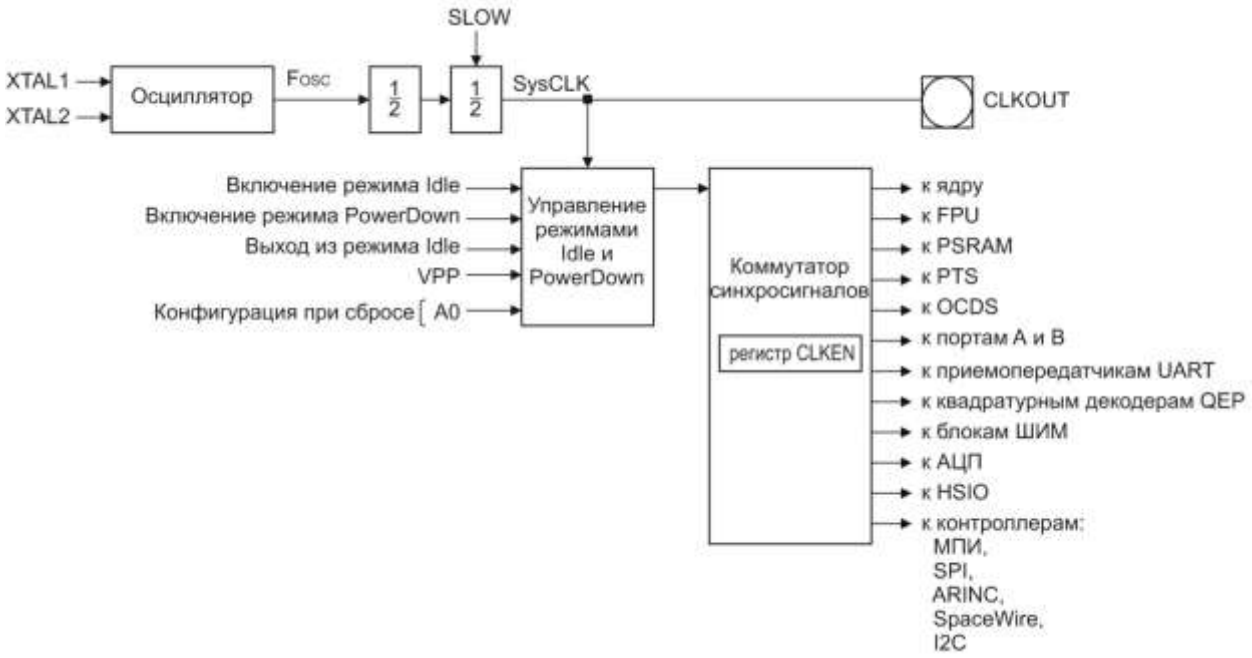


Рисунок 5.4 – Структурная схема синхронизации

Во время отладки микроконтроллера (с использованием JTAG) управление тактированием периферии осуществляется посредством дополнительных регистров CLKDEB_L и CLKDEB_H, значение которых переписывается в регистр CLKEN по сигналу прерывания DEBUG, см. раздел 24 «Программно-аппаратные средства отладки».

6 Типы данных и способы адресации

6.1 Типы операндов

Архитектура микроконтроллера поддерживает семь типов данных, которые обычно используются в управляющих алгоритмах:

- BIT – бит (беззнаковый);
- BYTE – байт (беззнаковый);
- WORD – слово (беззнаковый);
- DOUBLE-WORD – двойное слово (беззнаковый);
- SHORT-INTEGER – короткое целое (знаковый);
- INTEGER – целое (знаковый);
- LONG-INTEGER – длинное целое (знаковый).

В таблице 6.1 дается общий обзор типов операндов.

Таблица 6.1 – Типы операндов

Тип операнда	Число бит	Диапазон значений	Адресация
BIT	1	1 или 0	Как к компоненту байта
BYTE	8	от 0 до 255	К байту
WORD	16	от 0 до 65 535	К младшему байту (адрес должен быть кратным двум)
DOUBLE WORD	32	от 0 до 4 294 967 295	К младшему байту младшего слова (адрес должен быть кратным четырем)
SHORT INTEGER	8	от –128 до +127	К байту
INTEGER	16	от –32 768 до +32 767	К младшему байту (адрес должен быть кратным двум)
LONG INTEGER	32	от –2 147 483 648 до +2 147 483 647	К младшему байту младшего слова (адрес должен быть кратным четырем)

Для совместимости с программными средствами сторонних производителей необходимо придерживаться правил, принятых в программировании на языке «Си» для адресации 32-битных операндов.

Для каждого типа операнда есть соответствующий эквивалент для ассемблера и «Си». Ниже представлены типы операндов и их эквиваленты в порядке: тип операнда – эквивалент ассемблера/эквивалент «Си».

BYTE – BYTE/unsigned char
SHORT INTEGER – BYTE/char
WORD – WORD/unsigned int
INTEGER – WORD/int
DOUBLE WORD – LONG/unsigned long
LONG INTEGER – LONG/long

Операнд BIT

Операнд BIT – одноразрядная переменная, которая может принимать булевы значения «TRUE» и «FALSE». Бит адресуется как компонент байта или слова. Прямая адресация бита не поддерживается.

Операнд BYTE

Операнд BYTE – 8-битная переменная без знака, которая может принимать значения от 0 (00h) до 255 (FFh). Биты в пределах байта нумеруются от 0 до 7, бит 0 – младший бит. Для обозначения байтовой переменной используется символьное имя breg. Над байтовыми операндами могут выполняться арифметические операции и операции сравнения (вычитания) с контролем выхода результата за пределы допустимого диапазона по состоянию флага переноса C регистра PSW. Логические операции над байтовыми переменными осуществляются побитно.

Байтовые операнды могут размещаться в любом месте адресного пространства без ограничений по выравниванию.

Операнд WORD

Операнд WORD – 16-битная переменная без знака, которая может принимать значения от 0 (0000h) до 65535 (FFFFh). Биты в пределах слова нумеруются от 0 до 15, бит 0 – младший бит. Биты с 0 по 7 представляют собой младший байт слова, биты с 8 по 15 – старший байт слова. Для обозначения переменной слово используется символьное имя wreg. Над операндами-словами могут выполняться арифметические операции и операции сравнения (вычитания) с контролем выхода результата за пределы допустимого диапазона по состоянию флага переноса C регистра PSW. Логические операции над операндами-словами осуществляются побитно.

Операнды-слова могут размещаться в адресном пространстве только по четным адресам (по границе слова). Младший байт слова должен иметь четный адрес, а старший байт слова – на единицу больший, нечетный адрес. Адресом слова считается адрес его младшего байта.

Операнд DOUBLE WORD

Операнд DOUBLE WORD – 32-битная переменная без знака, которая может принимать значения от 0 (0000_0000h) до 4294967295 (FFFF_FFFFh). Биты в пределах двойного слова нумеруются от 0 до 31, бит 0 – младший бит. Биты с 0 по 15 представляют собой младшее слово, биты с 16 по 31 – старшее слово. Для обозначения переменной двойное слово используется символьное имя lreg. Над операндами-двойными словами могут выполняться арифметические операции и операции сравнения (вычитания) с контролем выхода результата за пределы допустимого диапазона по состоянию флага переноса C регистра PSW. Логические операции над операндами-двойными словами осуществляются побитно.

Операнды-двойные слова могут размещаться в адресном пространстве только по адресам, кратным 4 (по границе двойного слова). Адресом двойного слова считается адрес его младшего слова.

Операнд SHORT INTEGER

Операнд SHORT INTEGER – 8-битная переменная со знаком, которая может принимать значения от минус 128 (FFh) до плюс 127 (7Fh). Для обозначения используется символьное имя breg. При выполнении арифметических операций и выходе результата за пределы допустимого диапазона в регистре PSW устанавливается флаг переполнения V.

Операнды типа короткое целое могут размещаться в любом месте адресного пространства без ограничений по выравниванию.

Операнд INTEGER

Операнд INTEGER – 16-битная переменная со знаком, которая может принимать значения от минус 32768 (FFFFh) до плюс 32767 (7FFFh). Для обозначения используется символьное имя wreg.

При выполнении арифметических операций и выходе результата за пределы допустимого диапазона в регистре PSW устанавливается флаг переполнения V.

Операнды типа целое могу размещаться в адресном пространстве только по четным адресам (выравниваются по границе слова). Адресом операнда считается адрес его младшего байта.

Операнд LONG INTEGER

Операнд LONG INTEGER – 32-битная переменная со знаком, которая может принимать значения от минус 2147483648 (FFFF_FFFFh) до плюс 2147483647 (7FFF_FFFFh). Для обозначения используется символьное имя lreg.

При выполнении арифметических операций и выходе результата за пределы допустимого диапазона в регистре PSW устанавливается флаг переполнения V.

Операнды типа длинное целое могут размещаться в адресном пространстве только по адресам, кратным 4 (по границе двойного слова). Адресом операнда считается адрес его младшего слова.

6.2 Преобразование типов операндов

Команда LDBZE выполняет загрузку байтового операнда с автоматическим расширением нулем, т. е. преобразование байта в слово.

Команда LDBSE выполняет загрузку байта с автоматическим расширением знакового разряда, т. е. преобразование короткого целого в целое.

Для этой же цели можно воспользоваться командой расширения знака байта EXTB при условии, что байт предварительно загружен в слово.

Имеются возможности преобразования операндов типа целое в длинное целое с применением команды расширения знака EXT. Команда анализирует старший разряд (знаковый) в младшем слове и заполняет старшее слово его значением.

6.3 Способы адресации операндов и форматы команд

Форматы команд

Длина машинной инструкции в байтах зависит от типа команды и используемого способа адресации операндов. Команды могут иметь длину от 1 до 7 байт. Одна и та же команда может применяться с различными способами адресации операндов.

Наиболее простые форматы имеют команды, в которых операнды не задаются. Такие команды являются однобайтовыми и состоят только из кода операции. Например, команды RST (сброс микроконтроллера), SETC (установка флага переноса), DI (запрет всех прерываний).

Все остальные команды по числу обрабатываемых ими операндов делятся на одно-, двух- и трехоперандные.

В команде только один операнд может располагаться в произвольном месте адресного пространства и быть доступным с использованием базового способа адресации. По умолчанию, этот операнд является источником данных (обозначается символьным именем SRC или SRC2, если операндов три). Другой операнд может быть доступен только с использованием прямой регистровой адресации и является приемником данных (обозначается символьным именем DEST).

Если в команде три операнда, то первым указывается приемник данных DEST, вторым указывается первый источник данных SRC1 и третьим – второй источник данных SRC2. Операнд SRC1 и приемник DEST могут быть доступны только с использованием прямой регистровой адресации.

Тип базового способа адресации задается двумя младшими битами в первом байте машинной инструкции, т. е. коде команды.

В системе команд микроконтроллера используются четыре базовых способа адресации: прямая регистровая (di, код 00b), непосредственная (im, код 01b), косвенная (in, код 10b), индексная (ix, код 11b).

Регистры быстрых RISC-команд

Шестнадцать регистров, расположенных в начале адресного пространства ОЗУ микроконтроллера с адресами от 0000_0004h до 0000_0040h, представляют собой 32-разрядные регистры R0-R15, соответственно. При использовании этих регистров в качестве операндов в командах время их выполнения сокращается до одного машинного такта.

Указатели страниц данных (DPP0 – DPP3)

Система команд MCS-96 не предусматривает команд переходов и способов адресации для области памяти свыше 64 Кбайт. Для поддержки функциональности новой адресной области была расширена система команд. Для осуществления переходов и запусков подпрограмм в обоих режимах работы ядра появились команды «очень длинных» переходов VLCALL, VLJMP. При вызовах подпрограмм инструкциями SCALL, LCALL, при переходах на подпрограммы обработчиков прерываний в стеке сохраняется полный 32-разрядный адрес. Доступ к данным осуществляется посредством механизма адресации с помощью указателей страниц данных DPP.

Микроконтроллер имеет 4 указателя страниц данных DPP0, DPP1, DPP2, DPP3. Они позволяют расширить адрес в любом из режимов работы ядра до 32-разрядов.

Примечание – Указатель на страницу данных DPP0 не используется для адресов, участвующих в прямой адресации, меньших 0000_0060h. В этой области находятся регистры R0-R15 и сами указатели DPP.

В случае работы ядра в режиме MCS-96 работают только два указателя: DPP0 и DPP1. Формирование адреса в этом случае представлено на рисунке 6.1.

Особенностью использования указателей является то, что они расширяют адрес только при микрооперациях прямой адресации. Какой из указателей DPP будет применяться при формировании адреса, выбирается старшим битом адреса из команды.

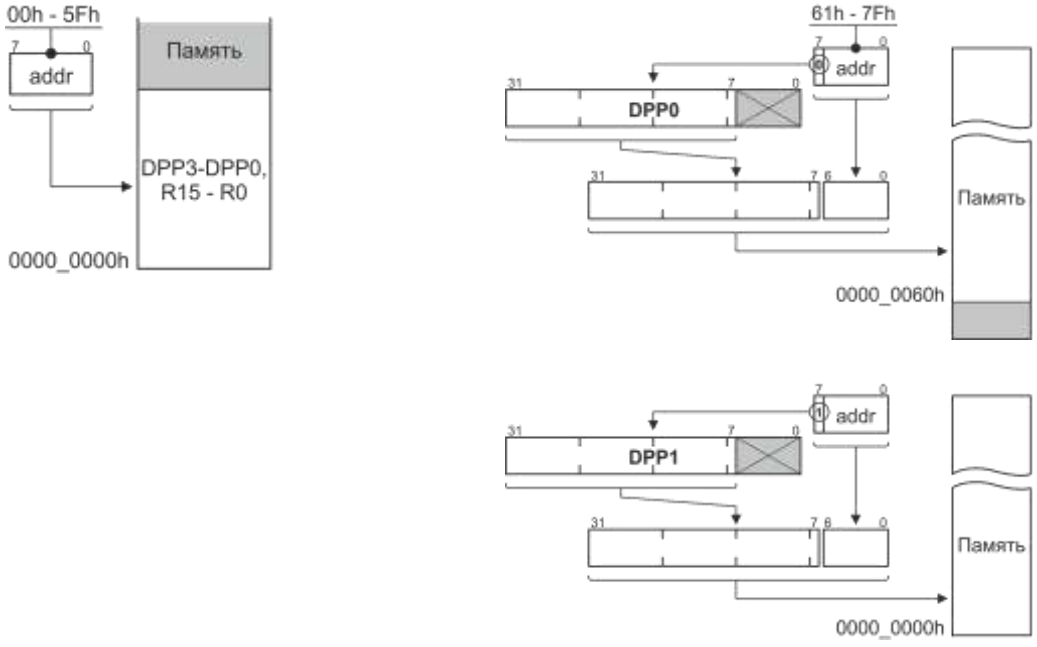


Рисунок 6.1 – Формирование адреса в 16-разрядном режиме

Рассмотрим это на примере. В случае выполнения команды ADD 60h, 62h запускаемые микрооперации будут выглядеть следующим образом:

1 Происходит чтение по прямому адресу значения ячейки 62h. Значение попадает в АЛУ.

2 Происходит чтение по прямому адресу значения ячейки 60h. Значение попадает в АЛУ.

3 Результат операции после вычисления по прямому адресу заносится в ячейку 60h.

В данном случае во всех трех стадиях выполнения команды используется прямая адресация, соответственно во всех них будет использоваться DPP (в данном примере DPP0). В случае выполнения команды ADD 60h, [62h] запускаемые микрооперации будут выглядеть следующим образом:

1 Происходит чтение по прямому адресу значения ячейки 62h.

2 Происходит чтение с использованием косвенной адресации по адресу, взятому из ячейки 62h. Это значение попадает в АЛУ.

3 Происходит чтение по прямому адресу значения ячейки 60h. Значение попадает в АЛУ.

4 Результат операции после вычисления по прямому адресу заносится в ячейку 60h.

В данном случае DPP используется при адресации всегда, кроме шага 2. Во всех случаях, когда адрес берется из ячейки памяти, указатели страниц не используются. Особенностью адресации в данном режиме работы ядра является то, что косвенной адресации доступно только 64 Кбайт, как и в MCS-96. То есть ячейка с адресом может находиться в любом месте адресного пространства, но считается оттуда только 16 бит конечного адреса.

Для режима работы с 32-разрядными данными применяются все четыре указателя страниц. Формирование адреса при этом представлено на рисунке 6.2.

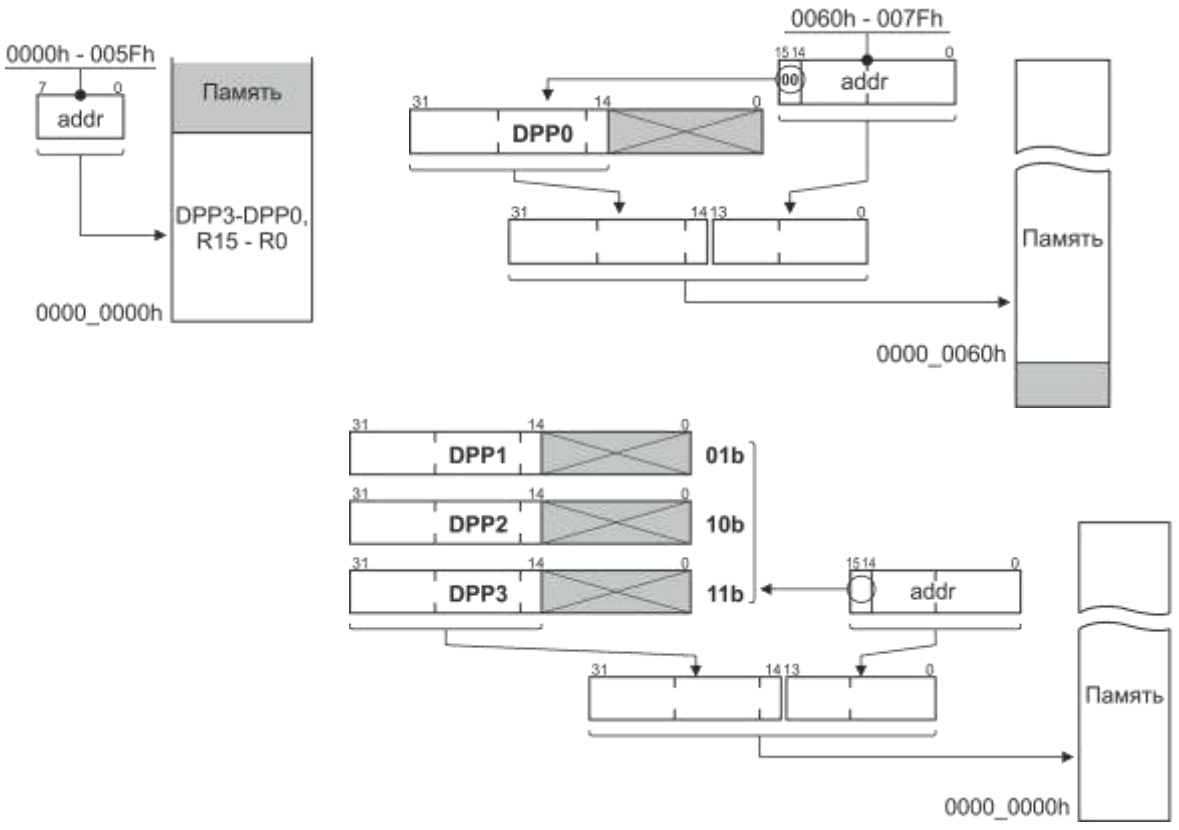


Рисунок 6.2 – Формирование адреса в 32-разрядном режиме

Особенности работы в данном режиме: определяют, какой будет использоваться указатель, уже два бита адреса команды; при операции чтения или записи по адресу, заданному косвенно, из ячейки памяти считывается уже 32-бит адреса. То есть и ячейка с указателем, и сама ячейка, откуда берутся данные для операции, – все это может находиться в любом месте адресного пространства.

Прямая регистровая адресация

Наиболее простой и быстрый способ адресации. В полях операндов на ассемблере указываются их прямые адреса. Следует помнить, что значение адреса операнда всегда расширяется до двойного слова, с использованием регистра DPP. Исключение составляют только регистры R0-R15.

Пример 1 – 16-разрядный режим:

```
ADD     CX, AX, BX  ; CX ← AX + BX
LD      AX, BX      ; AX ← BX
INC     DX          ; DX ← DX + 1
```

Регистры AX, BX, CX и DX являются 16-разрядными.

Пример 2 – 32-разрядный режим:

```
SWC     00, 01      ; Переключение в 32-разрязный режим
NOP                 ; Вставка
….                  ; не менее 16 команд NOP
NOP

ELD     AX, BX      ; AX ← BX
EINC    CX          ; CX ← CX + 1
```

Регистры AX, BX и CX являются 32-разрядными.

На рисунках 6.3 и 6.4 показаны механизмы выполнения команд пересылки слов LD и ELD для 16- и 32-разрядного режимов, соответственно. Символьное имя wreg на месте операнда указывает на то, что размер данных – слово, а символьное имя lreg – двойное слово. Полученный с использованием значения регистра DPP адрес addr_src является адресом регистра-источника данных, а адрес addr_dest – адресом регистра-приемника данных. Оба регистра могут располагаться в любом месте адресного пространства.

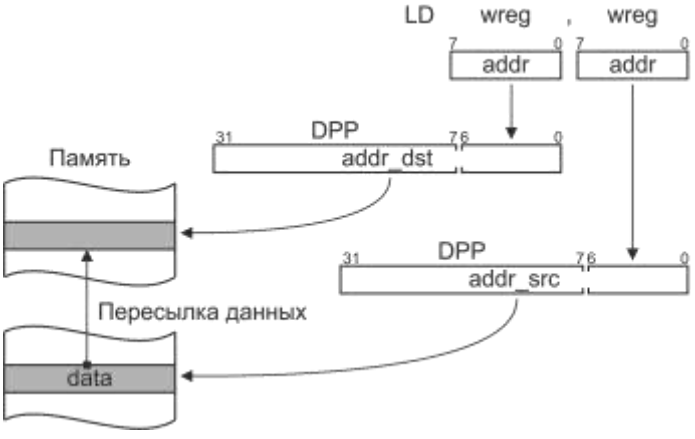


Рисунок 6.3 – Механизм команды пересылки слова с использованием прямой регистровой адресации

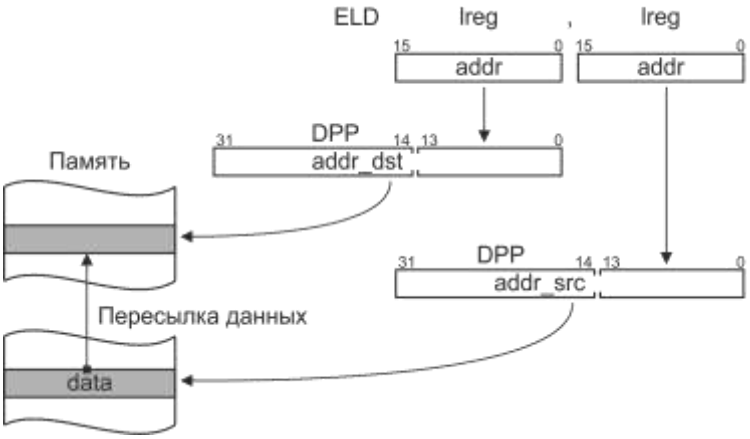


Рисунок 6.4 – Механизм команды пересылки двойного слова с использованием прямой регистровой адресации

Непосредственная адресация

Непосредственная адресация позволяет брать значение операнда непосредственно из поля команды (указывается после символа «#»). Значение операнда-источника не должно выходить за диапазон значений, соответствующий размеру операнда-приемника. Так, например, при использовании команд работы с байтами максимальное значение операнда-источника – FFh.

Следует помнить, что значение адреса операнда-приемника всегда расширяется до двойного слова, с использованием регистра DPP, поэтому он может располагаться в любом месте адресного пространства. Исключение составляют только регистры R0-R15.

Операнды могут задаваться в любой системе счисления: двоичной (b), восьмеричной (o или q), шестнадцатеричной (h) или десятичной (d или отсутствие буквы).

Пример – 16-разрядный режим:

```
ADD     AX, #0A1h    ; AX ← AX + A1h
LD      AX, #4724h   ; AX ← 4724h
```

Регистр AX является 16-разрядным.

На рисунке 6.5 показан механизм выполнения команды загрузки константы-слова.

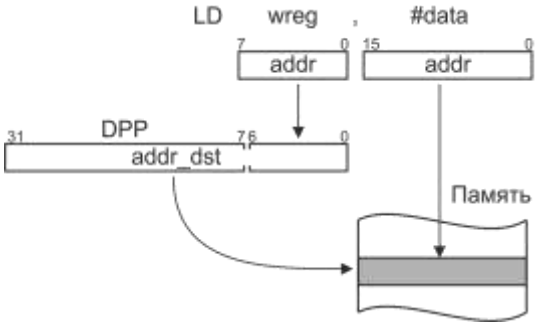


Рисунок 6.5 – Механизм команды загрузки слова с использованием непосредственной адресации

Косвенная адресация

Косвенная адресация осуществляет доступ к операнду, адрес которого размещен в переменной типа слово. Вычисляемый адрес должен соответствовать правилам выравнивания. Следует заметить, что косвенная адресация позволяет обращаться к операнду в любом месте адресного пространства. 8-битное поле reg внутри команды выбирает регистр, который содержит косвенный адрес. Команда может содержать только одну косвенную ссылку, обращение к добавочным операндам осуществляется с помощью прямой регистровой адресации.

Пример – 16-разрядный режим:

```
LD AX, [AX]          ; AX ← MEM_WORD (AX)
ADDB AL, BL, [CX]    ; AL ← BL + MEM_BYTE (CX)
```

Регистры AX и CX являются 16-разрядными.
Регистры AL, BL являются младшими байтами регистров AX и BX.

На рисунке 6.6 показан механизм выполнения команды пересылки слова LD с использованием косвенной адресации.

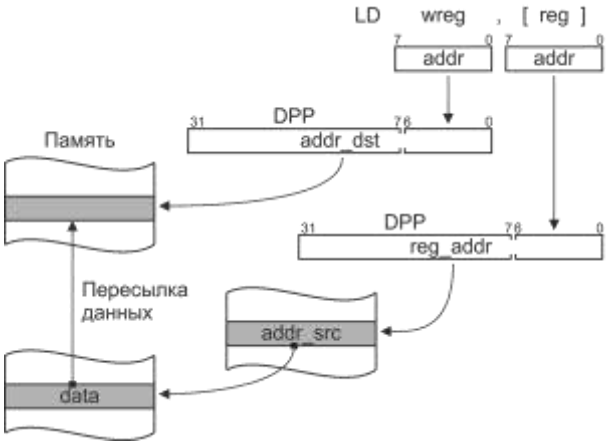


Рисунок 6.6 – Механизм команды загрузки слова с использованием косвенной адресации

Косвенная адресация с автоинкрементом

Косвенная адресация, при выполнении которой содержимое регистра указателя адреса автоматически увеличивается на единицу при работе с байтами и короткими целыми, на два – при работе со словами и целыми и на четыре – при работе с двойными словами и длинными целыми.

Пример – 16-разрядный режим:

```
LD AX, [BX]+         ; AX ← MEM_WORD(BX), BX ← BX + 2
ADDB AL, BL, [CX]+   ; AL ← BL + MEM_BYTE(CX)
                     ; CX ← CX + 1
```

Индексная адресация

При индексной адресации адрес одного из операндов вычисляется с помощью двух полей. Поле reg/ereg внутри команды выбирает регистр, который содержит косвенный

адрес. Поле offset в команде суммируется с косвенным адресом для формирования исполнительного адреса операнда addr_src.

Адрес другого операнда задается с помощью прямой регистровой адресации.

На рисунке 6.7 показан механизм выполнения команды пересылки слова LD с использованием индексной адресации.

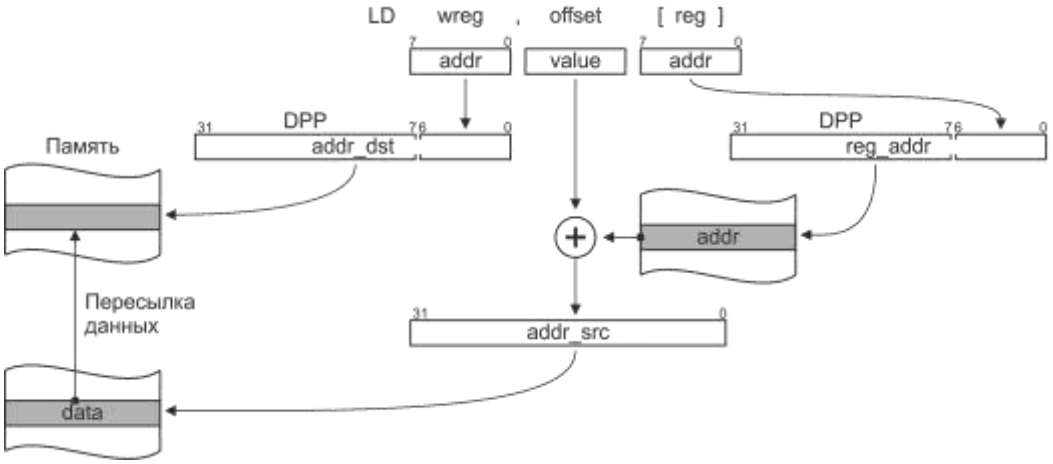


Рисунок 6.7 – Механизм команды загрузки слова LD с использованием индексной адресации

В зависимости от размера переменной offset индексная адресация может быть короткоиндексной и длинноиндексной.

Короткая индексная адресация

При короткоиндексной адресации поле offset является 8-битным, старший бит – знаковый. В связи с этим, исполнительный адрес может находиться на 128 байт ранее от косвенного адреса или на 127 байт после него.

Пример – 16-разрядный режим:

```
LD AX, 12[BX]          ; AX ← MEM_WORD (BX+12)
LD AX, –20[BX]         ; AX ← MEM_WORD (BX–20)
```

Длинная индексная адресация

Длинная индексная адресация также обеспечивает прямой доступ к любой ячейке адресного пространства. В режиме выполнения инструкций MCS-96 этот способ формирует адрес операнда добавлением 16-битного значения к регистровому слову. При этом короткая индексная адресация формирует адрес операнда добавлением 8-битного значения к регистровому слову. Регистр, хранящий начальный адрес, может находиться в любой области адресного пространства (так как адресация к нему прямая, т. е. с использованием DPP). В режиме выполнения расширенного набора 32-разрядных инструкций при длинной индексной адресации формируется адрес операнда добавлением 18-битного (18 бит – знак смещения) значения к двойному слову, хранящемуся в регистре. При этом короткая индексная адресация формирует адрес операнда добавлением 10-битного (10 бит – знак смещения) значения.

При длинноиндексной адресации поле offset является 16-битным без знакового бита. В связи с этим, исполнительный адрес может находиться в любом месте адресного пространства 0000h – FFFFh.

Следует помнить, что в 16-разрядном режиме, при вычислении исполнительного адреса, его значение не может быть более 0000_FFFFh.

Пример:
LD AX, TABLE [CX] ; AX ← MEM_WORD (CX+TABLE)

В 32-разрядном режиме, при вычислении исполнительного адреса, его значение может быть любым.

Адресация с использованием регистра нуля

Регистр нуля ZERO_REG может использоваться как переменная типа слово в длинноиндексной адресации. Такая комбинация позволяет адресоваться к любому участку памяти. Так как этот способ использует индексную адресацию, то доступ осуществляется медленнее, чем при прямой регистровой адресации.

Пример:
ADD AX, 1234 [ZERO_REG] ; AX ← AX + MEM_WORD (1234)

На рисунке 6.8 показан механизм выполнения команды пересылки слова LD с использованием индексной адресации и регистра нуля.

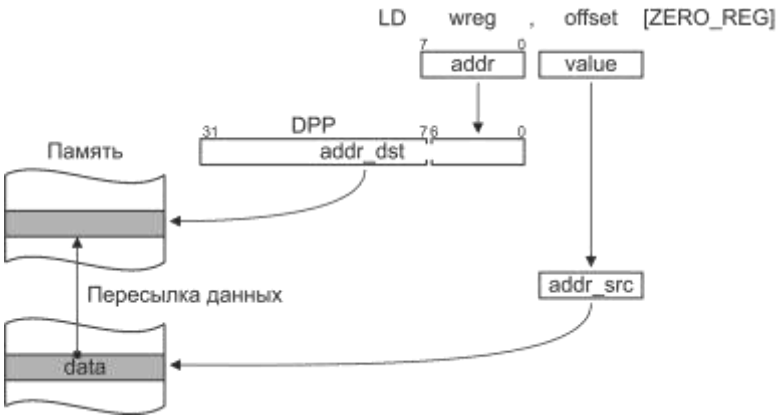


Рисунок 6.8 – Механизм команды загрузки слова с использованием индексной адресации и регистра нуля

Адресация с использованием указателя стека

Двойное слово с адресом 0000_40B0h регистрового файла содержит указатель стека SP. Кроме обычных операций указатель стека SP может также использоваться как переменная типа двойное слово при косвенной адресации для доступа к вершине стека или в короткоиндексной адресации для доступа к данным внутри стека.

Пример:
PUSH [SP] ; Копирование вершины стека
LD AX, 2[SP] ; AX ← NEXT_TO_TOP

Кроме того, имеются расширенные версии команд прерываемой и непрерываемой передачи блока.

Команда BMOV

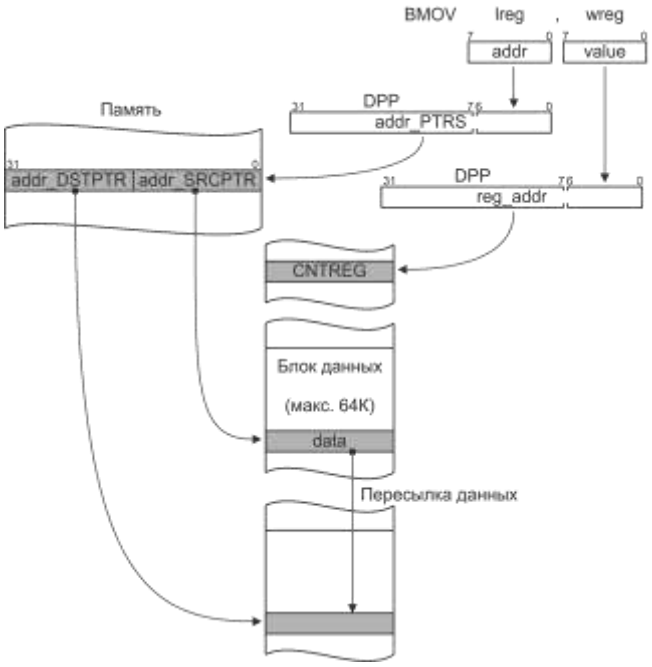


Рисунок 6.9 – Механизм команды пересылки слов BMOV (16-разрядный режим)

Команда EBMOV

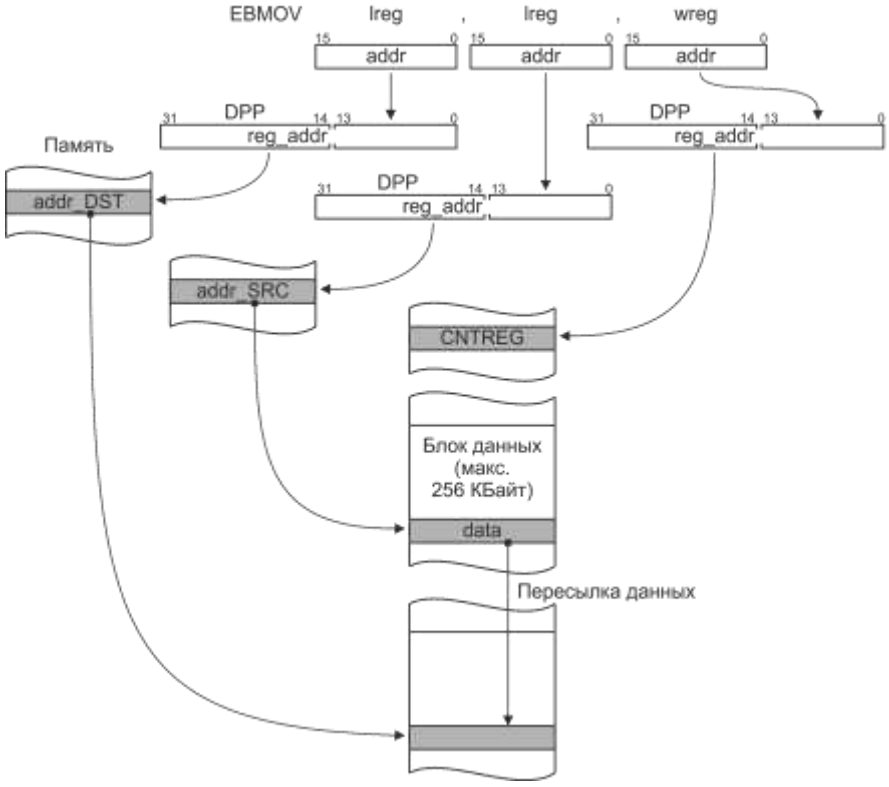


Рисунок 6.10 – Механизм команды пересылки слов EBMOV (32-разрядный режим)

Команда TIJMP

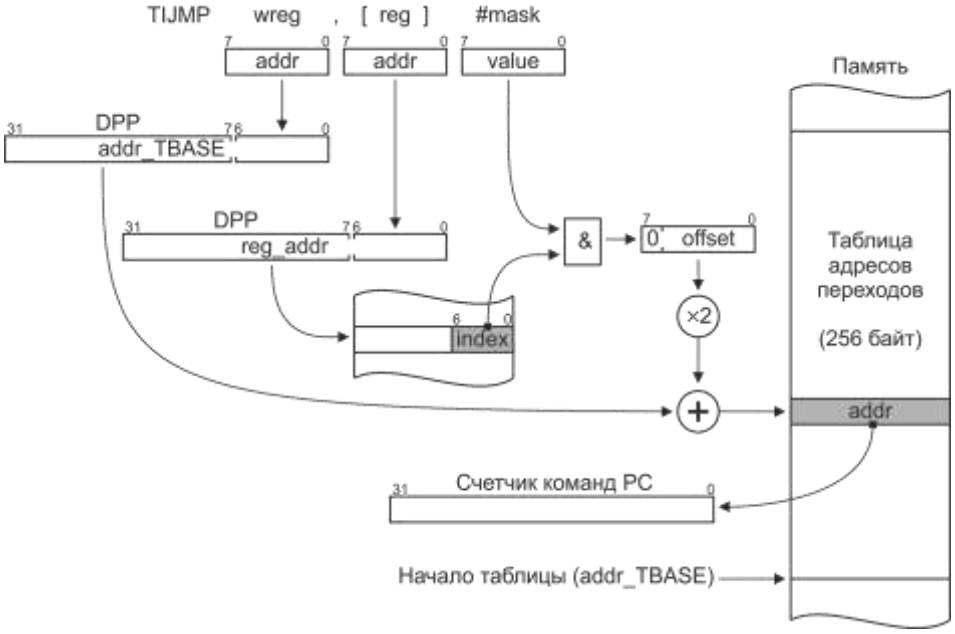


Рисунок 6.11 – Механизм команды TIJMP

7 Прерывания

Основной функцией микроконтроллера является обеспечение управления устройствами в реальном времени. Схема обслуживания прерываний позволяет событиям реального времени управлять выполнением программы. Встроенная периферия, внешний сигнал или команда могут сгенерировать запрос на прерывание. В простейшем случае ЦПУ получает запрос на прерывание, приостанавливает выполнение основной программы, осуществляет обслуживание прерывания и возвращается к прерванной программе.

Для обслуживания прерываний используются два блока: контроллер прерываний и блок периферийных транзакций (PTS).

Примечание – Описание регистров и карта адресов находятся в приложениях А и Б.

Контроллер прерываний является блоком программного обслуживания прерываний. Он имеет таблицу векторов прерываний и при возникновении запроса вызывает соответствующий обработчик. Он также используется в случае, если запрос на прерывание был направлен на PTS.

Блок периферийных транзакций PTS представляет собой блок аппаратно-микропрограммной обработки прерываний (пересылка блоков данных из одной области памяти в другую) без участия ЦПУ.

Все прерывания блока маскируются. Исключение составляет внешнее прерывание (INT63) по выводу NMI микроконтроллера, которое не имеет маски и может быть обслужено только контроллером прерываний.

Для маскируемых прерываний может быть выбран любой из двух блоков обработки прерываний.

Разрешение обслуживания прерываний контроллером прерываний включается командой EI. Команда DI запрещает обслуживание прерываний. Обслуживание прерываний блоком PTS разрешается командой EPTS. Команда DPTS запрещает блок PTS. Контроллер прерываний и блок PTS могут быть разрешены/запрещены независимо друг от друга.

Микроконтроллер поддерживает 64 вектора прерываний с именами INT0 – INT63. Приоритет прерываний установлен аппаратно и не может быть изменен. Прерывание INT63 имеет наивысший приоритет, а прерывание INT0 – низший.

Механизм обслуживания прерываний показан на рисунке 7.1 на примере обслуживания запроса INT43 от приемопередатчика UART2.

При возникновении запроса на прерывание в регистре INT_PEND0/1 устанавливается соответствующий бит. Далее если прерывание разрешено, т. е. в регистре масок INT_MASK0/1 установлен соответствующий бит, то сигнал поступает в блок управления. Блок управления прерываниями проверяет состояние битов разрешения прерываний I и PSE в регистре PSW. В случае если бит I сброшен, никакой запрос на прерывание не будет обслужен, но флаг в регистре INT_PEND0/1 будет оставаться до тех пор, пока не будет запущен механизм обслуживания прерывания. Флаг можно сбросить программно, записью нуля.

Если обслуживание прерываний разрешено (I = 1b), а PTS запрещен, то запрос на прерывание будет обслужен контроллером прерываний. Если PTS разрешен (PSE = 1b), то запрос на прерывание сначала будет обслужен блоком PTS (поскольку блок PTS имеет приоритет), а по окончании передан в контроллер прерываний.

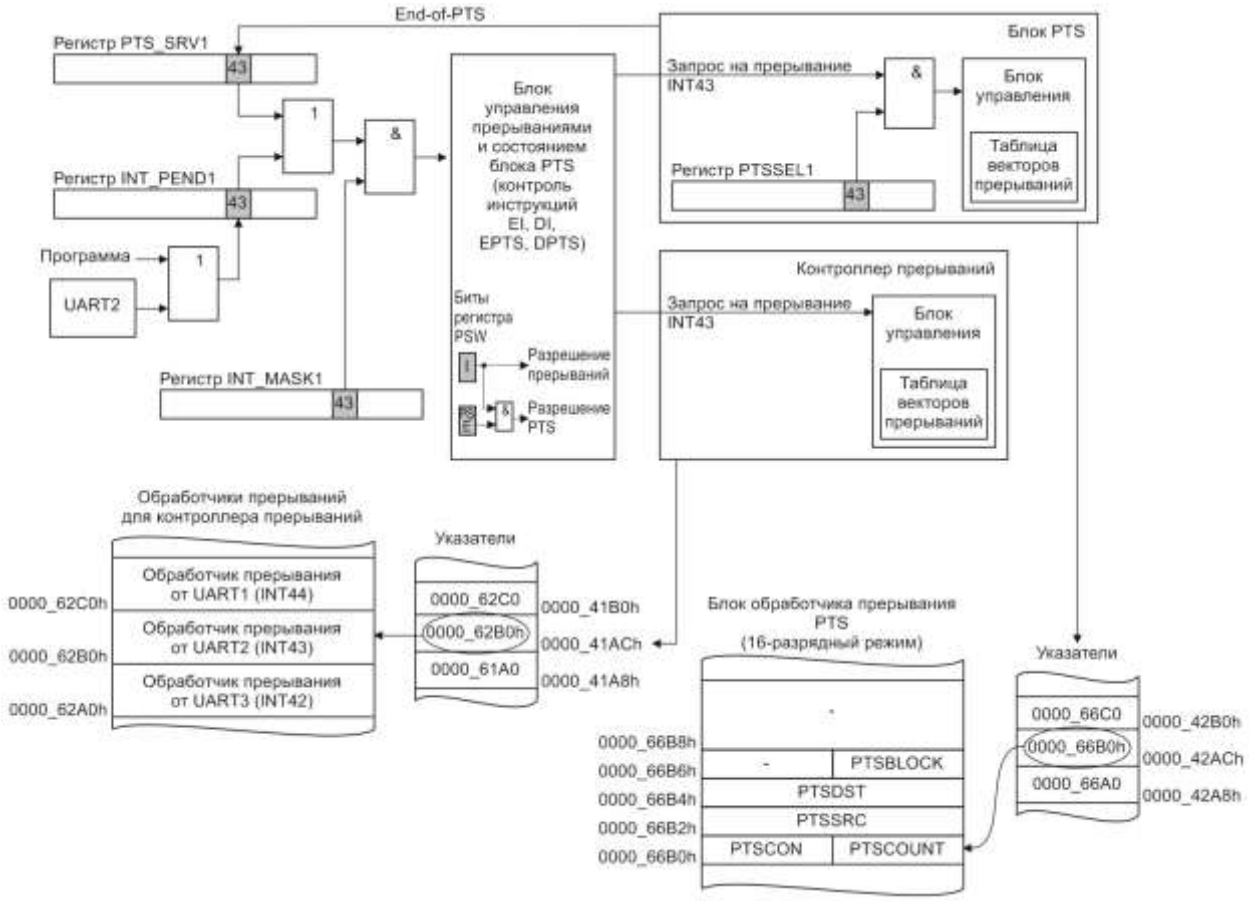


Рисунок 7.1 – Механизм обслуживания прерывания (на примере INT43 от UART2)

7.1 Контроллер прерываний

После того как запрос на прерывание был передан в контроллер прерываний соответствующий флаг в регистре INT_PEND0/1 сбрасывается аппаратно. Содержимое счетчика команд сохраняется на вершине стека. Согласно порядковому номеру запроса (INTxx) в таблице векторов выбирается регистр-указатель, который содержит адрес начала подпрограммы обработки прерывания, и этот адрес загружается в счетчик команд. Далее начинается выполнение подпрограммы. По окончании, ЦПУ проверяет состояние регистра INT_PEND0/1, и если он пуст, то в счетчик команд загружается содержимое из стека и продолжается выполнение основной программы.

Примечание – Адрес начала подпрограммы обслуживания прерывания может быть изменен пользователем и иметь любое значение в пределах доступного адресного пространства памяти программ.

7.2 Блок PTS

Поскольку блок PTS имеет приоритет над контроллером прерываний, то если он разрешен (PSE = 1b), запрос на прерывание передается ему. Для того чтобы запрос был обслужен дополнительно, должен быть установлен соответствующий бит разрешения в регистре PTSSEL0/1. Если обслуживание запроса разрешено, соответствующий флаг в регистре INT_PEND0/1 сбрасывается. Счетчик команд сохраняет текущее свое значение. Согласно порядковому номеру запроса (INTxx) в таблице векторов PTS выбирается регистр-указатель, который содержит адрес блока обслуживания прерывания.

Блок представляет собой группу регистров, расположенных в заранее определенной последовательности, которая не может быть изменена, а именно: PTSCOUNT, PTSCON, PTSSRC, PTSDST, PTSBLOCK.

Примечание – Адрес блока обслуживания прерывания может быть изменен пользователем и иметь любое значение в пределах доступного адресного пространства.

На рисунке 7.2 показана структура блока обслуживания прерываний для 16- и 32-разрядного режимов работы микроконтроллера.

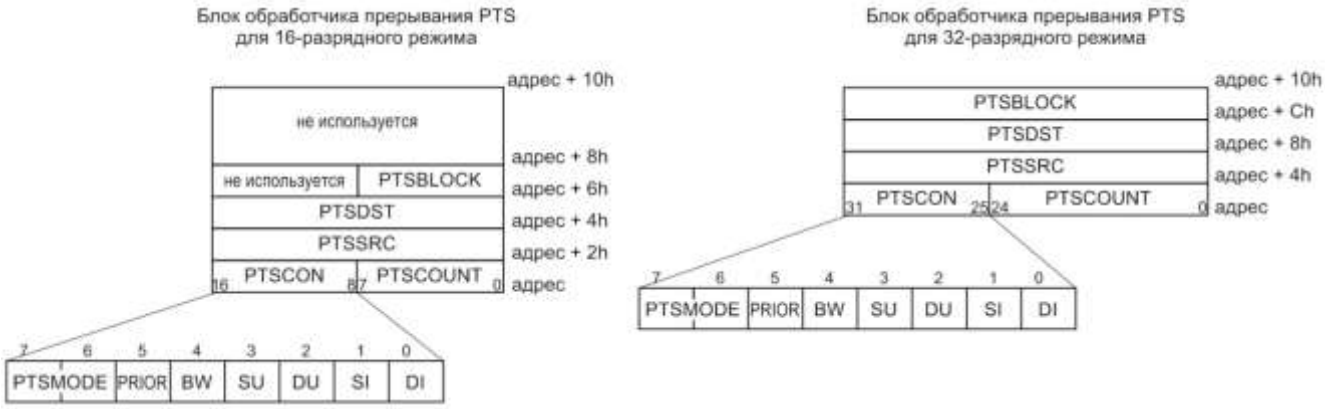


Рисунок 7.2 – Структура блока обработчика прерываний PTS для двух режимов работы микроконтроллера

Блок обслуживания прерываний управляет пересылкой (передачей) определенного количества байт/слов данных из одной области памяти в другую. Время обслуживания одного прерывания от поступления запроса до окончания пересылки является циклом. Передача может быть одиночной и блочной.

Регистры блока обслуживания прерываний

В зависимости от режима работы микроконтроллера разрядность регистров меняется, см. таблицу 7.1.

Таблица 7.1 – Разрядность регистров блока

Регистр	16-разрядный режим		32-разрядный режим	
	Разрядность	Адрес	Разрядность	Адрес
PTSCOUNT	8	адрес	24	адрес
PTSCON	8	адрес + 1h	8	адрес + 3h
PTSSRC	16	адрес + 2h	32	адрес + 4h
PTSDST	16	адрес + 4h	32	адрес + 8h
PTSBLOCK	8	адрес + 6h	32	адрес + Ch
Примечание – По умолчанию, каждому блоку выделено 16 байт адресного пространства (см. рисунок 7.2).				

Регистр PTSCOUNT задает количество раз (циклов) обслуживания прерывания от источника, которому соответствует блок. В конце каждого цикла обслуживания прерывания регистр декрементируется, и как только его значение достигнет нуля, будет сформировано прерывание End-of-PTS, которое установит в регистре PTSSRV0/1 флаг, соответствующий источнику. Как только начнется обслуживание прерывания, флаг аппаратно сбросится.

Регистр PTSCON задает параметры пересылки данных (см. рисунок 7.2):

- PTSMOD – задает тип передачи данных. При одиночной передаче за один цикл будет передано одно слово данных. При блочной передаче будет передано то количество слов, которое задается регистром PTSBLOCK;
- PRIOR – задает приоритет. Если блок PTS имеет приоритет, то во время пересылки данных ядро будет остановлено. В случае если приоритет у ядра, то ЦПУ будет вставлять циклы пересылки данных в общий поток выполнения основной программы;
- BW – задает размер одного слова данных для передачи. 8/16 бит в 16-разрядном режиме и 16/32 бит в 32-разрядном;
- SU – задает состояние регистра PTSSRC по окончании цикла передачи;
- DU – задает состояние регистра PTSDST по окончании цикла передачи;
- SI – задает действие регистра PTSSRC по окончании цикла передачи: инкрементироваться или нет;
- DI – задает действие регистра PTSDST по окончании цикла передачи: инкрементироваться или нет.

Регистр PTSSRC задает начальный адрес источника данных.

Регистр PTSDST задает начальный адрес приемника данных.

Регистр PTSBLOCK задает количество пересылаемых слов данных в одном цикле.

Одиночная передача

За один цикл от источника данных в приемник пересылается:

- один байт или одно слово в 16-разрядном режиме;
- одно слово или одно двойное слово в 32-разрядном режиме.

На этом обслуживание прерывания заканчивается, формируется сигнал End-of-PTS, который устанавливает соответствующий номеру прерывания флаг в регистре PTSSRV0/1.

Блочная передача

За один цикл от источника данных в приемник пересылается некоторое заданное количество байт/слов, после чего обслуживание прерывания заканчивается, формируется сигнал End-of-PTS, который устанавливает соответствующий номеру прерывания флаг в регистре PTSSRV.

После того как в регистре PTSSRV0/1 установился флаг и если прерывание разрешено соответствующим битом регистра INT_MASK0/1, генерируется запрос на прерывание, который передается для обслуживания в контроллер прерываний, а флаг в регистре PTSSRV сбрасывается. Далее запускается программа обслуживания прерывания, по окончании которой ЦПУ возвращается к выполнению основной программы.

Алгоритм обслуживания прерывания показан на рисунке 7.3.

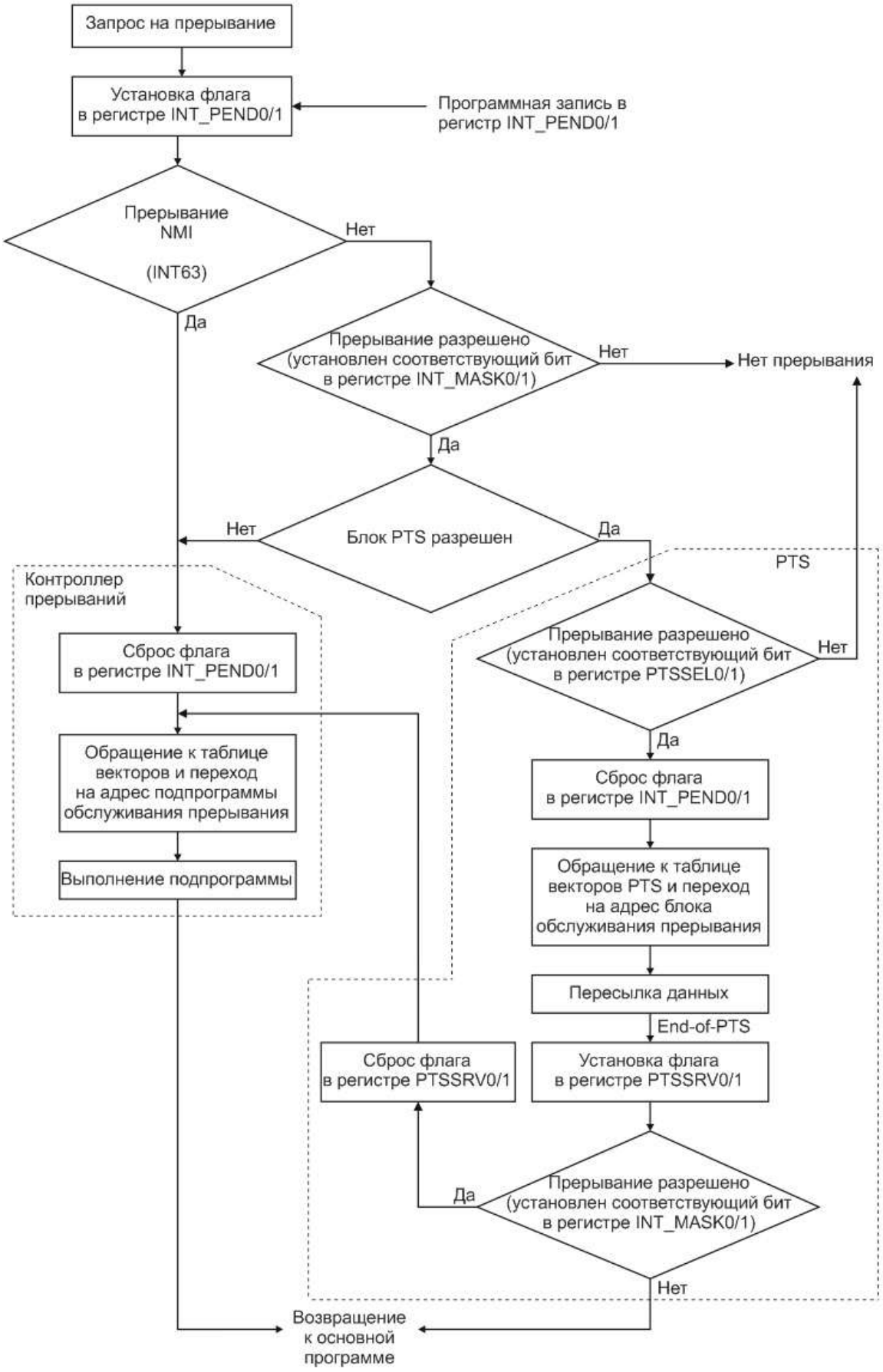


Рисунок 7.3 – Алгоритм обслуживания запроса на прерывание

Примечание – Флаги в регистрах INT_PEND0/1 и PTSSRV0/1 устанавливаются и сбрасываются аппаратно, но поскольку регистры доступны для записи и чтения, то флаги можно устанавливать и сбрасывать также программно.

Адреса указателей аппаратно зафиксированы и собраны в таблицы векторов прерываний. Начальные адреса обработчиков прерываний и блоков обработчиков прерываний PTS (на рисунке 7.2 обозначение «адрес») могут быть изменены по желанию пользователя. Таблицы векторов прерываний контроллера прерываний и блока PTS приведены в таблицах 7.2 и 7.3.

Глобальное маскирование прерываний осуществляется регистрами INT_MASK0 (прерыванию INT0 соответствует нулевой бит, прерыванию INT31 – 31 бит) и INT_MASK1 (прерыванию INT32 соответствует нулевой бит, прерыванию INT63 – 31 бит). Запросы прерываний фиксируются в регистрах INT_PEND0 и INT_PEND1. Маскирование прерываний блока PTS осуществляется регистрами PTSSEL0 (прерыванию INT0 соответствует нулевой бит, прерыванию INT31 – 31 бит) и PTSSEL1 (прерыванию INT32 соответствует нулевой бит, прерыванию INT62 – 30 бит). По окончании обслуживания по сигналу End-of-PTS запрос на прерывание фиксируется в соответствующем бите регистра PTSSRV0 или PTSSRV1.

Таблица 7.2 – Таблица векторов прерываний контроллера прерываний

Номер вектора	Прерывание, соответствующее вектору		Адрес указателя	Адрес обработчика прерывания
	Название	Источник		
INT63	NMI	Немаскируемое	41FCh	63F0h
INT62	TRAP1	Программное1	41F8h	63E0h
INT61	TRAP	Программное	41F4h	63D0h
INT60	UNIMPLEMENT	Невыполнимый код	41F0h	63C0h
INT59	ECC_DATA_ERR	Ошибка данных	41ECh	63B0h
INT58	ECC_COMM_ERR	Ошибка команды	41E8h	63A0h
INT57	ECC_RAM_ERR	Ошибка ОЗУ	41E4h	6390h
INT56	ECC_PSRAM_ERR	Ошибка PSRAM	41E0h	6380h
INT55	STACKOVER_ERR	Переполнение стека	41DCh	6370h
INT54	STACKDOWN_ERR	Опустошение стека	41D8h	6360h
INT53	WALIGN_ERR	Некорректный адрес	41D4h	6350h
INT52	DEBUG	OCDS	41D0h	6340h
INT51	WDT	Сторожевой таймер	41CCh	6330h
INT50	TIMER0	Системный таймер T0	41C8h	6320h
INT49	TIMER1	Системный таймер T1	41C4h	6310h
INT48	FPU_TRAP	Блок FPU	41C0h	6300h
INT47	SPWR0	SpaceWire0	41BCh	62F0h
INT46	SPWR1	SpaceWire1	41B8h	62E0h
INT45	UART0	UART0	41B4h	62D0h
INT44	UART1	UART1	41B0h	62C0h
INT43	UART2	UART2	41ACh	62B0h
INT42	UART3	UART3	41A8h	62A0h
INT41	SPI0	SPI0	41A4h	6290h
INT40	SPI1	SPI1	41a0h	6280h

Окончание таблицы 7.2

Номер вектора	Прерывание, соответствующее вектору		Адрес указателя	Адрес обработчика прерывания
	Название	Источник		
INT39	BSI0	МПИ0	419Ch	6270h
INT38	BSI1	МПИ1	4198h	6260h
INT37	BSI0_ECC_ERR	МПИ0	4194h	6250h
INT36	BSI1_ECC_ERR	МПИ1	4190h	6240h
INT35	ARINC1_0	ARINC0 или ARINC1	418Ch	6230h
INT34	ARINC3_2	ARINC2 или ARINC3	4188h	6220h
INT33	ARINC5_4	ARINC4 или ARINC5	4184h	6210h
INT32	ARINC7_6	ARINC6 или ARINC7	4180h	6200h
INT31	GPIO_LINE15	Порты	417Ch	61F0h
INT30	GPIO_LINE14	Порты	4178h	61E0h
INT29	GPIO_LINE13	Порты	4174h	61D0h
INT28	GPIO_LINE12	Порты	4170h	61C0h
INT27	GPIO_LINE11	Порты	416Ch	61B0h
INT26	GPIO_LINE10	Порты	4168h	61A0h
INT25	GPIO_LINE9	Порты	4164h	6190h
INT24	GPIO_LINE8	Порты	4160h	6180h
INT23	GPIO_LINE7	Порты	415Ch	6170h
INT22	GPIO_LINE6	Порты	4158h	6160h
INT21	GPIO_LINE5	Порты	4154h	6150h
INT20	GPIO_LINE4	Порты	4150h	6140h
INT19	GPIO_LINE3	Порты	414Ch	6130h
INT18	GPIO_LINE2	Порты	4148h	6120h
INT17	GPIO_LINE1	Порты	4144h	6110h
INT16	GPIO_LINE0	Порты	4140h	6100h
INT15	QEP0	Квадратурный декодер 0	413Ch	60F0h
INT14	QEP1	Квадратурный декодер 1	4138h	60E0h
INT13	I2C	I2C	4134h	60D0h
INT12	PWM0	Блок ШИМ0	4130h	60C0h
INT11	PWM1	Блок ШИМ1	412Ch	60B0h
INT10	PWM2	Блок ШИМ2	4128h	60A0h
INT9	ADC	АЦП	4124h	6090h
INT8	HSO	HSO	4120h	6080h
INT7	HSI	HSI	411Ch	6070h
INT6	–	–	4118h	6060h
INT5	–	–	4114h	6050h
INT4	–	–	4110h	6040h
INT3	–	–	410Ch	6030h
INT2	–	–	4108h	6020h
INT1	–	–	4104h	6010h
INT0	–	–	4100h	6000h

Таблица 7.3 – Таблица векторов прерываний блока PTS

Номер вектора	Прерывание, соответствующее вектору		Адрес указателя	Адрес блока
	Название	Источник		
INT62	TRAP1	Программное1	42F8h	67E0h
INT61	TRAP	Программное	42F4h	67D0h
INT60	UNIMPLEMENT	Невыполнимый код	42F0h	67C0h
INT59	ECC_DATA_ERR	Ошибка данных	42ECh	67B0h
INT58	ECC_COMM_ERR	Ошибка команды	42E8h	67A0h
INT57	ECC_RAM_ERR	Ошибка ОЗУ	42E4h	6790h
INT56	ECC_PSRAM_ERR	Ошибка PSRAM	42E0h	6780h
INT55	STACKOVER_ERR	Переполнение стека	42DCh	6770h
INT54	STACKDOWN_ERR	Опустошение стека	42D8h	6760h
INT53	WALIGN_ERR	Некорректный адрес	42D4h	6750h
INT52	DEBUG	OCDS	42D0h	6740h
INT51	WDT	Сторожевой таймер	42CCh	6730h
INT50	TIMER0	Системный таймер T0	42C8h	6720h
INT49	TIMER1	Системный таймер T1	42C4h	6710h
INT48	FPU_TRAP	Блок FPU	42C0h	6700h
INT47	SPWR0	SpaceWire0	42BCh	66F0h
INT46	SPWR1	SpaceWire1	42B8h	66E0h
INT45	UART0	UART0	42B4h	66D0h
INT44	UART1	UART1	42B0h	66C0h
INT43	UART2	UART2	42ACh	66B0h
INT42	UART3	UART3	42A8h	66A0h
INT41	SPI0	SPI0	42A4h	6690h
INT40	SPI1	SPI1	42a0h	6680h
INT39	BSI0	МПИ0	429Ch	6670h
INT38	BSI1	МПИ1	4298h	6660h
INT37	BSI0_ECC_ERR	МПИ0	4294h	6650h
INT36	BSI1_ECC_ERR	МПИ1	4290h	6640h
INT35	ARINC1_0	ARINC0 или ARINC1	428Ch	6630h
INT34	ARINC3_2	ARINC2 или ARINC3	4288h	6620h
INT33	ARINC5_4	ARINC4 или ARINC5	4284h	6610h
INT32	ARINC7_6	ARINC6 или ARINC7	4280h	6600h
INT31	GPIO_LINE15	Порты	427Ch	65F0h
INT30	GPIO_LINE14	Порты	4278h	65E0h
INT29	GPIO_LINE13	Порты	4274h	65D0h
INT28	GPIO_LINE12	Порты	4270h	65C0h
INT27	GPIO_LINE11	Порты	426Ch	65B0h
INT26	GPIO_LINE10	Порты	4268h	65A0h
INT25	GPIO_LINE9	Порты	4264h	6590h
INT24	GPIO_LINE8	Порты	4260h	6580h
INT23	GPIO_LINE7	Порты	425Ch	6570h
INT22	GPIO_LINE6	Порты	4258h	6560h
INT21	GPIO_LINE5	Порты	4254h	6550h
INT20	GPIO_LINE4	Порты	4250h	6540h
INT19	GPIO_LINE3	Порты	424Ch	6530h

Окончание таблицы 7.3

Номер вектора	Прерывание, соответствующее вектору		Адрес указателя	Адрес блока
	Название	Источник		
INT18	GPIO_LINE2	Порты	4248h	6520h
INT17	GPIO_LINE1	Порты	4244h	6510h
INT16	GPIO_LINE0	Порты	4240h	6510h
INT15	QEP0	Квадратурный декодер 0	423Ch	6400h
INT14	QEP1	Квадратурный декодер 1	4238h	6400h
INT13	I2C	I2C	4234h	6400h
INT12	PWM0	Блок ШИМ0	4230h	6400h
INT11	PWM1	Блок ШИМ1	422Ch	6400h
INT10	PWM2	Блок ШИМ2	4228h	6400h
INT9	ADC	АЦП	4224h	6400h
INT8	HSO	HSO	4220h	6400h
INT7	HSI	HSI	422Ch	6400h
INT6	–	–	4228h	6400h
INT5	–	–	4224h	6400h
INT4	–	–	4220h	6400h
INT3	–	–	420Ch	6400h
INT2	–	–	4208h	6420h
INT1	–	–	4204h	6410h
INT0	–	–	4200h	6400h

8 Блок вычислений с плавающей запятой FPU

Высокопроизводительный блок FPU арифметики с плавающей запятой обрабатывает числа с плавающей запятой в форматах с одинарной и двойной точностью (согласно стандарту IEEE-754), за исключением денормализованных чисел.

Управление блоком FPU и загрузка исходных данных осуществляется центральным процессорным устройством.

8.1 Состав блока

В состав блока FPU входят логика управления и блок вычислений, включающий два исполнительных модуля для выполнения операций над числами – вычислитель с конвейером и модуль итераций. Ячейки памяти для хранения исходных данных и результатов вычислений, а также регистры управления и состояния расположены в ОЗУ0, занимают область адресов с 0000_4300h по 0000_438Fh и доступны для чтения и записи.

На рисунке 8.1 показана общая схема взаимодействия блоков FPU, ЦПУ и ОЗУ0.

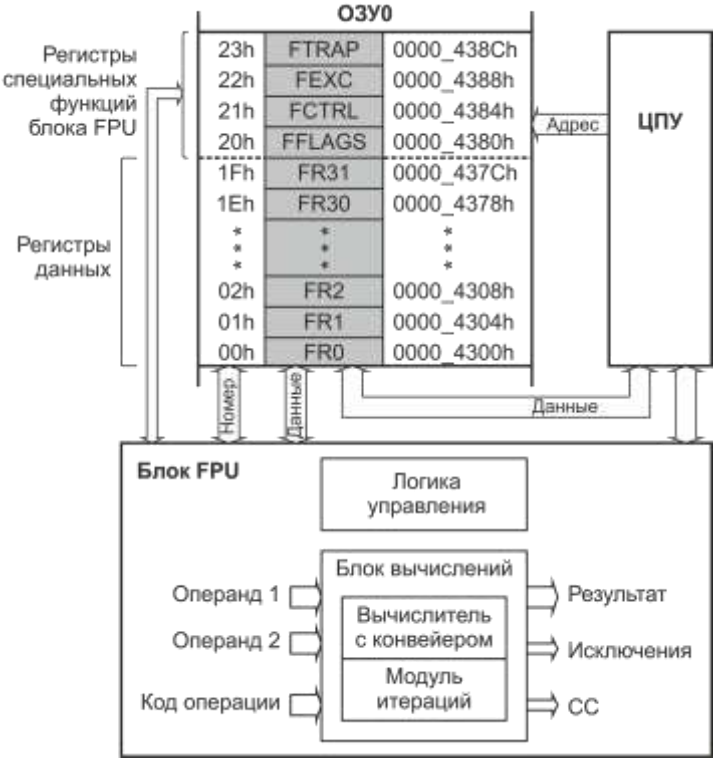


Рисунок 8.1 – Общая схема взаимодействия блоков FPU, ЦПУ и ОЗУ0

ОЗУ данных блока FPU является частью ОЗУ0 микроконтроллера и представляет собой набор 32-разрядных регистров FR0 – FR31. При операциях в формате с одинарной точностью в качестве операндов и приемника могут использоваться любые регистры. При операциях в формате с двойной точностью в качестве операндов и приемника могут использоваться только пары регистров, причем младший регистр в паре всегда четный (например, FR0-FR1, FR22-FR24).

Инструкции блока FPU могут выполняться одновременно с инструкциями целочисленной арифметики, реализуемой ЦПУ. Конвейер останавливается только в случае зависимости по данным или ресурсам.

8.2 Операции с плавающей запятой

Блок FPU поддерживает четыре типа операций с плавающей запятой:
- арифметические;
- преобразования типов данных;
- сравнения;
- дополнительные (абсолютное значение, отрицание, перемещение данных).

Краткие сведения об указанных оперциях приведены в таблицах 8.1 – 8.6. Подробная информация о системе команд блока FPU представлена в приложении Д.

В таблицах 8.1 – 8.6 приняты обозначения:

- последняя буква в мнемоническом обозначении команды указывает на формат выполняемой операции (S – одинарной точности, D – двойной точности);
- SP (формат) – число с плавающей запятой одинарной точности (32-разрядное);
- DP (формат) – число с плавающей запятой двойной точности (64-разрядное);
- INT (формат) – целое число (32-разрядное);
- UNF, NV, OF, UF, NX, DZ – типы исключительных ситуаций, которые могут возникнуть в результате выполнения операции (будут подробнее рассмотрены ниже);
- CC – код условия, см. регистр FFLAGS;
- NaN – неопределенность (особое состояние числа с плавающей запятой);
- qNaN – тихий NaN;
- sNaN – сигнальный NaN.

Таблица 8.1 – Арифметические операции

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Результат	Исключения	Описание
FADDS	E508h	SP	SP	SP	UNF, NV, OF, UF, NX	Сложение
FADDD	E509h	DP	DP	DP		
FSUBS	E50Ah	SP	SP	SP	UNF, NV, OF, UF, NX	Вычитание
FSUBD	E50Bh	DP	DP	DP		
FMULS	E50Ch	SP	SP	SP	UNF, NV, OF, UF, NX	Умножение
FMULD	E50Dh	DP	DP	DP		
FSMULD	E50Eh	SP	SP	DP	UNF, NV, OF, UF	
FDIVS	E50Fh	SP	SP	SP	UNF, NV, OF, UF, NX, DZ	Деление
FDIVD	E510h	DP	DP	DP		
FSQRTS	E511h	–	SP	SP	UNF, NV, NX	Квадратный корень
FSQRTD	E512h	–	DP	DP		

Таблица 8.2 – Операции преобразования типов данных

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Результат	Исключе-ния	Описание
FITOS	E513h	–	INT	SP	NX	Преобразование целого числа в число с плавающей запятой
FITOD	E514h	–		DP	–	
FSTOI	E515h	–	SP	INT	UNF, NV, NX	Преобразование числа с плавающей запятой в целое с округлением к нулю
FDTOI	E516h	–	DP			

Окончание таблицы 8.2

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Результат	Исключения	Описание
FSTOI_RND	E517h	–	SP	INT	UNF, NV, NX	Преобразование числа с плавающей запятой в целое с округлением. Режим округления задается пользователем
FDTOI_RND	E518h	–	DP			
FSTOD	E519h	–	SP	DP	UNF, NV	Преобразование формата числа с плавающей запятой
FDTOS	E51Ah	–	DP	SP	UNF, NV, OF, UF, NX	

Таблица 8.3 – Операции сравнения

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Результат	Исключения	Описание
FCMPS	E51Bh	SP	SP	CC	NV (возникает, если один из операндов sNaN)	Сравнение чисел с плавающей запятой
FCMPD	E51Ch	DP	DP			
FCMPES	E51Dh	SP	SP	CC	NV (возникает, если один из операндов NaN (тихий или сигнальный)	
FCMPED	E51Eh	DP	DP			

Таблица 8.4 – Дополнительные операции

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Результат	Исключения	Описание
FABSS	E51Fh	–	SP	SP	–	Получение абсолютного значения
FNEGS	E520h	–	SP	SP	–	Отрицание
FMOVS	E521h	–	SP	SP	–	Пересылка числа

Примечание – Операции с плавающей запятой выполняются блоком FPU, только если он включен (установлен бит FPUEN ргистра FCTRL). В противном случае никаких действий блок FPU не производит.

Система команд микроконтроллера расширена дополнительными командами пересылки данных, которые позволяют обращаться к регистрам FR0 – FR31 по их порядковым номерам (см. таблицу 8.5), а также командами условных переходов, позволяющими выполнять ветвления при выполнении программы пользователя в зависимости от результатов операций сравнения блока FPU (см. таблицу 8.6).

Примечание – Результат операции сравнения отражается в поле CC регистра FFLAGS.

Таблица 8.5 – Команды пересылки данных

Мнемоника команды	Код команды	Операнд 1	Операнд 2	Описание
FLD EFLD	E500h-E503h	FR	mem	Пересылка числа в регистр FR из любой ячейки памяти доступного адресного пространства
FSTFR	E504h	FR	FR	Пересылка числа из одного регистра в другой в пределах области ОЗУ0, выделенной блоку FPU
FST EFST	E505h-E507h	FR	mem	Пересылка числа в FPU (DEST, SRC)

Примечания

1 Дополнительные команды FLD, EFLD, FST и EFST являются аналогами команд LD, ELD, ST и EST, соответственно, с той разницей, что в дополнительных командах одним из операндов является регистр FR и в кодах команд используется его порядковый номер.

2 Для команды FLD/EFLD поддерживается прямая, непосредственная, косвенная и индексная адресация.

3 Для команды FST/EFST поддерживается прямая, косвенная и индексная адресация.

4 Команды FLD, EFLD, FST и EFST выполняются всегда, независимо от состояния (включен/выключен/занят) блока FPU.

5 Команда FSTFR является полным аналогом команды FMOVS, с тем отличием, что выполняется всегда, независимо от состояния (включен/выключен/занят) блока FPU.

Таблица 8.6 – Команды условных переходов (условие определяется состоянием поля CC регистра FFLAGS)

Мнемоника команды	Код команды	Операнд	Описание
FJNH	E5D1h	cadd	Переход, если CC = 10b
FJNE	E5D7h	cadd	Переход, если CC = 11b
FJH	E5D9h	cadd	Переход, если CC = 01b
FJE	E5DFh	cadd	Переход, если CC = 00b

Примечания

1 Для команд справедливо: если условие выполняется, то к текущему значению программного счетчика PC добавляется смещение между концом команды и адресом метки перехода (cadd), иначе управление передается следующей по порядку команде.

2 Команды выполняются всегда, не зависимо от состояния (включен/выключен/занят) блока FPU.

8.3 Функционирование блока

Включение блока FPU

Для разрешения работы блока следует установить бит FPUEN в регистре FCTRL.

В выключенном состоянии блок FPU не выполняет никаких операций.

Блок управления FPU

Блок управления FPU обеспечивает планирование, декодирование и диспетчеризацию операций с плавающей запятой для FPU, а также управление набором регистров для работы с плавающей запятой, регистром состояния операций с плавающей запятой (FSR) и

очередью отложенных прерываний операций с плавающей точкой (FQ). Выполнение операций с плавающей запятой идет параллельно с осуществлением целочисленных инструкций, целочисленный конвейер останавливается только в случае конфликтов операндов или результата.

Подготовка данных и параметров для вычисления

Блок FPU оперирует только с регистрами данных FR0-FR31 и обращается к ним по их порядковым номерам. Параметры операций задаются и контролируются посредством регистров специальных функций блока FPU.

Преобразование типов

Блок FPU выполняет арифметические, сравнения и дополнительные операции только над нормализованными числами, т. е. в регистрах, которые используются в качестве операндов, числа должны быть нормализованными.

Получить нормализованное число одинарной или двойной точности из целого 32-разрядного числа можно использованием команды преобразования.

Пример преобразования десятичных чисел 30 (1Eh) и 6 (6h).

```
FELD   FR0, #1E      ; загрузка константы в регистр
FITOS  FR0, FR0      ; преобразование целого числа в нормализованное число
                     ; одинарной точности и сохранение результата в FR0

FELD   FR3, #6
FITOS  FR4, FR3      ; преобразование целого числа в нормализованное число
                     ; одинарной точности и сохранение результата в FR4
```

Нормализованные числа одинарной точности являются 32-разрядными. Поэтому для их хранения можно использовать любые регистры из набора FR0–FR31.

Нормализованные числа двойной точности являются 64-разрядными. Поэтому для хранения каждого такого числа требуется пара регистров, например, FR0-FR1 (младший-старший). Порядковый номер младшего регистра должен быть четным.

```
FELD   FR0, #1E      ; загрузка константы в регистр
FITOD  FR0, FR0      ; преобразование целого числа в нормализованное число
                     ; двойной точности и сохранение результата в FR0-FR1
FELD   FR3, #6
FITOD  FR4, FR3      ; преобразование целого числа в нормализованное число
                     ; двойной точности и сохранение результата в FR4-FR5
```

Примечание – При вычислениях с использованием чисел двойной точности нельзя указывать регистры с нечетными номерами:

```
FSQRTD FR10, FR1   ; НЕКОРРЕКТНО задан регистр-операнд (FR1)
FSQRTD FR9, FR0    ; НЕКОРРЕКТНО задан регистр-приемник (FR9)
```

Полученные в результате вычислений нормализованные числа могут быть преобразованы в 32-разрядные целые числа с учетом режима округления, который задается полем RNDMODE регистра FCTRL.

Блок FPU поддерживает четыре режима округления: к ближайшему целому, к нулю, к $+\infty$ и к $-\infty$.

```
FDTOI_RND   FR10, FR8    ; преобразование с заданным режимом округления
                         ; нормализованного числа двойной точности
```

; в целое 32-разрядное число
; и сохранение результата в FR10

Команды FSTOI и FDTOI выполняют преобразование нормализованного числа в целое в режиме округления к нулю (этот режим задан по умолчанию, содержимое поля RNDMODE не важно).

Конвейер

Блок FPU конвейеризирован (конвейер имеет три ступени: загрузка, вычисление, результат) и обеспечивает старт одной инструкции при каждом цикле тактирования, за исключением FDIV и FSQRT, которые выполняются только поочередно. FDIV и FSQRT выполняются в отдельном делителе и не блокируют FPU при выполнении других параллельных операций.

Все инструкции, кроме FDIV и FSQRT, имеют период ожидания в три цикла, но для улучшения синхронизации контроллер FPU вставляет дополнительный цикл конвейера в канал передачи результата. Это обеспечивает период ожидания в четыре цикла тактирования на уровне инструкций. В таблице 8.7 представлена производительность блока FPU, а также время прохождения инструкции по конвейеру.

Таблица 8.7 – Производительность GRFPU с GRFPC

Инструкция	Пропускная способность	Период ожидания
FADDS, FADDD, FSUBS, FSUBD, FMULS, FMULD, FSMULD, FITOS, FITOD, FSTOI, FSTOI_RND, FDTOI, FDTOI_RND, FSTOD, FDTOS, FCMPS, FCMPD, FCMPES, FCMPED	1	4
FDIVS	14	16
FDIVD	15	17
FSQRTS	22	24
FSQRTD	23	25

Сложение, вычитание и умножение можно осуществлять каждый такт, обеспечивая высокую производительность для этих наиболее распространенных операций. Операции деления и вычисления квадратного корня имеют более низкую производительность и большую задержку ожидания ввиду сложности алгоритмов, но выполняются параллельно с другими операциями с плавающей запятой в неблокирующем итерационном модуле.

На рисунках 8.2 – 8.4 представлена работа конвейера при выполнении различных операций.

```
; преобразование

FITOD FR0, FR0
FITOD FR2, FR2
```

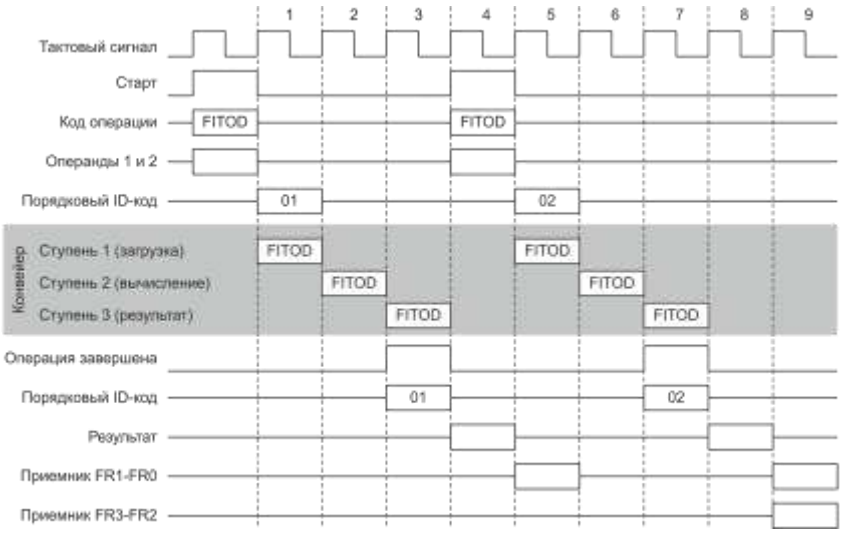


Рисунок 8.2 – Операции преобразования

; сложение и умножение

FADDD FR0, FR8, FR6
FMULD FR2, FR8, FR6

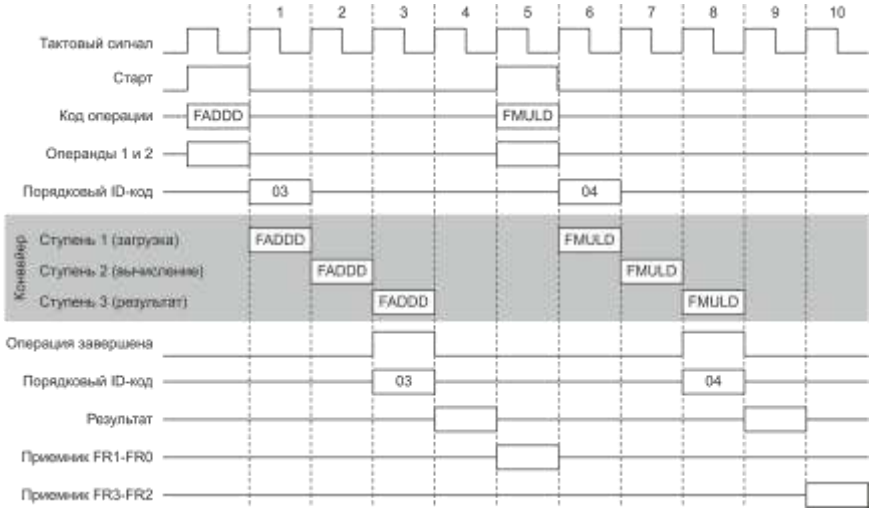


Рисунок 8.3 – Операции сложения и умножения

Такт 5. Запись результата и загрузка операндов для команды FMULD происходят одновременно. Результат от FADDD не готов. Нужен NOP:

```
FDIVD   FR4, FR24, FR26
FSUBD   FR0, FR18, FR16
NOP
NOP
NOP
FMULD    FR2, FR18, FR16
```

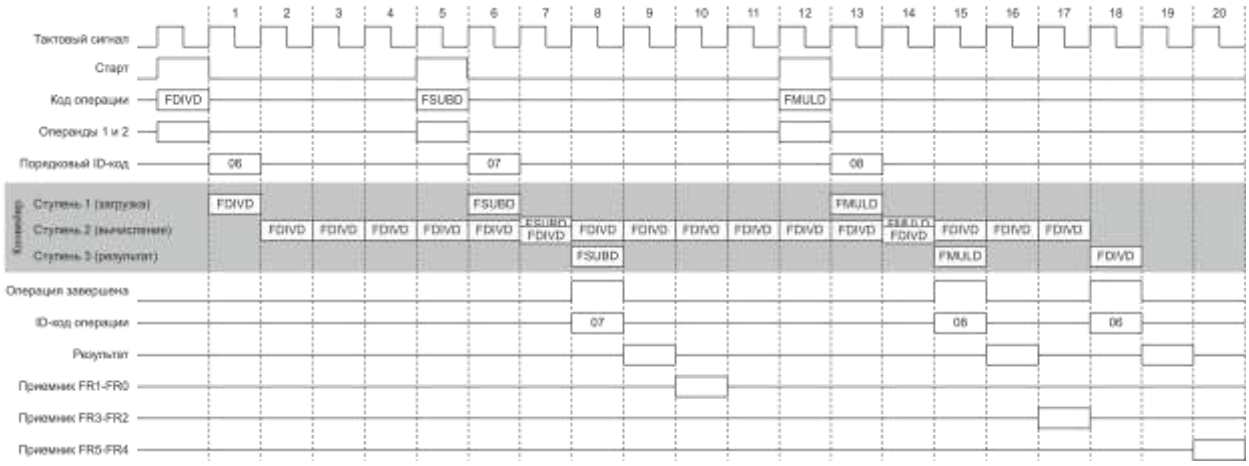


Рисунок 8.4 – Операции деления, вычитания и умножения

Выполнение операций с изменением их очередности и с различными периодами ожидания осуществляется через интерфейс FPU, который присваивает каждой операции, выдающей результат на выход, соответствующий идентификатор, как только она завершена.

Операции преобразования типов данных выполняются в модуле с конвейерной организацией и имеют производительность в один такт и период ожидания – в четыре такта.

Доступны функции сравнения, обеспечивающие обработку двух различных чисел типа QNaNs.

Также доступно перемещение, отрицание и получение абсолютного значения. Эти операции никогда не генерируют исключение незаконченности (сигнал о возникновении исключения незаконченности никогда не передается, так как при сравнении, преобразовании, получении абсолютного значения и перемещении обрабатываются денормализованные числа).

Исключения

Блок FPU обнаруживает исключения, определенные в стандарте IEEE-754:
- включая обнаружение недопустимых операций (NV);
- переполнение (OF);
- обнуление значащих разрядов (UF);
- деление на ноль (DZ);
- погрешность (NX).

Также реализована генерация специальных результатов таких, как NaNs и бесконечность.

Переполнение (OF) и обнуление значащих разрядов (UF) обнаруживаются перед округлением. Если в процессе выполнения операции происходит обнуление значащих разрядов, результат обнуляется (FPU не поддерживает денормализованные числа или постепенное обнуление значащих разрядов). Если один из компонентов – денормализованное число, не подлежащее обработке операциями преобразования или арифметическими операциями, подается сигнал о специальном исключении незаконченности

Денормализованные числа

Можно эмулировать редкие случаи операций с денормализованными числами, используя операции без плавающей запятой. Операции сравнения, перемещения, отрицания и получения абсолютного значения могут обрабатывать денормализованные числа, при этом не возникают исключения незаконченности. FPU не генерирует денормализованные числа во время выполнения арифметических операций и операций преобразования нормализованных чисел. Если точный результат операции – маленькое число (меньше, чем минимальное значение в нормальном формате), результат обнуляется (с обнулением значащих разрядов и установкой флагов погрешности).

Нестандартный режим

Блок FPU может работать в нестандартном режиме, где все денормализованные компоненты арифметических операций и операций преобразования рассматриваются, как нули со знаком. Вычисления выполняются с нулевыми компонентами, а не с денормализованными числами, с соблюдением всех правил арифметики с плавающей запятой, включая округление результатов и обнаружение исключений.

NaN

Блок FPU поддерживает обработку чисел типа NaN, как определено в стандарте IEEE-754. Операции при поступлении сигнала о NaN (тихий или сигнальный) и недопустимых операциях (например, деление бесконечности на бесконечность) генерируют недопустимое исключение и выводят в качестве результата тихий NaN. Операции над тихими NaN (qNaN), за исключением FCMPES и FCMPED, не выдают исключения и распространяют числа qNaN через операции с плавающей запятой, представляя результаты NaN согласно таблице 8.8.

Число «тихий NaN» равно 7FFFE00000000000h для результатов с удвоенной точностью и 7FFF0000 для результатов с одинарной точностью.

Таблица 8.8 – Операции с NaN (серым цветом отмечены ячейки с результатом)

		Операнд 2		
		FP	qNaN2	sNaN2
Операнд 1	Отсутствует	FP	qNaN2	qNaN
	FP	FP	qNaN2	qNaN
	qNaN1 (тихий)	qNaN1	qNaN2	qNaN
	sNaN1 (сигнальный)	qNaN	qNaN	qNaN

9 Таймеры

Микроконтроллер содержит два идентичных 64-разрядных таймера T0 и T1.

Инкрементный счетчик таймера синхронизируется сигналом системной тактовой частоты (SysCLK).

После сброса микроконтроллера счетчик находится в нулевом состоянии. При желании пользователь может изменить состояние счетчика записью любого значения в регистр TIM(0/1), который состоит из двух 32-разрядных регистров TIM(0/1)_HI и TIM(0/1)_LO.

После включения счетчик инкрементируется до тех пор, пока не достигнет значения, находящегося в регистре перезагрузки TIM(0/1)LOAD, который состоит из двух 32-разрядных регистров TIM(0/1)LOAD_HI и TIM(0/1)LOAD_LO. После сброса все биты регистра перезагрузки содержат единицы.

Как только счетчик достигает значения перезагрузки, он обнуляется, и при этом генерируется прерывание tim(0/1).

Включение и выключение таймера контролируется битом ON регистра TIM(0/1)CTRL. Запись нуля в бит ON останавливает счетчик, но не сбрасывает его.

Примечание – Таймер T0 не работает в режиме Idle. Таймер T1 продолжает работу при переводе микроконтроллера в режим Idle, что позволяет использовать его как часы реального времени.

10 Порты

В состав микроконтроллера входят восемь 8-разрядных портов ввода-вывода – PA0, PA1, PA2, PA3, PB0, PB1, PB2, PB3.

Порты PA0, PA1, PA2 и PA3 объединены в 32-разрядный порт A, а порты PB0, PB1, PB2 и PB3 – в 32-разрядный порт B. Дальнейшее описание относится к 32-разрядным портам.

После сброса все выводы порта конфигурируются как выводы общего назначения и находятся в третьем состоянии. Направление работы выводов определяется состоянием битов регистра OUTEN (при установленном соответствующем бите вывод работает как выход, при сброшенном бите вывод находится в третьем состоянии и работает как вход). Помимо этого выводы имеют альтернативные функции (от одной до четырех).

10.1 Вывод порта

На рисунке 10.1 представлена структурная схема вывода порта микроконтроллера. Регистры, показанные на данном рисунке, имеют 32-разрядный формат, где каждый бит управляет соответствующим выводом порта. Регистр ALTCTRL имеет формат, показанный на рисунке 10.2.

Схемы выводов порта идентичны.

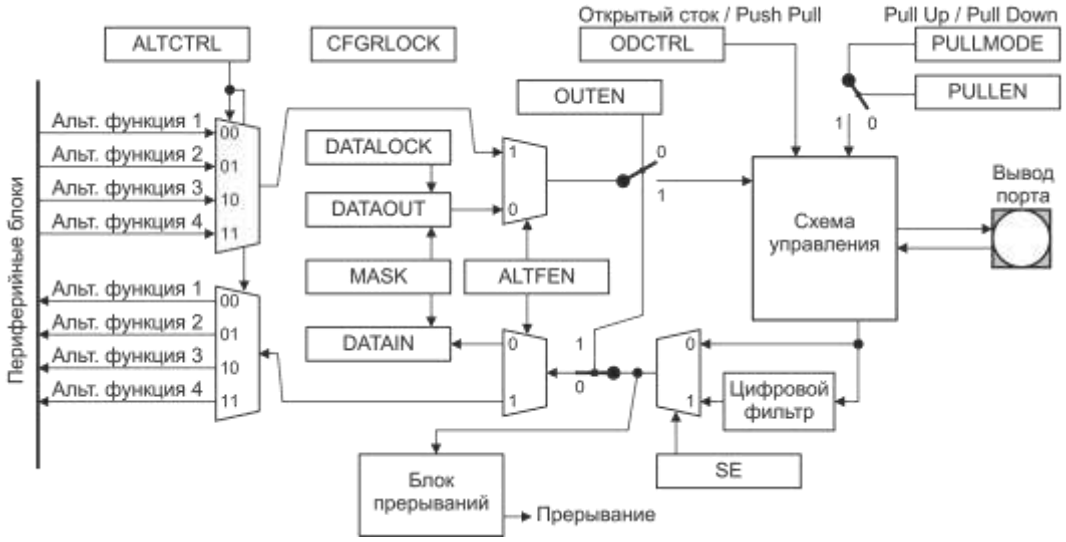


Рисунок 10.1 – Структурная схема вывода порта микроконтроллера и управляющие регистры

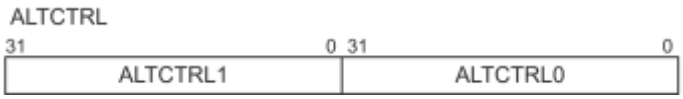


Рисунок 10.2 – Формат регистра ALTCTRL

Структурная схема цифрового фильтра, показанного на рисунке 10.1, представлена на рисунке 10.3.

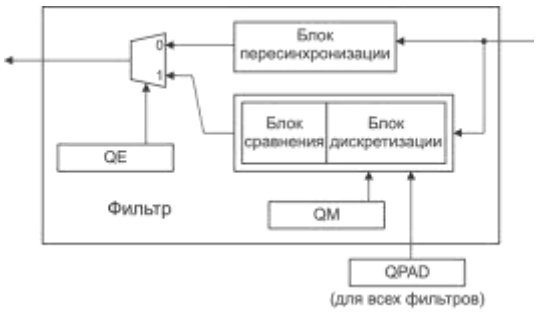


Рисунок 10.3 – Структурная схема цифрового фильтра вывода

Режим входа

По умолчанию бит (имеется в виду бит, соответствующий выводу порта) регистра OUTEN сброшен, вывод работает как вход, а полученные данные сохраняются в регистре DATAIN, откуда могут быть прочитаны.

Входной сигнал после схемы управления может передаваться для дальнейшей обработки как напрямую (асинхронный вход), так и через цифровой фильтр, что задается битом регистра SE. Фильтр позволяет синхронизировать входной сигнал с тактовой частотой работы микроконтроллера. Дополнительно есть возможность использования схемы накопления трех или шести отсчетов входного сигнала для помехоустойчивости вывода (блок дискретизации с блоком сравнения). Количество отсчетов задается регистром QM. Если результаты всех отсчетов совпадают, сигнал передается дальше по схеме, в противном случае состояние выхода блока сравнения не меняется. Временной интервал между отсчетами задается в количестве тактов системной частоты регистром QPAD, который в отличие от остальных является регистром порта, а не отдельного вывода, и поэтому временной интервал задается один для всех выводов порта. Для подключения схемы накопления отсчетов используется регистр QE.

Режим выхода

Для переключения вывода в режим выхода следует установить бит в регистре OUTEN.

Примечание – Установка единицы в регистре OUTEN вызовет аппаратный сброс соответствующего бита в регистре DATAIN и отключение его и регистров периферийных блоков от схемы управления. В то же время цифровой фильтр и мультиплексор, управляемый регистром SE, останутся подключенными к схеме управления, что позволяет использовать их для генерирования прерываний после настройки блока прерываний, показанного на рисунке 10.7.

Данные для передачи записываются в регистр DATAOUT, изменение состояния которого незамедлительно влияет на выводы порта. Регистр DATALOCK позволяет закрыть доступ для записи к любым битам регистра DATAOUT. Состояние бит, закрытых единицами, не может быть изменено (см. рисунок 10.4).

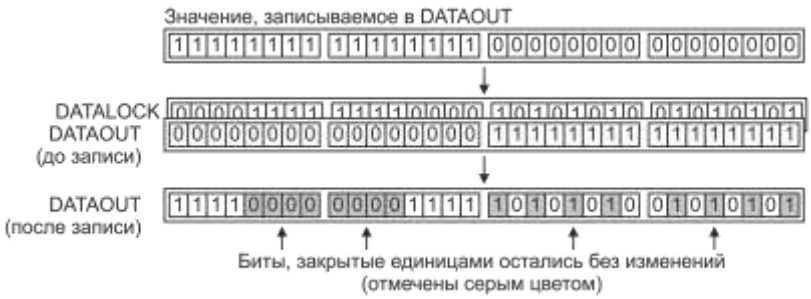


Рисунок 10.4 – Побитное ограничение возможности записи в регистр DATAOUT

Для возобновления доступа к битам регистра DATAOUT следует сбросить соответствующие биты регистра DATALOCK.

Маскирование

Для управления состоянием выводов порта дополнительно используется механизм маскирования. Он позволяет устанавливать желаемый уровень сигнала на нужном выводе, не затрагивая состояние других выводов. 32-разрядный порт условно разбит на четыре части: первый, второй, третий и четвертый байты. Для каждого байта имеется массив из 8-разрядных регистров – набор масок: 00h, 01h, 02h и т. д. до FFh (т. е. 256 масок). Таким образом, каждый порт имеет четыре набора масок. Массивы адресов указаны в таблице 10.1.

Таблица 10.1 – Адреса областей масок

Биты порта/ регистра	Байт порта/ регистра	Диапазоны адресов областей масок	
		Порт A	Порт B
7 – 0	Первый	0000_4400h – 0000_44FFh	0000_4900h – 0000_49FFh
15 – 7	Второй	0000_4500h – 0000_45FFh	0000_4A00h – 0000_4AFFh
24 – 16	Третий	0000_4600h – 0000_46FFh	0000_4B00h – 0000_4BFFh
31 – 25	Четвертый	0000_4700h – 0000_47FFh	0000_4C00h – 0000_4CFFh

Механизм маскирования записи байтов порта (на примере порта A) показан на рисунке 10.5.

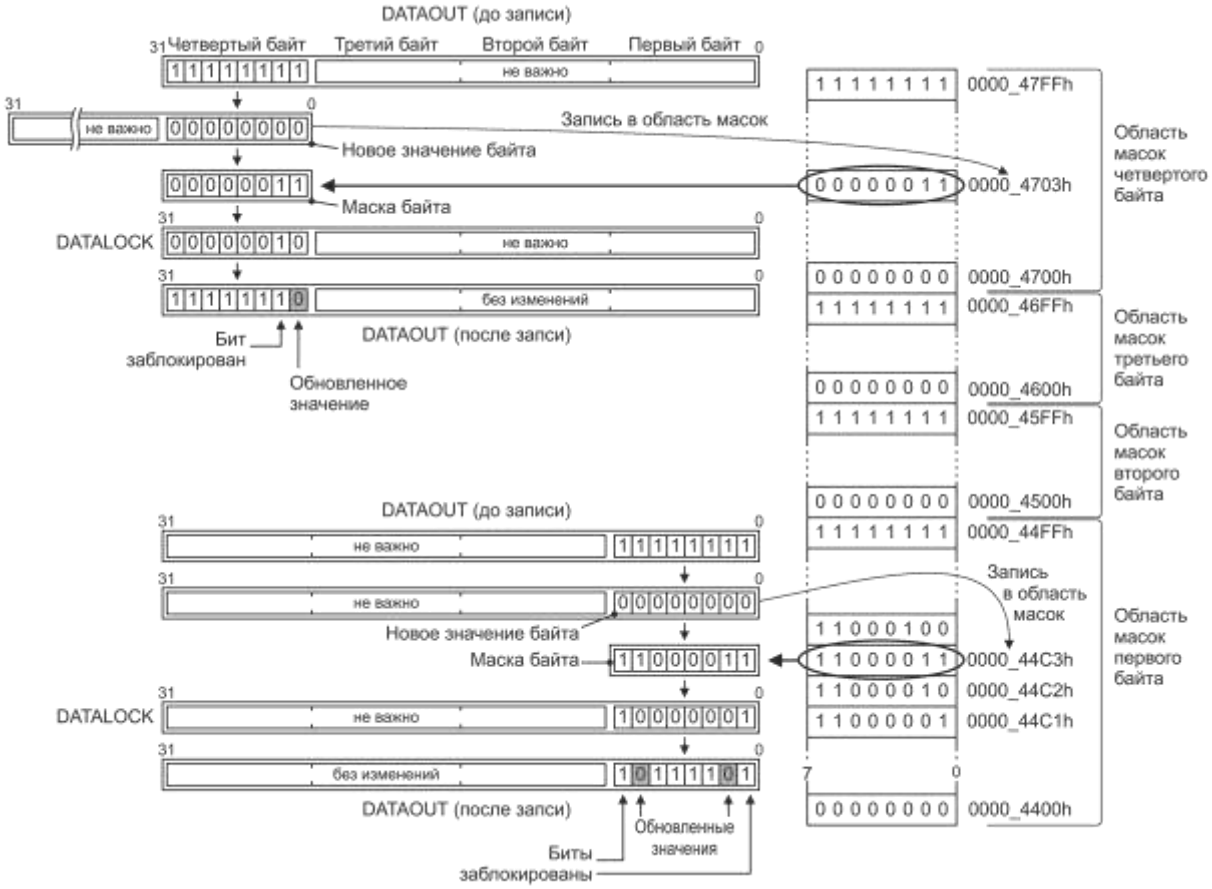


Рисунок 10.5 – Механизм изменения состояния байтов порта A с маскированием (на примере младшего и старшего байтов)

Для того чтобы изменить состояние одного из четырех байт регистра DATAOUT и соответствующих выводов порта, следует выполнить следующий алгоритм:

1 Выбрать маску (биты, закрытые нулями маски, останутся без изменений).

2 Вычислить адрес маски. Для этого к начальному адресу области масок, соответствующей байту, прибавить смещение, равное значению маски.

3 По адресу маски выполнить запись нового значения, которое должно быть занесено в регистр DATAOUT. Допускается любая запись (побайтная, пословная), поскольку значимым является только младший байт.

После этого записанное значение будет маскировано дважды: маской, по адресу которой была выполнена запись, и маской, имеющей наивысший приоритет соответствующего байта регистра DATALOCK.

