МИКРОСХЕМЫ ИНТЕГРАЛЬНЫЕ

1874ВЕ7Т, 1874ВЕ71Т

Руководство пользователя

2015

Содержание

Введение

В настоящем руководстве КФДЛ.431295.045 приведено описание архитектуры, функционального построения, системы команд и особенностей применения микросхем 1874ВЕ7Т, 1874ВЕ71Т. Основное отличие микросхемы 1874ВЕ71Т от 1874ВЕ7Т заключается в наличии интерфейса JTAG. Микросхема представляет собой СБИС спецстойкого однокристального 16-разрядного микроконтроллера с тактовой частотой до 24 МГц, регистровым ОЗУ объемом 2 024 байта, модулем отладки и сторожевым таймером. Периферия микроконтроллера представлена 3-канальным блоком ШИМ, двумя последовательными портами ввода-вывода типа UART, блоком высокоскоростного ввода-вывода HSI – HSO, контроллерами интерфейсов SPI, I2C, SpaceWire, ГОСТ Р 52070–2003 и 16-канальным (12-канальным для 1874ВЕ71Т) 12-разрядным АЦП с возможностью работы с дифференциальными сигналами и двумя дополнительными аппаратными битами повышения точности.

Изделие служит цели развития отечественной элементной базы для применения в различных системах управлениях, радиосвязи, радиолокации и других изделиях, требующих точные аналогово-цифровые преобразования в условиях радиационного воздействия.

Настоящее руководство КФДЛ.431295.045 может служить практическим руководством по применению микроконтроллера для разработчиков систем на основе ИС 1874ВЕ7Т, 1874ВЕ71Т.

1 Назначение и область применения

Архитектура микроконтроллера ориентирована на создание управляющих систем, функционирующих в режиме реального времени с возможностью адаптации и модификации под конкретные приложения. В микроконтроллере реализована полная программная совместимость с применяемым серийным МК 1874ВЕ05Т. Разработанный 16-разрядный микроконтроллер с АЦП, контроллерами интерфейсов ГОСТ Р 52070–2003 и SpaceWire и блоком ШИМ может использоваться в цифровой аппаратуре управления электродвигателями, средствах радиолокации и другой аппаратуре с повышенными требованиями по стойкости к спецвоздействиям.

Возможность гибкого управления энергопотреблением микроконтроллера (три режима пониженного энергопотребления, возможность отключения неиспользуемых блоков) позволит использовать ИС 1874ВЕ7Т, 1874ВЕ71Т в критичных к потреблению приложениях.

2 Краткое техническое описание микроконтроллера

2.1 Функциональные параметры

Функциональные параметры микроконтроллера 1874ВЕ7Т, 1874ВЕ71Т:

разрядность данных, бит	16
система команд и архитектура	AMCS-96
тактовая частота, МГц	до 24
регистровое ОЗУ, бит	2024 × 8
динамически конфигурируемая шина данных, бит	8 или 16
адресуемая память, бит	64К × 8
число параллельных 8-разрядных портов ввода-вывода	6/5*
число параллельных 4-разрядных портов ввода	1**
число источников прерывания	44
сервер периферийных транзакций PTS	1
число 16-разрядных таймеров/счетчиков	2
число последовательных портов ввода-вывода типа UART	2
число контроллеров интерфейса SPI	1
число контроллеров интерфейса I2C	1
число контроллеров интерфейса SpaceWire	1
число контроллеров интерфейса по ГОСТ Р 52070–2003	1
ОЗУ сообщений контроллера интерфейса ГОСТ Р 52070–2003, байт	4К
число блоков высокоскоростного ввода-вывода HSI-HSO	1
число каналов блока ШИМ	3
число каналов встроенного АЦП	16/12*
число разрядов АЦП	12 + 2
число программируемых 16-разрядных сторожевых таймеров	1
число модулей отладки DEBUG UNIT	1
порт JTAG	1**
число режимов энергопотребления	3

Микросхемы 1874ВЕ7Т, 1874ВЕ71Т выполнены в металлокерамическом планарном корпусе 4235.88-1 с четырехсторонним расположением выводов и предназначены для ручной и автоматической сборки в соответствии с ГОСТ РВ 20.39.412–97. Масса микросхем – не более 4,5 г.

Герметизация микросхем осуществляется шовной роликовой сваркой. Показатель герметичности микросхем по эквивалентному нормализованному потоку – не более $6{,}65 \cdot 10^{-3}$ Па·см^3/с.

Значение собственной резонансной частоты – не менее 7,2 кГц.

Структурная схема микроконтроллеров 1874ВЕ7Т, 1874ВЕ71Т приведена на рисунке 2.1. Условное графическое обозначение микроконтроллера 1874ВЕ7Т представлено на рисунке 2.2, микроконтроллера 1874ВЕ71Т – на рисунке 2.3. Назначение выводов, имеющих идентичную функциональность для обоих контроллеров, указано в таблице 2.1. Выводы микроконтроллеров, имеющие одинаковые номера, но отличающиеся по функциональному назначению, для 1874ВЕ7Т указаны в таблице 2.2, для 1874ВЕ71Т – в таблице 2.3.

---

* Первая цифра относится к ИС 1874ВЕ7Т, вторая – к ИС 1874ВЕ71Т.

** Только в ИС 1874ВЕ71Т.