Пояснение к примеру – в верхней части рисунка 10.5. Биты с 25 по 31 регистра DATAOUT, соответственно, и выводы с 25 по 31 порта A, находятся в состоянии логической единицы. Для того чтобы изменить состояние только выводов 25 и 26, следует выбрать маску 03h из области масок четвертого байта с начальным адресом 0000_4700h. Адрес маски 0000_4703h. Запись по этому адресу значения байта, у которого два младших бита равны нулю, должна привести к переводу выводов 25 и 26 в состояние логического нуля. Но поскольку изменение состояния 26 бита заблокировано регистром DATALOCK, то будет сброшен только 25 бит. Таким образом, 25 вывод порта будет переведен в состояние логического нуля, а состояние остальных выводов останется без изменения.

Механизм маскирования может быть также применен к полученным данным, расположенным в регистре DATAIN, и позволяет упростить программную реализацию проверки состояния отдельных бит в полученном байте или слове данных.

Аналогично регистру DATAOUT, регистр DATAIN также условно разбит на четыре части – первый, второй, третий и четвертый байты – каждой из которых соответствует набор (область) масок. Массивы адресов указаны в таблице 10.1.

Механизм чтения принятых данных с маскированием (на примере порта A) показан на рисунке 10.6.

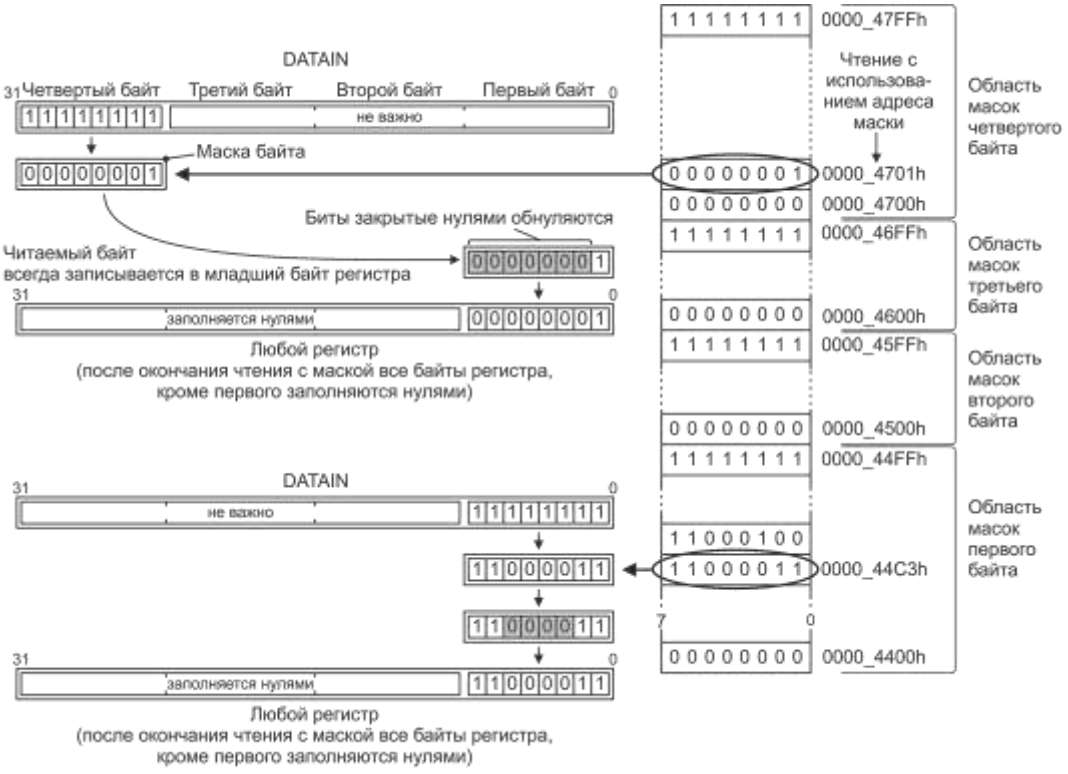


Рисунок 10.6 – Механизм чтения байтов принятых данных порта A с маскированием (на примере младшего и старшего байтов)

Для того чтобы прочитать состояние одного из четырех байт регистра DATAIN, следует выполнить следующий алгоритм:

1 Выбрать маску (биты, закрытые единицами маски, будут прочитаны без изменений, остальные – обнулены).

2 Вычислить адрес маски. Для этого к начальному адресу области масок, соответствующей байту, прибавить смещение, равное значению маски.

3 Выполнить чтение из адреса маски в любой регистр-приемник.

Далее результат побитного логического умножения прочитанного байта и маски будет помещен в младший байт (!) регистра-приемника. Остальные байты регистра-приемника будут заполнены нулями.

PullUp, PullDown, PushPull и открытый сток

Каждый вывод порта имеет схемы подтяжки PullUp и PullDown, которые могут подключаться и отключаться программно, а также может функционировать в режимах с открытым стоком и PushPull.

Схемы PullUp и PullDown могут быть подключены независимо от направления работы вывода установкой бита в регистре PULLEN. Выбор схемы определяется состоянием бита регистра PULLMODE.

По умолчанию, если вывод работает как выход, он работает в режиме PushPull. Для переключения в режим открытого стока следует установить бит в регистре ODCTRL.

Альтернативные функции

Вывод порта может иметь до четырех дополнительных альтернативных функций. Переключение вывода в режим альтернативной функции осуществляется установкой бита в регистре ALTFEN. Номер альтернативной функции задается битовым полем регистра ALTCTRL. Следует помнить, что при включении альтернативной функции, вывод порта должен быть дополнительно сконфигурирован соответствующим образом как вход или выход регистром OUTEN.

10.2 Прерывания

Внешние прерывания

Вывод порта может использоваться как вход внешнего прерывания. Входной сигнал с вывода порта, пройдя схему управления, передается в схему вывода, а также в блок прерываний, схема которого показана на рисунке 10.7.

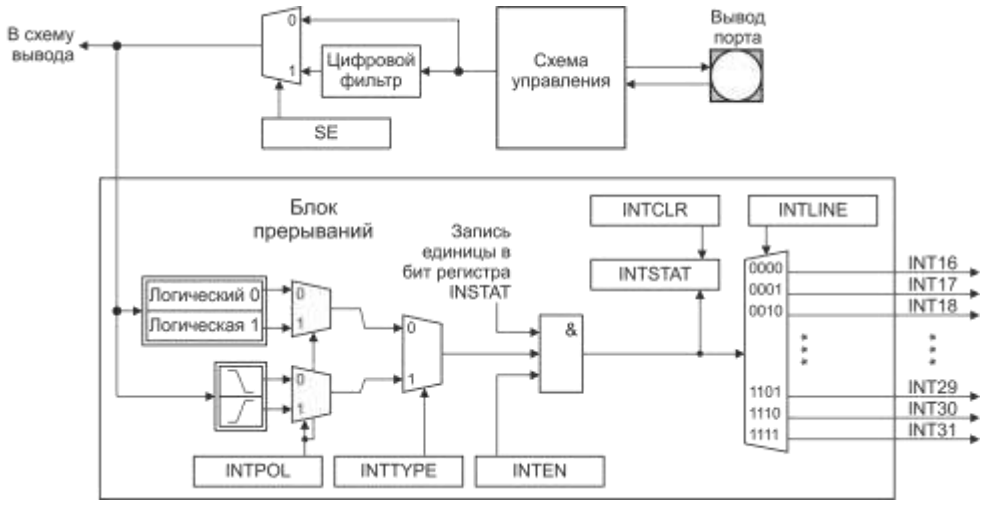


Рисунок 10.7 – Блок прерываний

Блок прерываний позволяет отслеживать следующие события на выводе:

- появление переднего фронта сигнала (переход сигнала из состояния логического нуля в состояние логической единицы);
- появление заднего фронта сигнала (переход сигнала из состояния логической единицы в состояние логического нуля);
- уровень сигнала.

Для того чтобы использовать вывод порта как вход внешнего прерывания следует разрешить формирование запроса прерывания от вывода установкой соответствующего бита в регистре INTEN.

Двум портам (A и B) микроконтроллера выделено 16 линий (векторов) прерываний INT16 – INT31. Запрос на прерывание от вывода может быть направлен на любую из них. На одну линию может быть направлено любое количество запросов. Номер линии задается в соответствующем поле регистра INTLINE, который имеет формат, показанный на рисунке 10.8.

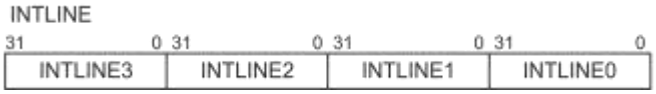


Рисунок 10.8 – Формат регистра INTLINE

Соответствующий выводу бит регистра INTPOL задает, какой уровень и какой фронт сигнала будет являться ожидаемым событием. Посредством регистра INTTYPE осуществляется выбор между событиями – уровень/фронт.

При обнаружении ожидаемого события в регистре INTSTAT будет установлен соответствующий флаг, который не сбрасывается аппаратно. В связи с этим в программе обслуживания прерывания для сброса флага следует использовать команду записи единицы в соответствующий флагу бит регистра INTCLR.

Примечание – Если выбрано событие «уровень сигнала», то до тех пор, пока на выводе будет оставаться заданный уровень, в соответствующий выводу бит регистра INSTAT будет записываться единица. Поэтому, как только флаг события будет сброшен, он будет тут же установлен снова, что приведет к постоянному генерированию запроса на прерывание от блока. Если уровень сигнала на выводе не меняется, то для выхода из подобной ситуации следует изменить ожидаемое событие, либо запретить его влияние посредством сброса бита разрешения в регистре INTEN.

Программные прерывания

Прерывание может быть сгенерировано программно, записью единицы в соответствующий бит (флаг) регистра INTSTAT. Сброс флага производится записью единицы в соответствующий флагу бит регистра INTCLR.

Примечание – Программная установка флага в регистре INTSTAT приведет к генерированию запроса на прерывание, только если установлен соответствующий бит разрешения в регистре INTEN. В противном случае, флаг установится, но прерывание не будет сгенерировано

Блок прерываний всегда подключен к выводу порта и реагирует на события в независимости от того, что является их источником – выходной сигнал или внешний входной. Это предоставляет пользователю еще один способ программного формирования запросов прерываний.

11 Модуль квадратурного декодера

Квадратурный декодер преобразует цифровой сигнал с датчика положения вала, позволяя вычислять скорость, направление вращения, а также текущее положение вала.

В состав квадратурного декодера входят (см. рисунок 11.1):

- настраиваемый обработчик сигналов входов;
- квадратурный преобразователь;
- счетчик позиции/блок управления;
- модуль захвата времени;
- таймер временных отсчетов;
- сторожевой таймер.

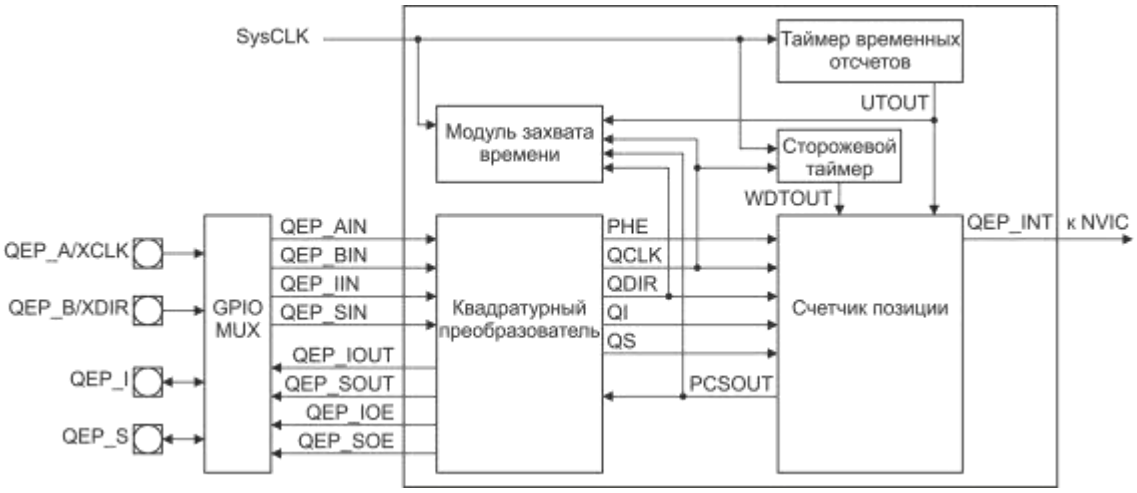


Рисунок 11.1 – Схема квадратурного декодера с мультиплексором входных/выходных сигналов

11.1 Обработчик сигналов входов

Квадратурный декодер использует два квадратурных вывода контроллера, работающих на вход. Также имеются специальный индексный вывод и вывод стробирования, которые могут работать на вход и выход.

QEP_A/XCLK и QEP_B/XDIR – в квадратурном режиме это два входа. Сигналы на входах сдвинуты по фазе на 90 градусов и по ним можно определить скорость и направление вращения ротора (см. рисунок 11.2). В режиме счета/направления сигналы на входах используются как тактовый и сигнал направления вращения ротора, по которым также можно вычислить скорость вращения.

QEP_I – индексный вход. Сигнал на входе сигнализирует о полном обороте ротора. Позволяет сбрасывать счетчик позиции поворота ротора.

QEP_S – пользовательский вход стробирования. Сигнал на входе может сбросить или защелкнуть счетчик позиции. Применяется при использовании концевых выключателей.

Сигналы на входах могут быть проинвертированы. Инверсия включается установкой соответствующего бита в регистре QDECCTL.

Установкой бита SWAP можно включить обратный счет, т. е. программно подать сигнал с вывода QEP_A на вход QB квадратурного преобразователя, а сигнал с вывода QEP_B подать на вход QA (входы A и B меняются местами).

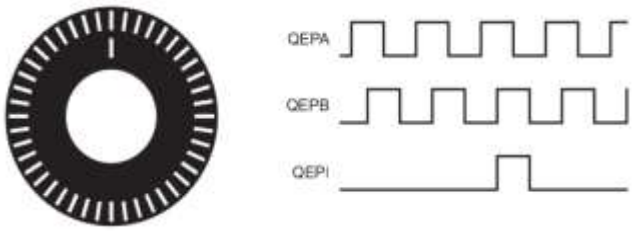


Рисунок 11.2 – Диаграмма входных сигналов

11.2 Квадратурный преобразователь

На рисунке 11.3 показана схема квадратурного преобразователя.

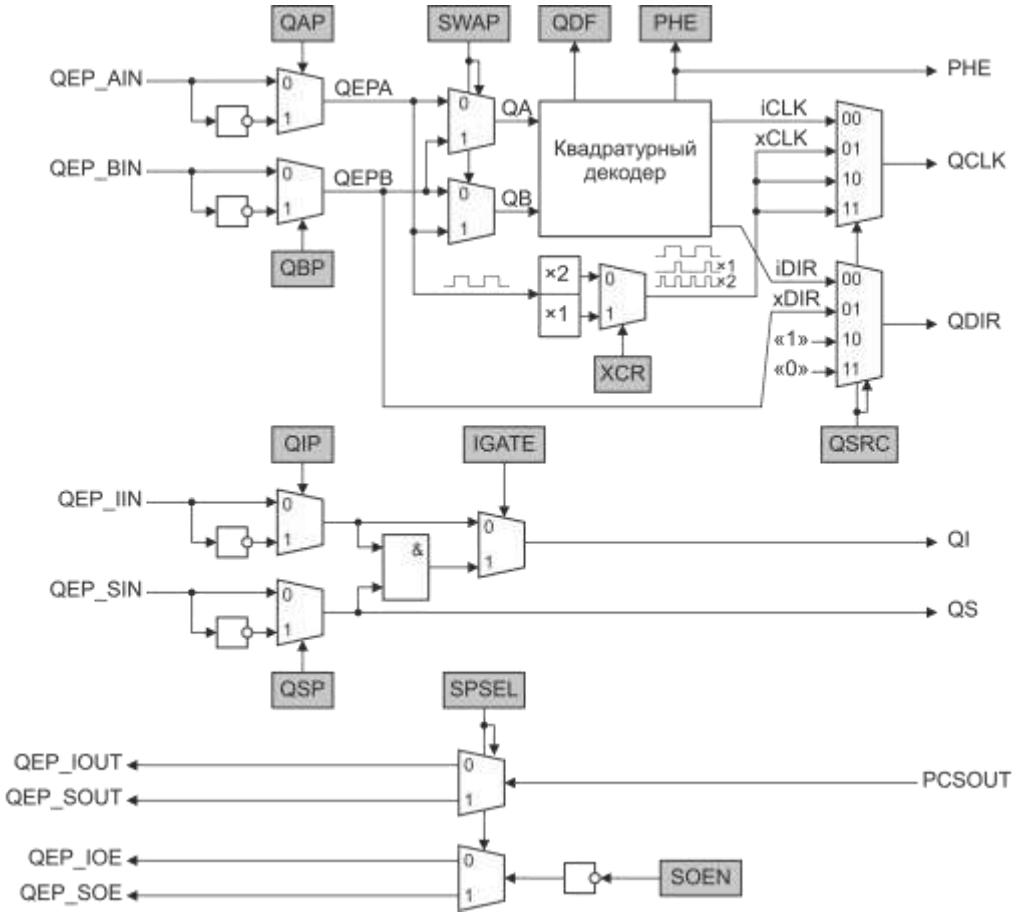


Рисунок 11.3 – Схема квадратурного преобразователя

Режимы работы
Квадратурный преобразователь поддерживает четыре режима работы:
- режим квадратурного счета;
- режим счета/направления;
- режим счета вверх;
- режим счета вниз.

Выбор режима зависит от значения поля QSRC регистра QDECCTL.

Режим квадратурного счета

Квадратурный преобразователь формирует сигнал направления вращения, тактовый сигнал и сигнал направления счета (вверх-вниз) для счетчика позиции.

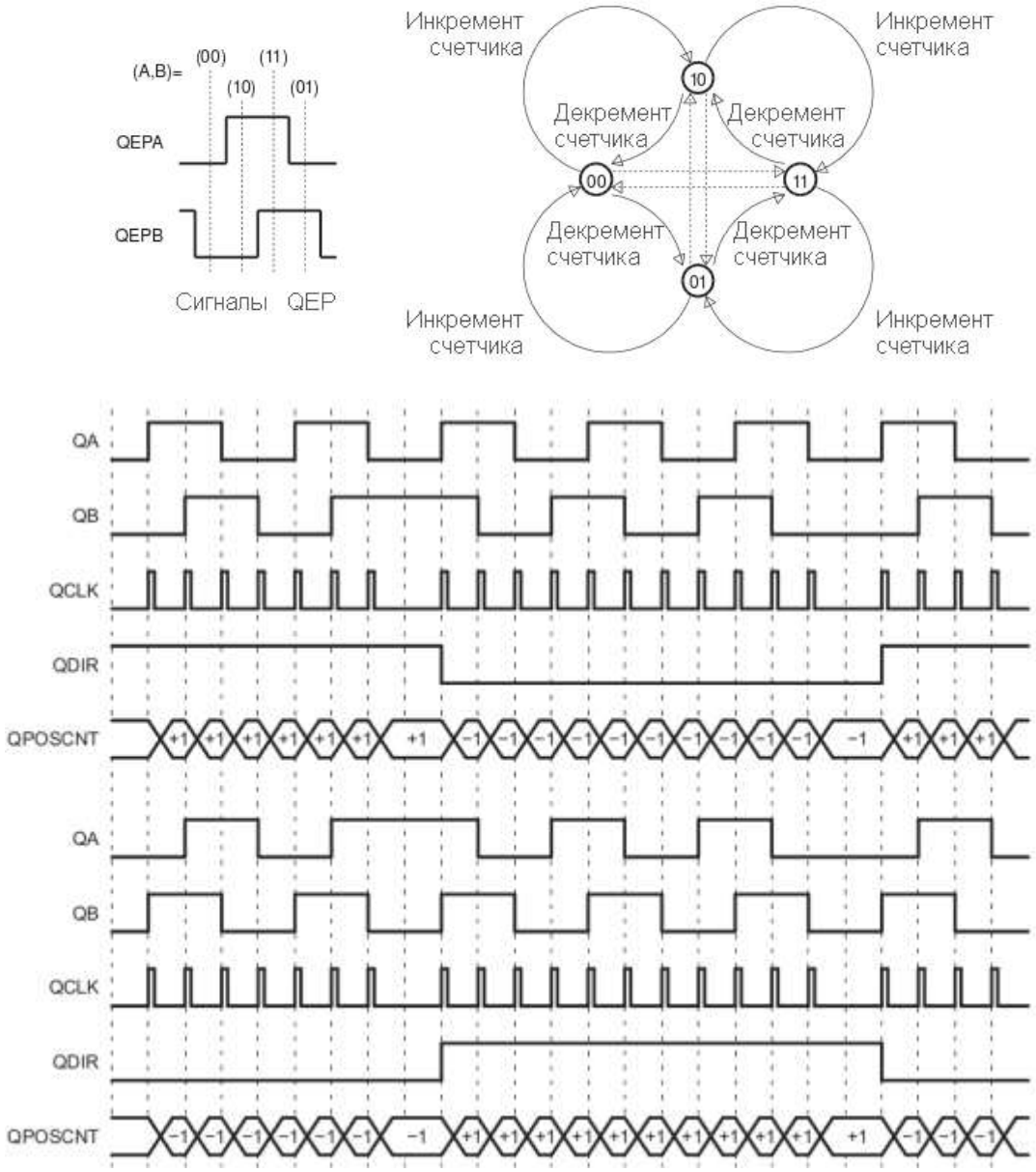


Рисунок 11.4 – Временная диаграмма и автомат состояний работы в квадратурном режиме счета

Направление вращения ротора определяется по порядку смены передних и задних фронтов на входах QEP_A и QEP_B. К примеру, если за передним фронтом сигнала на входе QEP_A следует передний фронт сигнала на входе QEP_B (см. рисунок 11.4), то направление вращения следует считать прямым, а счетчик позиции работает на увеличение. Если же за передним фронтом сигнала на входе QEP_B следует передний фронт сигнала на входе QEP_A, то направление вращения следует считать инверсным, а счетчик позиции работает на уменьшение. Если на обоих выводах зафиксировано одновременно два фронта, то такое состояние считается ошибочным.

Квадратурный преобразователь выдает четыре счетных импульса на один период входного сигнала, поскольку использует для счета передний и задний фронт сигналов.

Режим счета/направления

В режиме счета/направления вывод QEP_A работает как вход тактовых импульсов, а вывод QEP_B – как вход задания направления счета. Счетчик позиции в этом режиме работает по каждому переднему фронту сигнала на входе QEP_A.

Режим вверх

Режим вверх используется для вычисления частоты следования импульсов на вывод QEP_A. Фронт задается битом XCR регистра QDECCTL. Счетчик всегда работает на увеличение.

Режим вниз

Режим вниз используется для вычисления частоты следования импульсов на выводе QEP_A. Фронт задается битом XCR. Счетчик всегда работает на уменьшение.

11.3 Счетчик позиции

Работа счетчика позиции контролируется посредством регистров QEPCTL и QPOSCTL, которыми задается режим счета, сброса и хранения, а также логика для формирования внешнего сигнала синхронизации.

Режимы сброса счетчика позиции

Счетчик позиции может накапливать результат в течение многих оборотов вала, а может подсчитывать позицию только за один оборот, сбрасываясь каждый раз по событию прихода индексной метки. В зависимости от назначения могут использоваться следующие способы сброса счетчика позиции:

- по сигналу индексации;
- по переполнению;
- только по первому сигналу индексации;
- по таймеру временных отсчетов.

Режим задается полем PCRM регистра QEPCTL.

Счетчик сбрасывается в ноль при его переполнении или при превышении значения регистра максимального значения QPOSMAX. Флаг прерывания, возникающего при переполнении счетчика, устанавливается в регистре QFLG.

Режим сброса по сигналу индексации

Режим сброса по сигналу индексации включен по умолчанию.

При получении сигнала с вывода индексации QEP_I при счете вверх, счетчик обнулится по следующему фронту сигнала тактирования QCLK. Если же сигнал индексации был получен при счете вниз, то в счетчик будет загружено значение QPOSMAX (см. рисунок 11.5).

При получении первого сигнала индексации, схема дожидается любого изменения на квадратурных входах и запоминает значение этого события – фронт, активный вывод (QEP_A или QEP_B), а также направление вращения. Этот момент времени называется маркером индексации. При появлении этого события устанавливается бит FIMF регистра QEPSTS, а направление вращения сохраняется в бите FIDF. В дальнейшем, маркер индексации можно использовать для сохранения значения счетчика.

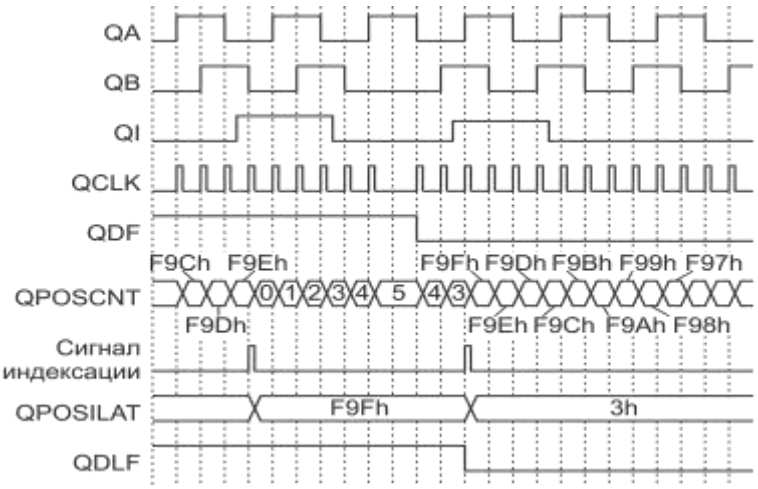


Рисунок 11.5 – Временная диаграмма сброса по сигналу индексации

По каждому сигналу индексации, включая маркер индексации, содержимое счетчика сохраняется в регистре QPOSILAT, а направление вращения – в бите QDLF регистра QEPSTS. Если при сохранении значение счетчика QPOSCNT не равно ни нулю, ни значению QPOSMAX, то выставляется флаг ошибки счетчика позиции (бит PCEF в регистре QEPSTS) и флаг прерывания (бит PCE в регистре QFLG). Флаг ошибки счетчика позиции обновляется с каждым индексом, а флаг прерывания может быть сброшен только программно.

Поле настройки события индексации для сохранения счетчика позиции IEL (регистр QEPCTL) игнорируется. Также только в этом режиме могут устанавливаться флаг ошибки счетчика позиции PCEF и флаг соответствующего прерывания.

Режим сброса по переполнению

Максимальное значение счетчика позиции задается регистром QPOSMAX. Если счетчик считает вверх и достигнуто максимальное значение, то со следующим тактом синхросигнала счетчик обнулится. Если счетчик считает вниз и достигнуто значение нуля, то со следующим тактом синхросигнала в счетчик будет загружено значение QPOSMAX. Сброс по событию индексации не производится.

Получение значений маркера индексации происходит аналогично тому, как это происходит в режиме сброса по сигналу индексации. Полученные значения могут использоваться при инициализации по маркеру индексации, если в поле IEL записано значение 11b в регистре QEPCTL. Временная диаграмма сброса показана на рисунке 11.6.

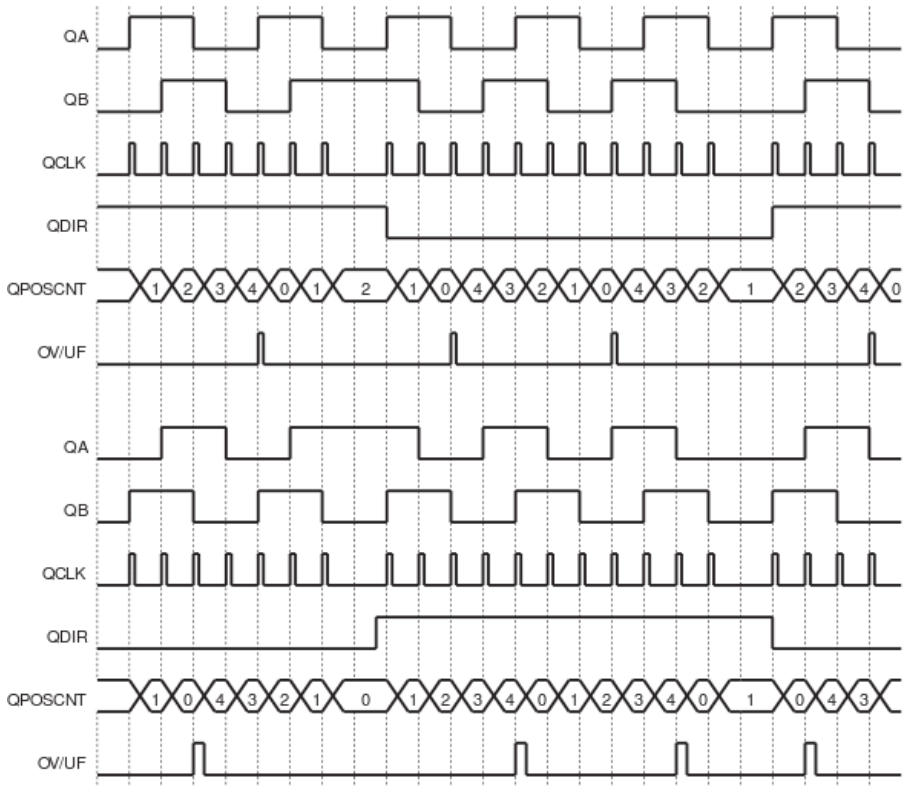


Рисунок 11.6 – Временная диаграмма сброса по сигналу переполнения

Режим сброса по первому сигналу индексации

Если было получено событие индексации при счете вверх, то счетчик обнулится со следующим тактом синхросигнала. Если же событие индексации было зафиксировано при счете вниз, то со следующим тактом синхросигнала в счетчик будет загружено значение QPOSMAX. При последующем счете сброс может произойти только при достижении нуля или значения QPOSMAX (т. е. аналогично режиму сброса по переполнению), а дальнейшие возможные события получения сигнала на выводе индексации влиять на сброс не будут.

Получение значений маркера индексации происходит аналогично тому, как это происходит в режиме сброса по сигналу индексации. Полученные значения могут использоваться при инициализации по маркеру индексации, если в поле IEL записано значение 11b.

Режим сброса по таймеру временных отсчетов

В этом режиме счетчик сбрасывается в ноль или загружается значением QPOSMAX, в зависимости от текущего режима счета (задается полем QSRC регистра QDECCTL), по событию срабатывания таймера временных отсчетов. В остальном режим аналогичен режиму сброса по переполнению.

Также можно настроить сохранение значения счетчика QPOSCNT в регистр QPOSLAT перед сбросом, для этого необходимо включить модуль захвата, установив бит CEN в регистре QCAPCTL. Этот режим удобен для измерения частоты.

Сохранение счетчика позиции

Внешние входы индексации и стробирования можно запрограммировать на формирование событий для сохранения значения счетчика позиции в регистры QPOSILAT и QPOSSLAT.

Сохранение по событию индексации

В некоторых задачах не требуется сбрасывать счетчик позиции по каждому сигналу индексации, и вместо этого может потребоваться увеличить разрядность счетчика до 32 бит (режимы, задаваемые значениями PCRM, равными 01b и 10b). В этом случае бит QDLF (направление вращения) в регистре QEPSTS будет перезаписываться по каждому сигналу индексации, а счетчик будет сохранять значение по следующим событиям индексации:

- по переднему фронту сигнала индексации (IEL = 01b);
- по заднему фронту сигнала индексации (при IEL = 10b);
- по маркеру индексации (при IEL = 11b).

Сохранение значения счетчика по маркеру индексации будет производиться только в присутствии сигнала индекса и по событию, эквивалентному сохраненному при первой индексации по маркеру. Если направление вращения изменится, то сохраненное в маркере значение типа фронта меняется на обратное. Это сделано с целью привязки индекса к квадратурному сигналу QA/QB, а также для более точной обработки индексации, чтобы исключить влияние ширины импульса на выводе индексации.

При сохранении значения счетчика в регистр QPOSILAT формируется флаг IEL прерывания индексации в регистре QFLG. В режиме сброса по сигналу индексации (PCRM = 00h) значение поля IEL в регистре QEPCTL игнорируется.

Сохранение по событию стробирования

Значение счетчика сохраняется в регистр QPOSSLAT по каждому переднему фронту сигнала на входе QEP_S, если сброшен бит SEL в регистре QEPCTL. Если же бит SEL установлен, то сохранение в QPOSSLAT происходит по переднему фронту сигнала строба на входе QEP_S при прямом направлении вращения и по заднему фронту для обратного вращения. При каждом сохранении счетчика в QPOSSLAT устанавливается флаг прерывания SEL.

Инициализация счетчика позиции

Счетчик событий может быть проинициализирован программно или по событиям:

- событие индексации;
- событие стробирования.

Входной сигнал индексации (QEPI) может использоваться для инициализации счетчика по переднему и заднему фронту. Если поле IEI = 10b, то счетчик загружается значением QPOSINIT по переднему фронту сигнала индексации. Аналогично, если IEI = 11b, то счетчик загружается значением QPOSINIT по заднему фронту сигнала индексации.

Если поле SEI = 10b, то счетчик загружается значением QPOSINIT по переднему фронту сигнала стробирования на входе QEP_S. Если SEI = 11b, то счетчик загружается значением QPOSINIT по переднему фронту сигнала стробирования, если идет счет вверх, и по заднему – если вниз.

Программно счетчик инициализируется при записи единицы в бит SWI регистра QEPCTL. Бит не сбрасывается автоматически, но повторная запись единицы также приведет к инициализации счетчика.

Компаратор счетчика позиции

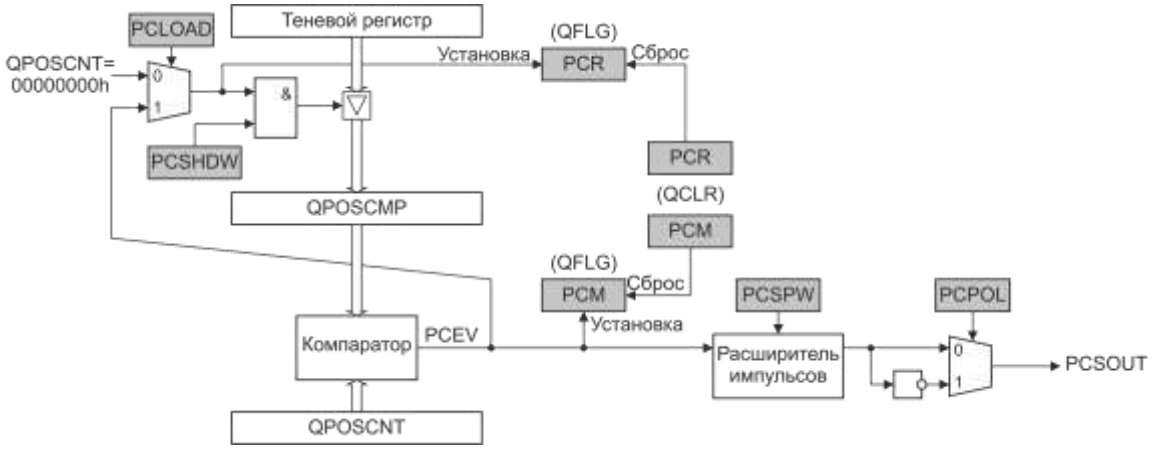


Рисунок 11.7 – Функциональная схема компаратора счетчика позиции

Компаратор (см. рисунок 11.7) сравнивает значение счетчика позиции с регистром QPOSCMP и при совпадении значений формирует прерывание, а также внешний синхросигнал, который может быть направлен на один из выводов: индексный вывод QEP_I или вывод стробирования QEP_S. Бит SPSEL в регистре QDECCTL определяет, на какой именно вывод будет направлен сигнал синхронизации, а бит SOEN в регистре QDECCTL разрешает этому выводу работать как выход.

Регистр QPOSCMP может использовать режим отложенной загрузки, когда отложенное значение берется из теневого регистра, а если режим отложенной загрузки выключен, то запись в QPOSCMP производится сразу в активный регистр.

Отложенная загрузка происходит по следующим событиям:

- по совпадению результатов сравнения;
- по обнулению счетчика QPOSCNT.

Флаг успешного сравнения PCM устанавливается, когда выполняется условие QPOSCNT = QPOSCMP, при этом также формируется синхроимпульс требуемой длительности для извещения внешнего устройства (сигнал PCSOUT). Настраиваемая длительность синхроимпульса контролируется специальной схемой задержки.

Флаг PCR готовности компаратора к отложенной загрузке значения сравнения выставляется, когда выполняется условие для отложенной записи, заданное битом PCLOAD в регистре QPOSCTL. При этом состояние флага включения режима отложенной загрузки PCSHDW (регистр QPOSCTL) не оказывает влияния на установку флага PCR и генерацию соответствующего прерывания.

11.4 Таймер временных отсчетов

Таймер, используемый для оповещения программного обеспечения о необходимости начать измерение скорости, представляет собой 32-разрядный таймер, работающий на частоте системного тактового сигнала. Включается установкой бита UTE в регистре QEPCTL. Когда значение таймера достигает порога (QUTMR = QUPRD), формируется прерывание и выставляется флаг UTO. Данный блок таймера может быть использован для вычисления скорости на высоких скоростях (см. рисунок 11.8), а также для сохранения счетчика позиции, регистра таймера и регистра периода в регистрах QPOSLAT, QCTMRLAT и QCPRDLAT, соответственно. Режим сохранения определяется состоянием бита QCLM в регистре QEPCTL.

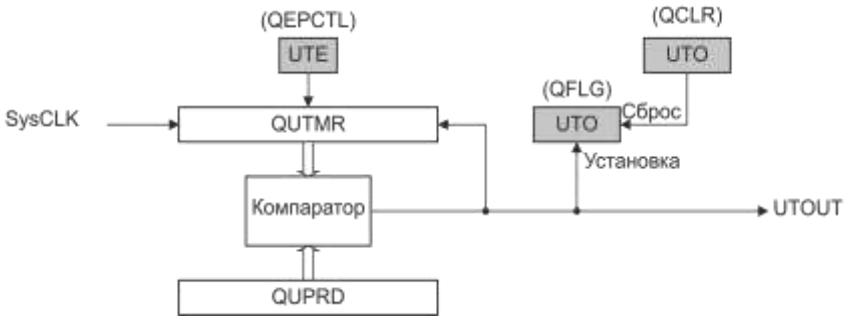


Рисунок 11.8 – Функциональная схема таймера временных отсчетов

11.5 Модуль захвата времени

Функциональная схема модуля захвата времени представлена на рисунке 11.9. Таймер использует тактовый сигнал и сигнал квадратурных событий с коэффициентом деления, программируемым полями CCPS и UUPS в регистре QCAPCTL. Коэффициент деления тактового сигнала можно менять в процессе работы, но в этом случае может произойти событие захвата, содержащее неверные данные. Если такая необходимость все же есть, то нужно выключить модуль захвата (сбросить бит CEN) и снять все маски прерываний. После изменения коэффициентов проинициализировать таймер QCTMR (записать нулевое значение), сбросить все статусы, вновь разрешить прерывания, и включить модуль захвата.

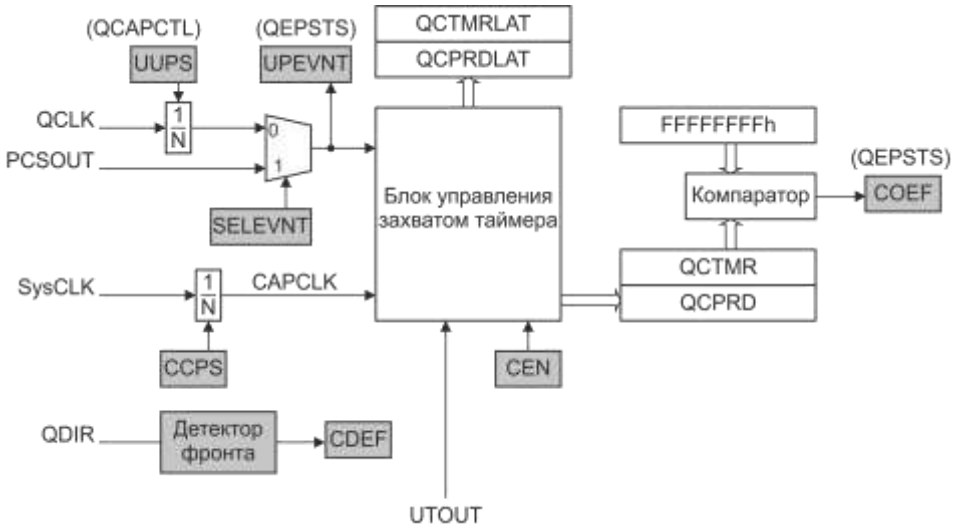


Рисунок 11.9 – Функциональная схема модуля захвата времени

Если бит SELEVNT в регистре QCAPCTL сброшен, то по деленному квадратурному событию значение таймера QCTMR загружается в регистр периода QCPRD, после чего таймер сбрасывается, и устанавливается флаг UPEVNT в регистре QEPSTS, означающий обновление регистра QCPRD. Флаг сбрасывается программно записью единицы.

При установленном бите SELEVNT обновление регистра периода происходит по сигналу от выхода компаратора PCSOUT.

Значение таймера можно использовать при измерениях скорости, если:

- его значение не превысило FFFFFFFFh;
- направление вращения за время измерения не изменилось.

Если между двумя событиями UPEVNT (т. е. во время измерения) таймер QCTMR переполнился, устанавливается флаг ошибки COEF в регистре QEPSTS. Если между двумя

событиями положения вала изменилось направление вращения, устанавливается флаг ошибки CDEF.

Значения таймера (QCTMR) и регистра периода (QCPRD) могут быть сконфигурированы для захвата в регистры QCTMRLAT и QCPRDLAT по событиям:

- прочитан регистр QPOSCNT;
- сработал сторожевой таймер.

Если бит QCLM сброшен, то при каждом чтении регистра счетчика позиции QPOSCNT регистры QCTMR и QCPRD загружаются в QCTMRLAT и QCPRDLAT, соответственно.

Если бит QCLM установлен, то при каждом срабатывании таймера временных отсчетов счетчик позиции, регистр таймера и регистр периода захватываются в регистры QPOSLAT, QCTMRLAT и QCPRDLAT, соответственно.

Измерения на малых скоростях вращения (низкая частота квадратурного сигнала) производятся следующим образом – таймер QCTMR, тактирующийся от системного тактового сигнала с делителем CCPS, по событию UPEVNT, сохраняет свое значение времени в регистре QCPRD, одновременно сбрасывается и выставляет флаг UPEVNT в регистре QEPSTS, чтобы сообщить программе об окончании измерения. Событие UPEVNT возникает каждые несколько тактов QCLK, в соответствии с запрограммированным коэффициентом деления UPPS в регистре QCAPCTL. Таким образом, зная количество квадратурных событий за измеренный отрезок времени, а также такой параметр, как количество квадратурных событий за полный оборот вала, можно вычислить скорость вращения.

Измерения на высоких скоростях (см. рисунок 11.10) могут производиться иначе. Таймер временных отсчетов формирует общую длительность измерения, счетчик позиции подсчитывает количество импульсов QCLK. Зная количество импульсов QCLK за один полный оборот, можно вычислить скорость вращения вала.

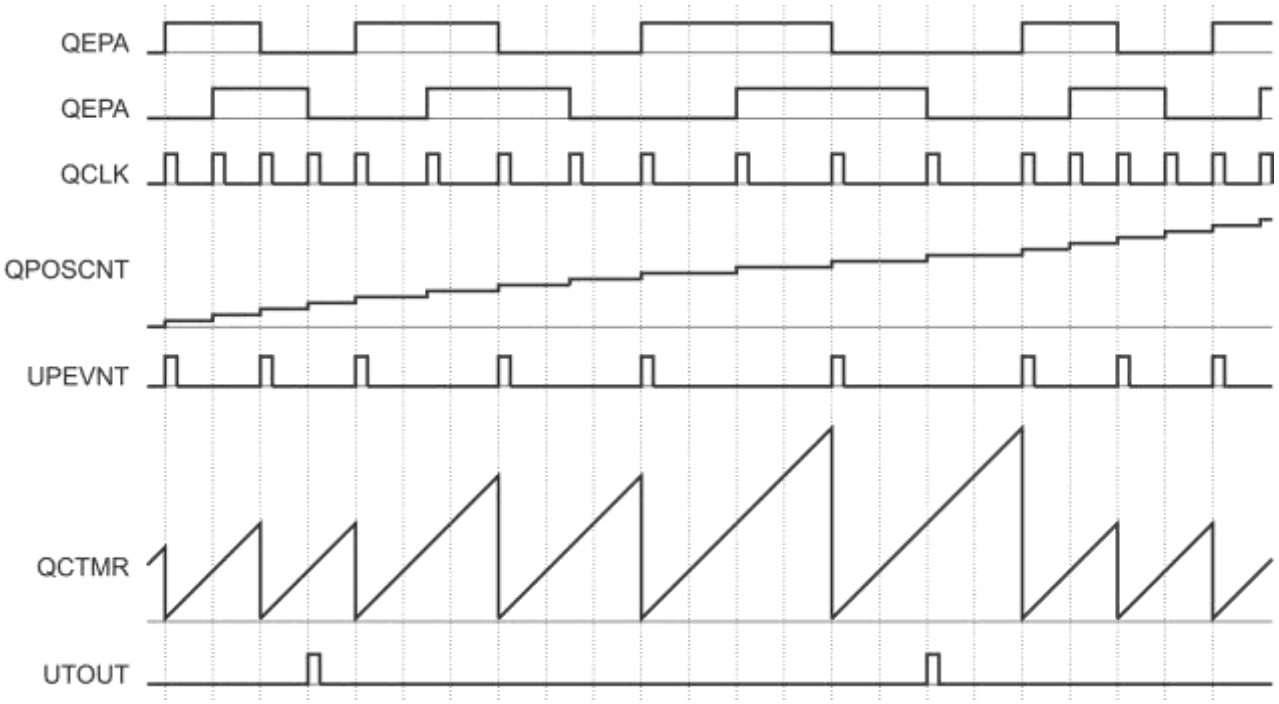


Рисунок 11.10 – Работа на высоких скоростях

Также существует и смешанный способ измерения скорости – по заданному значению счетчика позиции, с помощью компаратора счетчика позиции можно сформировать событие UPEVNT (необходимо установить бит SELEVNT в регистре QCAPCTL), которое, так же как и при измерениях на малых скоростях, позволит получить значение таймера QCTMR. Для использования этого способа измерения скорости необходимо разрешить прерывание по событию PCSOUT компаратора, и устанавливать в этом прерывании каждый раз порог сравнения компаратора QPOSCMP на заданное количество меток вперед по сравнению с текущей позицией счетчика QPOSCNT (в зависимости от направления вращения). Тогда, устанавливая QPOSCMP дальше от QPOSCNT с увеличением скорости вращения, можно поддерживать оптимальное захватываемое время, обеспечивающее максимальную точность измерения времени для всех диапазонов вращения. Этот способ измерения наиболее сложен, но и наиболее универсален.

11.6 Сторожевой таймер

Блок квадратурного декодера содержит 32-битный сторожевой таймер, который тактируется системным тактовым сигналом, деленным на 64, и сбрасывается любым квадратурным событием (перепад на выводе QEP_A/QEP_B). Если ни одного квадратурного события не было зафиксировано до события QWDTMR = QWDPRD, сторожевой таймер формирует флаг прерывания WTO в регистре QFLG. Регистр QWDPRD содержит значение срабатывания сторожевого таймера. Функциональная схема сторожевого таймера представлена на рисунке 11.11.

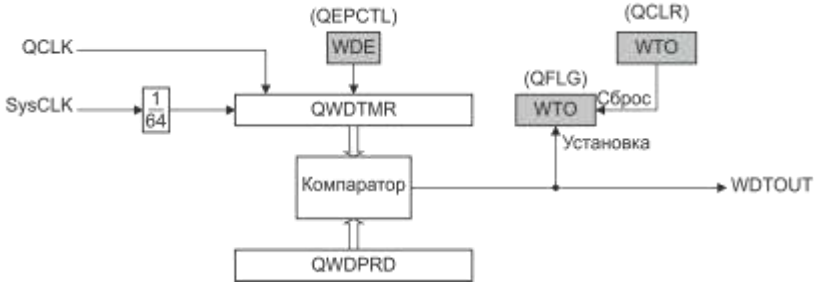


Рисунок 11.11 – Функциональная схема сторожевого таймера

11.7 Система прерываний

Блок квадратурного декодера содержит 11 источников прерываний, см. рисунок 11.12. Система прерываний состоит из регистра маски прерываний QEINT, регистра флагов прерываний QFLG, а также формирования внешнего прерывания INT по наличию активных флагов. Сброс флагов прерываний осуществляется через регистр QCLR. Сброс флага активности прерывания INT осуществляется записью в регистр QINTCLR. Также прерывание можно сформировать программной записью в регистр QFRC, но для этого необходимо предварительно включить счетчик позиции, установив бит QPEN в регистре QEPCTL.

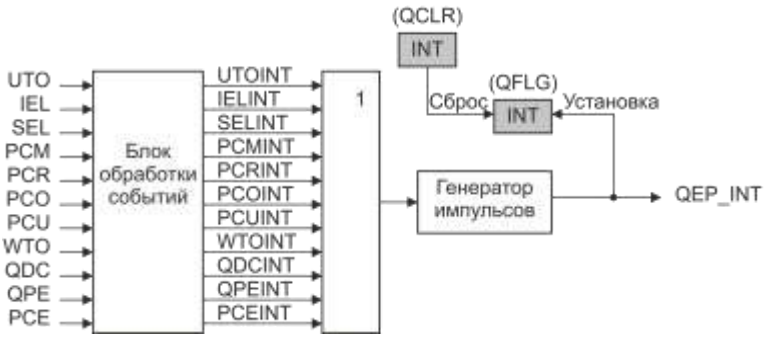


Рисунок 11.12 – Схема системы прерываний

На рисунке 11.13 показана схема формирования прерывания внутри блока обработки событий для события UTO (срабатывание таймера временных отсчетов). Схемы формирования прерываний для остальных событий идентичны.

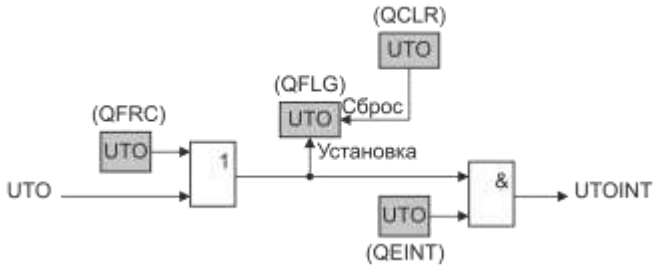


Рисунок 11.13 – Схема формирования прерывания UTOINT

12 Блоки ШИМ

Архитектура блоков ШИМ разработана по принципу минимальной нагрузки на процессор, что достигается автоматизацией формирования выходных импульсов с настраиваемыми пользователем параметрами. Так, после минимальных настроек, эти блоки способны работать самостоятельно, формируя выходные сигналы на выводах PWM_A и PWM_B микроконтроллера.

Микроконтроллер содержит три блока ШИМ, объединенных схемой синхронизации.

Каждый блок ШИМ поддерживает следующую функциональность:

- 16-разрядный таймер;
- два выхода PWM_A и PWM_B, которые могут работать в режиме фронтовой и центрированной модуляции как полностью независимо, так и комплементарно с разделением генератором «мертвого времени»;
- выходы PWM_A и PWM_B могут управляться в зависимости от событий цифровых компараторов блока АЦП, а также от событий блока аналоговых компараторов, обеспечивая автоматический релейный режим поддержания заданной величины;
- программное управление выходами ШИМ;
- программное задание фазы счетчиков таймера для координации работы нескольких блоков ШИМ;
- аппаратный контроль фазы при координации работы нескольких блоков ШИМ;
- предотвращение наложения фронтов за счет генератора «мертвого времени» с независимой схемой задержки переднего и заднего фронтов выходного сигнала;
- сигнал аварии может переводить выходы PWM_A и PWM_B в высокое, низкое или Z-состояние;
- однократная и циклическая обработка сигналов аварии;
- все события могут инициировать прерывания;
- программируемый предделитель событий позволяет снизить нагрузку на процессор при обработке прерываний;
- ШИМ-сигнал может модулироваться высокочастотным сигналом при использовании драйверов ключей с импульсным трансформатором.

Описание сигналов и выводов блока ШИМ:

- PWM_A и PWM_B – выходы ШИМ;
- TZ0 – TZ2 – входы, соединенные с выводами микроконтроллера PWM_TZ0 – PWM_TZ2, с которых принимаются сигналы аварии (общие для всех блоков ШИМ, и каждый блок может использовать или не использовать эти сигналы);
- PWM_SYNCI – вход, служащий для приема внешнего синхросигнала;
- PWM_SYNCO – внутренний сигнал синхронизации на выходе блока ШИМ, являющийся синхросигналом для остальных блоков ШИМ;
- TZINT, INT – внутренние сигналы прерываний, объединенные в один сигнал прерывания.

Функциональная схема блока ШИМ показана на рисунке 12.1.

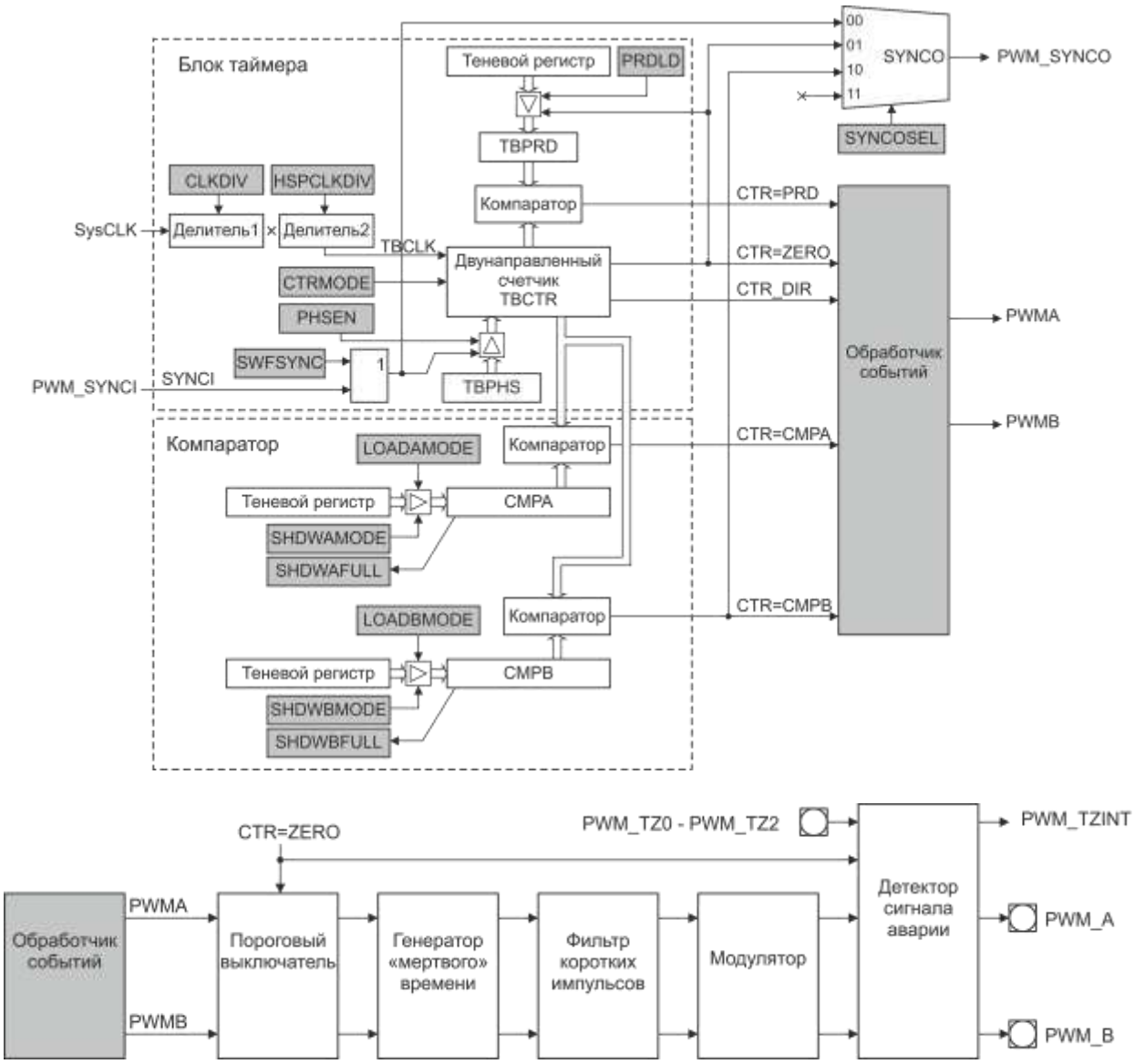


Рисунок 12.1 – Блок ШИМ

12.1 Таймер

Таймер представляет собой двунаправленный счетчик (TBCTR), тактируемый сигналом TBCLK, который формируется на основе синхросигнала SynCLK. Частота сигнала TBCLK задается произведением коэффициентов двух делителей. Коэффициенты задаются полями CLKDIV и HPCLKDIV регистра TBCTL. Для работы других блоков ШИМ счетчик позволяет формировать события такие, как совпадение по периоду CTR = PRD (TBCTR = TBPRD), совпадение с нулем CTR = Zero (TBCTR = 0000h), совпадение с регистрами CTR = CMPA и CTR = CMPB (TBCTR = CMPA и TBCTR = CMPB, соответственно). Событие TBCTR = FFFFh влияет только на флаг CTRMAX регистра TBSTS.

Всеми настройками работы счетчика таймера управляет регистр TBCTL.

Состояние счетчика отражают флаги регистра TBSTS.

На выходе первого блока ШИМ формируется сигнал SYNCO, который является, синхросигналом для остальных блоков ШИМ (см. рисунок 12.3а). Сигнал SYNCO имеет три источника – программно сгенерированный синхроимпульс, посредством записи единицы в бит SWFSYNC, события CTR = Zero и CTR = CMPB. Выбор источника осуществляется посредством поля SYNCOSEL.

В блоке таймера находятся регистры начальной фазы счета TBPHS и периода (максимального значения счетчика) TBPRD. Регистр периода имеет теневой регистр для синхронной загрузки значения, до которого счетчик осуществляет счет. Управление загрузкой осуществляется битом PRDLD.

Счетчик может работать в трех режимах счета (см. рисунок 12.2):

- вверх (от 0000h до значения TBPRD, затем сброс в 0000h и т. д.);
- вниз (от значения TBPRD до 0000h, затем загрузка значения TBPRD и т. д.);
- вверх-вниз (от 0000h до значения TBPRD, затем от значения TBPRD до 0000h и т. д.).

Параметры счета задаются полем CTRMODE.

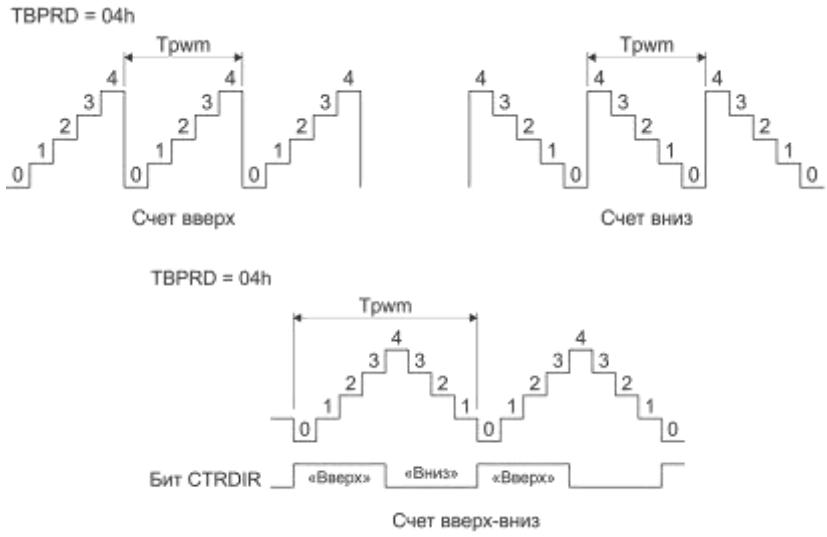


Рисунок 12.2 – Режимы работы счетчика при значении периода 0004h (Tpwm); для режима счета «вверх-вниз» дополнительно указано поведение флага CTRDIR

Синхронизация таймеров блоков ШИМ

Реализована схема синхронизациии блоков ШИМ, см. рисунок 12.3а.

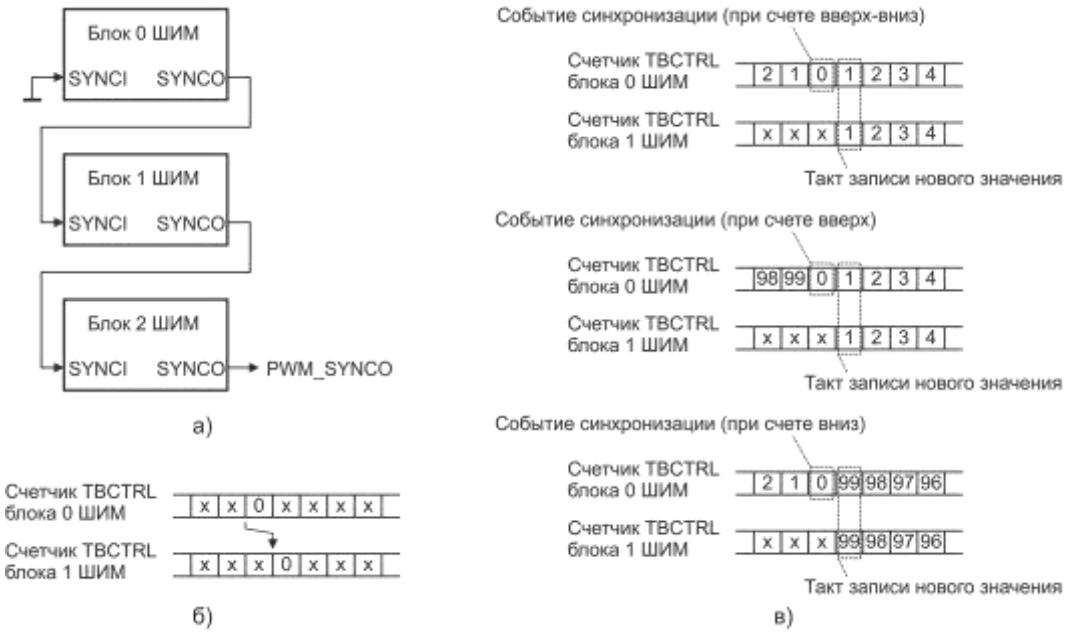


Рисунок 12.3 – Схема синхронизации модулей ШИМ

Система синхронизации таймеров включает в себя таймеры всех блоков ШИМ. Каждый блок ШИМ имеет вход синхронизации PWM_SYNCI и выход синхронизации PWM_SYNCO.

Если бит PHSEN установлен, то в счетчик таймера будет автоматически загружаться значение регистра TBPHS, при выполнении каждого из условий:

- изменение входного сигнала PWM_SYNCI (в этом случае загрузка значения TBPHS в регистр TBCTR происходит на следующий такт TBCLK после поступления импульса на вход PWM_SYNCI с задержкой в два системных такта, если TBCLK = SysCLK, или один такт, если TBCLK ≠ SysCLK);

- запись единицы в бит SWFSYNC (программная синхронизация), которая генерирует импульс синхронизации, аналогичный импульсу с входа PWM_SYNCI.

В режиме счета вверх-вниз необходимо запрограммировать бит PHSDIR, чтобы задать направление счета таймера после синхронизации.

Если бит PHSEN сброшен, блок ШИМ не будет реагировать на входной сигнал синхронизации, а только передавать напрямую этот сигнал на выход PWM_SYNCO, чтобы тактировать другие блоки ШИМ. Следующая особенность схемы: генерация и распространение сигнала синхронизации от блока ШИМ, – занимает один такт TBCLK.

К примеру, если по событию синхронизации блока 0 в счетчик блока 1 должен быть записан ноль, то этот ноль запишется только на следующий такт после события (см. рисунок 12.3б). Таким образом, при синхронизации от другого блока ШИМ нужно всегда учитывать этот такт, и записывать значение фазы, следующее по порядку, в соответствии с режимом счета (см. рисунок 12.3в).

12.2 Компаратор

Компаратор – это блок, сравнивающий значение счетчика таймера с заданными значениями порогов срабатывания. Значения хранятся в регистрах CMPA и CMPB. Значения, записываемые по адресам регистров CMPA и CMPB, предварительно размещаются в теневых регистрах. Это нужно для синхронной загрузки новых значений. Управление загрузкой регистров CMPA и CMPB осуществляется битами SHDWAMODE и SHDWBMODE, а также полями LOADAMODE и LOADBMODE регистра CMPCTL.

Блок компаратора формирует на выходах два события CTR = CMPA и CTR = CMPB, возникающие в случае совпадения значения счетчика с регистром CMPA и/или регистром CMPB, соответственно.

Для каждого компаратора событие может возникать:

- один раз за период, если счетчик считает вверх или вниз;

- один раз за период, если счетчик считает вверх-вниз, но при этом значение в регистре CMPA/CMPB равно 0000h или значению TBPRD;

- два раза за период, если счетчик считает вверх-вниз и при этом значение в регистре CMPA/CMPB лежит в диапазоне 0001h – (TBPRD – 1).

На рисунках 12.4 – 12.7 приведены примеры формирования сигналов событий.

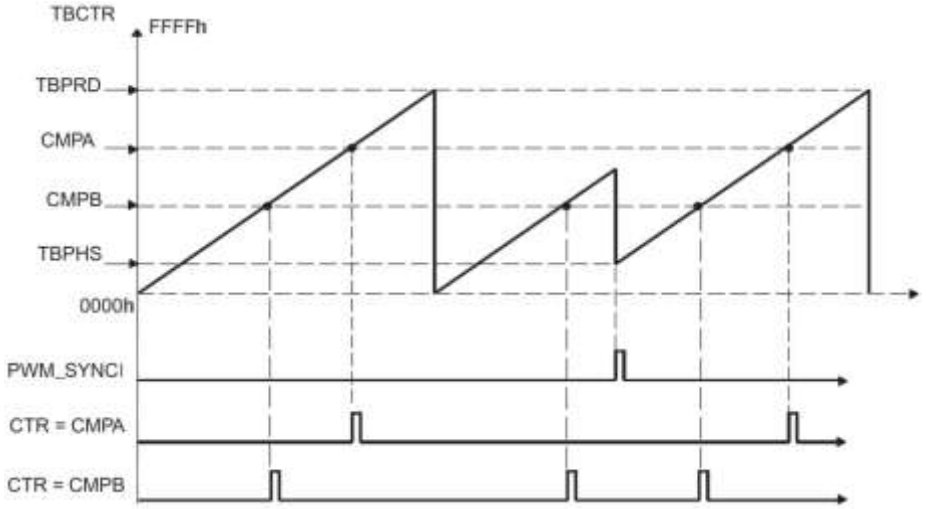


Рисунок 12.4 – Диаграмма работы при счете вверх

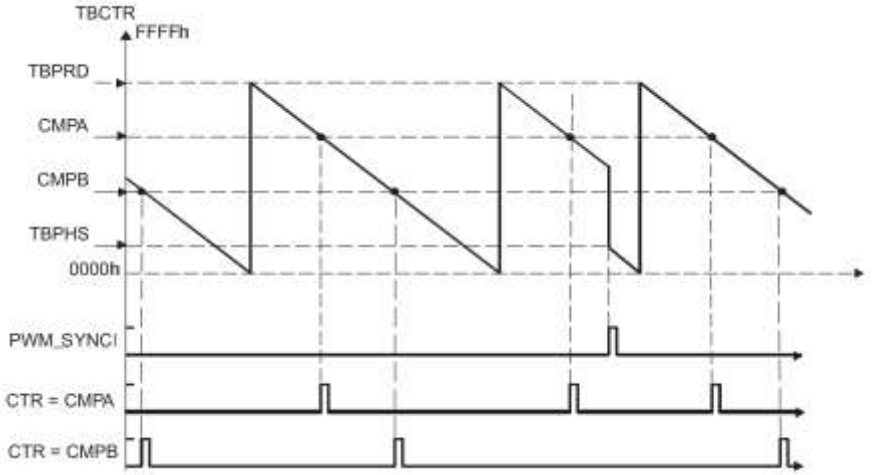


Рисунок 12.5 – Диаграмма работы при счете вниз

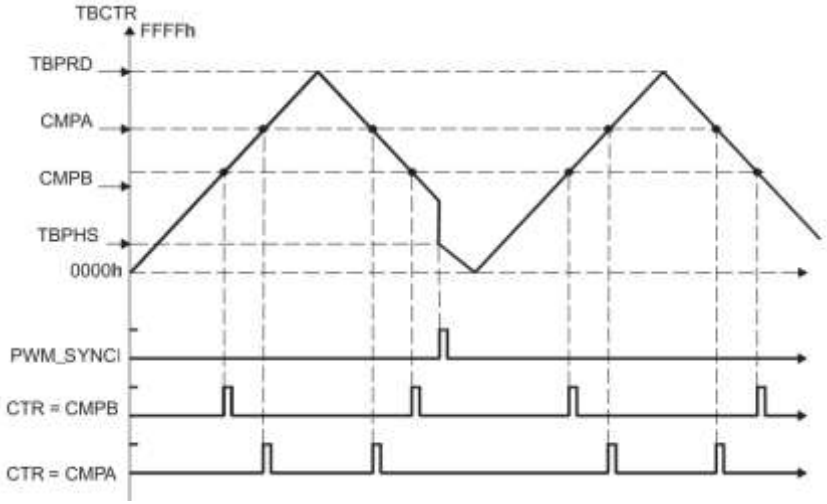


Рисунок 12.6 – Диаграмма работы при счете вверх-вниз. Синхронизация при счете вниз

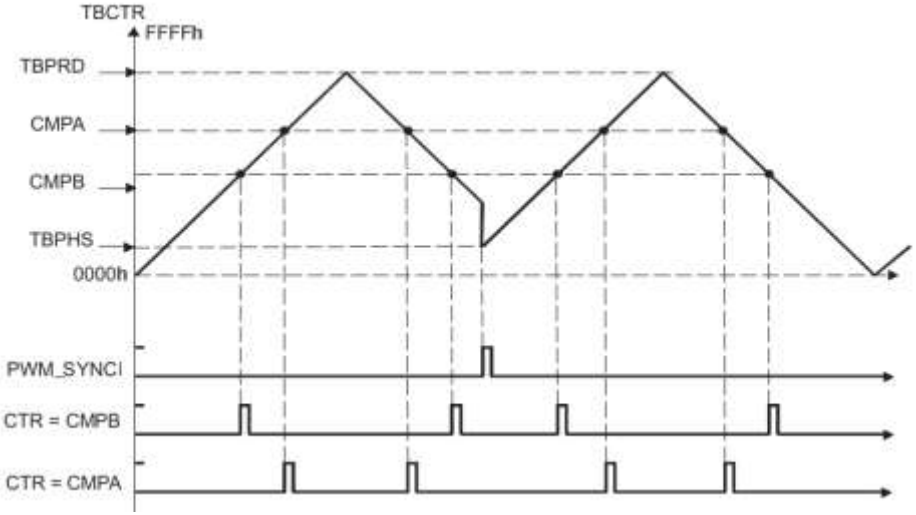


Рисунок 12.7 – Диаграмма работы при счете вверх-вниз. Синхронизация при счете вверх

12.3 Обработчик событий

Обработчик событий – блок, управляющий поведением сигналов на линиях PWMA и PWMB (см. рисунок 12.1) в зависимости от возникающих событий на входе блока и направления счета счетчика таймера. На поведение выходных сигналов влияют импульсы входных сигналов при возникновении событий: CTR = PRD, CTR = Zero, CTR = CMPA, CTR = CMPB.

Основные действия с сигналами PWMA и PWMB:

- переключение в единицу или ноль;
- инверсия (переключение в противоположное состояние);
- сохранение без изменений.

Поведение сигналов задается независимо друг от друга. Кроме этого, обработчик событий позволяет программно задавать состояние сигналов PWMA и PWMB и величину «мертвого времени» ШИМ. Управление работой блока производится посредством регистров AQCTLA, AQCTLB, AQSFRC, AQCSFRC.

Существует вероятность того, что несколько событий могут произойти одновременно. Для таких ситуаций обработчик событий использует систему приоритетов событий.

Таблица 12.1 – Распределение приоритетов событий при счете вверх

Событие	Приоритет
Программное	1 (наивысший)
CTR = TBPRD	2
CTR = CMPB (счет вверх) при счете вверх	3
CTR = CMPA (счет вверх) при счете вверх	4 (низший)

Таблица 12.2 – Распределение приоритетов событий при счете вниз

Событие	Приоритет
Программное	1 (наивысший)
CTR = Zero	2
CTR = CMPB (счет вниз) при счете вниз	3
CTR = CMPA (счет вниз) при счете вниз	4 (низший)

Таблица 12.3 – Распределение приоритетов событий при счете вверх-вниз

Событие	Приоритет
Программное	1 (наивысший)
CTR = CMPB (счет вверх) при счете вверх или CTR = CMPB (счет вниз) при счете вниз	2
CTR = CMPA (счет вверх) при счете вверх или CTR = CMPA (счет вниз) при счете вниз	3
CTR = Zero или CTL = PRD	4
CTR = CMPB (счет вверх) при счете вниз или CTR = CMPB (счет вниз) при счете вверх	5
CTR = CMPA (счет вверх) при счете вниз или CTR = CMPA (счет вниз) при счете вверх	6 (низший)

В режиме счета вверх:

- если компаратор запрограммирован так, что CMPA/CMPB ≤ TBPRD (счет вверх), то событие произойдет при CTR = CMPA/CMPB;
- если компаратор запрограммирован так, что CMPA/CMPB > TBPRD (счет вверх), то событие не произойдет;
- если компаратор запрограммирован на срабатывание при счете вниз, то событие не произойдет.

В режиме счета вниз:

- если компаратор запрограммирован так, что CMPA/CMPB ≤ TBPRD (счет вниз), то событие произойдет при CTR = CMPA/CMPB;
- если компаратор запрограммирован так, что CMPA/CMPB ≥ TBPRD (счет вниз), то событие произойдет при CTR = TBPRD;
- если компаратор запрограммирован на срабатывание при счете вверх, то событие не произойдет.

В режиме счета вверх-вниз (см. рисунок 12.8):

- если счетчик считает вверх, а компаратор запрограммирован так, что CMPA/CMPB < TBPRD (счет вверх), то событие произойдет при CTR = CMPA/CMPB;
- если CMPA/CMPB ≥ TBPRD (счет вверх), то событие произойдет при CTR = TBPRD;
- если счетчик считает вниз, а компаратор запрограммирован так, что CMPA/CMPB < TBPRD (счет вниз), то событие произойдет при CTR = CMPA/CMPB;
- если CMPA/CMPB ≥ TBPRD (счет вверх), то событие произойдет при CTR = TBPRD.

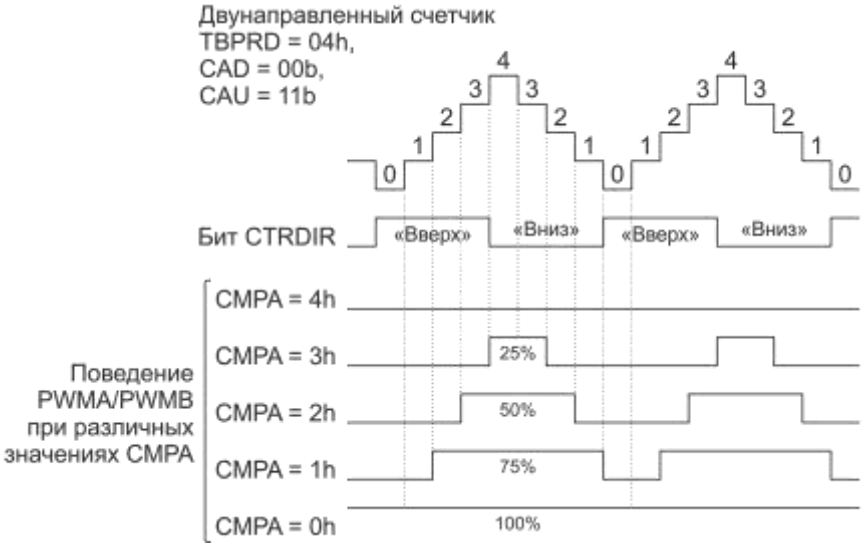


Рисунок 12.8 – Работа таймера при счете вверх-вниз с симметричным выходом (центрированная модуляция)

На рисунках 12.9 – 12.11 показано поведение линий PWMA и PWMB при различных видах модуляции. На рисунках приняты обозначения, пояснения к которым приведены в таблице 12.4.

Таблица 12.4 – Пояснения к обозначениям на рисунках 12.9 – 12.11

Обозначение	Пояснение

<table>
<tr><td>P
×</td><td>CA
×</td><td>CB
×</td><td colspan="2">События CTR = PRD, CTR = CMPA, CTR = CTRB, соответственно. Символ «×» указывает на то, что при возникновении этого события сигнал на линии PWMA/PWMB остается без изменений. Пунктирными линиями отмечены моменты возникновения события. Так, например (см. рисунок 12.9), при возникновении события CTR = CTRB, сигнал на линии PWMA остается без изменения, а сигнал на линии PWMB переключается в ноль</td></tr>
<tr><td colspan="2">Z
↑</td><td colspan="2">Z
↓</td><td>Событие CTR = Zero</td><td rowspan="3">Символы «↑»/«↓» указывают на то, что при возникновении этого события сигнал на линии PWMA/PWMB переключается в единицу/ноль</td></tr>
<tr><td colspan="2">CA
↑</td><td colspan="2">CA
↓</td><td>Событие CTR = CMPA</td></tr>
<tr><td colspan="2">CB
↑</td><td colspan="2">CB
↓</td><td>Событие CTR = CMPB</td></tr>
</table>

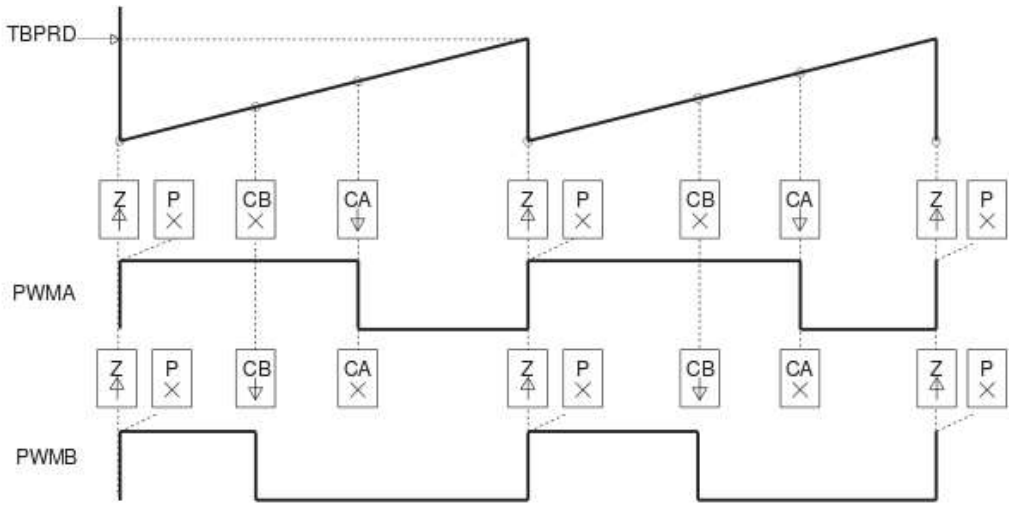


Рисунок 12.9 – Независимый режим работы выходов (фронтовая модуляция)

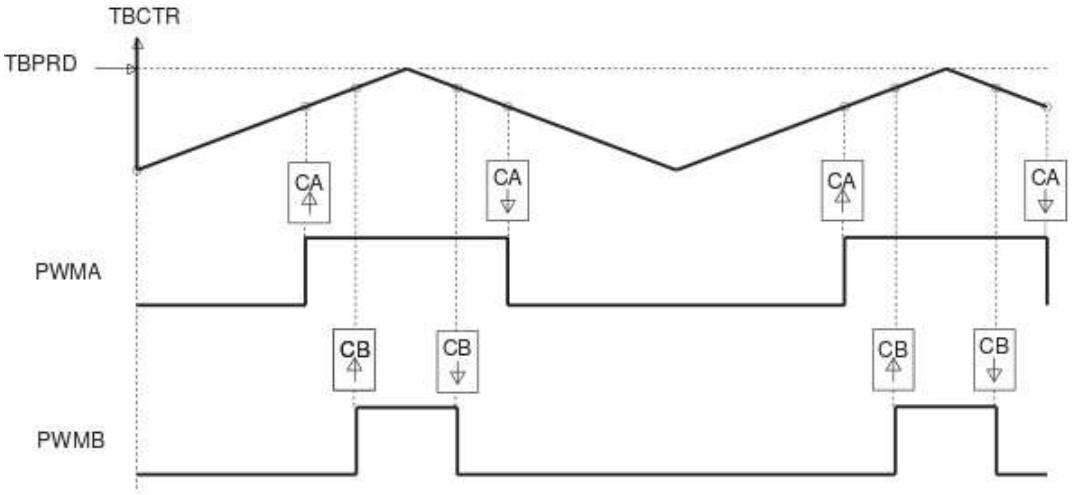


Рисунок 12.10 – Симметричный режим работы при счете вверх-вниз (центрированная модуляция)

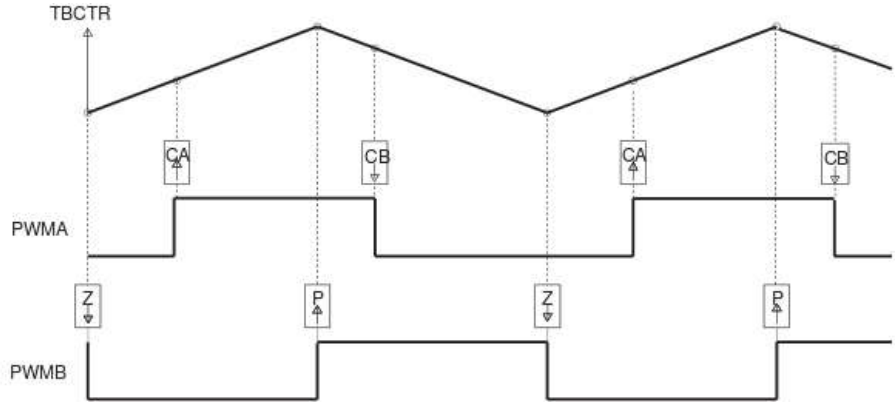


Рисунок 12.11 – Ассиметричный режим работы при счете вверх-вниз

12.4 Генератор задержки ШИМ

Блок имеет на входе сигналы ШИМ A и B с выходов обработчика событий, а на выходах повторяет эти сигналы, но со вставкой задержки («мертвое время») в момент переключения сигналов (если это необходимо).

Задержку можно учесть при программировании обработчика событий, но чтобы с высокой вероятностью избежать ошибок, желательно использовать генератор задержки ШИМ.

Основные функции генератора:

- генерация пары сигналов (PWMA и PWMB) с выдержкой интервалов (задержек) времени относительно сигнала PWMA;
- программирование задержки для активного высокого и активного низкого уровня сигналов каналов PWMA и PWMB;
- добавление программируемой задержки передних фронтов сигналов;
- добавление программируемой задержки для задних фронтов сигналов;
- возможность передачи сигналов с входов на выходы без изменений.

Генератор задержки ШИМ программируется посредством регистров DBCTL, DBRED и DBFED. Структурная схема генератора «мертвого времени» ШИМ представлена на рисунке 12.12.

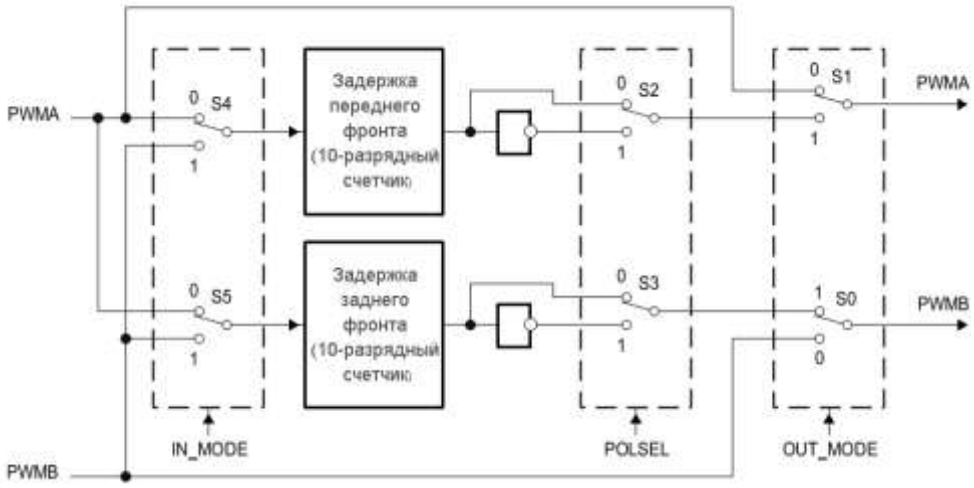


Рисунок 12.12 – Структурная схема генератора «мертвого времени» ШИМ

Функционирование

Генератор задержки ШИМ может работать с четырьмя источниками (фронты сигналов PWMA и PWMB). Выбор источника задается полем MODE регистра DBCTL.

Поле POLSEL позволяет задать инверсию (переключение значения на противоположное) сигнала после внесения задержки (см. рисунок 12.13).

Величины задержек по переднему и заднему фронту программируются раздельно посредством регистров DBRED и DBFED, соответственно.

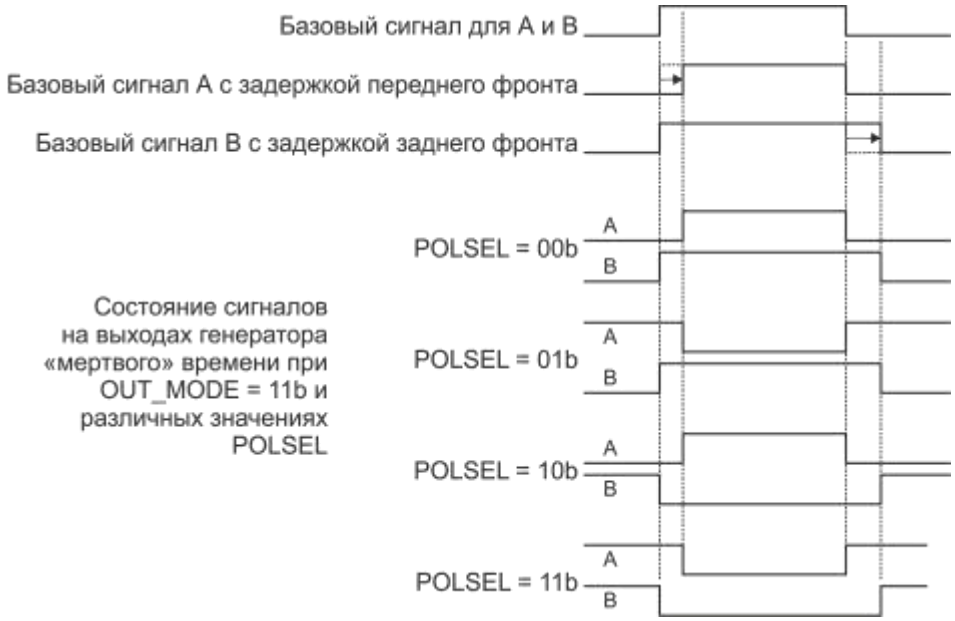


Рисунок 12.13 – Временные диаграммы работы генератора «мертвого времени» в типовой конфигурации

12.5 Фильтр коротких импульсов

Фильтр коротких импульсов предназначен для блокирования прохождения на выход импульсов с длительностью меньше заданной. Этот блок может применяться, если драйвер силового ключа инвертора не имеет такой функции, а для обеспечения правильного режима работы транзистора необходимо запретить открытие/закрытие транзистора на очень короткое время.

Основные функции фильтра:

- программируемая ширина минимального пропускаемого импульса;
- фильтр может быть отключен, если он не требуется.

Ширина минимального импульса, допускаемого к прохождению на выход, задается в регистре FWDTH и может принимать значение от 00h (фильтр выключен) до 0Fh. Импульсы длительностью меньше заданной пропускаться не будут.

12.6 Модулятор

Блок позволяет модулировать выходной ШИМ сигнал с помощью высокочастотных импульсов программируемой скважности. Модулирование требуется для управления силовыми ключами через импульсный трансформатор.

Основные функции модулятора:

- программируемая частота;
- программируемая ширина первого импульса;
- программируемая скважность второго и последующего импульсов;

- модулятор может быть отключен (бит CHPEN регистра PCCTL).

Модулятор программируется посредством регистра PCCTL. Структурная схема модулятора приведена на рисунке 12.14.

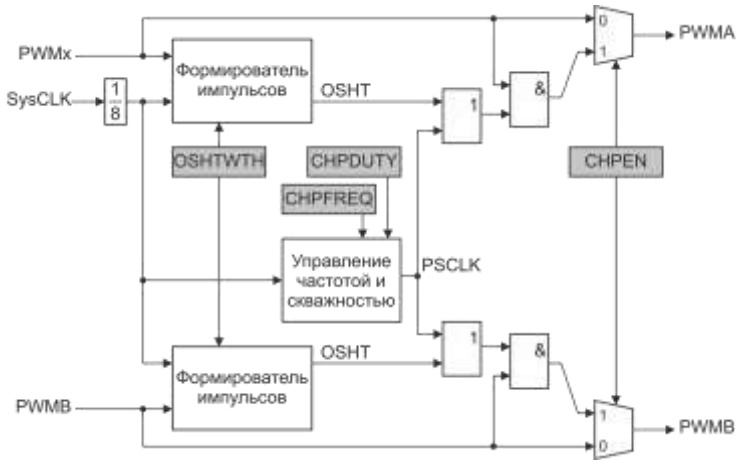


Рисунок 12.14 – Структурная схема модулятора

Функционирование

Ширина первого импульса программируется независимо полем OSHTWTH, это требуется для открывания ключа. Для остальных импульсов частота модуляции формируется на основе системной частоты при помощи делителя, программируемого полем SHPFREQ. Скважность импульсов программируется полем CHPDUTY.

Значения поля OSHTWTH лежат в диапазоне 0h – Fh.

Ширина L первого импульса определяется по формуле

$$L = T \times 8 \times (OSHTWTH + 1), \qquad (12.1)$$

где T – период синхросигнала SysCLK.

Значения поля CHPDUTY лежат в диапазоне 0h – 7h.

Скважность D (с шагом 12,5 %) последующих импульсов определяется по формуле

$$D = 12{,}5 \times (CHPDUTY + 1). \qquad (12.2)$$

На рисунке 12.15 приведен пример временных диаграмм работы модулятора.

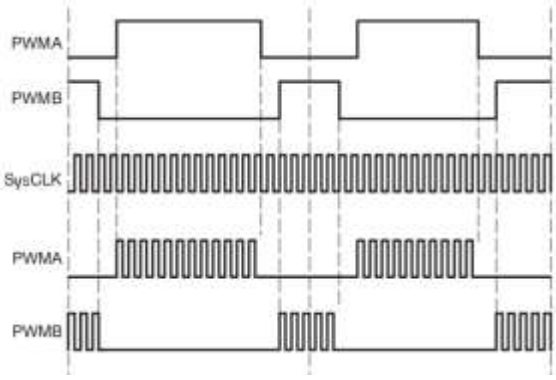


Рисунок 12.15 – Временные диаграммы работы модулятора

12.7 Детектор сигнала аварии

Блок контролирует выходы PWM_A и PWM_B и может переводить их в определенное (запрограммированное) состояние в случае, если поступит сигнал аварии.

Основные функции:

- входные сигналы аварии с выводов микроконтроллера PWM_TZ0 – PWM_TZ2 могут использоваться любым блоком ШИМ;
- в случае если поступит сигнал аварии, выводы ШИМ могут быть переведены в одно из состояний: логического нуля, логической единицы, высокоимпедансное или оставлены без изменения;
- поддерживается однократная блокировка выводов для ситуации короткого замыкания или перегрузки по току;
- поддерживается циклическая блокировка для режима ограничения тока;
- каждый входной источник сигнала аварии может быть обработан в однократном и циклическом режимах;
- поддерживается программная генерация сигнала аварии;
- детектор сигнала аварии может быть отключен, если он не требуется.

Детектор сигнала аварии программируется посредством регистров TZSEL, TZCTL, TZEINT, TZFLG, TZCLR и TZFRC.

Функционирование

Структурная схема детектора сигналов аварии показана на рисунке 12.16.

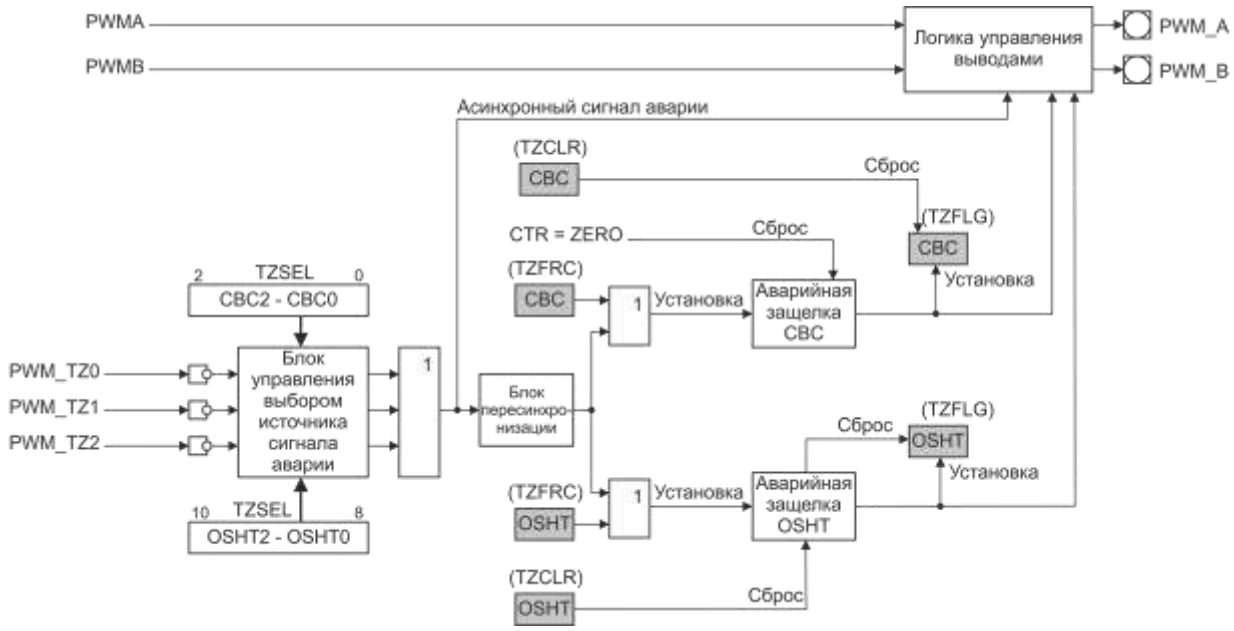


Рисунок 12.16 – Структурная схема детектора сигналов аварии

Переход входных сигналов аварии PWM_TZn из состояния логической единицы в состояние логического нуля формирует событие аварии. Каждый блок ШИМ может использовать или не использовать эти события в своей работе (программируется посредством регистра TZSEL). События могут формироваться синхронно (с цифровым фильтром помех) или асинхронно (программируется через регистры GPIO микроконтроллера). При синхронной обработке длительность импульса на входном сигнале сбоя должна быть не меньше периода синхросигнала TBCLK. Если же обработка производится в асинхронном режиме, то событие формируется и обрабатывается даже в

том случае, если по какой-либо причине отключилось тактирование. Каждый входной сигнал аварии должен быть настроен на однократное или циклическое формирование события аварии (программируется посредством регистра TZSEL).

При получении события аварии в режиме циклической обработки немедленно выполняется действие, заданное регистром TZCTL, и устанавливается флаг CBC в регистре TZFLG, а также генерируется прерывание PWM_TZINT (если разрешено в регистре TZEINT и контроллером прерываний). Аварийное удержание выводов заканчивается по событию TBCTR = 0000h при условии, что событие аварии уже неактивно. Таким образом, в режиме циклической обработки событие аварии сбрасывается в каждом периоде ШИМ, хотя флаг аварии CBC остается установленным до принудительного программного сброса. Если после сброса регистра флага CBC вновь будет получено событие аварии, то флаг установится вновь. На рисунке 12.17 показана схема формирования прерывания.

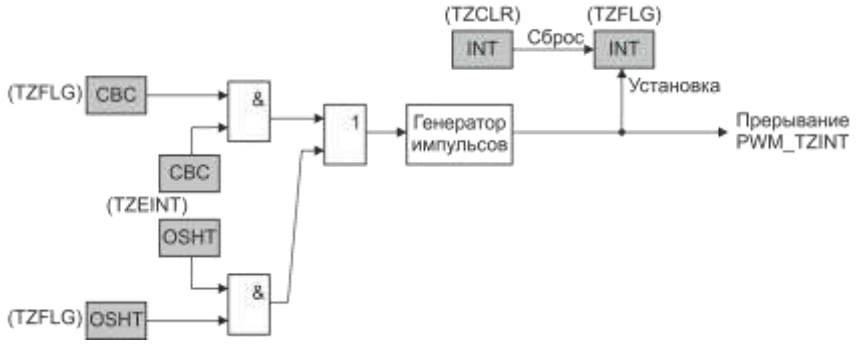


Рисунок 12.17 – Схема формирования прерывания

При получении события аварии в режиме однократной обработки немедленно выполняется действие, заданное регистром TZCTL, и устанавливается флаг OST в регистре TZFLG, а также генерируется прерывание PWM_TZINT (если разрешено в регистре TZEINT и контроллером прерываний). Аварийное удержание выводов заканчивается после принудительного программного сброса записью в бит OST регистра TZCLR.

Аварийное состояние выводов при получении события сбоя программируется индивидуально для выходов PWM_A и PWM_B полями TZA и TZB регистра TZCTL.

12.8 Триггер событий

Основные функции:

- получение событий, сформированных таймером и компаратором;
- использование информации о направлении счета (вверх-вниз);
- использование делителя событий, для формирования сигнала прерывания;
- предоставление доступа процессора к содержимому регистра флагов событий и счетчикам событий.

12.9 Функционирование

Триггер событий (не показан на рисунке 12.1) программируется посредством регистров ETSEL, ETPS, ETFLG, ETCLR и ETFRC. Функциональная схема триггера событий показана на рисунке 12.18.

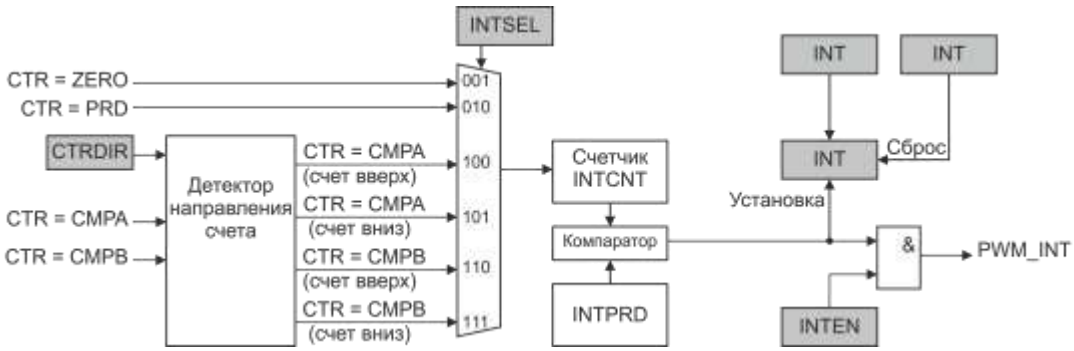


Рисунок 12.18 – Структурная схема триггера событий

Триггер может генерировать прерывания (если разрешено битом INTEN регистра ETSEL) по каждому первому, второму или третьему событию (поле INTPRD), которое задается полем INTSEL. Количество возникших событий отражается в поле INTCNT. Счетчик INTCNT считает от 00b до INTPRD и сбрасывается только вместе с отправкой активного прерывания.

Когда возникает совпадение INTCNT и INTPRD, то возможны варианты:

- если прерывание разрешено и сброшен флаг INT (регистр ETFLG), то генерируется прерывание и устанавливается флаг INT, а счетчик INTCNT сбрасывается в 00b и начинает считать заново;
- если прерывание запрещено или флаг INT установлен, то счетчик перестает считать события;
- если прерывание разрешено, но флаг от предыдущего прерывания еще не сброшен, то счетчик хранит свое максимально достигнутое значение (INTCNT = INTPRD) до сброса флага INT. Это позволяет обработать еще прерывание, пришедшее за то время, пока обрабатывалось предыдущее.

Каждая запись в INTPRD сбрасывает счетчик INTCNT. Запись единицы в бит INT регистра ETFRC увеличит значение счетчика на единицу. Если значение INTPRD = 00b, то счетчик отключен, а входные события игнорируются.

13 Блок высокоскоростного ввода-вывода HSIO

Два модуля – модуль высокоскоростного ввода HSI и модуль высокоскоростного вывода HSO, входящие в состав микроконтроллера, – формируют единый модуль HSIO, который может вести подсчет импульсов и импульсных последовательностей, измерять ширину поступающих импульсов, формировать импульсные последовательности и генерировать прерывания.

13.1 Модуль высокоскоростного ввода HSI

Основным назначением модуля HSI является мониторинг состояния своих входов HSI0 – HSI15 (альтернативные функции выводов A16 – A31 микроконтроллера) и фиксирование времени обнаружения запрограммированных событий (до 128) с использованием таймера с возможностью последующей выдачи сохраненной информации. Функциональная схема модуля показана на рисунке 13.1.

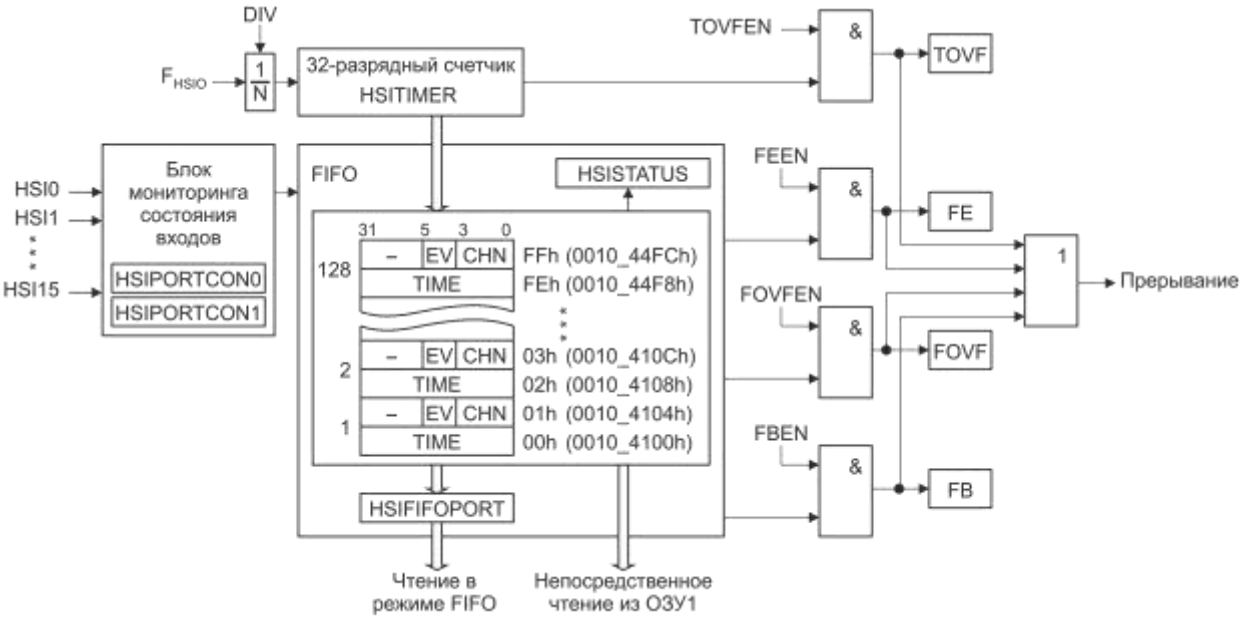


Рисунок 13.1 – Функциональная схема модуля HSI

Таймер модуля представляет собой 32-разрядный однонаправленный счетчик, инкрементирующийся в начале каждого машинного цикла (т. е. с частотой сигнала CLOCKOUT, равной частоте сигнала на входе XTAL1). Регистр HSITIMER содержит значение счетчика и всегда доступен для записи/чтения.

Частота входного синхросигнала таймера может быть уменьшена посредством делителя, значение которого задается полем DIV регистра HSICON. Для включения делителя значение DIV должно быть отлично от нуля. Таймер включается установкой бита START регистра HSISTATUS.

При переполнении таймера и если установлен бит TOVF, устанавливается флаг TOVF в регистре HSISTATUS, и генерируется прерывание.

Блок мониторинга состояния входов управляется регистрами HSIPORTCON0 и HSIPORTCON1, которые задают ожидаемые события для всех входов (каналов) модуля HSI.

Каждый вход оснащен защелкой, которая каждый такт синхросигнала CLOCKOUT фиксирует состояние входа, детектором фронта входного сигнала и счетчиком фронтов. Для правильного детектирования длительность стабильного состояния сигнала на входе должна быть не менее периода сигнала CLOCKOUT.

Для каждого канала независимо может быть задан один из 16 режимов работы посредством сответствующего поля CHx (x – номер канала), см. таблицу 13.1

Таблица 13.1 – Программируемые события на выводе

Номер режима	Ожидаемое событие при мониторинге входного сигнала		Значение EV
0h	мониторинг отключен		00b
1h	каждый положительный фронт		01b
2h	каждый отрицательны фронт		10b
3h	каждый	положительный фронт	01b
		отрицательный фронт	10b
4h	каждый второй положительный фронт		11b
5h	каждый четвертый положительный фронт		
6h	каждый шестой положительный фронт		
7h	каждый восьмой положительный фронт		
8h	каждый второй отрицательный фронт		
9h	каждый четвертый отрицательный фронт		
Ah	каждый шестой отрицательный фронт		
Bh	каждый восьмой отрицательный фронт		
Ch	каждый второй фронт		
Dh	каждый четвертый фронт		
Eh	каждый шестой фронт		
Fh	каждый 8 фронт		

Как только на входе обнаруживается запрограммированное событие, в накопительный буфер FIFO загружается 38-разрядное слово, состоящее из 32-разрядного значения таймера TIME (времени события), двух битов EV (индикатор типа события, см. таблицу 13.1) и четырех битов CHN (номер канала). Время загрузки в FIFO составляет один такт сигнала CLOCKOUT. Если два события обнаруживаются одновременно, они записываются с одинаковыми временами.

Если задан режим детектирования с использованием счетчиков фронтов (режимы 4h – Fh), то следует помнить, что при смене режима мониторинга счетчики не сбрасываются. При выключении модуля HSI сбросом бита START, счетчики также не сбрасываются.

Объем накопительного буфера FIFO позволяет сохранять до 128 слов, т. е. хранить информацию о 128 событиях. Как только FIFO заполняется полностью, прием информации прекращается до тех пор, пока не появится свободное место.

32-разрядные регистры буфера FIFO расположены в ОЗУ1 микроконтроллера и доступны в диапазоне адресов 0010_4100h – 0010_44FFh, см. рисунок 13.1. Все эти регистры доступны только для чтения, запись в них не имеет эффекта. Под каждое сохраняемое 38-разрядное слово отводятся два регистра – первый для хранения времени, второй для хранения индикатора события и номера канала.

Весь буфер FIFO можно очистить записью единицы в бит CLRBUFF регистра HSISTATUS.

Внутри блока FIFO регистры имеют собственные короткие адреса от 00h до FFh (порядковые номера).

Поле BUFWR регистра HSISTATUS представляет собой указатель на регистр накопительного буфера FIFO, начиная с которого будет произведена запись при обнаружении запрограммированного события. Начальное состояние указателя 00h, и по мере заполнения буфера оно будет инкрементироваться.

Поскольку модуль HSI мониторит 16 независимых входов, то в каждый момент времени может быть обнаружено от 0 до 16 событий, информация о которых будет записана в накопительный буфер FIFO, согласно приоритету каналов. Наивысший приоритет имеет

канал 0, низший – канал 15. На рисунке 13.2 показан пример одновременного обнаружения на входах HSI2, HSI7 и HSI12 запрограммированных событий и порядок записи информации о них в буфер FIFO.

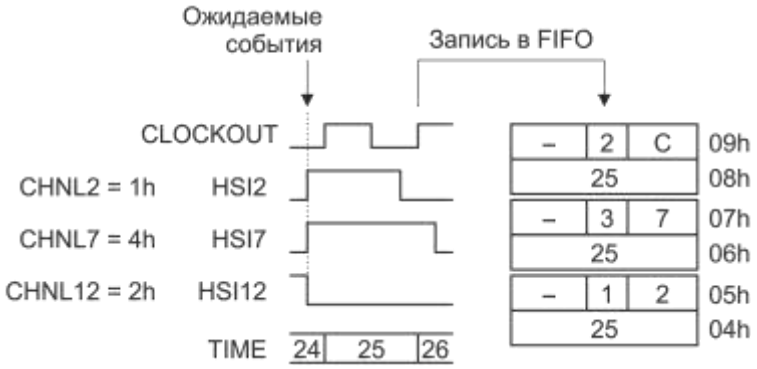


Рисунок 13.2 – Влияние приоритета канала на порядок записи в буфер FIFO

Поле BUFRD представляет собой указатель на регистр накопительного буфера FIFO, начиная с которого будет произведено чтение. Чтение информации из блока FIFO в данном случае производится через выходной буфер – регистр HSIFIFOPORT. Сначала читается 32-разрядное значение TIME и следующим циклом – информация о типе события EV и канале CHN. При каждом таком чтении происходит инкремент указателя BUFRD. Следует помнить, что при чтении регистров меняется только содержимое указателя, а содержимое прочитанных регистров сохраняется до тех пор, пока не будет перезаписано.

Поле BUFFILL является индикатором заполнения накопительного буфера FIFO. В отличие от указателей, которые ссылаются непосредственно на короткие адреса регистров буфера, индикатор BUFFILL хранит информацию об использовании накопительного буфера. При этом значение 00h означает, что буфер пуст, а значение FFh – что 255 регистров из 256 заняты. Если возникает ситуация, когда буфер FIFO заполнится на 100 %, т. е. все 256 регистров будут содержать непрочитанную информацию, поле BUFFILL сбросится в ноль, но при этом установится флаг FFULL регистра HSISTATUS. Данный флаг будет свидетельствовать о том, что буфер заполнен на 100 %, и он больше не может принимать информацию.

По мере чтения информации через HSIFIFOPORT и, следовательно, очистки накопительного буфера (при условии, что на входах в данный момент не обнаружено новых событий), флаг FFUL сбросится, а поле BUFFILL снова будет хранить актуальную информацию о состоянии буфера.

Примечание – Поля BUFWR, BUFRD, BUFFILL и флаг FFULL недоступны для записи и сохраняют свое состояние, когда модуль HSI выключен (сброшен бит START), а, следовательно, чтение регистра HSIFIFOPORT будет приводить к тому, что будет читаться один и тот же регистр, на который указывает BUFRD (поскольку модуль выключен, состояние этого указателя не изменяется). При очистке буфера FIFO битом CLRBUF регистра HSISTATUS поля BUFWR, BUFRD, BUFFILL и флаг FFULL также очищаются.

Режимы работы модуля

Модуль HSI способен функционировать в двух режимах – режиме FIFO и режиме постраничного анализа.

В режиме FIFO чтение накопительного буфера осуществляется через регистр HSIFIFOPORT, при этом подразумевается непрерывное использование модуля, когда программа пользователя отслеживает состояние FIFO с помощью прерываний и флагов и позволяет своевременно (во избежание переполнения) считывать информацию из буфера FIFO с целью избежать пропусков запрограммированных событий.

Вариантом работы с модулем HSI является режим постраничного анализа, который предполагает использование накопительного буфера FIFO в качестве страницы данных, содержащей 128 слов. Программа пользователя не считывает информацию из буфера через регистр HSIFIFOPORT, а ожидает полного заполнения буфера FIFO, после чего выключает модуль HSI сбросом бита START. После этого регистры буфера читаются программой непосредственно по их адресам в общем адресном пространстве. По завершении чтения рекомендуется очистить буфер FIFO записью единицы в бит CLRBUFF регистра HSISTATUS, а для продолжения работы с модулем включить его битом START.

Примечание – Режим FIFO и режим постраничного анализа не взаимоисключающие. Регистры буфера всегда могут быть прочитаны последовательно через регистр HSIFIFOPORT или непосредственно по их адресам в общем адресном пространстве. В режиме FIFO обращение к регистрам буфера не оказывает влияние на состояние указателей.

Прерывания

Модулю HSI выделена одна линия прерываний (int7), на которую можно подключить до четырех внутренних линий прерываний: одна – от таймера и три – от буфера FIFO. Разрешением прерываний управляют биты регистра HSICON, флаги разрешенных прерываний устанавливаются в регистре HSISTATUS. Каждый флаг прерывания устанавливается, если разрешено соответствующее ему прерывание. При возникновении прерывания следует проверять состояние всех флагов для определения источника прерывания. Флаги не сбрасываются аппаратно, а только программно. Рекомендуется сбрасывать все флаги, независимо от количества разрешенных прерываний.

Переполнение таймера

Прерывание от таймера возникает при его переполнении. Разрешением прерывания управляет бит TOVFEN, флагом прерывания является бит TOVF.

Переполнение FIFO

Прерывание от FIFO возникает, когда накопительный буфер заполнен настолько, что не может принять информацию. Разрешением прерывания управляет бит FOVFEN, флагом прерывания является бит FOVF. Прерывание по переполнению FIFO может возникнуть по двум причинам:

1 После того как накопительный буфер был полностью заполнен, обнаруживается очередное событие, информация о котором уже не может быть сохранена.

2 В ситуации, когда накопительный буфер заполнен не на 100 %, и на входах одновременно обнаруживаются несколько событий, для сохранения информации о которых требуется больше места, чем есть в буфере. Например, свободны шесть регистров (для трех событий), а обнаружены пять событий. В этом случае запись в накопительный буфер не производится и информация о событиях теряется.

Граница заполнения

Во избежание потери информации следует своевременно читать накопительный буфер. Для этого рекомендуется проверять состояние флага FFULL, а также состояние поля BUFFILL. Также можно задать значение границы заполнения накопительного буфера в поле FBORDER регистра HSICON. Как только значение индикатора заполнения BUFFILL достигнет или превысит значение FBORDER, и если установлен бит BORDEREN (разрешение прерывания по достижении границы заполнения), то будет сгенерировано прерывание, и установлен флаг FB.

Примечание – Отметим, что прерывание по достижении границы заполнения будет генерироваться до тех пор, пока значение индикатора BUFFILL не станет меньше значения FBORDER. В связи с этим при обслуживании прерывания рекомендуется читать регистр HSIFIFOPORT до тех пор, пока значение индикатора BUFFILL не станет значительно меньше значения FBORDER, и только после этого сбрасывать флаг FB.

Опустошение FIFO. Прерывание от FIFO возникает в случае, когда в накопительном буфере не осталось непрочитанных регистров (значения указателей BUFRD и BUFWR совпадают, а значение индикатора BUFFILL равно нулю). Также прерывание генерируется при программной очистке буфера битом CLRBUF. Разрешением прерывания управляет бит FEMPTYEN, флагом прерывания является бит FEMPTY.

13.2 Модуль высокоскоростного вывода HSO

Основным назначением модуля является сравнение запрограммированного значения/значений времени с текущим значением подключенного таймера и выполнение заданной команды (нескольких команд) при их совпадении. Выполняя заданные команды, модуль HSO может управлять своими выходами HSO0 – HSO15. Модуль HSO позволяет формировать независимые импульсные последовательности. Функциональная схема модуля показана на рисунке 13.3.

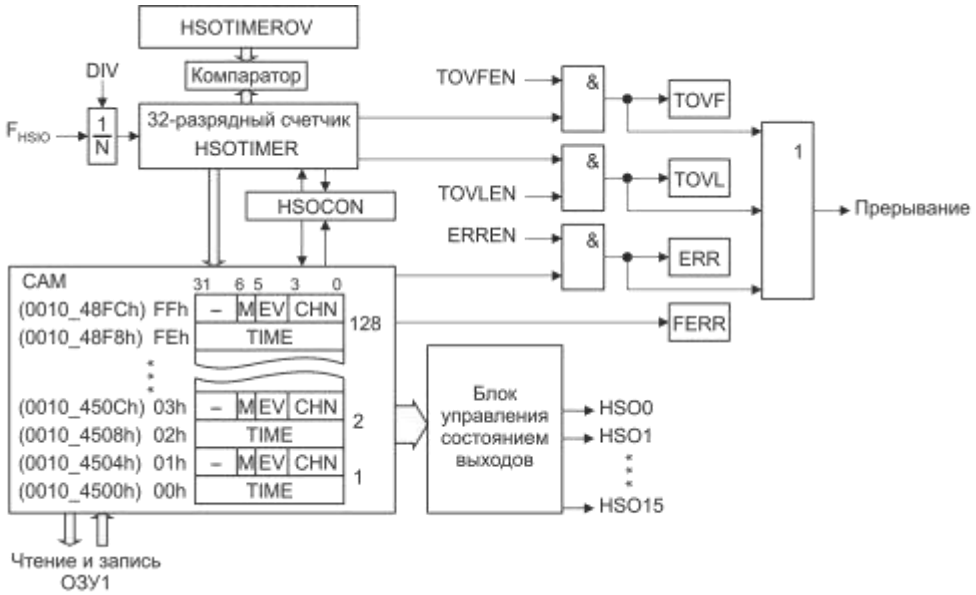


Рисунок 13.3 – Функциональная схема модуля HSO

Основным блоком модуля HSO является ассоциативное запоминающее устройство CAM для 128 39-разрядных расширенных слов.

Каждое слово CAM включает в себя 32-разрядное значение времени TIME для задания момента выполнения команды: 4 бита – номер канала CHN, 2 бита – команды EV и 1 бит – сохранения записи M.

Регистры CAM расположены в общем адресном пространстве микроконтроллера и выравнены по 4 байтам. Все они доступны для чтения и записи. Все регистры CAM можно очистить, если записать единицу в поле BC регистра HSOCON.

Бит M позволяет зафиксировать команду в CAM, в результате чего команда будет выполняться столько раз, сколько таймер будет достигать значения времени выполнения. Незафиксированная команда будет выполнена всего один раз, после чего ячейка CAM, в которой она была размещена, полностью будет очищена и готова для загрузки нового слова.

В зависимости от выполняемой команды (команд) модуль HSO может оказывать влияние на свои выходы HSO0 – HSO15. Какой именно выход будет задействован, определяется полем CHN регистров CAM. Влияние заключается в том, что модуль может устанавливать на этих выходах высокий и низкий уровни сигнала, также можно переключить вывод в третье состояние (Z-состояние). Поле EV задает уровень сигнала, который будет установлен на выходе (выходах) в результате выполнения команды, см. таблицу 13.2.

Таблица 13.2 – Команды CAM

Значение EV	Команды
00b	Ничего не делать
01b	Переключить вывод в состояние логической единицы
10b	Переключить вывод в состояние логического нуля
11b	Переключить вывод в третье состояние

Каждой команде в CAM соответствует определенное значение времени, при этом несколько команд могут иметь одинаковые значения времени. Если несколько команд с одинаковым значением времени и одинаковым выбранным каналом в поле CHN имеют разные значения в поле EV, то соответствующий вывод не изменит своего текущего состояния, а в регистре HSOCON будет активным флаг FERR. Данный флаг свидетельствует о возникновении ситуации, описанной выше, после чтения флаг FERR необходимо сбросить. Например, если в одной ячейке CAM записана команда HSO0, выполнение которой переключает выход HSO0 в единицу, а в другой ячейке, с тем же значением времени TIME, выбран тот же канал – HSO0, но в поле EV – значение 00b, т. е., – ничего не менять, то состояние выхода HSO0 не изменится, и активируется флаг FERR. После этого, если команды не зафиксированы, они будут удалены из CAM. Данному флагу FERR соответствует прерывание ERR регистра HSOCON; для того, чтобы разрешить его, необходимо установить бит SERR регистра HSOCON.

Бит ST управляет остановом/запуском модуля HSO. Установка бита приводит к запуску модуля, сброс – останавливает работу модуля. Пока модуль остановлен, все регистры также доступны для записи/чтения, но при этом они сохраняют свои значения, чтобы после старта модуль HSO мог продолжить работу.

Все значения времени, записанные в CAM, сравниваются с таймером одновременно в момент его переключения. Если при очередном переключении таймера обнаруживается совпадение с запрограммированным значением времени, то одновременно со следующим переключением таймера выполнится соответствующая команда. Если совпадений несколько, то выполняются несколько команд.

Таймер модуля HSO представлен независимым 32-разрядным счетчиком, который инкрементируется/декрементируется каждый такт машинного цикла. Регистр HSOTIMER содержит значение счетчика таймера и всегда доступен для записи/чтения. Таймер может работать через делитель, для этого достаточно в поле TIMER PRESCALER регистра HSOCON записать значение (отличное от нуля), на которое необходимо поделить тактовый сигнал. Таймер включается одновременно с включением модуля.

Направление счета таймера может быть любым и может меняться в процессе работы. Поле DIR регистра HSOCON определяет направление счета.

Независимо от направления счета таймер может генерировать прерывание переполнения таймера TOV. При прямом счете при прохождении значений из FFFF_FFFFh в ноль таймер сгенерирует прерывание TOV, при обратном счете прерывание будет сгенерировано при прохождении из нуля в FFFF_FFFFh. Данное прерывание может быть разрешено битом STOV в регистре HSOCON. В обработчике прерывания флаг TOV регистра HSOCON необходимо сбрасывать программно.

Регистр HSOTIMEROV содержит поле TIMER OVERLOAD, при достижении таймером значения, равного значению этого поля не равного нулю, может быть сгенерировано прерывание TOVL, если предварительно разрешить его, установив бит STOVL регистра HSOCON. Данное прерывание работает как для прямого, так и для обратного счета. Бит ROVL регистра HSOCON определяет, будет ли перезагружаться таймер при достижении значения TIMER OVERLOAD. Если бит ROVL сброшен, то таймер может генерировать прерывание TOVL, но при этом не перезагружается. Если бит ROVL установлен, то при прямом счете (бит DIR сброшен), по достижении значения TIMER OVERLOAD, таймер сбрасывается в ноль следующим тактом, при этом, если разрешены прерывания, то будут сгенерированы как TOVL, так и TOV. Если счет таймера обратный, то по достижении значения TIMER OVERLOAD таймером, последний перезагружается значением FFFF_FFFFh, и генерируются прерывания TOVL и TOV, если были разрешены, см. таблицу 13.3.

Таблица 13.3 – Команды CAM

ROVL	DIR	STOVL	STOV	Описание режима
0	0	1	1	При достижении таймером значения TIMER OVERLOAD, таймер не перезагружается, при этом генерируется прерывание TOVL, прерывание TOV будет сгенерировано при счете таймера из FFFF_FFFFh в ноль
	1	1	1	При достижении таймером значения TIMER OVERLOAD, таймер не перезагружается, при этом генерируется прерывание TOVL, прерывание TOV будет сгенерировано при счете таймера из ноля в FFFF_FFFFh
1	0	1	1	При достижении таймером значения TIMER OVERLOAD, таймер перезагружается в ноль, при этом генерируются оба прерывания TOVL и TOV
	1	1	1	При достижении таймером значения TIMER OVERLOAD, таймер перезагружается в FFFF_FFFFh, при этом генерируются оба прерывания TOVL и TOV

Все три прерывания TOVL, TOV и ERR модуля HSO приходятся на одну линию INT8. Также для каждого из этих событий можно запрограммировать возможность останова модуля при наступлении события. Для этого надо установить соответствующие биты OVLST, OVST и ERST регистра HSOCON (при этом не обязательно разрешать соответствующие прерывания):

- OVLST – остановить модуль HSO при достижении таймером значения перезагрузки TIMER OVERLOAD (TOVL);
- OVST – остановить модуль HSO при перезагрузке таймера (TOV);
- ERST – остановить модуль HSO при обнаружении ошибки в CAM (ERR).

Модуль останавливает таймер на следующем отсчете после наступления события. Например, если необходимо остановить таймер при переходе им значений из FFFF_FFFFh в ноль, то на момент останова в таймере уже будет значение 01h. Остановленный модуль HSO можно запустить, если установить бит ST регистра HSOCON.

14 Приемопередатчик UART

В состав микроконтроллера входят четыре идентичных универсальных асинхронных приемопередатчика UART0, UART1, UART2, UART3. Один из которых (UART0) расширен функционалом управления модемом (линии CTS, DCD, DSR, RTS, DTR и RI).

В состав приемопередатчика включен кодек последовательного интерфейса инфракрасной (ИК) передачи данных в соответствии с протоколом IrDASIR.

Также в состав приемопередатчика входят два буфера типа FIFO. Буфер приемника имеет разрядность 12, буфер передатчика – разрядность восемь. Каждый буфер может хранить до 32 байт данных, и каждый буфер может быть сконфигурирован (программно) как 32-байтный или как однобайтный.

Приемопередатчик обеспечивает:

- независимое маскирование прерываний от буфера передатчика, буфера приемника, по таймауту приемника, по изменению линий состояния модема, а также в случае обнаружения ошибки;
- возможность деления тактовой частоты в диапазоне от 1 до 65 535 (допускается использование нецелых коэффициентов деления, что позволяет использовать любой опорный генератор с частотой более 3,6864 МГц);
- возможность организации аппаратного управления потоком данных;
- поддержку функции управления модемом (линии CTS, DCD, DSR, RTS, DTR и RI).

Приемопередатчик реализует:

- передачу данных длиной от 5 до 8 бит со скоростью до 921 600 бит/с;
- контроль четности (проверочный бит выставляется по четности, нечетности, имеет фиксированное значение либо не передается);
- формирование одного или двух стоповых бит;
- обнаружение ложных стартовых битов;
- формирование и обнаружение сигнала разрыва линии.

Кодек ИК обмена данными IrDASIR обеспечивает:

- программный выбор обмена данными по линиям асинхронного приемопередатчика либо кодека ИК связи IrDASIR;
- поддержку функционирования с информационной скоростью до 115 200 бит/с в режиме полудуплекса;
- поддержку длительности бит для нормального режима и 3/16 для режима пониженного энергопотребления;
- программируемое деление опорной частоты для получения заданной длительности бит в режиме пониженного энергопотребления.

Наличие идентификационного регистра, однозначно идентифицирующего модуль, что позволяет операционной системе выполнять автоматическую конфигурацию.

Функциональные возможности

Режим работы приемопередатчика и скорость обмена данными контролируются регистром LCR_H и регистрами делителя IBRD и FBRD.

Устройство может формировать общее прерывание, возникающее в случае, возникновения прерывания от приемника (в том числе по таймауту), передатчика, а также по изменению состояния модема и в случае обнаружения ошибки.

В случае возникновения ошибки в структуре сигнала, четности данных, а также разрыва линии соответствующий бит ошибки устанавливается и сохраняется в буфере приемника. В случае переполнения буфера приемника также устанавливается соответствующий бит, а буфер становится недоступным для записи.

Приемопередатчик поддерживает режим модема (только UART0).

В случае активизации ИК передатчика, обмен информацией осуществляется не с помощью сигналов TX и RX, а посредством сигналов SIROUT и SIRIN.

В этом случае устройство переводит линию TXD в пассивное состояние (высокий уровень) и перестает реагировать на изменение состояния модема, а также сигнала на линии RXD. Протокол ИК передачи обеспечивает возможность обмена данными исключительно в режиме полудуплекса, то есть он не может передавать во время приема данных и принимать во время передачи данных.

Задержка между передачей и приемом должна составлять не менее 10 мс.

Функционирования ИК приемопередатчика

Кодер ИК передатчика

Кодер преобразует поток данных с выхода асинхронного передатчика, сформированный по закону модуляции без возврата к нулю (NRZ). Спецификация физического уровня протокола IrDASIR подразумевает использование модуляции с возвратом к нулю и инверсией (RZI), в соответствии с которой передача логического нуля соответствует излучению одного светового ИК импульса. Сформированный выходной поток импульсов подается на усилитель и, далее, на ИК светодиод.

Длительность импульса в режиме IrDA составляет три периода внутреннего тактового генератора с частотой Baud16, то есть, 3/16 периода времени, выделенного на передачу одного бита.

В режиме IrDA с пониженным энергопотреблением ширина импульса задана как 3/16 периода, выделенного на передачу бита, при скорости передачи данных 115 200 бит/с. Данное требование реализуется за счет формирования трех периодов тактового сигнала IrLPBaud16 с номинальной частотой 1,8432 МГц, в свою очередь, формируемого путем деления тактовой частоты модуля UART. Значение частоты IrLPBaud16 задается путем записи соответствующего коэффициента деления частоты в регистр ILPR.

Выход кодера имеет активное низкое состояние. При передаче логической единицы выход кодера остается в низком состоянии, при передаче логического нуля – формируется импульс, при этом выход кратковременно переводится в высокое состояние.

Как в нормальном режиме, так и в режиме пониженного энергопотребления использование нецелых значений коэффициента деления скорости передачи данных увеличивает джиттер фронтов импульсов данных. Наличие джиттера в случае использования дробных коэффициентов деления связано с тем, что интервалы между тактовыми импульсами будут нерегулярными – период сигнала в разное время будет содержать различное количество периодов тактового сигнала модуля UART. Можно показать, что в наихудшем случае величина джиттера в потоке ИК импульсов может достигать трех периодов синхросигнала.

Декодер ИК приемника

Декодер преобразует поток данных, сформированных по закону возврата к нулю, полученного от приемника ИК сигнала, и выдает поток данных без возврата к нулю на вход приемника UART. В неактивном состоянии вход декодера находится нормально в высоком состоянии. Выходной сигнал кодера имеет полярность, противоположную полярности входа декодера.

Обнаружение стартового бита осуществляется при низком уровне сигнала на входе декодера.

Примечание – Для того чтобы исключить ложные срабатывания UART от импульсных помех, на входе SIRIN игнорируются импульсы с длительностью менее, чем 3/16 длительности одного бита.

Функционирование блока UART

На рисунке 14.1 показана упрощенная функциональная схема приемопередатчика.

Генератор тактового сигнала приемопередатчика формирует синхросигнал последовательного обмена данными, который представляет собой последовательность импульсов с шириной, равной одному периоду тактового сигнала модуля UART, и частотой в 16 раз превышающей частоту передачи данных.

Буфер передатчика предназначен для хранения данных (полученных от ЦП) до тех пор, пока они не будут переданы внешнему устройству.

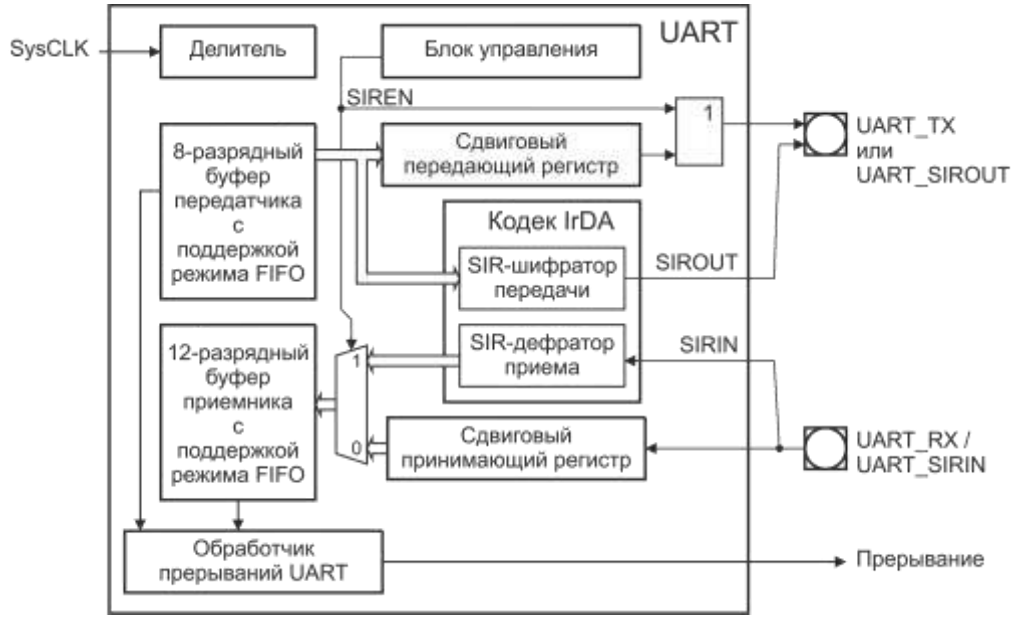


Рисунок 14.1 – Функциональная схема приемопередатчика

Буфер приемника предназначен для хранения данных и кодов ошибки (принятых от внешнего устройства) до тех пор, пока они не будут прочитаны ЦП.

Обработчик прерываний генерирует независимые маскируемые прерывания с активным высоким уровнем. Кроме того, формируется комбинированное прерывание путем объединения независимых прерываний по схеме ИЛИ.

Синхронизация

Существует ограничение на соотношение между тактовыми частотами SysCLK и UART_CLK:

$$UART\_CLK \leq 5/3 \times SysCLK \qquad (14.1)$$

Например, для достижения максимальной скорости передачи данных 921 600 бод (при Fuart_clk = 921 600 × 16 = 14,7456 МГц) частота SysCLK должна быть не менее 8,84736 МГц.

Для точной настройки частоты передачи данных используются два делителя. Коэффициент деления первого задается полем DIV_UART регистра UART_CLK. Коэффициент деления второго делителя имеет целую и дробную части, которые задаются регистрами IBRD и FBRD. Возможность задания нецелых коэффициентов деления позволяет осуществлять обмен данными на стандартных скоростях, используя в качестве источника тактовый сигнал с произвольной частотой более 3,6864 МГц.

Коэффициент деления K частоты сигнала UART_CLK рассчитывается по формуле

$$K = Fuart\_clk/(16 \times baud\_rate), \qquad (14.2)$$

где Fuart_clk – частота сигнала синхронизации блока UART в Гц;

baud_rate – скорость передачи в бодах.

Получившееся дробное десятичное число следует разделить на две части – целую и дробную.

Целая часть после преобразования в двоичный формат записывается в регистр IBRD.

Дробная часть умножается на 64 и округляется до ближайшего целого числа. Полученное число преобразовывается в двоичный формат и записывается в регистр FBRD.

Для примера, пусть требуемая скорость передачи данных 230 400 бит/с и частота тактового сигнала UART_CLK равна 4 МГц. Тогда:

$$K = (4 \times 10^6) / (16 \times 230\ 400) = 1{,}085.$$

Получившееся число разбивается на две части – 1 и 0,085.

В регистр IBRD записывается значение 0001h.

Значение (0,085 × 64) округляется и преобразовывается в 05h для записи в регистр FBRD.

Таким образом, реальные значения коэффициента деления частоты и скорости передачи будут следующими:

$$K = 1 + 5/64 = 1{,}078,$$

$$\text{baud\_rate} = (4 \times 10^6)/ (16 \times 1{,}078) = 231\ 911 \text{ бит/с}.$$

Ошибка установки скорости:

$$\Delta = ((231\ 911 – 230\ 400) / 230\ 400) \times 100\ \% = 0{,}656\ \%.$$

Максимальная ошибка установки скорости передачи данных:

$$\Delta = (1/64) \times 100\ \% = 1{,}560\ \%.$$

Такая ошибка возникает в случае K = 1, при этом разница накапливается в течение 64 тактовых интервалов.

Содержимое регистров LCR_H, IBRD и FBRD обновляется при записи в регистр LCR_H. Таким образом, для того, чтобы новые параметры коэффициента деления вступили в силу, после их записи в регистры IBRD и FBRD, необходимо осуществить запись в регистр LCR_H и только в такой последовательности.

Примечание – Изменение содержимого регистров IBRD, FBRD и LCR_H допускается только во время, когда приемопередатчик запрещен и не осуществляется передача/прием байта.

Передача и прием данных

Данные для передачи заносятся в буфер передатчика посредством записи в регистр DR. После записи хотя бы одного байта в буфер передатчика устанавливается флаг BUSY в регистре FR. Это состояние флага сохраняется, пока буфер передатчика не пуст (даже если работа приемопередатчика запрещена). Далее, если работа приемопередатчика разрешена (установлены биты UARTEN и TXE регистра CR), начинается передача информационного кадра с параметрами, указанными в регистре управления линией LCR_H. Передача данных продолжается до опустошения буфера передатчика (до окончания передачи всех байт). По окончании передачи сбрасывается флаг BUSY.

При приеме байта данных (установлены биты UARTEN и RXE регистра CR) для каждого бита производятся три выборки уровня, и решение о значении бита принимается по мажоритарному принципу.

В случае если приемник находился в неактивном состоянии (постоянный высокий уровень сигнала на линии UART_RxD), и произошла смена уровня входного сигнала с высокого на низкий (стартовый бит), включается счетчик, тактируемый внутренним сигналом, после чего отсчеты сигнала на входе приемника регистрируются каждые восемь тактов.

Стартовый бит считается достоверным в случае, если сигнал на линии UART_RxD сохраняет низкий логический уровень в течение восьми периодов внутреннего синхросигнала с момента включения счетчика. В противном случае переход в ноль рассматривается как ложный старт и игнорируется.

После обнаружения достоверного стартового бита очередной бит данных фиксируется каждые 16 отсчетов тактового сигнала. Производится регистрация всех бит данных (согласно запрограммированным параметрам) и бита четности (если включен режим контроля четности).

По окончании приема байта производится проверка присутствия корректного стопового бита (высокий логический уровень сигнала UART_RxD). После чего байт заносится в буфер приемника вместе с тремя битами признаков ошибки (см. рисунок 14.2) и битом переполнения буфера.

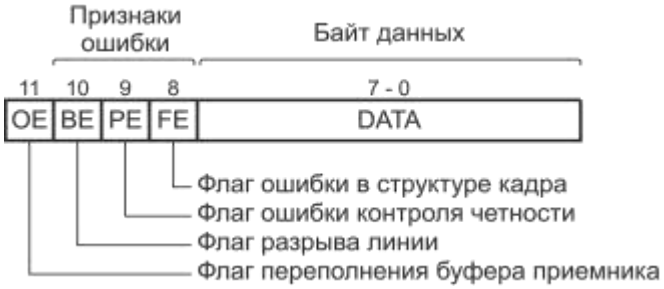


Рисунок 14.2 – 12-разрядная ячейка принимающего буфера

В 12-разрядной ячейке буфера байт данных располагается в области младших восьми бит, три бита признаков ошибки – в битах с 8 по 10.

Флаг переполнения буфера приемника выставляется в том случае, если к моменту, когда очередной кадр данных полностью принят, буфер уже заполнен. В этом случае принятый кадр остается в сдвиговом принимающем регистре и, в случае приема следующего кадра данных, будет потерян.

Как только в буфере приемника освобождается место для записи, кадр данных, находящийся в сдвиговом регистре, переписывается в буфер, а флаг переполнения сбрасывается.

Данные из буфера приемника можно прочитать посредством регистра DR. Состояние признаков ошибки и флага переполнения определяется чтением регистра RSR_ECR и относится к последнему байту, считанному из регистра DR, в связи с этим регистр DR всегда должен считываться первым.

Все флаги сбрасываются одновременно записью любого значения в регистр RSR_ECR или после сброса устройства.

Примечания

1 Необходимо запрещать работу приемопередатчика перед перепрограммированием его регистров управления. Если приемопередатчик переводится в отключенное состояние во время передачи или приема, то перед остановкой он завершает выполняемую операцию.

2 Целостность данных в буферах передатчика и приемника не гарантируется, если установился флаг BRK (разрыв линии), или если программное обеспечение произвело остановку приемопередатчика после его повторного перевода в разрешенное состояние.

Режим модема

Приемопередатчик может использоваться как оконечное устройство или как оборудование передачи данных. Сигналы модема в режиме оконечного устройства и их назначение представлено в таблице 14.1.

Таблица 14.1 – Назначение сигналов в режиме модема

Сигнал	Назначение в зависимости от режима работы		Режим работы вывода
	Оконечное устройство	Оборудование передачи данных	
UART_RTS	Готов к передаче данных	Запрос передачи данных	Выход
UART_CTS	Запрос передачи данных	Готов к передаче данных	Вход
UART_DTR	Приемник данных готов	Источник данных готов	Выход
UART_DSR	Источник данных готов	Приемник данных готов	Вход
UART_DCD	Обнаружен информационный сигнал	–	Вход
UART_RI	Индикатор вызова	–	Вход

Аппаратное управление потоком данных

Программно активируемый режим аппаратного управления потоком данных позволяет контролировать (приостанавливать и возобновлять) информационный обмен с помощью сигналов UART_RTS и UART_CTS. Иллюстрация взаимодействия двух устройств последовательной связи с аппаратным управлением потоком данных представлена рисунке 14.3.

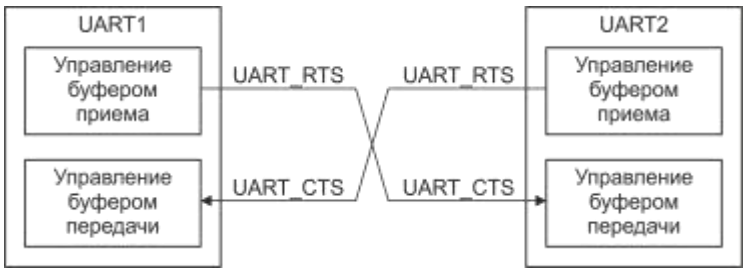


Рисунок 14.3 – Взаимодействие двух устройств последовательной связи с аппаратным управлением потоком данных

Если разрешено управление потоком данных по сигналу RTS, линия UART_RTS переводится в активное состоянии только после того, как в буфере приемника появляется заданное количество свободных ячеек.

Если разрешено управление потоком данных по сигналу CTS, передача данных осуществляется только после перевода линии UART_CTS в активное состояние.

Режим аппаратного управления потоком данных задается путем задания битов RTSEN и CTSEN в регистре управления CR.

Примечание – В случае если выбран режим управления потоком данных по RTS, программное обеспечение не может использовать бит RTSEN для проверки состояния линии RTS.

Логика управления потоком данных по RTS использует данные о превышении уровня заполнения буфера приемника. Сигнал на линии UART_RTS переводится в активное

состояние только после того, как в буфере приемника появляется заданное количество свободных ячеек. После достижения порогового уровня заполнения буфера приемника сигнал UART_RTS снимается (переводится в пассивное состояние), указывая, таким образом, на отсутствие свободного места для сохранения принятых данных. При этом дальнейшая передача данных должна быть прекращена по завершении передачи текущего кадра.

Обратно в активное состояние сигнал UART_RTS переводится после считывания данных из буфера приемника в количестве, достаточном для того, чтобы заполнение буфера оказалось ниже порогового уровня.

В случае если управление потоком данных по RTS запрещено, но при этом работа приемопередатчика разрешена, прием будет осуществляться до полного заполнения буфера приемника либо до завершения передачи данных.

В случае выбора одного из режимов с управлением потоком данных по CTS передатчик осуществляет проверку состояния линии UART_CTS перед началом передачи очередного байта данных. Передача осуществляется только в случае, если данная линия активна, и продолжается до тех пор, пока активное состояние линии сохраняется и буфер передатчика не пуст.

При переходе линии UART_CTS в неактивное состояние модуль завершает выдачу текущего передаваемого кадра, после чего передача данных прекращается.

Если управление потоком данных по CTS запрещено, и при этом работа приемопередатчика UART разрешена, данные будут выдаваться до опустошения буфера передатчика.

Работа кодека ИК обмена данными IrDASIR

Кодек обеспечивает сопряжение асинхронного потока данных, сформированного приемопередатчиком, с полудуплексным последовательным интерфейсом IrDASIR. Какая-либо аналоговая обработка сигнала при этом не выполняется. Назначение кодека – сформировать цифровой поток данных на вход приемника асинхронного сигнала и обработать цифровой поток данных с выхода передатчика.

Предусмотрено два режима работы.

В режиме IrDA уровень логического нуля передается на линию SIROUT в виде импульса с высоким логическим уровнем и длительностью 3/16 от выбранного периода следования бит данных. Логическая единица при этом передается в виде постоянного низкого уровня сигнала. Сформированный выходной сигнал далее подается на передатчик ИК сигнала, обеспечивая излучение светового импульса всякий раз при передаче нулевого бита. На приемной стороне световые импульсы воздействуют на базу фототранзистора ИК приемника, который в результате формирует низкий логический уровень. Это, в свою очередь, обуславливает низкий уровень на входе SIRIN.

В режиме IrDA с пониженным энергопотреблением длительность передаваемых импульсов ИК излучения устанавливается в три раза выше длительности импульсов внутреннего опорного сигнала IrLPBaud16. Данный режим активизируется путем установки бита SIRLP в регистре управления UARTCR.

Как в нормальном режиме, так и в режиме пониженного энергопотребления:

- кодирование осуществляется на основе бит данных, сформированных асинхронным передатчиком модуля;
- в ходе приема данных декодированные биты далее обрабатываются блоком асинхронного приема.

В соответствии со спецификацией физического уровня протокола IrDASIR, обмен данными должен осуществляться в режиме полудуплекса, при этом задержка между передачей и приемом данных должна составлять не менее 10 мс. Эта задержка должна формироваться программно. Необходимость ее введения обусловлена тем, что воздействие

передающего ИК светодиода на находящийся рядом ИК приемник может привести к искажению принимаемого сигнала или даже ввести приемный тракт в состояние насыщения. Задержка между окончанием передачи и началом приема данных именуется латентностью или временем установки (готовности) приемника.

Тактовый сигнал формируется путем деления частоты сигнала UARTCLK в соответствии с коэффициентом деления, записанным в регистре UARTILPR.

Коэффициент K деления вычисляется по формуле:

$$K = F\_UARTCLK / F\_IrLPBaud16, \quad (14.3)$$

где номинальное значение тактового сигнала Fbaud составляет 1,8432 МГц. Коэффициент деления должен быть выбран так, чтобы выполнялось соотношение:

$$1{,}42 \text{ МГц} < Fbaud < 2{,}12 \text{ МГц}.$$

На рисунке 14.4 показана модуляция данных в ИК режиме.

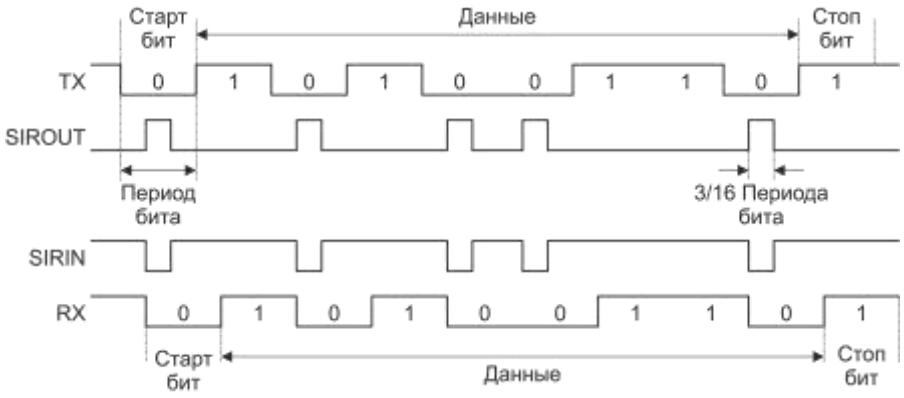


Рисунок 14.4 – Модуляция данных в ИК режиме

14.1 Прерывания

В модуле предусмотрено 11 маскируемых источников прерываний. В результате формируется один общий сигнал, представляющий собой комбинацию независимых сигналов, объединенных по схеме ИЛИ.

Сигналы запросов на прерывания:

- UART_RX – от приемника;
- UART_TX – от передатчика;
- UART_RT – по таймауту приемника;
- UART_MS – по состоянию модема;
- UART_E – по ошибке;
- UART_INT – логическое ИЛИ сигналов запросов на прерывания.

Каждый из сигналов может быть маскирован путем установки соответствующего бита в регистре маски IMSC.

Источник прерывания также можно определить, считав состояние регистра RIS или регистра MIS (маскированные прерывания).

В таблице 14.2 указаны значения параметров срабатывания прерываний приемника и передатчика в зависимости от порога заполнения буфера.

Таблица 14.2 – Параметры срабатывания прерываний в зависимости от порога наполнения буффера

Пороговый уровень	Количество незаполненных ячеек буфера передатчика	Количество заполненных ячеек буфера приемника
1/8	28	4
1/4	24	8
1/2	16	16
3/4	8	24
7/8	4	28

UART_RX

Прерывание возникает в случае обнаружения одного из событий:

- буфер приемника в режиме FIFO и его заполнение достигло заданного порогового значения;
- буфер приемника имеет одну ячейку (режим FIFO запрещен) и принят один кадр данных.

Линия прерывания переходит в высокое состояние и удерживается в нем до тех пор, пока из буфера не будет прочитан как минимум один байт или выполнен программный сброс прерывания (регистр ICR).

UART_TX

Прерывание возникает в случае обнаружения одного из событий:

- буфер передатчика в режиме FIFO и его опустошение достигло заданного порогового значения;
- буфер передатчика имеет одну ячейку (режим FIFO запрещен) и пуст.

Линия прерывания переходит в высокое состояние и удерживается в нем до тех пор, пока в буфер не будет записан как минимум один байт или выполнен программный сброс прерывания.

Запись данных в буфер передатчика допускается как перед разрешением работы приемопередатчика и прерываний, так и после разрешения.

Примечание – Прерывание передатчика работает по фронту, а не по уровню сигнала. В случае если работа приемопередатчика и прерывания от него разрешена до осуществления записи данных в буфер передатчика, прерывание не формируется. Прерывание возникает только при опустошении буфера.

UART_RT

Прерывание возникает в случае, если буфер приемника не пуст, и на вход приемника не поступало новых данных в течение периода времени, необходимого для передачи 32 бит. Прерывание сбрасывается после считывания данных из буфера приемника до его опустошения или программно.

UART_MS

Прерывание возникает в случае изменения любой из линий состояний модема (UART_CTS, UART_DCD, UART_DSR, UART_RI). Прерывание сбрасывается программно.

UART_E

Прерывание возникает в случае ошибки при приеме данных. Оно может быть вызвано:

- ошибкой в структуре кадра;
- ошибкой контроля четности;
- разрывом линии;
- переполнением буфера приемника.

Причину возникновения прерывания можно определить, прочитав содержимое регистра RIS либо регистра MIS.

Сброс прерывания осуществляется программно.

14.2 Программирование

Для программирования рекомендуется следующая последовательность действий:

- запретить работу приемопередатчика;
- дождаться окончания приема и/или передачи текущего байта данных;
- сбросить буфер передатчика посредством сброса бита FEN регистра LCR_H;
- изменить настройки регистра CR;
- разрешить работу приемопередатчика.

15 Контроллер интрефейса SpaceWire

Модуль обеспечивает взаимодействие микроконтроллера с другими устройствами как посредством подключения к сети, так и напрямую по протоколу SpaceWire. Связь осуществляется в полнодуплексном режиме. Скорость передачи/приема данных зависит от частоты работы микроконтроллера.

Особенности модуля последовательного интерфейса SpaceWire:

- возможность взаимодействия с устройствами, функционирующими на частоте, отличной от частоты работы модуля;
- 13 источников прерываний и возможность их выборочного включения;
- встроенные FIFO-буферы на прием и передачу;
- контроль состояний FIFO-буферов;
- возможность упрощенной отправки кодов времени;
- возможность корректировки временных соотношений состояний модуля;
- контроль паритета.

Общие сведения о сети SpaceWire

В общем случае сеть SpaceWire состоит из некоторого числа узлов-абонентов и маршрутизирующих коммутаторов (сетевых узлов). Узлы-абоненты представляют собой устройства, передающие и принимающие потоки данных. Узлы связаны с маршрутизирующими коммутаторами или друг с другом дуплексными каналами – линками. Каждый узел имеет один или несколько линк-портов и управляется хост-устройством, которое является источником данных. Хост-устройством может быть процессорный модуль, исполнительное устройство, датчик и т. п.

Маршрутизирующий коммутатор автономно обеспечивает передачу информации между своими входными и выходными портами. Принципиальное отличие узла-абонента от маршрутизирующего коммутатора в том, что трансляция данных между линк-портами узла-абонента, при необходимости, возможна только под управлением хост-устройства (т. е. реализуется программно).

Узел принимает данные от хост-устройства, кодирует их и отправляет в свой передатчик, подключенный непосредственно к линку. На другом конце линка данные принимает приемник, который декодирует их и передает другому хост-устройству или на выходной порт маршрутизирующего коммутатора.

Приемник и передатчик с необходимыми элементами управления и интерфейсом к хост-устройству образуют так называемый сетевой контроллер линка NIC (Network Interface Controller) сети SpaceWire. Контроллер линка управляет потоком данных в канале, контролирует соединение и рассоединение в канале, восстанавливает соединение после сбоев и др.

Для подробного ознакомления с протоколом передачи и организацией сети следует обратиться к спецификации ECSS-E-ST-50-12C «SpaceWire – Links, nodes, routers and networks» от 31.07.2008 г.

15.1 Инициализация и функционирование

Установление соединения инициируется записью единицы в бит LINKEN регистра управления CON. Далее следует установить один из двух битов LINKSTART или LINKAUTO (при одновременной установке бит LINKSTART имеет больший приоритет).

Передача и прием данных

Информация, передаваемая по сети SpaceWire, разбивается на пакеты. Размер пакета ненормирован стандартом. Пакет включает поле заголовка, содержащее адрес назначения, и поле данных, замыкаемое маркером конца пакета.

Маршрутизаторы, обрабатывающие адрес, по которому направляется пакет, обеспечивают также доставку пакета к нужному узлу, не входят в состав описываемого модуля. Механизм маршрутизации не будет рассмотрен. Предполагается, что модуль соединен с внешним устройством прямым каналом (точка-точка). В этом случае поле адреса остается пустым.

Для формирования пакета данных необходимо последовательно записывать данные, которые следует передать в регистр DATAT. Каждый байт, записанный в регистр DATAT, будет размещаться в FIFO-буфере. В FIFO-буфер могут быть загружены 128 байт (каждый со своим маркером EOP/EEP). После этого загрузка новых данных будет возможна только по мере того, как буфер будет освобождаться.

Степень заполнения буфера отражает поле CAPTX регистра состояния буферов CAPSTAT.

Данные из FIFO-буфера передаются в той же последовательности, в которой были записаны.

При приеме пакета данные будут последовательно записываться в принимающий FIFO-буфер. Всего могут быть приняты 128 байт (каждый со своим маркером EOP/EEP). После этого прием новых данных будет возможен только по мере того, как буфер будет освобождаться (при считывании полученных данных). Степень заполнения буфера отражает поле CAPRX регистра состояния буферов CAPSTAT.

Для чтения данных используется регистр DATAR. Данные считываются из принимающего FIFO-буфера в той же последовательности, в которой были приняты.

Передача и прием кодов времени

Для поддержания единого системного времени в сети существует специальный управляющий код времени.

Для отправки кода времени используется регистр TIMET. Код времени, который нужно передать, декрементируется на единицу и записывается в поле TRANSTIME. Бит TICK при этом устанавливать не следует. Т. е. в регистр TIMET записывается значение 00xxh, где «xx» – собственно 6-разрядный код времени.

Для инициации передачи нужно установить бит TICK. Это можно сделать записью значения 0100h в регистр TIMET.

Примечание – При инициации передачи кода времени установкой бита TICK в поле данных всегда записываются нули.

Как только бит TICK устанавливается, значение кода времени читается из поля TRANSTIME, инкрементируется на единицу и передается в сеть.

Модуль может принять код времени. Принятое значение записывается в поле RECTIME регистра TIMER. На аппаратном уровне постоянно производится проверка принятых кодов времени. Каждый последующий принятый код должен быть на единицу больше предыдущего. Если это условие соблюдается, то в конце приема каждого кода времени устанавливается флаг TIMEVALID в регистре TIMER. Если разница значений последовательно принятых кодов превышает единицу, код времени считается ошибочным, выставляется флаг TIMEINVALID, и значение внутреннего счетчика таймкодов будет перезаписано принятым значением RECTIME.

В случае если последовательно принятые коды совпадают, предполагается, что код времени принят повторно и ни один из двух флагов не устанавливается.

Контроль состояния

Состояние модуля отражает регистр STAT. Для управления прерываниями используется регистр INTMASK.

16 Контроллер интерфейса SPI

Контроллер интерфейса SPI реализует интерфейс последовательной синхронной связи в режиме ведущего (мастера) и ведомого устройства и обеспечивает обмен данными с подключенным ведомым или ведущим периферийным устройством в соответствии с одним из трех протоколов фирм Motorola, National Semiconductor, Texas Instruments.

Последовательный синхронный интерфейс SPI фирмы Motorola обеспечивает полнодуплексный обмен данными по четырехпроводной линии и программное задание фазы и полярности тактового сигнала.

Интерфейс Microwire фирмы National Semiconductor обеспечивает полудуплексный обмен данными с использованием 8-битных управляющих последовательностей.

Интерфейс SSI фирмы Texas Instruments обеспечивает полнодуплексный обмен данными по четырехпроводной линии и возможность перевода линии передачи данных в третье (высокоимпедансное) состояние.

Выбор интерфейса осуществляется посредством поля FRF регистра SPI_CR0.

В режиме мастера и в режиме ведомого устройства контроллер SPI обеспечивает:
- передачу данных, размещенных в буфере передатчика (восемь 16-разрядных ячеек);
- прием данных и размещение их в буфере приемника (восемь 16-разрядных ячеек).

Контроллер формирует сигналы прерываний по следующим событиям:
- необходимость обслуживания буферов приемника и/или передатчика;
- переполнение буфера приемника;
- наличие данных в буфере приемника по истечении времени таймаута.

Основные характеристики:
- программное управление скоростью обмена;
- программируемая длительность информационного кадра от 4 до 16 бит;
- независимое маскирование прерываний от буферов передатчика и приемника.

16.1 Структура контроллера SPI

Упрощенная функциональная схема контроллера SPI с блоком синхронизации показана на рисунке 16.1.

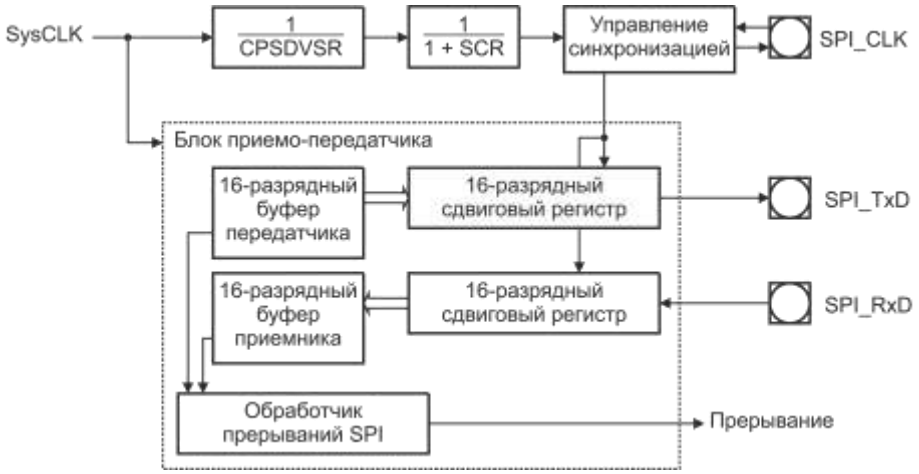


Рисунок 16.1 – Упрощенная функциональная схема контроллера SPI

Синхронизация

В режиме мастера на основе сигнала SPI_CLOCK посредством двух последовательно стоящих делителей формируется сигнал тактирования передачи и приема данных с частотой Fspi_clk, которую можно вычислить по формуле

$$Fspi\_clk = Fsysclk/(CPSDVSR\times(1+SCR)), \quad (16.1)$$

где – Fsysclk – частота входного синхросигнала;
CPSDVSR – коэффициент первого делителя частоты (задается в регистре SPI_SPCR);
SCR – коэффициент второго делителя частоты (задается в регистре SPI_SR0).

Сформированный синхросигнал подается на вывод SPI_CLK (сконфигурированный как выход) микроконтроллера и далее к подключенным внешним ведомым устройствам.

В режиме ведомого значения коэффициентов делителей не важны. Внешний синхросигнал подается на вывод SPI_CLK (сконфигурированный как вход) и тактирует прием и передачу данных.

Для корректной работы всегда должны соблюдаться условия:

- в режиме мастера для формируемого синхросигнала

$$Fspi\_clk \leq SysCLK/2; \quad (16.2)$$

- в режиме ведомого для входящего внешнего синхросигнала

$$Fspi\_clk \leq SysCLK/12. \quad (16.3)$$

Буферы приема и передачи

Для хранения передаваемых и принятых данных в контроллере SPI имеются два 16-разрядных буфера, организованных по типу FIFO. Каждый буфер может хранить до восьми слов данных. Буфер для передаваемых данных доступен только для записи, а буфер принятых данных – только для чтения.

Данные для передачи записываются в буфер через регистр SPI_DR. Допускается заранее заполнить буфер или записывать в него данные в течение работы контроллера. Состояние буфера можно контролировать с помощью битов TNF и TFE регистра SPI_SR. Если контроллер выключен (сброшен бит SSE регистра SPI_CR1), то запись в регистр SPI_DR приведет к тому, что данные будут размещены в буфере и будут переданы после включения контроллера. Если контроллер включен и выбран режим мастера, то в случае отсутствия данных в буфере запись в регистр SPI_DR приведет к немедленному началу передачи. Если запись данных в регистр SPI_DR происходит во время текущей передачи, то данные размещаются в буфере.

Полученные данные автоматически сохраняются в буфере принятых данных. Извлечь данные из буфера возможно чтением регистра SPI_DR. Состояние буфера можно контролировать с помощью битов RFFF и RNE регистра SPI_SR.

Размер передаваемого кадра данных может быть от 4 до 16 бит, что задается полем DSS регистра SPI_CR0. Если выбран размер кадра менее 16 бит, данные выравниваются по правой границе; неиспользуемые биты игнорируются.

Оба буфера могут генерировать четыре независимых маскируемых прерывания TXINTR (запрос на обслуживание буфера передатчика), RXINTR (запрос на обслуживание буфера приемника), RTINTR (таймаут ожидания чтения данных из буфера приемника), RORINTR (переполнение буфера приемника). Четыре линии прерываний объединены по ИЛИ. Появление любого из прерываний генерирует прерывание контроллера SPI, которое передается в контроллер прерываний NVIC.

Управление прерываниями и контроль их состояния осуществляется посредством регистров SPI_IMSC, SPI_RIS, SPI_MIS, SPI_ICR.

16.2 Функционирование

После сброса микроконтроллера работа приемопередатчика запрещена. Прежде чем разрешить работу битом SSE регистра SPI_CR1, следует сконфигурировать контроллер посредством регистров SPI_CR0 и SPI_CR1, а также, если это необходимо, запрограммировать маски прерываний.

Динамическое изменение конфигурации устройства не допускается.

Для протокола SPI дополнительно задаются полярность и фаза сигнала (биты SPH и SPO регистра SPI_CR0).

После разрешения работы приемопередатчик готов к обмену данными с внешними устройствами по линиям SPI_TxD (передача данных к внешнему устройству) и SPI_RxD (прием данных от внешнего устройства).

В зависимости от режима работы сигнал на линии SPI_FSS используется либо для кадровой синхронизации (интерфейс SSI, активное состояние – высокий уровень), либо для выбора устройства в режиме ведомого (интерфейсы SPI и Microwire, активное состояние – низкий уровень).

Во всех трех режимах SPI, Microwire и SSI синхросигнал SPI_CLK формируется только тогда, когда приемопередатчик готов к обмену данными. Перевод сигнала SPI_CLK в неактивное состояние используется как признак таймаута приемника, то есть наличия в буфере приемника необработанных данных по истечении заданного интервала времени.

Установка бита MS регистра SPI_CR1 включает режим ведомого устройства. В этом режиме разрешение или запрещение передачи данных через выход SPI_TxD контролируется битом SOD. На прием синхросигнала и данных состояние этого бита влияния не оказывает.

Интерфейс SPI

Реализует полнодуплексный режим передачи данных.

Включает одну линию синхронизации SPI_CLK, две линии приема и передачи данных SPI_RxD и SPI_TxD, а также линию выбора устройства (для режима ведомого) SPI_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPI_FSS должен подаваться низкий уровень сигнала в течение всей передачи кадра (последовательность передаваемых бит данных длиною от 4 до 16 бит).

Передача данных может быть одиночной (один кадр) или непрерывной (более одного кадра подряд). Данные передаются старшим битом вперед.

Особенностью интерфейса SPI является то, что в нем реализована возможность задания полярности и фазы тактового сигнала. Бит SPO регистра SPI_CR0 задает полярность тактового сигнала, т. е. определяет, какой уровень сигнала будет удерживаться на линии SPI_CLK в то время, когда линия не активна.

Бит SPH задает фазу тактового сигнала. Фактически он задает порядок считывания и выставления данных. По умолчанию бит SPH сброшен и выставление данных на линиях SPI_TxD и SPI_RxD происходит по переднему фронту сигнала синхронизации, а выборка – по заднему.

Передним всегда считается тот фронт сигнала, который является началом передачи первого бита (см. рисунок 16.2).

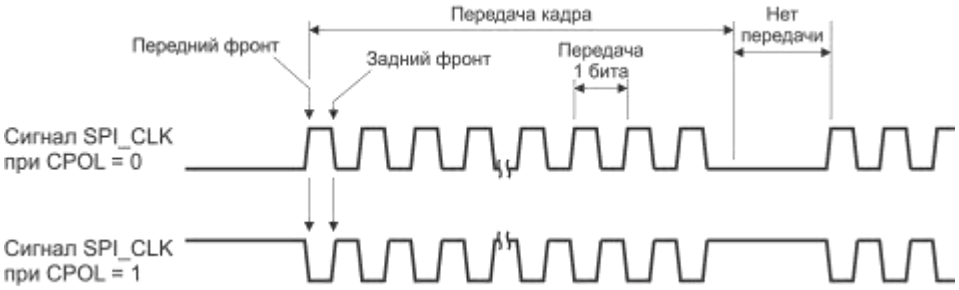


Рисунок 16.2 – Сигнал синхронизации SCK при разных состояниях бита CPOL

Комбинации битов SPO и SPH задают четыре режима обмена данными (см. рисунок 16.3).

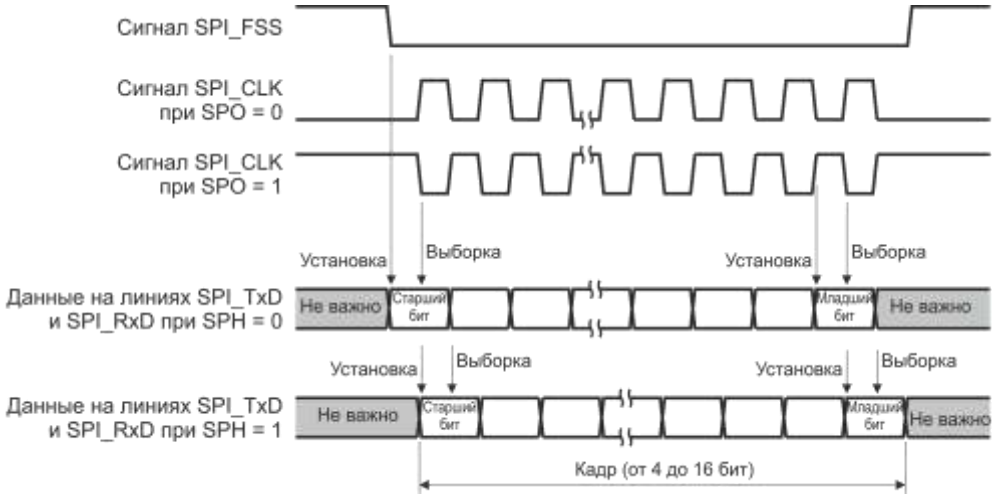


Рисунок 16.3 – Передача кадров данных в интерфейсе SPI

На рисунке 16.4 показано поведение сигналов при непрерывной передаче кадров данных.

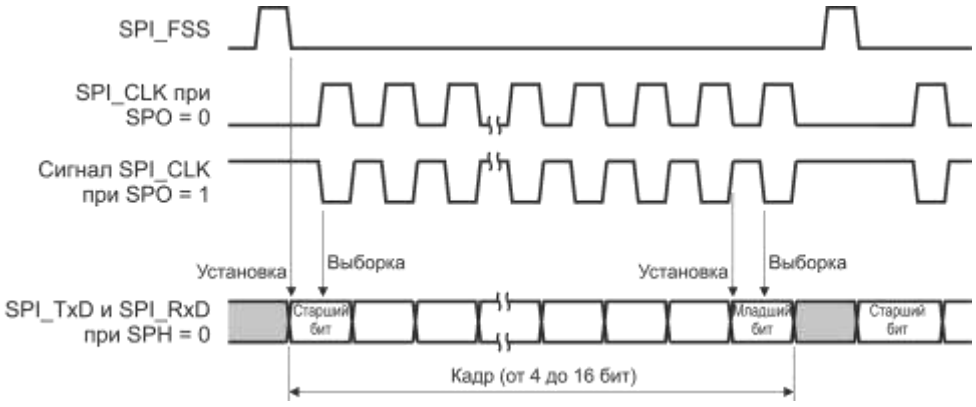


Рисунок 16.4 – Непрерывная передача кадров данных

В режиме непрерывной передачи данных при условии SPH = 0 на линии SPI_FSS должны формироваться импульсы между передачами кадров данных. Это связано с тем, что в этом режиме низкий уровень сигнала на линии SPI_FSS ведомого устройства блокирует запись в сдвиговый регистр. Поэтому мастер должен переводить линию SPI_FSS в высокий уровень по окончании передачи каждого кадра, разрешая, таким образом, запись новых

данных. По окончании приема последнего бита кадра линия SSP_FSS переводится в состояние логической единицы по истечении одного такта сигнала SPI_CLK.

В режиме непрерывной передачи данных при условии SPH = 1 низкий уровень сигнала на линии SPI_FSS не блокирует запись в сдвиговый регистр. Поэтому линия SPI_FSS может оставаться в состоянии нуля в течение передачи всех кадров и будет переведена в состояние логической единицы только по окончании передачи.

Интерфейс Microwire

Реализует полудуплексный режим передачи данных.

Включает линию синхронизации SPI_CLK, две линии приема и передачи данных SPI_RxD и SPI_TxD, а также линию выбора устройства (для режима ведомого) SPI_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPI_FSS должен подаваться низкий уровень сигнала в течение всей передачи кадра (последовательность передаваемых бит данных длиною от 4 до 16 бит).

Передача данных может быть одиночной (один кадр) или непрерывной (более одного кадра подряд). Данные передаются старшим битом вперед.

Перед началом передачи линия SPI_FSS переводится в низкое состояние.

Каждая передача начинается с передачи от мастера к ведомой 8-битной управляющей последовательности. В течение передачи этой последовательности приемник мастера не обрабатывает входящие данные. После того как управляющая последовательность передана и декодирована одним из ведомых устройств, этот ведомый выдерживает паузу в один такт синхросигнала и начинает передавать мастеру кадр данных (см. рисунок 16.5).

Выставление данных происходит по заднему фронту сигнала SPI_CLK, а считывание – по переднему.

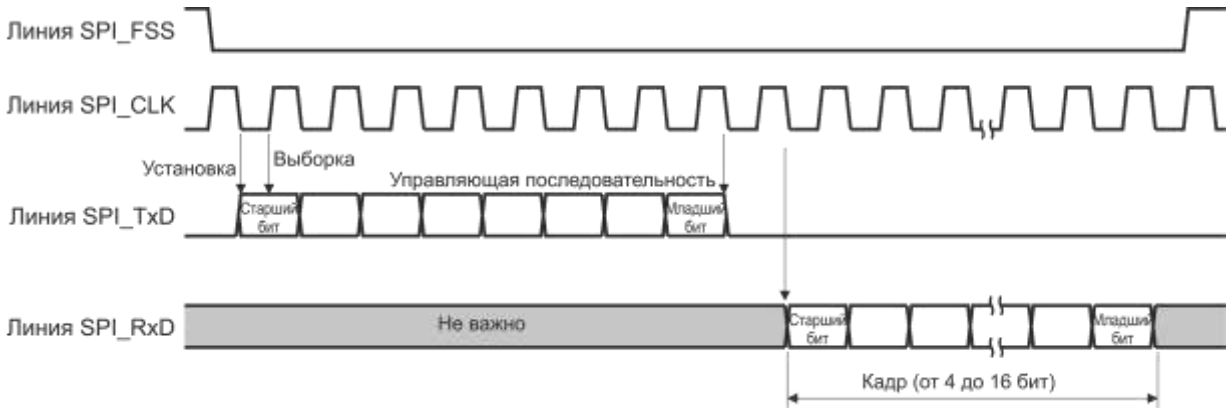


Рисунок 16.5 – Передача кадра данных в интерфейсе Microwire

По окончании приема данных линия SPI_FSS переводится в высокое состояние.

Примечание – В течение времени, когда передается управляющая последовательность, и между передачами линия SPI_RxD может находиться в третьем состоянии.

В режиме непрерывной передачи начало и завершение передачи нескольких кадров данных аналогично передаче одного кадра. Линия SPI_FSS удерживается в нуле в течение всего сеанса передачи. По окончании передачи одного кадра данных начинается передача управляющей последовательности без паузы (см. рисунок 16.6).

Примечание – Буферы FIFO приема и передачи данных не очищаются автоматически, даже в случае запрещения работы сбросом бита SSE.

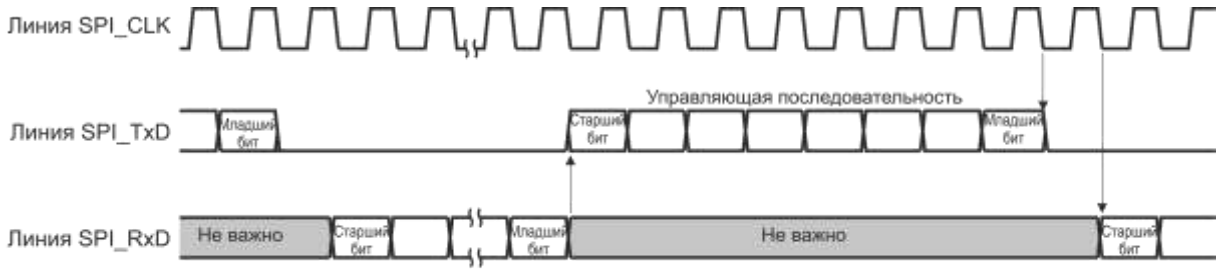


Рисунок 16.6 – Передача кадров данных в интерфейсе Microwire

Интерфейс SSI

Реализует полнодуплексный режим передачи данных.

Включает одну линию синхронизации SPI_CLK, две линии приема и передачи данных SPI_RxD и SPI_TxD, а также линию выбора устройства SPI_FSS.

Перед началом передачи каждого кадра на линии SPI_FSS формируется импульс длительностью в один период сигнала SPI_CLK. Далее, мастер и ведомый передают данные. Установка данных производится по переднему фронту синхросигнала, а выборка – по заднему (см. рисунок 16.7). Весь цикл передачи начинается сразу же после появления хотя бы одного элемента в буфере FIFO передатчика.

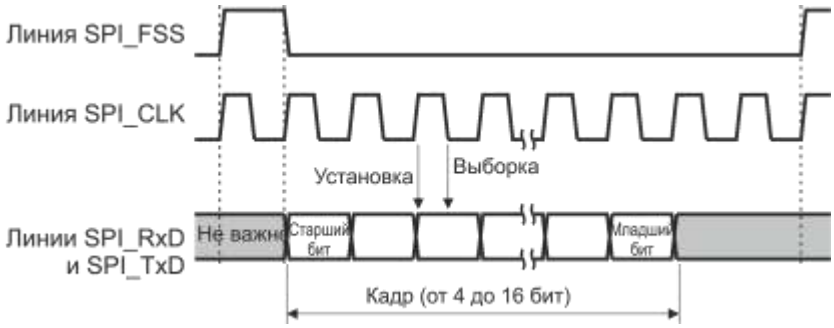


Рисунок 16.7 – Передача кадра данных в интерфейсе SSI

Режим непрерывной передачи кадров данных показан на рисунке 16.8.

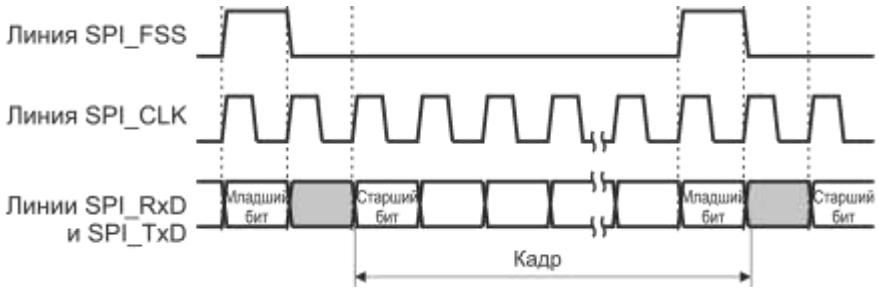


Рисунок 16.8 – Передача кадров данных в интерфейсе SSI

16.3 Прерывания

Четыре маскируемых источника прерываний объединены по ИЛИ, и в результате формируется один общий сигнал прерывания SPI.

Сигналы источников запросов на прерывания:

- TX – от приемника при заполнении его наполовину;
- RX – от передатчика при опустошении его наполовину;
- RT – по таймауту приемника (буфер приемника не пуст и не было попыток обращения к нему в течение времени, равного передаче 32 бит);
- ROR – по переполнению буфера приемника.

Каждый из сигналов может быть маскирован путем сброса соответствующего бита в регистре маски SPI_IMSC.

Для определения источника прерывания следует прочитать регистр SPI_RIS или регистр SPI_MIS (маскированные прерывания).

17 Контроллер интерфейса I2C

Модуль контроллера 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.

17.1 Протокол шины

Протокол I2C использует двухпроводной интерфейс для двусторонней связи между устройствами, подключенными к шине. Двунаправленная шина состоит из двух линий: данных SDA и тактового сигнала SCL. Эти линии подключены к источнику питания через подтягивающие резисторы. Шинные формирователи любых устройств, подключаемых к шине, выполняются по схеме с открытым коллектором или открытым стоком. Устройства могут выставить только низкий уровень на соответствующей линии. Следовательно, обе линии SDA и SCL реализуют функцию «монтажное И».

Протокол поддерживает режим мультимастер, в котором шина может контролироваться одним или несколькими устройствами из подключенных к шине. Каждое устройство, подключенное к шине, имеет свой адрес и может быть как приемником, так и передатчиком (некоторые – только приемниками).

Операции с данными

Устройство, которое начинает передачу данных, становится мастером. Мастер генерирует тактовый сигнал SCL, а также инициирует и завершает передачу данных по шине. За один такт сигнала SCL передается один бит данных по линии SDA (рисунок 17.1).

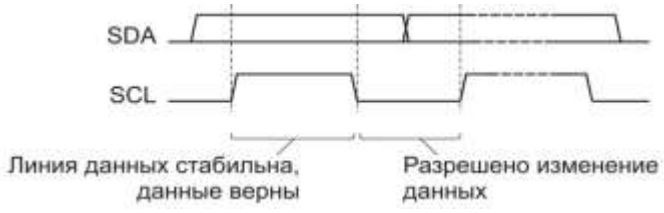


Рисунок 17.1 – Передача бита данных

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 (см. рисунок 17.2).

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 17.2).

Состояния старта и стопа формирует только мастер.

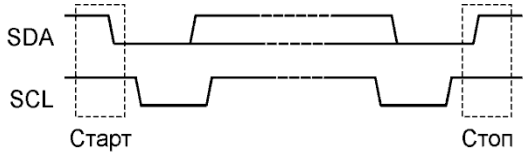


Рисунок 17.2 – Состояния старта и стопа

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 17.3).

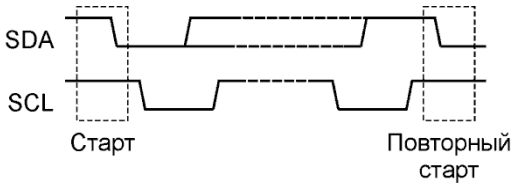


Рисунок 17.3 – Состояние повторного старта

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 17.4 приведен пример арбитража.

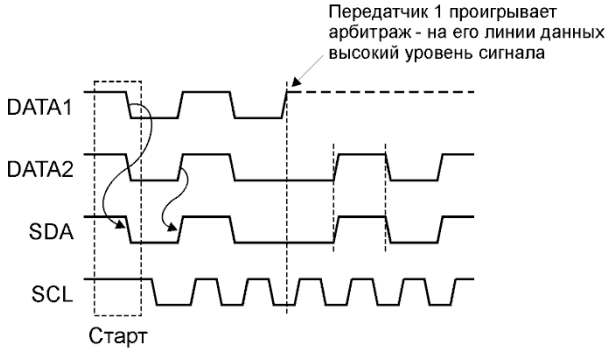


Рисунок 17.4 – Арбитраж на линии SDA

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 ST 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 17.5).

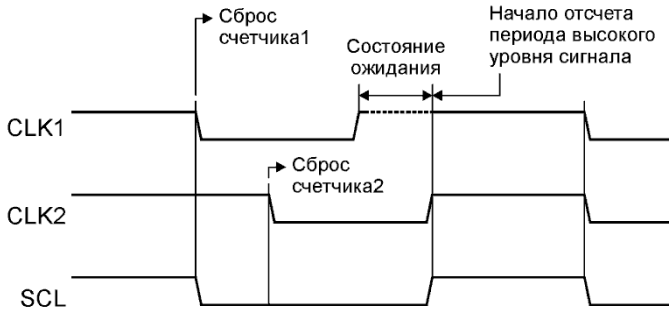


Рисунок 17.5 – Синхронизация

Линия SCL переводится в состояние «0» сразу, как только один из мастеров выставляет на своей линии тактового сигнала низкий уровень сигнала (CLK1 на рисунке 17.5). 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 17.5). 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 17.6).

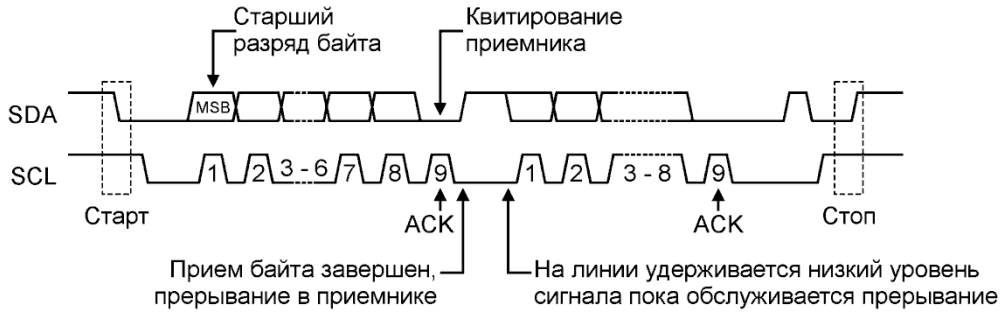


Рисунок 17.6 – Передача данных

Квитирование

Каждый байт посылки должен быть завершен квитированием, т. е. ответом на прием сигнала запроса подтверждения приема (ACK). На рисунках 17.6 и 17.7 показано положение момента квитирования в пределах посылки.

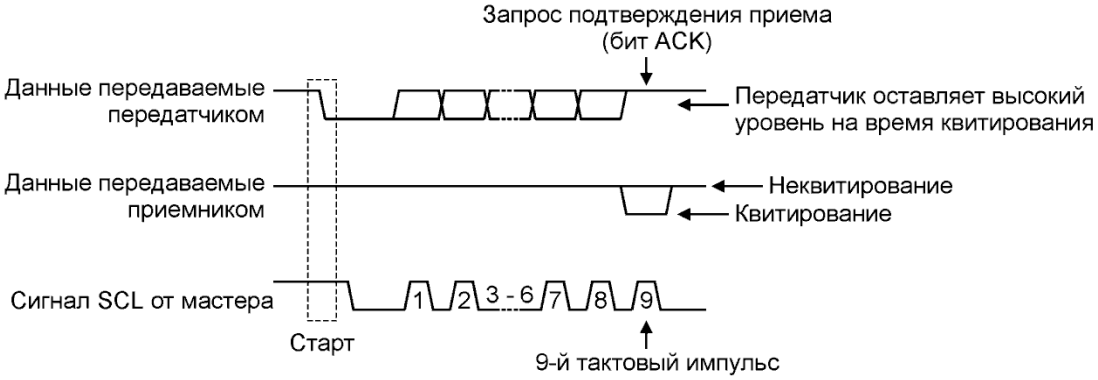


Рисунок 17.7 – Квитирование и неквитирование бита подтверждения ACK

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 17.7). 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 неквитировать 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 17.8 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»).

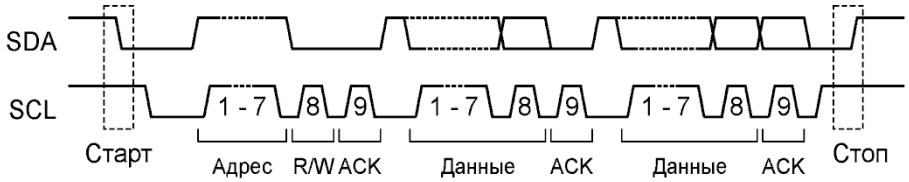


Рисунок 17.8 – Передача данных с 7-битной адресацией

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

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

Протокол SMBus/I2C поддерживает уникальную функцию – распознавание адреса отклика на сигнал предупреждения (Alert Response Address – ARA). В системах с несколькими ведомыми каждое устройство может послать мастеру сигнал предупреждения. Для этого используется дополнительная третья линия ALERT#, физически идентичная линиям SDA и SCL, реализованная по принципу монтажное «И». К этой линии также подключаются все устройства. Когда какому-то ведомому (или нескольким ведомым) необходимо обратиться к мастеру, он (или они) выставляет (выставляют) на линии ALERT# низкий уровень сигнала – это сигнал предупреждения (см. рисунок 17.9).

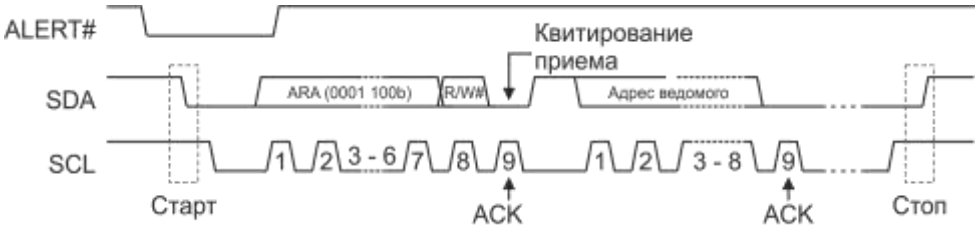


Рисунок 17.9 – Передача адреса отклика на сигнал предупреждения

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

стандартным правилам арбитража. Если после обслуживания ведомого мастер все еще обнаруживает на линии ALERT# низкий уровень сигнала, он понимает это как то, что сигнал предупреждения посылался несколькими ведомыми. Мастер снова отправляет ARA и затем общается со следующим ведомым. Появление на линии ALERT# высокого уровня сигнала означает, что все ведомые, которые требовали обращения, обслужены.

Примечание – Контроллер I2C, входящий в состав микросхем 1874BE10T и 1874BE10AT, не имеет выделенной линии ALERT#. При необходимости пользователем может быть задействован свободный вывод микроконтроллера и программно реализована передача сигнала предупреждения от ведомого к мастеру. Функция распознавания адреса отклика (ARA) и последующей отправки собственного адреса реализована полностью. Включить функцию можно установкой бита SMBARE в регистре CTL0.

Формат передачи данных с 10-битной адресацией

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

Основной идеей формата является передача 10-битного адреса в двух первых байтах, следующих сразу после старта. В первом байте передается значение 1111_0xxb, где «xx» – это два старших бита адреса и бит R/W# (на рисунке 17.10 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 17.10).

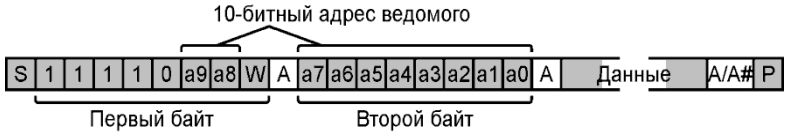


Рисунок 17.10 – Передача данных ведомому с 10-битным адресом (для расшифровки обозначений, применяемых на рисунке, следует обратиться к таблице 17.1)

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 17.11).

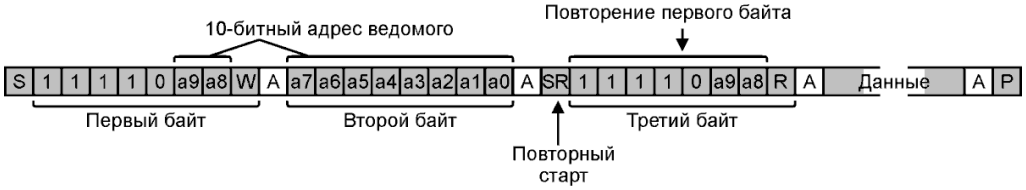


Рисунок 17.11 – Получение данных от ведомого с 10-битным адресом

На рисунках 17.10 и 17.11 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 17.1 с подробными пояснениями.

Таблица 17.1 – 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 ST
Цветное поле	Серым цветом обозначены биты, передаваемые от мастера к ведомому
Белое поле	Белым цветом обозначены биты, передаваемые от ведомого к мастеру

17.2 Функциональное описание

Структурная схема модуля I2C представлена на рисунке 17.12. Далее приводится краткое описание назначения блоков модуля.

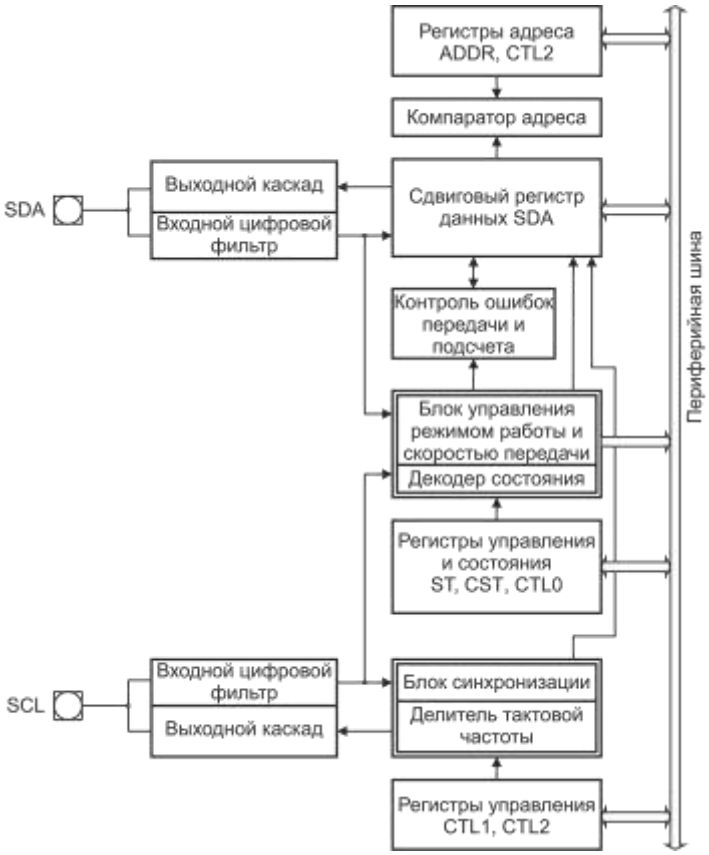


Рисунок 17.12 – Структурная схема модуля I2C

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:

- ST – содержит биты, отражающие текущую конфигурацию модуля I2C (мастер или ведомый, передатчик или приемник) и бит флага прерывания;
- CST – является одновременно регистром управления шиной и регистром состояния шины;
- CTL0 – управляет генерированием состояний старта, повторного старта и останова, а также квитированием;
- CTL1 и CTL2 – устанавливают параметры тактового сигнала в режиме мастера и контролируют режим 10-битной адресации.

Регистры адреса и компаратор адреса

В регистр адреса ADDR 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 CTL0), то компаратор сравнивает принятый адрес со значением 0000_000b. Если разрешено распознавание адреса отклика на сигнал предупреждения (установлен бит SMBARE регистра CTL0), то компаратор сравнивает принятый адрес со значением 0001_100b.

Если включен режим 10-битной адресации (одновременно установлены биты SAEN и S10EN регистров ADDR и CTL2, 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 CTL2. Старший бит второго полученного байта сравнивается со значением нулевого бита поля S10ADR, а оставшиеся семь битов – со значением битового поля ADDR регистра ADDR (см. рисунок 17.13).

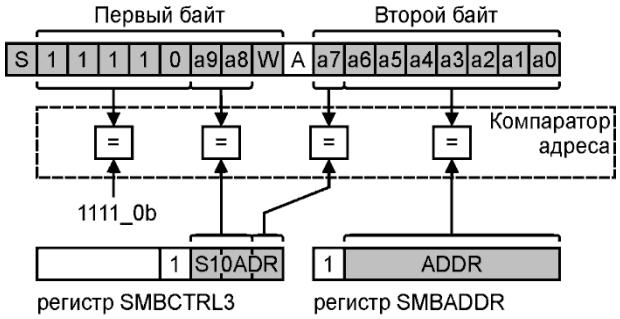


Рисунок 17.13 – Компаратор адреса в режиме 10-битной адресации

Сдвиговый регистр данных

Регистр SDA представляет собой сдвиговый регистр, используемый для приема и передачи данных. Старший бит регистра передается/принимается первым, младший бит – последним. Запись в регистр SDA возможна только, если установлен бит INT регистра ST. Регистр может быть прочитан в любой момент времени, но прочитанные данные будут гарантированно достоверными только при установленном бите INT. Регистр SDA 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 поля SCLFRQ регистра CTL1, 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 CTL2, 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 17.14.

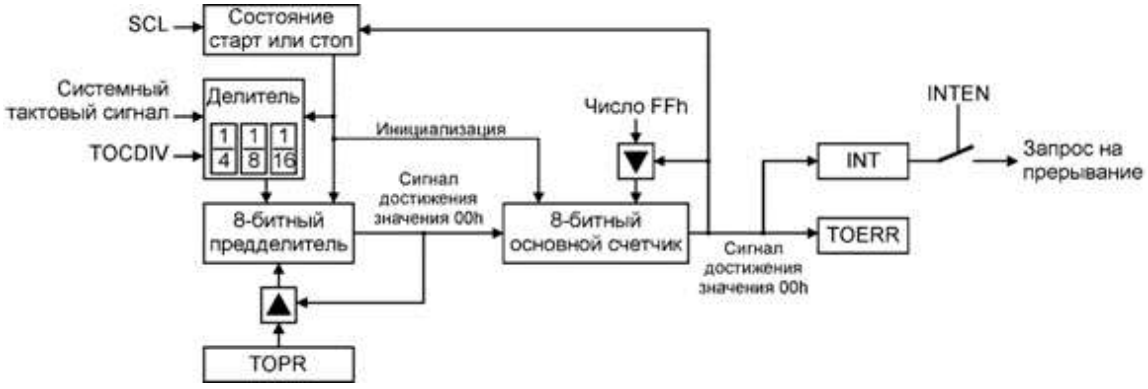


Рисунок 17.14 – Функциональная схема счетчика времени ожидания

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 TOPR. После достижения нуля счетчиком предделителя, он загружается значением из регистра TOPR. 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 CST. Дополнительно устанавливается флаг INT и если разрешено, генерируется прерывание.

Период времени ожидания определяется следующим выражением

$$Тожид = Tosc \times TOCDIV \times (SMBTOPR + 1) \times 256 , \qquad (17.1)$$

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 ST 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 CTL1);
- в течение времени простоя проверить, не подключен ли другой активный мастер к шине (бит BB регистра CST 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 I2CEN регистра APB_CLK. Переход в режим Idle подобен программному выключению модуля I2C (очистка бита ENABLE в регистре CTL1). Регистры CTL0, ST и CST очищаются, чтобы гарантировать нормальный старт после возобновления функционирования модуля.

Выход из режима Idle осуществляется записью единицы в бит I2CEN и включением модуля битом ENABLE.

17.3 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 23,6 до 750,3 кГц (при XTAL1 = 24 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 0,25 до 2 МГц (при XTAL1 = 24 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, поддерживающие 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 17.15. Расшифровка обозначений, принятых на рисунке, приведена в таблице 17.1).

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

Рисунок 17.15 – Переход в режим HS и обратно в режим FS

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 CTL1.

2 Если активен режим мастера, записать нужный коэффициент деления в битовое поле SCLFRQ в регистре CTL1 для выбора периода тактового сигнала SCL (для режима HS записать коэффициент деления в поле HSDIV регистра CTL2).

3 Если активен режим ведомого, необходимо:

- записать «собственный» адрес ведомого в битовое поле ADDR и установить бит SAEN регистра ADDR;
- для реализации 10-битной адресации записать старшие биты адреса в битовое поле S10AD и установить бит S10EN регистра CTL2;
- для включения функции распознавания адреса общего вызова установить бит GCMEN в регистре CTL0;
- для включения функции распознавания адреса отклика установить бит SMBARE в регистре CTL0.

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

5 Для разрешения формирования запроса на прерывание установить бит INTEN в регистре CTL0.

Функционирование

Модуль I2C может работать в режиме мастера или ведомого. Также он может функционировать как передатчик или приемник. В итоге 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 ST в битовое поле MODE и может быть прочитан программно. В таблицах 17.2 и 17.3 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 ST будет записан код 04h, соответствующий состоянию с мнемоническим обозначением «MTADPA». Более подробно каждый режим работы модуля I2C будет рассмотрен далее. На рисунках 17.16 – 17.23, поясняющих работу модуля в том или ином режиме, приняты обозначения, расшифровка которых приводится в таблице 17.1. Для получения дополнительной информации и понимания работы модуля I2C можно воспользоваться приложением В.

Таблица 17.2 – Коды функционирования модуля I2C в режиме FS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	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
–		0Dh – 0Fh		Зарезервировано. Не использовать!	–
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

Окончание таблицы 17.2

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	ACK/ NACK
Ведомый в режиме FS	–	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 В.					

Таблица 17.3 – Коды функционирования модуля I2C в режиме HS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	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, 35h зарезервированы и не доступны для использования. Дополнительная информация находится в приложении В.					

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 CTL0. Если бит BB в регистре CST 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 CTL0), сгенерируется прерывание.

Передача адреса и данных:

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

2 После записи в регистр SDA флаг INT сбрасывается программно установкой бита CLRST в регистре CTL0.

3 После сброса флага INT и по истечении времени, требуемого для установки данных, на линии SCL появляется тактовый сигнал и данные, хранящиеся в регистре SDA, 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 SDA 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 SDA 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 CST вызовет перенос содержимого регистра ошибок (не доступен программно) в регистр SDA 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 передатчика 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 SDA.

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 CTL0.

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

3 Сбросить флаг INT.

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

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 В.

На рисунке 17.16 представлено графическое пояснение к описанию режима.

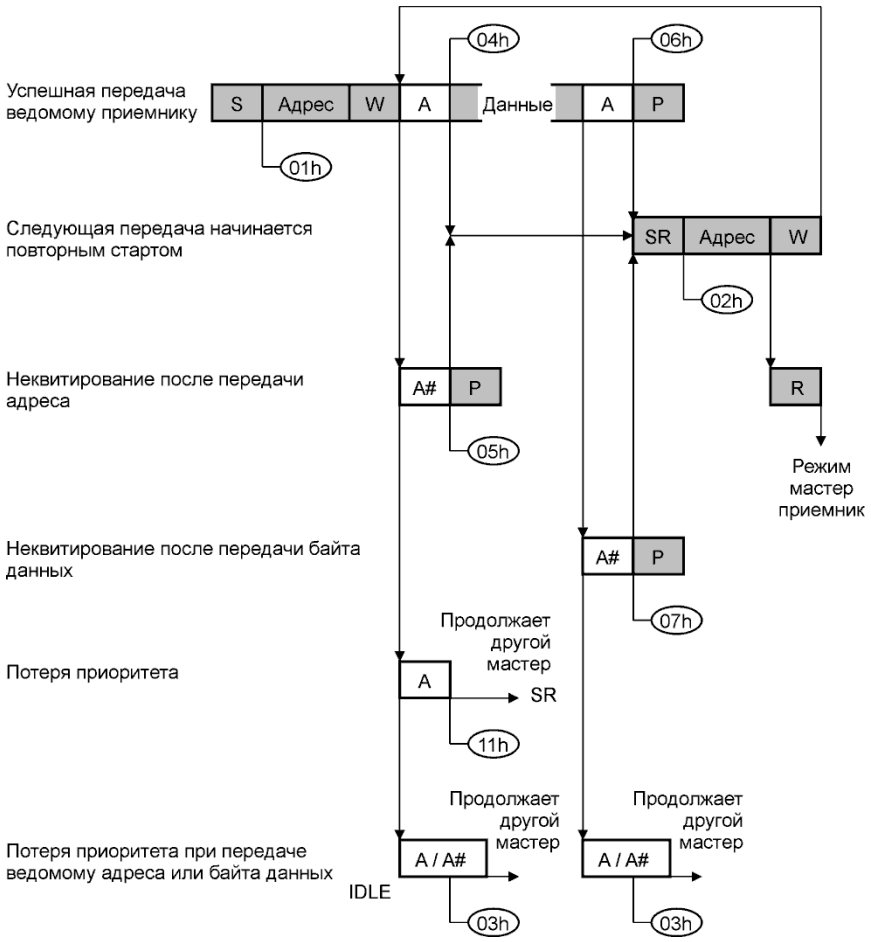


Рисунок 17.16 – Режим FS мастера передатчика

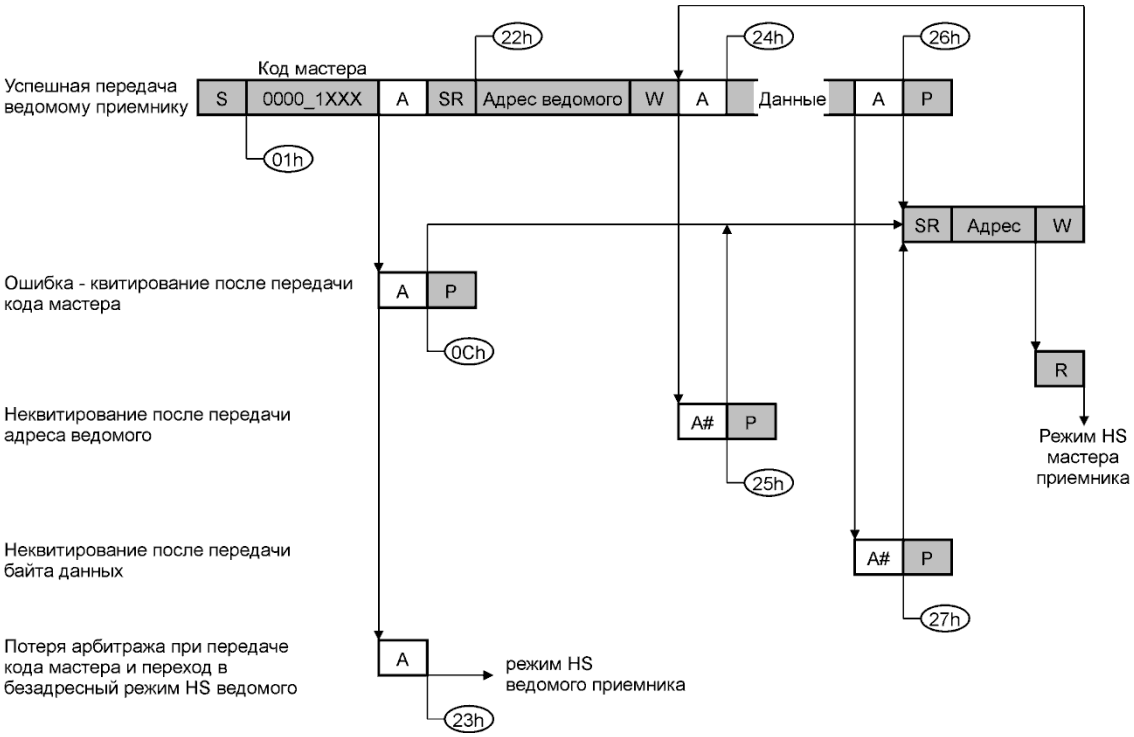


Рисунок 17.17 – Режим HS мастера передатчика

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 единицы в бит CLRST.

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

Дополнительно можно обратиться к приложению В.

На рисунке 17.17 представлено графическое пояснение к описанию режима.

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 SDA. 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 CTL0. В тоже время, если требуется отправка байта CRC, следует установить бит PECNEXT в регистре CST. 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 CST.

Дополнительно можно обратиться к приложению В.

На рисунке 17.18 представлено графическое пояснение к описанию режима.

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 В.

На рисунке 17.19 представлено графическое пояснение к описанию режима.

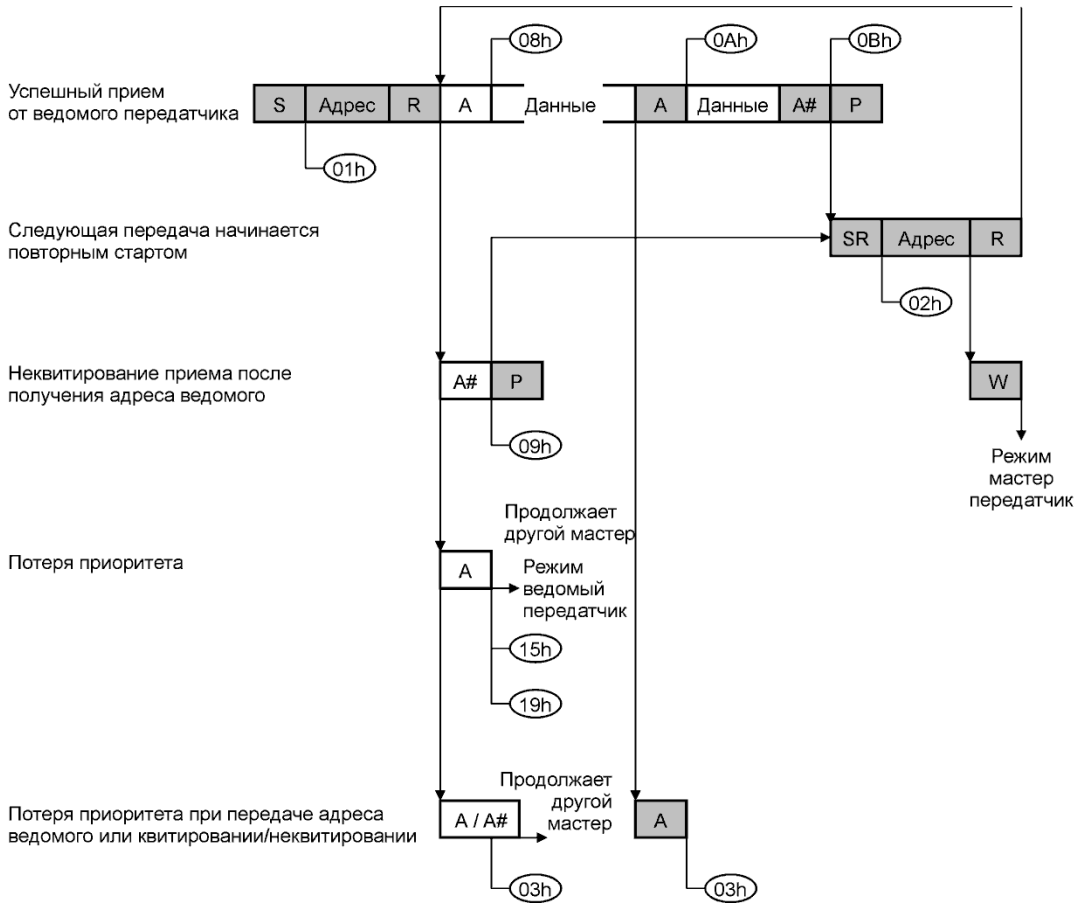


Рисунок 17.18 – Режим FS мастера приемника

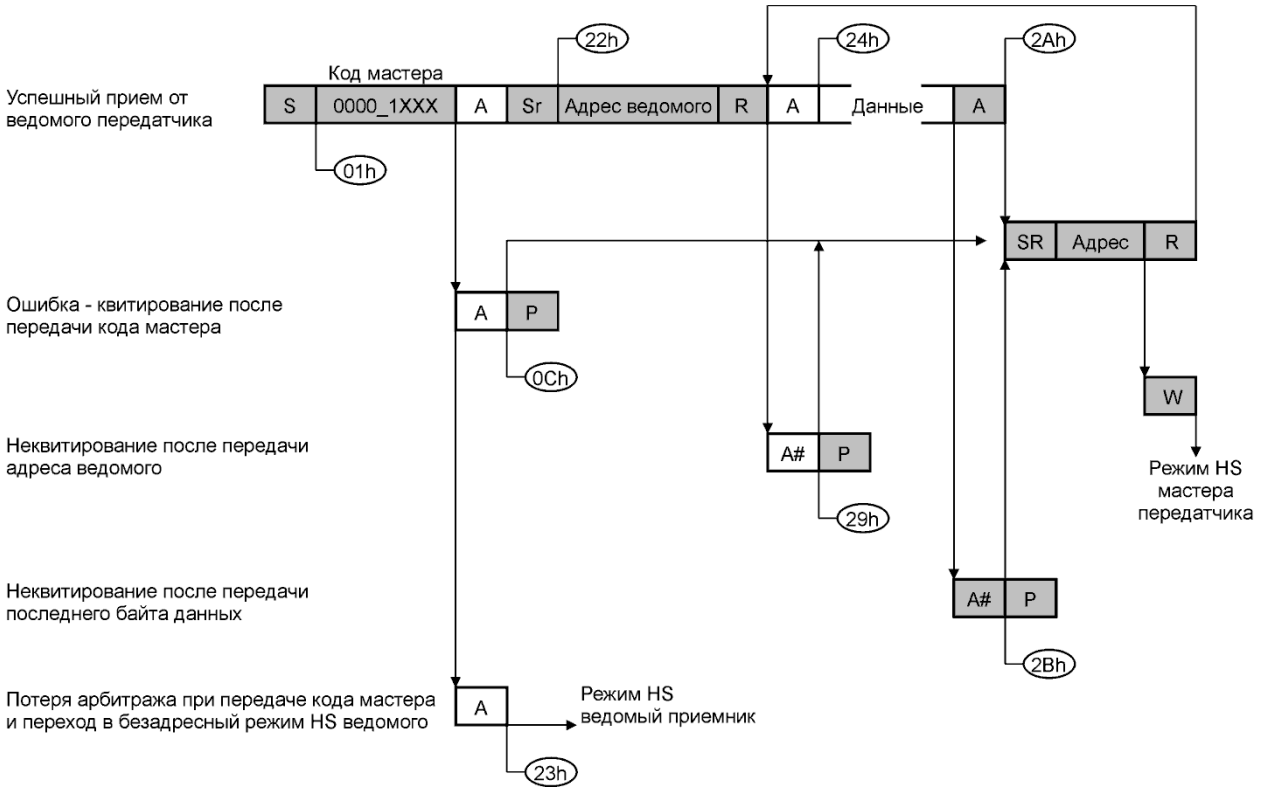


Рисунок 17.19 – Режим HS мастера приемника

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 передатчика может переключиться в режим ведомого приемника вследствие потери арбитража при передаче адреса.

После получения байта адреса ведомый сравнивает полученный адрес:

- полем ADDR регистра ADDR, 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 SDA.

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 SDA, а линия SCL удерживается в «0». После программного чтения регистра SDA 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.

Последовательность передачи бит в посылке при 10-битной адресации была рассмотрена ранее в подразделе 17.1 настоящего ТО.

Механизм распознавания адреса изложен в подразделе 17.2 настоящего ТО и показан на рисунке 17.13.

После корректного приема ведомым двух байтов и совпадении принятого адреса с собственным байты сохраняются в регистре SDA и сдвиговом регистре, прием квитируется, устанавливается флаг INT, а в поле MODE записывается соответствующий код состояния – 10h или 17h.

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 SDA и 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 (отправлена «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 SDA и сбросить флаг INT, после чего модуль I2C освободит шину.

Дополнительно можно обратиться к приложению В.

На рисунке 17.20 представлено графическое пояснение к описанию режима.

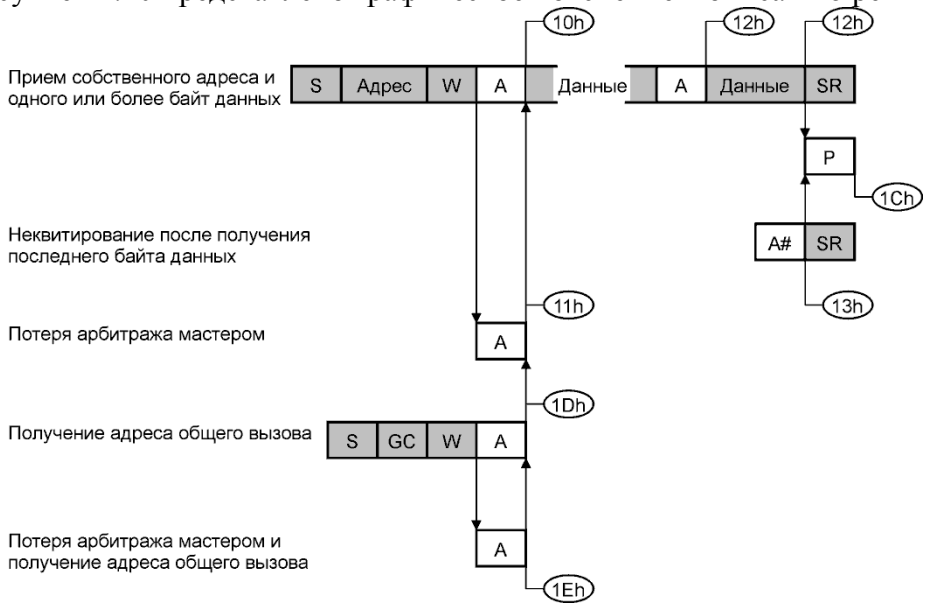


Рисунок 17.20 – Режим FS ведомого приемника

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 В.

На рисунке 17.21 представлено графическое пояснение к описанию режима.

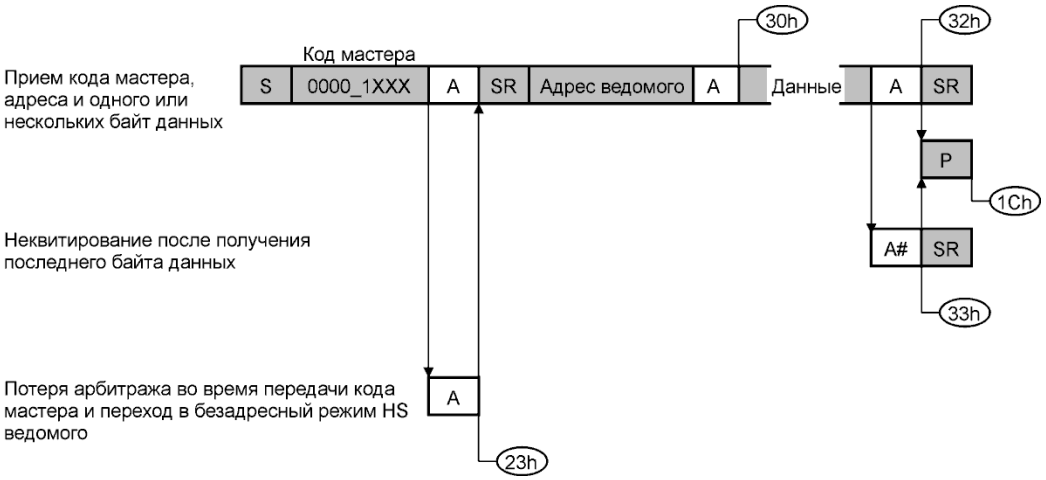


Рисунок 17.21 – Режим HS ведомого приемника

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 SDA следует записать данные.

Пока установлен флаг INT, линия SCL удерживается в «0». После записи данных в регистр SDA 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 SDA. Каждый последующий байт должен записываться в регистр SDA 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 приемника. Мастер приемника 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 SDA) для того, чтобы в регистр SDA 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 (подробнее – см. «Формат передачи данных с 7-битной адресацией» в подразделе 17.1).

Для включения функции распознавания адреса отклика следует установить бит SMBARE в регистре CTL0.

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 В.

На рисунке 17.22 представлено графическое пояснение к описанию режима.

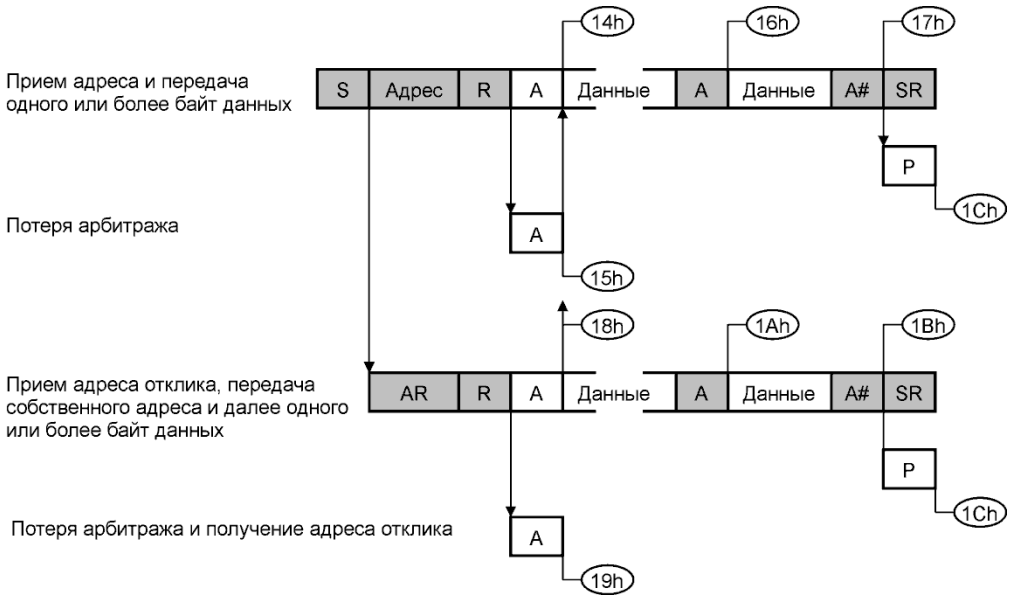


Рисунок 17.22 – Режим FS ведомого передатчика

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 В.

На рисунке 17.23 представлено графическое пояснение к описанию режима.

Рисунок 17.23 – Режим HS ведомого передатчика

Дополнительная информация о работе модуля I2C

1 Когда модуль I2C выключен, бит BB регистра CST очищен. Включение модуля в системе с более чем одним мастером может произойти в момент времени, когда по шине идет передача. Бит BB не сможет этого 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 CST. Если линия активна, отправить одиночный импульс по линии SCL, установив бит TGSCL в регистре CST;
- 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.

18 Контроллер интерфейса ARINC 429

Контроллер интерфейса ARINC 429 представляет собой блок, позволяющий осуществлять последовательный полудуплексный обмен данными между устройствами согласно стандарта ARINC 429, который является международным общепринятым стандартом для гражданской авиации и утверждает единые принципы создания сети передачи данных между авиационными системами. Отечественный стандарт на этот интерфейс – ГОСТ 18977-79.

Контроллер ARINC 429 имеет восемь идентичных внутренних блоков, каждый из которых может быть независимо сконфигурирован как передатчик или как приемник. Общая структура одного блока показана на рисунке 18.1.

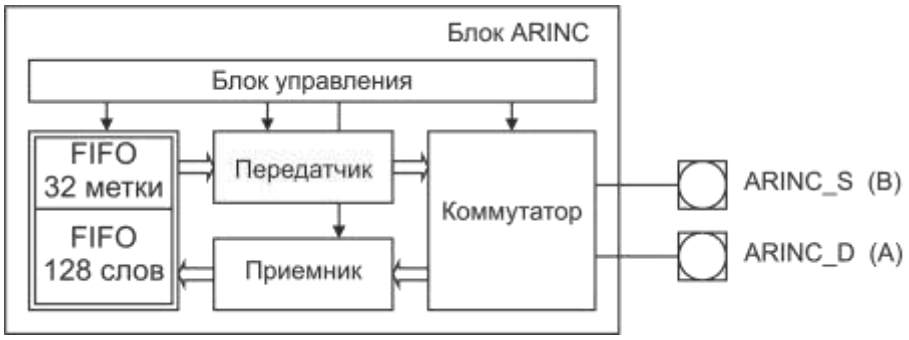


Рисунок 18.1 – Общая структура блока

Каждый блок имеет ОЗУ типа FIFO для хранения 128 слов данных и две линии ввода-вывода, которые соединены с выводами микроконтроллера. Назначение линий зависит от режима работы блока ARINC. Доступны два режима работы: стробирования (режимы DS) и преобразования сигнального уровня в логический (режим AB), см. таблицу 18.2.

Блок обладает возможностью гибкой настройки функционирования. Так в режиме передатчика может быть включен механизм вычисления четности и автоматическая вставка интервала между передачами слов, а в режиме приемника имеется возможность включения и отключения механизмов фильтрации слов и контроля по паритету.

18.1 Краткое описание стандарта

Стандарт ARINC 429 определяет требуемые формат данных и аппаратную часть для передачи информации по каналу связи. Аппаратная часть состоит из одного передатчика (или источника), соединенного с приемниками (от одного до двадцати) по одной витой паре. Данные могут быть переданы только в одну сторону. Для двусторонней передачи данных требуется два канала (или линии) связи. Устройства в большинстве случаев объединяются в топологию типа «Звезда» или «Шина». Каждое оконечное устройство может содержать более чем один передатчик и приемник, соединенные по разным шинам связи. Самое простое соединение – это «Точка-точка», обеспечивающее высокую надежность передачи данных. Разновидности соединений устройств, для передачи данных по протоколу ARINC 429, показаны на рисунке 18.1.

Передатчик может взаимодействовать с приемниками по одной витой паре. Каждый приемник мониторит линию передачи данных и в случае обнаружения собственных принимает их (без подтверждения). Приемники не имеют явных адресов.

Одна передача данных состоит из 32-битного слова, в котором 24 бит данных с актуальной информацией, и 8 бит метки, используемой для определения типа передаваемых данных. Каждое передаваемое слово состоит из 5 полей, см. таблицу 18.1.

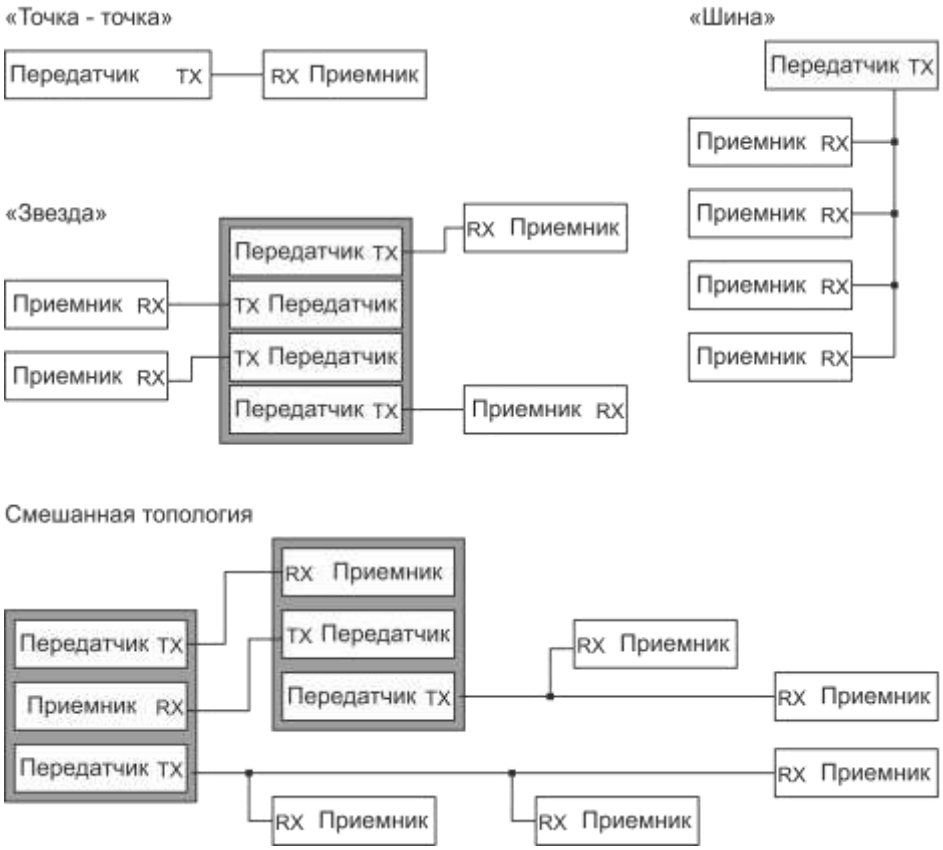
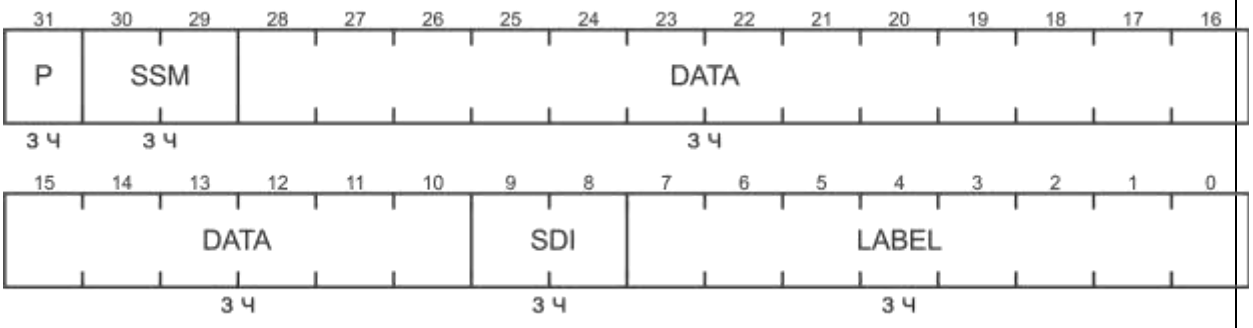


Рисунок 18.2 – Разновидности соединений устройств

Таблица 18.1 – Формат передаваемого слова



Поле	Бит	Описание	
P	31	Бит четности	
		0	Отключена проверка четности переданного слова
		1	Включена проверка четности переданного слова
SSM	30, 29	Поле знака/статуса	
DATA	28-10	Поле данных	
SDI	9, 8	Поле идентификации источник/приемник	
LABEL	7-0	Поле метки	

Слова передаются младшим битом вперед, но передача бит происходит в определенном порядке. Сначала передается метка (LABEL) как есть, а затем, начиная с восьмого бита, все остальное слово. Бит четности P передается последним, см. рисунок 18.3.

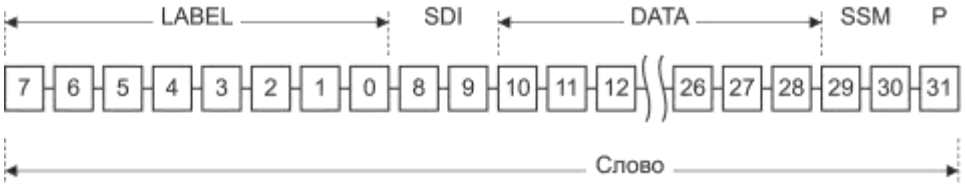


Рисунок 18.3 – Порядок передачи бит слова

Передаваемые 32-битные слова разделяются временными интервалами, в которых передача бит отсутствует. Длительность одного такого интервала должна быть не менее длительности передачи четырех бит. Использование данных интервалов исключает необходимость наличия отдельного синхросигнала. Частота передачи данных может быть задана в диапазоне от 12,5 кГц до 100 кГц.

18.2 Память данных блока

FIFO данных

Память данных блока ARINC представляет собой ОЗУ типа FIFO. В памяти может храниться до 128 слов. Загрузка данных в FIFO осуществляется посредством записи в регистр DATAWRITE. Выгрузка данных из FIFO осуществляется посредством чтения регистра DATAREAD. Для контроля состояния FIFO (пустое, заполнено наполовину, заполнено полностью) в регистре STAT есть биты-индикаторы.

Примечание – В режиме приемника FIFO доступно только для записи, а в режиме передатчика – только для чтения.

Полностью очистить FIFO данных можно записью единицы в бит FLUSHFIFO регистра CON.

FIFO меток и фильтрация

Помимо FIFO данных есть память меток сообщений, называемая FIFO меток объемом 32 байта. FIFO меток используется для фильтрации данных в режиме приемника по значению поля LABEL. Если значение метки в слове не совпало со значениями в FIFO меток, то слово будет проигнорировано, иначе слово будет размещено в FIFO данных. Включение фильтрации по меткам осуществляется битом LABCMPEN регистра CON. Контроль заполненности FIFO меток осуществляется посредством флага FIFOLF регистра STAT.

Загрузка значений в FIFO меток осуществляется посредством записи в поле LABVALWR регистра RXLABELFIFO. Одновременно с меткой указывается номер блока (от нуля до семи), к которому относится метка (поле FIFOMODULE).

Примечание – Запись в FIFO меток доступна только в режиме приемника.

Полностью очистить FIFO меток можно записью единицы в бит FLUSLAB регистра CON.

Фильтрация по полю SDI

Дополнительно можно фильтровать принимаемые данные по значению поля SDI. Значение этого поля задается битами SDIBIT9 и SDIBIT10 регистра CON. Включение фильтрации осуществляется битом SDICMPEN. При совпадении значения SDI в принимаемом слове с заданными в регистре CON, принимаемое слово будет записано в FIFO данных, иначе слово будет проигнорировано.

При включении фильтрации одновременно как по LABEL, так и по SDI, принимаемое слово запишется в FIFO данных при совпадении обоих условий.

18.3 Прерывания блока

В регистре состояния STAT имеются биты-флаги, которые устанавливаются при возникновении того или иного события, в случае если это разрешено соответствующими битами регистра CON.

В режиме приемника отслеживаются шесть событий:

- FIFO стало не пустым (флаг FIFONE);
- FIFO заполнилось наполовину (флаг FIFOHF);
- FIFO заполнилось полностью (флаг FIFOFULL);
- FIFO меток заполнилось полностью (флаг FIFOLF);
- ошибка четности (флаг PARITYERR);
- возникновение ошибки передачи, такой как сбой синхронизации или уменьшение длительности временного интервала между передачей слов (флаг RCVERR).

В режиме передатчика отслеживаются три события:

- была осуществлена попытка записи в заполненное FIFO (флаг FIFOFULL), т. е. возникла ситуация, когда заполнение FIFO данными происходит быстрее, чем блок их передает;
- FIFO опустошилось наполовину (флаг FIFOHF);
- FIFO опустошилось полностью (флаг FIFOEMPTY).

При возникновении любого из указанных событий, помимо соответствующего флага, также устанавливается бит INTSTAT, и генерируется прерывание.

В программе обработчике прерывания для определения источника прерывания следует прочитывать состояние регистра STAT. Биты-флаги событий не сбрасываются автоматически. Для сброса следует записать единицу в соответствующий бит.

Состояние бита INSTAT является логическим ИЛИ состояний битов-флагов. Поэтому пока будет оставаться установленным хотя бы один флаг, будет генерироваться прерывание. Как только все флаги будут сброшены, бит INTSTAT обнулится автоматически.

18.4 Работа с блоком ARINC

Перед началом работы следует разрешить тактирование всего контроллера ARINC 429 установкой бита ARINC в регистре CLKEN, сконфигурировать порты микроконтроллера (см. главу 10 «Порты»), и включить соответствующие альтернативные функции.

После этого для настройки каждого выбранного блока следует выполнить следующие действия:

- задать коэффициент деления;
- включить блок;
- задать режим работы – приемник или передатчик;
- разрешить необходимые прерывания;
- разрешить, если необходимо, использование бита четности;
- задать формат передачи данных.

Примечание – Для управления блоком используется регистр CON.

Расчет коэффициента деления

Коэффициент деления K задается полем PRESCALER регистра. Для вычисления его значения следует воспользоваться формулой:

$$K = \frac{Focs \times 100000}{baudrate \times (8 \times [DIV8])}, \qquad (18.1)$$

где Fosc – тактовая частота работы микроконтроллера в МГц;

baudrate – скорость передач данных в Гц;

DIV8 – значение бита DIV8 регистра CON.

Пример.

Требуемая стандартная скорость передачи данных (baudrate) – 100 кГц. В этом случае бит DIV8 сброшен, и поэтому коэффициент 8 не используется.

Тактовая частота работы микроконтроллера – 50 МГц.

Используя формулу 18.1, получим:

$$K = \frac{50 \times 100000}{100000 \times (8 \times [0])} = 50.$$

В поле PRESCALER следует записать значение 0101000b.

Для получения частоты передачи 12,5 кГц следует установить бит DIV8. Теперь при расчете коэффициент 8 будет использоваться, а коэффициент K останется тем же.

Включение блока

Для включения блока следует установить бит EN.

Выбор режима работы

По умолчанию блок находится в режиме приемника (бит TX сброшен). Для включения режима передатчика следует установить бит TX.

Разрешение прерываний

В режиме приемника для разрешения прерываний следует установить бит INTEN и соответствующие ожидаемым событиям биты с 31 по 27.

В режиме передатчика для разрешения прерываний следует установить бит INTEN и соответствующие ожидаемым событиям биты с 31 по 29.

Бит четности

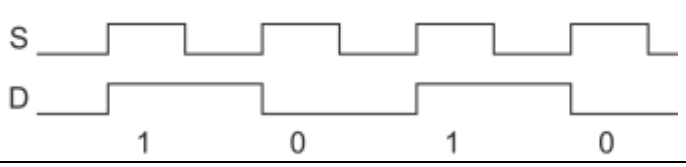
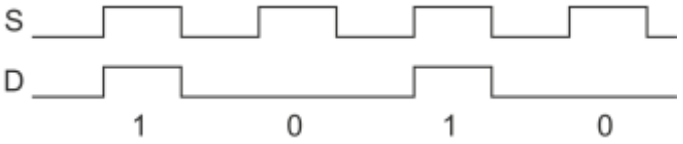
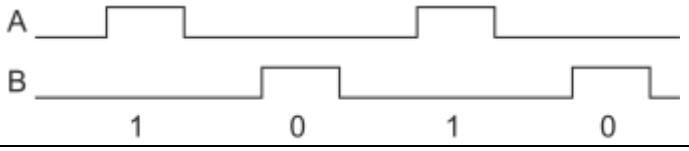
Для включения проверки на четность полученных данных следует установить бит PARITYEN. Тогда блок будет автоматически вычислять значение бита четности и сверять его с полученным. Для включения дополнения бита четности до нечетного следует установить бит PARITY.

При проверке, если принятый бит четности окажется неверным, данные не будут помещены в FIFO, а в регистре STAT будет установлен флаг PARITYERR.

Формат передачи данных

Как в режиме приемника, так и в режиме передатчика блок поддерживает три формата потока данных, см. таблицу 18.2.

Таблица 18.2 – Формат передаваемого слова

Биты регистра CON		Формат передачи данных	Примечание
ASYNCRX	AB/DS		
0	0	DS 	Для передатчика и приемника
1	0	DS 	Только для приемника
Не важно	1	AB 	Для передатчика и приемника

Первый и второй форматы – стробирование (DS). Формат задан по умолчанию (бит AB/DS = 0).

В режиме приемника микроконтроллер принимает стробирующие импульсы по линии ARINC_S, а данные – по линии ARINC_D. В зависимости от того, как долго бит данных удерживается на линии (от фронта до фронта строба или в течение строба), следует задать состояние бита ASYNCRX.

В режиме передатчика микроконтроллер формирует стробирующие импульсы по линии ARINC_S, а данные по линии ARINC_D. Бит ASYNCRX задает момент выдачи бита данных на линию – по переднему фронту строба или по заднему.

Третий формат – преобразования сигнального уровня в логический (AB). Формат задается установкой бита AB/DS как для передатчика, так и для приемника.

Строб на линии A (вывод ARINC_D) указывает на то, что передаваемый бит является единицей, а строб на линии B (вывод ARINC_S) – на то, что передаваемый бит является нулем.

19 Контроллер интерфейса МПИ (по ГОСТ Р 52070-2003)

Модуль представляет собой устройство, поддерживающее обмен данными с другими устройствами (контроллерами) посредством магистрального последовательного интерфейса (МПИ) в соответствии с ГОСТ Р 52070–2003 (аналогом является стандарт MIL-STD-1553B).

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.

Каждый модуль МПИ имеет два канала приема-передачи: основной – с выводами Mn_TxD0, Mn_TxDN0, Mn_RxD0, Mn_RxDN0, Mn_BLOCK0, и резервный – с выводами Mn_TxD1, Mn_TxDN1, Mn_RxD1, Mn_RxDN1, Mn_BLOCK1, а также вход внешнего опорного сигнала Mn_EXTCLK (где n – номер модуля 0 или 1), см. рисунок 19.1. Оба канала полностью идентичны. Каждый модуль может функционировать в одном из трех режимов:

- контроллер шины (КШ);
- оконечное устройство (ОУ);
- монитор шины (МШ).

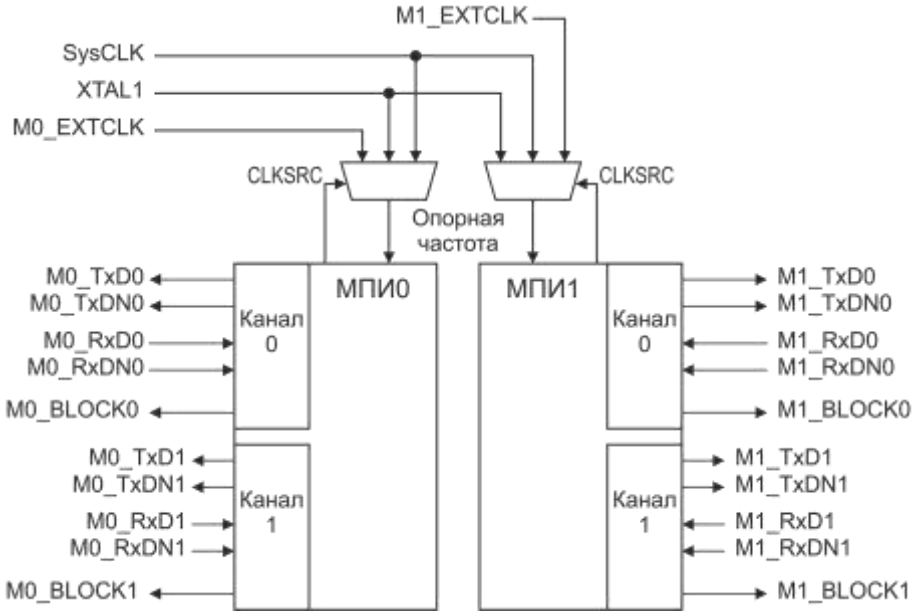


Рисунок 19.1 – Модули МПИ0 и МПИ1

Источники опорной частоты для блока МПИ0:

- вход M0_EXTCLK микроконтроллера (внешний источник);
- вход XTAL1 микроконтроллера (внешний источник);
- тактовый сигнал микроконтроллера SysCLK.

Источники опорной частоты для блока МПИ1:

- вход M1_EXTCLK микроконтроллера (внешний источник);
- вход XTAL1 микроконтроллера (внешний источник);
- тактовый сигнал микроконтроллера SysCLK.

При использовании сигнала со входа Mn_EXTCLK или XTAL1 его частота должна иметь одно из следующих значений (МГц): 12, 16, 20, 24, 28, 32, 36, 40, 44, 48 или 52.

При использовании внутреннего синхросигнала Sys_CLK его частота должна иметь одно из следующих значений (МГц): 24, 32, 40, 48, 56, 64, 72, 80, 88, 96 или 104.

Выбор источника сигнала опорной частоты осуществляется полем CLKSRC, а задание значения частоты – полем DIV регистра CONF.

19.1 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 последовательность сообщений.

Для организации последовательности сообщений используются области памяти SFR1 с адресами от 0010_0000h до 0010_0FFFh и от 0010_1000h до 0010_1FFFh для МПИ0 и МПИ1, соответственно. В этих областях следует формировать блоки данных для каждого сообщения. Блоки данных состоят из управляющей и передаваемой информации, и в них также резервируется место для приема данных из линии и записи информации о результате обмена.

На рисунке 19.2 представлен пример размещения блоков данных в области SFR1.

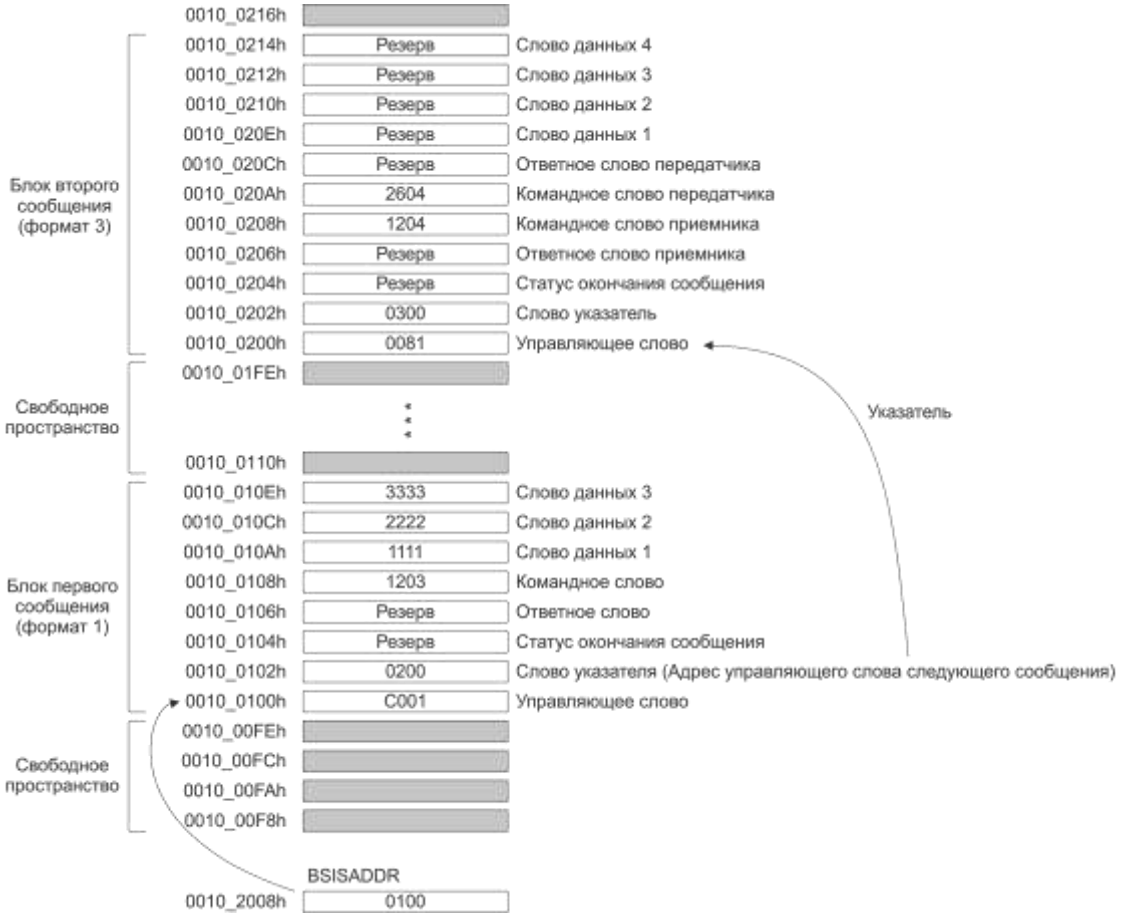


Рисунок 19.2 – Пример размещения двух блоков данных в ОЗУ

Пояснение к рисунку 19.2.

Блок данных первого сообщения размещен в диапазоне адресов с 0010_0100h по 0010_010Fh и имеет в своем составе указатель на начало следующего блока сообщения. В регистре ADDR находится значение 0100h – младшее слово адреса управляющего слова первого блока. Блок данных второго сообщения размещен с 0010_0200h по 0010_0215h.

Примечание – При указании адреса управляющего слова используется только младшее слово адреса, поскольку старшее слово аппаратно предопределено как 0010h.

Порядок размещения блоков сообщений в пределах соответствующей области памяти не важен, поскольку все они связаны посредством указателей.

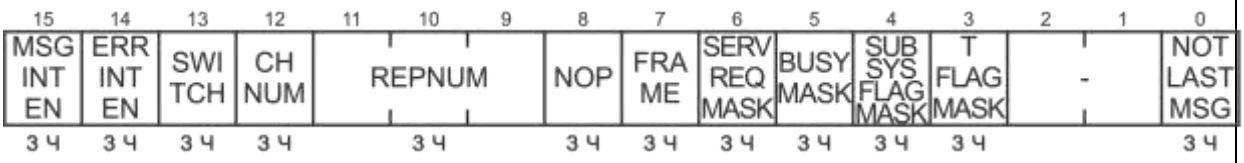
Структура блока данных зависит от формата сообщения. Так, на рисунке 19.2 блок данных первого сообщения имеет формат 1, а блок данных второго – формат 3. Всего предусмотрено 10 форматов, описание которых приведено в таблице 19.6.

Каждый блок данных сообщения состоит из:

- управляющего слова, которое содержит параметры управления только для этого сообщения и записывается пользователем;
- слова указателя, который содержит адрес управляющего слова следующего блока и записывается пользователем;
- слова статуса окончания сообщения, которое записывается аппаратно по окончании сообщения;
- ответного слова, которое записывается аппаратно после принятия ответного слова (второго ответного слова в формате 3);
- командного слова, которое записывается пользователем;
- слов данных, которые записываются пользователем, если это слова для передачи или аппаратно, если это слова, принятые из линии.

Подробное описание слов представлено в таблицах 19.1 – 19.5.

Таблица 19.1 – Управляющее слово (УС)



Поле	Бит	Описание	
MSGINT EN	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 CON	
ERRINT EN	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 CON	
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 CON)

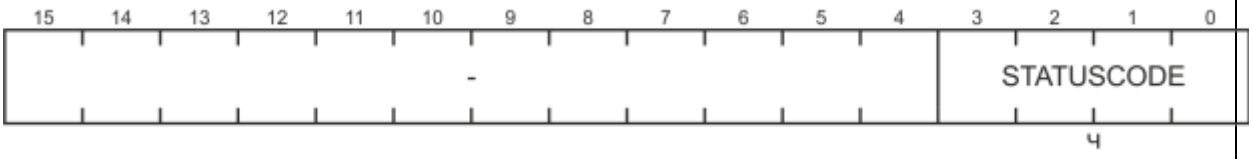
Окончание таблицы 19.1

<table>
<tr><th>Поле</th><th>Бит</th><th colspan="3">Описание</th></tr>
<tr><td>CHNUM</td><td>12</td><td colspan="3">Выбор канала для передачи/приема</td></tr>
<tr><td></td><td></td><td>0</td><td colspan="2">Основной</td></tr>
<tr><td></td><td></td><td>1</td><td colspan="2">Резервный</td></tr>
<tr><td>REPNUM</td><td>11- 9</td><td colspan="3">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</td></tr>
<tr><td></td><td></td><td>000</td><td>Повтор запрещен</td><td rowspan="8">Разрешено, если установлен бит REPEN регистра CON</td></tr>
<tr><td></td><td></td><td>001</td><td>1</td></tr>
<tr><td></td><td></td><td>010</td><td>2</td></tr>
<tr><td></td><td></td><td>011</td><td>3</td></tr>
<tr><td></td><td></td><td>100</td><td>4</td></tr>
<tr><td></td><td></td><td>101</td><td>5</td></tr>
<tr><td></td><td></td><td>110</td><td>6</td></tr>
<tr><td></td><td></td><td>111</td><td>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)</td></tr>
<tr><td>NOP</td><td>8</td><td colspan="3">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</td></tr>
<tr><td>FRAME</td><td>7</td><td colspan="3">Указатель формата сообщения</td></tr>
<tr><td></td><td></td><td>0</td><td colspan="2">Используется любой формат, за исключением форматов 3 и 8</td></tr>
<tr><td></td><td></td><td>1</td><td colspan="2">Используется формат 3 или формат 8</td></tr>
<tr><td>SERVREQ MASK</td><td>6</td><td colspan="3">Маска бита SERVREQ ответного слова</td></tr>
<tr><td>BUSY MASK</td><td>5</td><td colspan="3">Маска бита BUSY ответного слова</td></tr>
<tr><td>SUBSYS FLAG MASK</td><td>4</td><td colspan="3">Маска бита SUBSYSFLAG ответного слова</td></tr>
<tr><td>TFLAG MASK</td><td>3</td><td colspan="3">Маска бита TFLAG ответного слова</td></tr>
<tr><td>NOT LASTMSG</td><td>0</td><td colspan="3">Индикатор последнего сообщения.
Сообщение, в управляющем слове которого этот бит не установлен, является последним в цепочке сообщений</td></tr>
<tr><td>–</td><td>2, 1</td><td colspan="3">Зарезервировано</td></tr>
<tr><td colspan="5">Примечание – Если бит маски установлен, то маска считается включенной. Бит ответного слова, закрытый соответствующей маской, считается сброшенным.</td></tr>
</table>

Таблица 19.2 – Слово указателя

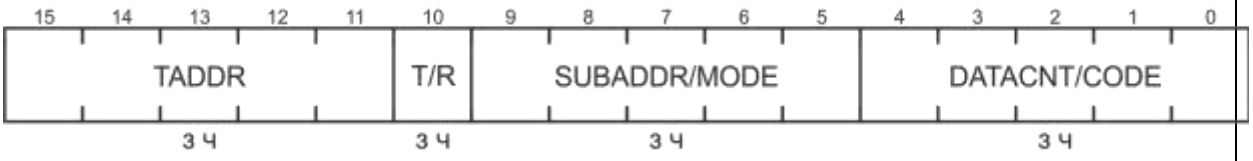
Поле	Бит	Описание
NEXTMSGADDR	15-0	Адрес управляющего слова следующего сообщения

Таблица 19.3 – Слово статуса окончания сообщения



Поле	Бит	Описание	
STATUSCODE	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
		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 19.4 – Командное слово (КС)



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 19.5 – Слово данных (СД)

Поле	Бит	Описание
DATAWORD	15-0	Данные

Инициализация

Для запуска последовательности сообщений необходимо:

1 Задать источник опорной частоты и коэффициент деления опорной частоты в полях CLKSRC и DIV регистра CONF.

2 Установить режим контроллера шины, записав 01b в поле MODE регистра CONF.

3 Сформировать блоки сообщений, и занести адрес управляющего слова первого блока (стартовый адрес) в регистр ADDR. В примере на рисунке 19.1 стартовый адрес – 0100h.

4 Установить паузу между сообщениями в регистре SPACE в микросекундах. По умолчанию пауза составляет 4 мкс.

5 Для инициализации передачи следует установить бит START в регистре CON. После этого запись в регистр CONF будет заблокирована, и на аппаратном уровне выполнятся следующие действия:

- чтение управляющего слова сообщения, на которое указывает регистр ADDR;
- проверка состояния бита NOP управляющего слова и переход к управляющему слову следующего сообщения, если бит установлен или чтение командного слова сообщения и передача его в канал, заданный битом CHNUM;
- передача/получение всех остальных слов сообщения в порядке, установленном форматом сообщения.

6 После успешного окончания 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.

7 В случае обнаружения ошибки проверяется состояние бита REPEN регистра CON. Если бит установлен и значение поля REPNUM управляющего слова больше нуля, 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 CON и бит SWITCH в управляющем слове, при каждом повторе сообщения будет происходить переключение канала на альтернативный.

Прерывания

Управление прерываниями осуществляется с помощью битов USUAL_INT_EN и ERR_INT_EN регистра CON, а также MSGINTEN и ERRINTEN в управляющем слове каждого сообщения. При сброшенных битах USUAL_INT_EN и ERR_INT_EN прерываний не будет.

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, 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.

Для определения источника прерывания используется регистр STAT, где сохраняется код источника 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.

Форматы основных сообщений применяются для передачи информации, предназначенной одному из удаленных терминалов с получением от него соответствующего ответа. Описание форматов приведено в таблице 19.6.

Таблица 19.6 – Форматы основных сообщений

Название формата	Алгоритм передачи	Порядок размещения слов в блоке сообщения
Формат 1 (см. пример на рисунке 19.2)	Передача данных от контроллера шины к удаленному терминалу	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 удаленного терминала 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 удаленного терминала к удаленному терминалу	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 удаленному терминалу	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 удаленному терминалу 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.
Формат 6	Передача команды управления и одного слова данных от контроллера шины к удаленному терминалу	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 удаленным терминалам без получения от них ответных слов. Сообщение является групповым, если в нем адрес удаленного терминала равен 1Fh. Каждый удаленный терминал, который может принять команду общего вызова после ее обнаружения, устанавливает соответствующий флаг в ответном слове, но само ответное слово не передает. Описания форматов приведено в таблице 19.7.

Таблица 19.7 – Форматы групповых сообщений

Название формата	Алгоритм передачи	Порядок размещения слов в блоке сообщения
Формат 7	Передача данных (в групповом сообщении) от контроллера шины к удаленным терминалам	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 удаленного терминала к удаленным терминалам	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 удаленным терминалам	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 удаленным терминалам	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 0h или 1Fh в поле SUBADDR/MODE. Поле DATACNT/CODE команды управления является кодом команды. Команды управления применяются только для управления оконечными устройствами (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 команды управления с кодами 10h – 15h применяются с одним словом данных.

Возможность использования той или иной команды управления в групповых сообщениях и состояние бита T/R указаны в таблице 19.8.

Таблица 19.8 – Команды управления, передаваемые в командном слове

Код команды	Название команды управления	Бит T/R	Возможность применения	
			в групповых сообщениях	со словом данных
00h	Принять управление интерфейсом	1	–	–
01h	Синхронизация	1	+	
02h	Передать ответное слово	1	–	
03h	Начать самоконтроль ОУ	1	+	
04h	Блокировать передатчик	1	+	
05h	Разблокировать передатчик			
06h	Блокировать признак неисправности ОУ			
07h	Разблокировать признак неисправности ОУ			
08h	Установить ОУ в исходное состояние			
10h	Передать векторное слово	1	–	+
11h	Синхронизация со словом данных	0	+	
12h	Передать последнюю команду	1	–	
13h	Передать слово ВСК ОУ			
14h	Блокировать выбранный передатчик	0	+	
15h	Разблокировать выбранный передатчик			
09h-0Fh, 16h-1Fh	Зарезервировано. Не использовать!			

19.2 Удаленный терминал (УТ)

Удаленный терминал (или оконечное устройство) выполняет команды контроллера шины. Каждому удаленному терминалу присваивается адрес в диапазоне от 01h до 1Eh.

В режиме удаленного терминала модулю МПИ0 выделяется область памяти SFR1 с адресами от 0010_0000h до 0010_0FFFh, а модулю МПИ1 – от 0010_1000h до 0010_1FFFh.

Таким образом, каждый модуль имеет свою независимую память сообщений. Если модуль не используется, указанная область 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.

Для МПИ0 адресация блоков начинается с 0010_0000h. Половина области сообщений с адресами от 0010_0040h до 0010_0FFFh выделена для записи и хранения принятых слов данных (на рисунке 19.3 отмечена «Для приема»). Вторая половина – для хранения слов данных, записанных пользователем и предназначенных для передачи. Для МПИ1 адресация блоков начинается с 0010_1000h. Выбор одной из двух областей осуществляется битом T/R командного слова.

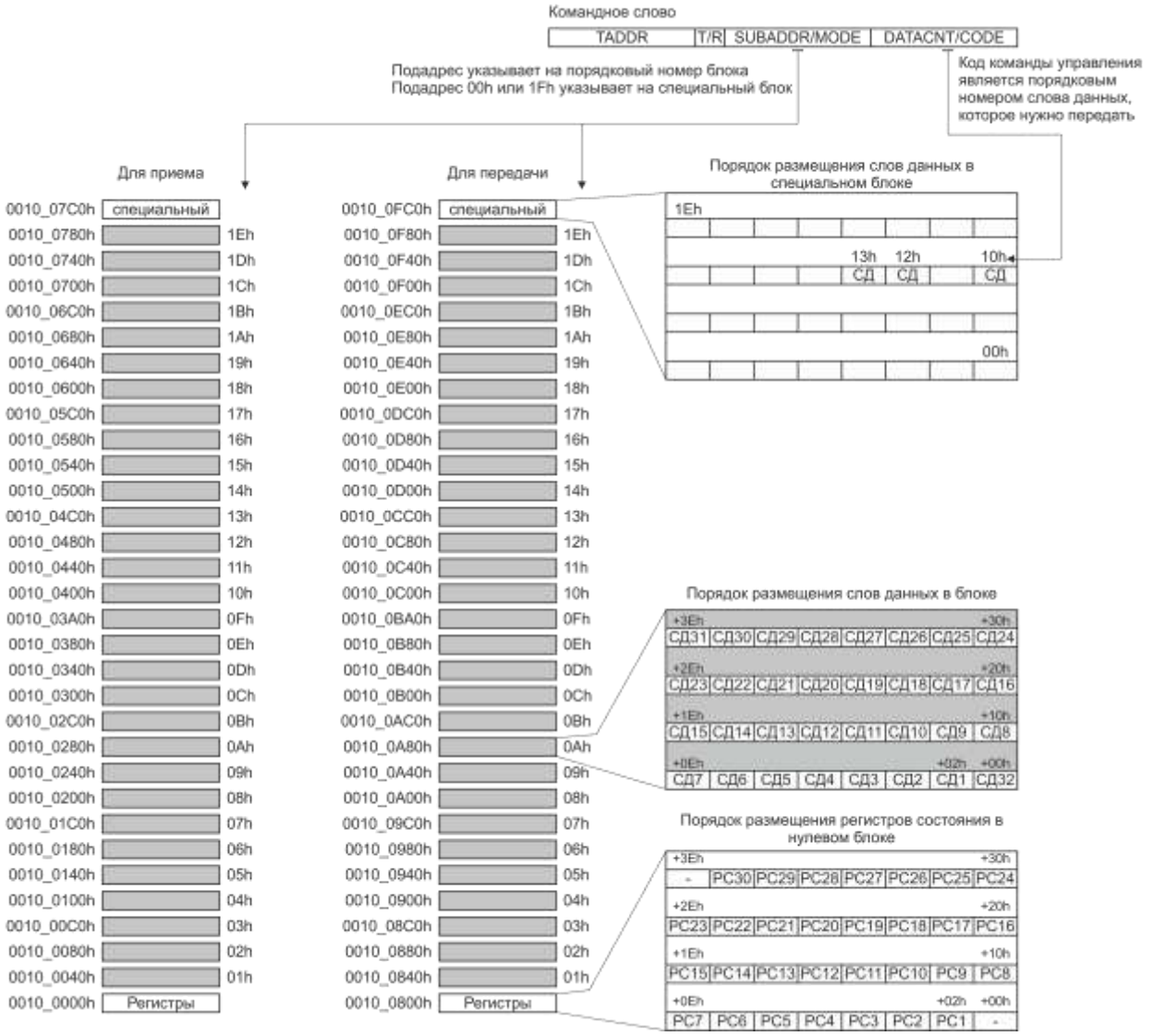


Рисунок 19.3 – Структура памяти сообщений МПИ0

В нулевом блоке области располагаются регистры состояния памяти подадреса. Тридцать первый блок данных является специальным. Остальные 30 блоков могут заполняться принятыми данными. Структура всех блоков идентична. Каждый блок может хранить до 32 слов данных.

Прием

Получив командное слово, оконечное устройство анализирует его.

T/R = 0 указывает на то, что нужно принять данные. SUBADDR указывает на номер блока, в который следует записать слова данных. DATACNT определяет количество данных, которые следует принять (DATACNT = 00h означает, что будут переданы 32 слова данных).

Принимаемые слова данных записываются в блок последовательно, начиная со второй ячейки СД1, затем в СД2, СД3 и т. д. 32 слово данных записывается в первую ячейку блока, обозначенную СД32.

Если значение поля SBADDR/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 19.9).

Таблица 19.9 – 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
10h	0010_0FE0h
12h	0010_0FE4h
13h	0010_0FE6h

Передача

Получив командное слово, оконечное устройство анализирует его.

T/R = 1 указывает на то, что нужно передать данные. SBADDR указывает на номер блока, из которого следует передавать данные. DATACNT указывает, какое количество данных следует передать (DATACNT = 00h означает, что требуется передать 32 слова данных).

Первым передается слово из ячейки СД1, затем из СД2, СД3 и т. д. Слово данных из ячейки СД32 будет передано последним.

Если значение поля SBADDR/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 специальный блок. 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.

Групповые сообщения

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 CON.

Также можно включить функцию распознавания командного и ответного слов по десятому биту, установив бит B10EN. Это необходимо для нормальной работы монитора шины, если он имеется в общей сети.

В режиме удаленного терминала аппаратно реализуется циклический возврат данных. Если терминал получает команду на прием данных в подадрес 1Eh, а сразу за этим команду на передачу такого же количества слов из подадреса 1Eh, то будут переданы данные, полученные в предыдущем сообщении. Если же количество слов, принятых в подадрес 1Eh, будет отличаться от количества запрошенных из подадреса 1Eh слов, то переданы будут данные из 30 блока данных, начиная с адреса 0010_0F82h (обычный режим работы).

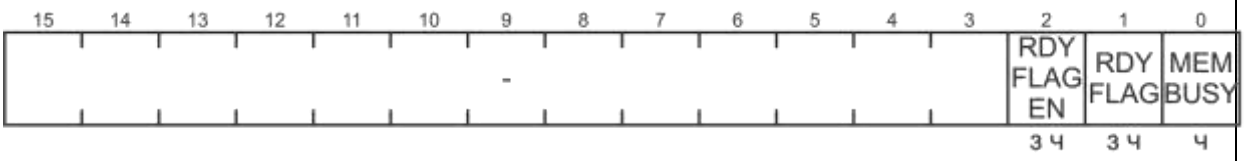
Регистры состояния памяти подадреса

Для исключения потери и искажения информации при записи данных в память каждый подадрес имеет два регистра состояния: один регистр – для принимающей области, другой – для передающей. Регистры показывают, идет ли обмен данными с областями памяти этого подадреса или нет, а также режимы работы с этим подадресом, см. таблицу 19.10.

Адрес регистра состояния задается смещением относительно адреса начала области сообщений. Смещение вычисляется как 2×SUBADDR.

Так, например, адрес регистра состояния памяти подадреса 07h для области сообщений, предназначенной для приема, будет 0010_0000h + 2×07h = 0010_000Eh, а для области сообщений, предназначенной для передачи – 0010_0800h + 2×07h = 0010_080Eh, см. рисунок 19.3. Приведенный в примере расчет адресов справедлив для модуля МПИ0. Для аналогичного подадреса модуля МПИ1 адреса будут 0010_100Eh и 0010_180Eh.

Таблица 19.10 – Регистр состояния памяти подадреса



Поле	Бит	Описание	
RDY FLAG EN	2	Разрешение работы с флагом готовности RDYFLAG	
		0	Запрещено
		1	Разрешено
RDY FLAG	1	Флаг готовности	
MEM BUSY	0	0	Обмена данными не происходит
		1	В настоящий момент происходит обмен данными с этой областью памяти
–	15-3	Зарезервировано	

Если установлен бит RDYFLAGEN, т. е. работа с флагом готовности RDFLAG разрешена, то модуль МПИ функционирует следующим образом.

Если получена команда на прием, то флаг RDYFLAG регистра, соответствующего принимающей области памяти рабочего подадреса, устанавливается аппаратно после окончания приема. После прочтения принятых данных, пользователь должен программно сбросить флаг. Если пришла новая команда на прием по этому же подадресу, а RDYFLAG не сброшен, то ответное слово передается с установленным битом BUSY. Для подадреса 1Eh при циклической передаче флаг сбрасывается аппаратно после передачи данных в линию.

Для передачи данных пользователь программно устанавливает флаг готовности RDYFLAG, соответствующий передающей области памяти подадреса, после записи всех слов данных, предназначенных к передаче для этого подадреса. Флаг сбрасывается аппаратно после окончания передачи. Если в момент получения команды на передачу RDYFLAG не установлен, в ответном слове передается бит BUSY=1, а данные не передаются.

Если работа с флагом RDYFLAG запрещена, то для доступа к данным необходимо проверить состояние флага MEMBUSY. Если он сброшен, требуется сбросить бит RDYFLAG, прочитать или записать данные, и снова установить RDYFLAG. В этом режиме работы, если флаг RDYFLAG сброшен, то в ответном слове бит BUSY всегда будет установлен. Если читать или записывать данные при установленном бите MEMBUSY, возможны чтение или передача некорректных данных.

Инициализация

Для включения удаленного терминала необходимо:

1 Задать источник опорной частоты и коэффициент деления опорной частоты в полях CLKSRC и DIV регистра CONF.

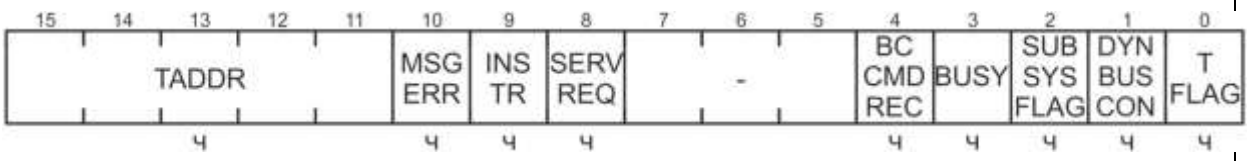
2 Установить режим удаленного терминала, записав 10b в поле MODE регистра CONF.

3 Записать собственный адрес удаленного терминала в поле TADDR регистра CON. При необходимости разрешить работу с групповым адресом, установив бит BCMSGEN регистра CON.

4 Запустить удаленный терминал установкой бита START регистра CON. При этом блокируются любые изменения регистра CONF и поля TADDR регистра CON. После установки бита START оконечное устройство 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 19.11.

Таблица 19.11 – Ответное слово (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)
SERVREQ	8	Запрос на обслуживание Устанавливается посредством бита SERVREQ регистра CON
BCCMDREC	4	Принято групповое сообщение. Устанавливается, если принято групповое сообщение, и установлен бит BCMSGEN в регистре CON
BUSY	3	Абонент занят. Устанавливается посредством бита BUSY регистра CON. Указывает на то, что подчиненное устройство (абонент) занят
SUBSYS FLAG	2	Неисправность абонента. Устанавливается посредством бита SUBSYSFLAG регистра CON. Указывает на ошибки в работе подчиненного устройства (абонента)
DYNBUSCON	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 CON
TFLAG	0	Неисправность удаленного терминала. Устанавливается посредством бита TFLAG регистра CON. 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 CON. Если оба бита сброшены, то прерываний не будет.

Если установлен бит USUAL_INT_EN, то прерывания будут происходить после безошибочного окончания сообщения.

Если установлен бит ERR_INT_EN, то прерывания будут происходить после сообщений, завершенных с ошибкой.

Прерывания никогда не формируются после выполнения команд управления:

- передать ответное слово (02h);
- блокировать признак неисправности удаленного терминала (04h);
- разблокировать признак неисправности удаленного терминала (07h);
- передать последнюю команду (12h).

Для определения причины прерывания используется регистр STAT, где сохраняется код причины прерывания и данные о последней переданной команде.

19.3 Монитор шины (МШ)

Монитор шины непрерывно «слушает» оба канала (основной и резервный) и сохраняет в памяти результаты мониторинга. Монитор шины имеет собственный 32-разрядный таймер, который запускается при включении режима и отсчитывает интервалы времени по 0,5 мкс. Информация о каждом сообщении в линии сохраняется в информационном блоке, который формируется в SFR1. Для МПИ0 доступный диапазон адресов от 0010_0000h до 0010_0FFFh, для МПИ1 – от 0010_1000h до 0010_1FFFh.

Длина каждого информационного блока составляет от 5 до 40 слов в зависимости от сообщения. Информационный блок сохраняется в ОЗУ от младшего адреса к старшему и имеет следующий формат:

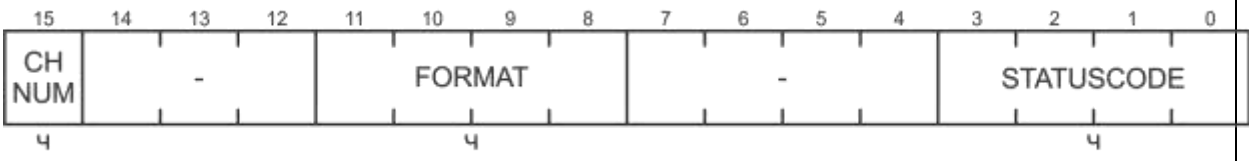
- адрес следующего информационного блока. Если адрес равен 0, значит текущее сообщение полностью еще не принято;
- старшее слово таймера на момент принятия достоверного командного слова;
- младшее слово таймера на момент принятия достоверного командного слова;
- информационное слово (см. таблицу 19.12).
- все принятые слова (от 5 до 40) сообщения, начиная с командного, по порядку принятия.

Монитор шины отличает командное слово от ответного по десятому биту. Это необходимо учитывать при проектировании всей системы.

Пример заполнения памяти представлен на рисунке 19.4.

Рисунок 19.4 – Пример заполнения SFR1 в режиме монитора шины

Таблица 19.12 – Информационное слово



Поле	Бит	Описание	
CHNUM	15	Обнаруженный канал	
		0	Основной
		1	Резервный
FORMAT	11-8	Формат сообщения	
STATUS CODE	3-0	Код статуса окончания сообщения	
		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 передатчика (форматы 3 и 8)
		0110	Неправильный синхроимпульс в ответном слове передатчика (форматы 3 и 8)
		0111	Ненулевое ответное слово передатчика (форматы 3 и 8)
		1000	Ошибка манчестерского кода в ответном слове передатчика (форматы 3 и 8)
		1001	Отсутствие ответного слова передатчика (форматы 3 и 8)
		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 передатчика (форматы 3 и 8)
		1110	Отсутствие второго командного слова или нарушение непрерывности при приеме второго командного слова в форматах 3 и 8
		1111	Неверное командное слово (недопустимая комбинация бит) передатчика в форматах 3 и 8
–	14-12, 7-4	Зарезервировано	

В регистре CON 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:

1 Задать источник опорной частоты и коэффициент деления опорной частоты в полях CLKSRC и DIV регистра CONF.

2 Установить режим монитора шины, записав 11b в поле MODE регистра CONF.

3 Установить требуемый режим выборки в поле MONMODE регистра CON, а также, если это необходимо, контролируемые адрес и подадрес.

4 Задать частоту прерываний посредством поля INTNUM регистра CON.

5 Запустить монитор шины, установив бит START регистра CON. При этом блокируются любые изменения регистра CONF и полей ADDR, SUBADDR и MONMODE регистра CON.

После установки бита START монитор шины переходит в режим ожидания командного слова, соответствующему режиму выборки.

Приняв достоверное, соответствующее выбранному режиму командное слово, монитор шины фиксирует и сохраняет в информационном блоке по адресу, указанному в регистре ADDR, 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 ADDR. По этому же адресу в первое слово следующего информационного блока записывается 0000h. Далее все повторяется.

Прерывания

В поле INTNUM можно задать через сколько записанных информационных блоков генерируется прерывание. При этом в регистре STAT хранится адрес первого еще непрочитанного информационного блока, а регистре ADDR – адрес информационного блока, который будет заполнен после принятия следующего сообщения. После чтения регистра STAT, 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 Блок АЦП

Аналого-цифровой преобразователь (АЦП) реализован по схеме последовательного приближения и имеет структурную схему, представленную на рисунке 21.1. АЦП может быть запрограммирован на работу в режиме однократного/непрерывного преобразования выбранного канала или в режиме последовательного сканирования каналов.

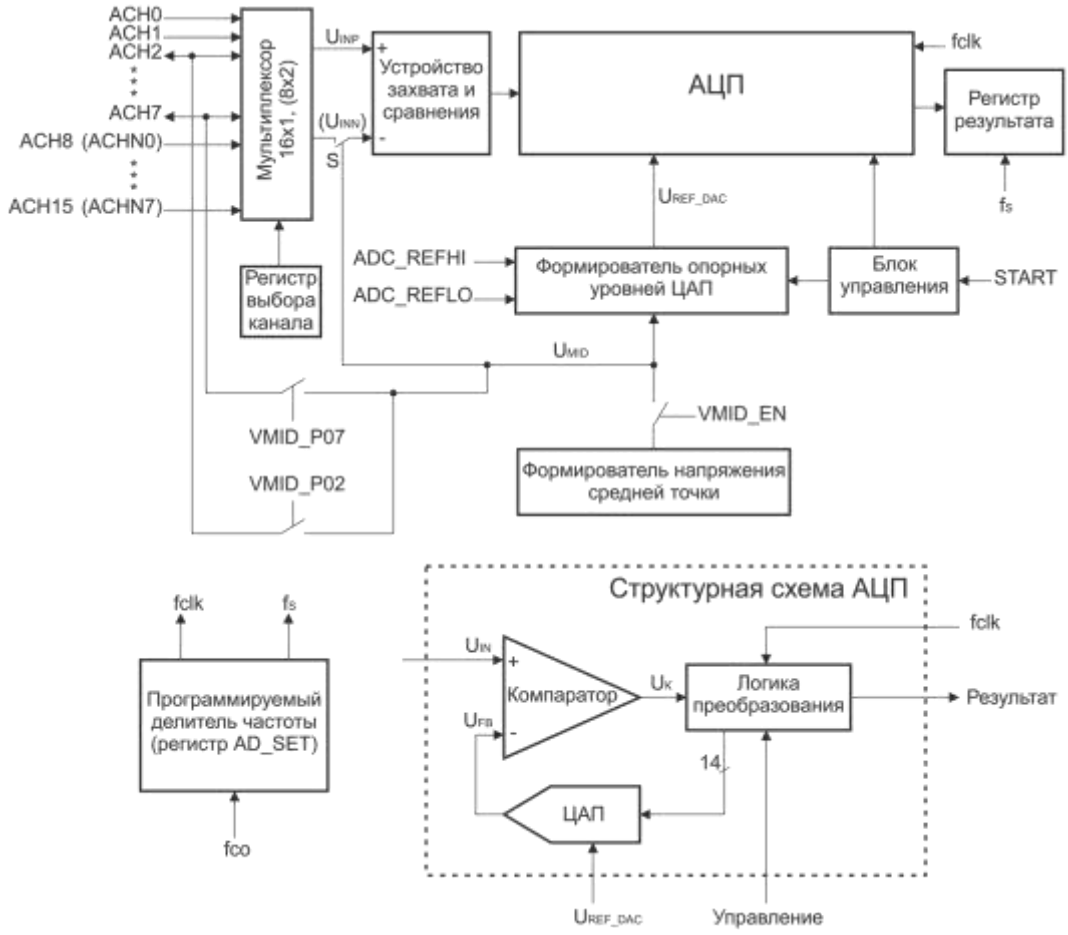


Рисунок 21.1 – Структурная схема модуля АЦП

Мультиплексор подключает один из входов к устройству захвата и хранения в режиме однополярного включения входов – таким образом, организуются 16 каналов. В режиме дифференциального включения входов организуются восемь пар каналов (например, ACH0-ACHN0, ACH1-ACHN1 и т. д.).

Устройство захвата и хранения фиксирует уровень напряжения входного сигнала в момент начала преобразования на входных емкостях и обеспечивает его хранение до окончания преобразования.

Преобразователь включает в свой состав компаратор, логику преобразования с регистром последовательного приближения и цифро-аналоговый преобразователь (ЦАП). В результате преобразования на выходе получается цифровой код, соответствующий входному напряжению ΔU_{IN} с точностью до одного U_{LSB}. Результат преобразования сохраняется в регистре результата АЦП и может быть считан до конца следующего преобразования.

Настройка диапазона преобразования осуществляется изменением величины опорных напряжений на входах ADC_REFHI и ADC_REFLO.

Формирователь напряжения средней точки позволяет задать напряжение на уровне приблизительно (1/2) U_{CC2}. Напряжение средней точки может задаваться внешним источником через вывод ACH2 или ACH7, при этом внутренний источник должен быть отключен.

Блок управления с программируемым делителем частоты fadc позволяет осуществлять дискретную подстройку частоты преобразования f_{CLK_ADC}. Коэффициент деления задается посредством регистра SET.

21.1 Функциональные возможности

АЦП преобразует аналоговые уровни напряжений на входе в цифровые значения в режиме однополярного/дифференциального включения входов (режим задается битом EN_DIF регистра CON). Передаточная характеристика АЦП для режима однополярного включения входа показана на рисунке 21.2.

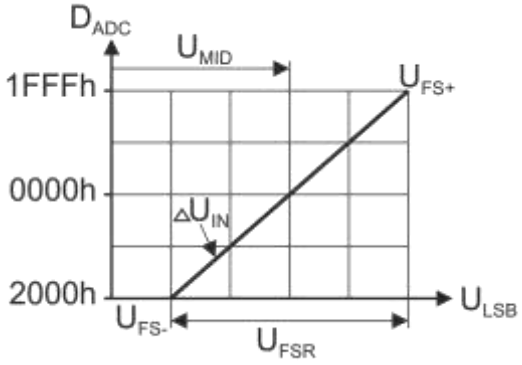


Рисунок 21.2 – Передаточная характеристика АЦП для режима однополярного включения входа (в общем виде)

Передаточная характеристика АЦП для режима дифференциального включения входов показана на рисунке 21.3.

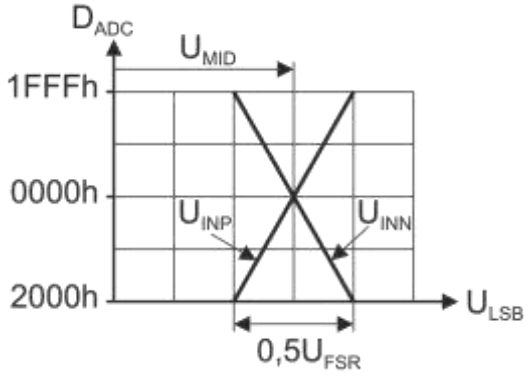


Рисунок 21.3 – Передаточная характеристика АЦП для режима дифференциального включения входов U_{INP} и U_{INN} относительно средней точки U_{MID} (дополнительно см. рисунок 21.6)

Соотношение режимных параметров в канале АЦП

Термины, обозначения и соотношения режимных параметров АЦП приведены в соответствии с ГОСТ 29109–91.

Приведенные ниже соотношения (21.1) – (21.5) позволяют перевести входное напряжение в цифровой код, рассчитать шаг квантования (при заданной разрядности),

определить коэффициент преобразования и диапазон преобразуемых напряжений АЦП, а также осуществить обратный перевод цифрового кода в напряжение.

Шаг квантования (единица младшего разряда) определяется отношением:

$$U_{LSB} = \frac{U_{FSR}}{2^N}, \tag{21.1}$$

где U_{FSR} – диапазон напряжений преобразования АЦП, при выходном коде 2^N;

N – количество разрядов.

Диапазон преобразования определяется отношением:

$$U_{FSR} = U_{REF} \cdot K_G, \tag{21.2}$$

где U_{REF} – опорное напряжение АЦП;

K_G – коэффициент преобразования АЦП (тангенс угла наклона прямой преобразования).

Значение опорного напряжения АЦП U_{REF} расчитывается как:

$$U_{REF} = U_{REFH} - U_{REFL}. \tag{21.3}$$

Коэффициент преобразования указан на графике, представленном на рисунке 21.4.

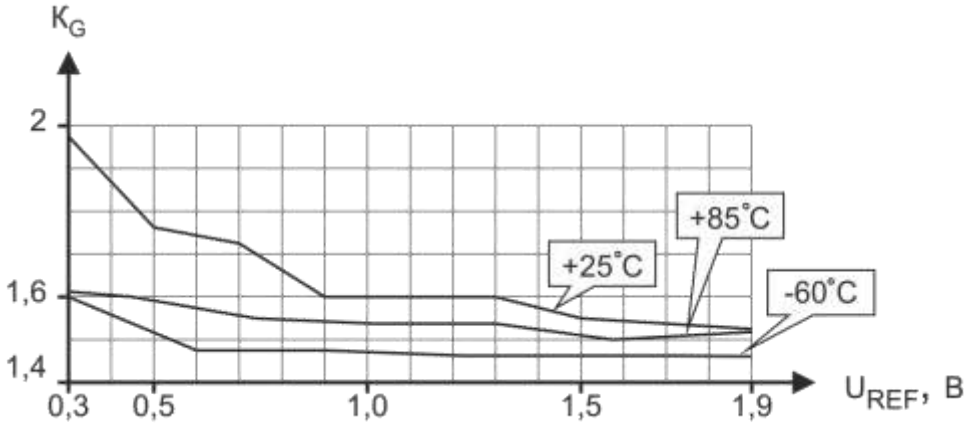


Рисунок 21.4 – Зависимость коэффициента преобразования АЦП от значения опорного напряжения

Тогда передаточная характеристика преобразования напряжения канала АЦП, показанная на рисунках 21.2 и 21.3 в режиме дифференциального включения входа, описывается формулой (21.4), а в режиме однополярного включения – формулой (21.5):

$$D_{ADC} = \frac{(U_{INP} - U_{INN}) \cdot 8192}{(U_{REFH} - U_{REFL}) \cdot K_G}. \tag{21.4}$$

$$D_{ADC} = 2 \times \frac{(U_{INP} - U_{INN}) \cdot 8192}{(U_{REFH} - U_{REFL}) \cdot K_G}. \tag{21.5}$$

Типовая зависимость диапазона преобразования U_{FSR} АЦП ИС от величины опорного напряжения в диапазоне температур от минус 60 до плюс 85 °С приведена на рисунке 21.5.

Напряжения внешних опорных источников U_{REFH} и U_{REFL} устанавливаются, исходя из условий применения микроконтроллера.

Максимальное значение диапазона преобразования входных сигналов U_{FSR} достигается при значении $U_{REF} = (U_{REFH} – U_{REFL}) = 2{,}1$ В (согласно графику на рисунке 21.5).

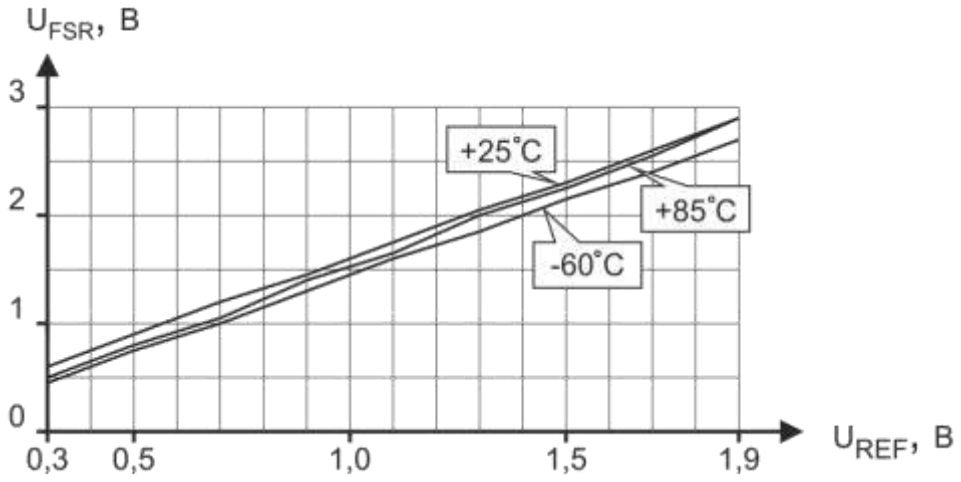


Рисунок 21.5 – Типовая зависимость U_{FSR} АЦП от опорного напряжения U_{REF}

Диаграмма входных сигналов при измерении динамических параметров АЦП SINAD и THD приведена на рисунке 21.6. Динамические и статические параметры АЦП приведены в таблицах 2.2 и 2.3.

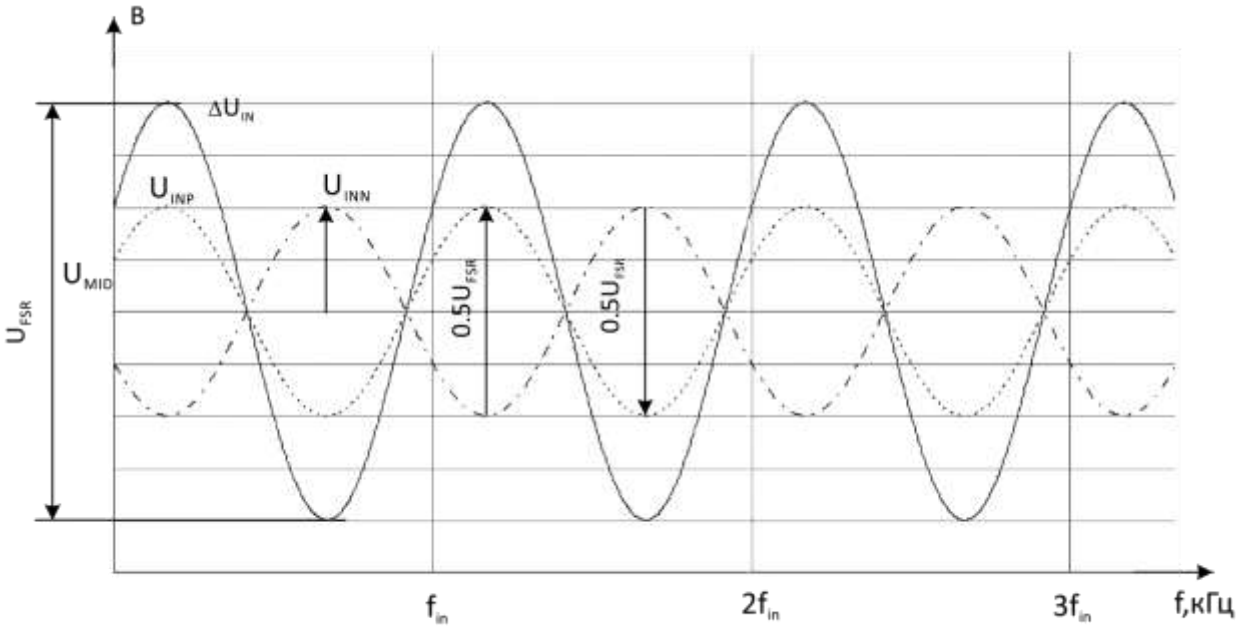


Рисунок 21.6 − Временные диаграммы входных напряжений АЦП при измерении динамических параметров

Диаграммы зависимостей общих гармонических искажений THD и отношения сигнал/(шум+искажения) SINAD в канале АЦП в режиме одиночного входа представлены на рисунках 21.7 и 21.8.

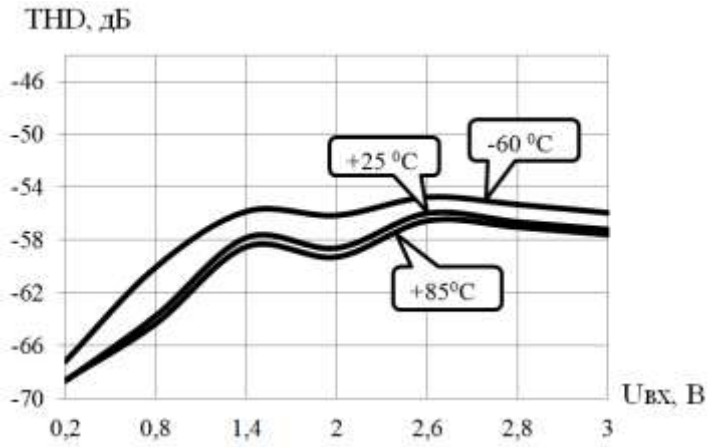


Рисунок 21.7 – Зависимость общих гармонических искажений THD канала АЦП в режиме одиночного входа в диапазоне температур от минус 60 до 85 °С при U_{CC2} = 3,3 В, U_{REF} = 2,1 В, f_s = 75 кГц

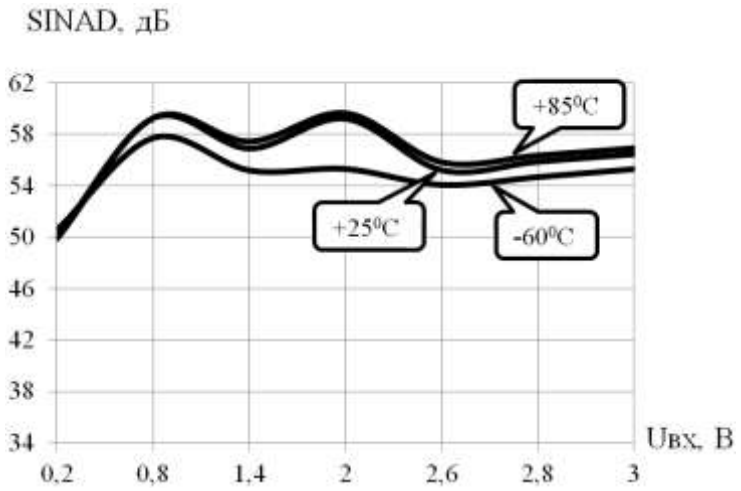


Рисунок 21.8 – Зависимость отношения сигнал/(шум+искажения) SINAD в канале АЦП в режиме одиночного входа в диапазоне температур от минус 60 до 85 °С при U_{CC2} = 3,3 В, U_{REF} = 2,1 В, f_s = 75 кГц

АЦП выполняет преобразование за 10 тактов синхросигнала, который формируется из системного тактового сигнала (с частотой f_{ADC}) путем его деления. Коэффициент деления определяется битовым полем ADC_CLK регистра SET. Частота синхросигнала АЦП не должна превышать 1,5 МГц (частота дискретизации 150 кГц).

Входные опорные напряжения АЦП подаются на входы ADC_REFHI и ADC_REFLO микроконтроллера. При использовании внутреннего источника средней точки АЦП биты VMID_ACH7 и VMID_ACH2 регистра EN должны быть сброшены.

Внешний источник средней точки подключается к выводу ACH7 или ACH2, а его использование разрешается установкой бита VMID_ACH7 или VMID_ACH2, соответственно. При этом внутренний источник напряжения средней точки должен быть выключен (сброшен бит VMID_EN регистра EN). При использовании внешнего источника напряжения средней точки необходимо выбирать значение напряжения, исходя из условия:

$$1{,}485\ \text{В} < U_{MID} < 1{,}815\ \text{В при } U_{CC} = 3{,}3\ \text{В}.$$

Управление режимом работы входов осуществляется битом EN_DIF регистра CON. В режиме однополярного включения входа преобразуемый сигнал подается на

вход U_{INP}, на инверсный вход U_{INN} устройства захвата и хранения аппаратно подается напряжение средней точки U_{MID}. В этом режиме доступен любой из 16 (или 12) входов АЦП.

В режиме дифференциального включения входов выполняется преобразование разницы напряжения сигналов между входами ACHx и ACHNx. При этом сигнал с входа ACHx передается на вход U_{INP}, а сигнал ACHNx – на инверсный вход U_{INN}.

Для фильтрации высокочастотных составляющих сигнала, не входящих в полосу преобразования АЦП, на все используемые входы ACHx рекомендуется устанавливать входные RC-фильтры с частотой среза $0{,}5f_S$ (см. рисунок 21.9). Для расчета RC-фильтра следует использовать соотношение:

$$f = \frac{1}{2 \cdot \pi \cdot R \cdot C}, \qquad (21.6)$$

где f – частота среза, Гц;
R – сопротивление резистора, Ом;
C – емкость конденсатора, мкФ.

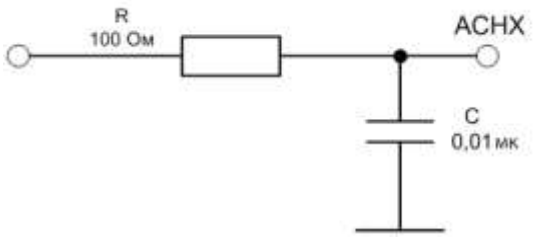


Рисунок 21.9 – Входной RC-фильтр АЦП при f_S = 150 кГц, f_{IN} = 15 кГц

21.2 Программирование АЦП

После подачи напряжения питания на выводы питания микроконтроллера модуль АЦП выключен, что обеспечивает снижение общей потребляемой мощности от источников питания.

Регистр EN

Инициализация АЦП запускается посредством регистра EN. Для включения необходимо:

- установить бит EN;
- включить источник напряжения средней точки АЦП установкой бита VMID_EN, либо, если предполагается использование внешнего источника, разрешить использование вывода ACH7 или ACH2 установкой бита VMID_ACH7 или VMID_ACH2, соответственно.

Включение источника напряжения средней точки АЦП выполняется в течение приблизительно 100 мкс. Не рекомендуется запускать преобразование в течение этого времени во избежание получения неправильного результата.

Регистр CON

Регистр обеспечивает выбор режима работы АЦП, управление входным мультиплексором для выбора канала и запуск преобразования.

АЦП может работать в одном из трех режимов (задается полем MODE):

- одиночного преобразования выбранного канала;
- многократного преобразования выбранного канала;
- последовательного сканирования каналов.

В режиме одиночного преобразования АЦП осуществляет однократный запуск преобразования сигнала на входе канала, заданного полем CHANNEL. По окончании преобразования полученный результат заносится в регистр результата. Режим удобен в

случаях, когда требуется преобразование определенного канала либо нескольких выборочных каналов.

В режиме непрерывного преобразования АЦП осуществляет многократный запуск преобразования сигнала на входе выбранного канала. По окончании преобразования полученный результат заносится в регистр результата и сразу же автоматически запускается следующий цикл преобразования. Так, АЦП будет работать до тех пор, пока не будет остановлен программно сбросом бита START. Режим удобен при оцифровке непрерывно меняющегося сигнала.

В режиме сканирования каналов АЦП осуществляет многократный запуск преобразования, последовательно меняя каналы (в сторону каналов с большим номером). При первом запуске начнется преобразование сигнала на входе канала, указанного в поле CHANNEL. По окончании преобразования полученный результат заносится в регистр результата, номер канала заносится в регистр номера, и сразу же автоматически запускается следующий цикл преобразования, но уже для следующего по порядку канала. После окончания преобразования сигнала на входе канала 15 будет запущено преобразование для нулевого канала, потом для первого и так далее. Так, АЦП будет работать до тех пор, пока не будет остановлен программно сбросом бита START.

В режиме сканирования каналов запуск преобразования производится последовательно для всех 16 каналов, независимо от режима включения входов (однополярного или дифференциального), поэтому режим является полно-функциональным только при однополярном включении входов (управляется битом EN_DIF).

При использовании дифференциального включения входов в режиме сканирования каналов после окончания преобразования канала 7 АЦП должен быть остановлен программно, а затем снова запущен в заданном режиме, начиная сканирование с канала 0–7. Если после окончания преобразования для седьмого канала АЦП не будет остановлен, то автоматически начнется преобразование для восьмого канала, затем девятого и так далее с выдачей неправильного результата.

Запуск преобразования начинается сразу после установки бита START. В режиме одиночного преобразования бит аппаратно сбрасывается после запуска, поэтому по окончании преобразования АЦП останавливается автоматически. В режимах непрерывного преобразования выбранного канала и сканирования каналов бит START не сбрасывается, поэтому для прекращения преобразования бит следует очищать программно. Следует помнить, что преобразование, в течение которого был сброшен бит START, будет прервано, а результат потерян.

Сразу после запуска преобразования (в любом режиме работы) устанавливается бит BUSY в регистре RESULT, являющийся индикатором выполнения преобразования.

Регистр RESULT

Регистр результата преобразования. Перезагружается новым значением по окончании каждого преобразования, поэтому должен своевременно считываться программно.

Регистр CHANNEL

Регистр номера канала, для которого выполнялось преобразование. Перезагружается по окончании каждого преобразования (одновременно с регистром RESULT), поэтому должен своевременно считываться программно.

22 Сторожевой таймер

Сторожевой таймер позволяет сбросить систему в случае отказа программного обеспечения. Пользователь может включать таймер по собственному усмотрению.

Сторожевой таймер представляет собой 32-разрядный обратный счетчик, который загружается значением из регистра RELOAD. Счетчик уменьшается на единицу по каждому положительному фронту тактового сигнала.

Функциональная схема сторожевого таймера показана на рисунке 22.1

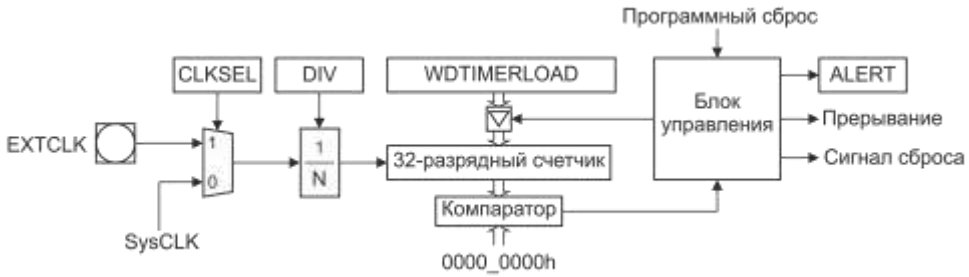


Рисунок 22.1– Функциональная схема сторожевого таймера

Управление таймером осуществляется посредством регистра CTRL. По умолчанию, после сброса микроконтроллера сторожевой таймер выключен.

Перед включением таймера следует задать начальное значение счетчика. Это значение записывается в регистр перезагрузки RELOAD (по умолчанию, состояние регистра 0000_FFFFh). Начальное значение должно быть не меньше 02h. В случае записи значения 00h или 01h, в счетчик будет загружено значение 02h.

Счетчик таймера может переключаться с частотой внутреннего синхросигнала микроконтроллера SysCLK или с частотой внешнего синхросигнала (асинхронного по отношению к SysCLK), подаваемого на вывод 167 (альтернативная функция WDT_EXTCLK) микроконтроллера. Выбор осуществляется битом CLKSEL регистра CTRL. Частота выбранного синхросигнала может быть уменьшена с помощью делителя, значение которого задается полем DIV.

Примечания

1 При использовании внешнего синхросигнала, следует включить соответствующую альтернативную функцию вывода.

2 Внешний синхросигнал является асинхронным по отношению к сигналу SysCLK и может иметь более высокую частоту.

Включение таймера осуществляется установкой бита WDTEN. Установка бита активирует загрузку значения из регистра RELOAD в регистр счетчика и включение счетчика.

Примечание – После включения сторожевой таймер может быть выключен только сбросом микроконтроллера.

Когда счетчик достигает значения нуля, в регистре STAT устанавливается флаг ALERT (если до этого момента он не был установлен), и генерируется запрос прерывания (если установлен бит INTEN в регистре CTRL). После этого в счетчик снова загружается значение из регистра LOAD, и счет продолжается.

Если при очередном обнулении счетчика флаг ALERT окажется установленным, блок управления сформирует сигнал, по которому будет активирован сброс микроконтроллера. Для предотвращения сброса следует очистить бит ALERT до того как счетчик достигнет

нуля. Для это нужно последовательно записать в регистр CLR два значения – сначала E1h, а затем 1Eh. После этого флаг ALERT будет сброшен, а счетчик таймера перезагружен.

Примечания

1 Последовательная запись значений E1h, 1Eh в регистр CLR всегда активирует перезагрузку счетчика сторожевого таймера значением из регистра RELOAD, независимо от состояния флага ALERT.

2 Регистр CLR не доступен для хранения данных, а только для сброса счетчика.

3 Между командой записи E1 и командой записи 1E могут располагаться другие команды программы.

Алгоритм работы сторожевого таймера показан на рисунке 22.2.

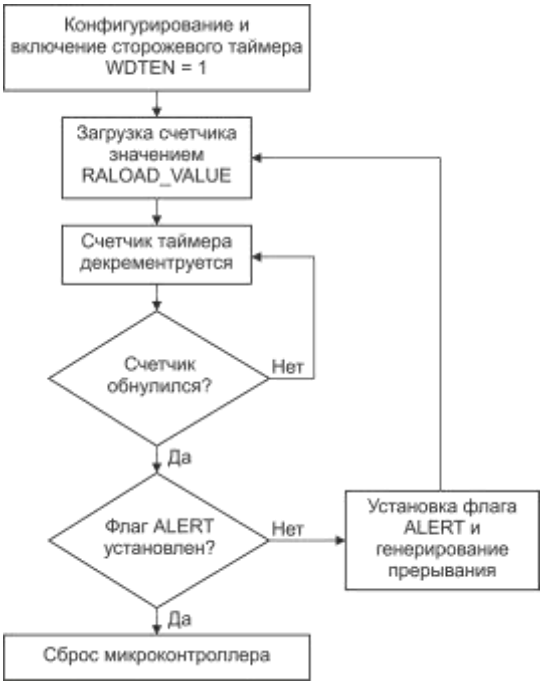


Рисунок 22.2 – Алгоритм работы сторожевого таймера

23 Специальные режимы работы микроконтроллеров

Микроконтроллеры поддерживают специальные режимы работы с пониженным энергопотреблением.

Режим Idle – работают только встроенные функциональные устройства, а микроконтроллер находится в режиме ожидания разрешенного прерывания от внешнего или внутреннего устройства.

Режим PowerDown – вся внутренняя синхронизация фиксируется в состоянии логического нуля, и генератор отключается. Внутреннее ОЗУ и большинство периферийной памяти сохраняет своё значение, если сохраняется напряжение питания. В этом режиме ток потребления составляет всего несколько микроампер.

Режим Slow – рабочая частота микроконтроллера уменьшается в два раза.

Помимо этих режимов, существует возможность независимого отключения тактовых сигналов периферийных устройств. Это позволяет гибко управлять общим потреблением, используя наиболее оптимальный режим в каждый момент времени работы системы.

23.1 Режим Idle

Тактирование ЦПУ зафиксировано в состоянии логического нуля, но периферия тактируется, и сигнал CLKOUT остается активным. ЦПУ останавливает выполнение команд, но внутреннее ОЗУ, таймеры/счетчики, последовательные порты, сервер периферийных транзакций и система прерываний продолжают функционировать. Потребление энергии уменьшается по сравнению с нормальным режимом работы.

Выводы управления шиной к внешней памяти переведены в неактивное состояние. Если сторожевой таймер разрешен, то он продолжает работать, поэтому микроконтроллер должен выходить периодически из режима Idle, чтобы сбросить таймер.

Вход в режим Idle

Вход в режим Idle осуществляется по команде MOD #1.

Выход из режима Idle

Прерывание или аппаратный сброс выведут устройство из режима Idle. Из-за того, что вся периферия остается активной в этом режиме, любой разрешенный источник прерывания может выработать прерывание. По прерыванию возобновляется тактирование ЦПУ и начинается выполнение соответствующей подпрограммы обслуживания прерывания. Только прерывание, назначенное на обработку стандартным методом, может вывести микроконтроллер из режима Idle. Если обработка осуществляется PTS, то микроконтроллер может выполнять передачи, не выходя из режима Idle. При этом во время передачи выводы управления шиной к внешней памяти принимают свои активные состояния.

Если микроконтроллер был выведен из режима Idle стандартной процедурой обработки прерываний, то после завершения подпрограммы обслуживания прерывания ЦПУ выбирает и затем выполняет команду, которая следует за командой MOD #1.

23.2 Режим PowerDown

Режим функционирования с очень низким потреблением энергии. Все сигналы внутреннего тактирования зафиксированы в состоянии логического нуля, генератор отключен, блок АЦП переведен в режим Sleep (по умолчанию). Ток потребления АЦП снижается до тока утечки устройства. Если значение входного напряжения сохраняется, то регистры специальных функций и регистры ОЗУ сохраняют данные.

Выводы управления шиной переведены в неактивное состояние. Все выходы сохраняют значения, находящиеся в их защелках.

Запрещение режима PowerDown

Режим запрещен, если во время конфигурации микроконтроллера (старт или сброс) вывод A0 был подключен к земле.

Переход в режим PowerDown

До перехода в режим обязательно должен быть завершен обмен через последовательные порты, и завершены все аналоговые преобразования. В противном случае возможна потеря данных. Кроме того, если сторожевой таймер разрешен, его регистр должен быть очищен, чтобы устройство смогло выйти из режима PowerDown «чисто», иначе сторожевой таймер может сбросить микроконтроллер до стабилизации тактового генератора. В режиме PowerDown счетчик сторожевого таймера не меняет своего значения.

Переход в режим PowerDown осуществляется по команде MOD #2.

Примечание – На выводе NMI должен удерживаться низкий уровень сигнала, пока устройство находится в режиме PowerDown.

Выход из режима PowerDown

Выход из режима PowerDown обеспечивается одним из двух способов:

1 Удержание низкого уровня сигнала на выводе VPP в течение, как минимум 50 нс, приведет к разблокированию внутреннего генератора и возобновлению тактирования микроконтроллера. Выполнение программы возобновится с команды, следующей за командой MOD #2.

2 Удержание низкого уровня сигнала на выводе RESET до тех пор, пока генератор не стабилизируется. Схема генератора должна характеризоваться определенным временем наступления стабильной генерации. Когда используется внешнее тактирование, RESET должен удерживаться в низком уровне, по крайней мере, один такт. Микроконтроллер выйдет из режима PowerDown, произойдет сброс, и начнется выполнение программы с начального адреса.

23.3 Режим Slow

Режим, в котором тактовая частота работы микроконтроллера снижена в два раза, т. е. составляет 1/4 внешнего сигнала тактирования на выводе XTAL1. Режим включается установкой бита SLOW в регистре CLKEN.

24 Программно-аппаратные средства отладки

Для создания программного обеспечения рекомендуется использовать программный продукт CodeMaster++[28].

Программный продукт CodeMaster++[28] представляет собой набор программно-аппаратных средств, предназначенный для разработки и отладки систем на базе микроконтроллеров 1874ВЕ10Т, 1874ВЕ10АТ АО «НИИЭТ».

Адаптер JEM-96 обеспечивает взаимодействие между интегрированной средой разработки CodeMaster++[28], установленной на персональном компьютере, и отладочными ресурсами, встроенными в микроконтроллеры 1874ВЕ10Т и 1874ВЕ10АТ, а также выполнение отладочных функций.

Адаптер взаимодействует с микроконтроллером через интерфейс JTAG.

Адаптер JEM-96 обеспечивает отладочные функции:

- сброс микроконтроллера с последующим остановом программы пользователя;
- запуск на выполнение программы пользователя в режиме реального времени;
- останов программы пользователя в произвольный момент времени;
- определение текущего адреса выполнения программы пользователя при останове;
- изменение текущего адреса выполнения программы пользователя в останове;
- четыре точки останова по адресу выполнения программы;
- шаг низкого уровня (выполнение одной инструкции микроконтроллера) с заходом в подпрограммы;
- шаг низкого уровня без захода в подпрограммы;
- шаг высокого уровня (выполнение программы по строкам исходного кода на языке С) с заходом в подпрограммы;
- шаг высокого уровня без захода в подпрограммы;
- запись и чтение памяти данных;
- запись и чтение памяти регистров специального назначения.

В зависимости от выбранного интерфейса отладки, контакты разъема адаптера JEM-96 подключаются к микроконтроллеру в соответствии с таблицей 24.1.

Таблица 24.1 – Интерфейс отладки JTAG

Номер вывода JEM-96	Название контакта адаптера JEM-96	Вывод микроконтроллера
3	DCS – разрешение работы по SPI	Не используется
4	DBG# – выбор стартового адреса в памяти программ	TRST
5	DIN/TDI – вход последовательных данных	TDI
7	NMI/TMS – прерывание пользовательской программы в произвольный момент времени	TMS
9	DCLK/TCK – тактовая частота	TCK
11	DOUT/TDO – выход последовательных данных	TDO
1, 2	VCCtrg – напряжение питания +3,3 В	VDD
13	RST# – сброс	RESET
6, 8, 10, 12, 14	GND – общий	GND
Примечание – Отладка по JTAG возможна при активном уровне сигнала на входе DEBUG_EN.		

24.1 Модуль отладки OCDS

Модуль OCDS предназначен для упрощения процесса отладки программного обеспечения пользователя, а также для контроля хода выполнения программы. Блок формирует несколько типов откликов по нескольким типам событий. События могут быть следующие:

- обращение к адресам внешней памяти;
- появление на шине адресов заданного адреса операнда;
- появление на шине данных заданного слова;
- появление на шине данных заданного младшего байта;
- появление на шине данных заданного старшего байта;
- превышение значения основного счетчика команд заданной величины;
- точное совпадение значения основного счетчика команд с заданной величиной.

При обнаружении одного или нескольких указанных событий может формироваться отклик (задается в регистре DEBCTRL независимо для каждого события):

- прерывание DEBUG (INT52);
- аппаратный сброс микроконтроллера.

Для задания адресов, байт/слов данных и значений счетчика команд используются ряд регистров системы отладки.

Регистры DEBPCEQ1 и DEBPCEQ2

Регистры DEBPCEQ1 и DEBPCEQ2 используются, если требуется отследить момент, когда счетчик команд достигнет желаемого значения. Как только значение счетчика команд и значение в регистре DEBPCEQ1 и/или DEBPCEQ2 совпадут, это будет считаться событием. Отклик задается полями PCEQ1 и PCEQ2.

Регистр DEBADDR

Регистр DEBADDR используется, если требуется отследить появление на шине адресов желаемого значения. Как только значение слова адреса операнда на шине и значение в регистре совпадут, это будет считаться событием. Отклик задается полем ADDR.

Регистр DEBDAT

Регистр DEBDAT используется, если требуется отследить появление на внутренней шине данных желаемого значения. Как только значение слова данных на шине и значение, записанное в регистре, совпадут, это будет считаться событием. Отклик задается полем DAT.

Регистр DEBPC

Регистр DEBPC используется, если требуется отследить момент, когда счетчик команд превысит желаемое значение. Как только значение счетчика команд станет равным или больше значения, записанного в регистре DEBPC, это будет считаться событием. Отклик задается полем PC.

Важно помнить, что значение счетчика команд на момент возникновения события указывает адрес команды, следующей за выполняемой, и может отличаться от реального хода выполнения программы (при наличии перехода). При защите от сбоев не стоит настраивать регистр DEBPC на адрес, располагающийся непосредственно за последней командой в адресном пространстве (обычно команда перехода), так как возникновение этого значения в блоке основного счетчика команд не является ошибкой (но перехода по нему не будет).

Обращение к внешней памяти

Если требуется отследить момент, когда происходит выборка команды из области внешней памяти, то для этого нужно запрограммировать отклик посредством поля EXTACCESS (по умолчанию значение поля равно 00b и обращение к внешней памяти не воспринимается как событие).

24.2 Функционирование модуля отладки и управление его работой

По умолчанию состояние регистра DEBCTRL равно 0000h, что запрещает любые действия модуля отладки. Логика модуля OCDS начинает отслеживать желаемые события сразу же после того, как для этого события будет запрограммирован отклик.

Для наглядности на рисунке 24.1 представлена структурная схема логики модуля OCDS.

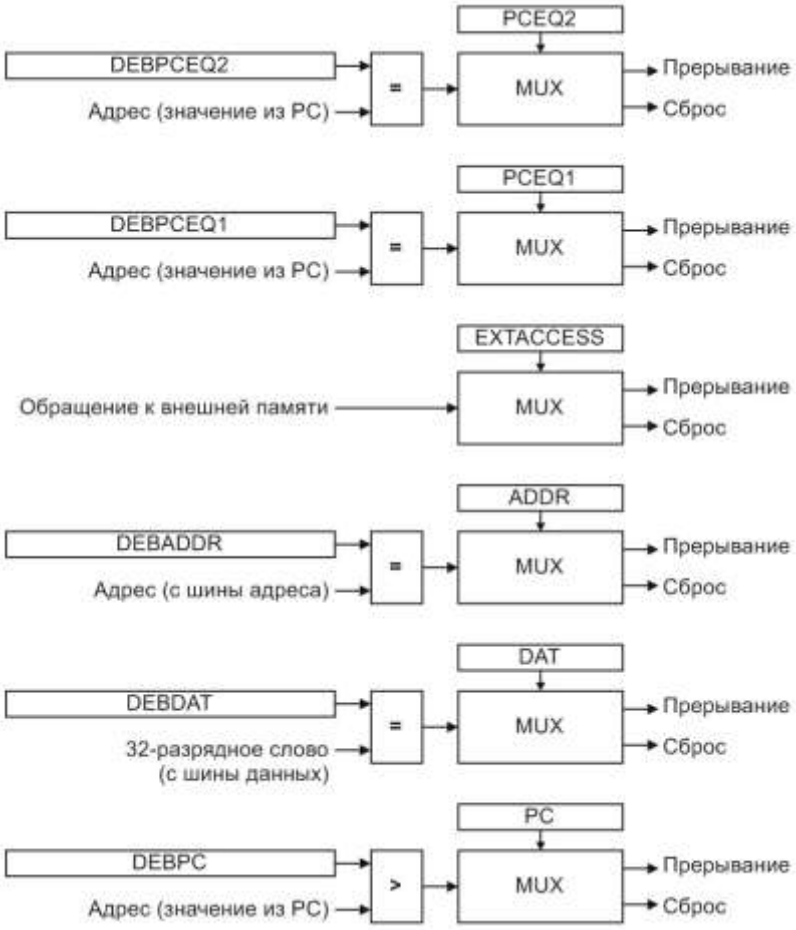


Рисунок 24.1 – Структурная схема работы модуля отладки

Шесть линий откликов «Прерывание» объединяются по ИЛИ и формируют, таким образом, одну линию прерывания DEBUG, показанную на рисунке 24.2. Аналогично объединяются между собой линии «Сброс».

Время отклика на запрос определяется так же, как и для стандартных прерываний.

Регистр HAPPPC

Из схемы на рисунке 24.2 видно, что при обнаружении любого события, независимо от запрограммированного отклика, будет происходить запись текущего состояния счетчика команд PC в регистр HAPPPC. Особенностью регистра является то, что он не сбрасывается во время сброса микроконтроллера.

Регистр CURRPC

Значение регистра CURRPC всегда совпадает со значением счетчика команд.

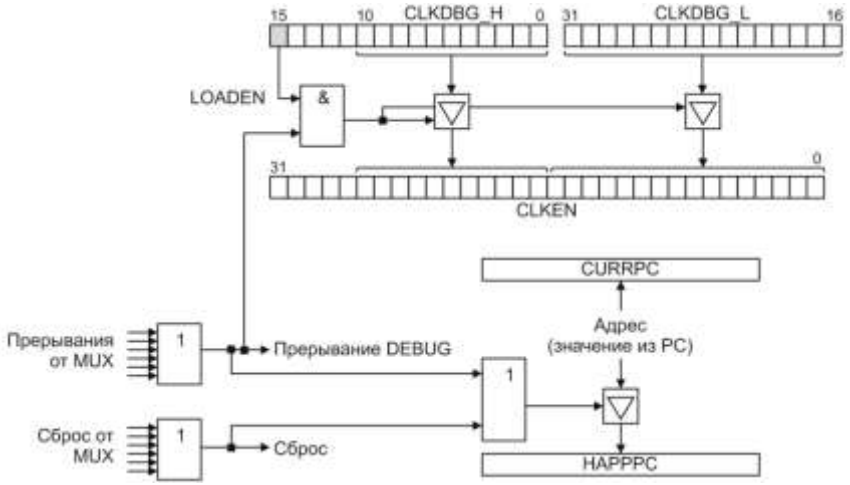


Рисунок 24.2 – Дополнительная структурная схема работы модуля отладки

Управление синхронизацией периферийных устройств

Одновременно с возникновением прерывания DEBUG может осуществляться перезапись регистра CLKEN (регистр управления тактовыми сигналами периферийных устройств) при условии, что это разрешено битом LOADEN регистра CLKDBG_H. Это позволяет отключать/включать периферию. При этом управляющие регистры и регистры состояний будут сохранять свои значения, которые были на момент отключения и будут доступны только для чтения.

Биты старшего слова регистра CLKDBG_L соответствуют младшему слову регистра CLKEN, а младшие 11 бит регистра CLKDBG_H – битам с 16 по 26 регистра CLKEN.

Заключение

В настоящем КФДЛ.431295.059ТО было представлено описание архитектуры, функционального построения и периферии микроконтроллеров 1874ВЕ10Т, 1874ВЕ10АТ. Техническое описание может служить практическим руководством по применению микроконтроллеров для разработчиков систем на его основе и программистов.

Приложение А (обязательное) Регистры микроконтроллеров

В таблицах А.1.1 – А.1.7, А.2.1, А.2.2, А.3.1, А.4.1 – А.4.5, А.5.1 – А.5.4, А.6.1 – А.6.4, А.7.1 – А.7.5, А.8.1, А.8.2, А.9.1 – А.9.8, А.10.1 – А.10.10, А.11.1 – А.11.6, А.12.1 – А.12.3, А.13.1 – А.13.8, А.14.1 – А.14.6, А.15.1 – А.15.11, А.16.1 – А.16.33, А.17.1 – А.17.6, А.18.1 – А.18.8 приведено описание регистров с указанием их мнемонических обозначений, разрядности, форматов, назначений и назначений их битов. Информация об адресах регистров представлена в приложении Б настоящего технического описания.

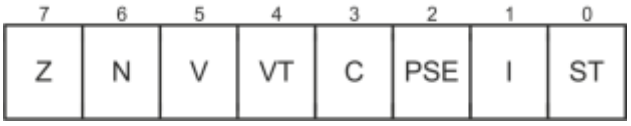
А.1 Системные регистры

Таблица А.1.1 – Системные регистры общего назначения

Мнемоническое обозначение	Доступ к регистру	Описание	Сброс
ZERO_REG	Чтение Запись	32-разрядный регистр нуля	0h
R0 – R15	Чтение Запись	32-разрядные регистры данных быстрых команд	0h
DPP0, DPP1	Чтение Запись	32-разрядные регистры указателей страниц в адресном пространстве. (Для 16- и 32-разрядного режима)	0h
DPP2, DPP3	Чтение Запись	32-разрядные регистры указателей страниц в адресном пространстве. (Только для 16- и 32-разрядного режима)	0h

Таблица А.1.2 – Регистр слова состояния программы

PSW (не доступен программно) Сброс: --h



Поле	Бит	Описание
Z	7	Флаг нуля. Устанавливается, если результат операции равен нулю. При операциях сложения с переносом и вычитания с заемом этот флаг никогда не устанавливается, но он сбрасывается, если результат не нулевой. Таким образом, флаг нулевого результата показывает наличие нулевого или ненулевого результата при вычислениях с большой точностью
N	6	Флаг отрицательного результата. Устанавливается, если после выполнения операции получен отрицательный результат. Флаг правильный, если даже случилось переполнение. При всех сдвиговых операциях и операции нормализации этот флаг устанавливается равным старшему значащему биту результата, даже если счётчик сдвигов равен нулю

Продолжение таблицы А.1.2

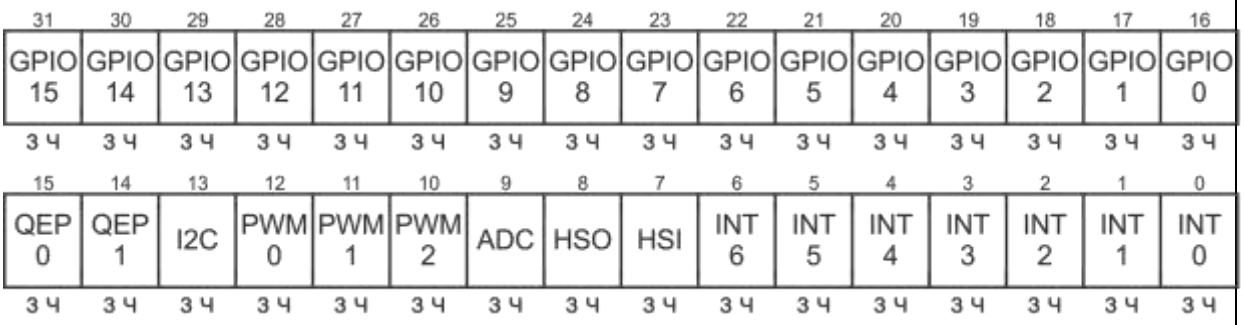
<table>
<tr><th>Поле</th><th>Бит</th><th>Описание</th></tr>
<tr><td>V</td><td>5</td><td>Флаг переполнения.
Устанавливается, если после выполнения операции получен результат, выходящий за границы диапазона значений типа данных приемника.
В 16-разрядном режиме флаг устанавливается:
- при сдвиговых операциях (SHL, SHLB, SHLL), если старший значащий бит операнда изменился в процессе сдвига;
- при операциях деления, если при выполнении команды:
- DIVUB результат > FFh;
- DIVU результат > FFFFh;
- DIVB результат < 81h (– 127) или > 7Fh (+127);
- DIV результат < 8001h (-32767) или > 7FFFh (+32767).

В 32-разрядном режиме флаг устанавливается:
- при сдвиговых операциях (SHL, SHLB, ESHLL), если старший значащий бит операнда изменился в процессе сдвига;
- при операциях деления, если при выполнении команды:
- DIVUB результат > FFh;
- EDIVU результат > FFFF_FFFFh;
- DIVB результат < 81h (-127) или > 7Fh (+127);
- EDIV результат < 8000_0001h (–2147483647) или > 7FFF_FFFFh (+2147483647).</td></tr>
<tr><td>VT</td><td>4</td><td>Дополнительный флаг переполнения.
Устанавливается, когда флаг переноса C установлен.
Очищается только командами CLRVT, JVT, JNVT. Это позволяет тестировать возможное состояние переполнения в конце последовательности родственных арифметических операций, что более эффективно, чем тестирование флага переполнения после любой операции</td></tr>
<tr><td>C</td><td>3</td><td>Флаг переноса.
Устанавливается, для индикации арифметического переноса из старшего разряда ALU или состояние, когда последний значащий бит выдвигается за пределы операнда. Если вычитание содержит заём, флаг переноса сбрасывается.
Если значение сдвинутых битов:
– < 1/2 LSB, то C = 0;
– ≥ 1/2 LSB, то C = 1.
Обычно результат округляется до большего значения при установленном флаге переноса. Флаг ST позволяет улучшить точность округления.
Если значение сдвинутых битов:
– 0 и бит ST = 0, то C = 0;
– > 0 и < 1/2 LSB и ST = 1, то C = 0;
– = 1/2 LSB и ST = 0, то C = 1;
– > 1/2 LSB и < 1 LSB и ST = 1, то C = 1</td></tr>
<tr><td>PSE</td><td>2</td><td>Бит разрешения PTS.
Устанавливается командой EPTS. Сбрасывается командой DPTS.
Разрешает или запрещает работу сервера периферийных транзакций
<table>
<tr><td>0</td><td>Запрещено</td></tr>
<tr><td>1</td><td>Разрешено (для обслуживания прерываний должен быть установлен бит I)</td></tr>
</table></td></tr>
</table>

Окончание таблицы А.1.2

Поле	Бит	Описание
I	1	Бит глобального разрешения прерываний. Устанавливается командой EI. Сбрасывается командой DI. Разрешает или запрещает обслуживание всех прерываний, кроме прерываний NMI.
		0 Запрещено
		1 Разрешено
ST	0	Дополнительный флаг переноса (после операции умножения имеет неопределенное состояние). Устанавливается, чтобы показать, что во время сдвига вправо, был установлен флаг переноса С, а затем сброшен (т. е. через перенос прошла единица). Флаг ST может использоваться вместе с флагом переноса С для более точного округления.

Таблица А.1.3 – Младшие регистры прерываний



PTSSEL0
Регистр разрешения прерываний PTS.
Установка бита разрешает обслуживание соответствующего прерывания блоком PTS

PTSSRV0
Регистр флагов прерываний.
Установленный бит указывает на прерывание END-OF-PTS. Сбрасывается аппаратно, одновременно с началом обслуживания прерывания

INT_MASK0
Регистр масок прерываний.
Установка бита разрешает обслуживание соответствующего прерывания

INT_PEND0
Регистр ждущих прерываний.
Установленный бит указывает на прерывание ожидающее обслуживания. Сбрасывается аппаратно, одновременно с началом обслуживания прерывания (если ждущих прерываний несколько, то приоритет имеет прерывание с наибольшим номером)

Поле	Бит	Описание	Номер прерывания
GPIO15	31	Прерывание порта по линии 15	INT31
GPIO14	30	Прерывание порта по линии 14	INT30
GPIO13	29	Прерывание порта по линии 13	INT29
GPIO12	28	Прерывание порта по линии 12	INT28
GPIO11	27	Прерывание порта по линии 11	INT27
GPIO10	26	Прерывание порта по линии 10	INT26
GPIO9	25	Прерывание порта по линии 9	INT25

Окончание таблицы А.1.3

Поле	Бит	Описание	Номер прерывания
GPIO8	24	Прерывание порта по линии 8	INT24
GPIO7	23	Прерывание порта по линии 7	INT23
GPIO6	22	Прерывание порта по линии 6	INT22
GPIO5	21	Прерывание порта по линии 5	INT21
GPIO4	20	Прерывание порта по линии 4	INT20
GPIO3	19	Прерывание порта по линии 3	INT19
GPIO2	18	Прерывание порта по линии 2	INT18
GPIO1	17	Прерывание порта по линии 1	INT17
GPIO0	16	Прерывание порта по линии 0	INT16
QEP0	15	Прерывание квадратурного декодера QEP0	INT15
QEP1	14	Прерывание квадратурного декодера QEO1	INT14
I2C	13	Прерывание контроллера I2C	INT13
PWM0	12	Прерывание блока ШИМ0	INT12
PWM1	11	Прерывание блока ШИМ1	INT11
PWM2	10	Прерывание блока ШИМ2	INT10
ADC	9	Прерывание блока АЦП	INT9
HSO	8	Прерывание блока HSO	INT8
HSI	7	Прерывание блока HSI	INT7
INT6-INT0	6-0	Зарезервировано	INT6 – INT0
Примечание – Для всех регистров значение после сброса 00h.			

Таблица А.1.4 – Старшие регистры прерываний

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NMI	TRAP 1	TRAP	UN IMP	ECC ERR DATA	ECC ERR COM	ECC ERR RAM	ECC ERR PSRAM	SPE ERR OVER	SPE ERR DOWN	ERR WORD ALIGN	DE BUG	WDT	SYS TIM 0	SYS TIM 1	FPU TRAP
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
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SP WR0	SP WR1	UART 0	UART 1	UART 2	UART 3	SPI 0	SPI 1	BSI 0	BSI 1	ECC BSI0	ECC BSI1	AR INC 0_1	AR INC 2_3	AR INC 4_5	AR INC 6_7
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

PTSSEL1
Регистр разрешения прерываний PTS.
Установка бита разрешает обслуживание соответствующего прерывания блоком PTS

PTSSRV1
Регистр флагов прерываний.
Установленный бит указывает на прерывание END-OF-PTS. Сбрасывается аппаратно, одновременно с началом обслуживания прерывания

INT_MASK1
Регистр масок прерываний.
Установка бита разрешает обслуживание соответствующего прерывания

INT_PEND1
Регистр ждущих прерываний.
Установленный бит указывает на прерывание ожидающее обслуживания. Сбрасывается аппаратно, одновременно с началом обслуживания прерывания (если ждущих прерываний несколько, то приоритет имеет прерывание с наибольшим номером)

Продолжение таблицы А.1.4

Поле	Бит	Описание	Номер прерывания
NMI	31	Немаскируемое внешнее прерывание NMI	INT63
TRAP1	30	Прерывание TRAP1	INT62
TRAP	29	Прерывание TRAP	INT61
UNIMP	28	Прерывание при обнаружении невыполнимого кода операции	INT60
ECCERRDATA	27	Прерывание при обнаружении более одного ошибочного разряда в данных (используется код Хемминга)	INT59
ECCERRCOM	26	Прерывание при обнаружении более одного ошибочного разряда в коде команды (используется код Хемминга)	INT58
ECCERRRAM	25	Прерывание при обнаружении более одного ошибочного разряда в данных при обращении к ОЗУ (используется код Хемминга)	INT57
ECCERRPSRAM	24	Прерывание при обнаружении более одного ошибочного разряда в данных при обращении к PSRAM (используется код Хемминга)	INT56
SPEERROVER	23	Прерывание при переполнении стека	INT55
SPEERRDOWN	22	Прерывание при опустошении стека	INT54
ERRWORDALIGN	21	Прерывание при указании некорректного адреса регистра	INT53
DEBUG	20	Прерывание при отладке	INT52
WDT	19	Прерывание сторожевого таймера	INT51
SYSTIM0	18	Прерывание системного таймера T0	INT50
SYSTIM1	17	Прерывание системного таймера T1	INT49
FPUTRAP	16	Прерывание блока вычислений с плавающей запятой (блок FPU)	INT48
SPWR0	15	Прерывание контроллера SpaceWire0	INT47
SPWR1	14	Прерывание контроллера SpaceWire1	INT46
UART0	13	Прерывание приемопередатчика UART0	INT45
UART1	12	Прерывание приемопередатчика UART1	INT44
UART2	11	Прерывание приемопередатчика UART2	INT43
UART3	10	Прерывание приемопередатчика UART3	INT42
SPI0	9	Прерывание контроллера SPI0	INT41
SPI1	8	Прерывание контроллера SPI1	INT40
BSI0	7	Прерывание контроллера МПИ0	INT39
BSI1	6	Прерывание контроллера МПИ1	INT38
ECCBSI0	5	Прерывание при обнаружении более одного ошибочного разряда в данных при обращении контроллера МПИ0 к собственному ОЗУ (используется код Хемминга)	INT37
ECCBSI1	4	Прерывание при обнаружении более одного ошибочного разряда в данных при обращении контроллера МПИ1 к собственному ОЗУ (используется код Хемминга)	INT36

Окончание таблицы А.1.4

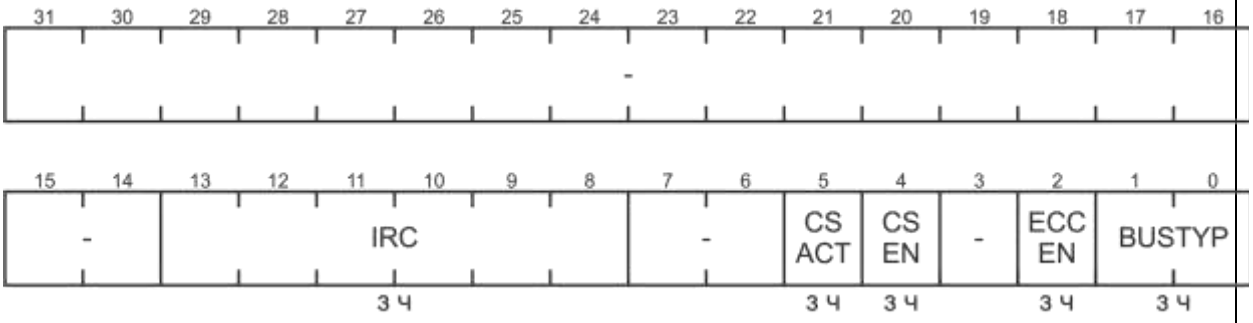
Поле	Бит	Описание	Номер прерывания
ARINC0_1	3	Прерывание контроллеров ARINC0 и ARINC1 (объединены по ИЛИ)	INT35
ARINC2_3	2	Прерывание контроллеров ARINC2 и ARINC3 (объединены по ИЛИ)	INT34
ARINC4_5	1	Прерывание контроллеров ARINC4 и ARINC5 (объединены по ИЛИ)	INT33
ARINC6_7	0	Прерывание контроллеров ARINC6 и ARINC7 (объединены по ИЛИ)	INT34
Примечание – Для всех регистров значение после сброса 00h.			

Таблица А.1.5 – Регистры окна x (x – номер окна от 0 до 7)

Мнемоническое обозначение	Доступ к регистру	Описание	Сброс
BUSxADDRLO	Чтение Запись	32-разрядный регистр нижней границы адресного окна x	0h
BUSxADDRHI	Чтение Запись	32-разрядный регистр верхней границы адресного окна x	0h
Примечание – Для регистра BUS7ADDRHI значение после сброса FFFF_FFFFh.			

Таблица А.1.6 – Регистр управления шиной окна x (x – от 0 до 7)

BUSxCON Сброс: 0h



Поле	Бит	Описание	
IRC	13-8	Поле задания количества циклов ожиданий (машинных циклов) при READY = 0	
		00h – 3Eh	0 тактов – 62 такта
		3Fh	Управление сигналом READY
CSACT	5	Бит задания активного уровня сигнала выбора внешнего устройства	
CSEN	4	Бит разрешения сигнала выбора внешнего устройства	
		0	Запрещен
		1	Разрешен
ECCEN	2	Бит включения кодирования кодом Хемминга	
		0	Выключено
		1	Включено
BUSTYP	1, 0	Конфигурация внешней шины	
		00b	8-разрядная
		01b	16-разрядная
		10b	32-разрядная
		11b	Зарезервировано
–	31-14, 7, 6	Зарезервировано	

Таблица А.1.7 – Регистры адреса некорректных данных и команд, кодированных кодом Хемминга

Мнемоническое обозначение	Доступ к регистру	Описание	Сброс
ECCDATAERR	Чтение Запись	32-разрядный регистр адреса некорректных данных. Если при чтении кодированных данных были обнаружены ошибки как минимум в двух разрядах, то в регистр записывается адрес расположения этих данных	0h
ECCCOMERR	Чтение Запись	32-разрядный регистр адреса некорректной команды. Если при чтении кодированной команды были обнаружены ошибки как минимум в двух разрядах, то в регистр записывается значение счетчика команд	0h
SP	Чтение Запись	32-разрядный регистр указателя стека (Указатель стека)	0000_ 3D18h
SP_OVRFLOW	Чтение Запись	32-разрядный регистр переполнения указателя стека. Задает значение, по достижении которого указателем стека (при наполнении стека) формируется прерывание	FFFF_ FFF0h
SP_DNFFLOW	Чтение Запись	32-разрядный регистр опустошения указателя стека. Задает значение, по достижении которого указателем стека (при опустошении стека) формируется прерывание	0000_ 000Fh

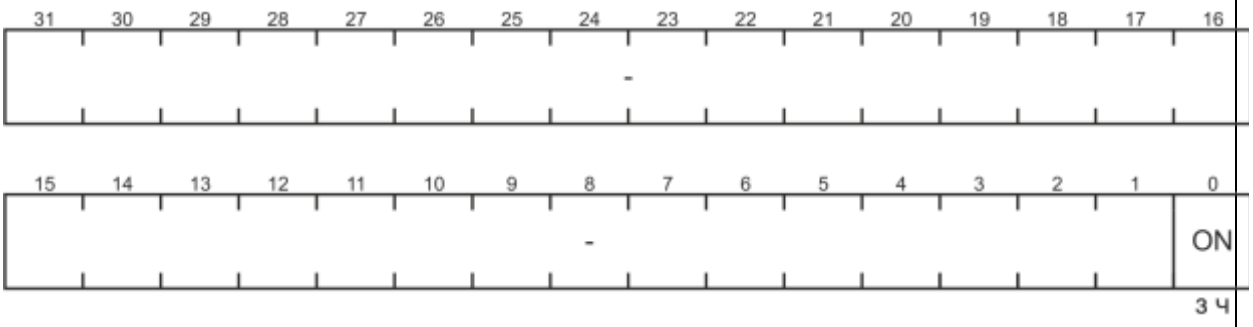
A.2 Регистры таймеров T0 и T1

Таблица А.2.1 – Регистры таймеров T0 и T1

Мнемоническое обозначение	Доступ к регистру	Описание	Сброс
VALUE_L	Чтение Запись	32-разрядный регистр младшего слова значения системного таймера	0h
VALUE_H	Чтение Запись	32-разрядный регистр старшего слова значения системного таймера	0h
RELOAD_L	Чтение Запись	32-разрядный регистр младшего слова значения перезагрузки системного таймера. Задает младшую часть 64-разрядного числа, по достижении которого таймер обнуляется.	
RELOAD_H	Чтение Запись	32-разрядный регистр старшего слова значения перезагрузки системного таймера. Задает старшую часть 64-разрядного числа, по достижении которого таймер обнуляется.	

Таблица А.2.2 – Регистр включения системного таймера

CTRL Сброс: 0h



Поле	Бит	Описание	
ON	0	Бит включения системного таймера	
		0	Выключен
		1	Включен
–	31-1	Зарезервировано	

A.3 Регистры векторов прерываний

Таблица А.3.1 – Регистр вектора прерывания x (x – номер прерывания от 0 до 63)

Мнемони-ческое обозначение	Доступ к регистру	Описание	Сброс
INTx	Чтение Запись	32-разрядный регистр вектора прерывания x. Содержит адрес начала подпрограммы обработки прерывания	См. таблицы 7.2, 7.3
PTSx	Чтение Запись	32-разрядный регистр вектора прерывания PTSx. Содержит адрес начала подпрограммы обработки прерывания PTS	

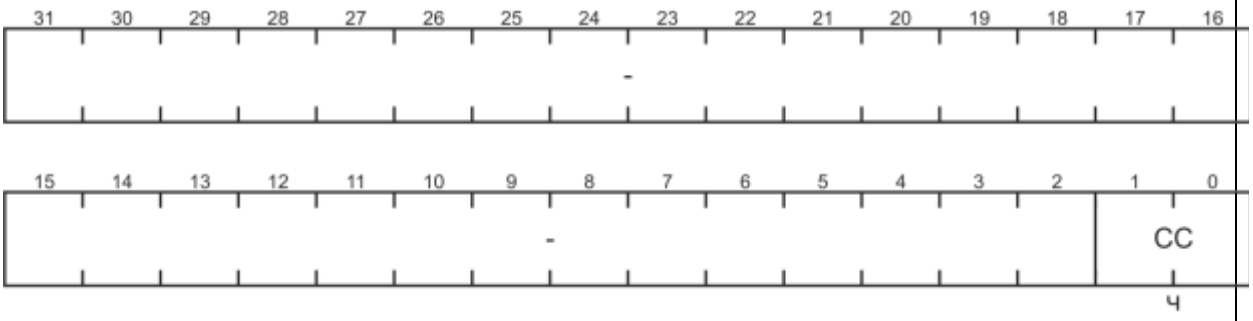
A.4 Регистры блока FPU

Таблица А.4.1 – Регистр данных (x – номер регистра от 0 до 31)

Мнемоническое обозначение	Доступ к регистру	Описание	Сброс
FRx	Чтение Запись	32-разрядный регистр данных	0h

Таблица А.4.2 – Регистр флагов операций сравнения

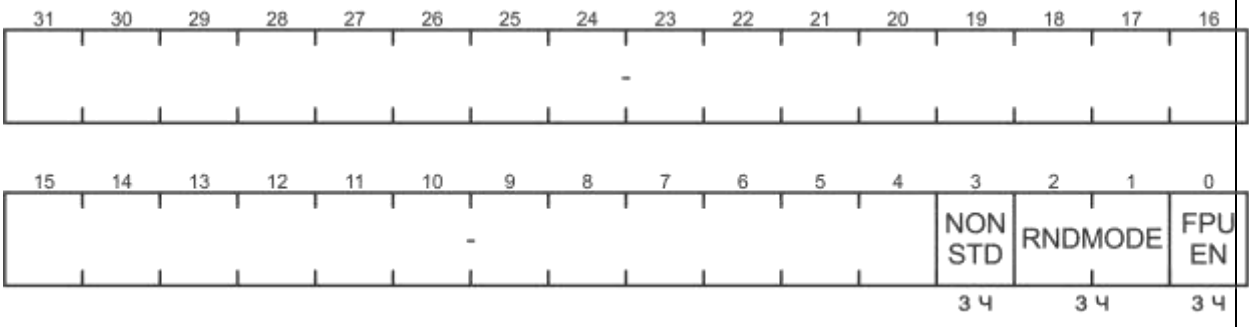
FFLAGS 0000_4380h (20h) Сброс: 0h



Поле	Биты	Описание	
CC	1, 0	Флаг результата операции	
		00	Равенство операндов
		01	Операнд 1 (SRC1) < операнд 2 (SRC2)
		10	Операнд 2 < операнд 1
		11	Неопределен (хотя бы один из операндов NaN)
–	31-2	Зарезервировано	

Таблица А.4.3 – Регистр управления

FCTRL 0000_4384h (21h) Сброс: 0h



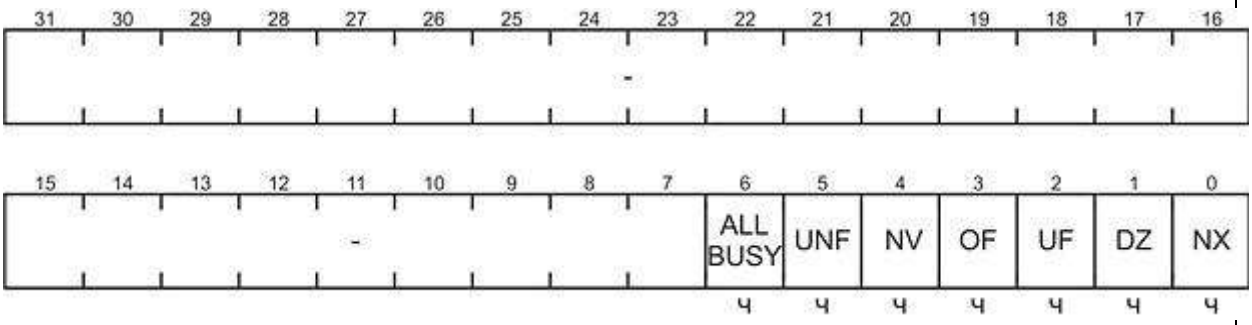
Поле	Биты	Описание	
NONSTD	3	Ненормальный режим	
		0	Ненормальный режим
		1	Денормализованный операнд преобразуется в ноль

Окончание таблицы А.4.3

Поле	Биты	Описание	
RNDMODE	2-1	Направление округления	
		00	К ближайшему целому
		01	К нулю
		10	К плюс бесконечности
		11	К минус бесконечности
FPUEN	0	Бит разрешения работы блока	
		0	Выключен
		1	Включен
–	31-4, 7-5	Зарезервировано	

Таблица А.4.4 – Регистр исключительных состояний

FEXC 0000_4388h (22h) Сброс: 0h

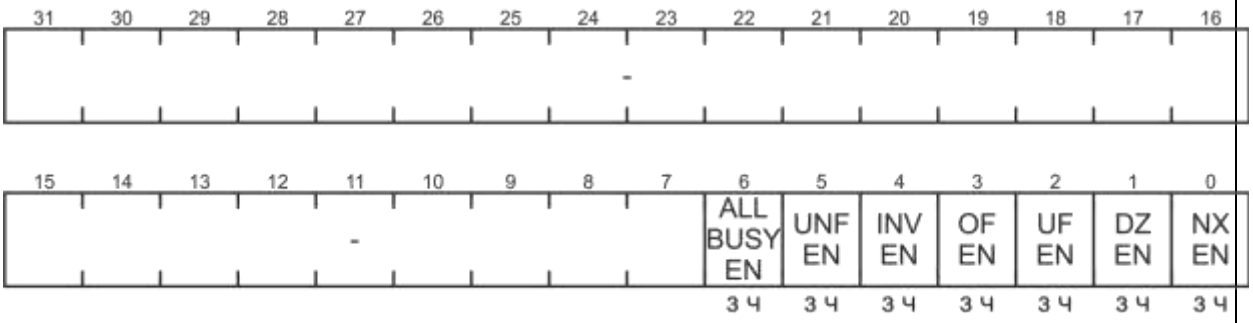


Поле	Биты	Описание
ALLBUSY	6	Флаг занятости блока вычислений
UNF	5	Неоконченная операция. Генерируется арифметической операцией или операцией преобразования при денормализованном операнде/операндах
NV	4	Некорректная операция
OF	3	Переполнение числа
UF	2	Антипереполнение
DZ	1	Деление на ноль
NX	0	Неточный результат
–	31-7	Зарезервировано

Примечание – Флаги устанавливаются при возникновении соответствующего события. Установка флагов и генерирование прерываний разрешаются регистром FTRAP. Флаги аппаратно сбрасываются по окончании обслуживания прерываний.

Таблица А.4.5 – Регистр разрешения прерываний по исключительным состояниям

FTRAP 0000_438Ch (23h) Сброс: h



Поле	Биты	Описание
ALLBUSYEN	6	Разрешение флага занятости
UNFEN	5	Разрешение прерывания «Неоконченная операция»
INVEN	4	Разрешение прерывания «Некорректная операция»
OFEN	3	Разрешение прерывания «Переполнение числа»
UFEN	2	Разрешение прерывания «Антипереполнение»
DZEN	1	Разрешение прерывания «Деление на ноль»
NXEN	0	Разрешение прерывания «Неточный результат»
–	31-7	Зарезервировано

Примечание – Для разрешения прерывания следует установить соответствующий бит.

A.5 Регистры портов (GPIO)

Таблица А.5.1 – 32-разрядные регистры портов A и B

Мнемоника	Описание	Сброс
DATAIN	Регистр входных данных порта (только чтение)	0h
DATAOUT	Регистр выходных данных порта	0h
QPAD	Регистр значения временного интервала	0h

32-разрядные регистры, представленные в таблице А.5.2, управляют выводами порта и доступны для записи и чтения. В таблице А.5.2 для каждого регистра приведено описание одного его бита. Функционирование всех битов регистра идентично.

8-разрядные порты PA0 – PA3 объединены в 32-разрядный порт A, а 8-разрядные порты PB0 – PB3 объединены в 32-разрядный порт B, см. рисунок А.5.1. Каждому выводу 32-разрядного порта соответствует один бит регистра – нулевому выводу соответствует нулевой бит, 31 выводу – 31 бит.

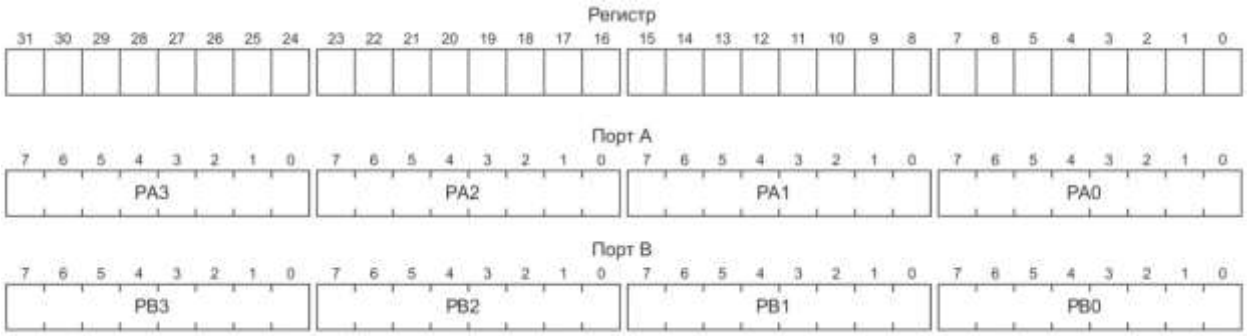


Рисунок А.5.1 – Соответствие выводов портов и битов регистров управления

Таблица А.5.2 – Регистры портов (канала)

Общий формат регистров управления выводами порта

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16
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

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

Мнемоническое обозначение регистра и его назначение	Работа вывода в зависимости от состояния соответствующего ему бита	
DATALOCK Регистр защиты битов регистра DATAOUT от записи	0	Бит регистра DATAOUT доступен для записи
	1	Бит регистра DATAOUT не доступен для записи
OUTEN Регистр выбора направления работы порта	0	Вывод работает как вход
	1	Вывод работает как выход

Продолжение таблицы А.5.2

Мнемоническое обозначение регистра и его назначение	Работа вывода в зависимости от состояния соответствующего ему бита	
ALTEN Регистр включения альтернативной функции	0	Альтернативная функция отключена
	1	Альтернативная функция включена
ODCTRL Регистр типа выхода порта	0	Открытый коллектор
	1	Push-Pull
PULLEN Регистр включения подтяжки вывода	0	Подтяжка отключена
	1	Подтяжка включена
PULLMODE Регистр типа подтяжки вывода	0	Pull-Down
	1	Pull-Up
SE Регистр выбора фильтрованного входного сигнала	0	В схему подается не фильтрованный входной сигнал
	1	В схему подается сигнал с выхода цифрового фильтра
QE Регистр выбора типа фильтрации входного сигнала	0	Простая пересинхронизация
	1	Пересинхронизация с фильтрацией помех
QM Регистр параметров фильтрации помех	0	По трем отсчетам
	1	По шести отсчетам
INTPOL Регистр выбора уровня/фронта входного сигнала	0	Уровень логического нуля/задний фронт
	1	Уровень логической единицы/передний фронт
INTTYPE Регистр выбора типа ожидаемого события	0	Уровень сигнала
	1	Фронт сигнала
INTEN Регистр разрешения генерирования прерывания	0	Запрещено
	1	Разрешено
INTSTAT Регистр флагов прерываний	0	Нет прерывания или бит уже сброшен. Бит сбрасывается только программно, записью единицы в соответствующий бит регистра INTCLR
	1	Если генерирование прерывания разрешено (соответствующим битом регистра INTEN), то: - при обнаружении ожидаемого события будет сгенерировано прерывание, и установлен флаг; - пользователь может сгенерировать прерывание программно, записью единицы в регистр INSTAT. Если генерирование прерывания запрещено, то: - при обнаружении ожидаемого события не будет сгенерировано прерывание, и флаг не установится;

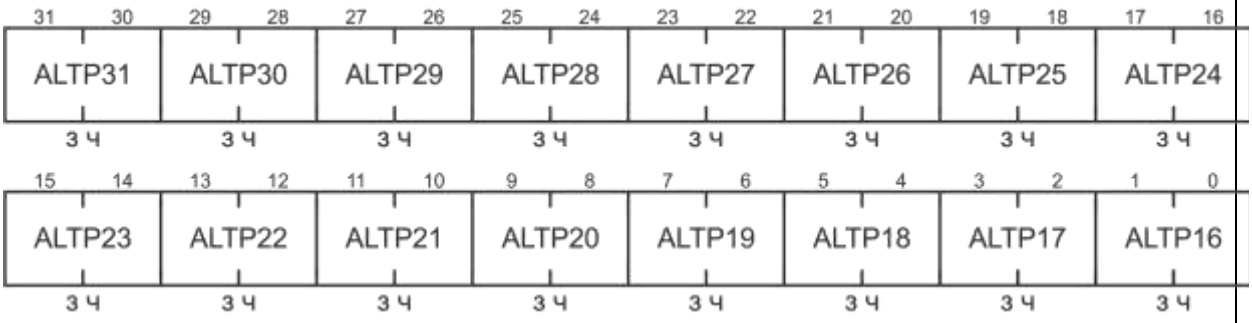
		- программная запись единицы в регистр INSTAT установит флаг, но прерывание не будет сгенерировано

Окончание таблицы А.5.2

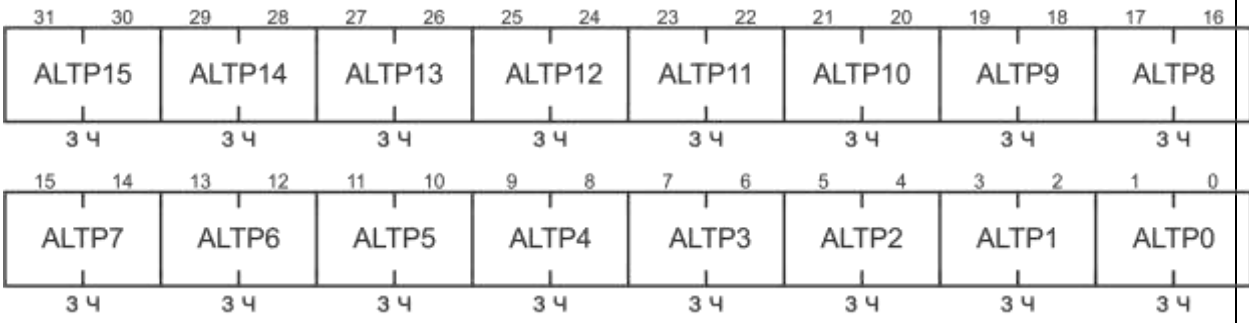
Мнемоническое обозначение регистра и его назначение	Работа вывода в зависимости от состояния соответствующего ему бита	
INTCLR Регистр сброса флагов прерываний (только запись)	0	Нет действий
	1	Запись единицы сбросит соответствующий флаг в регистре INTSTAT
CFGLOCK Регистр глобальной защиты конфигурации выводов	0	Защита отключена
	1	Установка бита запрещает изменение состояний, соответствующих данному выводу управляющих битов в регистрах: OUTEN, ALTEN, ODCTRL, PULLEN, PULLMODE, SE, QE, QM, INTPOL, INTTYPE, INTEN, ALTCTRL, INTLINE
Примечание – Состояние всех регистров после сброса – 0h.		

Таблица А.5.3 – Регистры выбора альтернативных функций выводов

ALTCTRL1 Сброс: 0h



ALTCTRL0 Сброс: 0h



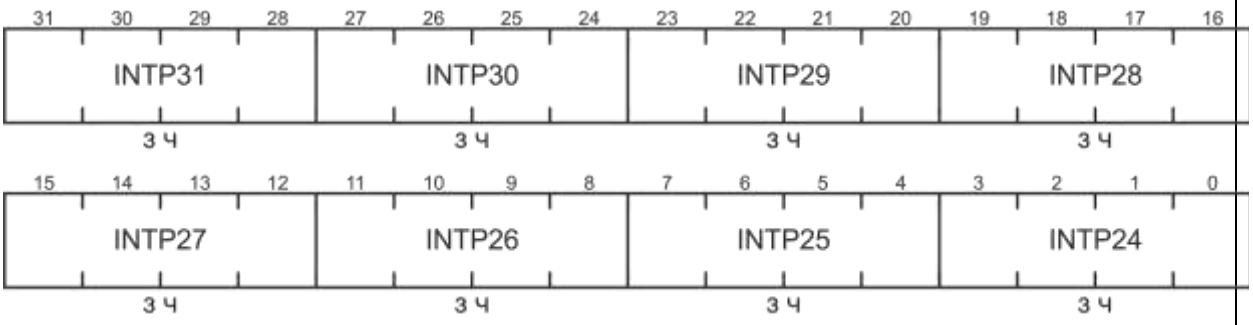
Поле ALTPn задает номер альтернативной функции n-ого вывода порта. Для включения альтернативной функции следует установить соответствующий n-й бит в регистре ALTEN.

ALTPn	Порядковый номер альтернативной функции
00	Первая
01	Вторая
10	Третья

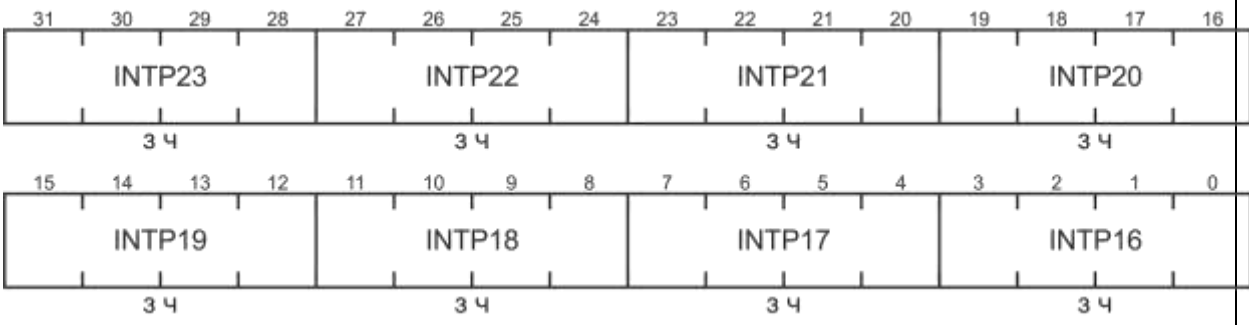
11	Зарезервировано. Не использовать!

Таблица А.5.4 – Регистры выбора линий прерываний для выводов

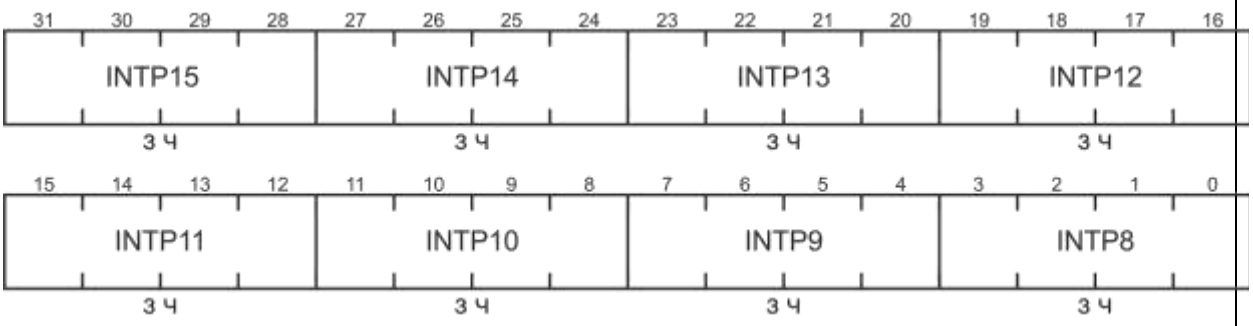
INTLINE3 Сброс: 0h



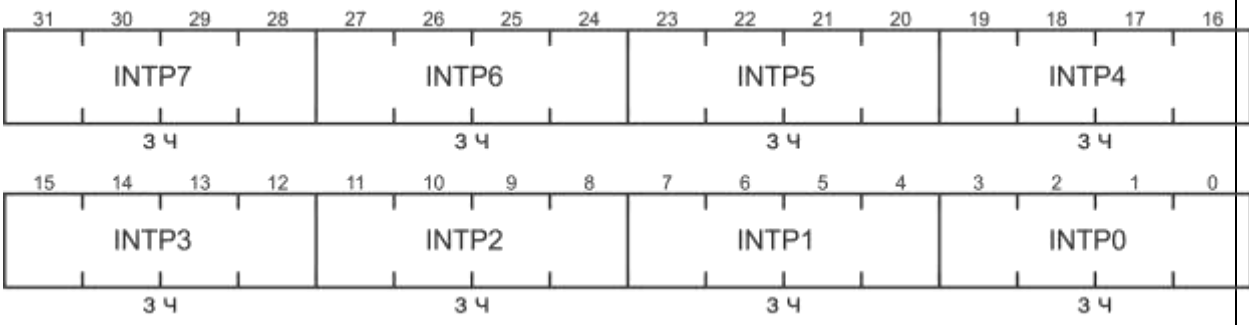
INTLINE2 Сброс: 0h



INTLINE1 Сброс: 0h



INTLINE0 Сброс: 0h



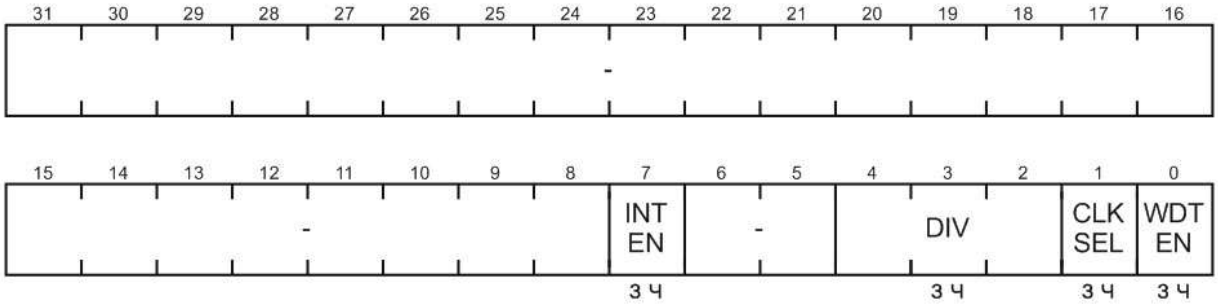
Окончание таблицы А.5.4

Поле INTPn задает линию прерывания для n-ого вывода порта	
INTPn	Прерывание (см. таблицы 7.2 и 7.3)
0h	INT16
1h	INT17
2h	INT18
3h	INT19
4h	INT20
5h	INT21
6h	INT22
7h	INT23
8h	INT24
9h	INT25
Ah	INT26
Bh	INT27
Ch	INT28
Dh	INT29
Eh	INT30
Fh	INT31

A.6 Регистры сторожевого таймера

Таблица А.6.1 – Регистр управления

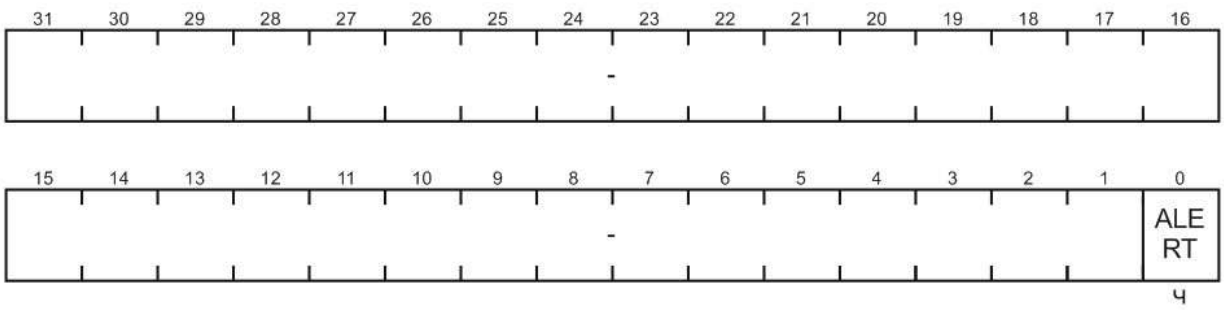
CTRL Сброс: 0000_0000h



Поле	Бит	Описание	
INTEN	7	Бит разрешения генерирования прерывания сторожевого таймера	
		0	Запрещено
		1	Разрешено
DIV	4–2	Коэффициент делителя частоты синхросигнала счетчика	
		000	Делитель выключен
		001	1/2
		010	1/4
		011	1/8
		100	1/16
		101	1/32
		110	1/64
		111	1/128
CLKSEL	1	Бит выбора источника синхросигнала счетчика	
		0	Синхросигнал микроконтроллера SysCLK
		1	Внешний синхросигнал с вывода 30 микроконтроллера
WDTEN	0	Бит включения сторожевого таймера (после установки бит обнуляется только при сбросе микроконтроллера)	
		0	Выключен
		1	Включен
–	31-8, 6, 5	Зарезервировано	

Таблица А.6.2 – Регистр состояния

STAT Сброс: 0000_0000h

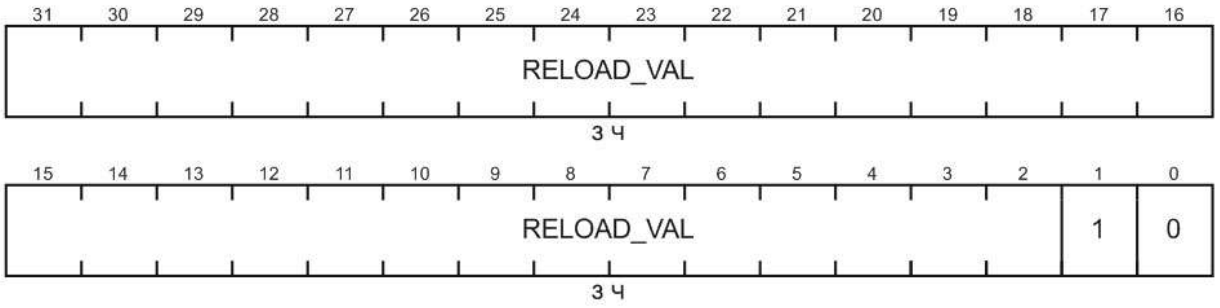


Окончание таблицы А.6.2

Поле	Бит	Описание
ALERT	0	Флаг предупреждения от сторожевого таймера. Устанавливается каждый раз, когда счетчик достигает значения нуля. Флаг сбрасывается последовательной записью значений E1h, 1Eh в регистр CLR (0000_4D80h). Если в момент обнуления счетчика флаг ALERT установлен, сторожевой таймер генерирует сигнал сброса микроконтроллера
–	31-1	Зарезервировано

Таблица А.6.3 – Регистр перезагрузки

RELOAD Сброс: 0000_FFFFh



Поле	Бит	Описание
RELOAD_VAL	31-0	Значение перезагрузки счетчика сторожевого таймера (минимальное значение 2h). Это значение загружается в счетчик сторожевого таймера после его включения, при обнулении счетчика (при условии, что флаг ALERT сброшен) и при последовательной записи значений E1h, 1Eh в регистр CLR

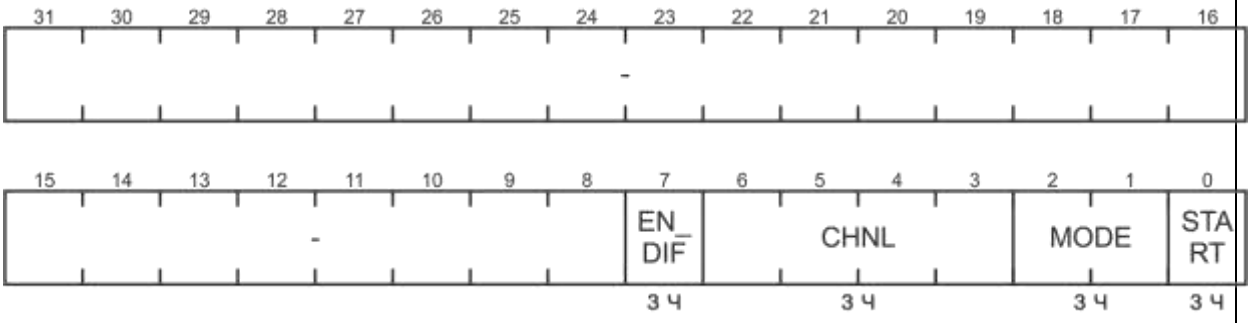
Таблица А.6.4 – Регистры счетчика

Мнемоническое обозначение	Доступ к регистру	Назначение и описание	Сброс
CNT	Чтение	32-разрядный регистр текущего значения счетчика	FFFFFFFFh
CLR	Запись	32-разрядный регистр сброса сторожевого таймера. Таймер сбрасывается последовательной записью значений E1h, 1Eh в этот регистр.	00000000h

A.7 Регистры блока АЦП

Таблица А.7.1 – Регистр управления

CON Сброс: 0h

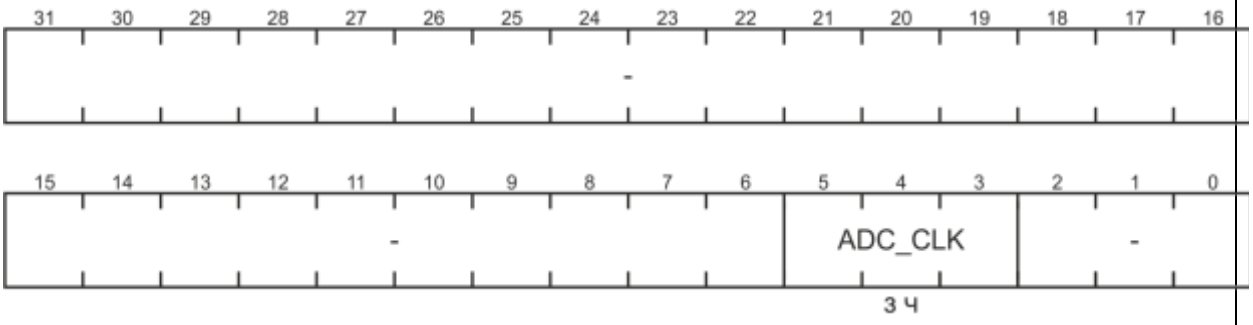


Поле	Бит	Описание	
EN_DIF	7	Бит выбора режима входов АЦП	
		0	Однополярное включение
		1	Дифференциальное включение
CHNL	6-3	Поле задания номера канала	
		0000	Канал 0
		0001	Канал 1
		0010	Канал 2
		0011	Канал 3
		0100	Канал 4
		0101	Канал 5
		0110	Канал 6
		0111	Канал 7
		1000	Канал 8
		1001	Канал 9
		1010	Канал 10
		1011	Канал 11
		1100	Канал 12
		1101	Канал 13
		1110	Канал 14

		1111	Канал 15
MODE	2, 1	Поле выбора режима работы АЦП	
		00	Единичное преобразование выбранного канала
		01	Многократное преобразование выбранного канала
		10	Последовательное сканирование каналов
		11	Зарезервировано
START	0	Бит запуска аналого-цифрового преобразования	
		0	Нет действий/остановка преобразования
		1	Запуск преобразования
–	31-8	Зарезервировано	

Таблица А.7.2 – Регистр настроек

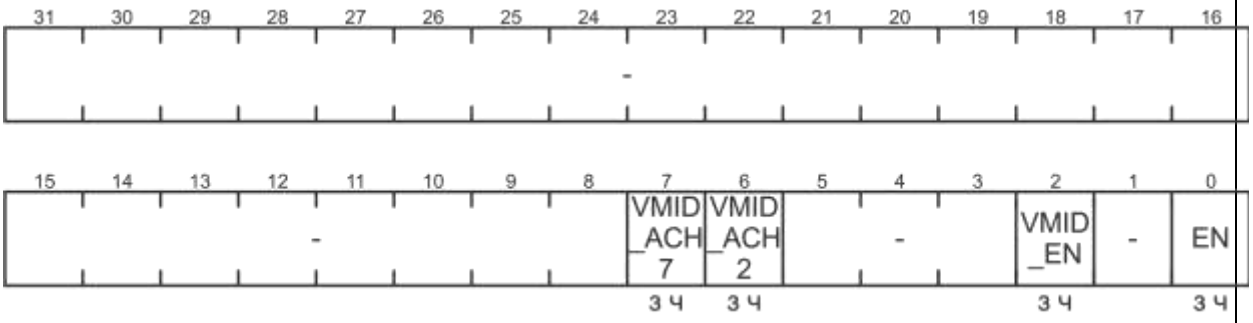
SET Сброс: 0h



Поле	Бит	Описание		
ADC_CLK	5-3	Поле коэффициента настройки частоты преобразования		
		Код	K_{CLK_ADC}	Примечание
		000	1	–
		001	1/2	–
		010	1/4	–
		011	1/8	1,5 МГц (частота XTAL1 = 24 МГц)
		100	1/16	0,75 МГц (частота XTAL1 = 24 МГц)
		101	1/32	–
		110	1/64	–
		111	1/128	–
–	31-6, 2-0	Зарезервировано		

Таблица А.7.3 – Регистр инициализации

EN Сброс: 0h



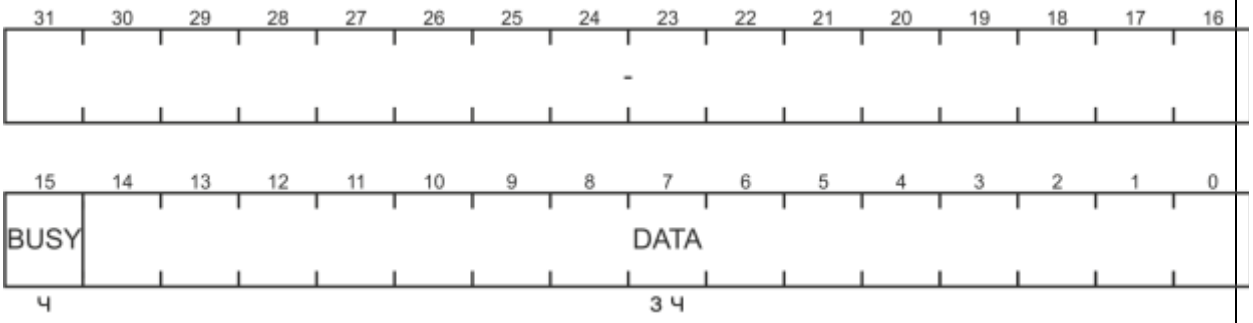
Поле	Бит	Описание	
VMID_ ACH7	7	Бит подключения внешнего источника напряжения средней точки с вывода ACH7 к цепи U_{MID} (см. рисунок 13.1)	
		0	Отключен
		1	Подключен. Биты VMID_ACH2 и VMID_EN должны быть сброшены

Окончание таблицы А.7.3

Поле	Бит	Описание	
VMID_ ACH2	6	Бит подключения внешнего источника напряжения средней точки с вывода ACH2 к цепи U_{MID} (см. рисунок 13.1)	
		0	Отключен
		1	Подключен. Биты VMID_ACH7 и VMID_EN должны быть сброшены
VMID_EN	2	Бит подключения внутреннего источника напряжения средней точки	
		0	Отключен
		1	Подключен. Биты VMID_ACH2 и VMID_ACH7 должны быть сброшены
EN	0	Бит включения АЦП	
		0	Выключен
		1	Включен
–	31-8, 5-3, 1	Зарезервировано	

Таблица А.7.4 – Регистр результата

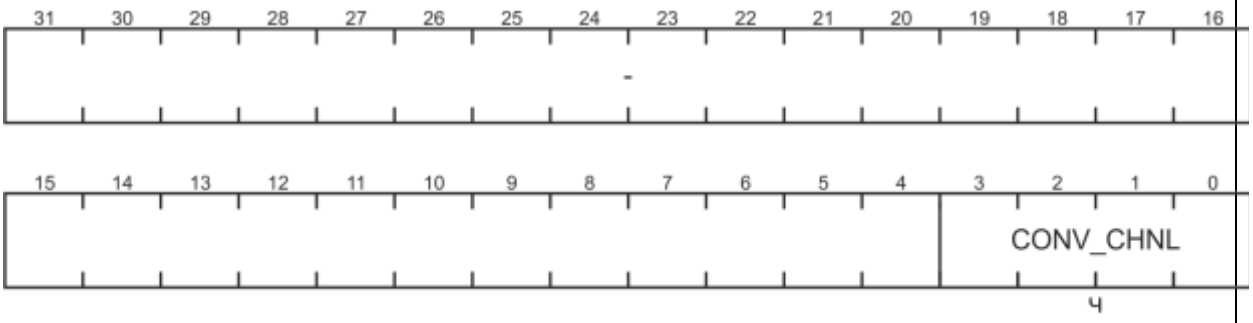
RESULT Сброс: 0h



Поле	Бит	Описание	
BUSY	15	Индикатор занятости АЦП	
		0	Нет действий. АЦП в режиме ожидания
		1	Выполняется преобразование
DATA	14-0	Поле результата преобразования	
–	31-16	Зарезервировано	

Таблица А.7.5 – Регистр номера канала

CHANNEL Сброс: 0h



Поле	Бит	Описание	
CONV_ CHNL	3-0	Поле номера канала, над которым выполняется преобразование	
		0000	Канал 0
		0001	Канал 1
		0010	Канал 2
		0011	Канал 3
		0100	Канал 4
		0101	Канал 5
		0110	Канал 6
		0111	Канал 7
		1000	Канал 8
		1001	Канал 9
		1010	Канал 10
		1011	Канал 11
		1100	Канал 12
		1101	Канал 13
		1110	Канал 14
		1111	Канал 15
–	31-4	Зарезервировано	

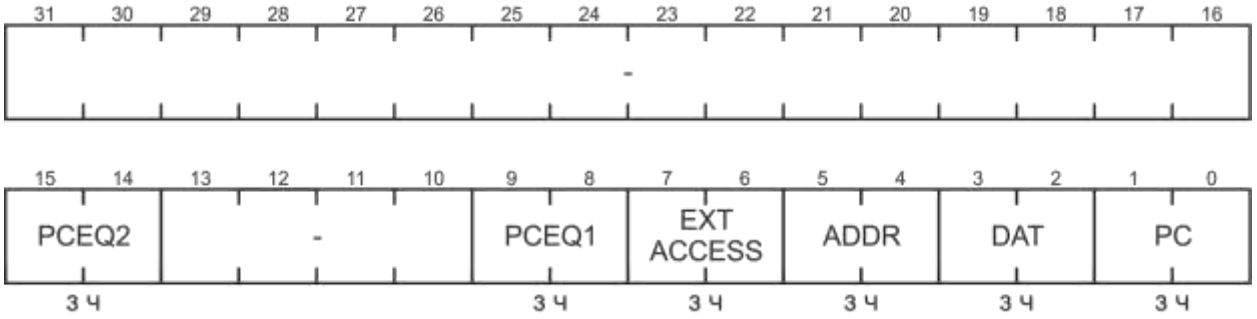
A.8 Регистры модуля OCDS

Таблица А.8.1 – 32-разрядные регистры модуля отладки

Мнемоническое обозначение	Доступ к регистру	Назначение и описание	Сброс
DEBPC	Чтение Запись	Регистр адреса блока отладки. Как только значение счетчика команд станет равным или больше значения, записанного в регистре, это будет считаться событием. Отклик задается полем PC регистра DEBCTRL.	00h
DEBDAT	Чтение Запись	Регистр значения на шине данных. Как только значение слова данных на шине и значение, записанное в регистре совпадут, это будет считаться событием. Отклик задается полем DAT регистра DEBCTRL.	00h
DEBADDR	Чтение Запись	Регистр значения на шине адреса. Как только значение слова адреса операнда на шине и значение в регистре совпадут, это будет считаться событием. Отклик задается полем ADDR регистра DEBCTRL.	00h
CURRPC	Чтение	Регистр текущего значения счетчика команд. Значение регистра всегда совпадает со значением счетчика команд.	00h
HAPPC	Чтение	Регистр состояния счетчика команд. При обнаружении любого события, независимо от запрограммированного отклика, будет происходить запись текущего состояния счетчика команд PC в регистр HAPPC. Регистр не сбрасывается во время сброса микроконтроллера.	00h
DEBPCEQ1, DEBPCEQ2	Чтение Запись	Регистры адреса. Как только значение счетчика команд и значение в регистре DEBPCEQ1 и/или DEBPCEQ2 совпадут, это будет считаться событием. Отклик задается полями PCEQ1 и PCEQ2 регистра DEBCTRL.	00h, 00h
CLKDBG_L CLKDBG_H	Чтение Запись	Регистры управления тактированием в режиме отладки. Подробное описание регистров см. в таблицах А.12.2, А.12.3.	00h, 00h

Таблица А.8.2 – Регистр управления

DEBCTRL Сброс: 0h



Поле	Бит	Описание*
PCEQ2	15, 14	Совпадение текущего значения счетчика команд со значением регистра DEBPCEQ2
PCEQ1	9, 8	Совпадение текущего значения счетчика команд со значением регистра DEBPCEQ1
EXTACCESS	7, 6	Выборка команды из области внешней памяти
ADDR	5, 4	Появление значения, заданного в регистре DEBADDR на внутренней шине адресов операндов
DAT	3, 2	Появление значения, заданного в регистре DEBDAT на внутренней шине данных
PC	1, 0	Текущее значение счетчика команд больше значения регистра DEBPC
–	31-16 13-10	Зарезервировано

* Здесь указано событие, после обнаружения которого, система отладки выполнит запрограммированное действие. Действие зависит от кода, записанного в соответствующем поле:

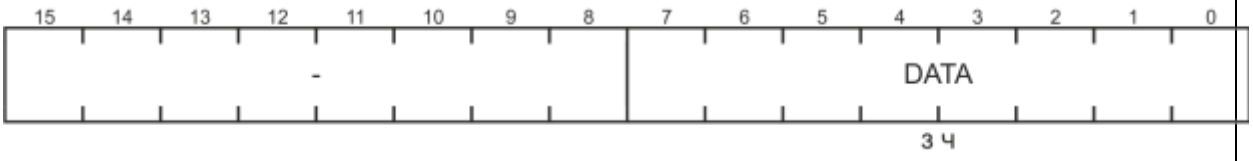
00 – нет действий; 10 – не используется;

01 – прерывание DEBUG (2060h); 11 – сброс микроконтроллера.

A.9 Регистры контроллера I2C

Таблица А.9.1 – Сдвиговый регистр данных

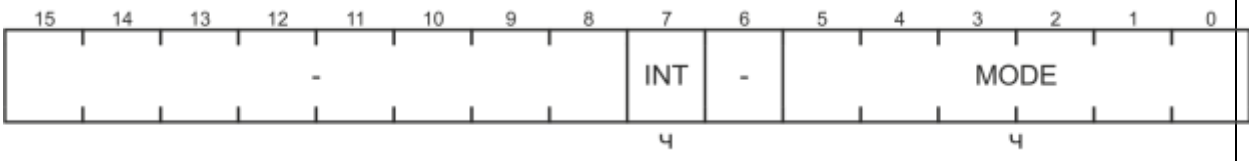
SDA Сброс: 00XXh



Поле	Биты	Описание
DATA	7-0	Поле данных
–	15-8	Зарезервировано

Таблица А.9.2 – Регистр состояния

ST Сброс: 0000h



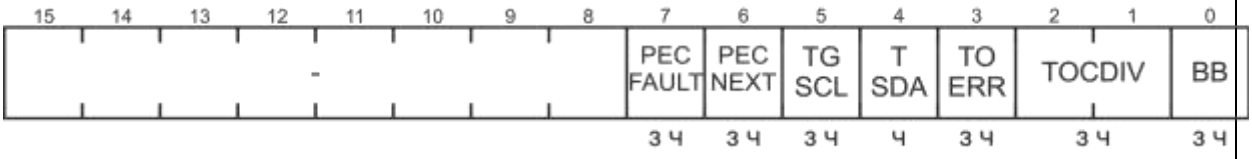
Поле	Бит	Описание
INT	7	Флаг прерывания. Устанавливается после девятого такта сигнала SCL (когда SCL = 0) в любое запрограммированное время. Условия выставления флага INT: - во время приема/передачи как в режиме мастера, так и в режиме ведомого; - при совпадении адреса (адреса ведомого, адреса отклика или адреса общего вызова) содержимое регистра SDA 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 CTL0 или выключением модуля I2C (обнуление бита ENABLE в регистре CTL1). 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.9.2

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
–	15-8, 6	Зарезервировано

Таблица А.9.3 – Регистр управления и статуса

CST Сброс: 0000h



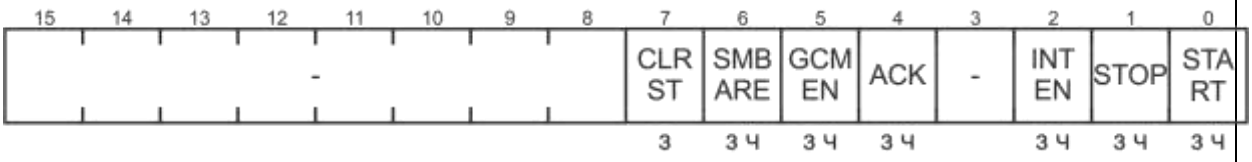
Поле	Бит	Описание
PECFAULT	7	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной суммой, значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, 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 SDA. 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 CTL0
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.9.3

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 CTL0
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
–	15-8	Зарезервировано

Таблица А.9.4 – Регистр управления 0

CTL0 Сброс: 0000h



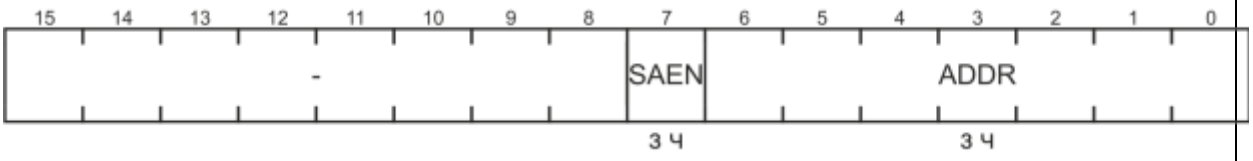
Поле	Бит	Описание
CLRST	7	Бит сброса флага прерывания INT. Запись «0» в бит CLR игнорируется. Запись «1» в бит CLR сбросит флаг INT в регистре ST. Чтение этого бита всегда возвращает «0»
SMBARE	6	Бит управления реакцией на получение адреса отклика
	0	Полученный адрес не проверяется на совпадение с адресом 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.9.4

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)
–	15-8, 3	Зарезервировано

Таблица А.9.5 – Регистр собственного адреса

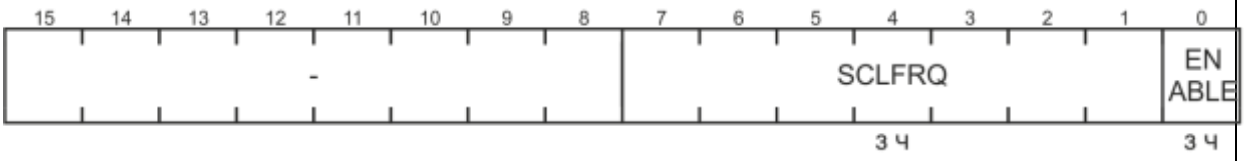
ADDR Сброс: 0000h



Поле	Бит	Описание
SAEN	7	Бит разрешения распознавания адреса
		0 Безадресный режим
		1 Включена функция распознавания принятого адреса
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#)
–	15-8	Зарезервировано

Таблица А.9.6 – Регистр управления 1

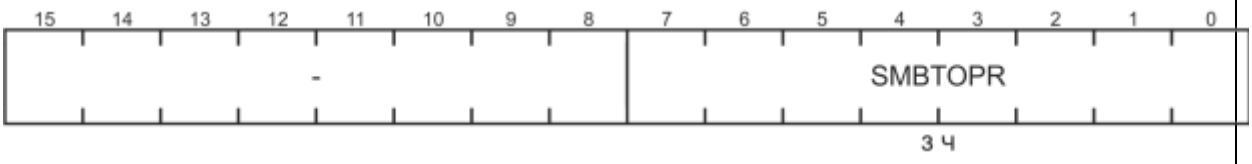
CTL1 Сброс: 0000h



Поле	Биты	Описание
SCLFRQ	7-1	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме мастера. Длительности высокого T_{SCLH} и низкого T_{SCLL} уровней сигнала SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формуле $T_{SCLH} = T_{SCLL} = 2 \times SCLFRQ \times (1/fosc)$. (A.9.1) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{SCLH} + T_{SCLL})$. (A.9.2) 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 CTL0, ST, CST сброшены
		1 Модуль включен
–	15-8	Зарезервировано

Таблица А.9.7 – Регистр загрузки предделителя

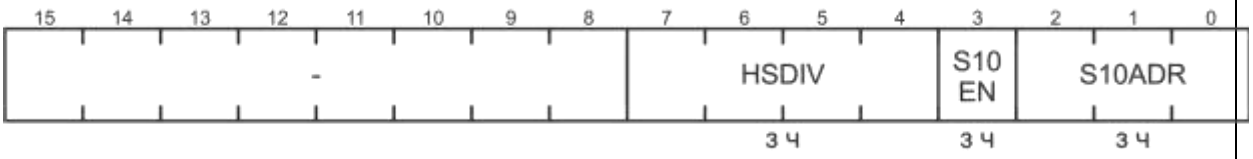
TOPR Сброс: 0000h



Поле	Биты	Описание
SMBTOPR	7-0	Поле значения перезагрузки предделителя
–	15-8	Зарезервировано

Таблица А.9.8 – Регистр управления 2

CTL2 Сброс: 0000h

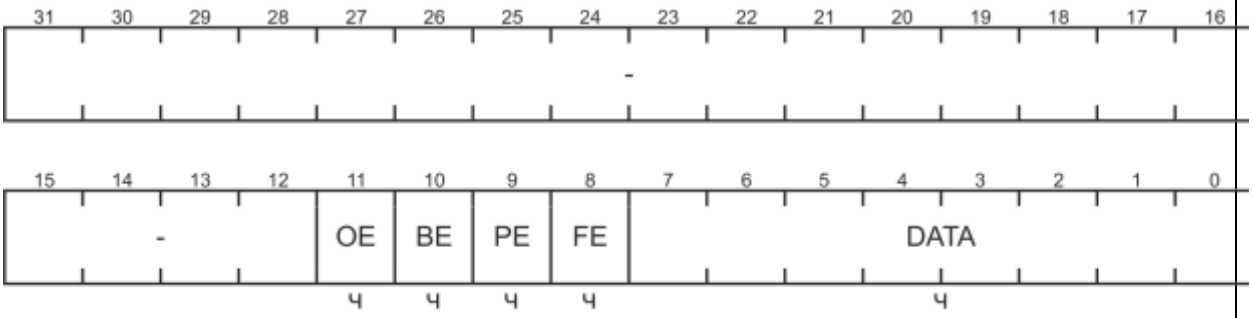


Поле	Биты	Описание	
HSDIV	7-4	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме HS мастера. Длительности высокого T_{HSCLH} и низкого T_{HSCLL} уровней сигнала на выводе SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формулам $T_{HSCLH} = HSDIV \times (1/fosc)$, (A.9.3) $T_{HSCLL} = 2 \times HSDIV \times (1/fosc)$. (A.9.4) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{HSCLH} + T_{HSCLL})$. (A.9.5) 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	Запрещено
		1	Разрешено при условии, что установлен бит SAEN в регистре ADDR
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]	
–	15-8	Зарезервировано	

A.10 Регистры блока UART

Таблица А.10.1 – Регистр данных

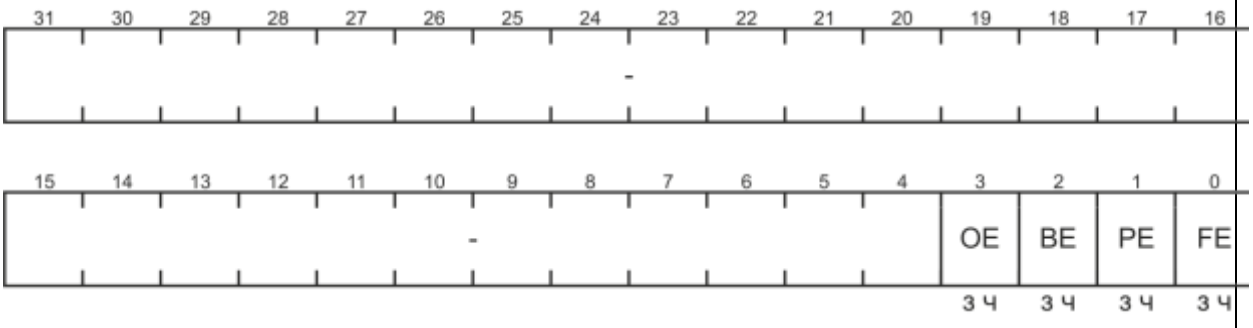
DR Сброс: 00h



Поле	Биты	Описание
OE, BE, PE, FE	11, 10, 9, 8	См. описание бит в регистре ECR (см. таблицу А.10.2)
DATA	7-0	Поле данных. Результатом записи в поле DATA является размещение байта в буфере передатчика, а результатом чтения – считывание байта из буфера приемника
–	31-12	Зарезервировано

Таблица А.10.2 – Регистр состояния приемника и сброса ошибки приемника

ECR Сброс: 00h



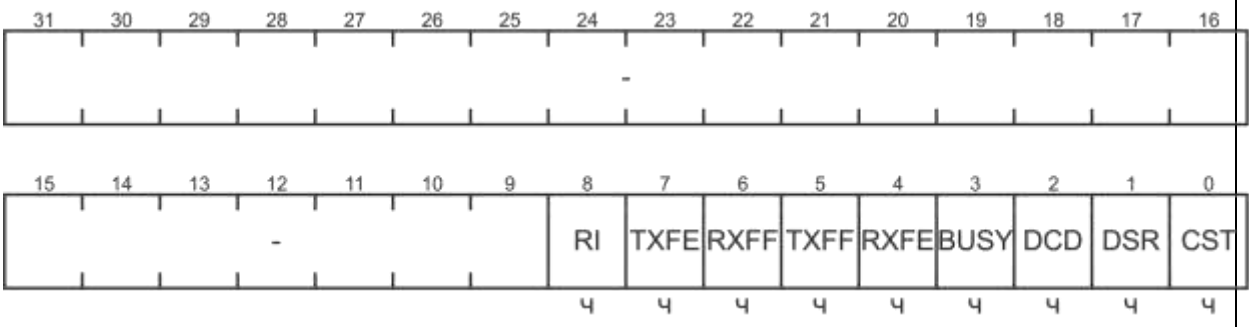
Поле	Биты	Описание	
OE	3	Флаг переполнения буфера приемника	
		0	В буфере есть свободное место или бит был сброшен после записи в регистр ECR. Содержимое буфера остается верным, так как перезаписан был только сдвиговый регистр. Центральный процессор должен считать данные для того, чтобы освободить буфер
		1	Буфер заполнен, а данные продолжают поступать

Окончание таблицы А.10.2

Поле	Биты	Описание	
BE	2	Флаг разрыва линии	
		0	Нормальная работа или бит был сброшен после записи в регистр ECR
		1	Обнаружен признак разрыва линии, то есть наличие низкого логического уровня на входе приемника в течение времени, большего, чем длительность передачи полного кадра данных (включая стартовый, стоповый биты и бит проверки на четность). При включенном режиме FIFO данная ошибка ассоциируется с последним байтом, поступившим в буфер. В случае обнаружения разрыва линии в буфер загружается только один нулевой кадр. Прием данных возобновляется только после перехода линии в логическую единицу и последующего обнаружения корректного стартового бита
PE	1	Флаг ошибки контроля четности	
		0	Нормальная работа или бит был сброшен после записи в регистр ECR
		1	Четность принятого кадра данных не соответствует установкам битов EPS и SPS в регистре управления линией LCR_H. При включенном режиме FIFO данная ошибка ассоциируется с байтом, находящимся на вершине буфера
FE	0	Флаг ошибки в структуре кадра	
		0	Нормальная работа или бит был сброшен после записи в регистр ECR
		1	В принятом символе не обнаружен корректный стоповый бит (единица). При включенном режиме FIFO данная ошибка ассоциируется с байтом, находящимся на вершине буфера
–	31-4	Зарезервировано	
Все флаги сбрасываются записью единицы			

Таблица А.10.3 – Регистр флагов

FR Сброс: 00000090h



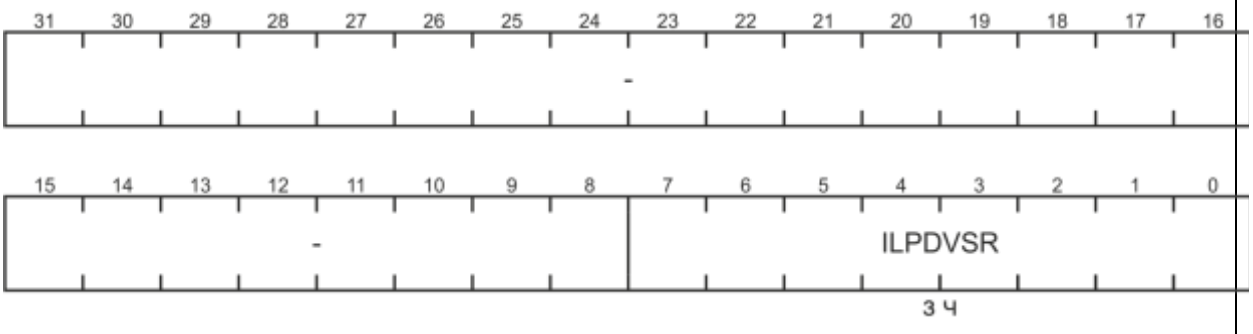
Поле	Бит	Описание	
RI	8	Инверсия линии UARTRI	
		0	Выключено
		1	Включено

Окончание таблицы А.10.3

Поле	Биты	Описание	
TXFE/ RXFE	7/ 4	Флаг пустоты буфера передатчика/приемника. Установка флага зависит от состояния бита FEN регистра LCR_H	
		0	Буфер не пуст
		1	Буфер пуст
		Примечание – Бит TXFE/RXFE не дает никакой информации о наличии данных в сдвиговом передающем регистре.	
RXFF/ TXFF	6/ 5	Флаг заполнения буфера приемника/передатчика. Установка флага зависит от состояния бита FEN регистра LCR_H (т. е. включен режим FIFO или нет)	
		0	Буфер не заполнен
		1	Если режим FIFO запрещен, бит устанавливается, когда буферный регистр приемника/передатчика занят. Если режим FIFO разрешен бит устанавливается, если заполнен буфер приемника/передатчика
BUSY	3	Бит занятости блока UART	
		0	Блок не занят
		1	Блок передает данные на линию. Бит остается установленным до тех пор, пока данные, включая стоповые биты, не будут полностью переданы. Также бит устанавливается при наличии данных в буфере передатчика, вне зависимости от состояния приемопередатчика (даже если он запрещен)
DCD	2	Индикатор инверсии сигнала на линии UART_DCD	
		0	Прямая передача
		1	Инверсия
DSR	1	Индикатор инверсии сигнала на линии UART_DSR	
		0	Прямая передача
		1	Инверсия
CST	0	Индикатор инверсии сигнала на линии UART_CTS	
		0	Прямая передача
		1	Инверсия
–	31-9	Зарезервировано	

Таблица А.10.4 – Регистр управления ИК обменом

ILPR Сброс: 00h

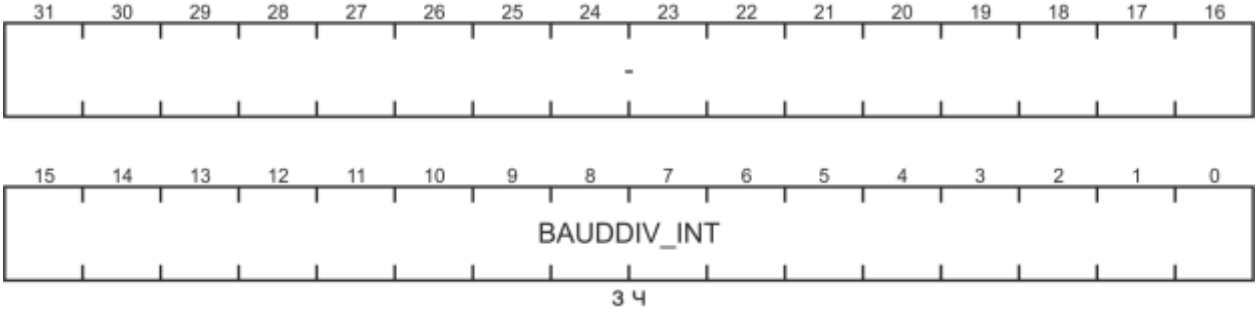


Окончание таблицы А.10.4

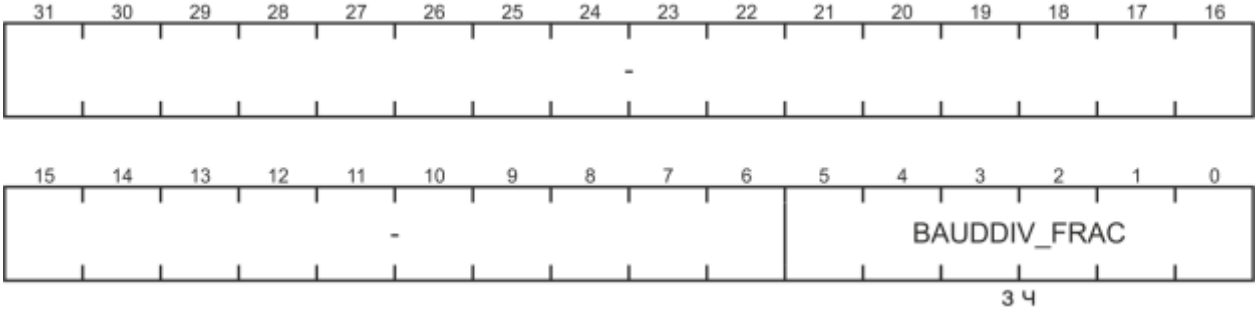
Поле	Биты	Описание
ILPDVSR	7-0	Поле задания коэффициента деления частоты для формирования тактового сигнала. 00h – запрещенное значение. При этом импульсы не формируются. Требуемое значение коэффициента деления для формирования сигнала IrLPBaud16 вычисляется по формуле: ILPDVSR = F_UARTCLK / F_IrLPBaud16, (А.10.1) где номинальное значение частоты F_IrLPBaud16 составляет 1,8432 МГц. Коэффициент деления должен быть установлен таким образом, чтобы выполнялось соотношение 1,42 МГц < F_IrLPBaud16 < 2,12 МГц, что, в свою очередь, гарантирует формирование кодеком импульсов данных с длительностью (1,41 – 2,11) мкс (в три раза длиннее периода сигнала IrLPBaud16)
–	31-8	Зарезервировано

Таблица А.10.5 – Регистры целой и дробной частей делителя скорости обмена данными

IBRD Сброс: 00h



FBRD Сброс: 00h

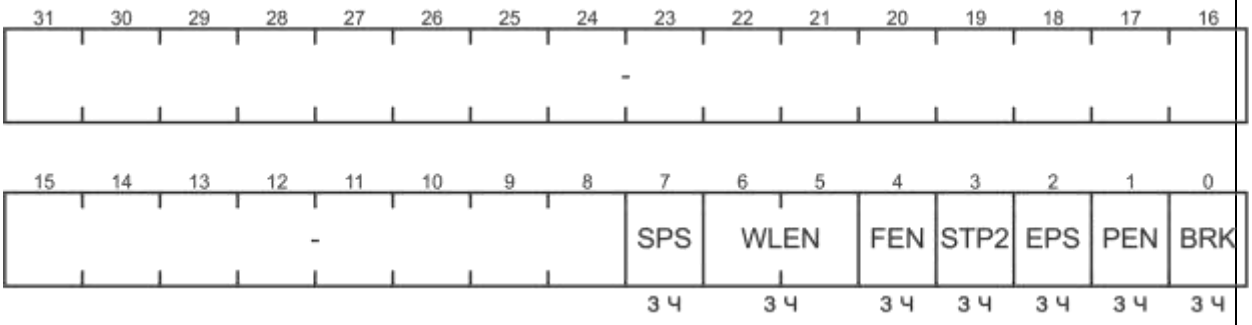


Поле	Биты	Описание
BAUDDIV_INT	15-0	Целая часть коэффициента деления частоты для формирования тактового сигнала передачи данных. Минимальное значение 0001h
BAUDDIV_FRAC	5-0	Дробная часть коэффициента деления частоты для формирования тактового сигнала передачи данных. При BAUDDIV_INT = FFFFh, значение BAUDDIV_FRAC может быть только 00h. Невыполнение этого условия приведет к прерыванию приема/передачи

Остальные биты регистров зарезервированы

Таблица А.10.6 – Регистр управления линией

LCR_H Сброс: 00h



Поле	Биты	Описание	
SPS	7	Бит разрешения передачи бита четности с фиксированным значением. Состояние бита не важно, если бит PEN сброшен	
		0	Запрещено
		1	На месте бита четности передается инверсное значение бита EPS, оно же проверяется при приеме данных. При EPS = 0 на месте бита четности передается единица. При EPS = 1 на месте бита четности передается ноль
WLEN	6, 5	Поле количества передаваемых/принимаемых информационных бит	
		00	5 бит
		01	6 бит
		10	7 бит
		11	8 бит
FEN	4	Бит включения режима FIFO буфера приемника и передатчика	
		0	Выключен
		1	Включен
STP2	3	Бит выбора режима передачи стопового бита	
		0	Один стоповый бит
		1	Два стоповых бита
		Примечание – Приемник не проверяет наличие дополнительного стопового бита в кадре.	
EPS	2	Бит паритета. Состояние бита не важно, если бит PEN сброшен	
		0	Бит четности дополняет количество единиц в информационной части кадра до нечетного числа
		1	Бит четности дополняет количество единиц в информационной части кадра до четного числа
PEN	1	Бит включения проверки четности	
		0	Выключена. Кадр не содержит бита четности
		1	Включена. Бит четности передается в кадре и проверяется при приеме данных
BRK	0	Флаг разрыва линии	
		0	Нормальная работа
		1	Если бит установлен, то по завершении передачи текущего символа на выходе передатчика устанавливается низкий уровень сигнала. Для правильного выполнения этой операции программное обеспе-чение должно обеспечить передачу сигнала разрыва в течение, как минимум, времени передачи двух информационных кадров
–	31-8	Зарезервировано	

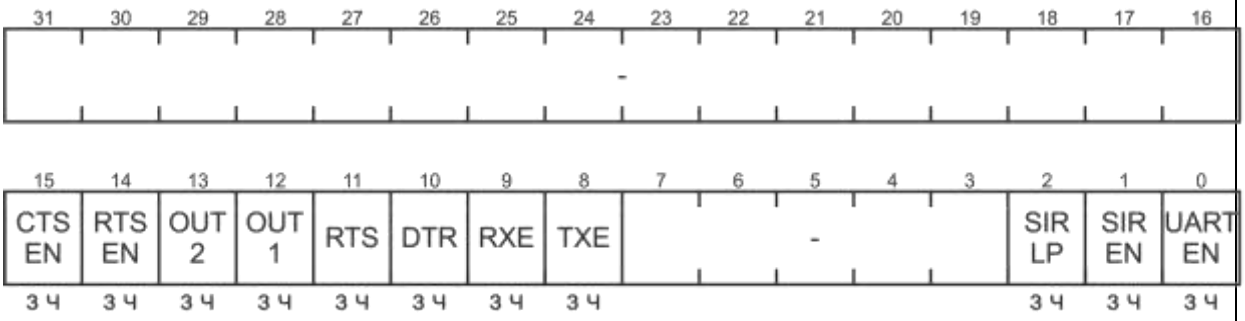
Примечание – Дополнительная информация о комбинациях состояний битов SPS, EPS и PEN для контроля паритета представлена в таблице А10.7.

Таблица А.10.7 – Зависимость бита паритета от состояния битов регистра LCR_H

Биты регистра LCR_H			Наличие и состояние бита паритета
SPS	EPS	PEN	
Не важно	Не важно	0	Не передается, не проверяется
0	0	1	Проверка нечетности слова данных
0	1	1	Проверка четности слова данных
1	0	1	Бит четности постоянно равен единице
1	1	1	Бит четности постоянно равен нулю

Таблица А.10.8 – Регистр управления

CR Сброс: 00000300h



Поле	Биты	Описание	
CTSEN	15	Бит разрешения аппаратного управления потоком данных по линии CTS	
		0	Запрещено
		1	Разрешено. Данные передаются в линию только при активном значении сигнала UART_CTS
RTSEN	14	Бит разрешения аппаратного управления потоком данных по линии RTS	
		0	Запрещено
		1	Разрешено. Запрос данных от внешнего устройства осуществляется только при наличии свободного места в буфере приемника
OUT2*	13	Бит управления статусом модема. Для подключения DTE (управляющее устройство) может использоваться в качестве RI (индикатор звонка)	
		0	Нет действий
		1	Выход UART0_OUT2 переводится в состояние логического нуля
OUT1*	12	Бит управления статусом модема. Для подключения DTE может использоваться в качестве DCD (индикатор наличия несущей частоты в линии)	
		0	Нет действий
		1	Выход UART0_OUT1 переводится в состояние логического нуля
RTS*	11	Бит программного управления состоянием линии модема UART0_RTS	
		0	Высокий уровень
		1	Низкий уровень
DTR*	10	Бит программного управления состоянием линии модема UART0_DTR	
		0	Высокий уровень
		1	Низкий уровень

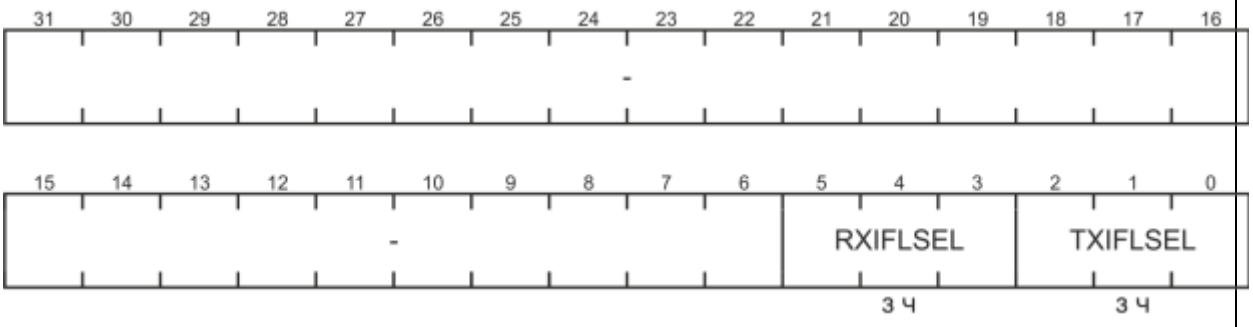
RXE	9	Бит разрешения приема. Прием данных осуществляется либо по интерфейсу асинхронного последовательного обмена, либо по интерфейсу ИК обмена SIR, в зависимости от значения бита SIREN	
		0	Запрещено
		1	Разрешено

Окончание таблицы А.10.8

Поле	Биты	Описание	
TXE	8	Бит разрешения передачи. Передача осуществляется либо по интерфейсу асинхронного последовательного обмена, либо по интерфейсу ИК обмена SIR, в зависимости от значения бита SIREN. В случае перевода приемопередатчика в запрещенное состояние в ходе передачи данных, он завершает	
		0	Запрещено
		1	Разрешено
SIRLP	2	Выбор режима ИК обмена с пониженным энергопотреблением:	
		0	0 – длительность импульсов данных равна 3/16 длительности передачи бита
		1	1 – длительность импульсов данных равна трем тактам сигнала IrLPBaud16 вне зависимости от выбранной скорости передачи данных. Выбор этого режима снижает энергопотребление, однако может привести к уменьшению дальности связи
SIREN	1	Разрешение работы кодека ИК передачи данных IrDASIR:	
		0	0 – запрещено. Сигнал nSIROUT находится в низком состоянии, данные на входе SIRIN не обрабатываются
		1	1 – разрешено. Данные передаются на выход nSIROUT и принимаются с входа SIRIN. Линия UARTTXD находится в высоком состоянии. Данные на входе UARTRXD и линиях состояния модема не обрабатываются. В случае если UARTEN=0, значение бита не играет роли
UARTEN	0	Бит разрешения работы приемопередатчика	
		0	Запрещено. Перед остановкой завершается текущая передача
		1	Разрешено
–	31-16, 7-3	Зарезервировано	
Знаком «*» отмечены биты управления модемом доступные только для блока UART0.			

Таблица А.10.9 – Регистр порога прерывания по заполнению буфера в режиме FIFO

IFLS Сброс: 00000012h



Поле	Биты	Описание		
RXIFLSEL/ TXIFLSEL	5-3/ 2-0	Порог заполнения/опустошения буфера приемника/передатчика, по достижении которого будет генерироваться прерывание		
		Код	Для приемника	Для передатчика
		000	Заполнение на 1/8	Опустошение до 1/8

Окончание таблицы А.10.9

Поле	Биты	Описание		
RXIFLSEL/ TXIFLSEL	5-3/ 2-0	001	Заполнение на 1/4	Опустошение до 1/4
		Код	Для приемника	Для передатчика
		010	Заполнение на 1/2 (по умолчанию)	Опустошение до 1/2 (по умолчанию)
		011	Заполнение на 3/4	Опустошение до 3/4
		100	Заполнение на 7/8	Опустошение до 7/8
		Остальные комбинации зарезервированы		
–	31-6	Зарезервировано		

Таблица А.10.10 – Регистр прерываний

Регистр маски прерываний
IMSC Сброс: 0000000xh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

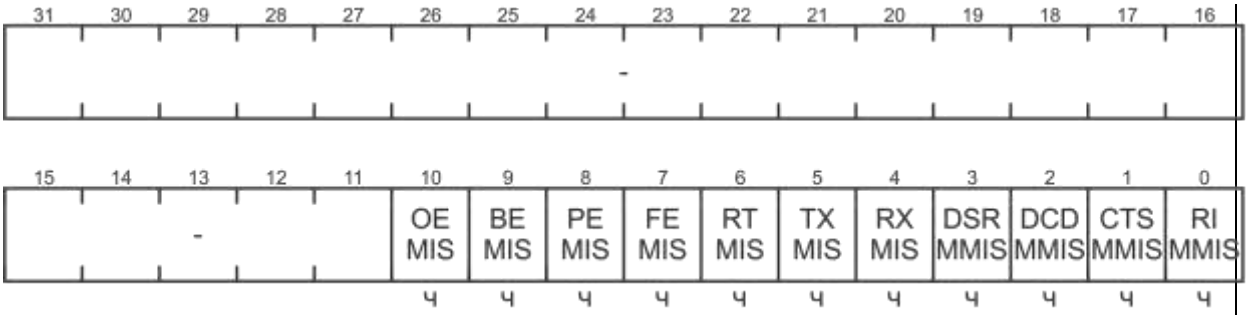
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-					OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	DSR MIM	DCD MIM	CTS MIM	RI MIM
					З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Регистр состояния прерываний
RIS Сброс: 00h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-					OE RIS	BE RIS	PE RIS	FE RIS	RT RIS	TX RIS	RX RIS	DSR RMIS	DCD RMIS	CTS RMIS	RI RMIS
					Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч	Ч

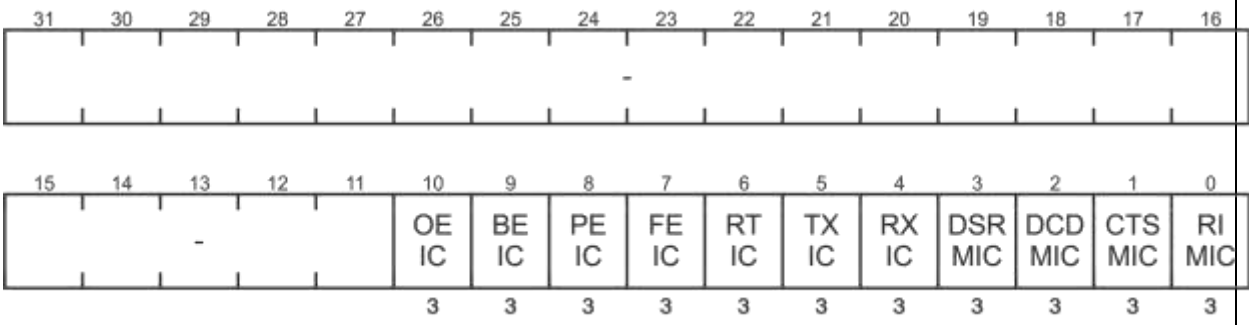
Регистр состояния прерываний с маскированием
MIS Сброс: 0000000xh



Окончание таблицы А.10.10

Регистр сброса прерываний

ICR Сброс: 00h



Поле		Бит	Описание
OE	IM/ RIS/ MIS/ IC	10	Переполнение буфера приемника
BE		9	Разрыв линии
PE		8	Ошибка контроля четности
FE		7	Ошибка в структуре кадра
RT		6	Таймаут приема данных
TX		5	Порог опустошения буфера передатчика
RX		4	Порог переполнения буфера приемника
DSR	MIM/ RMIS/ MMIS/ MIC	3	Изменение состояния линии UART_DSR
DCD		2	Изменение состояния линии UART_DCD
CTS		1	Изменение состояния линии UART_CTS
RI		0	Изменение состояния линии UART_RI
–		31-11	Зарезервировано

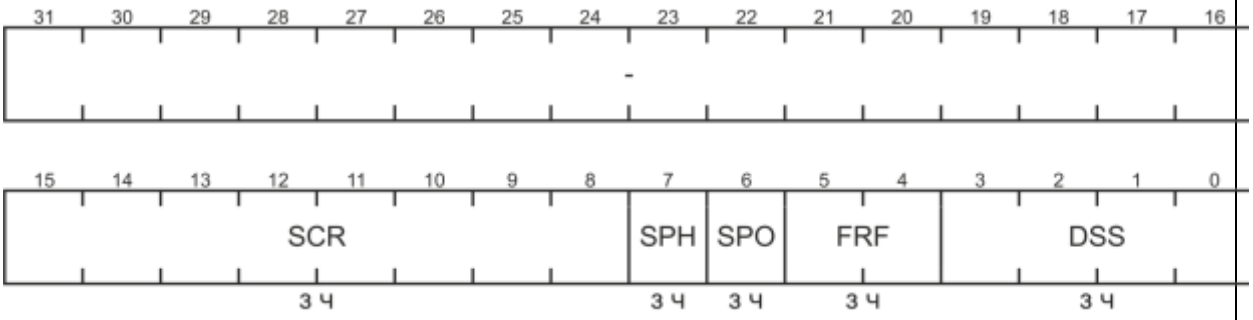
Примечание – Функционирование регистров:

- При возникновении прерываний устанавливаются соответствующие им немаскируемые флаги в регистре RIS.
- Установка бит в регистре IMSC формирует маску.
- В регистре MIS устанавливаются только те флаги, которые не закрыты маской регистра IMSC.
- Запись единиц в биты регистра ICR сбрасывает соответствующие им флаги в регистрах RIS и MIS, а также прерывания, вызвавшие установку этих флагов.

A.11 Регистры контроллера SPI

Таблица А.11.1 – Регистр управления 0

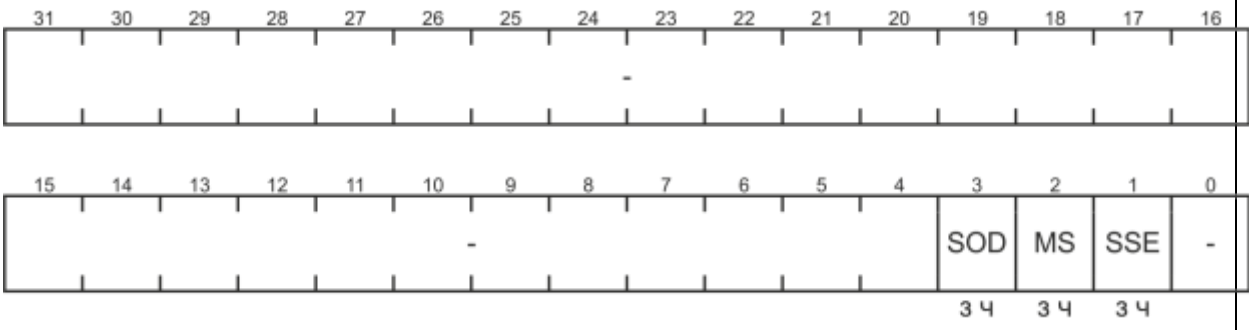
CR0 Сброс: 00h



Поле	Биты	Описание	
SCR	15-8	Коэффициент деления второго делителя. Может принимать значения от 00h до FFh	
SPH	7	Фаза сигнала SSPCLKOUT (только для протокола обмена SPI)	
		0	Выборка данных по переднему фронту синхросигнала, а установка по заднему
		1	Выборка данных по заднему фронту синхросигнала, а установка по переднему
SPO	6	Полярность сигнала SSPCLKOUT (только для протокола обмена SPI)	
		0	В режиме ожидания линия SPI_CLK удерживается в состоянии логического нуля
		1	В режиме ожидания линия SPI_CLK удерживается в состоянии логической единицы
FRF	5, 4	Поле выбора протокола обмена информацией	
		00	SPI
		01	SSI
		10	Microwire
		11	Зарезервировано
DSS	3-0	Размер слова данных:	
		0h-2h	Зарезервировано
		3h	4 бит
		4h	5 бит
		5h	6 бит
		6h	7 бит
		7h	8 бит
		8h	9 бит
		9h	10 бит
		Ah	11 бит
		Bh	12 бит
		Ch	13 бит
		Dh	14 бит
		Eh	15 бит
		Fh	16 бит
–	31-16	Зарезервировано	

Таблица А.11.2 – Регистр управления 1

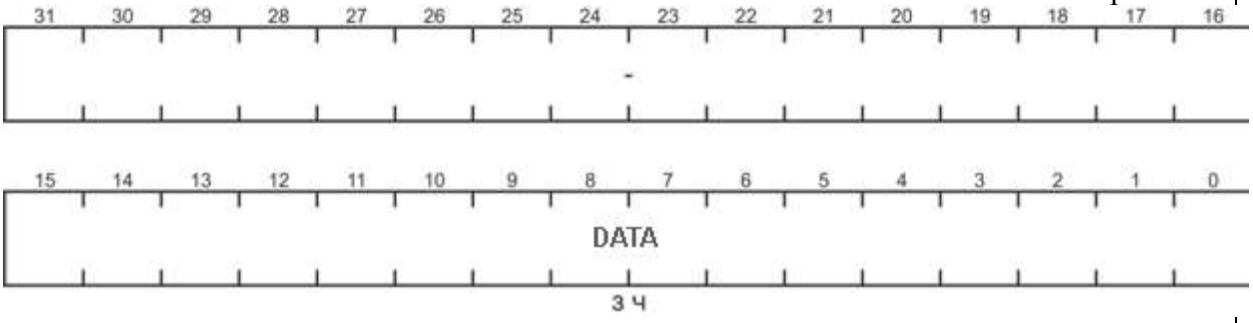
CR1 Сброс: 00h



Поле	Бит	Описание
SOD	3	Бит запрета передачи данных. В режиме мастера значение бита игнорируется. В режиме ведомого бит контролирует выход данных. Пока бит сброшен, передача и прием данных разрешены. Установка бита блокирует передачу данных и переводит вывод SPI_TXD в состояние слабой логической единицы, при этом прием тактового сигнала и прием данных не блокируются
MS	2	Бит выбора режима работы
		0 Мастер
		1 Ведомый
SSE	1	Бит разрешения работы приемопередатчика
		0 Запрещено
		1 Разрешено
–	31-4, 0	Зарезервировано

Таблица А.11.3 – Регистр данных

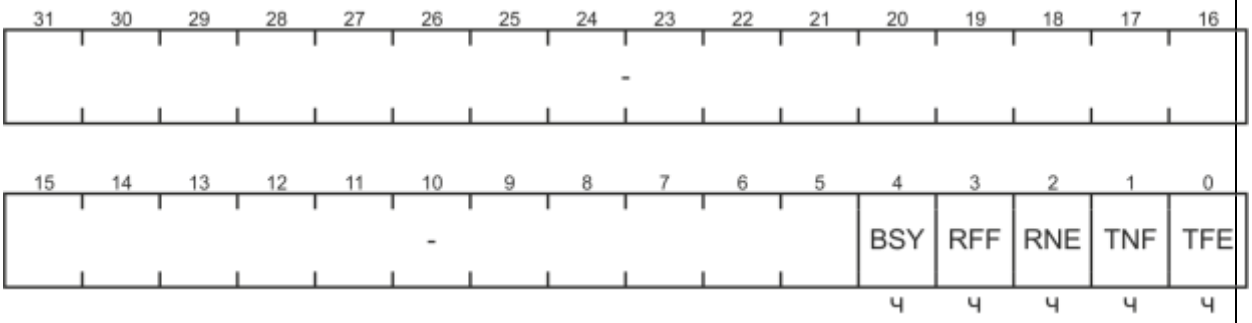
DR Сброс: 00h



Поле	Бит	Описание
DATA	15-0	16-разрядный буфер FIFO приемника и передатчика. Данные для передачи записываются в регистр. В случае если размер передаваемых данных менее 16 бит, перед записью в регистр они должны быть выравнены по правой границе. Неиспользуемые биты игнорируются. Принятые данные автоматически выравниваются по правой границе в блоке приемника. При чтении регистр возвращает принятые данные
–	31-16	Зарезервировано

Таблица А.11.4 – Регистр состояния

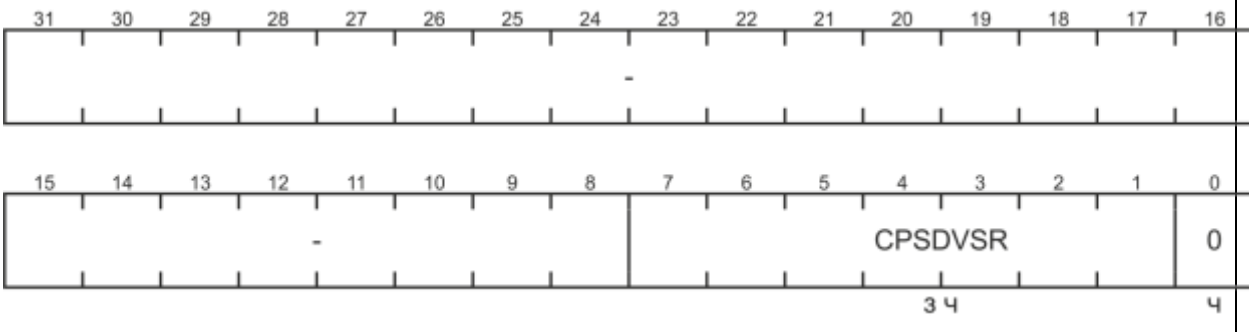
SR Сброс: 00000003h



Поле	Бит	Описание	
BSY	4	Флаг активности	
		0	Приемопередатчик не активен
		1	Приемопередатчик передает/принимает данные либо буфер FIFO передатчика не пуст
RFF	3	Флаг заполнения буфера FIFO приемника	
		0	Не заполнен
		1	Заполнен
RNE	2	Индикатор того, что буфер FIFO приемника не пуст	
		0	Пуст
		1	Не пуст
TNF	1	Индикатор того, что буфер FIFO передатчика не заполнен	
		0	Заполнен
		1	Не заполнен
TFE	0	Флаг пустоты буфера FIFO передатчика	
		0	Не пуст
		1	Пуст
–	31-5	Зарезервировано	

Таблица А.11.5 – Регистр делителя тактовой частоты

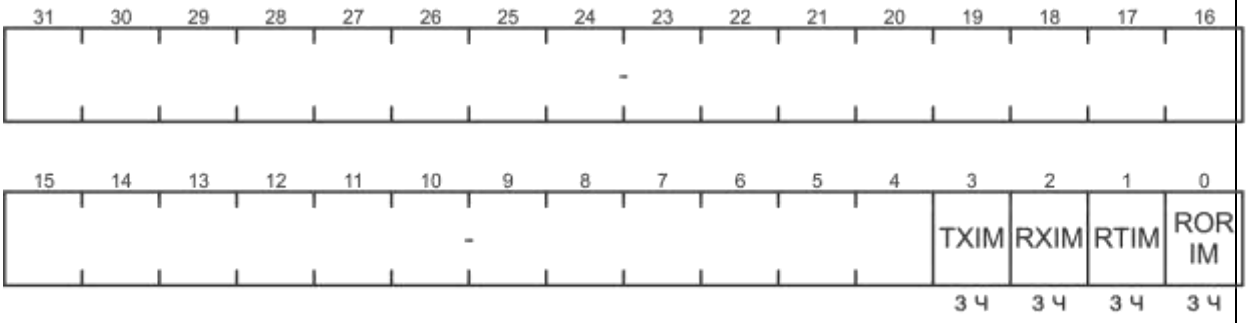
CPSR Сброс: 00h



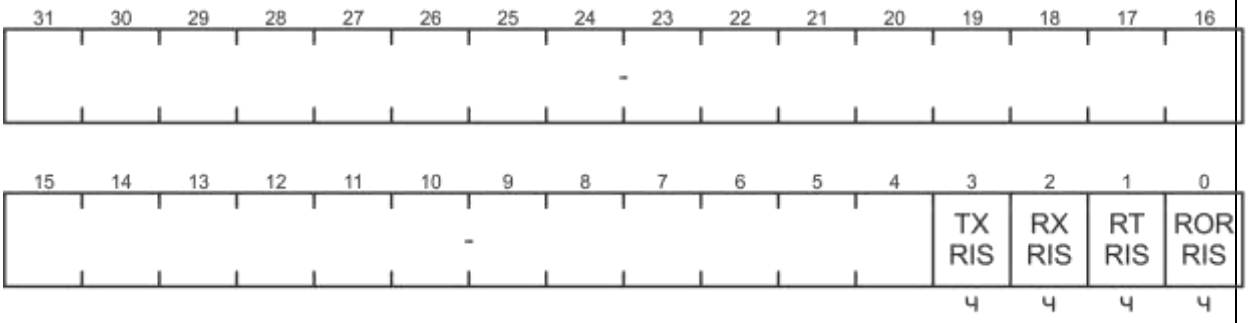
Поле	Биты	Описание
CPSDVSR	7-0	Коэффициент деления первого делителя. Может принимать четные значения от 02h до FEh
–	31-8	Зарезервировано

Таблица А.11.6 – Регистр прерываний

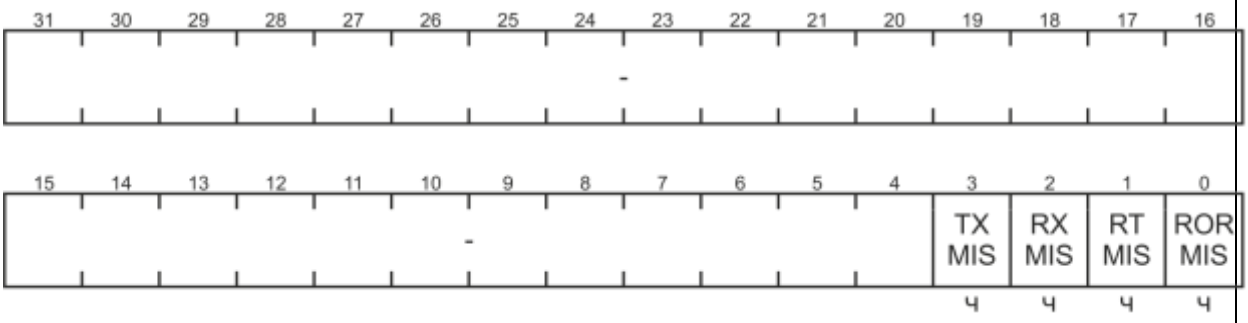
Регистр маски прерываний Сброс: 00000008h
SPI_IMSC



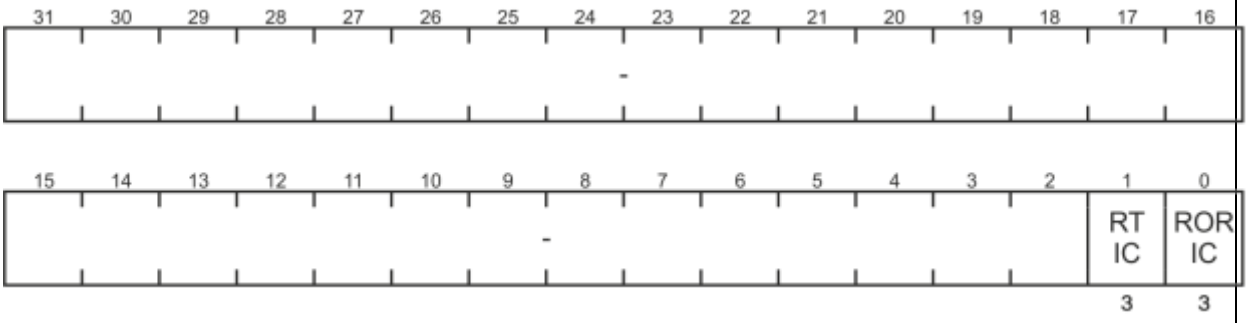
Регистр состояния прерываний
SPI_RIS Сброс: 00000003h



Регистр состояния прерываний с маскированием
SPI_MIS Сброс: 00h



Регистр сброса прерываний Сброс: 00h
SPI_ICR



Окончание таблицы А.11.6

Поле		Бит	Описание
TX	RIS/ IM/ MIS/ IC	3	Буфер передатчика опустошен наполовину
RX		2	Буфер приемника заполнен наполовину
RT		1	Таймаут приема данных
ROR		0	Переполнению буфера приемника
–		31-4	Зарезервировано

Примечание – Функционирование регистров:

- При возникновении прерываний устанавливаются соответствующие им немаскируемые флаги в регистре RIS.

- Установка/сброс бит в регистре IMSC формирует маску. По умолчанию все биты сброшены, и установка флагов запрещена. Для того чтобы убрать маску, следует установить соответствующий бит.

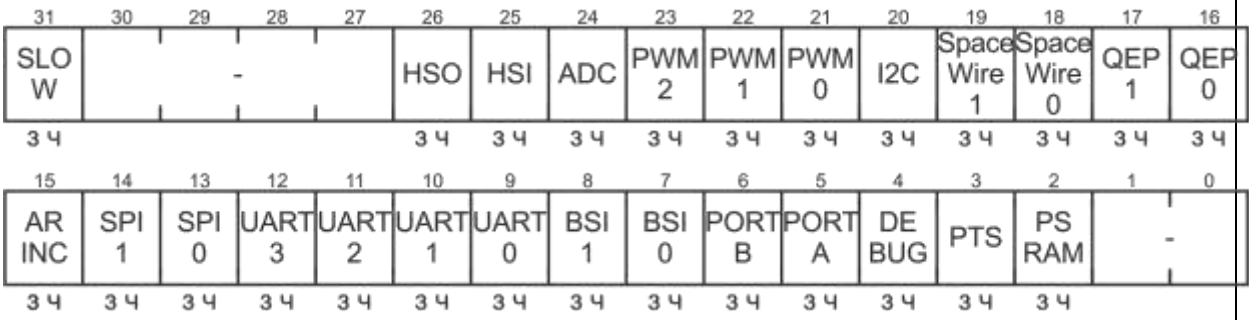
- В регистре MIS устанавливаются только те флаги, которые не закрыты маской регистра IMSC.

- Запись единиц в биты регистра ICR сбрасывает соответствующие им флаги в регистрах RIS и MIS, а также прерывания, вызвавшие установку этих флагов. Биты RTRIS и RTMIS также сбрасываются после чтения буфера приемника.

A.12 Регистры блока управления тактированием

Таблица А.12.1 – Регистр управления тактированием

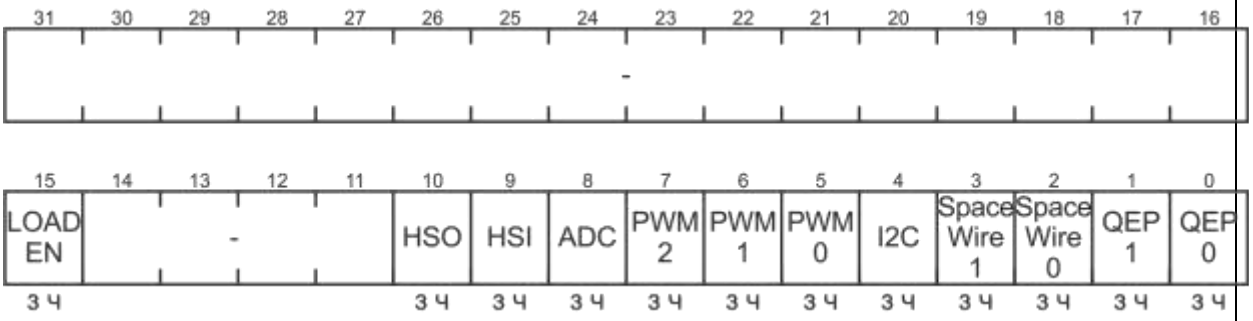
CLKEN Сброс: 7FFFh



Поле	Бит	Описание
SLOW	31	Бит включения режима Slow (понижение частоты тактирования в два раза)
HSO	26	Бит разрешения тактирования HSO
HSI	25	Бит разрешения тактирования HSI
ADC	24	Бит разрешения тактирования АЦП
PWM2	23	Бит разрешения тактирования блока ШИМ2
PWM1	22	Бит разрешения тактирования блока ШИМ1
PWM0	21	Бит разрешения тактирования блока ШИМ0
I2C	20	Бит разрешения тактирования I2C
SpaceWire 1	19	Бит разрешения тактирования SpaceWire 1
SpaceWire 0	18	Бит разрешения тактирования SpaceWire 0
QEP1	17	Бит разрешения тактирования квадратурного декодера 1
QEP0	16	Бит разрешения тактирования квадратурного декодера 0
ARINC	15	Бит разрешения тактирования ARINC
SPI1	14	Бит разрешения тактирования SPI1
SPI0	13	Бит разрешения тактирования SPI0
UART3	12	Бит разрешения тактирования UART3
UART2	11	Бит разрешения тактирования UART2
UART1	10	Бит разрешения тактирования UART1
UART0	9	Бит разрешения тактирования UART0
BSI1	8	Бит разрешения тактирования МПИ1
BSI0	7	Бит разрешения тактирования МПИ0
PORTB	6	Бит разрешения тактирования порта B
PORTA	5	Бит разрешения тактирования порта A
DEBUG	4	Бит разрешения тактирования при отладке
PTS	3	Бит разрешения тактирования блока PTS
PSRAM	2	Бит разрешения тактирования внутренней памяти программ
–	30-27, 1, 0	Зарезервировано
Для всех бит справедливо:		
0	Тактирование запрещено	
1	Тактирование разрешено	

Таблица А.12.2 – Регистр управления тактированием в режиме отладки

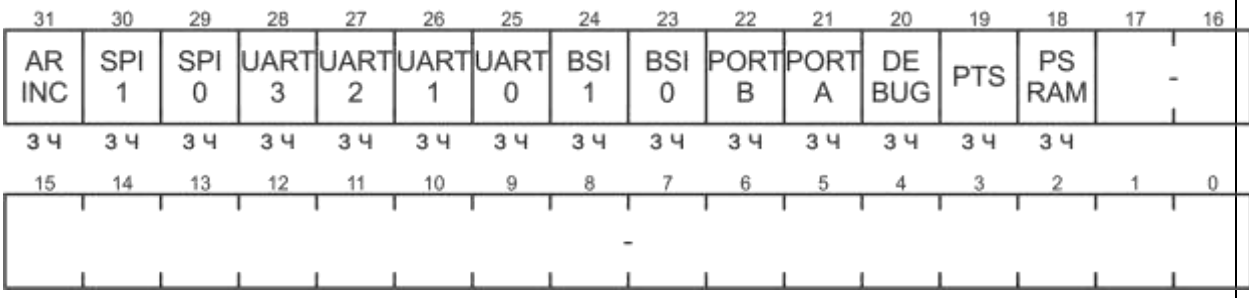
CLKDBG_H Сброс: 0h



Поле	Бит	Описание	
LOADEN	15	Бит разрешения загрузки регистра CLKEN по прерыванию DEBUG	
		0	Запрещена
		1	Разрешена
HSO	10	Бит разрешения тактирования HSO	
HSI	9	Бит разрешения тактирования HSI	
ADC	8	Бит разрешения тактирования АЦП	
PWM2	7	Бит разрешения тактирования блока ШИМ2	
PWM1	6	Бит разрешения тактирования блока ШИМ1	
PWM0	5	Бит разрешения тактирования блока ШИМ0	
I2C	4	Бит разрешения тактирования I2C	
SpaceWire1	3	Бит разрешения тактирования SpaceWire 1	
SpaceWire0	2	Бит разрешения тактирования SpaceWire 0	
QEP1	1	Бит разрешения тактирования квадратурного декодера 1	
QEP0	0	Бит разрешения тактирования квадратурного декодера 0	
–	31-16, 14-11	Зарезервировано	
Для всех бит справедливо:			
0	Тактирование запрещено		
1	Тактирование разрешено		

Таблица А.12.3 – Регистр управления тактированием в режиме отладки

CLKDBG_L Сброс: 0h

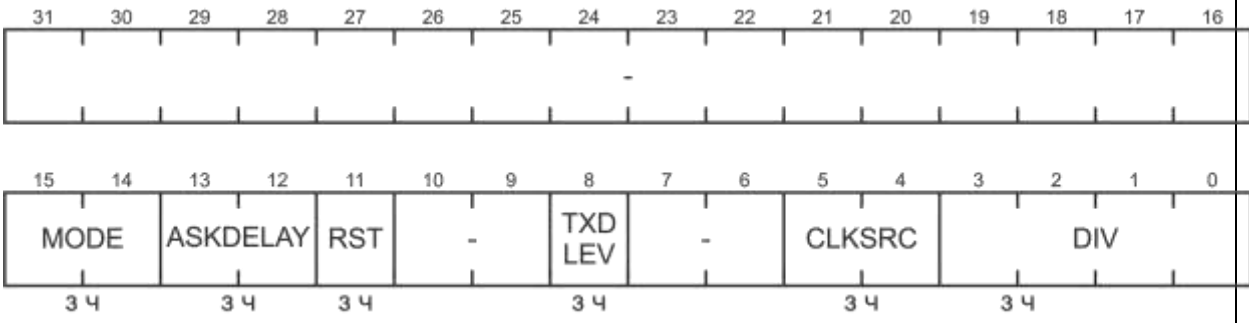


Поле	Бит	Описание
ARINC	31	Бит разрешения тактирования ARINC
SPI1	30	Бит разрешения тактирования SPI1
SPI0	29	Бит разрешения тактирования SPI0
UART3	28	Бит разрешения тактирования UART3
UART2	27	Бит разрешения тактирования UART2
UART1	26	Бит разрешения тактирования UART1
UART0	25	Бит разрешения тактирования UART0
BSI1	24	Бит разрешения тактирования МПИ1
BSI0	23	Бит разрешения тактирования МПИ0
PORTB	22	Бит разрешения тактирования порта B
PORTA	21	Бит разрешения тактирования порта A
DEBUG	20	Бит разрешения тактирования при отладке
PTS	19	Бит разрешения тактирования блока PTS
PSRAM	18	Бит разрешения тактирования внутренней памяти программ
–	17-0	Зарезервировано
Для всех бит справедливо:		
0	Тактирование запрещено	
1	Тактирование разрешено	

A.13 Регистры контроллера МПИ

Таблица А.13.1 – Регистр конфигурации 0

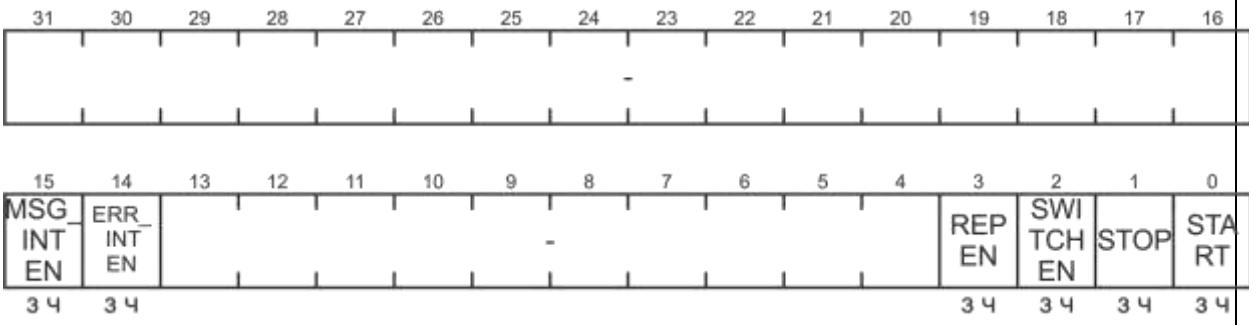
CONF Сброс: 0000_0000h



Поле	Бит	Описание	
MODE	15, 14	Режимы работы	
		00	–
		01	Контроллер шины (КШ)
		10	Удаленный терминал (УТ)
		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	Бит сброса устройства. Запись единицы в этот бит приводит к сбросу модуля и очистке всех его регистров. На данные, хранящиеся в ОЗУ, этот бит не влияет. Бит сбрасывается аппаратно	
TXD LEV	8	Бит выбора уровня сигнала на выводах Mn_TxDm, Mn_TxDNm при их неактивном состоянии (n – номер модуля, m – номер канала)	
		0	На выводах удерживается низкий уровень сигнала
		1	На выводах удерживается высокий уровень сигнала
CLK SRC	5, 4	Бит выбора источника опорной частоты	
		00, 11	XTAL1 (внешний источник)
		01	SysCLK
		10	M_EXTCLK (внешний источник)
DIV	3-0	Коэффициент делителя входной частоты (зависит от источника)	
		Источник	Значение коэффициента
		XTAL1	DIV = F_{XTAL1}/8. (A.13.1) При этом значение тактовой частоты F_{XTAL1} (в МГц) должно быть только: 24, 32, 40, 48, 56, 64, 72, 80, 88, 96 или 104
		SysCLK	Не важно
		M_EXTCLK	DIV = F_{MEXT}/4. (A.13.2) При этом значение тактовой частоты F_{MEXT} (в МГц) должно быть только: 12, 16, 20, 24, 28, 32, 36, 40, 44, 48 или 52 МГц
_	31-16, 10, 9, 7, 6	Зарезервировано	

Таблица А.13.2 – Регистр конфигурации 1 в режиме контроллера шины

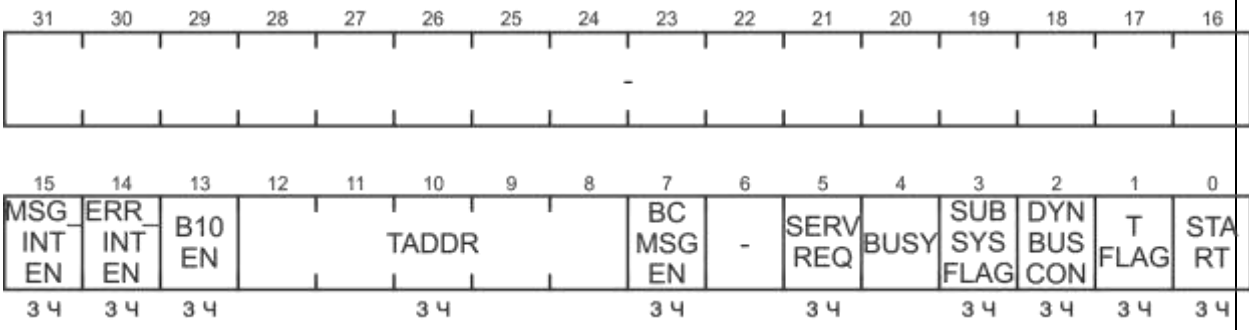
CON (КШ) Сброс: 0000_0000h



Поле	Бит	Описание	
MSG_INTEN	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_INTEN	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	
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.	
–	31-16, 13-4	Зарезервировано	

Таблица А.13.3 – Регистр конфигурации 1 в режиме удаленного терминала

CON (УТ) Сброс: 0000_0000h



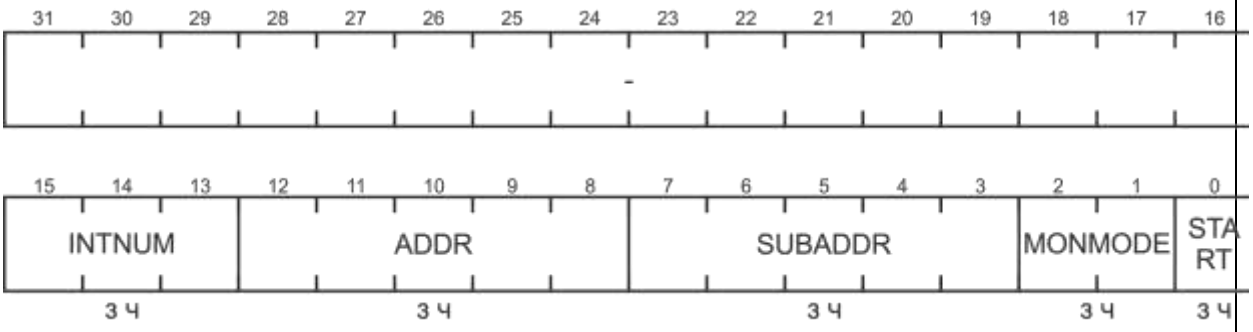
Поле	Бит	Описание	
MSG_INTEN	15	Бит разрешения прерывания после успешного окончания сообщения	
		0	Запрещено
		1	Разрешено
ERR_INTEN	14	Бит разрешения прерывания, если сообщение завершено с ошибкой	
		0	Запрещено
		1	Разрешено
B10EN	13	Бит разрешения контроля признака слова	
		0	Запрещено. 10-й бит принятого 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	
BUSY*	4	Бит занятости абонента	
SUBSYSFLAG*	3	Бит неисправности абонента	
DYNBUSCON*	2	Принято управление интерфейсом	
TFLAG*	1	Бит неисправности ОУ	
START	0	Бит включения ОУ. Пока этот бит установлен, собственный адрес ОУ и регистр BSICONFIG не могут быть изменены	
–	31-16, 6	Зарезервировано	

_____

* Запись единицы в этот бит установит соответствующий бит в ответном слове.

Таблица А.13.4 – Регистр конфигурации 1 в режиме монитора шины

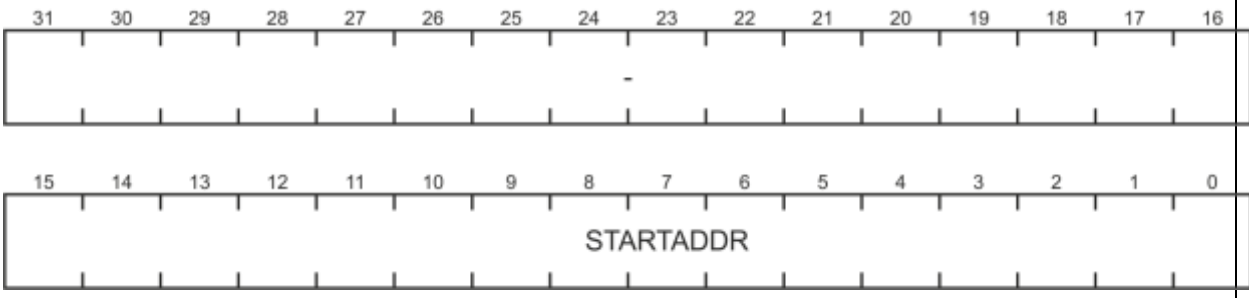
CON (МШ) Сброс: 0000_0000h



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 = 1Fh.	
MONMODE	2, 1	Режим выборки сообщений	
		00	Все сообщения
		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	
–	31-16	Зарезервировано	

Таблица А.13.5 – Регистр начального адреса

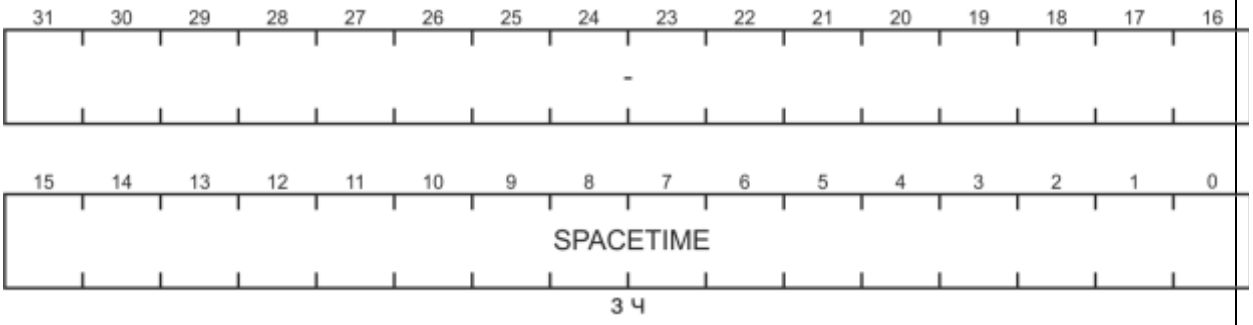
ADDR Сброс: 0000_0000h



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
–	31-16	Зарезервировано

Таблица А.13.6 – Регистр паузы между сообщениями

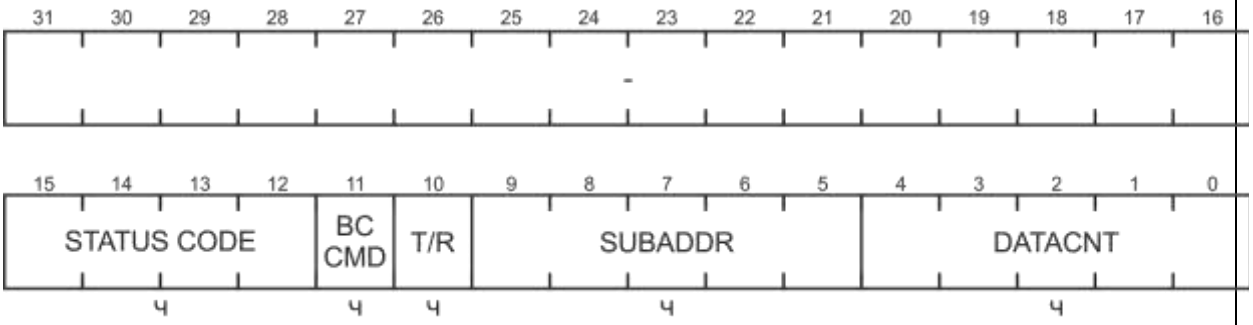
SPACE Сброс: 0000_0000h



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
–	31-16	Зарезервировано

Таблица А.13.7 – Регистр состояния в режимах контроллера шины и удаленного терминала

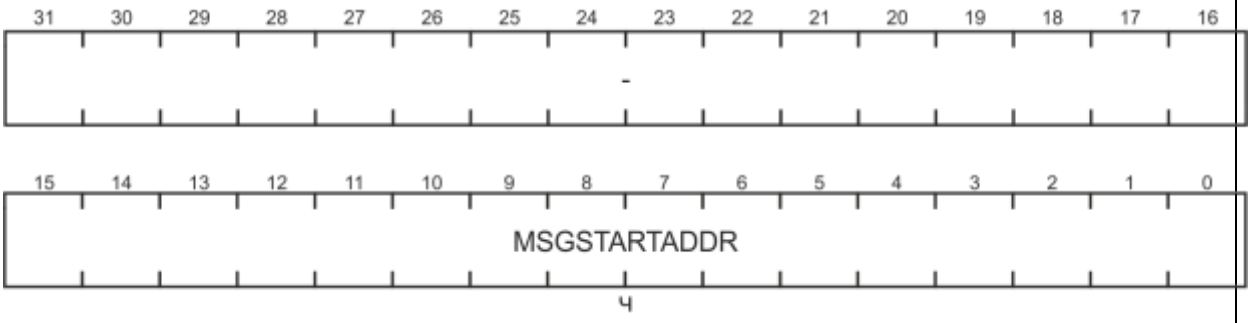
STAT (УТ) Сброс: 0000_0000h



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	
–	31-16	Зарезервировано	

Таблица А.13.8 – Регистр состояния в режиме монитора шины

STAT (МШ) Сброс: 0000_0000h

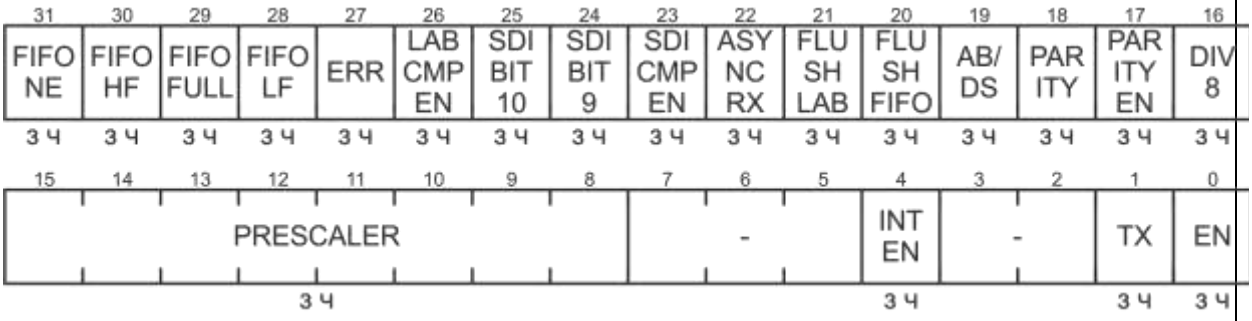


Поле	Бит	Описание
MSGSTARTADDR	15-0	Адрес начала непрочитанного информационного блока
–	31-16	Зарезервировано

A.14 Регистры контроллера ARINC

Таблица А.14.1 – Регистр управления частотой

CON (режим приемника) Сброс: 0h



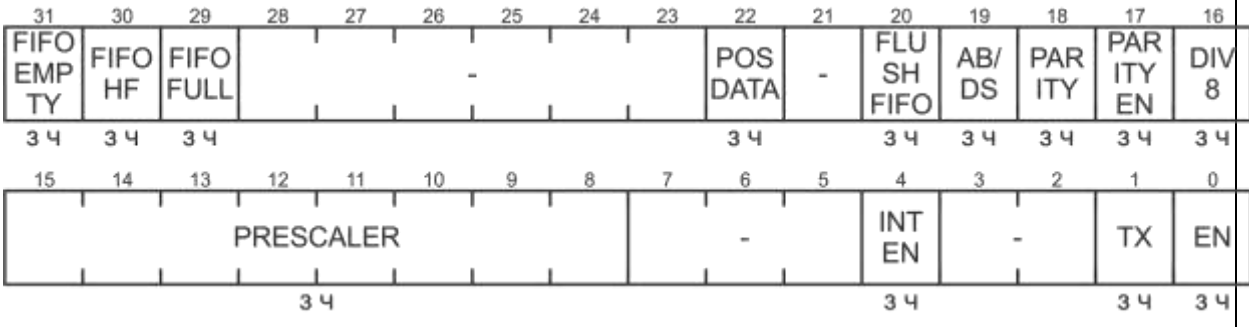
Поле	Бит	Описание	
FIFONE	31	Маска прерывания для бита FIFONE регистра STAT	
		0	Запрещено
		1	Разрешено
FIFOHF	30	Маска прерывания для бита FIFOHF регистра STAT	
		0	Запрещено
		1	Разрешено
FIFOFULL	29	Маска прерывания для бита FIFOFULL регистра STAT	
		0	Запрещено
		1	Разрешено
FIFOLF	28	Маска прерывания для бита FIFOLF регистра STAT	
		0	Запрещено
		1	Разрешено
ERR	27	Маска прерывания для бита ERR регистра STAT	
		0	Запрещено
		1	Разрешено
LABCMP EN	26	Включение сравнения принятой посылки с метками из FIFO меток	
		0	Выключено
		1	Включено
SDIBIT10	25	Значение десятого бита SDI для идентификации (с битом SDIBIT9)	
SDIBIT9	24	Значение девятого бита SDI для идентификации (с битом SDIBIT10)	
SDICMP EN	23	Бит разрешения сравнения по битам SDI	
		0	Запрещено
		1	Разрешено
ASYNCRX	22	Бит выбора длительности бита данных в режиме DS	
		0	Период
		1	Полупериод
FLUSH LAB	21	Бит сброса FIFO меток. Запись единицы в бит очищает FIFO меток	

Окончание таблицы А.14.1

Поле	Бит	Описание	
FLUSH FIFO	20	Бит сброса FIFO данных. Запись единицы в бит очищает FIFO данных	
AB/DS	19	Бит выбора формата передачи/приема данных	
		0	Режим DS (стробирование)
		1	Режим AB (сигнальный уровень, преобразованный в логический)
PARITY	18	Бит включения дополнения бита четности	
		0	До четного
		1	До нечетного
PARITY EN	17	Бит включения проверки на четность	
		0	Выключено
		1	Включено
DIV8	16	Бит включения делителя скорости передачи данных	
		0	100 кГц
		1	12,5 кГц
PRESCALER	15-8	Коэффициент деления частоты до 1 МГц	
INTEN	4	Бит разрешения прерывания блока	
		0	Запрещено
		1	Разрешено
TX	1	Бит выбора режима работы блока	
		0	Приемник
		1	Передатчик
EN	0	Бит разрешения работы блока	
		0	Выключен
		1	Включен
–	7-5, 3, 2	Зарезервировано	

Таблица А.14.2 – Регистр управления

CON (режим передатчика) Сброс: 0h



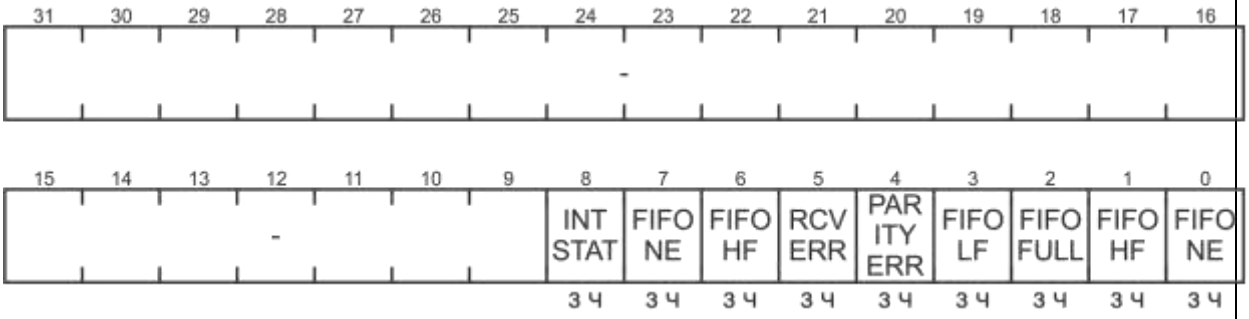
Поле	Бит	Описание	
FIFO EMPTY	31	Маска прерывания по опустошению FIFO данных	
		0	Запрещено
		1	Разрешено
FIFO HF	30	Маска прерывания по заполнению FIFO данных наполовину	
		0	Запрещено
		1	Разрешено

Окончание таблицы А.14.2

Поле	Бит	Описание	
FIFO FULL	29	Маска прерывания по заполнению FIFO данных	
		0	Запрещено
		1	Разрешено
POS DATA	22	Бит выбора фронта строба, по которому следует выставлять данные (при сброшенном бите AB/DS)	
		0	По заднему
		1	По переднему
FLUSH FIFO	20	Бит сброса FIFO данных Запись единицы в бит очищает FIFO данных	
AB/DS	19	Бит выбора формата передачи данных	
		0	Режим DS (стробирование)
		1	Режим AB (сигнальный уровень, преобразованный в логический)
PARITY	18	Бит включения дополнения бита четности	
		0	До четного
		1	До нечетного
PARITY EN	17	Бит включения проверки на четность	
		0	Включено
		1	Выключено
DIV8	16	Бит включения делителя скорости передачи данных	
		0	100 кГц
		1	12,5 кГц
PRESCALER	15-8	Коэффициент деления частоты до 1 МГц	
INTEN	4	Бит разрешения прерывания блока	
		0	Запрещено
		1	Разрешено
TX	1	Бит выбора режима работы блока	
		0	Приемник
		1	Передатчик
EN	0	Бит разрешения работы блока	
		0	Выключено
		1	Включено
–	28-23, 21, 7-5, 3, 2	Зарезервировано	

Таблица А.14.3 – Регистр состояния

STAT (режим приемника) Сброс: 0h



Поле	Бит	Описание
INT STAT	8	Индикатор прерывания. Остается установленным до тех пор, пока установлен хотя бы один флаг. Сбрасывается аппаратно, как только сброшены все флаги.
FIFO NE	7	Индикатор того, что FIFO данных не пусто
FIFO HF	6	Индикатор того, что FIFO данных заполнено наполовину
RCV ERR	5	Флаг ошибки при приеме
PARITY ERR	4	Флаг ошибки четности при приеме
FIFO LF	3	Флаг полного заполнения FIFO меток
FIFO FULL	2	Флаг полного заполнения FIFO данных
FIFO HF	1	Флаг заполнения FIFO данных наполовину
FIFO NE	0	Флаг того, что FIFO данных не пусто
–	31-9	Зарезервировано

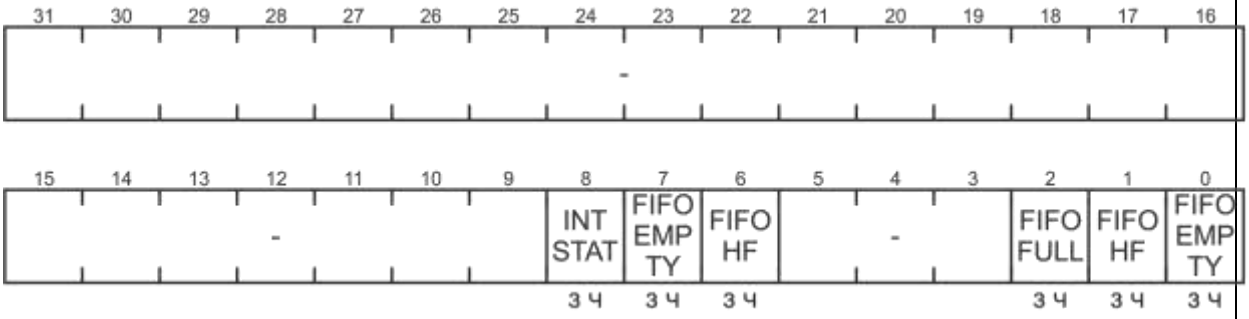
Примечания

1 Бит-индикатор остается установленным до тех пор, пока выполняется соответствующее условие.

2 Флаг устанавливается, как только выполняется соответствующее условие. Для сброса бита-флага следует записать в него единицу.

Таблица А.14.4 – Регистр состояния

STAT (режим передатчика) Сброс: 0h



Поле	Бит	Описание
INTSTAT	8	Индикатор прерывания. Остается установленным до тех пор, пока установлен хотя бы один флаг. Сбрасывается аппаратно, как только сброшены все флаги.
FIFOEMPTY	7	Индикатор того, что FIFO данных пусто
FIFOHF	6	Индикатор того, что FIFO данных заполнено наполовину
FIFOFULL	2	Флаг полного заполнения FIFO данных
FIFOHF	1	Флаг заполнения FIFO данных наполовину
FIFOEMPTY	0	Флаг полного опустошения FIFO данных
–	31-9, 5-3	Зарезервировано

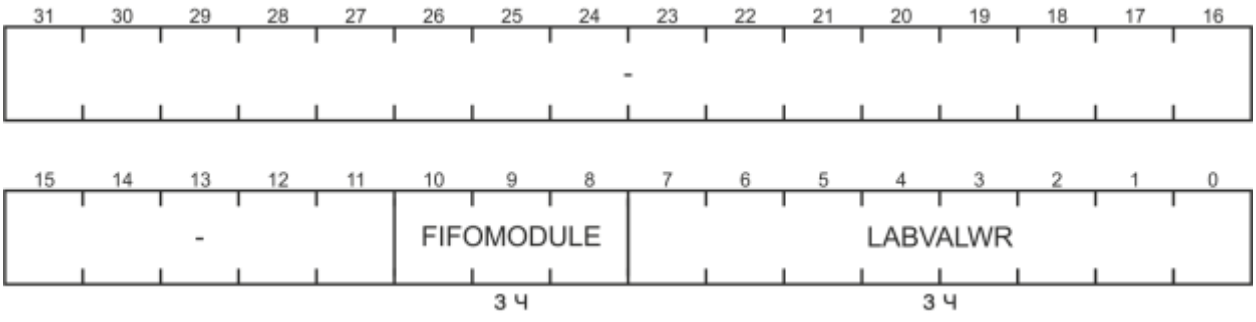
Примечания

1 Бит-индикатор остается установленным до тех пор, пока выполняется соответствующее условие.

2 Флаг устанавливается, как только выполняется соответствующее условие. Для сброса бита-флага следует записать в него единицу.

Таблица А.14.5 – Регистр FIFO

RXLABFIFO Сброс: 0h



Поле	Бит	Описание
FIFOMODULE	10-8	Номер модуля FIFO меток, в который будет записано значение метки
LABVALWR	7-0	Значение метки для записи в FIFO
–	31-11	Зарезервировано

Таблица А.14.6 – Регистры шлюза FIFO

Мнемоника	Назначение и описание	Сброс
DATA_READ	32-разрядный регистр шлюза FIFO для чтения принятых данных	00h
DATA_WRITE	32-разрядный регистр шлюза FIFO для записи данных для отправки	00h

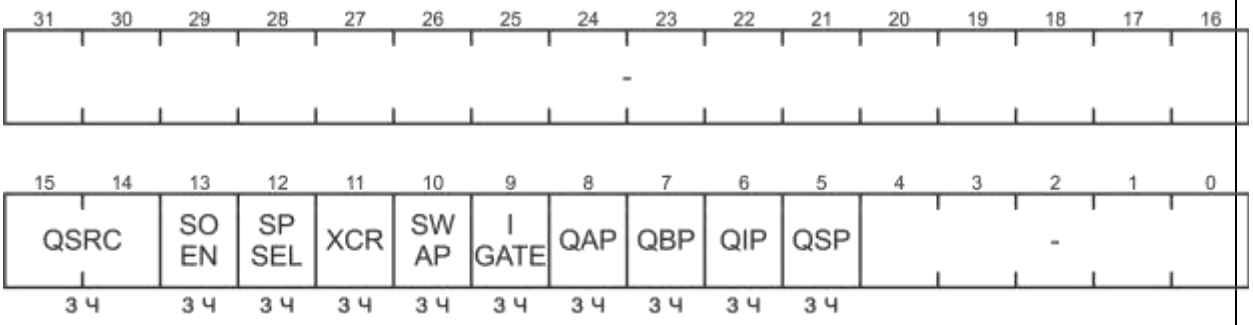
A.15 Регистры квадратурного декодера

Таблица А.15.1 – 32-разрядные регистры

Мнемоника	Назначение и описание	Сброс
QPOSCNT	Регистр счетчика позиции. Доступен только для чтения	00h
QPOSINIT	Регистр инициализации счетчика позиции	00h
QPOSMAX	Регистр максимального значения счетчика позиции	00h
QPOSCMP	Регистр сравнения счетчика позиции	00h
QPOSILAT	Регистр хранения позиции по индексации. Доступен только для чтения	00h
QPOSSLAT	Регистр хранения позиции по стробу. Доступен только для чтения	00h
QPOSLAT	Регистр хранения позиции по таймеру временных отсчетов. Доступен только для чтения	00h
QUTMR	Регистр таймера временных отсчетов Доступен только для чтения	00h
QUPRD	Регистр порога таймера временных отсчетов	00h
QWDTMR	Регистр счета сторожевого таймера. Доступен только для чтения	00h
QWDPRD	Регистр длительности счета сторожевого таймера	00h

Таблица А.15.2 – Регистр управления входами

QDECCTL Сброс: 00h



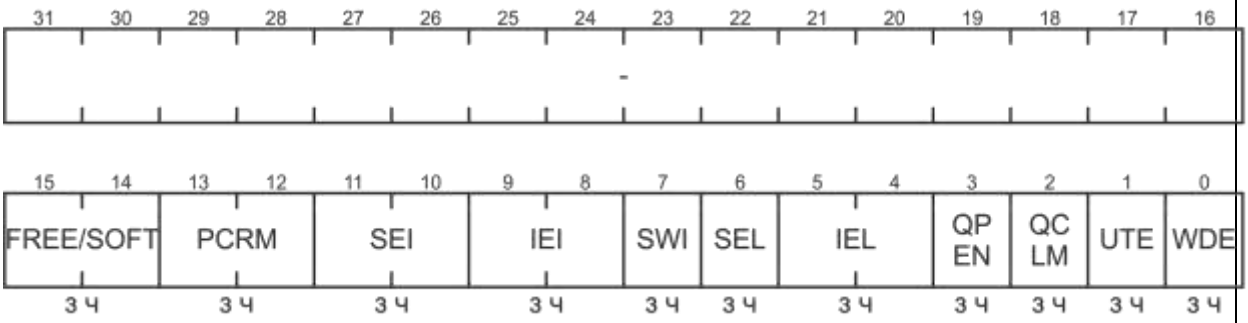
Поле	Биты	Описание	
QSRC	15, 14	Режим работы	
		00	Квадратурный
		01	Счета/направления
		10	Счет вверх (QCLK=xCLK, QDIR=1),
		11	Счет вниз (QCLK=xCLK, QDIR=0).
SOEN	13	Бит разрешения выдачи выходного сигнала компаратора	
		0	Запрещено
		1	Разрешено
SPSEL	12	Бит выбора вывода для выдачи выходного сигнала компаратора	
		0	Стробирующий вывод
		1	Индексный вывод
XCR	11	Бит выбора фронта квадратурного входа	
		0	Передний фронт
		1	Передний и задний фронты

Окончание таблицы А.15.2

Поле	Биты	Описание	
SWAP	10	Бит обмена входов QEPA и QEPB	
		0	Нет действий
		1	Входы QEPA и QEPB меняются местами
IGATE	9	Бит включения стробирования входного сигнала индексации	
QAP	8	Бит включения инвертирования входного сигнала с QEPA	
QBP	7	Бит включения инвертирования входного сигнала с QEPB	
QIP	6	Бит включения инвертирования входного сигнала с QEPI	
QSP	5	Бит включения инвертирования входного сигнала с QEPS	
–	31-16, 4-0	Зарезервировано	
Примечание – Для битов с 9 по 5 справедливо: 0 – выключено, 1 – включено.			

Таблица А.15.3 – Регистр управления квадратурного декодера

QEPCTL Сброс: 00h



Поле	Биты	Описание	
FREE/ SOFT	15, 14	Поле управления счетчиками QPOSCNT, QWDTMR, QUTMR, QCTMR в режиме отладки	
		00	Принудительная блокировка счета
		01	Счет до переполнения
		10, 11	Разблокирование счета
PCRM	13, 12	Поле задания события для сброса счетчика позиции	
		0 0	Событие индексации
		0 1	Достижение максимальной позиции
		1 0	Первое событие индексации
		1 1	Окончание временного отсчета
SEI	11, 10	Поле задания события стробирования для инициализации счетчика позиции (QPOSCNT = QPOSINIT)	
		00, 01	Работа без инициализации
		10	Передний фронт сигнала QEPS
		11	Передний фронт QEPS при вращении по часовой стрелке или задний фронт QEPS при вращении против часовой стрелки
IEI	9, 8	Поле задания события индексации для инициализации счетчика позиции (QPOSCNT = QPOSINIT)	

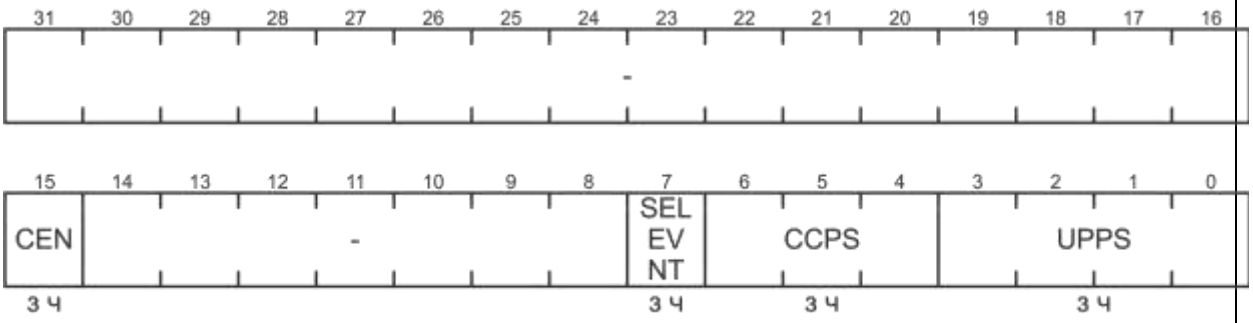
		00, 01	Работа без инициализации
		10	По переднему фронту сигнала QEPI
		11	По заднему фронту сигнала QEPI

Окончание таблицы А.15.3

Поле	Биты	Описание	
SWI	7	Бит программной инициализации счетчика позиции. Не сбрасывается аппаратно	
		0	Нет действий
		1	Запись единицы загружает счетчик позиции QPOSCNT значением QPOSINIT
SEL	6	Бит задания события стробирования для сохранения значения счетчика позиции (QPOSSLAT = POSCNT)	
		0	По переднему фронту QEPS
		1	По переднему фронту QEPS при вращении по часовой стрелке или по заднему фронту QEPS при вращении против часовой стрелки
IEL	5, 4	Поле задания события индексации для сохранения значения счетчика позиции (QPOSILAT = POSCNT)	
		00	Без сохранения
		01	По переднему фронту сигнала индексации
		10	По заднему фронту сигнала индексации
		11	По маркеру индексации
QPEN	3	Бит разрешения работы счетчика позиции	
		0	Запись нуля останавливает счетчик и сбрасывает его
		1	Работа разрешена
QCLM	2	Бит задания события сохранения значения регистров модуля захвата	
		0	По чтению QPOSCNT регистры QCTMR и QCPRD сохраняются в регистры QCTMRLAT и QCPRDLAT, соответственно.
		1	По окончанию временного отсчета регистры QPOSCNT, QCTMR и QCPRD сохраняются в регистры QPOSLAT, QCTMRLAT и QCPRDLAT, соответственно
UTE	1	Бит разрешения работы таймера временных отсчетов	
		0	Запрещено
		1	Разрешено
WDE	0	Бит разрешения работы сторожевого таймера	
		0	Запрещено
		1	Разрешено
–	31-16	Зарезервировано	

Таблица А.15.4 – Регистр захвата

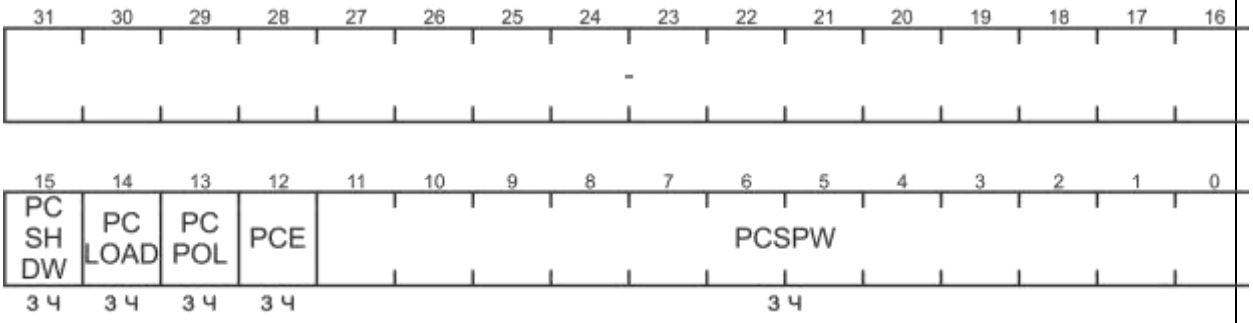
QCAPCTL Сброс: 00h



Поле	Биты	Описание	
CEN	15	Бит разрешения работы модуля захвата времени	
		0	Запрещено
		1	Разрешено
SELEVNT	7	Бит сброса таймера	
		0	По деленному квадратурному событию
		1	По получении сигнала PCSOUT от компаратора
CCPS	6-4	Поле задания делителя системного такта	
		000	Нет деления
		001	1/2
		010	1/4
		011	1/8
		100	1/16
		101	1/32
		110	1/64
		111	1/128
UPPS	3-0	Поле задания делителя квадратурного сигнала	
		0h	Нет деления
		1h	1/2
		2h	1/4
		3h	1/8
		4h	1/16
		5h	1/32
		6h	1/64
		7h	1/128
		8h	1/256
		9h	1/512
		Ah	1/1024
		Bh	1/2048
		Ch-Fh	Зарезервировано
–	31-16, 14-8	Зарезервировано	

Таблица А.15.5 – Регистр управления счетчиком позиции

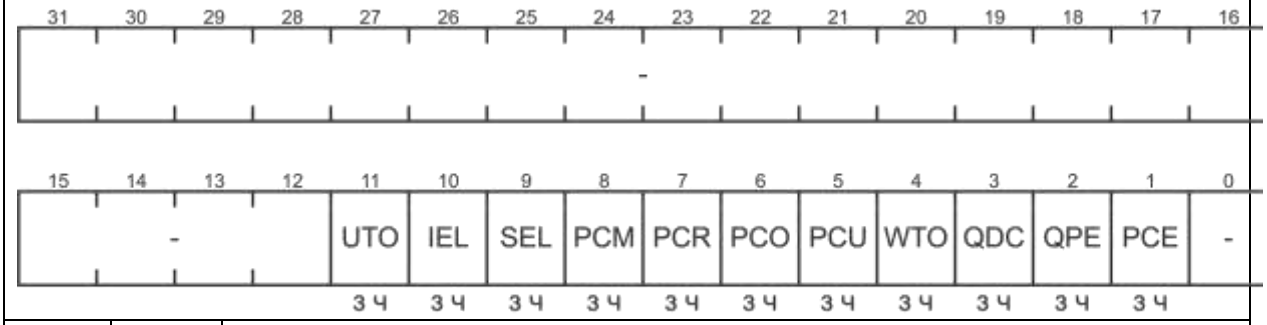
QPOSCTL Сброс: 00h



Поле	Биты	Описание	
PCSHDW	15	Бит разрешения режима отложенной загрузки	
		0	Запрещено
		1	Разрешено
PCLOAD	14	Бит выбора события загрузки в режиме отложенной записи	
		0	Загрузка отложенного значения в активный регистр по событию QPOSCNT = 0.
		1	Загрузка по QPOSCNT = QPOSCMP
PCPOL	13	Бит выбора полярности выхода синхронизации	
		0	Активная единица
		1	Активный ноль
PCE	12	Бит разрешения работы компаратора	
		0	Запрещено
		1	Разрешено
PCSPW	11-0	Поле задания ширины импульса выхода синхронизации	
		000h	Отсутствие импульса
		001h	2 × P
		…	…
		007h	8 × P
		…	…
		FFFh	4096 × P
		P – период системного тактового сигнала	
–	31-16	Зарезервировано	

Таблица А.15.6 – Регистр масок прерываний

QEINT Сброс: 00h

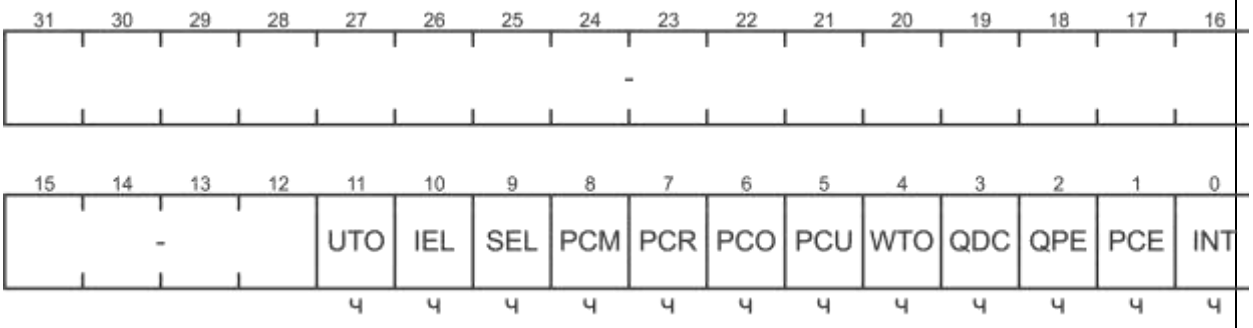


Поле	Бит	Описание
UTO	11	Бит разрешения прерывания по срабатыванию таймера временных отсчетов
IEL	10	Бит разрешения прерывания по событию индексации
SEL	9	Бит разрешения прерывания по событию стробирования
PCM	8	Бит разрешения прерывания по срабатыванию компаратора
PCR	7	Бит разрешения прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Бит разрешения прерывания при достижении счетчиком позиции максимального значения QPOSMAX при счете вверх
PCU	5	Бит разрешения прерывания при достижении счетчиком позиции минимального значения при счете вниз
WTO	4	Бит разрешения прерывания при срабатывании сторожевого таймера
QDC	3	Бит разрешения прерывания при смене направления вращения
QPE	2	Бит разрешения прерывания по ошибке фазы на квадратурном входе
PCE	1	Бит разрешения прерывания счетчика позиции
–	31-12, 0	Зарезервировано

Примечание – Установленный бит разрешает генерирование соответствующего прерывания, сброшенный – запрещает.

Таблица А.15.7 – Регистр флагов прерываний

QFLG Сброс: 00h



Поле	Бит	Описание
UTO	11	Флаг прерывания по срабатыванию таймера временных отсчетов
IEL	10	Флаг прерывания по событию индексации
SEL	9	Флаг прерывания по событию стробирования

Окончание таблицы А.15.7

Поле	Биты	Описание
PCM	8	Флаг прерывания по срабатыванию компаратора
PCR	7	Флаг прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Флаг прерывания при достижении счетчиком позиции максимального значения QPOSMAX при счете вверх
PCU	5	Флаг прерывания при достижении счетчиком позиции минимального значения при счете вниз
WTO	4	Флаг прерывания при срабатывании сторожевого таймера
QDC	3	Флаг прерывания при смене направления вращения
QPE	2	Флаг прерывания по ошибке фазы на квадратурном входе
PCE	1	Флаг прерывания ошибки счетчика позиции
INT	0	Флаг выходного прерывания блока квадратурного декодера
–	31-12	Зарезервировано

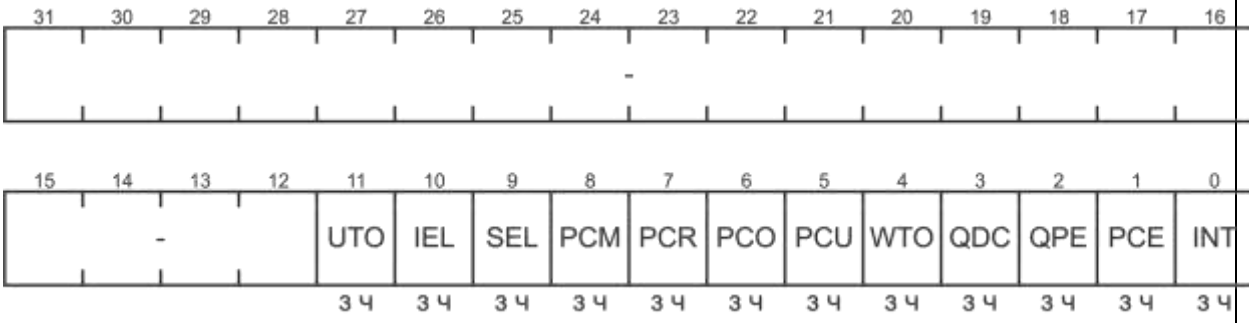
Примечания

1 Установленный бит является индикатором запроса соответствующего прерывания.

2 Сброс флагов прерываний осуществляется посредством регистра QCLR.

Таблица А.15.8 – Регистр сброса флагов прерываний

QCLR Сброс: 00h



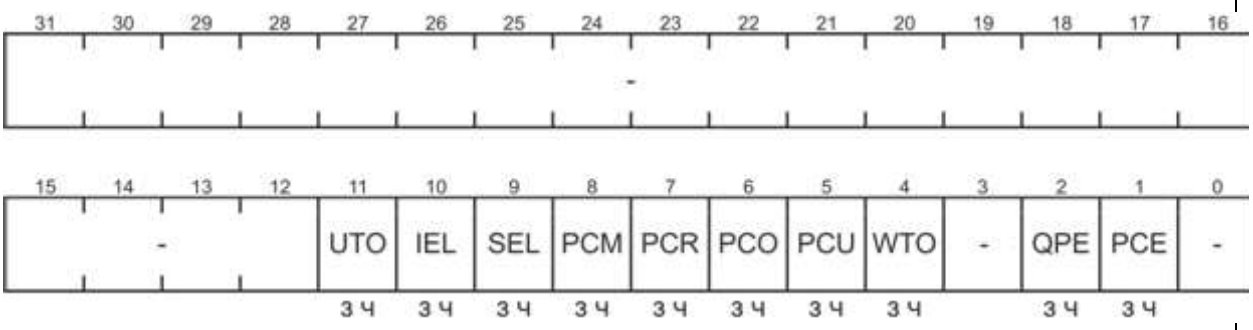
Примечания

1 Запись единицы в бит сбрасывает соответствующий флаг прерывания в регистре QFLG.

2 Биты с 31 по 12 зарезервированы.

Таблица А.15.9 – Регистр эмуляции прерываний

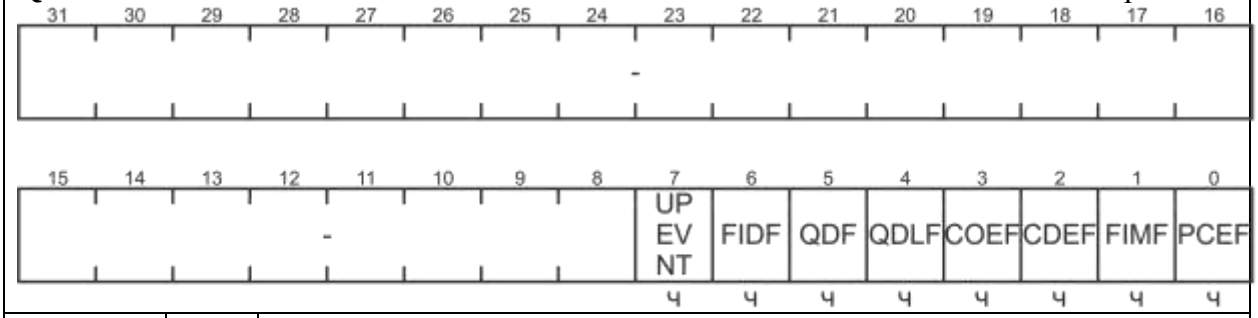
QFRC Сброс: 00h



Примечание – Запись единицы в бит устанавливает соответствующий флаг прерывания в регистре QFLG. Биты с 31 по 12, 3 и 0 зарезервированы.

Таблица А.15.10 – Регистр статуса

QEPSTS Сброс: 00h



Поле	Бит	Описание	
UPEVNT	7	Флаг сброса QCTMR и обновления QCPRD	
		0	Нет событий
		1	Зафиксировано событие сброса и обновления
		Сбрасывается записью 1	
FIDF	6	Индикатор направления вращения по событию первого импульса индексации	
		0	Против часовой стрелки (счет вниз)
		1	По часовой стрелке (счет вверх)
QDF	5	Флаг направления вращения. Обновляется по каждому событию на входах квадратур	
		0	Вращение вала ротора против часовой стрелки
		1	Вращение вала ротора по часовой стрелке
QDLF	4	Флаг направления вращения. Обновляется по каждому сигналу индексации	
		0	Вращение вала ротора против часовой стрелки
		1	Вращение вала ротора по часовой стрелке
COEF	3	Флаг ошибки переполнения счетчика QCTMR модуля захвата	
		0	Ошибка отсутствует
		1	Произошло переполнение
		Сбрасывается записью 1	
CDEF	2	Флаг ошибки изменения направления вращения вала ротора между двумя событиями UPEVNT	
		0	Ошибка отсутствует
		1	Произошло изменение направления вращения во время измерения
		Сбрасывается записью 1	
FIMF	1	Флаг приема первого импульса сигнала индексации	
		0	Импульсов нет либо первый импульс уже был принят
		1	Принят первый импульс сигнала индексации
		Сбрасывается записью 1	
PCEF	0	Флаг ошибки счетчика позиции. Обновляется по каждому сигналу индексации	
		0	Во время последнего сигнала индексации ошибки не возникло
		1	Ошибка счетчика позиции
–	31-8	Зарезервировано	

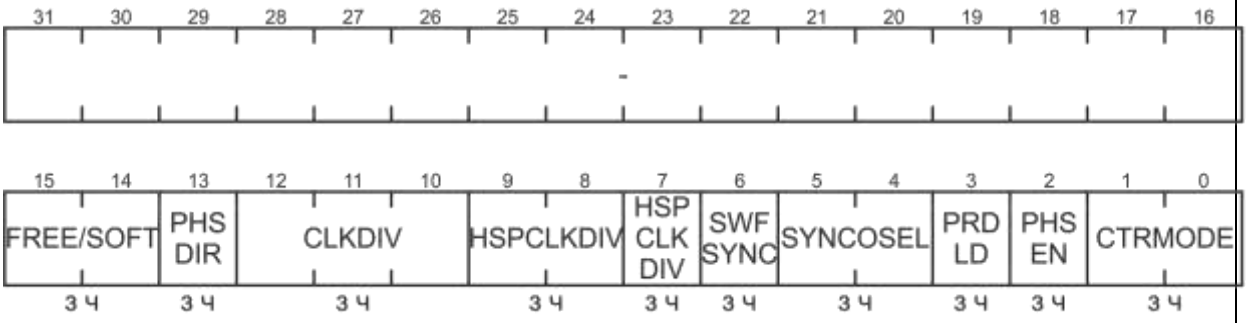
Таблица А.15.11 – Регистр управления таймером

QCTMR	Регистр таймера блока захвата	00h
QCPRD	Регистр длительности измерения блока захвата	00h
QCTMRLAT	Регистр хранения таймера блока захвата. Доступен только для чтения	00h
QCPRDLAT	Регистр хранения длительности измерения блока захвата	00h
INTCLR	Регистр сброса прерываний декодера	00h

A.16 Регистры блока ШИМ

Таблица А.16.1 – Регистр управления таймером

TBCTL Сброс: 00h



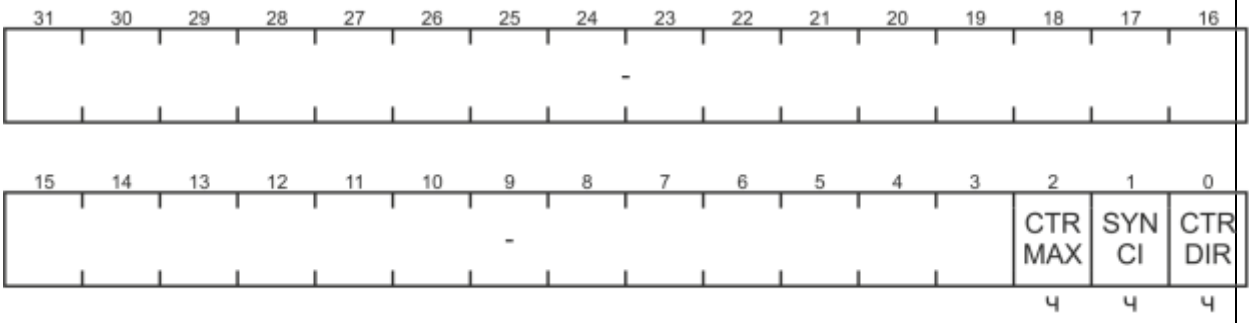
Поле	Биты	Описание	
FREE/ SOFT	15, 14	Поле задания поведения счетчика ШИМ после перехода в режим останова во время отладки	
		00	Счетчик будет остановлен на следующий такт TBCLK
		01	Счетчик будет остановлен по достижении события: - TBCTR = TBPRD (при счете вверх); - TBCTR = 0000h (при счете вниз или вверх-вниз).
		1x	Счетчик продолжит работу
PHSDIR	13	Бит задания фазового направления (используется только при двунаправленном счете). Задает направление счета после синхронизации. Загружается вместе с регистром фазы TBPHS	
		0	Вниз
		1	Вверх
CLKDIV	12-10	Поле задания первого делителя тактовой частоты	
		000	1
		001	1/2
		010	1/4
		011	1/8
		100	1/16
		101	1/32
		110	1/64
		111	1/128

Окончание таблицы А.16.1

Поле	Биты	Описание	
HSPCLKDIV	9-7	Поле задания второго делителя тактовой частоты. Конечное значение делителя является произведением значений делителей, задаваемых полями CLKDIV и HSPCLKDIV	
		000	1
		001	1/2
		010	1/4
		011	1/6
		100	1/8
		101	1/10
		110	1/12
		111	1/14
SWFSYNC	6	Бит программной эмуляции появления синхроимпульса	
		0	Нет действий
		1	Запись единицы вызывает появление синхроимпульса в цепи PWM_SYNCI
SYNCOSEL	5, 4	Поле выбора источника для выходного сигнала синхронизации PWM_SYNCO	
		00	PWM_SYNCI
		01	CTR = 0000h
		10	CTR = CMPB
		11	Выдача синхроимпульса запрещена
PRDLD	3	Бит управления загрузкой регистра TBPRD	
		0	Режим отложенной загрузки регистра TBPRD разрешен
		1	Запись в TBPRD будет произведена сразу в активный регистр
PHSEN	2	Бит разрешения загрузки счетчика таймера	
		0	Запрещено
		1	Разрешена загрузка счетчика TBCTR значением регистра фазы TBPHS при получении события синхронизации (импульс на входе PWM_SYNCI или запись в бит SWFSYNC)
CTRMODE	1, 0	Поле задания направления счета	
		00	Вверх
		01	Вниз
		10	Вверх-вниз
		11	Счетчик остановлен
–	31-16	Зарезервировано	

Таблица А.16.2 – Регистр статуса таймера

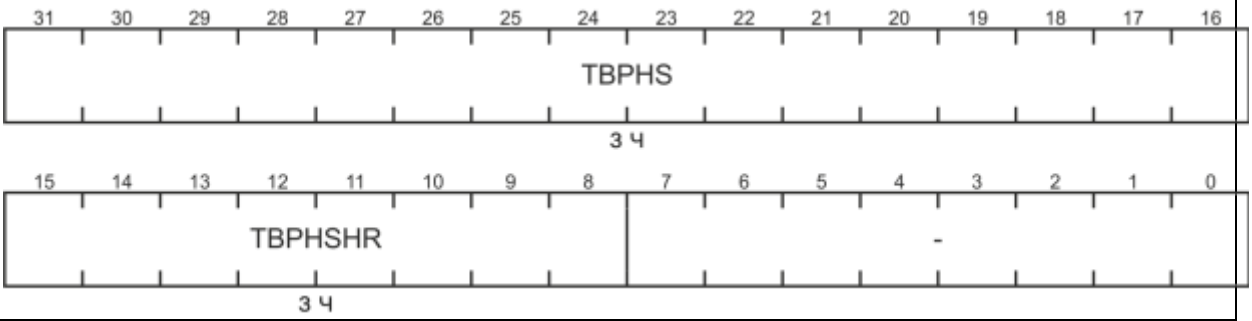
TBSTS Сброс: 00h



Поле	Бит	Описание	
CTRMAX	2	Флаг достижения счетчиком таймера своего максимального значения FFFFh	
		0	Значение не достигнуто или флаг был сброшен
		1	Значение было достигнуто
		Запись единицы сбрасывает флаг	
SYNCI	1	Флаг синхронизации	
		0	Синхронизация не достигнута или флаг был сброшен
		1	Синхронизация произошла
		Запись единицы сбрасывает флаг	
CTRDIR	0	Флаг направления счета таймера	
		0	Вниз
		1	Вверх
–	31-3	Зарезервировано	

Таблица А.16.3 – Регистр фазы

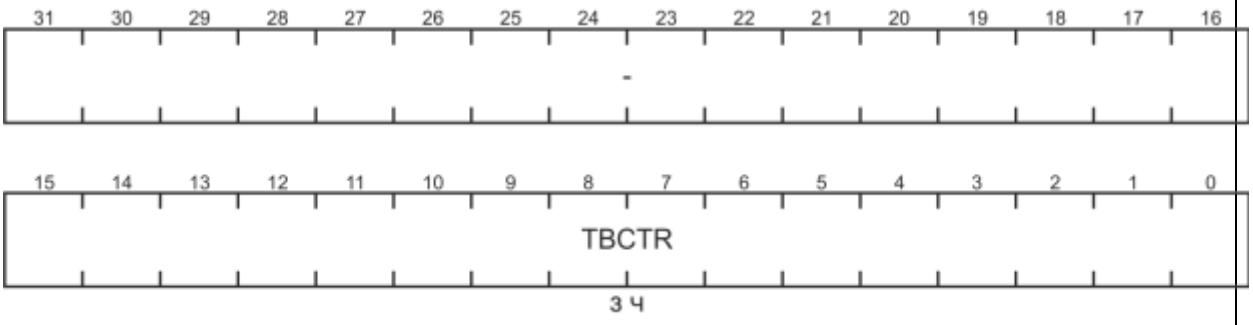
TBPHS Сброс: 00h



Поле	Биты	Описание
TBPHS	31-16	Поле задания начальной фазы таймера при получении сигнала синхронизации
TBPHSHR	15-8	Поле дополнительных разрядов начальной фазы таймера. Доступен в блоке ШИМ высокого разрешения. Если бит PHSEN сброшен, то синхронизация отключена, и таймер не будет загружаться значением TBPHS. Если бит PHSEN установлен, то по получению события синхронизации в счетчик таймера TBCTR будет загружено значение TBPHS
–	7-0	Зарезервировано

Таблица А.16.4 – Регистр текущего значения таймера

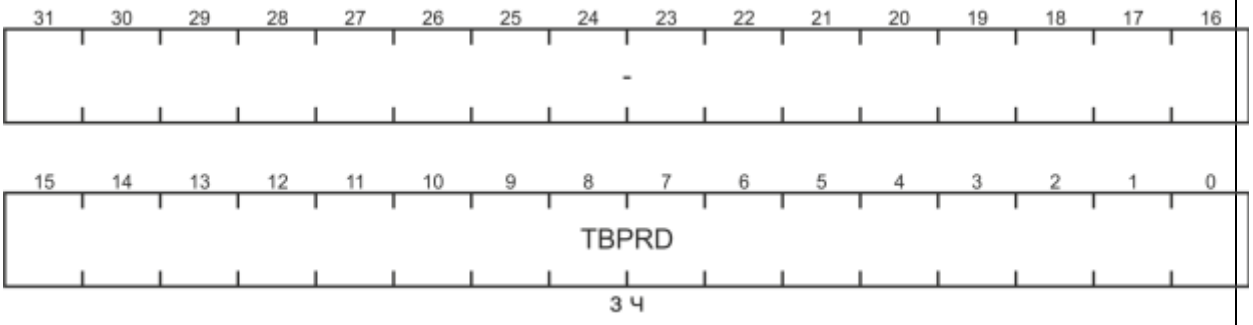
TBCTR Сброс: 00h



TBCTR	15-0	Текущее значение счетчика таймера. Запись в регистр изменяет значение таймера. Запись происходит асинхронно с TBLK и не использует отложенный механизм загрузки
–	31-16	Зарезервировано

Таблица А.16.5 – Регистр максимального значения таймера

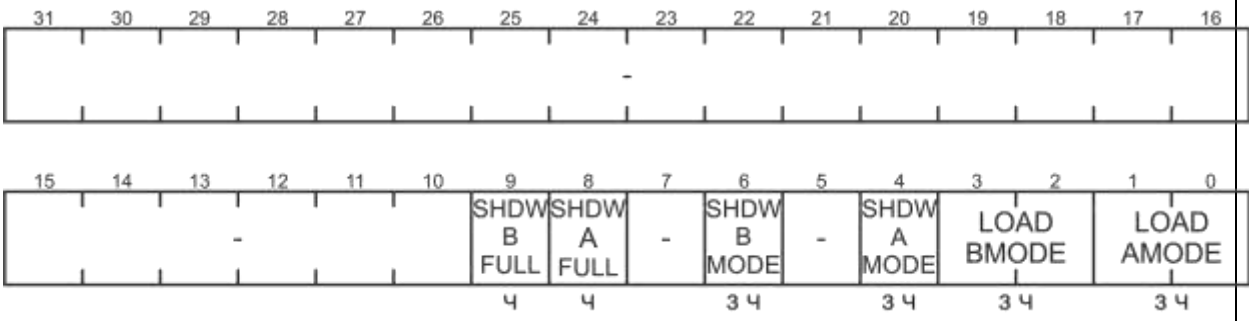
TBPRD Сброс: 00h



Поле	Биты	Описание
TBPRD	15-0	Период таймера (максимальное значение счета таймера). Отложенная загрузка в этот регистр программируется битом PRDLD регистра TBCTL. По умолчанию бит PRDLD сброшен и запись в регистр TBPRD приводит к записи в теневой регистр. Активный регистр будет загружен по событию TBCTR = Zero. Если бит PRDLD установлен, то запись выполняется напрямую в активный регистр
–	31-16	Зарезервировано

Таблица А.16.6 – Регистр управления компаратором

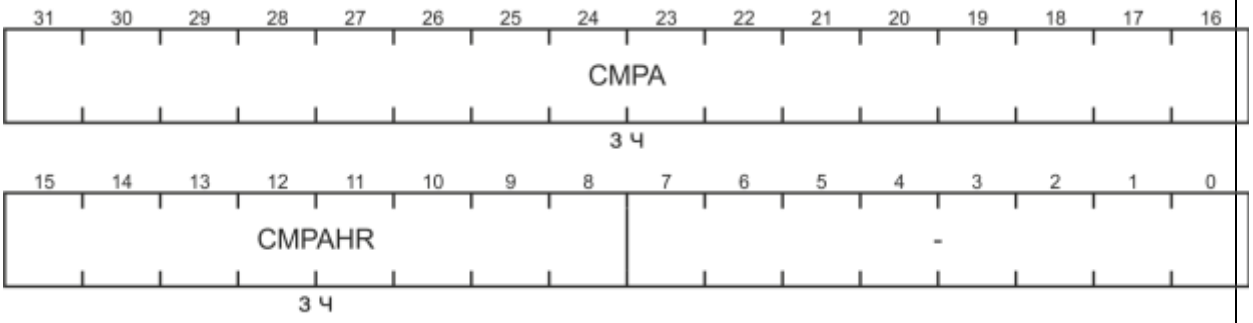
CMPCTL Сброс: 00h



Поле	Биты	Описание	
SHDWB FULL	9	Флаг отложенной загрузки в регистр CMPB	
		0	Нет действий
		1	Активный регистр загружен значением из теневого регистра
SHDWA FULL	8	Флаг отложенной загрузки в регистр CMPA	
		0	Нет действий
		1	Активный регистр загружен значением из теневого регистра
SHDWB MODE	6	Бит управления загрузкой регистра CMPB	
		0	Значение, записываемое в регистр CMPB, размещается в теневом регистре (отложенная загрузка)
		1	Производится загрузка напрямую в активный регистр
SHDWA MODE	4	Бит управления загрузкой регистра CMPA	
		0	Значение, записываемое в регистр CMPA, размещается в теневом регистре (отложенная загрузка)
		1	Производится загрузка напрямую в активный регистр
LOADB MODE	3, 2	Поле задания события загрузки отложенного значения в регистр CMPB (при условии, что бит SHDWBMODE сброшен)	
		00	CTR = Zero
		01	CTR = PRD
		10	CTR = Zero или CTR = PRD
		11	Загрузка запрещена
LOADA MODE	1, 0	Поле задания события загрузки отложенного значения в регистр CMPA (при условии, что бит SHDWAMODE сброшен)	
		00	CTR = Zero
		01	CTR = PRD
		10	CTR = Zero или CTR = PRD
		11	Загрузка запрещена
–	31-10, 7, 5	Зарезервировано	

Таблица А.16.7 – Регистр порога срабатывания А

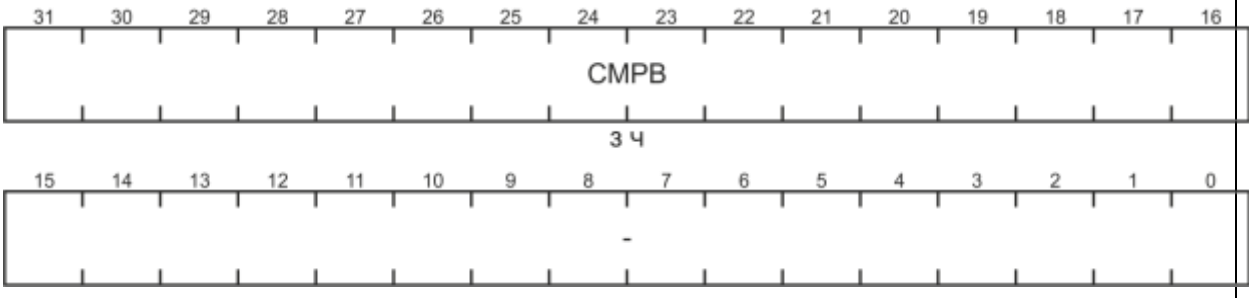
CMPA Сброс: 00h



Поле	Биты	Описание
CMPA	31-16	Активное значение порога срабатывания канала А, которое сравнивается со значением счетчика таймера. При совпадении значений формируется событие CTR = CMPA, которое влияет на поведение сигналов на линиях PWMA и PWMB
CMPAHR	15-8	Дополнительные младшие биты значения порога срабатывания канала А (используются только для блока ШИМ высокого разрешения). Отложенная загрузка включается и работает также как и для поля CMPA
–	7-0	Зарезервировано

Таблица А.16.8 – Регистр порога срабатывания В

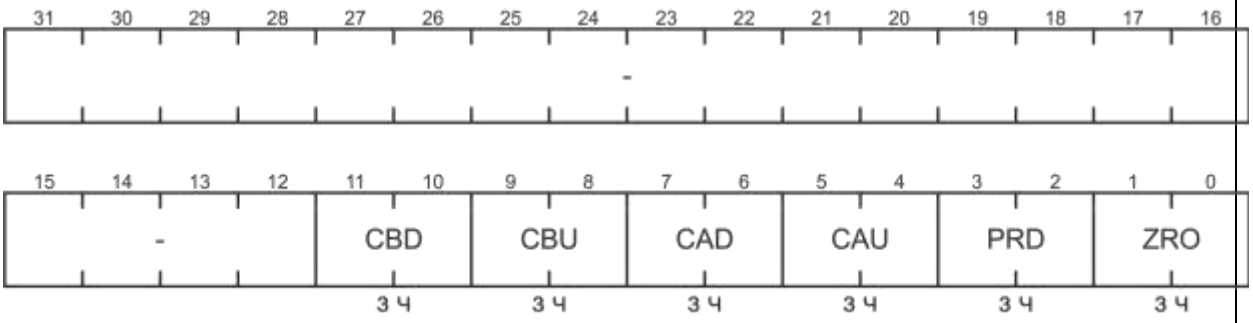
CMPB Сброс: 00h



Поле	Биты	Описание
CMPB	31-16	Активное значение порога срабатывания канала В, которое сравнивается со значением счетчика таймера. При совпадении значений формируется событие CTR = CMPB, которое влияет на поведение сигналов на линиях PWMA и PWMB
–	15-0	Зарезервировано

Таблица А.16.9 – Регистр обработчика для выхода A/B

AQCTLA/AQCTLB Сброс: 00h

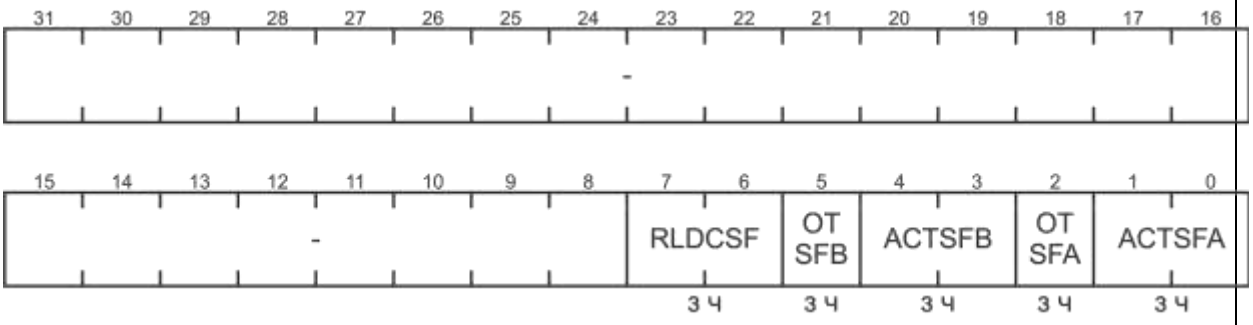


Поле	Биты	Описание
CBD	11, 10	Действие на выводе PWM_A/PWM_B при CTR = CMPB при счете вниз
CBU	9, 8	Действие на выводе PWM_A/PWM_B при CTR = CMPB при счете вверх
CAD	7, 6	Действие на выводе PWM_A/PWM_B при CTR = CMPA при счете вниз
CAU	5, 4	Действие на выводе PWM_A/PWM_B при CTR = CMPA при счете вверх
PRD	3, 2	Действие на выводе PWM_A/PWM_B при CTR = PRD
ZRO	1, 0	Действие на выводе PWM_A/PWM_B при CTR = Zero
–	31-12	Зарезервировано

Для каждого события может быть задано одно из четырех действий:
00 – нет реакции;
01 – переключение PWM_A/PWM_B в ноль;
10 – переключение PWM_A/PWM_B в единицу;
11 – инверсия PWM_A/PWM_B.

Таблица А.16.10 – Регистр программного управления однократным действием

AQSFRC Сброс: 00h



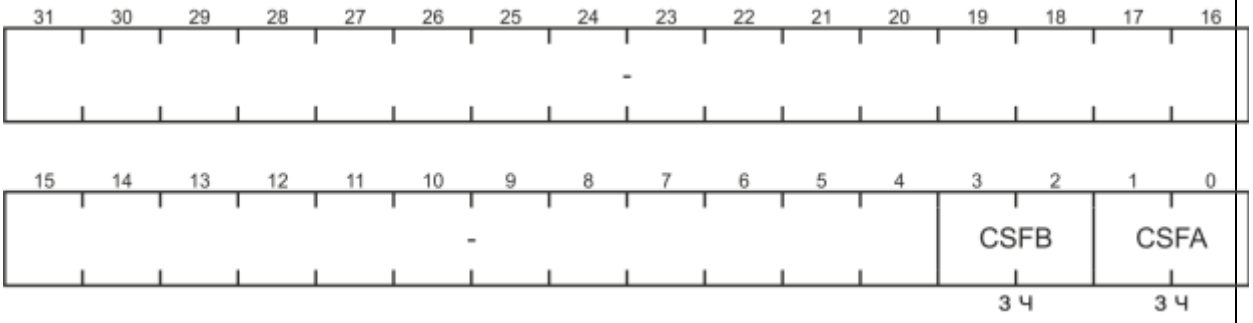
Поле	Биты	Описание	
RLDCSF	7, 6	Выбор действия с выходным сигналом на выводе	
		00	Загрузка по значению счетчика, равного нулю
		01	Загрузка по значению счетчика, равного периоду
		10	Загрузка по значению счетчика, равного нулю или периоду
		11	Немедленное обновление
OTSFB	5	Запись единицы инициирует однократный импульс – событие для формирования выхода	

Окончание таблицы А.16.10

Поле	Биты	Описание	
ACTSFB	4, 3	Выбор действия с выходным сигналом на выводе	
		00	Нет действий
		01	PWM_B = 0
		10	PWM_B = 1
		11	Инверсия EPWMB
OTSFA	2	Запись единицы инициирует однократный импульс – событие для формирования выхода	
ACTSFA	1, 0	Выбор действия с выходным сигналом на выводе	
		00	Нет действий
		01	PWM_A = 0
		10	PWM_A = 1
		11	Инверсия PWM_A
–	31-8	Зарезервировано	

Таблица А.16.11 – Регистр обработчика для циклического программного управления

AQCSFRC Сброс: 00h

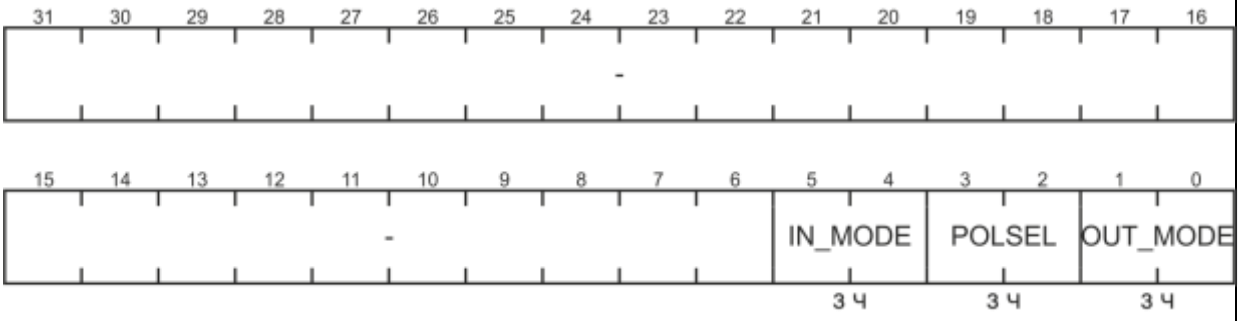


Поле	Биты	Описание
CSFB/ CSFA	3, 2/ 1, 0	Поле задания циклического воздействия на выход PWM_B/PWM_A
–	31-4	Зарезервировано

Может быть задано одно из четырех воздействий:
00 – нет реакции;
01 – значение 0 на выходе PWM_B/PWM_A;
10 – значение 1 на выходе PWM_B/PWM_A;
11 – нет реакции.

Таблица А.16.12 – Регистр управления генератором «мертвого времени» ШИМ

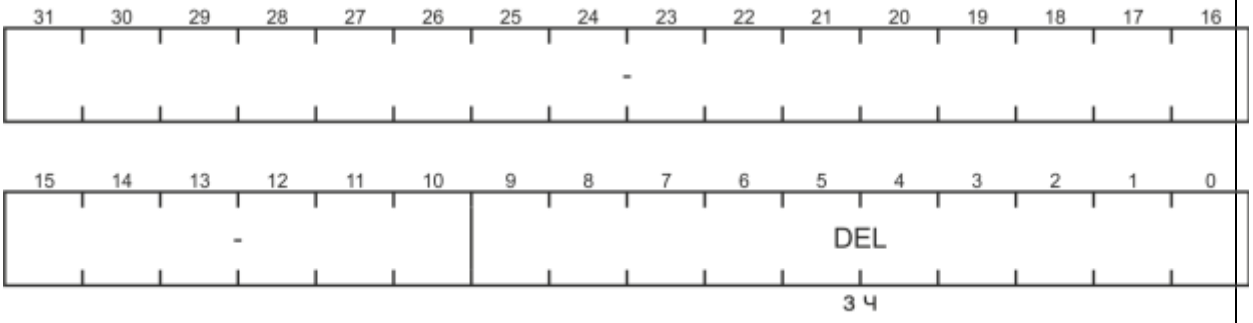
DBCTL Сброс: 00h



Поле	Биты	Описание	
IN_MODE	5, 4	Поле выбора источника для контроля по фронту и срезу. Старший бит поля управляет ключом S5, младший – ключом S4 (см. рисунок 14.12)	
		00	Сигнал PWMA используется для контроля по переднему и заднему фронтам
		01	Сигнал PWMA используется для контроля по заднему фронту, а сигнал PWMB – по переднему
		10	Сигнал PWMA используется для контроля по переднему фронту, а сигнал PWMB – по заднему
		11	Сигнал PWMB используется для контроля по переднему и заднему фронтам
POLSEL	3, 2	Поле задания полярности сигнала на выходе. Старший бит поля управляет ключом S3, а младший – ключом S2 (см. рисунок 14.12)	
		00	Инверсия запрещена
		01	Инверсия только на выводе PWM_A
		10	Инверсия только на выводе PWM_B
		11	Инверсия на выводах PWM_A и PWM_B
OUT_MODE	1, 0	Поле выбора фронта, для которого включена задержка («мертвое время»). Старший бит поля управляет ключом S1, а младший – ключом S0 (см. рисунок 14.12)	
		00	Не задано
		01	Задний фронт PWMB
		10	Передний фронт PWMA
		11	Передний фронт PWMA и задний фронт PWMB
–	31-6	Зарезервировано	

Таблица А.16.13 – Регистр управления «мертвым временем»

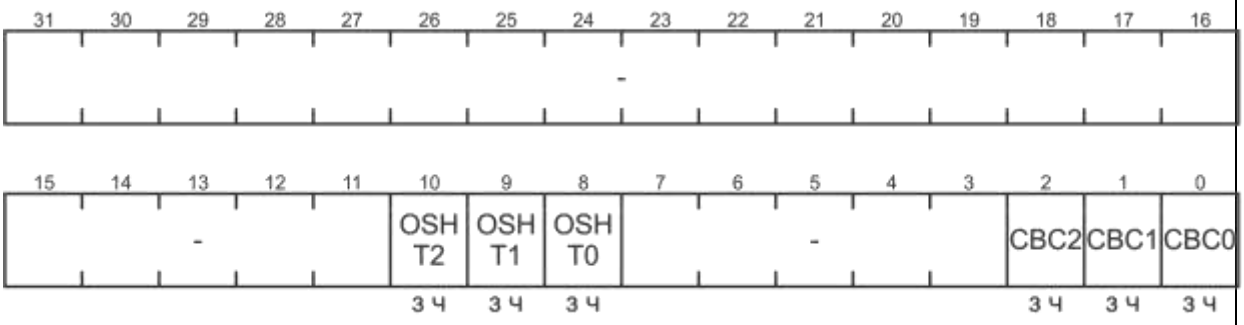
DBRED Сброс: 00h
DBFED Сброс: 00h



Поле	Биты	Описание
DEL (DBRED)	9-0	Величина задержки переднего фронта для генератора «мертвого времени» ШИМ (в периодах тактового сигнала TBCLK)
DEL (DBFED)	9-0	Величина задержки заднего фронта для генератора «мертвого времени» ШИМ (в периодах тактового сигнала TBCLK)
–	31-10	Зарезервировано

Таблица А.16.14 – Регистр источника сигнала аварии

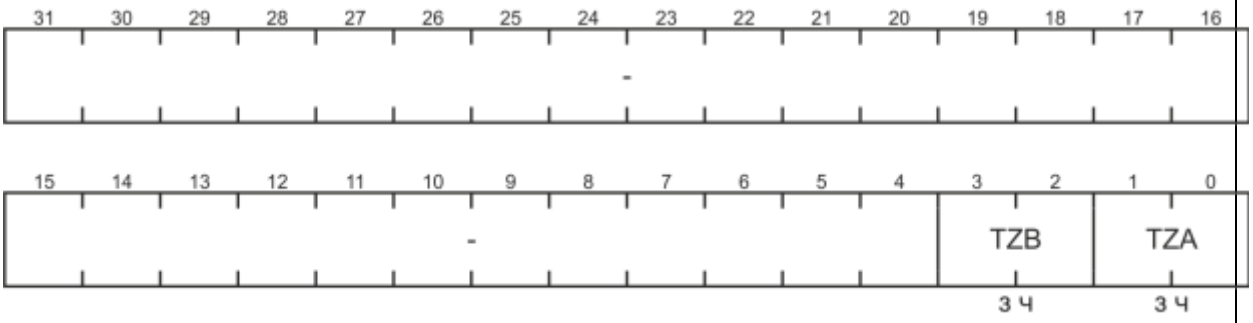
TZSEL Сброс: 00h



Поле	Бит	Описание	
OSHTn	10-8	Бит разрешения источника сигнала аварии с вывода PWM_TZn в однократном режиме	
		0	Запрещено
		1	Разрешено
CBCn	2-0	Бит разрешения источника сигнала аварии с вывода PWM_TZn в циклическом режиме	
		0	Запрещено
		1	Разрешено
–	31-11, 7-3	Зарезервировано	
Примечание – n – порядковый номер от 0 до 5.			

Таблица А.16.15 – Регистр управления детектором сигнала аварии

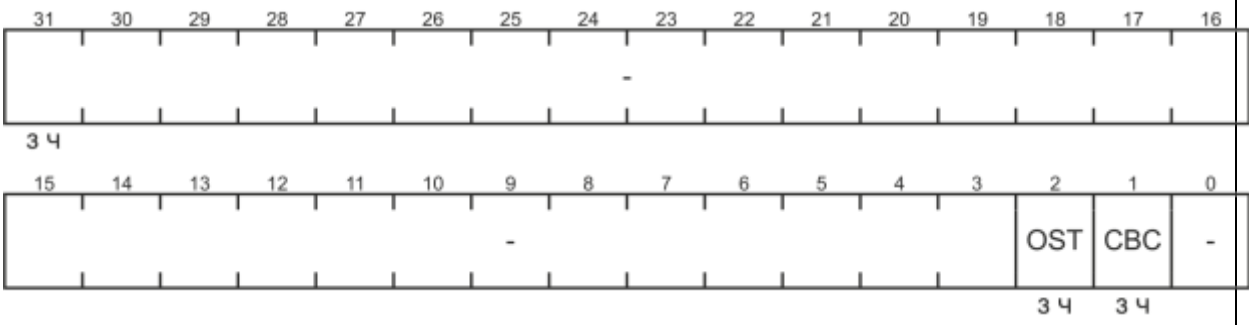
TZCTL Сброс: 00h



Поле	Биты	Описание	
TZB/ TZA	3, 2/ 1, 0	Поле задания поведения вывода PWM_B/PWM_A в случае получения сигнала аварии. Источник сигнала аварии при этом определяется регистром TZSEL	
		00	Переключение в третье состояние
		01	Переключение в единицу
		10	Переключение в ноль
		11	Нет действий
–	31-4	Зарезервировано	

Таблица А.16.16 – Регистр маски прерываний детектора сигнала аварии

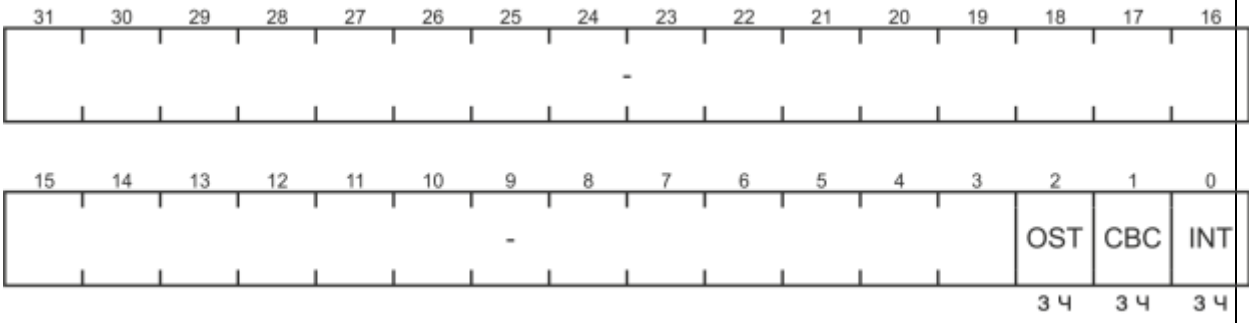
TZEINT Сброс: 00h



Поле	Бит	Описание	
OST	2	Бит разрешения генерации прерывания в однократном режиме обработки аварии	
		0	Запрещено
		1	Разрешено
CBC	1	Бит разрешения генерации прерывания в циклическом режиме обработки аварии	
		0	Запрещено
		1	Разрешено
–	31-3, 0	Зарезервировано	

Таблица А.16.17 – Регистр флагов прерываний детектора сигнала аварии

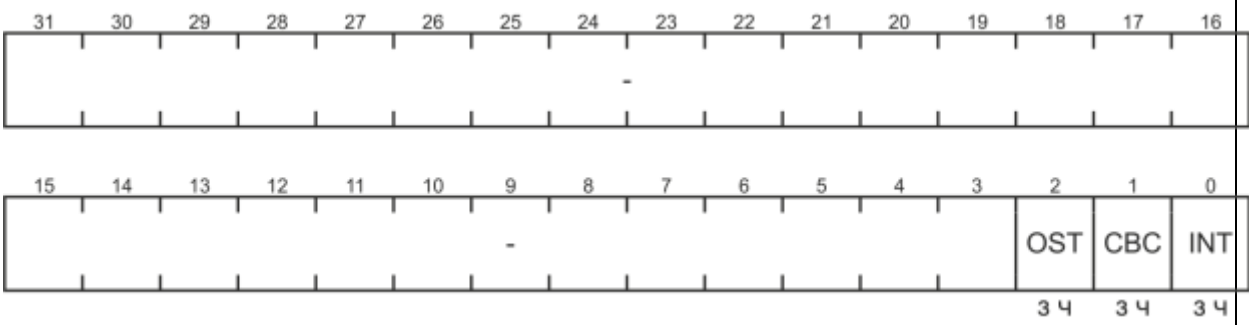
TZFLG Сброс: 00h



Поле	Бит	Описание
OST	2	Флаг прерывания в однократном режиме обработки аварии
		0 Нет прерывания
		1 Запрос на прерывание
		При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
CBC	1	Флаг прерывания в циклическом режиме обработки аварии
		0 Нет прерывания
		1 Запрос на прерывание
		При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
INT	0	Флаг внешнего прерывания NVIC
		0 Нет прерывания
		1 Запрос на прерывание
		Если флаг был сброшен, а один из флагов CBC или OST установлен, флаг установится снова
–	31-3	Зарезервировано

Таблица А.16.18 – Регистр сброса флагов прерываний детектора сигнала аварии

TZCLR Сброс: 00h

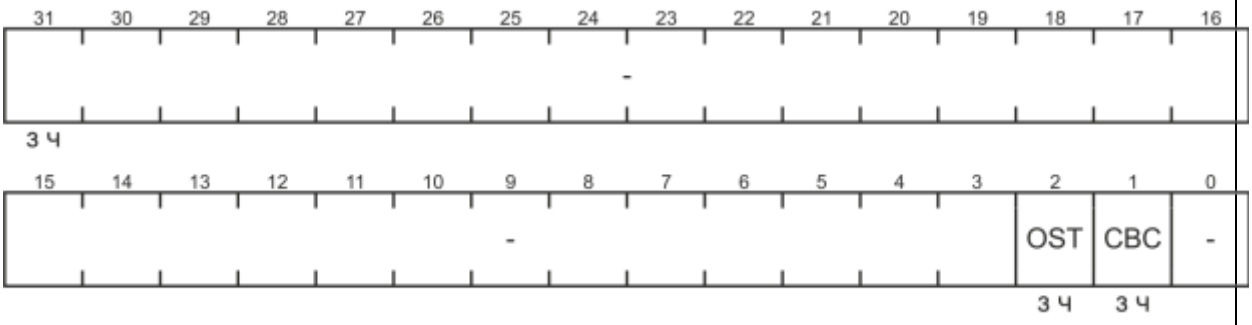


Окончание таблицы А.16.18

Поле	Бит	Описание
OST	2	Бит сброса флага прерывания в однократном режиме обработки аварии. Запись единицы сбрасывает бит OST в регистре TZFLG
CBC	1	Бит сброса флага прерывания в циклическом режиме обработки аварии. Запись единицы сбрасывает бит CBC в регистре TZFLG
INT	0	Бит сброса флага внешнего прерывания NVIC. Запись единицы сбрасывает бит INT в регистре TZFLG
–	31-3	Зарезервировано

Таблица А.16.19 – Регистр программной эмуляции сигнала аварии

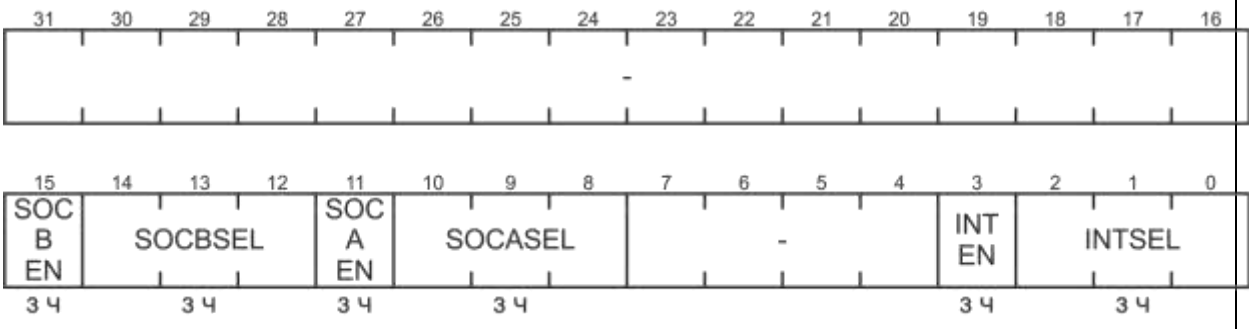
TZFRC Сброс: 00h



Поле	Бит	Описание
OST	2	Бит программной генерации сигнала аварии в однократном режиме. Запись единицы устанавливает бит OST в регистре TZFLG
CBC	1	Бит программной генерации сигнала аварии в циклическом режиме. Запись единицы устанавливает бит CBC в регистре TZFLG
–	31-3, 0	Зарезервировано
Примечание – Чтение битов OST и CBC всегда возвращает нули.		

Таблица А.16.20 – Регистр источника триггера событий

ETSEL Сброс: 00h



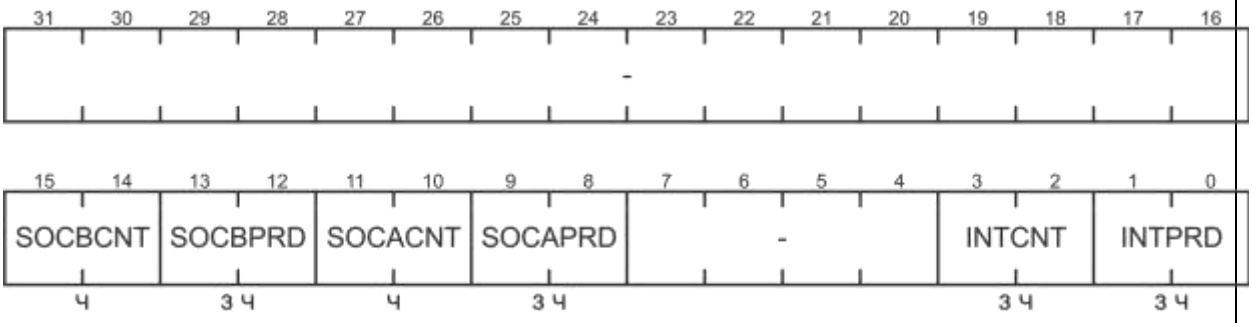
Поле	Биты	Описание	
SOCBEN/ SOCAEN	15/ 11	Бит разрешения генерации внешнего сигнала PWM_SOCB/ PWM_SOCA для запуска АЦП	
		0	Запрещено
		1	Разрешено

Окончание таблицы А.16.20

Поле	Биты	Описание	
SOCBSEL/ SOCASEL/ INTSEL	14-12/ 10-8/ 2-0	Поле выбора события, по которому будет сформирован импульс PWM_SOCB/PWM_SOCA/PWM_INT	
		000	Зарезервировано
		001	CTR = Zero
		010	CTR = PRD
		011	Зарезервировано
		100	CTR = CMPA при счете вверх
		101	CTR = CMPA при счете вниз
		110	CTR = CMPB при счете вверх
		111	CTR = CMPB при счете вниз
INTEN	3	Бит разрешения генерации внешнего прерывания PWM_INT	
		0	Запрещено
		1	Разрешено
–	31-16, 7-4	Зарезервировано	

Таблица А.16.21 – Регистр предделителя триггера событий

ETPS Сброс: 00h



Поле	Биты	Описание	
SOCBCNT/ SOCACNT	15, 14/ 11, 10	Счетчик событий сигнала PWM_SOCB/PWM_SOCA	
		00	Не было ни одного события
		01	Одно событие
		10	Два события
		11	Три события
SOCBPRD/ SOCAPRD	13, 12/ 9, 8	Поле задания количества событий, заданных полем SOCBSEL/SOCASEL регистра ETSEL, по которым будет сформирован сигнал запуска АЦП PWM_SOCB/PWM_SOCA. Для разрешения генерации сигнала нужно установить бит SOCBEN/SOCAEN регистра ETSEL. Сигнал будет сформирован, даже если флаг SOCB/SOCA (регистр ETFLG) предыдущего сигнала не был сброшен. Как только сигнал PWM_SOCB/PWM_SOCA отправлен, счетчик SOCBCNT/SOCACNT автоматически сбрасывается	
		0 0	Выдача сигнала PWM_SOCBA/PWM_SOCA запрещена

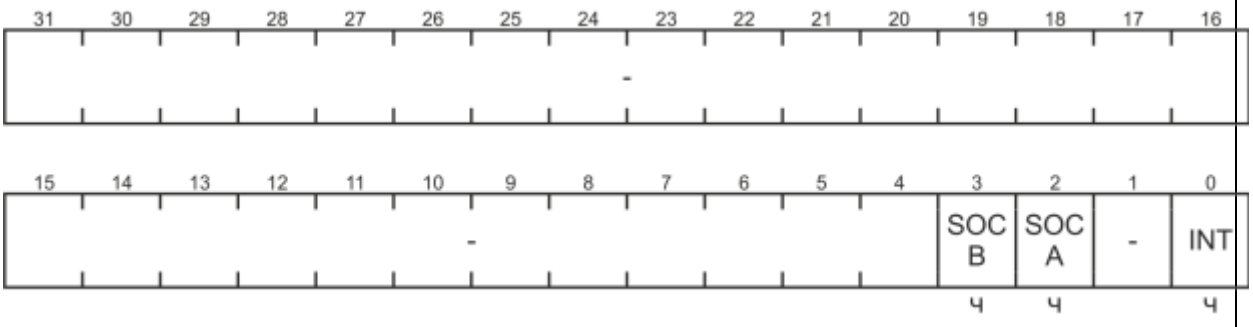
		0 1	По первому событию (SOCBCNT/SOCACNT = 01)
		1 0	По второму событию (SOCBCNT/SOCACNT = 10)
		1 1	По третьему событию (SOCBCNT/SOCACNT = 11)

Окончание таблицы А.16.21

Поле	Биты	Описание	
INTCNT	3, 2	Значение счетчика событий прерываний	
		00	Не было ни одного события
		01	Одно событие
		10	Два события
		11	Три события
		Счетчик автоматически сбрасывается, когда сформировано прерывание и перестает считать, когда достигает значения INTPRD	
INTPRD	1, 0	Поле задания количества событий, заданных полем INTSEL регистра ETSEL, по которым будет сформировано внешнее прерывание PWM_INT. Для разрешения генерации прерывания нужно установить бит INTEN в регистре ETSEL. Если флаг прерывания INT (регистр ETFLG) установлен от предыдущего прерывания, то текущее прерывание не будет активировано до сброса этого флага (сбрасывается записью единицы в бит INT регистра ETCLR). Такой механизм позволяет обрабатывать одно прерывание, в то время как другое ждет своей очереди	
		00	Прерывания по каждому событию
		01	Прерывания по первому событию (INTCNT = 01)
		10	Прерывания по второму событию (INTCNT = 10)
		11	Прерывания по третьему событию (INTCNT = 11)
–	31-16, 7-4	Зарезервировано	

Таблица А.16.22 – Регистр флагов триггера событий

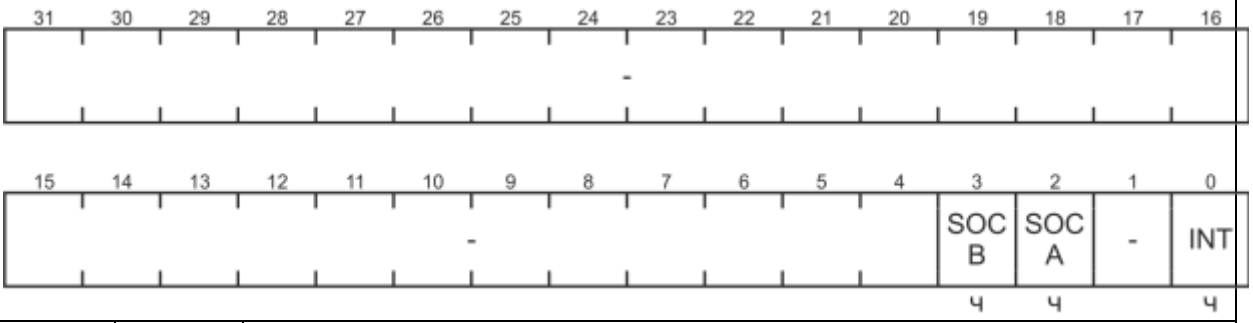
ETFLG Сброс: 00h



Поле	Бит	Описание	
SOCB/ SOCA/ INT	3/ 2/ 0	Флаг внешнего сигнала АЦП PWM_SOCB/PWM_SOCA/PWM_INT	
		0	Не установлен или сброшен
		1	Установлен
–	31-4, 1	Зарезервировано	

Таблица А.16.23 – Регистр сброса флагов триггера событий

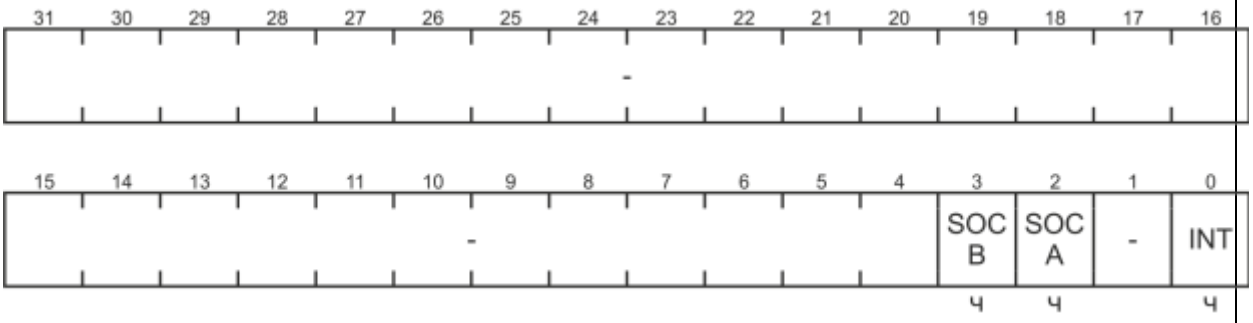
ETCLR Сброс: 00h



Поле	Бит	Описание	
1	2	3	
SOCB/ SOCA/ INT	3/ 2/ 0	Бит сброса флага SOCB/SOCA/INT в регистре ETFLG	
		0	Нет действий
		1	Запись единицы сбрасывает флаг
–	31-4, 1	Зарезервировано	

Таблица А.16.24 – Регистр программной эмуляции флагов триггера событий

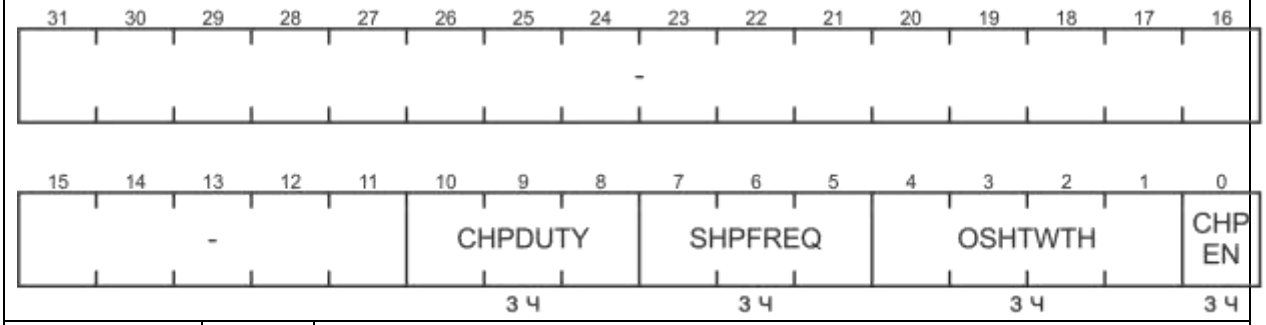
ETFRC Сброс: 00h



Поле	Бит	Описание	
SOCB/ SOCA/ INT	3/ 2/ 0	Бит установки флага SOCB/SOCA/INT в регистре ETFLG	
		0	Нет действий
		1	Запись единицы устанавливает флаг
–	31-4, 1	Зарезервировано	

Таблица А.16.25 – Регистр управления модулятором

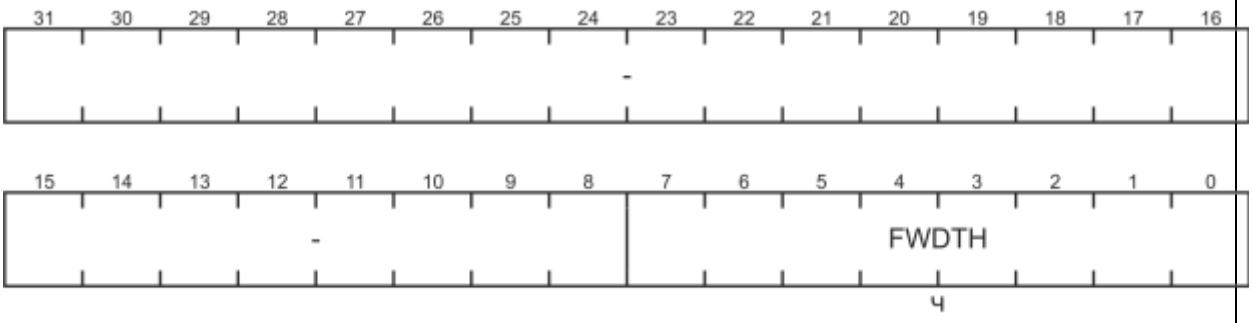
PCCTL Сброс: 00h



Поле	Биты	Описание	
CHPDUTY	10-8	Поле задания скважности второго и последующих импульсов	
		000	1/8 (13,5 %)
		001	2/8 (25,0 %)
		…	…
		110	7/8 (87,5 %)
		111	Зарезервировано
SHPFREQ	7-5	Поле выбора делителя частоты синхронизации для задания частоты второго и последующих импульсов	
		000	1
		001	1/2
		…	…
		110	1/7
		111	1/8
OSHTWTH	4-1	Поле задания ширины первого импульса	
		0h	1 × fclk/8
		1h	2 × fclk/8
		…	…
		Eh	15 × fclk/8
		Fh	16 × fclk/8
CHPEN	0	Бит разрешения работы модулятора	
		0	Запрещено
		1	Разрешено
–	31-11	Зарезервировано	

Таблица А.16.26 – Регистр ширины фильтрации

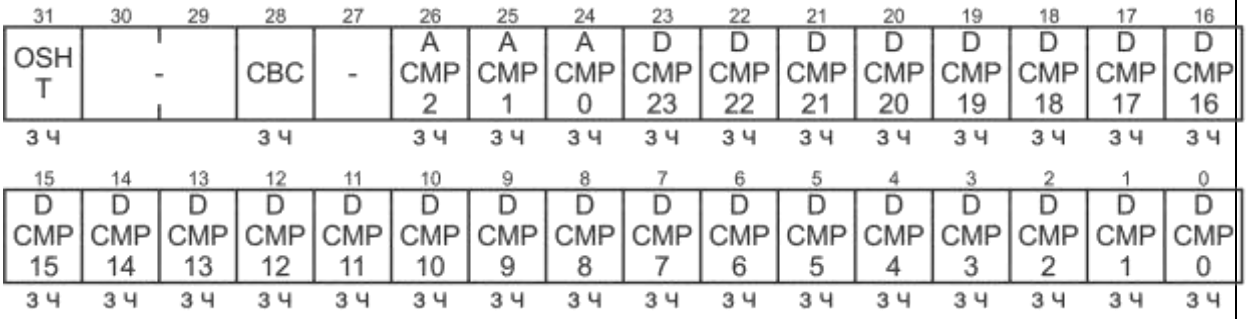
FWDTH Сброс: 00h



Поле	Биты	Описание	
FWDTH	7-0	Поле задания ширины фильтрации коротких импульсов (от 0 до 25,6 мкс с шагом 0,1 мкс)	
		0h	Фильтр выключен
		1h	Фильтр 0,1 мкс
		…	…
		Fh	25,6 мкс
–	31-8	Зарезервировано	

Таблица А.16.27 – Регистр источника сигнала события удержания

HDSEL Сброс: 00h



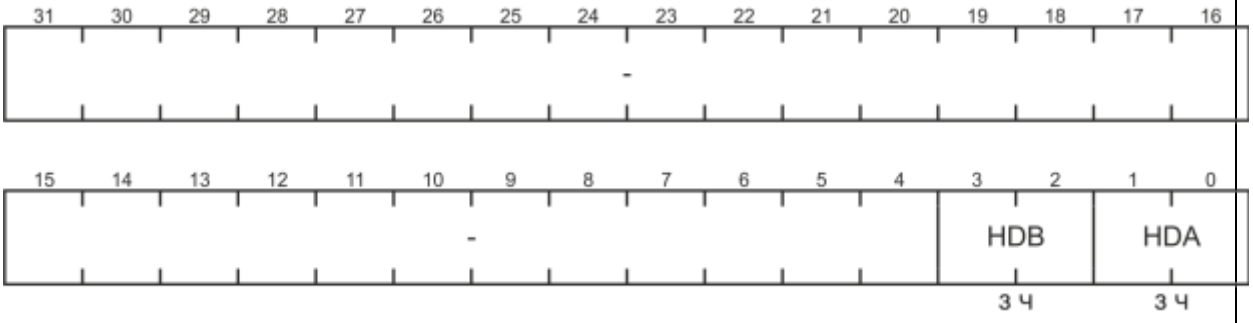
Поле	Биты	Описание	
OSHT	31	Бит разрешения события по источнику DCMP/ACMP в однократном режиме обработки аварии	
		0	Запрещено
		1	Разрешено
CBC	28	Бит разрешения события по источнику DCMP/ACMP в циклическом режиме обработки аварии	
		0	Запрещено
		1	Разрешено
ACMP2	26	Бит включения аналогового компаратора 2	
		0	Выключен
		1	Включен

Окончание таблицы А.16.27

Поле	Биты	Описание	
ACMP1	25	Бит включения аналогового компаратора 1	
		0	Выключен
		1	Включен
ACMP0	24	Бит включения аналогового компаратора 0	
		0	Выключен
		1	Включен
DCMP23–DCMP0	23-0	Бит выбора цифрового компаратора (0 – 23) блока АЦП, с выхода которого берется сигнал для формирования события удержания	
		0	Не выбран
		1	Выбран
–	30, 29, 27	Зарезервировано	

Таблица А.16.28 – Регистр управления детектором событий удержания

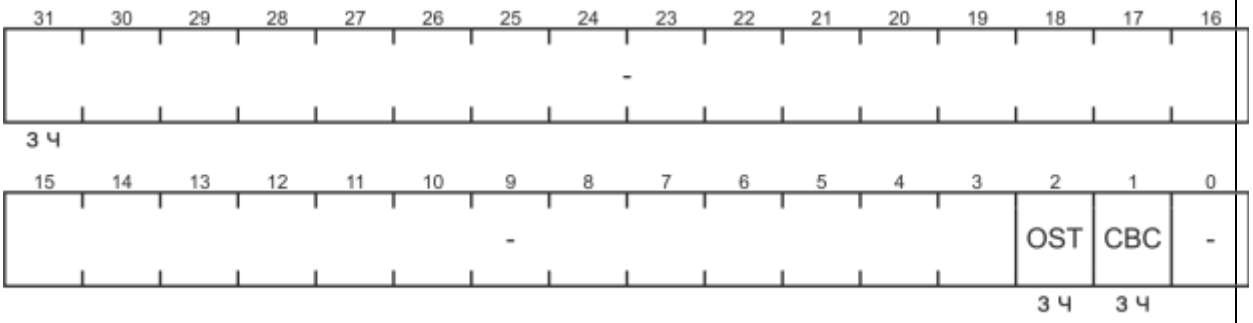
HDCTL Сброс: 00h



Поле	Биты	Описание	
HDB/ HDA	3, 2/ 1, 0	Поле задания поведения сигнала PWMB/PWMA в случае сбоя (аварии). (Источник сбоя определяется регистром HDSEL)	
		00	Зарезервировано
		01	Переключается в состояние единицы
		10	Переключается в состояние нуля
		11	Остается без изменений
–	31-4	Зарезервировано	

Таблица А.16.29 – Регистр маски прерывания порогового выключателя

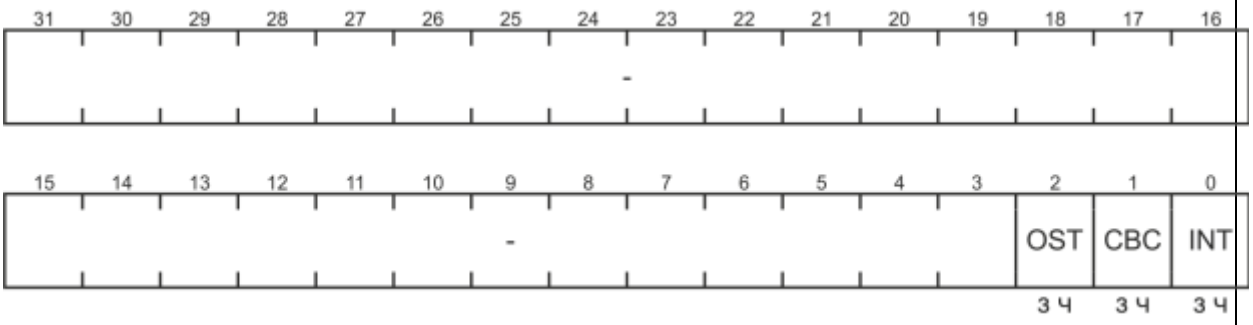
HDEINT Сброс: 00h



Поле	Бит	Описание
OST	2	Бит разрешения генерации прерывания в однократном режиме
		0 Запрещено
		1 Разрешено
CBC	1	Бит разрешения генерации прерывания в циклическом режиме
		0 Запрещено
		1 Разрешено
–	31-3, 0	Зарезервировано

Таблица А.16.30 – Регистр флагов прерывания порогового выключателя

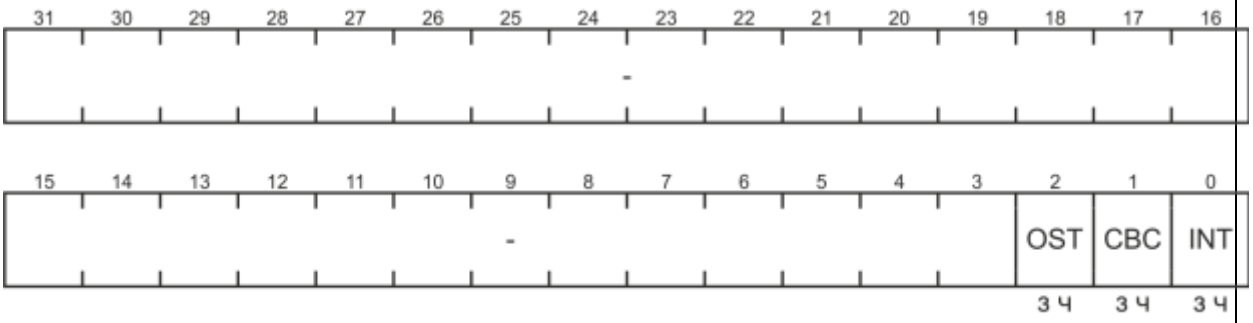
HDFLG Сброс: 00h



Поле	Бит	Описание
OST, CBC	2, 1	Флаг прерывания в однократном режиме, флаг прерывания в циклическом режиме
		0 Нет прерывания
		1 Запрос на прерывание
		При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
INT	0	Флаг внешнего прерывания NVIC
		0 Нет прерывания
		1 Запрос на прерывание
		Если флаг был сброшен, а один из флагов CBC или OST установлен, флаг установится снова
–	31-3	Зарезервировано

Таблица А.16.31 – Регистр сброса флагов порогового выключателя

HDCLR Сброс: 00h

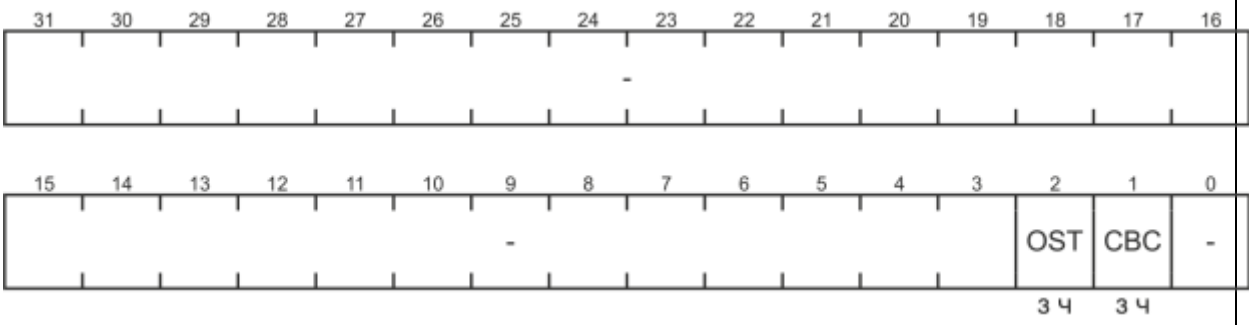


Поле	Бит	Описание
OST	2	Бит сброса флага прерывания в однократном режиме
CBC	1	Бит сброса флага прерывания в циклическом режиме
INT	0	Бит сброса флага внешнего прерывания NVIC
–	31-3	Зарезервировано

Примечание – Запись единицы в бит регистра сбрасывает соответствующий бит в регистре HDFLG.

Таблица А.16.32 – Регистр программной активации порогового выключателя

HDFRC Сброс: 00h

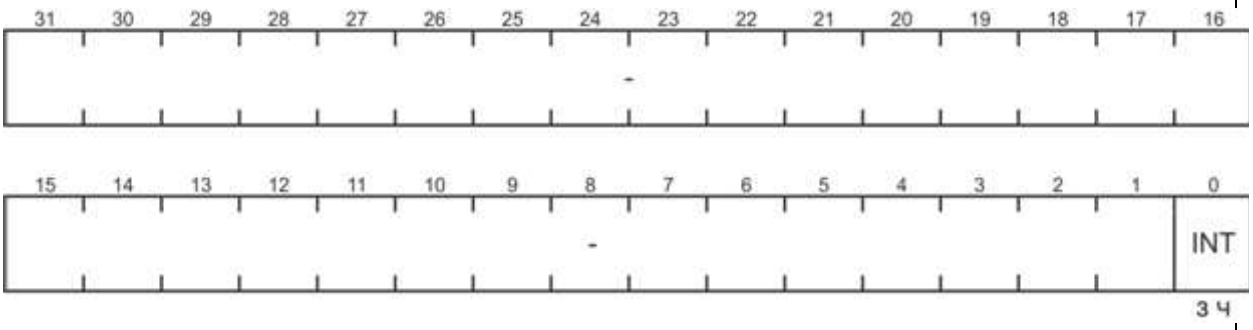


Поле	Бит	Описание	
OST	2	Бит активации порогового выключателя в однократном режиме обработки аварии	
		0	Нет действий
		1	Запись единицы активирует выключатель и устанавливает флаг OST в регистре HDFLG
CBC	1	Бит активации порогового выключателя в циклическом режиме обработки аварии	
		0	Нет действий
		1	Запись единицы активирует выключатель и устанавливает флаг CBC в регистре HDFLG
–	31-3, 0	Зарезервировано	

Таблица А.16.33 – Регистры сброса прерываний порогового выключателя, детектора событий аварии и таймера блока ШИМ

HDINTCLR
TZINTCLR
INTCLR

Сброс: 00h

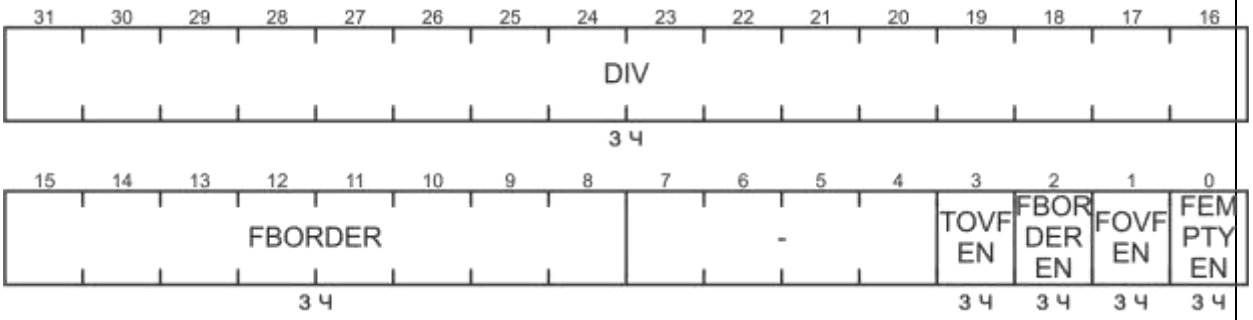


Поле	Бит	Описание
INT	0	Бит сброса прерывания. Запись единицы в бит сбрасывает запрос прерывания. Запись в бит должна производиться программой обработки прерывания, во избежание повторного запуска программы обслуживания прерывания
–	31-1	Зарезервировано

A.17 Регистры блока HSIO

Таблица А.17.1 – Регистр управления модулем HSI

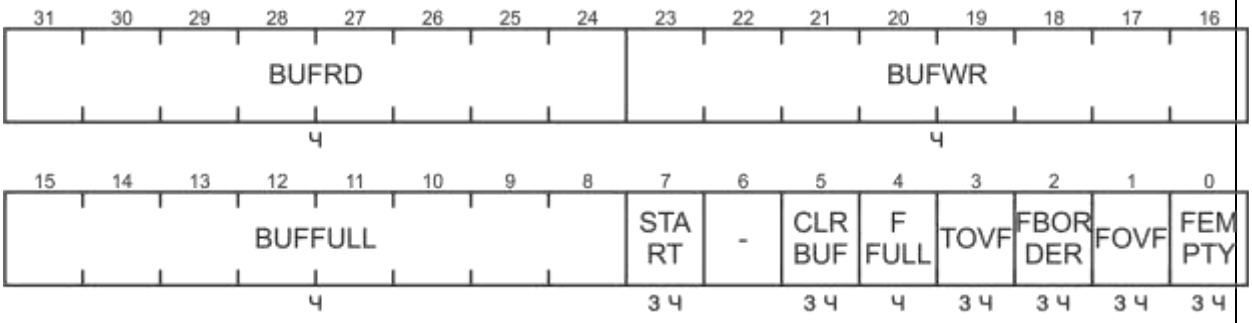
HSICON Сброс: 00h



Поле	Бит	Описание
DIV	31-16	Коэффициент делителя. Делитель включен, если значение DIV не равно нулю
FBORDER	15-8	Верхняя граница буфера FIFO. Значение указывает на количество зафиксированных событий, по достижении которого будет сгенерировано прерывание
TOVFEN	3	Бит разрешения прерывания при переполнении таймера
		0 Запрещено
		1 Разрешено
FBORDER EN	2	Бит разрешения прерывания при достижении количества зафиксированных событий значения, указанного в поле FBORDER
		0 Запрещено
		1 Разрешено
FOVFEN	1	Бит разрешения прерывания при переполнении буфера FIFO
		0 Запрещено
		1 Разрешено
FEMPTY EN	0	Бит разрешения прерывания при полном опустошении буфера FIFO
		0 Запрещено
		1 Разрешено
–	7-4	Зарезервировано

Таблица А.17.2 – Регистр состояния модуля HSI

HSISTAT Сброс: 00h

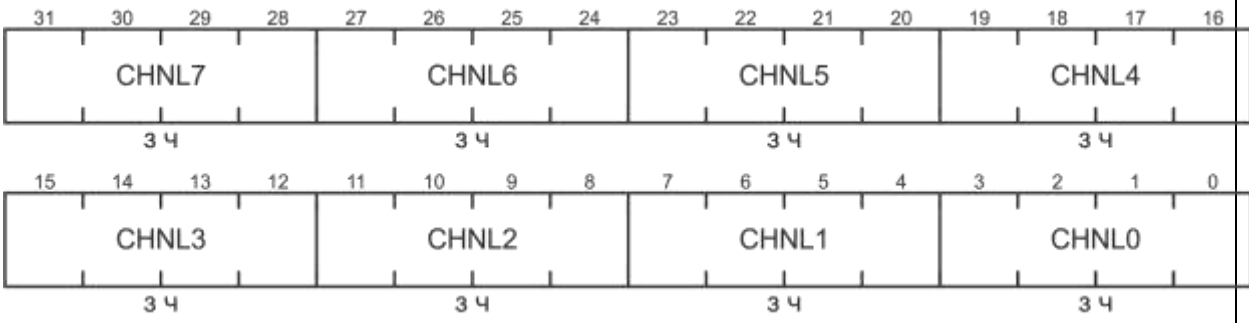


Поле	Бит	Описание
BUFRD	31-24	Указатель на регистр FIFO, начиная с которого будет произведено чтение буфера. Содержит порядковый номер регистра буфера
BUFWR	23-16	Указатель на регистр FIFO, начиная с которого будет произведена запись в буфер. Содержит порядковый номер регистра
BUFFULL	15-8	Счетчик количества непрочитанных регистров FIFO. BUFFULL = 00h указывает на то, что буфер FIFO пуст, а BUFFULL = FFh – на то, что заполнены все 256 регистров буфера FIFO
START	7	Бит запуска модуля HSI
		0 – Выключен. Таймер остановлен, мониторинг входов не осуществляется
		1 – Включен. Таймер функционирует, ведется мониторинг входов
CLRBUF	5	Бит очистки FIFO. Запись единицы в этот бит приводит к полному опустошению буфера FIFO. Состояние бита не изменяется
FFULL	4	Флаг полного заполнения буфера FIFO
		0 – В буфере есть свободные регистры
		1 – Свободных регистров нет
TOVF	3	Флаг переполнения таймера
		0 – Нет переполнения
		1 – Таймер достиг максимального значения
FBORDER	2	Флаг достижения верхней границы FIFO. Не устанавливается до тех пор, пока количество зафиксированных событий меньше значения в поле FBORDER регистра HSICON
FOVF	1	Флаг переполнения буфера FIFO. Устанавливается в случае попытки записи в буфер FIFO информации об одновременно обнаруженных событиях, количество которых превышает количество свободных регистров в буфере
		0 – Переполнение не произошло
		1 – Произошло переполнение
FEMPTY	0	Флаг опустошения буфера FIFO
		0 – Буфер не пуст
		1 – Буфер пуст

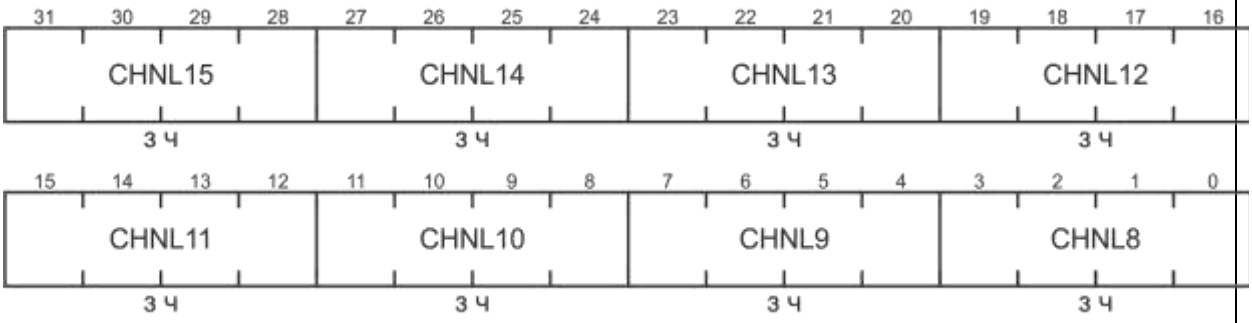
–	6	Зарезервировано

Таблица А.17.3 – Регистр событий модуля HSI

HSICHNLCFG0 Сброс: 00h



HSICHNLCFG1 Сброс: 00h



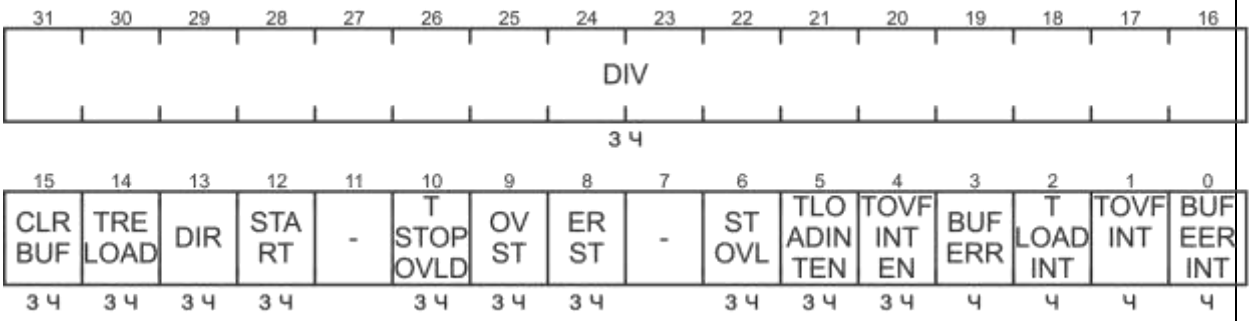
Поле	Бит	Описание	
CHNLx	31-28, 27-24, 23-20, 19-16, 15-12, 11-8, 7-4, 3-0	Поле задания ожидаемого события для сигнала на входе HSIx (x от 0 до 15)	
		0h	Мониторинг выключен
		1h	Все положительные фронты
		2h	Все отрицательные фронты
		3h	Все фронты
		4h	Каждый второй положительный фронт
		5h	Каждый четвертый положительный фронт
		6h	Каждый шестой положительный фронт
		7h	Каждый восьмой положительный фронт
		8h	Каждый второй отрицательный фронт
CHNLx	31-28, 27-24, 23-20, 19-16, 15-12, 11-8, 7-4, 3-0	9h	Каждый четвертый отрицательный фронт
		Ah	Каждый шестой отрицательный фронт
		Bh	Каждый восьмой отрицательный фронт
		Ch	Каждый второй фронт
		Dh	Каждый четвертый фронт
		Eh	Каждый шестой фронт
		Fh	Каждый восьмой фронт

Таблица А.17.4 – 32-разрядные регистры

Мнемоника	Назначение	Сброс
HSITIMER	Регистр счетчика таймера модуля HSI	00000000h
HSIFIFO	Регистр чтения буфера FIFO модуля HSI. Только чтение	00000000h

Таблица А.17.5 – Регистр управления модулем HSO

HSOCON Сброс: 00h



Поле	Бит	Описание	
DIV	31-16	Коэффициент делителя. Делитель включен, если значение DIV не равно нулю	
CLRBUF	15	Бит очистки буфера CAM. Запись единицы в этот бит приводит к полному опустошению буфера CAM. Состояние бита не изменяется	
TRELOAD	14	Бит управления перезагрузкой таймера	
		0	По достижении значения регистра HSOTIMEROV таймер не сбрасывается, а продолжает счет
		1	По достижении значения регистра HSOTIMEROV таймер сбрасывается в ноль
DIR	13	Направление счета таймера	
		0	Вверх (инкремент)
		1	Вниз (декремент)
START	12	Бит запуска модуля HSO	
		0	Выключен. Таймер остановлен, мониторинг входов не осуществляется
		1	Включен. Таймер функционирует, ведется мониторинг входов
TSTOP OVLD	10	Бит управления работой таймера при достижении значения, находящегося в регистре HSOTIMEROV	
		0	Не останавливается
		1	Останавливается
OVST	9	Бит разрешения остановки таймера при опустошении	
		0	Запрещено
		1	Разрешено
ERST	8	Бит управления таймером	
		0	Нет действий
		1	Таймер остановится, если будет обнаружена ошибка буфера
STOVL	6	Бит разрешения прерывания при переполнении таймера	
		0	Запрещено
		1	Разрешено
TLOAD INTEN	5	Бит разрешения прерывания при переполнеии таймера	
		0	Запрещено
		1	Разрешено
TOVF	4	Бит разрешения прерывания при ошибки буфера	

INTEN		0	Запрещено
		1	Разрешено

Окончание таблицы А.17.5

Поле	Биты	Описание	
BUFERR	3	Флаг ошибки буфера	
TLOAD INT	2	Бит разрешения прерывания при полном опустошении буфера FIFO	
		0	Запрещено
		1	Разрешено
TOVF INT	1	Бит разрешения прерывания при переполнении таймера	
		0	Запрещено
		1	Разрешено
BUFERR INT	0	Бит разрешения прерывания при переполнении таймера	
		0	Запрещено
		1	Разрешено
–	11, 7	Зарезервировано	

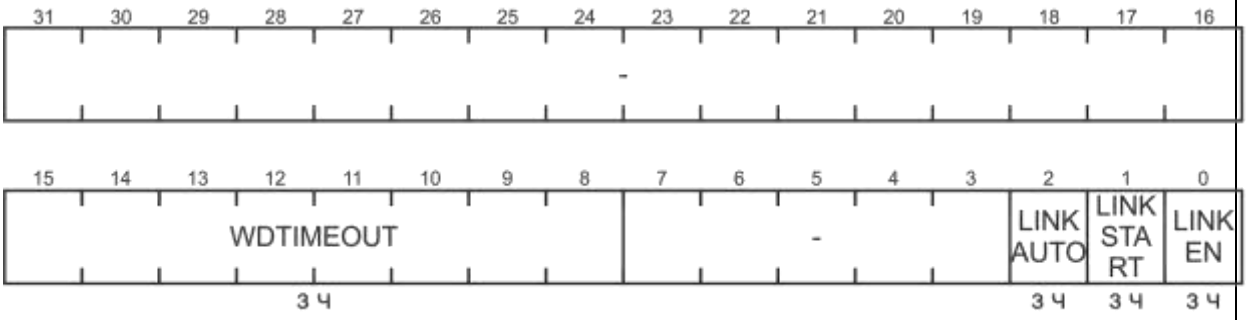
Таблица А.17.6 – Регистр событий модуля HSI

Мнемоника	Назначение	Сброс
HSOTIMEROV	Регистр перезагрузки счетчика таймера модуля HSO	00000000h
HSOTIMER	Регистр счетчика таймера модуля HSO	00000000h

A.18 Регистры контроллера SpaceWire

Таблица А.18.1 – Регистр управления

CON Сброс: FF00h

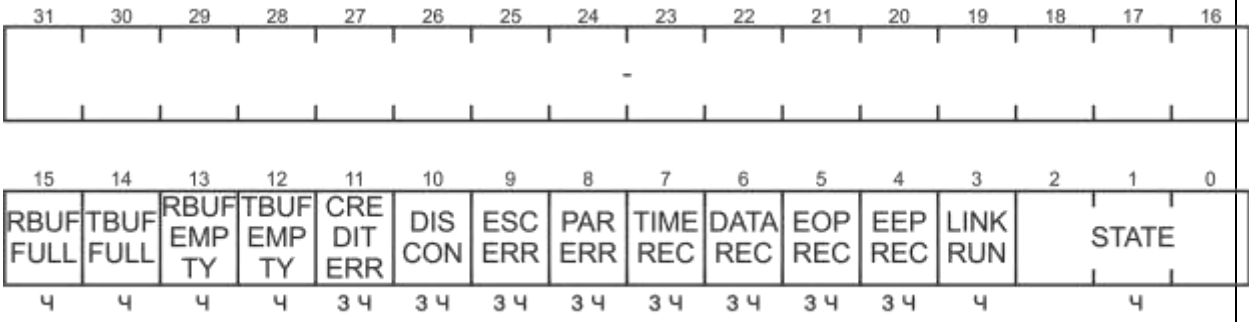


Поле	Бит	Описание	
WDTIMEOUT	15-8	Временной коэффициент. Поле устанавливает значение времени потери соединения 800 нс, состояния сброса 6,4 мкс и других состояний 12,8 мкс. Значение временного коэффициента может быть получено из формулы: $WDTIMEOUT_{DEC} = k \times F_{XTAL1}$, (А.18.1) где $WDTIMEOUT_{DEC}$ – значение в десятичном формате, F_{XTAL1} – значение внешней частоты, МГц, k – коэффициент. При выключенном внутреннем делителе внешней частоты (сброшен бит SLOW регистра CLKEN) k = 0,4. При включенном (бит SLOW установлен) – k = 0,2. После получения значения в десятичном формате, его следует перевести в шестнадцатеричный формат и записать в поле WDTIMEOUT. Так, например, при внешней частоте микроконтроллера, равной 20 МГц, скорость передачи данных составит 10 Мбит/с. При выключенном делителе частоты $WDTIMEOUT_{DEC} = 0,4 \times 20 = 8$ и, таким образом, в поле WDTIMEOUT следует записать значение 08h	
LINKAUTO	2	Бит автоматического установления связи. Устанавливается, если нужно перевести модуль в режим ожидания передачи данных от другого устройства	
		0	Нет связи
		1	Модуль ожидает NULL-символ
LINKSTART	1	Бит установления связи. Устанавливается, если модуль является инициатором передачи	
		0	Нет связи
		1	Инициация передачи
LINKEN	0	Бит установки соединения	
		0	Модуль отключен от сети
		1	Модуль подключен к сети
–	31-16,	Зарезервировано	

	7-3	

Таблица А.18.2 – Регистр состояния

STAT Сброс: 3001h



Поле	Бит	Описание
RBUFFULL	15	Индикатор заполнения принимающего FIFO-буфера
		0 В буфере есть как минимум одна свободная ячейка
		1 Буфер заполнен
TBUFFULL	14	Индикатор заполнения передающего FIFO-буфера
		0 В буфере есть как минимум одна свободная ячейка
		1 Буфер заполнен
RBUFEMPTY	13	Индикатор пустоты принимающего FIFO-буфера
		0 Как минимум одна ячейка буфера заполнена
		1 Буфер пуст
TBUFEMPTY	12	Индикатор пустоты передающего FIFO-буфера
		0 Как минимум одна ячейка буфера заполнена
		1 Буфер пуст
CREDITERR	11	Бит устанавливается при ошибке кредитования буфера приемника или передатчика (неправильном определении свободного места в устройстве). Флаг сбрасывается программно
DISCON	10	Флаг потери соединения. Устанавливается при обнаружении потери соединения с сетью SpaceWire. Маскируется битом DISCONINT регистра INTMASK. Сбрасывается программно
ESCERR	9	Флаг ошибки последовательности управляющих символов. Устанавливается при обнаружении ошибки следования управляющих символов во время приема. Маскируется битом ESCINT регистра INTMASK. Сбрасывается программно
PARERR	8	Флаг ошибки паритета. Устанавливается при обнаружении ошибки четности принятой информации. Маскируется битом PARINT регистра INTMASK. Сбрасывается программно
TIMEREC	7	Флаг приема кода времени. Устанавливается, если принятый байт является кодом времени. Маскируется битом TIMEINT регистра INTMASK. Флаг сбрасывается программно
DATAREC	6	Флаг приема данных. Устанавливается по окончании приема байта данных.

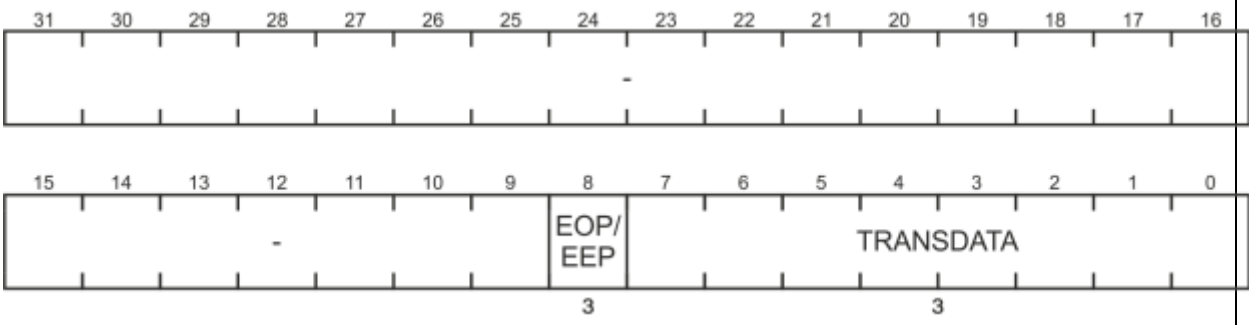
		Маскируется битом DATAINT регистра INTMASK. Сбрасывается программно

Окончание таблицы А.18.2

Поле	Бит	Описание	
EOPREC	5	Флаг приема маркера EOP. Устанавливается, если принят код 0100h, соответствующий маркеру конца пакета. Маскируется битом EOPINT. Сбрасывается программно	
EEPREC	4	Флаг приема маркера EEP. Устанавливается, если принят код 0101h, соответствующий маркеру ошибки пакета. Маскируется битом EEPINT. Сбрасывается программно	
LINKRUN	3	Индикатор соединения с сетью SpaceWire	
		0	Связь не установлена
		1	Связь установлена
STATE	2-0	Текущее состояние конечного автомата модуля	
		000	Reset
		001	Error Reset
		010	Error Wait
		011	Ready
		100	Started
		101	Connecting
		110	Run
		111	Зарезервировано
		Для ознакомления с работой конечного автомата следует обратиться к спецификации ECSS-E-ST-50-12C для SpaceWire	
–	31-16	Зарезервировано	

Таблица А.18.3 – Регистр отправляемых данных

DATAT Сброс: 00h



Поле	Бит	Описание
EOP/EEP	8	Маркер конца/ошибки пакета. Этот бит используется совместно с полем TRANSDATA, как указатель окончания передачи пакета. Таким образом, если в регистр DATAT было записано значение 00xxh, то это указывает на байт данных («xx» – байт данных). Если в регистр DATAT было записано значение: – 0100h, то это указатель конца пакета;

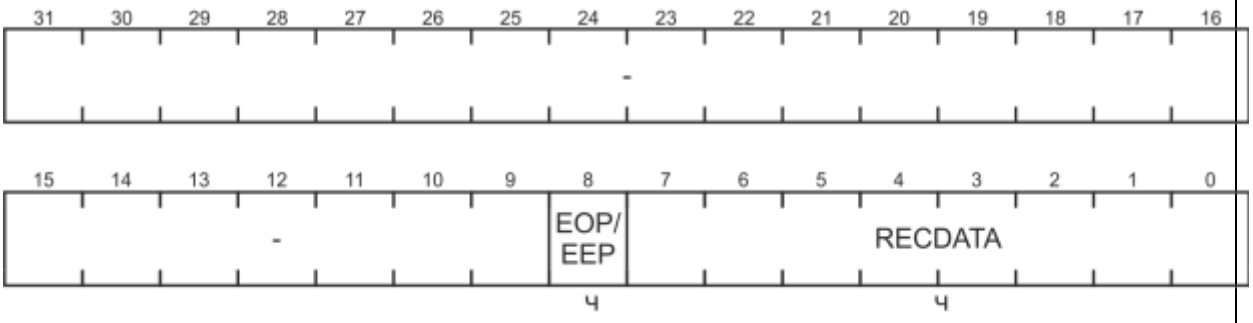
		– 0101h, то это указатель ошибки, обнаруженной передатчиком в переданных данных

Окончание таблицы А.18.3

Поле	Бит	Описание
TRANS DATA	7-0	Байт данных для передачи. При записи байт из этого поля загружается в передающий FIFO-буфер
–	31-9	Зарезервировано

Таблица А.18.4 – Регистр принятых данных

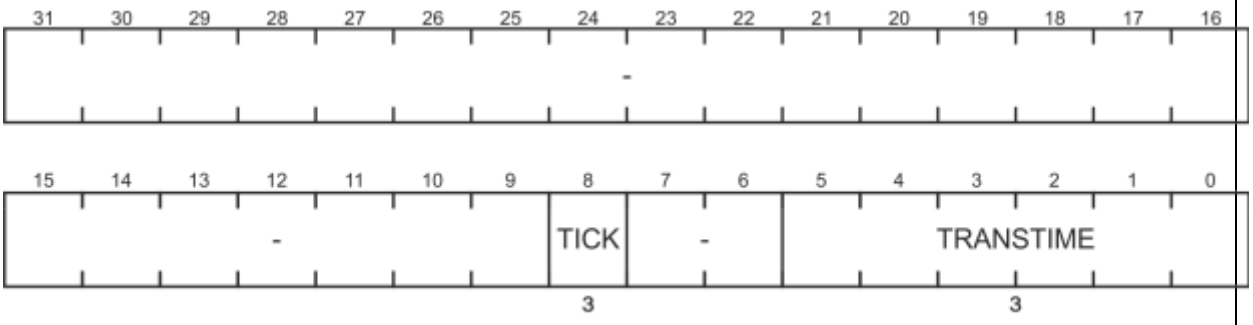
DATAR Сброс: 00h



Поле	Бит	Описание
EOP/EEP	8	Маркер конца/ошибки пакета. Этот бит используется совместно с полем RECDATA, как указатель окончания передачи пакета. Таким образом, если при обращении к регистру DATAR было прочитано значение 00xxh, то это указывает на байт данных («xx» – байт данных). Если прочитано значение: – 0100h, то это указатель конца пакета; – 0101h, то это указатель ошибки, обнаруженной передатчиком в передаваемых данных
RECDATA	7-0	Байт принятых данных. При чтении в это поле переписывается байт из принимающего FIFO-буфера
–	31-9	Зарезервировано

Таблица А.18.5 – Регистр отправляемого кода времени

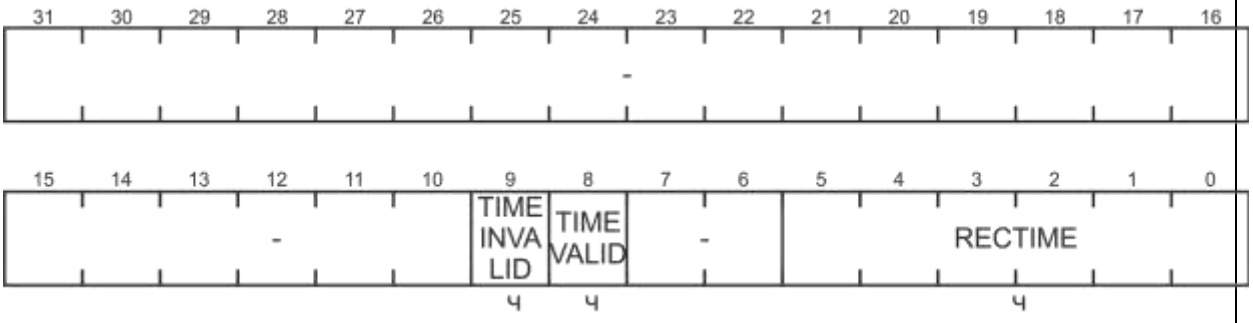
TIMET Сброс: 00h



Поле	Бит	Описание
TICK	8	Бит инициализации отправки кода времени
TRANSTIME	5-0	Поле кода времени
–	31-9, 7-6	Зарезервировано

Таблица А.18.6 – Регистр принятого кода времени

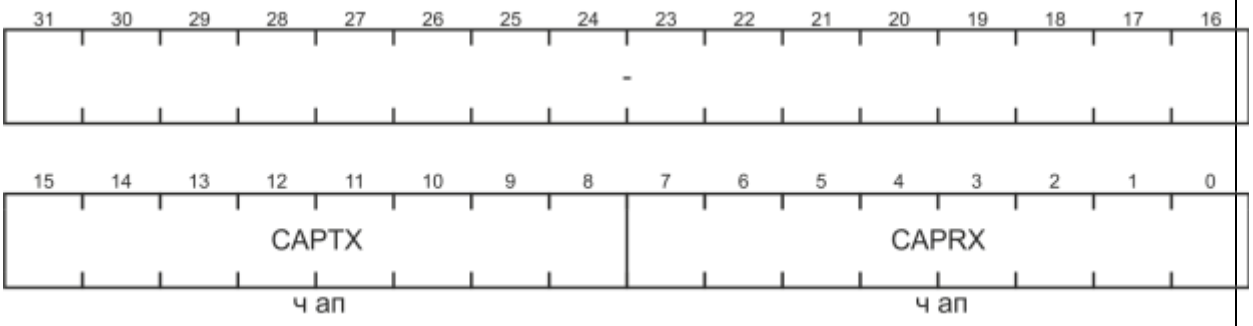
TIMER Сброс: 00h



Поле	Бит	Описание
TIMEINVALID	9	Флаг некорректного принятого кода времени
TIMEVALID	8	Флаг корректного принятого кода времени
RECTIME	5-0	Принятый код времени
–	31-10, 7, 6	Зарезервировано

Таблица А.18.7 – Регистр состояния буферов

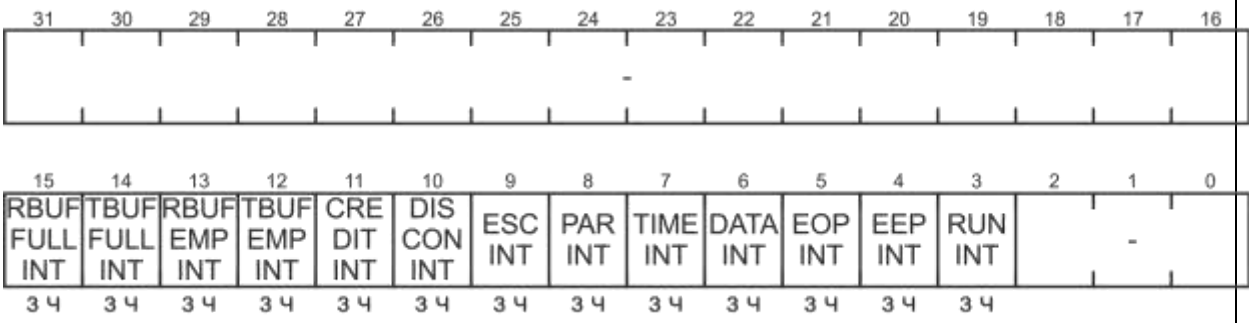
CAPSTAT Сброс: 00h



Поле	Бит	Описание
CAPTX	15-8	Число занятых ячеек (байт) передающего буфера
CAPRX	7-0	Число занятых ячеек (байт) принимающего буфера
–	31-16	Зарезервировано

Таблица А.18.8 – Регистр маски прерываний

INTMASK Сброс: 00h



Поле	Бит	Описание
RBUFFULLINT	15	Разрешает прерывание при полном заполнении буфера приемника
TBUFFULLINT	14	Разрешает прерывание при полном заполнении буфера передатчика
RBUFEMPINT	13	Разрешает прерывание, когда пустой приемный буфер принимает хотя бы один байт данных (прерывание по выходу из пустого состояния)
TBUFEMPINT	12	Разрешает прерывание, когда передающий буфер переходит из частично заполненного в пустое состояние (прерывание по опустошению)
CREDITINT	11	Разрешает прерывание при ошибке кредитования буферов приемника и передатчика
DISCONINT	10	Разрешает прерывание при потере связи
ESCINT	9	Разрешает прерывание при получении неверной последовательности управляющих символов
PARINT	8	Разрешает прерывание по ошибке четности в полученном символе
TIMEINT	7	Разрешает прерывание по получению кода времени (временной метки)
DATAINT	6	Разрешает прерывание по получению одного байта данных
EOPINT	5	Разрешает прерывание по получению пакета без ошибки
EEPINT	4	Разрешает прерывание по получению пакета с ошибкой
RUNINT	3	Разрешает прерывание по входу в состояние RUN
–	31-16, 2-0	Зарезервировано

Приложение Б (обязательное) **Карта памяти микроконтроллеров**

Области памяти микроконтроллеров, мнемоника и названия регистров приведены в таблицах Б.1 – Б.29.

Примечание – Области памяти, отмеченные как зарезервированные, не использовать. Запись по зарезервированному адресу не эффективна, а чтение возвращает случайное значение.

Таблица Б.1 – Карта области системных регистров общего назначения

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_0000h	ZERO_REG	00h	Нулевой регистр
0000_0004h	R0	00h	Регистры быстрых команд
0000_0008h	R1	00h	
0000_000Ch	R2	00h	
0000_0010h	R3	00h	
0000_0014h	R4	00h	
0000_0018h	R5	00h	
0000_001Ch	R6	00h	
0000_0020h	R7	00h	
0000_0024h	R8	00h	
0000_0028h	R9	00h	
0000_002Ch	R10	00h	
0000_0030h	R11	00h	
0000_0034h	R12	00h	
0000_0038h	R13	00h	
0000_003Ch	R14	00h	
0000_0040h	R15	00h	
0000_0044h–0000_004Fh	–	–	Зарезервировано
0000_0050h	DPP0	00h	Регистр указатель нулевой страницы данных
0000_0054h	DPP1	00h	Регистр указатель первой страницы данных
0000_0058h	DPP2	00h	Регистр указатель второй страницы данных
0000_005Ch	DPP3	00h	Регистр указатель третьей страницы данных

Таблица Б.2 – Область регистров общего назначения

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_0060h – 0000_3FFFh	Задается пользователем	00h	ОЗУ0

Таблица Б.3 – Карта области системных регистров

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4000h	PTSSEL0	00h	Регистр разрешения прерываний PTS
0000_4004h	PTSSRV0	00h	Регистр флагов прерываний PTS
0000_4008h	INT_MASK0	00h	Регистр маски прерываний
0000_400Ch	INT_PEND0	00h	Регистр флагов прерываний
0000_4010h	PTSSEL1	00h	Регистр разрешения прерываний PTS
0000_4014h	PTSSRV1	00h	Регистр флагов прерываний PTS
0000_4018h	INT_MASK1	00h	Регистр маски прерываний
0000_401Ch	INT_PEND1	00h	Регистр флагов прерываний

Таблица Б.4 – Карта области системных регистров (для адресных окон)

Мнемоническое обозначение (x – номер окна)	Адресные окна							
	Адрес 0000_xxxx							
	0	1	2	3	4	5	6	7
BUSxADDR_LO	4020h	4030h	4040h	4050h	4060h	4070h	4080h	4090h
BUSxADDR_HI	4024h	4034h	4044h	4054h	4064h	4074h	4084h	4094h
BUSxCON	4028h	4038h	4048h	4058h	4068h	4078h	4088h	4098h
Примечание – Двойные слова с адресами 0000_402Ch, 0000_403Ch, 0000_404Ch, 0000_405Ch, 0000_406Ch, 0000_407Ch, 0000_408Ch, 0000_0000_409Ch являются зарезервированными.								

Таблица Б.5 – Мнемоника и названия регистров адресных окон

Мнемоника	Сброс	Название
BUSxADDR_LO	00h	Регистр нижней границы адресного окна x
BUSxADDR_HI	00h	Регистр верхней границы адресного окна x
BUSxCON	00h	Регистр управления шиной окна x
Примечание – x является номером окна и может принимать значения от 0 до 7.		

Таблица Б.6– Карта области системных регистров

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_40A0h	ECCDATAERR	00h	Регистр адреса некорректных данных
0000_40A4h	ECCCOMERR	00h	Регистр адреса некорректной команды
0000_40A8h, 0000_40AFh	–	–	Зарезервировано
0000_40B0h	SP	0000_3D18h	Указатель стека
0000_40B4h	SP_OVRFLOW	FFFF_FFF0h	Регистр переполнения указателя стека
0000_40B8h	SP_DNFLOW	0000_000Fh	Регистр опустошения указателя стека
0000_40BCh	–	–	Зарезервировано

Таблица Б.7 – Карта области регистров таймеров T0 и T1

Мнемоническое обозначение	Таймер Адрес T0	Таймер Адрес T1	Сброс	Название регистра
VALUE_L	0000_40C0h	0000_40E0h	0h	Регистр младшего слова значения системного таймера
VALUE_H	0000_40C4h	0000_40E4h	0h	Регистр старшего слова значения системного таймера
RELOAD_L	0000_40C8h	0000_40E8h	FFFF_FFFFh	Регистр младшего слова значения перезагрузки системного таймера
RELOAD_H	0000_40CCh	0000_40ECh	FFFF_FFFFh	Регистр старшего слова значения перезагрузки системного таймера
CTRL	0000_40D0h	0000_40F0h	0h	Регистр включения системного таймера
Примечание – Адреса в диапазонах 0000_40D4h – 0000_40DFh и 0000_40F4h – 0000_40FFh являются зарезервированными.				

Таблица Б.8 – Карта области регистров контроллера прерываний

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4100h	INT0	0000_6000h	Регистры векторов прерываний с 0 по 63 (номер вектора указан в мнемоническом обозначении)
0000_4104h	INT1	0000_6010h	
0000_4108h	INT2	0000_6020h	
0000_410Ch	INT3	0000_6030h	
0000_4110h	INT4	0000_6040h	
0000_4114h	INT5	0000_6050h	
0000_4118h	INT6	0000_6060h	
0000_411Ch	INT7	0000_6070h	
0000_4120h	INT8	0000_6080h	
0000_4124h	INT9	0000_6090h	
0000_4128h	INT10	0000_60A0h	
0000_412Ch	INT11	0000_60B0h	
0000_4130h	INT12	0000_60C0h	
0000_4134h	INT13	0000_60D0h	
0000_4138h	INT14	0000_60E0h	
0000_413Ch	INT15	0000_60F0h	
0000_4140h	INT16	0000_6100h	
0000_4144h	INT17	0000_6110h	
0000_4148h	INT18	0000_6120h	
0000_414Ch	INT19	0000_6130h	
0000_4150h	INT20	0000_6140h	
0000_4154h	INT21	0000_6150h	
0000_4158h	INT22	0000_6160h	
0000_415Ch	INT23	0000_6170h	
0000_4160h	INT24	0000_6180h	
0000_4164h	INT25	0000_6190h	
0000_4168h	INT26	0000_61A0h	
0000_416Ch	INT27	0000_61B0h	

Продолжение таблицы Б.8

Адрес регистра	Мнемони-ческое обозначение	Сброс	Название регистра
0000_4170h	INT28	0000_61C0h	Регистры векторов прерываний с 0 по 63 (номер вектора указан в мнемоническом обозначении)
0000_4174h	INT29	0000_61D0h	
0000_4178h	INT30	0000_61E0h	
0000_417Ch	INT31	0000_61F0h	
0000_4180h	INT32	0000_6200h	
0000_4184h	INT33	0000_6210h	
0000_4188h	INT34	0000_6220h	
0000_418Ch	INT35	0000_6230h	
0000_4190h	INT36	0000_6240h	
0000_4194h	INT37	0000_6250h	
0000_4198h	INT38	0000_6260h	
0000_419Ch	INT39	0000_6270h	
0000_41A0h	INT40	0000_6280h	
0000_41A4h	INT41	0000_6290h	
0000_41A8h	INT42	0000_62A0h	
0000_41ACh	INT43	0000_62B0h	
0000_41B0h	INT44	0000_62C0h	
0000_41B4h	INT45	0000_62D0h	
0000_41B8h	INT46	0000_62E0h	
0000_41BCh	INT47	0000_62F0h	
0000_41C0h	INT48	0000_6300h	
0000_41C4h	INT49	0000_6310h	
0000_41C8h	INT50	0000_6320h	
0000_41CCh	INT51	0000_6330h	
0000_41D0h	INT52	0000_6340h	
0000_41D4h	INT53	0000_6350h	
0000_41D8h	INT54	0000_6360h	
0000_41DCh	INT55	0000_6370h	
0000_41E0h	INT56	0000_6380h	
0000_41E4h	INT57	0000_6390h	
0000_41E8h	INT58	0000_63A0h	
0000_41ECh	INT59	0000_63B0h	
0000_41F0h	INT60	0000_63C0h	
0000_41F4h	INT61	0000_63D0h	
0000_41F8h	INT62	0000_63E0h	
0000_41FCh	INT63	0000_63F0h	
0000_4200h	PTS0	0000_6400h	Регистры векторов прерываний блока PTS с 0 по 63 (номер вектора указан в мнемоническом обозначении)
0000_4204h	PTS1	0000_6410h	
0000_4208h	PTS2	0000_6420h	
0000_420Ch	PTS3	0000_6430h	
0000_4210h	PTS4	0000_6440h	
0000_4214h	PTS5	0000_6450h	
0000_4218h	PTS6	0000_6460h	
0000_421Ch	PTS7	0000_6470h	

Продолжение таблицы Б.8

Адрес регистра	Мнемони-ческое обозначение	Сброс	Название регистра
0000_4220h	PTS8	0000_6480h	Регистры векторов прерываний блока PTS с 0 по 63 (номер вектора указан в мнемоническом обозначении)
0000_4224h	PTS9	0000_6490h	
0000_4228h	PTS10	0000_64A0h	
0000_422Ch	PTS11	0000_64B0h	
0000_4230h	PTS12	0000_64C0h	
0000_4234h	PTS13	0000_64D0h	
0000_4238h	PTS14	0000_64E0h	
0000_423Ch	PTS15	0000_64F0h	
0000_4240h	PTS16	0000_6500h	
0000_4244h	PTS17	0000_6510h	
0000_4248h	PTS18	0000_6520h	
0000_424Ch	PTS19	0000_6530h	
0000_4250h	PTS20	0000_6540h	
0000_4254h	PTS21	0000_6550h	
0000_4258h	PTS22	0000_6560h	
0000_425Ch	PTS23	0000_6570h	
0000_4260h	PTS24	0000_6580h	
0000_4264h	PTS25	0000_6590h	
0000_4268h	PTS26	0000_65A0h	
0000_426Ch	PTS27	0000_65B0h	
0000_4270h	PTS28	0000_65C0h	
0000_4274h	PTS29	0000_65D0h	
0000_4278h	PTS30	0000_65E0h	
0000_427Ch	PTS31	0000_65F0h	
0000_4280h	PTS32	0000_6600h	
0000_4284h	PTS33	0000_6610h	
0000_4288h	PTS34	0000_6620h	
0000_428Ch	PTS35	0000_6630h	
0000_4290h	PTS36	0000_6640h	
0000_4294h	PTS37	0000_6650h	
0000_4298h	PTS38	0000_6660h	
0000_429Ch	PTS39	0000_6670h	
0000_42A0h	PTS40	0000_6680h	
0000_42A4h	PTS41	0000_6690h	
0000_42A8h	PTS42	0000_66A0h	
0000_42ACh	PTS43	0000_66B0h	
0000_42B0h	PTS44	0000_66C0h	
0000_42B4h	PTS45	0000_66D0h	
0000_42B8h	PTS46	0000_66E0h	
0000_42BCh	PTS47	0000_66F0h	
0000_42C0h	PTS48	0000_6700h	
0000_42C4h	PTS49	0000_6710h	
0000_42C8h	PTS50	0000_6720h	
0000_42CCh	PTS51	0000_6730h	

Окончание таблицы Б.8

Адрес регистра	Мнемони-ческое обозначение	Сброс	Название регистра
0000_42D0h	PTS52	0000_6740h	Регистры векторов прерываний блока PTS с 0 по 63 (номер вектора указан в мнемоническом обозначении)
0000_42D4h	PTS53	0000_6750h	
0000_42D8h	PTS54	0000_6760h	
0000_42DCh	PTS55	0000_6770h	
0000_42E0h	PTS56	0000_6780h	
0000_42E4h	PTS57	0000_6790h	
0000_42E8h	PTS58	0000_67A0h	
0000_42ECh	PTS59	0000_67B0h	
0000_42F0h	PTS60	0000_67C0h	
0000_42F4h	PTS61	0000_67D0h	
0000_42F8h	PTS62	0000_67E0h	
0000_42FCh	PTS63	0000_67F0h	

Таблица Б.9 – Карта области регистров блока FPU

Адрес регистра	Адрес в блоке	Мнемоническое обозначение	Сброс	Название регистра
0000_4300h	00h	FR0	00h	Регистры данных блока FPU с 0 по 31 (номер регистра указан в мнемоническом обозначении)
0000_4304h	01h	FR1	00h	
0000_4308h	02h	FR2	00h	
0000_430Ch	03h	FR3	00h	
0000_4310h	04h	FR4	00h	
0000_4314h	05h	FR5	00h	
0000_4318h	06h	FR6	00h	
0000_431Ch	07h	FR7	00h	
0000_4320h	08h	FR8	00h	
0000_4324h	09h	FR9	00h	
0000_4328h	0Ah	FR10	00h	
0000_432Ch	0Bh	FR11	00h	
0000_4330h	0Ch	FR12	00h	
0000_4334h	0Dh	FR13	00h	
0000_4338h	0Eh	FR14	00h	
0000_433Ch	0Fh	FR15	00h	
0000_4340h	10h	FR16	00h	
0000_4344h	11h	FR17	00h	
0000_4348h	12h	FR18	00h	
0000_434Ch	13h	FR19	00h	
0000_4350h	14h	FR20	00h	
0000_4354h	15h	FR21	00h	
0000_4358h	16h	FR22	00h	
0000_435Ch	17h	FR23	00h	
0000_4360h	18h	FR24	00h	
0000_4364h	19h	FR25	00h	
0000_4368h	1Ah	FR26	00h	
0000_436Ch	1Bh	FR27	00h	

Окончание таблицы Б.9

Адрес регистра	Адрес в блоке	Мнемоническое обозначение	Сброс	Название регистра
0000_4370h	1Ch	FR28	00h	Регистры данных блока FPU с 0 по 31 (номер регистра указан в мнемоническом обозначении)
0000_4374h	1Dh	FR29	00h	
0000_4378h	1Eh	FR30	00h	
0000_437Ch	1Fh	FR31	00h	
0000_4380h	20h	FFLAGS	00h	Регистр флага CC блока
0000_4384h	24h	FCTRL	00h	Регистр управления блока
0000_4388h	22h	FEXC	00h	Регистр исключительных состояний блока
0000_438Ch	23h	FTRAP	00h	Регистр разрешения прерываний по исключительным состояниям
0000_4390h – 0000_43FFh	–	–	–	Зарезервировано

Таблица Б.10 – Области масок портов A и B

Порты		Название
A	B	
0000_4400h – 0000_44FFh	0000_4900h – 0000_49FFh	Область масок первого байта
0000_4500h – 0000_45FFh	0000_4A00h – 0000_4AFFh	Область масок второго байта
0000_4600h – 0000_46FFh	0000_4B00h – 0000_4BFFh	Область масок третьего байта
0000_4700h – 0000_47FFh	0000_4C00h – 0000_4CFFh	Область масок четвертого байта
Примечание – После сброса микроконтроллера в каждой области масок находятся значения от 00h до FFh.		

Таблица Б.11 – Карта областей регистров портов A и B

Мнемоническое обозначение	Порты		Сброс	Название
	A	B		
DATAOUT	0000_4800h	0000_4D00h	0h	Регистр выходных данных
DATAIN	0000_4804h	0000_4D04h	XXh	Регистр принятых данных
DATALOCK	0000_4808h	0000_4D08h	0h	Регистр запрета доступа к регистру выходных данных
PULLEN	0000_480Ch	0000_4D0Ch	0h	Регистр включения режима подтяжки
PULLMODE	0000_4810h	0000_4D10h	0h	Регистр выбора типа подтяжки
DEN	0000_4814h	0000_4D14h	0h	Регистр разрешения передачи данных
ODCTL	0000_4818h	0000_4D18h	0h	Регистр включения режима открытого стока
PCTL0	0000_481Ch	0000_4D1Ch	0h	Регистр 0 выбора альтернативных функций
PCTL1	0000_4820h	0000_4D20h	0h	Регистр 1 выбора альтернативных функций
ALTFEN	0000_4824h	0000_4D24h	0h	Регистр включения альтернативной функции
SE	0000_4828h	0000_4D28h	0h	Регистр подключения фильтра

QE	0000_482Ch	0000_4D2Ch	0h	Регистр подключения схемы отсчетов
QM	0000_4830h	0000_4D30h	0h	Регистр количества отсчетов
QPAD	0000_4834h	0000_4D34h	0h	Регистр временного интервала
CONLOCK	0000_4838h	0000_4D38h	0h	Регистр сохранения конфигурации порта

Окончание таблицы Б.11

Мнемоническое обозначение	Порты		Сброс	Название
	A	B		
INTPOL	0000_483Ch	0000_4D3Ch	0h	Регистр выбора логического уровня сигнала прерывания
INTTYPE	0000_4840h	0000_4D40h	0h	Регистр выбора фронта сигнала прерывания
INTEN	0000_4844h	0000_4D44h	0h	Регистр разрешения прерывания
INTCLR	0000_4848h	0000_4D48h	0h	Регистр сброса флага прерывания
INTLINESEL0	0000_484Ch	0000_4D4Ch	0h	Регистр 0 выбора линии прерывания
INTLINESEL1	0000_4850h	0000_4D50h	0h	Регистр 1 выбора линии прерывания
INTLINESEL2	0000_4854h	0000_4D54h	0h	Регистр 2 выбора линии прерывания
INTLINESEL3	0000_4858h	0000_4D58h	0h	Регистр 3 выбора линии прерывания
INTSTAT	0000_485Ch	0000_4D5Ch	0h	Регистр флагов прерываний
Примечание – Адреса в диапазонах 0000_4860h – 0000_48FFh и 0000_4D60h – 0000_4D6Fh являются зарезервированными.				

Таблица Б.12 – Карта области регистров блока сторожевого таймера

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4D70h	CTRL	00h	Регистр управления WDT
0000_4D74h	STAT	00h	Регистр состояния WDT
0000_4D78h	RELOAD	00h	Регистр значения загрузки WDT
0000_4D7Ch	CNT	00h	Регистр счетчика WDT
0000_4D80h	CLR	–	Регистр сброса счетчика WDT
0000_4D84h – 0000_4D9Fh	–	–	Зарезервировано

Таблица Б.13 – Карта области регистров блока АЦП

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4DA0h	CON	00h	Регистр управления АЦП
0000_4DA4h	SET	00h	Регистр установок АЦП
0000_4DA8h	EN	00h	Регистр инициализации АЦП
0000_4DACh	RESULT	00h	Регистр результата АЦП
0000_4DB0h	CHANNEL	00h	Регистр номера канала АЦП
0000_4DB4h – 0000_4DFFh	–	–	Зарезервировано

Таблица Б.14 – Карта области регистров блока OCDS

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4E00h	DEBPC	00h	Регистр адреса блока отладки
0000_4E04h	DEBDAT	00h	Регистр значения на шине данных
0000_4E08h	DEBADDR	00h	Регистр значения на шине адреса
0000_4E0Ch	DEBCTRL	00h	Регистр управления блока отладки
0000_4E10h	CURRPC	00h	Регистр текущего значения счетчика команд
0000_4E14h	HAPPC	00h	Регистр состояния счетчика команд
0000_4E18h	DEBPCEQ1	00h	Регистр адреса блока отладки
0000_4E1Ch	DEBPCEQ2	00h	Регистр адреса блока отладки
Примечание – Адреса с 0000_4E20h по 0000_4E3Fh являются зарезервированными.			

Таблица Б.15 – Карта области регистров контроллера I2C

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4E40h	SDA	xxh	Сдвиговый регистр данных
0000_4E44h	ST	00h	Регистр состояния
0000_4E48h	CST	00h	Регистр управления и статуса
0000_4E4Ch	CTL0	00h	Регистр 0 управления
0000_4E50h	ADDR	00h	Регистр собственного адреса
0000_4E54h	CTL1	00h	Регистр 1 управления
0000_4E58h	TOPR	00h	Регистр загрузки предделителя
0000_4E5Ch	CTL2	00h	Регистр 2 управления
0000_4E60h – 0000_4EFFh	–	–	Зарезервировано

Таблица Б.16 – Карта области регистров блока UART0

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4F00h	DR	00h	Регистр данных UART0
0000_4F04h	ECR	00h	Регистр состояния приемника и сброса ошибки приемника UART0
0000_4F08h – 0000_4F14h	–	–	Зарезервировано
0000_4F18h	FR	0090h	Регистр флагов UART0
0000_4F1Ch	–	–	Зарезервировано
0000_4F20h	ILPR	00h	Регистр управления ИК обменом
0000_4F24h	IBRD	00h	Регистр целой части делителя UART0
0000_4F28h	FBRD	00h	Регистр дробной части делителя UART0
0000_4F2Ch	LCRH	00h	Регистр управления линией UART0
0000_4F30h	CR	0300h	Регистр управления UART0
0000_4F34h	IFLS	0012h	Регистр порога прерывания по заполнению буфера FIFO UART0
0000_4F38h	IMSC	00h	Регистр маски прерывания UART0
0000_4F3Ch	RIS	00h	Регистр состояния прерываний UART0
0000_4F40h	MIS	00h	Регистр состояния прерываний с маскированием UART0
0000_4F44h	ICR	00h	Регистр сброса прерывания UART0

0000_4F48h – 0000_4F7Fh	–	–	Зарезервировано
Примечание – Адреса регистров блоков UART1, UART2 и UART3 см. в таблице Б.23.			

Таблица Б.17 – Карта области регистров контроллера SPI0

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4F80h	CR0	0000h	Регистр управления 0 SPI0
0000_4F84h	CR1	00h	Регистр управления 1 SPI0
0000_4F88h	DR	0xh	Буфер FIFO приемника и передатчика SPI0
0000_4F8Ch	SR	03h	Регистр состояния SPI0
0000_4F90h	CPSR	00h	Регистр делителя тактовой частоты SPI0
0000_4F94h	IMSC	00h	Регистр маски прерывания SPI0
0000_4F98h	RIS	08h	Регистр состояния прерываний без учета маскирования SPI0
0000_4F9Ch	MIS	00h	Регистр состояния прерываний с учетом маскирования SPI0
0000_4FA0h	ICR	00h	Регистр сброса прерывания SPI0
0000_4FA4h – 0000_4FEFh	–	–	Зарезервировано
Примечание – Адреса регистров блока SPI1 см. в таблице Б.24.			

Таблица Б.18 – Карта области регистров блока управления тактированием

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_4FF0h	CLKEN	7FFF_FFFFh	Регистр управления тактированием
0000_4FF4h	CLKDBG_L	00h	Регистр управления тактированием в режиме отладки (младшее слово)
0000_4FF8h	CLKDBG_H	FFFF_FFFFh	Регистр управления тактированием в режиме отладки (старшее слово)
0000_4FFCh	–	–	Зарезервировано

Таблица Б.19 – Область внутреннего ОЗУ команд и данных

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0000_5000h – 0000_5FFFh	–	00h	PSRAM
0001_0000h – 0001_3FFFh	Задается пользователем	00h	ОЗУ1

Таблица Б.20 – Адреса регистров контроллеров МПИ

Мнемоника	Контроллеры		Название
	МПИ0	МПИ1	
RAM	0010_0000h – 0010_0FFFh	0010_1000h – 0010_1FFFh	ОЗУ контроллера
BSICONFIG1	0010_2000h	0010_2020h	Регистр 1 конфигурации
BSICONFIG2	0010_2004h	0010_2024h	Регистр 2 конфигурации
BSISADDR	0010_2008h	0010_2028h	Регистр адреса управляющего слова
BSISPACE	0010_200Ch	0010_202Ch	Регистр паузы
BSISTAT	0010_2010h	0010_2030h	Регистр состояния
–	0010_2014h – 0010_201Fh	0010_2034h – 0010_203Fh	Зарезервировано

Таблица Б.21 – Зарезервированная область памяти (не использовать)

Адрес регистра	Назначение
0010_2040h – 0010_2FFFh	Зарезервировано

Таблица Б.22 – Карта области регистров контроллера ARINC

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0010_3000h	XCON0	44_0900h (приемник)/ 00h (передатчик)	Регистры конфигурации блока с 0 по 7 (номер регистра указан в мнемоническом обозначении)
0010_3004h	XCON1		
0010_3008h	XCON2		
0010_300Ch	XCON3		
0010_3010h	XCON4		
0010_3014h	XCON5		
0010_3018h	XCON6		
0010_301Ch	XCON7		
0010_3020h	XSTAT0	00h	Регистры состояния блока с 0 по 7 (номер регистра указан в мнемоническом обозначении)
0010_3024h	XSTAT1	00h	
0010_3028h	XSTAT2	00h	
0010_302Ch	XSTAT3	00h	
0010_3030h	XSTAT4	00h	
0010_3034h	XSTAT5	00h	
0010_3038h	XSTAT6	00h	
0010_303Ch	XSTAT7	00h	
0010_3040h	RXLABFIFO	00h	Регистр FIFO
0010_3044h – 0010_304Fh	–	–	Зарезервировано
0010_3050h	XDATA0	00h	Регистры данных блока с 0 по 7 (номер регистра указан в мнемоническом обозначении)
0010_3054h	XDATA1	00h	
0010_3058h	XDATA2	00h	
0010_305Ch	XDATA3	00h	
0010_3060h	XDATA4	00h	
0010_3064h	XDATA5	00h	
0010_3068h	XDATA6	00h	
0010_306Ch	XDATA7	00h	
0010_3070h – 0010_30FFh	–	–	Зарезервировано

Таблица Б.23 – Адреса регистров приемопередатчиков UART

Мнемо-ника	Приемопередатчики			Сброс	Название
	Адрес 0010_xxxx				
	UART1	UART2	UART3		
DR	3100h	3180h	3200h	00h	Регистр данных
ECR	3104h	3184h	3204h	00h	Регистр состояния приемника и сброса ошибки приемника
–	3108h – 3117h	3188h – 3197h	3208h – 3217h	–	Зарезервировано
FR	3118h	3198h	3218h	90h	Регистр флагов
–	311Ch –	319Ch –	321Ch –	–	Зарезервировано

	311Fh	319Fh	321Fh		

Окончание таблицы Б.23

Мнемо-ника	Приемопередатчики Адрес 0010_xxxx			Сброс	Название
	UART1	UART2	UART3		
ILPR	3120h	31A0h	3220h		
IBRD	3124h	31A4h	3224h	00h	Регистр целой части делителя
FBRD	3128h	31A8h	3228h	00h	Регистр дробной части делителя
LCRH	312Ch	31ACh	322Ch	00h	Регистр управления линией
CR	3130h	31B0h	3230h	300h	Регистр управления
IFLS	3134h	31B4h	3234h	12h	Регистр порога прерывания по заполнению буфера FIFO
IMSC	3138h	31B8h	3238h	00h	Регистр маски прерывания
RIS	313Ch	31BCh	323Ch	00h	Регистр состояния прерываний
MIS	3140h	31C0h	3240h	00h	Регистр состояния прерываний с маскированием
ICR	3144h	31C4h	3244h	00h	Регистр сброса прерывания
–	3148h – 317Fh	31C8h – 31FFh	3248h – 327Fh	–	Зарезервировано
Примечание – Адреса регистров, блока UART0 см. в таблице Б.16.					

Таблица Б.24 – Карта области регистров контроллера SPI1

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0010_3280h	CR0	0000h	Регистр управления 0 SPI1
0010_3284h	CR1	00h	Регистр управления 1 SPI1
0010_3288h	DR	0xh	Буфер FIFO приемника и передатчика SPI1
0010_328Ch	SR	03h	Регистр состояния SPI1
0010_3290h	CPSR	00h	Регистр делителя тактовой частоты SPI1
0010_3294h	IMSC	00h	Регистр маски прерывания SPI1
0010_3298h	RIS	08h	Регистр состояния прерываний без учета маскирования SPI1
0010_329Ch	MIS	00h	Регистр состояния прерываний с учетом маскирования SPI1
0010_32A0h	ICR	00h	Регистр сброса прерывания SPI1
0010_32A4h – 0010_32FFh	–	–	Зарезервировано
Примечание – Адреса регистров блока SPI0 см. в таблице Б.17.			

Таблица Б.25 – Адреса регистров квадратурных декодеров

Мнемоника	Декодеры		Название
	Адрес 0010_xxxx		
	QEP0	QEP1	
QPOSCNT	3300h	3380h	Регистр счетчика позиции
QPOSINIT	3304h	3384h	Регистр инициализации счетчика позиции
QPOSMAX	3308h	3388h	Регистр максимального значения счетчика позиции
QPOSCMP	330Ch	338Ch	Регистр сравнения счетчика позиции
QPOSILAT	3310h	3390h	Регистр хранения позиции по индексации
QPOSSLAT	3314h	3394h	Регистр хранения позиции по стробу
QPOSLAT	3318h	3398h	Регистр хранения позиции по сторожевому таймеру
QUTMR	331Ch	339Ch	Регистр таймера временных отсчетов
QUPRD	3320h	33A0h	Регистр порога таймера временных отсчетов
QWDTMR	3324h	33A4h	Регистр сторожевого таймера
QWDPRD	3328h	33A8h	Регистр длительности сторожевого отсчета
QDECCTL	332Ch	33ACh	Регистр управления входами
QEPCTL	3330h	33B0h	Регистр управления квадратурного декодера
QCAPCTL	3334h	33B4h	Регистр блока захвата
QPOSCTL	3338h	33B8h	Регистр управления счетчиком позиции
QEINT	333Ch	33BCh	Регистр масок прерываний
QFLG	3340h	33C0h	Регистр флагов прерываний
QCLR	3344h	33C4h	Регистр сброса прерываний
QFRC	3348h	33C8h	Регистр генерации прерываний
QEPSTS	334Ch	33CCh	Регистр статуса
QCTMR	3350h	33D0h	Регистр таймера
QCPRD	3354h	33D4h	Регистр длительности измерения
QCTMRLAT	3358h	33D8h	Регистр хранения таймера
QCPRDLAT	335Ch	33DCh	Регистр хранения длительности измерения
–	3360h – 336Fh	33E0h – 33EFh	Зарезервировано
INTCLR	3370h	33F0h	Регистр сброса прерываний
–	3374h – 337Fh	33F4h – 33FFh	Зарезервировано
Примечание – Состояние всех регистров после сброса 00h.			

Таблица Б.26 – Адреса регистров блоков ШИМ

Мнемоника	Блоки ШИМ			Название
	Адрес 0010_xxxx			
	PWM0	PWM1	PWM2	
TBCTL	3400h	3500h	3600h	Регистр управления таймером
TBSTS	3404h	3504h	3604h	Регистр статуса таймера
TBPHS	3408h	3508h	3608h	Регистр фазы
TBCTR	340Ch	350Ch	360Ch	Регистр текущего значения таймера ШИМ
TBPRD	3410h	3510h	3610h	Регистр максимального значения таймера
CMPCTL	3414h	3514h	3614h	Регистр управления компаратором
CMPA	3418h	3518h	3618h	Регистр порога срабатывания A
CMPB	341Ch	351Ch	361Ch	Регистр порога срабатывания B
AQCTLA	3420h	3520h	3620h	Регистр обработчика для выхода A

Продолжение таблицы Б.26

Мнемоника	Блоки ШИМ			Название
	Адрес 0010_xxxx			
	PWM0	PWM1	PWM2	
AQCTLB	3424h	3524h	3624h	Регистр обработчика для выхода B
AQSFRC	3428h	3528h	3628h	Регистр обработчика для однократного программного управления
AQSFRC	3428h	3528h	3628h	Регистр обработчика для однократного программного управления
AQCSFRC	342Ch	352Ch	362Ch	Регистр обработчика для циклического программного управления
DBCTL	3430h	3530h	3630h	Регистр управления генератором «мертвого времени» блока ШИМ
DBRED	3434h	3534h	3634h	Регистр задержки фронта
DBFED	3438h	3538h	3638h	Регистр задержки среза
TZSEL	343Ch	353Ch	363Ch	Регистр источника сигнала аварии
TZCTL	3440h	3540h	3640h	Регистр управления детектором событий аварии
TZEINT	3444h	3544h	3644h	Регистр маски прерывания детектора событий аварии
TZFLG	3448h	3548h	3648h	Регистр флагов прерывания детектора событий аварии
TZCLR	344Ch	354Ch	364Ch	Регистр сброса флагов прерывания детектора событий аварии
TZFRC	3450h	3550h	3650h	Регистр программной эмуляции сигнала аварии
ETSEL	3454h	3554h	3654h	Регистр источника триггера событий
ETPS	3458h	3558h	3658h	Регистр предделителя триггера событий
ETFLG	345Ch	355Ch	365Ch	Регистр флагов триггера событий
ETCLR	3460h	3560h	3660h	Регистр сброса флагов триггера событий
ETFRC	3464h	3564h	3664h	Регистр программной эмуляции событий
PCCTL	3468h	3568h	3668h	Регистр управления модулятором
HRCNFG	346Ch	356Ch	366Ch	Регистр конфигурации блока ШИМ высокого разрешения
FWDTH	3470h	3570h	3670h	Регистр ширины фильтрации
–	3474h – 3487h	3574h – 3587h	3674h – 3687h	Зарезервировано
HDSEL	3488h	3588h	3688h	Регистр источника события удержания
HDCTL	348Ch	358Ch	368Ch	Регистр управления детектором событий удержания
HDEINT	3490h	3590h	3690h	Регистр маски прерывания порогового выключателя
HDFLG	3494h	3594h	3694h	Регистр флагов прерывания порогового выключателя
HDCLR	3498h	3598h	3698h	Регистр сброса флагов порогового выключателя
HDFRC	349Ch	359Ch	369Ch	Регистр программной активации порогового выключателя

HDINTCLR	34A0h	35A0h	36A0h	Регистр сброса прерывания порогового выключателя

Окончание таблицы Б.26

Мнемоника	Блоки ШИМ Адрес 0010_xxxx			Название
	PWM0	PWM1	PWM2	
TZINTCLR	34A4h	35A4h	36A4h	Регистр сброса прерывания порогового выключателя
INTCLR	34A8h	35A8h	36A8h	Регистр сброса прерывания порогового выключателя
–	34ACh – 34FFh	35ACh – 35FFh	36ACh – 36FFh	Зарезервировано
Примечание – Состояние всех регистров после сброса 00h.				

Таблица Б.27 – Зарезервированная область памяти (не использовать)

Адрес регистра	Назначение
0010_3700h – 0010_40FCh	Зарезервировано

Таблица Б.28 – Регистры контроллеров SpaceWire

Мнемоника	Адреса блоков		Сброс	Название
	SpaceWire0	SpaceWire1		
CON	0010_4000h	0010_4020h	FF00h	Регистр управления
STAT	0010_4004h	0010_4024h	3001h	Регистр состояния
DATAT	0010_4008h	0010_4028h	00h	Регистр отправляемых данных
DATAR	0010_400Ch	0010_402Ch	00h	Регистр принятых данных
TIMET	0010_4010h	0010_4030h	00h	Регистр отправляемого кода времени
TIMER	0010_4014h	0010_4034h	00h	Регистр принятого кода времени
CAPSTAT	0010_4018h	0010_4038h	00h	Регистр состояния буферов
INTMASK	0010_404Ch	0010_403Ch	00h	Регистр маски прерываний

Таблица Б.29 – Регистры блока HSIO

Адрес регистра	Мнемоническое обозначение	Сброс	Название регистра
0010_4100h – 0010_44FFh	–	00h	Область FIFO блока HSI. Регистры доступны только для чтения.
0010_4500h – 0010_485Fh	–	00h	Зарезервировано
0010_4860h – 0010_48FCh	–	00h	Область CAM блока HSO. Регистры доступны для записи и чтения
0010_4900h	HSICON	00h	Регистр управления модулем HSI
0010_4904h	HSISTAT	00h	Регистр состояния модуля HSI
0010_4908h	HSICHNLCFG0	00h	Регистр событий модуля HSI
0010_490Ch	HSICHNLCFG1	00h	Регистр событий модуля HSI
0010_4910h	HSITIMER	00h	Регистр счетчика таймера модуля HSI
0010_4914h	HSIFIFO	00h	Регистр чтения FIFO модуля HSI
0010_4918h	HSOCON	00h	Регистр управления модулем HSO
0010_491Ch	HSOTIMEROV	00h	Регистр перезагрузки счетчика таймера модуля HSO
0010_4920h	HSOTIMER	00h	Регистр счетчика таймера модуля HSO
0010_4924h – 0010_4FFFh	–	–	Зарезервировано

Приложение В (обязательное) **Коды состояний функционирования блока I2C**

В таблицах В.1 – В.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 В.1 – Исключительные состояния

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.2 – Режим FS мастера передатчика (дополнительно см. таблицу В.4)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.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 В.3 – Режим FS мастера приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.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 В.4 – Режим FS мастера передатчика (дополнительно см. таблицу В.2)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.5 – Режим FS ведомого приемника (дополнительно см. таблицу В.7)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.6 – Режим FS ведомого передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.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 В.7 – Режим FS ведомого приемника (дополнительно см. таблицу В.5)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.8 – Режим HS мастера передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.9 – Режим HS мастера приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.10 – Режим HS ведомого приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 В.11 – Режим HS ведомого передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				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 Г (обязательное) **Система команд микроконтроллеров**

Система команд микроконтроллеров включает в себя команды, поддерживаемые трансляторами с языка ассемблера для 16- и 32-разрядных микроконтроллеров семейства MCS-196, и команды программных прерываний TRAP и TRAP1, используемых в инструментальных средствах разработки. Команды сложения, вычитания, умножения, деления, логических операций, сравнения, загрузки, работы с битами и стеком, а также группа команд блока FPU поддерживают до четырех типов адресации.

В таблице Г.1 представлена информация о командах микроконтроллера и вариантах их использования. Для каждой команды указано: мнемоническое название, название выполняемой операции, описание и порядок следования операндов, состояние флагов регистра PSW, возникающее в результате выполнения команды, варианты использования, с указанием для каждого варианта его опкода, длины и времени выполнения.

В таблице Г.1 приняты обозначения:

DEST	–	операнд-приемник (только прямая адресация);
SRC	–	операнд-источник (поддерживаются различные режимы адресации);
SRC1	–	первый операнд-источник (только прямая адресация);
SRC2	–	второй операнд-источник (поддерживаются различные режимы адресации);
CADD	–	метка перехода;
COUNT	–	операнд-указатель количества сдвигов;
lreg	–	регистр двойного слова (32 бита) с адресом, кратным четырем;
wreg	–	регистр слова (16 бит) с адресом, кратным двум;
breg	–	регистр байта (8 бит);
#data	–	константа: - от 0h до FFh для 16-разрядного режима; - от 0h до FFFFh для 32-разрядного режима;
reg	–	регистр, используемый для косвенной адресации (8-битный адрес);
ereg	–	регистр, используемый для косвенной адресации (16 битный адрес);
boffset	–	8-разрядная константа в режиме короткоиндекной адресации, используемая для задания смещения со знаком в диапазоне от –128 до +127;
woffset	–	16-разрядная константа в режиме длинноиндексной адресации, используемая для задания смещения без знака в диапазоне от 0 до +65535;
eboffset	–	10-разрядная константа в режиме короткоиндекной адресации с использованием значения счетчика PC, используемая для задания смещения со знаком в диапазоне от –512 до +511;
ewoffset	–	18-разрядная константа в режиме длинноиндексной адресации, используемая для задания смещения со знаком в диапазоне от –131072 до +131071;
cadd	–	адрес в программном коде, указанный непосредственно в диапазоне от 0h до FFh или как название метки перехода;
ecadd	–	адрес в программном коде, указанный непосредственно в диапазоне от 0h до FFFFh или как название метки перехода;
#count	–	4-/5-разрядная константа для задания количества сдвигов (от 0 до 31) при сдвиговых операциях.

Для представления влияния, оказываемого результатом выполнения команды на флаги регистра PSW, приняты обозначения:

√ – команда модифицирует флаг (устанавливает или сбрасывает, если требуется);

– – команда не модифицирует флаг;

↓ – команда сбрасывает флаг, если требуется, но не устанавливает;

↑ – команда устанавливает флаг, если требуется, но не сбрасывает;

1 – команда устанавливает флаг;

0 – команда сбрасывает флаг;

? – команда оставляет флаг в неопределенном состоянии.

Код	x0	x1	x2	x3	x4	x5	x6	x7
0x	SKIP di	CLR di	NOT di	NEG di	XCH di	DEC di	EXT di	INC di
1x	SWC	CLRB di	NOTB di	NEGB di	XCHB di	DECB di	EXTB di	INCB di
2x	SJMP ix							
3x	JBC ix							
	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	AND 3op				ADD 3op			
	di	im	in	ix	di	im	in	ix
5x	ANDB 3op				ADDB 3op			
	di	im	in	ix	di	im	in	ix
6x	AND 2op				ADD 2op			
	di	im	in	ix	di	im	in	ix
7x	ANDB 2op				ADDB 2op			
	di	im	in	ix	di	im	in	ix
8x	OR				XOR			
	di	im	in	ix	di	im	in	ix
9x	ORB				XORB			
	di	im	in	ix	di	im	in	ix
Ax	LD				ADDC			
	di	im	in	ix	di	im	in	ix
Bx	LDB				ADDCB			
	di	im	in	ix	di	im	in	ix
Cx	ST di	BMOV in	ST		STB di	CMPL	STB	
			in	ix		di	in	ix
Dx	JNST ix	JNH ix	JGT ix	JNC ix	JNVT ix	JNV ix	JGE ix	JNE ix
Ex	DJNZ ix	DJNZW ix	TIJMP	BR in	VLCALL ix	F-INSTR	VLJMP ix	LJMP ix
Fx	RET in	TRAP1 di	PUSHF di	POPF di	PUSHA di	POPA di	MOD im	TRAP di

Рисунок Г.1, лист 1 – Система команд микроконтроллера для 16-разрядного режима

Код	x8	x9	xA	xB	xC	xD	xE	xF
0x	SHR di	SHL di	SHRA di	XCH ix	SHRL di	SHLL di	SHRAL di	NORML di
1x	SHRB di	SHLB di	SHRAB di	XCHB ix	-	-	-	-
2x	SCALL ix							
3x	JBS ix							
	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	SUB 3op				MULU 3op *			
	di	im	in	ix	di	im	in	ix
5x	SUBB 3op				MULUB 3op *			
	di	im	in	ix	di	im	in	ix
6x	SUB 2op				MULU 2op *			
	di	im	in	ix	di	im	in	ix
7x	SUBB 2op				MULUB 2op *			
	di	im	in	ix	di	im	in	ix
8x	CMP				DIVU *			
	di	im	in	ix	di	im	in	ix
9x	CMPB				DIVUB *			
	di	im	in	ix	di	im	in	ix
Ax	SUBC				LDBZE			
	di	im	in	ix	di	im	in	ix
Bx	SUBCB				LDBSE			
	di	im	in	ix	di	im	in	ix
Cx	PUSH				POP	BMOVI	POP	
	di	im	in	ix	di		in	ix
Dx	JST ix	JH ix	JLE ix	JC ix	JVT ix	JV ix	JLT ix	JE ix
Ex	-	-	-	-	DPTS di	EPTS di	Прерыв.	LCALL ix
Fx	CLRC di	SETC di	DI di	EI di	CLRVT di	NOP di	*	RST di

*FE

Код	xC	xD	xE	xF
4x	MUL 3op			
	di	im	in	ix
5x	MULB 3op			
	di	im	in	ix
6x	MUL 2op			
	di	im	in	ix
7x	MULB 2op			
	di	im	in	ix
8x	DIV			
	di	im	in	ix
9x	DIVB			
	di	im	in	ix

Рисунок Г.1, лист 2

Код	x0	x1	x2	x3	x4	x5	x6	x7
0x	SKIP di	ECLR di	ENOT di	ENEG di	ELDS di	EDEC di	EXT di	EINC di
1x	SWC	CLRB di	NOTB di	NEGB di	XCHB di	DECB di	EXTB di	INCB
2x	SJMP ix							
3x	JBC ix							
	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	ELDB				ADD 3op			
	di	im	in	ix	di	im	in	ix
5x	ELDW				ADDB 3op			
	di	im	in	ix	di	im	in	ix
6x	EAND 2op				EADD 2op			
	di	im	in	ix	di	im	in	ix
7x	ANDB 2op				ADDB 2op			
	di	im	in	ix	di	im	in	ix
8x	EOR				EXOR			
	di	im	in	ix	di	im	in	ix
9x	ORB				XORB			
	di	im	in	ix	di	im	in	ix
Ax	ELD				EADDC			
	di	im	in	ix	di	im	in	ix
Bx	LDB				ADDCB			
	di	im	in	ix	di	im	in	ix
Cx	ST di	EBMOV in	EST		STB di	-	STB	
			in	ix			in	ix
Dx	JNST ix	JNH ix	JGT ix	JNC ix	JNVT ix	JNV ix	JGE ix	JNE ix
Ex	DJNZ ix	EDJNZ ix	ETIJMP	EBR in	VLCALL ix	F-INSTR	VLJMP ix	LJMP ix
Fx	RET in	TRAP1 di	PUSHF di	POPF di	PUSHA di	POPA di	MOD im	TRAP di

Рисунок Г.2, лист 1 – Система команд микроконтроллера для 32-разрядного режима

Код	x8	x9	xA	xB	xC	xD	xE	xF
0x	SHR di	SHL di	SHRA di	XCH ix	ESHRL di	ESHLL di	ESHRAL di	NORML di
1x	SHRB di	SHLB di	SHRAB di	XCHB ix	EST short ix pc			
2x	SCALL ix							
3x	JBS ix							
	bit 0	bit 1	bit2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	SUB 3op				MULU 3op *			
	di	im	in	ix	di	im	in	ix
5x	SUBB 3op				MULUB 3op *			
	di	im	in	ix	di	im	in	ix
6x	ESUB 2op				EMULU 2op *			
	di	im	in	ix	di	im	in	ix
7x	SUBB 2op				MULUB 2op *			
	di	im	in	ix	di	im	in	ix
8x	ECMP				EDIVU *			
	di	im	in	ix	di	im	in	ix
9x	CMPB				DIVUB *			
	di	im	in	ix	di	im	in	ix
Ax	ESUBC				ELDZE			
	di	im	in	ix	di	im	in	ix
Bx	SUBCB				ELDSE			
	di	im	in	ix	di	im	in	ix
Cx	EPUSH				EPOP	EBMOVI in	EPOP	
	di	im	in	ix	di		in	ix
Dx	JST ix	JH ix	JLE ix	JC ix	JVT ix	JV ix	JLT ix	JE ix
Ex	EST long ix pc				DPTS di	EPTS di	Прерыв.	LCALL ix
Fx	CLRC di	SETC di	DI di	EI di	CLRVT di	NOP di	*	RST di

*FE

Код	xC	xD	xE	xF
4x	MUL 3op			
	di	im	in	ix
5x	MULB 3op			
	di	im	in	ix
6x	EMUL 2op	-	EMUL 2op	-
	di		in	
7x	MULB 2op			
	di	im	in	ix
8x	EDIV			-
	di	im	in	
9x	DIVB			
	di	im	in	ix

Рисунок Г.2, лист 2

В таблице Г.1 время выполнения команды указано в количестве машинных тактов (мт). Выполнение некоторых команд может быть ускорено до одного машинного такта при использовании режима прямой адресации, а также при условии, что все операнды попадают в область регистров быстрых RISK команд R0-R15 (на рисунках Г.1 и Г.2 отмечены серым цветом). Для этих команд время выполнения указывается двумя или тремя цифрами (в зависимости от количества операндов) через слеш, как показано на рисунке Г.3.

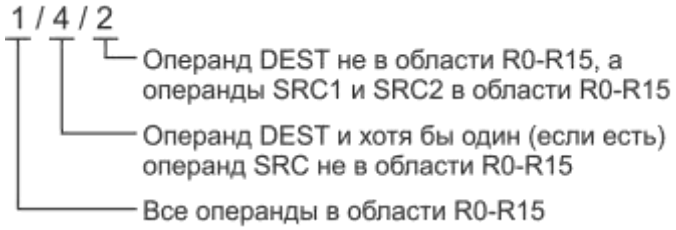


Рисунок Г.3 – Назначение позиций в строке при указании времени выполнения команды

Для команд арифметического и логического сдвига (SHR, SHRA, ESHLL и др.) назначение позиций в строке при указании времени выполнения показано на рисунке Г.4.

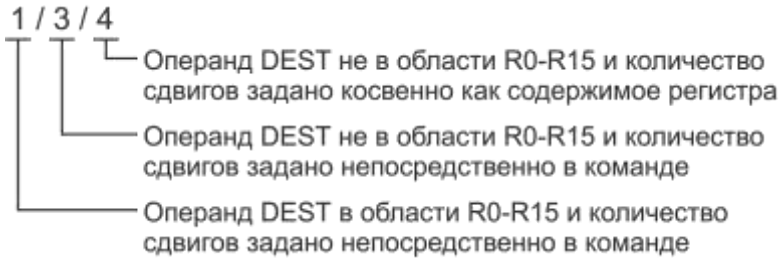


Рисунок Г.4 – Назначение позиций в строке при указании времени выполнения команды сдвига

Таблица Г.1 – Система команд микроконтроллера

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ADD **EADD**	**Сложение слов** (DEST, SRC)			
	16-разрядный режим			
	ADD wreg, wreg	64h	3	1 / 4
	ADD wreg, #data	65h	4	4
	ADD wreg, [reg]	66h	3	5
	ADD wreg, [reg]+			6
	ADD wreg, boffset[reg]	67h	4	6
	ADD wreg, woffset[reg]		5	6
	Сложение слова SRC и слова DEST и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC)			
	32-разрядный режим			
	EADD lreg, lreg	64h	5	1 / 4
	EADD lreg, #data	65h	7	4
	EADD lreg, [ereg]	66h	5	5
	EADD lreg, [ereg]+			6
	EADD lreg, ewoffset[ereg]	67h	7	6
	Сложение двойного слова SRC и двойного слова DEST и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC)			
ADD	**Сложение слов** (DEST, SRC1, SRC2)			
	ADD wreg, wreg, wreg	44h	4	1 / 4
	ADD wreg, wreg, #data,	45h	5	4
	ADD wreg, wreg, [reg]	46h	4	5
	ADD wreg, wreg, [reg]+			6
	ADD wreg, wreg, boffset[reg]	47h	5	6
	ADD wreg, wreg, woffset[reg]		6	6
	Сложение слова SRC1 и слова SRC2 и сохранение суммы в DEST (DEST) ← (SRC1) + (SRC2)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ADDB	**Сложение байт** (DEST, SRC)			
	ADDB breg, breg	74h	3	1 / 4
	ADDB breg, #data,	75h		4
	ADDB breg, [reg]	76h		5
	ADDB breg, [reg]+			6
	ADDB breg, boffset[reg]	77h	4	6
	ADDB breg, woffset[reg]		5	6
	Сложение байта SRC и байта DEST и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC)			
ADDB	**Сложение байт** (DEST, SRC1, SRC2)			
	ADDB breg, breg, breg	54h	4	1 / 4 / 2
	ADDB breg, breg, #data,	55h		4
	ADDB breg, breg, [reg]	56h		5
	ADDB breg, breg, [reg]+			6
	ADDB breg, breg, boffset[reg]	57h	5	6
	ADDB breg, breg, woffset[reg]		6	6
	Сложение байта SRC1 и байта SRC2 и сохранение суммы в DEST (DEST) ← (SRC1) + (SRC2)			
ADDC	**Сложение слов с переносом** (DEST, SRC)			
	16-разрядный режим			
	ADDC wreg, wreg	A4h	3	1 / 4
	ADDC wreg, #data	A5h	4	4
	ADDC wreg, [reg]	A6h	3	5
	ADDC wreg, [reg]+			6
	ADDC wreg, boffset[reg]	A7h	4	6
	ADDC wreg, woffset[reg]		5	6
	Сложение слова SRC, слова DEST и флага переноса C и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC) + (C)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EADDC	**Сложение двойных слов с переносом** (DEST, SRC)			
	32-разрядный режим			
	EADDC lreg, lreg	A4h	5	1 / 4
	EADDC lreg, #data	A5h	7	4
	EADDC lreg, [ereg]	A6h	5	5
	EADDC lreg, [ereg]+			6
	EADDC lreg, ewoffset[ereg]	A7h	7	6
	Сложение двойного слова SRC, двойного слова DEST и флага переноса C и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC) + (C)			
ADDCB	**Сложение байт с переносом** (DEST, SRC)			
	ADDCB breg, breg	B4h	3	1 / 4
	ADDCB breg, #data,	B5h		4
	ADDCB breg, [reg]	B6h		5
	ADDCB breg, [reg]+			6
	ADDCB breg, boffset[reg]	B7h	4	6
	ADDCB breg, woffset[reg]		5	6
	Сложение байта SRC, байта DEST и флага переноса C и сохранение суммы в приемнике (DEST) ← (DEST) + (SRC) + (C)			
AND	**Логическое «И» слов** (DEST, SRC)			
	16-разрядный режим			
	AND wreg, wreg	60h	3	1 / 4
	AND wreg, #data	61h	4	4
	AND wreg, [reg]	62h	3	5
	AND wreg, [reg]+			6
	AND wreg, boffset[reg]	63h	4	6
	AND wreg, woffset[reg]		5	6
	Логическое умножение слова SRC и слова DEST и сохранение результата в приемнике (DEST) ← (DEST) AND (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EAND	**Логическое «И» двойных слов** (DEST, SRC)			
	32-разрядный режим			
	EAND lreg, lreg	60h	5	1 / 4
	EAND lreg, #data	61h	7	4
	EAND lreg, [ereg]	62h	5	5
	EAND lreg, [ereg]+			6
	EAND lreg, ewoffset[ereg]	63h	7	6
	Логическое умножение двойных слов SRC и DEST и сохранение результата в приемнике (DEST) ← (DEST) AND (SRC)			
AND	**Логическое «И» слов** (DEST, SRC1, SRC2)			

Z	N	C	V	VT	ST
√	√	0	0	–	–

16-разрядный режим			
AND wreg, wreg, wreg	40h	4	1 / 4
AND wreg, wreg, #data,	41h	5	4
AND wreg, wreg, [reg]	42h	4	5
AND wreg, wreg, [reg]+			6
AND wreg, wreg, boffset[reg]	43h	5	6
AND wreg, wreg, woffset[reg]		6	6
Логическое умножение слова SRC1 и слова SRC2 и сохранение результата в приемнике (DEST) ← (SRC1) AND (SRC2)			
32-разрядный режим			
См. ELDB			
ANDB	**Логическое «И» байт** (DEST, SRC)		

Z	N	C	V	VT	ST
√	√	0	0	–	–

ANDB breg, breg	70h	3	1 / 4
ANDB breg, #data,	71h		4
ANDB breg, [reg]	72h		5
ANDB breg, [reg]+			6
ANDB breg, boffset[reg]	73h	4	6
ANDB breg, woffset[reg]		5	6
Логическое умножение байта SRC и байта DEST и сохранение результата в приемнике (DEST) ← (DEST) AND (SRC)			

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="11">ANDB</td><td colspan="4">Логическое «И» байт (DEST, SRC1, SRC2)
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>√</td><td>0</td><td>0</td><td>–</td><td>–</td></tr></table></td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>ANDB breg, breg, breg</td><td>50h</td><td rowspan="4">4</td><td>1 / 4 / 2</td></tr>
<tr><td>ANDB breg, breg, #data,</td><td>51h</td><td>4</td></tr>
<tr><td>ANDB breg, breg, [reg]</td><td rowspan="2">52h</td><td>5</td></tr>
<tr><td>ANDB breg, breg, [reg]+</td><td>6</td></tr>
<tr><td>ANDB breg, breg, boffset[reg]</td><td rowspan="2">53h</td><td>5</td><td>6</td></tr>
<tr><td>ANDB breg, breg, woffset[reg]</td><td>6</td><td>6</td></tr>
<tr><td colspan="4">Логическое умножение байта SRC1 и байта SRC2 и сохранение результата в приемнике

(DEST) ← (SRC1) AND (SRC2)</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td colspan="4">См. ELDW</td></tr>
<tr><td rowspan="4">BMOV</td><td colspan="4">Непрерываемое перемещение блоков данных (DEST, SRC)</td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>BMOV lreg, wreg</td><td>C1h</td><td>3</td><td>–</td></tr>
<tr><td colspan="4">Перемещение блоков слов данных из одной зоны памяти в другую.
Регистр lreg (PTRS) содержит указатели на источник SRCPTR (младшее слово) и приемник DSTPTR (старшее слово), его содержимое изменяется в процессе выполнения команды. Регистр wreg (CNTREG) задает количество перемещаемых слов, его значение декрементируется после пересылки каждого слова. Блоки данных могут находиться в любом месте памяти, но не должны перекрываться.

COUNT ← (CNTREG)
LOOP: SRCPTR ← (PTRS)
DSTPTR ← (PTRS+2)
(DSTPTR) ← (SRCPTR)
(PTRS) ← SRCPTR+2
(PTSR+2) ← DSTPTR+2
COUNT ← COUNT – 1
Если COUNT ≠ 0, тогда переход на LOOP

Примечание – Количество слов, которое может быть перемещено, должно быть задано в диапазоне 0h – FFh.</td></tr>
</table>

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EBMOV	**Непрерываемое перемещение блоков данных** (DEST, SRC)			
	32-разрядный режим			
	EBMOV lreg, lreg, wreg	C1h	7	
	Перемещение блоков двойных слов данных из одной зоны памяти в другую. Первый операнд lreg содержит адрес регистра с указателем на приемник DSTPTR, второй операнд lreg – адрес регистра с указателем на источник SRCPTR, третий операнд – wreg (CNTREG) задает количество перемещаемых двойных слов, его значение декрементируется после пересылки каждого двойного слова. Блоки данных могут находиться в любом месте памяти, но не должны перекрываться. COUNT ← (CNTREG) LOOP: SRCPTR ← (PTRS) DSTPTR ← (PTRS+4) (DSTPTR) ← (SRCPTR) (PTRS) ← SRCPTR+4 (PTSR+4) ← DSTPTR+4 COUNT ← COUNT – 1 Если COUNT ≠ 0, тогда переход на LOOP Примечание – Количество двойных слов, которое может быть перемещено, должно быть задано в диапазоне 0h – FFFFh.			
	Примечание – Важно помнить, что никакое прерывание не будет обслужено до окончания выполнения команды BMOV, поэтому при пересылке больших массивов данных лучше использовать команду BMOVI.			
BMOVI	**Прерываемое перемещение блоков** (DEST, SRC)			
	16-разрядный режим			
	BMOVI lreg, wreg	CDh	3	–
	Перемещение блоков слов данных из одной зоны памяти в другую. Команда идентична команде BMOV, за исключением того, что она является прерываемой. Для возобновления пересылки данных следует повторно выполнить команду BMOVI, не изменяя содержимое регистров. Примечание – Количество слов, которое может быть перемещено, должно быть задано в диапазоне 0h – FFh.			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EBMOVI	**Прерываемое перемещение блоков** (DEST, SRC)			
	32-разрядный режим			
	EBMOVI lreg, lreg, wreg	CDh	7	–
	Перемещение блоков двойных слов данных из одной зоны памяти в другую. Команда идентична команде EBMOV, за исключением того, что она является прерываемой. Для возобновления пересылки данных следует повторно выполнить команду EBMOVI, не изменяя содержимое регистров. Примечание – Количество двойных слов, которое может быть перемещено, должно быть задано в диапазоне 0h – FFFFh.			
BR **EBR**	**Косвенный переход** (DEST)			
	16-разрядный режим			
	BR [reg]	E3h	2	2
	32-разрядный режим			
	EBR [ereg]	E3h	3	2
	Загрузка в программный счетчик PC адреса, косвенно задаваемого операндом и продолжение выполнения программы с этого адреса PC ← (DEST)			
CLR **ECLR**	**Очистка слова** (DEST)	Z N C V VT ST 1 0 0 0 – –		
	16-разрядный режим			
	CLR wreg	01h	2	2
	32-разрядный режим			
	ECLR lreg	01h	3	2
	Запись нуля в приемник DEST (DEST) ← 0h			
CLRB	**Очистка байта** (DEST)	Z N C V VT ST 1 0 0 0 – –		
	CLRB breg	11h	2	2
	Запись нуля в младший байт приемника DEST (DEST (младший байт)) ← 0h			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
CLRC	**Очистка флага переноса**			
	CLRC	F8h	1	1
	Запись нуля в бит C регистра PSW C ← 0b			
CLRVT	**Очистка дополнительного флага переполнения**			
	CLRVT (прямая адресация)	FCh	1	1
	Запись нуля в бит VT регистра PSW VT ← 0b			
CMP **ECMP**	**Сравнение слов** (DEST, SRC)			
	16-разрядный режим			
	CMP wreg, wreg	88h	3	1 / 4
	CMP wreg, #data	89h	4	4
	CMP wreg, [reg]	8Ah	3	5
	CMP wreg, [reg]+			5
	CMP wreg, boffset[reg]	8Bh	4	6
	CMP wreg, woffset[reg]		5	6
	Вычитание слова SRC из слова DEST. Флаги в регистре PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается (DEST) – (SRC)			
	32-разрядный режим			
	ECMP lreg, lreg	88h	5	1 / 4
	ECMP lreg, #data	89h	7	4
	ECMP lreg, [ereg]	8Ah	5	5
	ECMP wreg, [ereg]+			6
	ECMP lreg, ewoffset[ereg]	8Bh	7	6
	Вычитание двойного слова SRC из двойного слова DEST. Флаги в регистре PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается (DEST) – (SRC)			

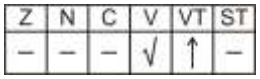
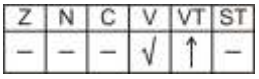
Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
CMPB	**Сравнение байт** (DEST, SRC)			
	CMPB breg, breg	98h	3	1 / 4
	CMPB breg, #data,	99h		4
	CMPB breg, [reg]	9Ah		5
	CMPB breg, [reg]+			5
	CMPB breg, boffset[reg]	9Bh	4	6
	CMPB breg, woffset[reg]		5	6
	Вычитание байта SRC из байта DEST. Флаги в регистре PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается (DEST) – (SRC)			
CMPL	**Сравнение длинных слов** (DEST, SRC)			
	16-разрядный режим			
	CMPL lreg, lreg	C5h	3	1 / 4
	Вычитание двойного слова SRC из двойного слова DEST. Операнды определяются с использованием режима прямой адресации. Флаги в регистре PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается (DEST) – (SRC)			
	32-разрядный режим			
	См. ECMP			
DEC **EDEC**	**Декремент слова** (DEST)			
	16-разрядный режим			
	DEC wreg	05h	2	1 / 3
	32-разрядный режим			
	EDEC lreg	05h	3	1 / 3
	Уменьшение величины операнда на единицу (DEST) ← (DEST) – 1			

Продолжение таблицы Г.1

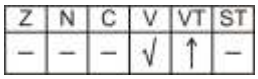
Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
DECB	**Декремент байта** (DEST)			
	DECB breg	15h	2	1 / 3
	Уменьшение величины операнда на единицу (DEST) ← (DEST) – 1			
DI	**Запрещение обслуживания всех прерываний**			
	DI	FAh	1	1
	Запрещается обслуживание всех прерываний. Запросы на прерывания не могут идти сразу после этой команды В регистре PSW бит I ← 0b			
DIV **EDIV**	**Знаковое деление слов** (DEST, SRC)			
	16-разрядный режим			
	DIV lreg, wreg	FE8Ch	4	8
	DIV lreg, #data	FE8Dh	5	8
	DIV lreg, [reg]	FE8Eh	4	9
	DIV lreg, [reg]+			10
	DIV lreg, boffset[reg]	FE8Fh	5	10
	DIV lreg, woffset[reg]		6	10
	32-разрядный режим			
	EDIV lreg, wreg	FE8Ch	6	
	EDIV lreg, #data	FE8Dh	6	
	EDIV lreg, [ereg]	FE8Eh	6	
	EDIV lreg, [ereg]+			
	Деление целого двойного слова DEST на целое слово SRC, с использованием знаковой арифметики и сохранением результата в приемнике DEST. Слово частного сохраняется в младшем слове приемника, а остаток – в старшем слове. Если результат деления превышает размер слова, то в приемнике сохраняются только младшие 15 бит результата (самый старший бит является знаковым), а остальные отбрасываются и устанавливаются флаги V и VT в регистре PSW. (DEST (младшее слово)) ← Младшее слово результата (DEST)/(SRC) (DEST (старшее слово)) ← Остаток от (DEST)/(SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
DIVB	**Знаковое деление байт** (DEST, SRC) 			
	DIVB wreg, breg	FE9Ch	4	6
	DIVB wreg, #data	FE9Dh		6
	DIVB wreg, [reg]	FE9Eh		7
	DIVB wreg, [reg]+			8
	DIVB wreg, boffset[reg]	FE9Fh	5	8
	DIVB wreg, woffset[reg]		6	8
	Деление целого слова DEST на целый байт SRC, с использованием знаковой арифметики. Частное сохраняется в младшем байте (с меньшим адресом) приемника, а остаток – в старшем байте одновременно (DEST (младший байт)) ← (DEST)/(SRC) (DEST (старший байт))← Остаток от (DEST)/(SRC)			
DIVU **EDIVU**	**Беззнаковое деление слов** (DEST, SRC) 			
	16-разрядный режим			
	DIVU lreg, wreg	8Ch	3	8
	DIVU lreg, #data	8Dh	4	8
	DIVU lreg, [reg]	8Eh	3	9
	DIVU lreg, [reg]+			10
	DIVU lreg, boffset[reg]	8Fh	4	10
	DIVU lreg, woffset[reg]		5	10
	32-разрядный режим			
	EDIVU lreg, wreg	8Ch	5	
	EDIVU lreg, #data	8Dh	5	
	EDIVU lreg, [ereg]	8Eh	5	
	EDIVU lreg, [ereg]+			
	EDIVU lreg, ewoffset[ereg]	8Fh	7	
	Деление целого двойного слова DEST на целое слово SRC, с использованием беззнаковой арифметики и сохранением результата в приемнике DEST. Слово частного сохраняется в младшем слове приемника, а остаток – в старшем слове. Если результат превышает размер слова, то в приемнике сохраняются только младшие 16 бит результата, а остальные отбрасываются и устанавливаются флаги V и VT в регистре PSW. (DEST (младшее слово)) ← Младшее слово результата (DEST)/(SRC) (DEST (старшее слово)) ← Остаток от (DEST)/(SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
DIVUB	**Беззнаковое деление байт** (DEST, SRC)			
	DIVUB wreg, breg	9Ch	3	6
	DIVUB wreg, #data	9Dh		6
	DIVUB wreg, [reg]	9Eh		7
	DIVUB wreg, [reg]+			8
	DIVUB wreg, boffset[reg]	9Fh	4	8
	DIVUB wreg, woffset[reg]		5	8
	Деление целого слова DEST на целый байт SRC, с использованием беззнаковой арифметики. Частное сохраняется в младшем байте (с меньшим адресом) приемника, а остаток – в старшем байте одновременно (DEST (младший байт)) ← (DEST)/(SRC) (DEST (старший байт))← Остаток от (DEST)/(SRC)			
DJNZ	**Декремент байта и переход при отсутствии нуля** (CADD, ADDR)			
	DJNZ breg, cadd	E0h	3	4
	Уменьшение значения байта на единицу и в случае неравенства результата нулю – переход на указанный адрес (метку). Если результат равен нулю, то управление передается следующей по порядку команде. Для вычисления перехода к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды DJNZ и адресом метки перехода. Смещение может быть в диапазоне от –128 до +127 (COUNT) ← (COUNT) – 1 Если (COUNT) ≠ 0, тогда PC ← PC + disp			
DJNZW	**Декремент слова и переход при отсутствии нуля** (CADD, ADDR)			
	16-разрядный режим			
	DJNZW wreg, cadd	E1h	3	4
	Уменьшение значения слова на единицу и, в случае неравенства результата нулю, переход на указанный адрес (метку). Если результат равен нулю, то управление передается следующей по порядку команде. Для вычисления перехода к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода. Смещение может быть в диапазоне от –128 до +127 (COUNT) ← (COUNT) – 1 Если (COUNT) ≠ 0, тогда PC ← PC + disp			



Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EDJNZ	**Декремент двойного слова и переход при отсутствии нуля** (CADD, ADDR)			
	32-разрядный режим			
	EDJNZ lreg, cadd	E1h	4	4
	Уменьшение значения слова на единицу и, в случае неравенства результата нулю, переход на указанный адрес (метку). Если результат равен нулю, то управление передается следующей по порядку команде. Для вычисления перехода к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода. Смещение может быть в диапазоне от –128 до +127 (COUNT) ← (COUNT) – 1 Если (COUNT) ≠ 0, тогда PC ← PC + disp			
DPTS	**Запрет PTS**			
	DPTS	ECh	1	1
	Блокирование сервера периферийных транзакций PTS В регистре PSW бит PSE ← 0b			
EI	**Разрешение обслуживания всех прерываний**			
	EI	FBh	1	1
	Разрешается обслуживание всех прерываний. Запросы на прерывания не могут идти сразу после этой команды В регистре PSW бит I ← 1b			
EPTS	**Разрешение PTS**			
	EPTS	EDh	1	1
	Деблокирование сервера периферийных транзакций PTS В регистре PSW бит PSE ← 1b			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EXT	**Знаковое расширение слова до двойного слова** (DEST)			
	EXT lreg (прямая адресация)	06h	2	4
	Знаковое расширение младшего слова операнда до двойного слова Если (DEST (младшее слово)) < 8000h, тогда (DEST (старшее слово)) ← 0000h, иначе (DEST (старшее слово)) ← FFFFh			
EXTB	**Знаковое расширение байта до слова** (DEST)			
	EXTB wreg (прямая адресация)	16h	2	3
	Знаковое расширение младшего байта операнда до слова Если (DEST (младший байт)) < 80h, тогда (DEST (старший байт)) ← 00h, иначе (DEST (старший байт)) ← FFh			
IDLPD	Не используется. См. описание команды **MOD**			
INC **EINC**	**Инкремент слова** (DEST)			
	16-разрядный режим			
	INC wreg	07h	2	1 / 3
	32-разрядный режим			
	EINC lreg	07h	3	1 / 3
	Увеличение значения операнда на единицу (DEST) ← (DEST) + 1			
INCB	**Инкремент байта** (DEST)			
	INC breg	17h	2 байта	1 / 3
	Увеличение значения байта операнда на единицу (DEST) ← (DEST) + 1			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
JBC	**Переход, если бит очищен**			
	JBC breg, 0, cadd	30h	3	2
	JBC breg, 1, cadd	31h		
	JBC breg, 2, cadd	32h		
	JBC breg, 3, cadd	33h		
	JBC breg, 4, cadd	34h		
	JBC breg, 5, cadd	35h		
	JBC breg, 6, cadd	36h		
	JBC breg, 7, cadd	37h		
	Если очищен бит в байте breg, то осуществляется переход на указанный адрес (метку), т. е. к значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода. Если бит установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если бит = 0b, тогда PC ← PC + disp			
JBS	**Переход, если бит установлен**			
	JBS breg, 0, cadd	38h	3	2
	JBS breg, 1, cadd	39h		
	JBS breg, 2, cadd	3Ah		
	JBS breg, 3, cadd	3Bh		
	JBS breg, 4, cadd	3Ch		
	JBS breg, 5, cadd	3Dh		
	JBS breg, 6, cadd	3Eh		
	JBS breg, 7, cadd	3Fh		
	Если установлен бит в байте breg, то осуществляется переход на указанный адрес (метку), т. е. к значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода. Если бит очищен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если бит = 1b, тогда PC ← PC + disp			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
JC	**Переход, если C = 1**			
	JC cadd	DBh	2	1
	Если установлен флаг C в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если C = 1b, тогда PC ← PC + disp			
JE	**Переход, если Z = 1**			
	JE cadd	DFh	2	1
	Если установлен флаг Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если Z = 1b, тогда PC ← PC + disp			
JGE	**Переход, если N = 0**			
	JGE cadd	D6h	2	1
	Если сброшен флаг N в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если N = 0b, тогда PC ← PC + disp			
JGT	**Переход, если N = 0 и Z = 0**			
	JGT cadd	D2h	2	1
	Если сброшены флаги N и Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если N = 0b и Z = 0b, тогда PC ← PC + disp			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
JH	**Переход, если C = 1 и Z = 0**			
	JH cadd	D9h	2	1
	Если установлен флаг C и сброшен флаг Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если сброшен флаг C или установлен флаг Z, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если C = 1b и Z = 0b, тогда PC ← PC + disp			
JLE	**Переход, если N = 1 или Z = 1**			
	JLE cadd	DAh	2	1
	Если установлен хотя бы один из флагов N и Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если оба флага сброшены, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если N = 1b или Z = 1b, тогда PC ← PC + disp			
JLT	**Переход, если N = 1**			
	JLT cadd	DEh	2	1
	Если установлен флаг N в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если N = 1b, тогда PC ← PC + disp			
JNC	**Переход, если C = 0**			
	JNC cadd	D3h	2	1
	Если сброшен флаг C в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если C = 0b, тогда PC ← PC + disp			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
JNE	**Переход, если Z = 0**			
	JNE cadd	D7h	2	1
	Если сброшен флаг Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если Z = 0b, тогда PC ← PC + disp			
JNH	**Переход, если C = 0 или Z = 1**			
	JNH cadd	D1h	2	1
	Если сброшен флаг C или установлен флаг Z в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если установлен флаг C и сброшен флаг Z, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если C = 0b и Z = 1b, тогда PC ← PC + disp			
JNST	**Переход, если ST = 0**			
	JNST cadd	D0h	2	1
	Если сброшен флаг ST в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если ST = 0b, тогда PC ← PC + disp			
JNV	**Переход, если V = 0**			
	JNV cadd	D5h	2	1
	Если сброшен флаг V в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если V = 0b, тогда PC ← PC + disp			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
JNVT	**Переход, если VT = 0**			
	JNVT cadd	D4h	2	1
	Если сброшен флаг VT в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг установлен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если VT = 0b, тогда PC ← PC + disp			
JST	**Переход, если ST = 1**			
	JST cadd	D8h	2	1
	Если установлен флаг ST в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если ST = 1b, тогда PC ← PC + disp			
JV	**Переход, если V = 1**			
	JV cadd	DDh	2	1
	Если установлен флаг V в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если V = 1b, тогда PC ← PC + disp			
JVT	**Переход, если VT = 1**			
	JVT cadd	DCh	2	1
	Если установлен флаг VT в регистре PSW, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd). Если флаг сброшен, то управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если VT = 1b, тогда PC ← PC + disp			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
LCALL	**Длинный вызов**			
	LCALL cadd	EFh	3	3
	Вызов подпрограммы, расположенной по указанному адресу (метке). Содержимое программного счетчика загружается в стек (адрес возврата из подпрограммы), после чего к значению программного счетчика PC добавляется 16-битное смещение (disp), равное промежутку от конца команды до метки перехода. Метка перехода не может располагаться за пределами адресов 0000h – FFFFh, которые доступны для младшего слова PC. При вычислении адреса перехода значение старшего слова PC не изменяется. Подпрограмма должна заканчиваться командой RET SP ← SP – 4 (SP) ← PC PC ← PC + (disp)			
VLCALL	**Абсолютный вызов**			
	VLCALL ecadd	E4h	5	3
	Вызов подпрограммы, расположенной по указанному адресу (метке). Содержимое программного счетчика загружается в стек (адрес возврата из подпрограммы), после чего в программный счетчик PC загружается 32-разрядное значение адреса метки перехода. Операнд может иметь любой адрес во всем адресном пространстве. Подпрограмма должна заканчиваться командой RET SP ← SP – 4 (SP) ← PC PC ← (ecadd)			
LD	**Пересылка слова** (DEST, SRC)			
	16-разрядный режим			
	LD wreg, wreg	A0h	3	1 / 2
	LD wreg, #data	A1h	4	1
	LD wreg, [reg]	A2h	3	3
	LD wreg, [reg]+			4
	LD wreg, boffset[reg]	A3h	4	4
	LD wreg, woffset[reg]		5	4
	Загрузка значения слова SRC в слово DEST (DEST) ← (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ELD	**Пересылка двойного слова** (DEST, SRC)			
	32-разрядный режим			
	ELD lreg, lreg	A0h	5	1 / 2
	ELD lreg, #data	A1h	7	1
	ELD lreg, [ereg]	A2h	5	3
	ELD lreg, [ereg]+			4
	ELD lreg, ewoffset[ereg]	A3h	7	4
	Загрузка значения двойного слова SRC в двойное слово DEST (DEST) ← (SRC)			
ELDS	**Пересылка двойного слова** (DEST, SRC)			
	16-разрядный режим			
	См. XCH			
	32-разрядный режим			
	ELDS lreg, lreg	04h	3	1 / 2
	Загрузка значения двойного слова SRC в двойное слово DEST. Адресация регистров как в 16-разрядном режиме (DEST) ← (SRC)			
LDB	**Пересылка байта** (DEST, SRC)			
	LDB breg, breg	B0h	3	1 / 2
	LDB breg, #data	B1h		1
	LDB breg, [reg]	B2h		3
	LDB breg, [reg]+			4
	LDB breg, boffset[reg]	B3h	4	4
	LDB breg, woffset[reg]		5	4
	Загрузка значения байта SRC в байт DEST (DEST) ← (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ELDB	**Пересылка байта** (DEST, SRC)			
	16-разрядный режим			
	См. AND			
	32-разрядный режим			
	ELDB breg, breg	40h	5	1 / 2
	ELDB breg, #data	41h	4	1
	ELDB breg, [ereg]	42h	5	3
	ELDB breg, [ereg]+			4
	ELDB breg, ewoffset[ereg]	43h	7	4
	Загрузка значения байта SRC в байт DEST (DEST) ← (SRC)			
ELDW	**Пересылка слова** (DEST, SRC)			
	16-разрядный режим			
	См. ANDB			
	32-разрядный режим			
	ELDW wreg, wreg	50h	5	1 / 2
	ELDW wreg, #data	51h		1
	ELDW wreg, [ereg]	52h		3
	ELDW wreg, [ereg]+			4
	ELDW wreg, ewoffset[ereg]	53h	7	4
	Загрузка значения байта SRC в байт DEST (DEST) ← (SRC)			
LDBSE	**Загрузка байта со знаковым расширением** (DEST, SRC)			
	16-разрядный режим			
	LDBSE wreg, breg	BCh	3	2
	LDBSE wreg, #data	BDh		2
	LDBSE wreg, [reg]	BEh		3
	LDBSE wreg, [reg]+			4
	LDBSE wreg, boffset[reg]	BFh	4	4
	LDBSE wreg, woffset[reg]		5	4
	Знаковое расширение байта SRC и загрузка его в слово DEST (DEST (младший байт)) ← (SRC) Если (SRC) < 80h, тогда (DEST (старший байт)) ← 00h, иначе (DEST (старший байт)) ← FFh			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ELDSE	**Загрузка слова со знаковым расширением** (DEST, SRC)			
	32-разрядный режим			
	ELDSE lreg, wreg	BCh	5	2
	ELDSE lreg, #data	BDh		2
	ELDSE lreg, [ereg]	BEh		3
	ELDSE lreg, [ereg]+			4
	ELDSE lreg, ewoffset[ereg]	BFh	7	4
	Знаковое расширение слова SRC и загрузка его в двойное слово DEST (DEST (младшее слово)) ← (SRC) Если (SRC) < 8000h, тогда (DEST (старшее слово)) ← 0000h, иначе (DEST (старшее слово)) ← FFFFh			
LDBZE **ELDZE**	**Загрузка байта (слова) с беззнаковым расширением** (DEST, SRC)			
	16-разрядный режим			
	LDBZE wreg, breg	ACh	3	2
	LDBZE wreg, #data	ADh		2
	LDBZE wreg, [reg]	AEh		3
	LDBZE wreg, [reg]+			4
	LDBZE wreg, boffset[reg]	AFh	4	4
	LDBZE wreg, woffset[reg]		5	4
	Беззнаковое расширение байта SRC и загрузка его в слово DEST (DEST (младший байт)) ← (SRC) (DEST (старший байт)) ← 00h			
	32-разрядный режим			
	ELDZE wreg, wreg	ACh	5	2
	ELDZE wreg, #data	ADh		2
	ELDBZE wreg, [reg]	AEh		3
	ELDZE wreg, [reg]+			4
	ELDZE wreg, ewoffset[reg]	AFh	7	4
	Беззнаковое расширение слова SRC и загрузка его в двойное слово DEST (DEST (младшее слово)) ← (SRC) (DEST (старшее слово)) ← 0000h			

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">LJMP</td><td colspan="4">Длинный переход
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>?</td></tr></table></td></tr>
<tr><td>LJMP cadd</td><td>E7h</td><td>3</td><td>1</td></tr>
<tr><td colspan="4">Переход по указанному адресу (метке). Для вычисления перехода к текущему значению программного счетчика PC добавляется 16-битное смещение (disp), равное промежутку от конца команды до метки перехода.

Метка перехода не может располагаться за пределами адресов 0000h – FFFFh, которые доступны для младшего слова PC. При вычислении адреса перехода значение старшего слова PC не изменяется.

PC ← PC + (disp)</td></tr>
<tr><td rowspan="3">VLJMP</td><td colspan="4">Абсолютный переход
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>?</td></tr></table></td></tr>
<tr><td>VLJMP ecadd</td><td>E6h</td><td>5</td><td>1</td></tr>
<tr><td colspan="4">Переход по указанному адресу (метке). В программный счетчик PC загружается 32-разрядное значение адреса метки перехода. Операнд может иметь любой адрес во всем адресном пространстве.

PC ← (ecadd)</td></tr>
<tr><td rowspan="4">MOD</td><td colspan="4">Включение специальных режимов</td></tr>
<tr><td>MOD #1 (режим IDLE)
MOD #2 (режим POWERDOWN)
MOD #3 (режим EINIT)</td><td>F6h</td><td>2</td><td>5</td></tr>
<tr><td colspan="4">Результат выполнения команды зависит от значения 8-битной величины операнда. Контроллер шины завершает цикл упреждающей выборки перед остановкой ЦПУ или сбросом.
После выполнения EINIT нельзя выключить прерывания NMI, ECC_errors, stack errors, а также, запрещается модификация регистров ADDRSELx_LO, ADDRSELx_HI, BUSCONx, INTBUSCON и переключение режима работы блока PTS командой SWC.</td></tr>
</table>

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
MUL **EMUL**	**Знаковое умножение слов** (DEST, SRC)			
	16-разрядный режим			
	MUL lreg, wreg	FE6Ch	4	1 / 5
	MUL lreg, #data	FE6Dh	5	5
	MUL lreg, [reg]	FE6Eh	4	6
	MUL lreg, [reg]+			7
	MUL lreg, boffset[reg]	FE6Fh	5	7
	MUL lreg, woffset[reg]		6	7
	Перемножение слов DEST и SRC с использованием знаковой арифметики и размещение результата в двойном слове-приемнике. (DEST) ← (DEST) × (SRC)			
	32-разрядный режим			
	EMUL lreg, lreg	FE6Ch	6	
	EMUL lreg, [ereg]	FE6Eh	6	
	EMUL lreg, [ereg]+			
	Перемножение двойных слов DEST и SRC с использованием знаковой арифметики. Если результат умножения превышает 32 разряда, то его младшие 31 бит (самый старший бит является знаковым) помещаются в DEST, а остальные биты отбрасываются, и устанавливается флаг V в регистре PSW. (DEST) ← (DEST) × (SRC)			
MUL	**Знаковое умножение слов** (DEST, SRC1, SRC2)			
	MUL lreg, wreg, wreg	FE4Ch	5	1 / 5 / 3
	MUL lreg, wreg, #data	FE4Dh	6	5
	MUL lreg, wreg, [reg]	FE4Eh	5	6
	MUL lreg, wreg, [reg]+			7
	MUL lreg, wreg, boffset[reg]	FE4Fh	6	7
	MUL lreg, wreg, woffset[reg]		7	7
	Перемножение слов SRC1 и SRC2 с использованием знаковой арифметики и размещение двойного слова результата в приемнике (DEST) ← (SRC1) × (SRC2)			

Z	N	C	V	VT	ST
–	–	–	–	–	?

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
MULB	**Знаковое умножение байт** (DEST, SRC)			
	MULB wreg, breg	FE7Ch	4	1 / 4
	MULB wreg, #data	FE7Dh		4
	MULB wreg, [reg]	FE7Eh		5
	MULB wreg, [reg]+			6
	MULB wreg, boffset[reg]	FE7Fh	5	6
	MULB wreg, woffset[reg]		6	6
	Перемножение слова DEST и байта SRC с использованием знаковой арифметики и размещение слова результата в приемнике (DEST) ← (DEST) × (SRC)			
MULB	**Знаковое умножение байт** (DEST, SRC1, SRC2)			
	MULB wreg, breg, breg	FE5Ch	5	1 / 4 / 2
	MULB wreg, breg, #data	FE5Dh		4
	MULB wreg, breg, [reg]	FE5Eh		5
	MULB wreg, breg, [reg]+			6
	MULB wreg, breg, boffset[reg]	FE5Fh	6	6
	MULB wreg, breg, woffset[reg]		7	6
	Перемножение байта SRC1 и байта SRC2 с использованием знаковой арифметики и размещение слова результата в приемнике (DEST) ← (SRC1) × (SRC2)			
MULU	**Беззнаковое умножение слов** (DEST, SRC)			
	16-разрядный режим			
	MULU lreg, wreg	6Ch	3	1 / 5
	MULU lreg, #data	6Dh	4	5
	MULU lreg, [reg]	6Eh	3	6
	MULU lreg, [reg]+			7
	MULU lreg, boffset[reg]	6Fh	4	7
	MULU lreg, woffset[reg]		5	7
	Перемножение слова DEST и слова SRC с использованием беззнаковой арифметики и размещение двойного слова результата в приемнике (DEST) ← (DEST) × (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EMULU	**Беззнаковое умножение слов** (DEST, SRC)			
	32-разрядный режим			
	EMULU lreg, lreg	6Ch	5	1 / 4
	EMULU lreg, #data	6Dh	7	5
	EMULU lreg, [ereg]	6Eh	5	5
	EMULU lreg, [ereg]+			6
	EMULU lreg, ewoffset[ereg]	6Fh	7	6
	Перемножение двойных слов DEST и SRC с использованием беззнаковой арифметики. Если результат умножения превышает 32 разряда, то его младшие 32 бита помещаются в DEST, а остальные биты отбрасываются, и устанавливается флаг V в регистре PSW (DEST) ← (DEST) × (SRC)			
MULU	**Беззнаковое умножение слов** (DEST, SRC1, SRC2)			

Z	N	C	V	VT	ST
–	–	–	–	–	?

MULU lreg, wreg, wreg	4Ch	4	1 / 5 / 3
MULU lreg, wreg, #data	4Dh	5	5
MULU lreg, wreg, [reg]	4Eh	4	6
MULU lreg, wreg, [reg]+			7
MULU lreg, wreg, boffset[reg]	4Fh	5	7
MULU lreg, wreg, woffset[reg]		6	7
Перемножение слова SRC1 и слова SRC2 с использованием беззнаковой арифметики и размещение двойного слова результата в приемнике (DEST) ← (SRC1) × (SRC2)			
MULUB	**Беззнаковое умножение байт** (DEST, SRC)		

Z	N	C	V	VT	ST
–	–	–	–	–	?

MULUB wreg, breg	7Ch	3	1 / 4
MULUB wreg, #data	7Dh		4
MULUB wreg, [reg]	7Eh		5
MULUB wreg, [reg]+			6
MULUB wreg, boffset[reg]	7Fh	4	6
MULUB wreg, woffset[reg]		5	6
Перемножение слова DEST и байта SRC с использованием беззнаковой арифметики и размещение слова результата в приемнике (DEST) ← (DEST) × (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
MULUB	**Беззнаковое умножение байт** (DEST, SRC1, SRC2)			
	MUL wreg, breg, breg	5Ch	4	1 / 4 / 2
	MUL wreg, breg, #data	5Dh		4
	MUL wreg, breg, [reg]	5Eh		5
	MUL wreg, breg, [reg]+			6
	MUL wreg, breg, boffset[reg]	5Fh	5	6
	MUL wreg, breg, woffset[reg]		6	6
	Перемножение байта SRC1 и байта SRC2 с использованием беззнаковой арифметики и размещение слова результата в приемнике (DEST) ← (SRC1) × (SRC2)			
NEG **ENEG**	**Изменение знака слова**			
	16-разрядный режим			
	NEG wreg	03h	2	1 / 3
	32-разрядный режим			
	ENEG lreg	03h	3	1 / 3
	Изменение знака слова операнда на противоположный (DEST) ← – (DEST)			
NEGB	**Изменение знака байта**			
	NEGB breg	13h	2	1 / 3
	Изменение знака байта операнда на противоположный (DEST) ← – (DEST)			
NOP	**Нет операции**			
	NOP	FDh	1	1
	Пустая однобайтная команда. Управление передается следующей по порядку команде PC ← PC + 1			

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">NORML</td><td colspan="4">Нормализация двойного слова (SRC, DEST)
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>?</td><td>0</td><td>–</td><td>–</td><td>–</td></tr></table></td></tr>
<tr><td>NORML lreg, breg</td><td>0Fh</td><td>3</td><td>6</td></tr>
<tr><td colspan="4">Сдвиг двойного слова SRC влево до тех пор, пока его старший значащий бит не окажется равным единице или пока не будет совершен 31 сдвиг. Если после 31 сдвига в старшем значащем бите остался ноль, то процесс сдвига прекращается и устанавливается флаг нуля Z в регистре PSW. В приемнике DEST сохраняется число совершенных сдвигов

(COUNT) ← 00h
Пока (старший бит SRC = 0b) и (COUNT)<31)
Выполнять
(SRC) ← (SRC) × 2
(COUNT) ← (COUNT) + 1
По окончании (DEST) ← (COUNT)</td></tr>
<tr><td rowspan="5">NOT

ENOT</td><td colspan="4">Инверсия слова
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>√</td><td>0</td><td>0</td><td>–</td><td>–</td></tr></table></td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>NOT wreg</td><td>02h</td><td>2</td><td>1 / 3</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td>ENOT lreg</td><td>02h</td><td>3</td><td>1 / 3</td></tr>
<tr><td colspan="4">Побитное инвертирование слова – каждая единица меняется на ноль, каждый ноль меняется на единицу

(DEST) ← NOT (DEST)</td></tr>
<tr><td rowspan="3">NOTB</td><td colspan="4">Инверсия байта
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>√</td><td>0</td><td>0</td><td>–</td><td>–</td></tr></table></td></tr>
<tr><td>NOTB breg</td><td>12h</td><td>2</td><td>1 / 3</td></tr>
<tr><td colspan="4">Побитное инвертирование байта – каждая единица меняется на ноль, каждый ноль меняется на единицу

(DEST) ← NOT (DEST)</td></tr>
</table>

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
OR **EOR**	**Логическое «ИЛИ» слов** (DEST, SRC)			
	16-разрядный режим			
	OR wreg, wreg	80h	3	1 / 4
	OR wreg, #data	81h	4	4
	OR wreg, [reg]	82h	3	5
	OR wreg, [reg]+			6
	OR wreg, boffset[reg]	83h	4	6
	OR wreg, woffset[reg]		5	6
	32-разрядный режим			
	EOR lreg, lreg	80h	5	1 / 4
	EOR lreg, #data	81h	7	4
	EOR lreg, [ereg]	82h	5	5
	EOR lreg, [ereg]+			6
	EOR lreg, ewoffset[ereg]	83h	7	6
	Логическое сложение слова SRC и слова DEST и сохранение результата в приемнике (DEST) ← (DEST) OR (SRC)			
ORB	**Логическое «ИЛИ» байт** (DEST, SRC)			
	ORB breg, breg	90h	3	1 / 4
	ORB breg, #data	91h		4
	ORB breg, [reg]	92h		5
	ORB breg, [reg]+			6
	ORB breg, boffset[reg]	93h	4	6
	ORB breg, woffset[reg]		5	6
	Логическое сложение байта SRC и байта DEST и сохранение результата в приемнике (DEST) ← (DEST) OR (SRC)			
POP	**Чтение слова из стека**			
	16-разрядный режим			
	POP wreg	CCh	2	1 / 3
	POP [reg]	CEh	2	3
	POP [reg]+			4
	POP boffset[reg]	CFh	3	4
	POP woffset[reg]		4	4
	Чтение слова из вершины стека и размещение его в приемнике (DEST) ← (SP) SP ← SP + 2			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
EPOP	**Чтение слова из стека**			
	32-разрядный режим			
	EPOP lreg	CCh	3	1 / 3
	EPOP [ereg]	CEh	3	3
	EPOP [ereg]+			4
	EPOP ewoffset[ereg]	CFh	5	4
	Чтение двойного слова из вершины стека и размещение его в приемнике (DEST) ← (SP) SP ← SP + 4			
POPA	**Чтение всего из стека**	Z N C V VT ST √ √ √ √ √ √		
	POPA	F5h	1	4
	Команда используется взамен команды POPF для поддержки 64 прерываний. Из стека читаются два двойных слова, которые размещаются в регистрах INT_MASK1 и INT_MASK0, затем читается еще одно слово и его младший байт размещается в регистре PSW. В итоге указатель стека SP инкрементируется на десять. Запросы на прерывания не могут идти сразу после этой команды INT_MASK1 ← (SP) SP ← SP + 4 INT_MASK0 ← (SP) SP ← SP + 4 PSW ← (SP) SP ← SP + 2			
POPF	**Чтение флагов из стека**	Z N C V VT ST √ √ √ √ √ √		
	POPF	F3h	1	2
	Чтение слова из вершины стека и размещение его младшего байта в регистре PSW. Запросы на прерывания не могут идти сразу после этой команды (PSW) ← (SP) SP ← SP + 2			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
PUSH **EPUSH**	**Загрузка слова в стек**			
	16-разрядный режим			
	PUSH wreg	C8h	2	1 / 2
	PUSH #data	C9h	3	2
	PUSH [reg]	CAh	2	3
	PUSH [reg]+			4
	PUSH boffset[reg]	CBh	3	4
	PUSH woffset[reg]		4	4
	Загрузка значения операнда в стек (SP) ← SP – 2 (SP) ← (DEST)			
	32-разрядный режим			
	EPUSH lreg	C8h	3	1 / 2
	EPUSH #data	C9h	5	2
	EPUSH [ereg]	CAh	3	3
	EPUSH [ereg]+			4
	EPUSH ewoffset[ereg]	CBh	5	4
	Загрузка значения операнда в стек (SP) ← SP – 4 (SP) ← (DEST)			
PUSHA	**Загрузка всего в стек**	Z N C V VT ST 0 0 0 0 0 0		
	PUSHA	F4h	1	4
	Команда используется взамен команды PUSHF для поддержки 64 прерываний. В стек загружаются: слово, младший байт которого – это PSW, и два двойных слова – регистры INT_MASK0 и INT_MASK1, после чего регистры PSW, INT_MASK0 и INT_MASK1 очищаются. Указатель стека SP декрементируется на десять. Запросы на прерывания не могут идти сразу после этой команды SP ← SP – 2 (SP) ← PSW PSW ← 00h SP ← SP – 4 (SP) ← INT_MASK0 INT_MASK0 ← 00h SP ← SP – 4 (SP) ← INT_MASK1 INT_MASK1 ← 00h			

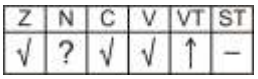
Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
PUSHF	**Загрузка флагов в стек**			
	PUSHF	F2h	1	2
	Загрузка слова, младший байт которого является значением регистра PSW в вершину стека, и очистка регистра PSW. Все прерывания автоматически запрещаются (поскольку, в том числе сбрасывается и бит I). Запросы на прерывания не могут идти сразу после этой команды SP ← SP – 2 (SP) ← PSW PSW ← 0000h			
RET	**Возврат из подпрограммы**			
	RET	F0h	1	3
	Считывание значения адреса (адреса возврата) из вершины стека и размещение его в программном счетчике PC. Указатель стека SP инкрементируется на два PC ← (SP) SP ← SP + 4			
RST	**Сброс системы**			
	RST	FFh	1	40
	Очистка регистра PSW, загрузка в программный счетчик PC значения стартового адреса, загрузка в регистры SFR их начальных значений. В результате выполнения этой команды вывод RESET микроконтроллера находится в состоянии логического нуля 16 машинных тактов PSW ← 00h PC ← стартовый адрес SFR ← начальное значение SFR Вывод RESET в состояние логического нуля			

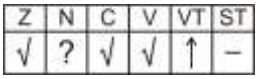
Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
SCALL	**Короткий вызов**			
	SCALL cadd	28h – 2Fh	2	3
	Вызов подпрограммы, расположенной по указанному адресу (метке). Содержимое программного счетчика загружается в стек (адрес возврата из подпрограммы), после чего к значению программного счетчика PC добавляется смещение, задаваемое байтом (disp), и тремя младшими битами (xxx) кода команды, равное промежутку от конца команды до метки перехода. Смещение может быть в диапазоне от –1024 до +1023 включительно. Подпрограмма должна заканчиваться командой RET SP ← SP – 4 (SP) ← PC PC ← PC + ((xxx)(disp))			
SETC	**Установка флага переноса** Z N C V VT ST – – 1 – – –			
	SETC	F9h	1	1
	Запись единицы в бит C регистра PSW C ← 1b			
SHL	**Логический сдвиг слова влево** (DEST, COUNT) Z N C V VT ST √ ? √ √ ↑ –			
	SHL wreg, #count SHL wreg, breg	09h	3	1 / 3 / 4
	Сдвиг слова DEST влево столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра Temp ← (COUNT) Пока Temp ≠ 0 Выполнять C ← Вытесненный старший бит операнда DEST (DEST) ← (DEST) × 2 Temp ← Temp – 1			

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">SHLB</td><td colspan="4">Логический сдвиг байта влево
(DEST, COUNT)
 
</td></tr>
<tr><td>SHLB breg, #count
SHLB breg, breg</td><td>19h</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">Сдвиг байта DEST влево столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный старший бит операнда DEST
(DEST) ← (DEST) × 2
Temp ← Temp – 1</td></tr>
<tr><td rowspan="6">SHLL

ESHLL</td><td colspan="4">Логический сдвиг двойного слова влево
(DEST, COUNT)
 
</td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>SHLL lreg, #count
SHLL lreg, breg</td><td>0Dh</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td>ESHLL lreg, #count</td><td>0Dh</td><td>4</td><td>1 / 3</td></tr>
<tr><td colspan="4">Сдвиг двойного слова DEST влево столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C.
Для команды SHLL количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра.
Для команды ESHLL количество сдвигов может быть от 0 до 31 и задано непосредственно

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный старший бит операнда DEST
(DEST) ← (DEST) × 2
Temp ← Temp – 1</td></tr>
</table>

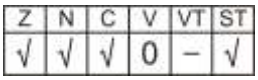
Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">SHR</td><td colspan="4">Логический сдвиг слова вправо
(DEST, COUNT)
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>√</td><td>0</td><td>√</td><td>0</td><td>–</td><td>√</td></tr></table></td></tr>
<tr><td>SHR wreg, #count
SHR wreg, breg</td><td>08h</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">Сдвиг слова DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра.
В начале своего выполнения команда очищает дополнительный флаг переноса ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
<tr><td rowspan="3">SHRA</td><td colspan="4">Арифметический сдвиг слова вправо
(DEST, COUNT)
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>√</td><td>√</td><td>√</td><td>0</td><td>–</td><td>√</td></tr></table></td></tr>
<tr><td>SHRA wreg, #count
SHRA wreg, breg</td><td>0Ah</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">Сдвиг слова DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями, если старший значащий бит исходного слова был ноль, и единицами – в противном случае. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра.
В начале своего выполнения команда очищает дополнительный флаг переноса ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
</table>

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">SHRAB</td><td colspan="4">Арифметический сдвиг байта вправо
(DEST, COUNT)
 
</td></tr>
<tr><td>SHRAB breg, #count
SHRAB breg, breg</td><td>1Ah</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">Сдвиг байта DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями, если старший значащий бит исходного байта был ноль, и единицами в противном случае. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано прямо как содержимое какого-либо регистра.
В начале своего выполнения команда очищает дополнительный флаг переноса ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
</table>

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="6">SHRAL

ESHRAL</td><td colspan="4">Арифметический сдвиг двойного слова вправо (DEST, COUNT)
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>√</td><td>√</td><td>0</td><td>–</td><td>√</td></tr></table></td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>SHRAL lreg, #count
SHRAL lreg, breg</td><td>0Eh</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td>ESHRAL lreg, #count</td><td>0Eh</td><td>4</td><td>1 / 3</td></tr>
<tr><td colspan="4">Сдвиг двойного слова DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями, если старший значащий бит исходного двойного слова был ноль, и единицами в противном случае. Последний вытесненный бит остается в переносе C.
Для команды SHRAL количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано, как содержимое какого-либо регистра.
Для команды ESHRAL количество сдвигов может быть от 0 до 31 и задано непосредственно.
В начале своего выполнения команда очищает дополнительный флаг переноса ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
</table>

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="3">SHRB</td><td colspan="4">Логический сдвиг байта вправо
(DEST, COUNT)
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>√</td><td>0</td><td>√</td><td>0</td><td>–</td><td>√</td></tr></table></td></tr>
<tr><td>SHRB breg, #count
SHRB breg, breg</td><td>18h</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">Сдвиг байта DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C. Количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра. В начале своего выполнения команда очищает флаг ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
<tr><td rowspan="6">SHRL

ESHRL</td><td colspan="4">Логический сдвиг двойного слова вправо (DEST, COUNT)
<table><tr><td>Z</td><td>N</td><td>C</td><td>V</td><td>VT</td><td>ST</td></tr><tr><td>√</td><td>0</td><td>√</td><td>0</td><td>–</td><td>√</td></tr></table></td></tr>
<tr><td colspan="4">16-разрядный режим</td></tr>
<tr><td>SHRL lreg, #count
SHRL lreg, breg</td><td>0Ch</td><td>3</td><td>1 / 3 / 4</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td>ESHRL lreg, #count</td><td>0Ch</td><td>4</td><td>1 / 3</td></tr>
<tr><td colspan="4">Сдвиг двойного слова DEST вправо столько раз, сколько установлено операндом счетчиком (правый операнд). Освобождающиеся биты заполняются нулями. Последний вытесненный бит остается в переносе C.
Для команды SHRL количество сдвигов может быть от 0 до 15 и задано непосредственно или от 0 до 31 и задано как содержимое какого-либо регистра. Для команды ESHRL количество сдвигов может быть от 0 до 31 и задано непосредственно.
В начале своего выполнения команда очищает флаг переноса ST в регистре PSW. Если в течение сдвига в перенос C попадает единица, флаг ST устанавливается

Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1</td></tr>
</table>

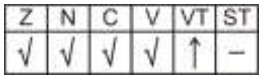
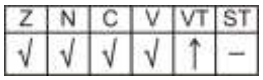
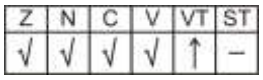
Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
SJMP	**Короткий переход**			
	SJMP cadd	20h – 27h	2	1
	Переход по указанному адресу (метке). Для вычисления перехода к текущему значению программного счетчика PC добавляется смещение, задаваемое байтом (disp) и тремя младшими битами (xxx) кода команды, равное промежутку от конца команды до метки перехода. Смещение может быть в диапазоне от –1024 до +1023 включительно PC ← PC + ((xxx)(disp))			
SKIP	**Нет операции (двухбайтная)**			
	SKIP breg	00h	2	1
	Пустая двухбайтная команда. Управление передается следующей по порядку команде. Значение операнда не важно PC ← PC + 2			
ST **EST**	**Сохранение слова** (SRC, DEST)			
	16-разрядный режим			
	ST wreg, wreg	C0h	3	2
	ST wreg, [reg]	C2h		3
	ST wreg, [reg]+			4
	ST wreg, boffset[reg]	C3h	4	4
	ST wreg, woffset[reg]		5	4
	32-разрядный режим			
	ST wreg, wreg	C0h	3	2
	EST lreg, [ereg]	C2h	5	3
	EST lreg, [ereg]+			4
	EST lreg, ewoffset[ereg]	C3h	7	4
	При индексной адресации возможно использование значения счетчика PC, к которому добавляется 10-битное или 18-битное смещение со знаком			
	EST lreg, eboffset[PC]	1Ch, 1Dh, 1Eh, 1Fh	4	4
	EST lreg, ewoffset[PC]	E8h, E9h, EAh, EBh	5	4
	Сохранение слова SRC в слове DEST (DEST) ← (SRC) Примечания 1 Команда ST с прямой адресацией может использоваться как в 16-разрядном, так и в 32-разрядном режимах. 2 Команда EST с прямой адресацией не реализована.			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
STB	**Сохранение байта** (SRC, DEST)			
	STB breg, breg	C4h	3	2
	STB breg, [reg]	C6h		3
	STB breg, [reg]+			4
	STB breg, boffset[reg]	C7h	4	4
	STB breg, woffset[reg]		5	4
	Сохранение байта SRC в байте DEST (DEST) ← (SRC)			
SUB **ESUB**	**Вычитание слова** (DEST, SRC)	Z: √ N: √ C: √ V: √ VT: ↑ ST: –		
	16-разрядный режим			
	SUB wreg, wreg	68h	3	1 / 4
	SUB wreg, #data	69h	4	4
	SUB wreg, [reg]	6Ah	3	5
	SUB wreg, [reg]+			6
	SUB wreg, boffset[reg]	6Bh	4	6
	SUB wreg, woffset[reg]		5	6
	Вычитание двойного слова SRC из двойного слова DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW. (DEST) ← (DEST) – (SRC)			
	32-разрядный режим			
	ESUB lreg, lreg	68h	5	1 / 4
	ESUB lreg, #data	69h	7	4
	ESUB lreg, [ereg]	6Ah	5	5
	ESUB lreg, [ereg]+			6
	ESUB lreg,ewoffset[ereg]	6Bh	7	6
	Вычитание слова SRC из слова DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW. (DEST) ← (DEST) – (SRC)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
SUB	**Вычитание слов** (DEST, SRC1, SRC2) 			
	SUB wreg, wreg, wreg	48h	4	1 / 4 / 2
	SUB wreg, wreg, #data	49h	5	4
	SUB wreg, wreg, [reg]	4Ah	4	5
	SUB wreg, wreg, [reg]+			6
	SUB wreg, wreg, boffset[reg]	4Bh	5	6
	SUB wreg, wreg, woffset[reg]		6	6
	Вычитание слова SRC2 из слова SRC1 и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (SRC1) – (SRC2)			
SUBB	**Вычитание байт** (DEST, SRC) 			
	SUBB breg, breg	78h	3	1 / 4
	SUBB breg, #data	79h		4
	SUBB breg, [reg]	7Ah		5
	SUBB breg, [reg]+			6
	SUBB breg, boffset[reg]	7Bh	4	6
	SUBB breg, woffset[reg]		5	6
	Вычитание байта SRC из байта DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (DEST) – (SRC)			
SUBB	**Вычитание байт** (DEST, SRC1, SRC2) 			
	SUBB breg, breg, breg	58h	4	1 / 4 / 2
	SUBB breg, breg, #data	59h		4
	SUBB breg, breg, [reg]	5Ah		5
	SUBB breg, breg, [reg]+			6
	SUBB breg, breg, boffset[reg]	5Bh	5	6
	SUBB breg, breg, woffset[reg]		6	6
	Вычитание байта SRC2 из байта SRC1 и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (SRC1) – (SRC2)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
SUBC **ESUBC**	**Вычитание слов с переносом** (DEST, SRC)			
	16-разрядный режим			
	SUBC wreg, wreg	A8h	3	1 / 4
	SUBC wreg, #data	A9h	4	4
	SUBC wreg, [reg]	AAh	3	5
	SUBC wreg, [reg]+			6
	SUBC wreg, boffset[reg]	ABh	4	6
	SUBC wreg, woffset[reg]		5	6
	Вычитание слова SRC и флага переноса из слова DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (DEST) – (SRC) – (1 – C)			
	32-разрядный режим			
	ESUBC lreg, lreg	A8h	5	1 / 4
	ESUBC lreg, #data	A9h	7	4
	ESUBC lreg, [ereg]	AAh	5	5
	ESUBC lreg, [ereg]+			6
	ESUBC lreg, ewoffset[ereg]	ABh	7	6
	Вычитание двойного слова SRC и флага переноса из двойного слова DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (DEST) – (SRC) – (1 – C)			
SUBCB	**Вычитание байт с переносом** (DEST, SRC)			
	SUBCB breg, breg	B8h	3	1 / 4
	SUBCB breg, #data	B9h	3	4
	SUBCB breg, [reg]	BAh	3	5
	SUBCB breg, [reg]+			6
	SUBCB breg, boffset[reg]	BBh	4	6
	SUBCB breg, woffset[reg]		5	6
	Вычитание байта SRC и флага переноса из байта DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW (DEST) ← (DEST) – (SRC) – (1 – C)			

Продолжение таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td>SWC</td><td colspan="4">Переключение системы команд</td></tr>
<tr><td></td><td colspan="4">
<table>
<tr><th rowspan="2">Синтаксис</th><th colspan="2">Разрядность</th><th rowspan="5">10h</th><th rowspan="5">3</th><th rowspan="5">–</th></tr>
<tr><th>PTS, бит</th><th>ядра, бит</th></tr>
<tr><td>SWC 00, 00</td><td>16</td><td>16</td></tr>
<tr><td>SWC 00, 01</td><td>16</td><td>32</td></tr>
<tr><td>SWC 01, 00</td><td>32</td><td>16</td></tr>
<tr><td>SWC 01, 01</td><td>32</td><td>32</td></tr>
</table>
</td></tr>
<tr><td></td><td colspan="4">Переключение режимов работы ядра микроконтроллера и сервера периферийных транзакций PTS. После команды SWC необходимо использовать не менее 16 команд NOP</td></tr>
<tr><td>TIJMP</td><td colspan="4">Косвенный табличный переход
(TBASE, INDEX, INDEX_MASK)</td></tr>
<tr><td></td><td colspan="4">16-разрядный режим</td></tr>
<tr><td></td><td>TIJMP wreg, [reg], #mask</td><td>E2h</td><td>4</td><td>5</td></tr>
<tr><td></td><td colspan="4">Продолжение выполнения программы с адреса, выбранного из таблицы адресов переходов.
Операнд TBASE содержит адрес начала таблицы переходов. Таблица адресов переходов может размещаться на странице памяти размером до 256 байт в любой свободной области, выравненной по границе слова.
Операнд INDEX содержит адрес регистра, в котором находится 7-битный индекс перехода (как при косвенной адресации).
Байтовый непосредственный операнд INDEX_MASK содержит маску, которая накладывается по «И» на INDEX при расчете величины смещения OFFSET (старший бит маски всегда равен нулю).
Адрес перехода DEST X определяется путем добавления к адресу начала таблицы переходов (TBASE) удвоенного значения смещения OFFSET.

OFFSET ← (INDEX) AND (INDEX_MASK)
DEST X ← TBASE + (2×OFFSET)
PC ← (DEST X)

Примечание – Для задания таблицы адресов переходов следует использовать псевдокоманду ассемблера DCW. При этом автоматически выполняется выравнивание таблицы по границе слова.</td></tr>
</table>

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
ETIJMP	**Косвенный табличный переход** (TBASE, INDEX, INDEX_MASK)			
	32-разрядный режим			
	ETIJMP lreg, [lreg], #mask	E2h	6	5
	Продолжение выполнения программы с адреса, выбранного из таблицы адресов переходов. Операнд TBASE содержит адрес начала таблицы переходов. Таблица адресов переходов может размещаться на странице памяти размером до 256 байт в любой свободной области, выравненной по границе двойного слова. Операнд INDEX содержит адрес регистра, в котором находится 7-битный индекс перехода (как при косвенной адресации). Байтовый непосредственный операнд INDEX_MASK содержит маску, которая накладывается по «И» на INDEX при расчете величины смещения OFFSET (старший бит маски всегда равен нулю). Адрес перехода DEST X определяется путем добавления к адресу начала таблицы переходов (TBASE) удвоенного значения смещения OFFSET. OFFSET ← (INDEX) AND (INDEX_MASK) DEST X ← TBASE + (4×OFFSET) PC ← (DEST X) Примечание – Для задания таблицы адресов переходов следует использовать псевдокоманду ассемблера DCD. При этом автоматически выполняется выравнивание таблицы по границе двойного слова.			
TRAP **TRAP1**	**Программное прерывание**			
	TRAP	F7h	1	1
	TRAP1	F1h	1	1
	Выполнение этой команды эквивалентно вызову процедуры обслуживания прерывания и не зависит от состояния флага I в регистре PSW. Запросы на прерывания не могут идти сразу после этой команды. SP ← SP – 4 (SP) ← PC PC ← (2010h)			

Продолжение таблицы Г.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
XCH	**Обмен слов** (DEST, SRC)			
	16-разрядный режим			
	XCH wreg, wreg	04h	3	4
	XCH wreg, boffset[reg]	0Bh	4	6
	XCH wreg, woffset[reg]		5	6
	32-разрядный режим			
	См. ELDS и XCH			
	Обмен словами между SRC и DEST (DEST) ↔ (SRC)			
XCHB	**Обмен байт** (DEST, SRC)			
	XCHB breg, breg	14h	3	4
	XCHB breg, boffset[reg]	1Bh	4	6
	XCHB breg, woffset[reg]		5	6
	Обмен байтами между SRC и DEST (DEST) ↔ (SRC)			
XOR	**Логическое исключающее «ИЛИ» слов** (DEST, SRC)			

Z	N	C	V	VT	ST
√	√	0	0	–	–

16-разрядный режим			
XOR wreg, wreg	84h	3	1 / 4
XOR wreg, #data	85h	4	4
XOR wreg, [reg]	86h	3	5
XOR wreg, [reg]+			6
XOR wreg, boffset[reg]	87h	4	6
XOR wreg, woffset[reg]		5	6
Логическое сложение по модулю два слова SRC и слова DEST и сохранение результата в приемнике (DEST) ← (DEST) XOR (SRC)			

Окончание таблицы Г.1

<table>
<tr><th>Мнемо-ника</th><th>Операция и варианты использования команды</th><th>Опкод</th><th>Длина, байт</th><th>Время выполнения, мт</th></tr>
<tr><td rowspan="9">EXOR</td><td colspan="4">Логическое исключающее «ИЛИ» слов
(DEST, SRC)</td></tr>
<tr><td colspan="4">32-разрядный режим</td></tr>
<tr><td>EXOR lreg, lreg</td><td>84h</td><td>5</td><td>1 / 4</td></tr>
<tr><td>EXOR lreg, #data</td><td>85h</td><td>7</td><td>4</td></tr>
<tr><td>EXOR lreg, [ereg]</td><td rowspan="2">86h</td><td rowspan="2">5</td><td>5</td></tr>
<tr><td>EXOR lreg, [ereg]+</td><td>6</td></tr>
<tr><td>EXOR lreg, ewoffset[ereg]</td><td>87h</td><td>7</td><td>6</td></tr>
<tr><td colspan="4">Логическое сложение по модулю двух двойных слов SRC и DEST и сохранение результата в приемнике

(DEST) ← (DEST) XOR (SRC)</td></tr>
<tr><td rowspan="8">XORB</td><td colspan="4">Логическое исключающее «ИЛИ» байт
(DEST, SRC)
<table><tr><th>Z</th><th>N</th><th>C</th><th>V</th><th>VT</th><th>ST</th></tr><tr><td>√</td><td>√</td><td>0</td><td>0</td><td>–</td><td>–</td></tr></table></td></tr>
<tr><td>XORB breg, breg</td><td>94h</td><td rowspan="4">3 байта</td><td>1 / 4</td></tr>
<tr><td>XORB breg, #data</td><td>95h</td><td>4</td></tr>
<tr><td>XORB breg, [reg]</td><td rowspan="2">96h</td><td>5</td></tr>
<tr><td>XORB breg, [reg]+</td><td>6</td></tr>
<tr><td>XORB breg, boffset[reg]</td><td rowspan="2">97h</td><td>4 байта</td><td>6</td></tr>
<tr><td>XORB breg, woffset[reg]</td><td>5 байт</td><td>6</td></tr>
<tr><td colspan="4">Логическое сложение по модулю два байта SRC и байта DEST и сохранение результата в приемнике

(DEST) ← (DEST) XOR (SRC)</td></tr>
</table>

Особенности системы команд при работе в 16-разрядном режиме

1 Опкоды 1Ch, 1Dh, 1Eh, 1Fh, E8h, E9h, EAh, EBh являются зарезервированными и при обнаружении вызывают генерирование прерывания невыполнимого кода.

2 Опкод E5h используется в качестве старшего байта двухбайтного опкода команд блока FPU, опкод FEh используется в качестве старшего байта двухбайтного опкода команд знакового деления DIV, DIVB и знакового умножения MUL, MULB.

3 Опкод EEh является зарезервированным, но при обнаружении не вызывает генерирование прерывания невыполнимого кода.

Особенности системы команд при работе в 32-разрядном режиме

1 Опкоды C5h и EEh являются зарезервированными.

2 Опкод E5h используется в качестве старшего байта двухбайтного опкода команд блока FPU.

3 Опкод FEh используется в качестве старшего байта двухбайтного опкода команд знакового деления EDIV, DIVB и знакового умножения EMUL, MULB.

Приложение Д

(обязательное)

Система команд блока FPU микроконтроллеров

В таблице Д.1 для каждой команды указано:

- мнемоническое название;
- название выполняемой операции, описание и порядок следования операндов;
- состояние флагов регистра PSW, возникающее в результате выполнения команды;
- варианты использования, с указанием для каждого варианта его опкода, длины и времени выполнения (время выполнения указано в количестве машинных тактов (мт)).

В таблице Д.1 приняты обозначения:

DEST	–	операнд-приемник;
RES	–	операнд-приемник результата вычисления/действия;
SRC	–	операнд-источник;
SRC1	–	первый операнд-источник;
SRC2	–	второй операнд-источник;
fpu_reg	–	регистр блока FPU с номером от 00h до 1Fh;
sfpu_reg	–	регистр блока FPU одинарной точности с номером от 00h до 1Fh;
dfpu_reg	–	регистр блока FPU двойной точности с четным номером от 00h до 1Eh;
wreg	–	регистр слова (16 бит) во внутреннем регистровом файле, с четным адресом;
lreg	–	регистр двойного слова (32 бит) во внутреннем регистровом файле с адресом, кратным четырем;
#data	–	8-/16-/32-разрядная константа;
reg	–	8-/16-/32-разрядный регистр во внутреннем регистровом файле, используемый для косвенной адресации с 8-разрядным адресом;
boffset	–	8-разрядная константа или 8-разрядный регистр во внутреннем регистровом файле. Используется при короткоиндексной адресации для задания смещения со знаком;
woffset	–	16-разрядная константа или 16-разрядный регистр во внутреннем регистровом файле. Используется при длинноиндексной адресации для задания смещения со знаком;
cadd	–	8-/16-/32-разрядный адрес в программном коде, указанный непосредственно или как название метки перехода;
#count	–	4-/5-разрядная константа для задания количества сдвигов (от 0 до 31) при сдвиговых операциях.

E5																	
Код	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF	Код
0x	FLD				FSTFR	FST			FADDS	FADDD	FSUBS	FSUBD	FMULS	FMULD	FSMULD	FDIVS	0x
	di	im	in	ix	di	di	in	ix	di	di	di	di	di	di	di	di	
1x	FDIVD di	FSQRTS di	FSQRTD di	FITOS di	FITOD di	FSTOI di	FDTOI di	FSTOI_RND di	FDTOI_RND di	FSTOD di	FDTOS di	FCMPS di	FCMPD di	FCMPES di	FCMPED di	FABSS di	1x
2x	FNEGS di	FMOVS di	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2x
3x - Cx	-																3x - Cx
Dx	-	FJNH ix	-	-	-	-	-	FJNE ix	-	FJH ix	-	-	-	-	-	FJE ix	Dx
Ex, Fx	-																Ex, Fx
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF	

Цветом отмечены команды, операндами которых могут быть только регистры FR0-FR31 блока FPU с адресами 00h-1fh

Рисунок Д.1 – Система команд блока FPU для 16-разрядного режима работы микроконтроллера

E5																	
Код	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF	Код
0x	EFLD	FLD	EFLD		FSTFR	EFST			FADDS	FADDD	FSUBS	FSUBD	FMULS	FMULD	FSMULD	FDIVS	0x
	di	im	in	ix	di	di	in	ix	di	di	di	di	di	di	di	di	
1x	FDIVD di	FSQRTS di	FSQRTD di	FITOS di	FITOD di	FSTOI di	FDTOI di	FSTOI_RND di	FDTOI_RND di	FSTOD di	FDTOS di	FCMPS di	FCMPD di	FCMPES di	FCMPED di	FABSS di	1x
2x	FNEGS di	FMOVS di	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2x
3x - Cx	-																3x - Cx
Dx	-	FJNH ix	-	-	-	-	-	FJNE ix	-	FJH ix	-	-	-	-	-	FJE ix	Dx
Ex, Fx	-																Ex, Fx
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF	

Цветом отмечены команды, операндами которых могут быть только регистры FR0-FR31 блока FPU с адресами 00h-1fh

Рисунок Д.2 – Система команд блока FPU для 32-разрядного режима работы микроконтроллера

Таблица Д.1 – Система команд FPU

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FABSS	**Абсолютное значение числа** (RES, SRC)			
	FABSS sfpu_reg, sfpu_reg	E51Fh	4	4
	Вычисление абсолютного значения SRC и сохранение в RES (SRC (знаковый бит) ← 0b (RES) ← (SRC)			
FADDD **FADDS**	**Сложение чисел** (RES, SRC1, SRC2)			
	FADDD dfpu_reg, dfpu_reg, dfpu_reg	E509h	5	4
	FADDS sfpu_reg, sfpu_reg, sfpu_reg	E508h	5	4
	Сложение числа SRC1 и числа SRC2 и сохранение суммы в RES (RES) ← (SRC1) + (SRC2)			
FCMPD **FCMPS**	**Сравнение вещественных чисел** (SRC1, SRC2)			
	FCMPD dfpu_reg, dfpu_reg	E51Ch	4	4
	FCMPS sfpu_reg, sfpu_reg	E51Bh	4	4
	Число SRC1 сравнивается с числом SRC2. Результат сравнения отражается в поле CC регистра флагов FFLAGS. Некорректная исключительная ситуация NV возникает в случае, если хотя бы один из операндов сигнальное не число (сигнальный NaN). Если SRC1 = SRC2, то CC ← 00b, если SRC1 < SRC2, то CC ← 01b, если SRC1 > SRC2, то CC ← 10b, если SRC1 или SRC2 не число, то CC ← 11b			

UNF	INV	OF	UF	DZ	NX
-	-	-	-	-	-

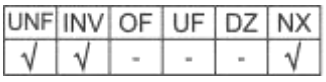
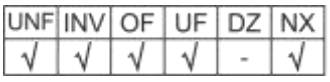
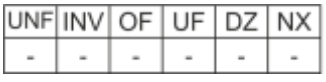
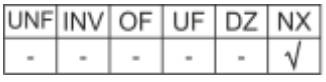
UNF	INV	OF	UF	DZ	NX
√	√	√	√	-	√

UNF	INV	OF	UF	DZ	NX
-	√	-	-	-	-

Продолжение таблицы Д.1

Мнемоника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FCMPED **FCMPES**	**Сравнение вещественных чисел** (SRC1, SRC2) UNF: - ; INV: √ ; OF: - ; UF: - ; DZ: - ; NX: -			
	FCMPED dfpu_reg, dfpu_reg	E51Eh	4	4
	FCMPES sfpu_reg, sfpu_reg	E51Dh	4	4
	Число SRC1 сравнивается с числом SRC2. Результат сравнения отражается в поле CC регистра флагов FFLAGS. Некорректная исключительная ситуация NV возникает в случае, если хотя бы один из операндов не число (сигнальный или тихий NaN). Если SRC1 = SRC2, то CC ← 00b, если SRC1 < SRC2, то CC ← 01b, если SRC1 > SRC2, то CC ← 10b, если SRC1 или SRC2 не число, то CC ← 11b			
FDIVD **FDIVS**	**Деление чисел** (RES, SRC2, SRC1) UNF: √ ; INV: √ ; OF: √ ; UF: √ ; DZ: √ ; NX: √			
	FDIVD dfpu_reg, dfpu_reg, dfpu_reg	E510h	5	17
	FDIVS sfpu_reg, sfpu_reg, sfpu_reg	E50Fh	5	16
	Деление числа SRC2 на SRC1 и сохранение результата в младшем двойном слове RES, а остатка – в старшем слове RES. (RES (младшее слово)) ← (SRC2)/(SRC1) (RES (старшее слово)) ← Остаток от (SRC2)/(SRC1)			
FDTOI **FSTOI**	**Преобразование вещественного числа в целое** (RES, SRC) UNF: √ ; INV: √ ; OF: - ; UF: - ; DZ: - ; NX: √			
	FDTOI sfpu_reg, dfpu_reg	E516h	4	4
	FSTOI sfpu_reg, sfpu _reg	E515h	4	4
	Преобразование вещественного числа в целое число и размещение результата в RES. Результат округляется в режиме «округления до нуля» (RES) ← (SRC)			

Продолжение таблицы Д.1

Мнемоника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FDTOI_RND **FSTOI_RND**	**Преобразование вещественного числа в целое** (RES, SRC)  			
	FDTOI_RND sfpu_reg, dfpu_reg	E518 h	4	4
	FSTOI_RND sfpu_reg, sfpu _reg	E517 h	4	4
	Преобразование вещественного числа в целое число и размещение результата в RES. Режим округления задается полем RNDMODE регистра FCTRL (RES) ← (SRC)			
FDTOS	**Преобразование вещественных чисел** (RES, SRC)  			
	FDTOS sfpu_reg, dfpu_reg	E51A h	4	4
	Преобразование вещественного числа двойной точности в вещественное число одинарной точности и размещение результата в RES (RES) ← (SRC)			
FITOD **FITOS**	**Преобразование целого числа в вещественное** (RES, SRC)   			
	FITOD dfpu_reg, fpu_reg	E514 h	4	4
	FITOS sfpu_reg, fpu _reg	E513 h	4	4
	Преобразование целого числа в вещественное целое и размещение результата в RES (RES) ← (SRC)			

Продолжение таблицы Д.1

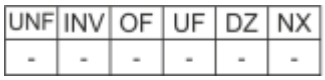
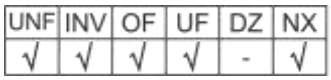
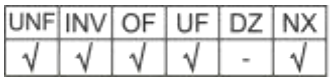
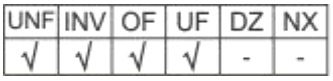
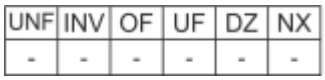
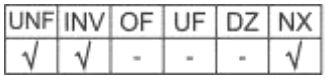
Мнемоника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FJE	**Переход, если операнды равны**			
	FJE cadd	E5DFh	3	1
	Если результат операции сравнения показал равенство операндов, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd), иначе управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если в регистре FFLAGS поле CC = 00b, тогда PC ← PC + disp			
FJH	**Переход, если первый операнд больше второго**			
	FJH cadd	E5D9h	3	1
	Если результат операции показал превышение значения первого операнда (SRC1) над значением второго (SRC2), то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd), иначе управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если в регистре FFLAGS поле CC = 10b, тогда PC ← PC + disp			

Продолжение таблицы Д.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FJNE	**Переход, если операнды не равны**			
	FJNE cadd	E5D7h	3	1
	Если результат операциии сравнения показал, что значения операндов не равны, то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd), иначе управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если в регистре FFLAGS поле CC ≠ 00b, тогда PC ← PC + disp			
FJNH	**Переход, если первый операнд меньше второго**			
	FJNH cadd	E5D1h	3	1
	Если результат операции показал превышение второго операнда (SRC2) над значением первого (SRC1), то к текущему значению программного счетчика PC добавляется смещение (disp) между концом команды и адресом метки перехода (cadd), иначе управление передается следующей по порядку команде. Смещение может быть в диапазоне от –128 до +127 Если в регистре FFLAGS поле CC = 01b, тогда PC ← PC + disp			
FLD **EFLD**	**Пересылка числа в FPU** (DEST, SRC)			
	16-разрядный режим			
	FLD fpu_reg, wreg	E500h	4	1
	FLD fpu_reg, #data	E501h	7	1
	FLD fpu_reg, [reg]	E502h	4	2
	FLD fpu_reg, [reg]+			3
	FLD fpu_reg, boffset[reg] (255)	E503h	5	3
	32-разрядный режим			
	EFLD fpu_reg, lreg	E500h	5	1
	EFLD fpu_reg, #data	E501h	7	1
	EFLD fpu_reg, [reg]	E502h	5	2
	EFLD fpu_reg, [reg]+			3
	EFLD fpu_reg, woffset[reg]	E503h		3

	Загрузка в регистр блока FPU слова из регистра SRC (DEST) ← (SRC)

Продолжение таблицы Д.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FMOVS	**Пересылка числа** (DEST, SRC) 			
	FMOVS sfpu_reg, sfpu_reg	E521 h	4	4
	Копирование значения SRC в DEST (DEST) ← (SRC)			
FMULD **FMULS** **FSMUL D**	**Умножение чисел** (RES, SRC1, SRC2)   			
	FMULD dfpu_reg, dfpu_reg, dfpu_reg	E50D h	5	4
	FMULS sfpu_reg, sfpu_reg, sfpu_reg	E50C h	5	4
	FSMULD dfpu_reg, sfpu_reg, sfpu_reg	E50Eh	5	4
	Умножение числа SRC1 и числа SRC2 и сохранение результата в RES. Команда FSMULD позволяет получить произведение двойной точности двух чисел одинарной точности (RES) ← (SRC1) × (SRC2)			
FNEGS	**Изменение знака числа** (RES, SRC) 			
	FNEGS sfpu_reg, sfpu_reg	E520 h	4	4
	Изменение знака числа SRC на противоположный и размещение результата в RES (RES) ← – (SRC)			
FSQRTD **FSQRTS**	**Вычисление квадратного корня числа** (RES, SRC) 			
	FSQRTD dfpu_reg, dfpu_reg	E512h	4	25

	FSQRTS sfpu_reg, sfpu_reg	E511h	4	24
	Вычисление квадратного корня числа SRC и размещение результата в RES (RES) ← $\sqrt{(SRC)}$			

Продолжение таблицы Д.1

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина, байт	Время выполнения, мт
FST **EFST**	**Пересылка числа из FPU** (SRC, DEST)			
	16-разрядный режим			
	FST fpu_reg, wreg	E505h	4	1
	FST fpu_reg, [reg]	E506h	4	2
	FST fpu_reg, [reg]+			3
	FST fpu_reg, boffset[reg] (255)	E507h	5	3
	32-разрядный режим			
	EFST fpu_reg, lreg	E505h	5	1
	EFST fpu_reg, [reg]	E506h	5	2
	EFST fpu_reg, [reg]+			3
	EFST fpu_reg, woffset[reg]	E507h		3
	Пересылка числа из регистра блока FPU в регистр DST (DEST) ← (SRC)			
FSTFR	**Пересылка чисел внутри FPU** (SRC, DEST)			
	FSTFR fpu_reg, fpu_reg	E504h	4	1
	Пересылка числа из регистра SRC в регистр DST блока FPU (DEST) ← (SRC)			
FSTOD	**Преобразование вещественных чисел** (RES, SRC)			

UNF	INV	OF	UF	DZ	NX
√	√	-	-	-	-

FDTOS dfpu_reg, sfpu_reg	E519h	4	4

Преобразование вещественного числа одинарной точности в вещественное число двойной точности и размещение результата в RES

(RES) ← (SRC)

<table>
<tr><td rowspan="4">FSUBD

FSUBS</td><td colspan="4">Вычитание числа (RES, SRC1, SRC2)
<table><tr><td>UNF</td><td>INV</td><td>OF</td><td>UF</td><td>DZ</td><td>NX</td></tr><tr><td>√</td><td>√</td><td>√</td><td>√</td><td>-</td><td>√</td></tr></table></td></tr>
<tr><td>FSUBD dfpu_reg, dfpu_reg, dfpu_reg</td><td>E50Bh</td><td>4</td><td>4</td></tr>
<tr><td>FSUBS sfpu_reg, sfpu_reg, sfpu_reg</td><td>E50Ah</td><td>4</td><td>4</td></tr>
<tr><td colspan="4">Вычитание слова SRC2 из слова SRC1 и сохранение результата в RES. Результат операции см. флаг CC в регистре FCTRL

(RES) ← (SRC1) – (SRC2)</td></tr>
</table>

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Подпись	Дата
	измененных	замененных	новых	аннулированных				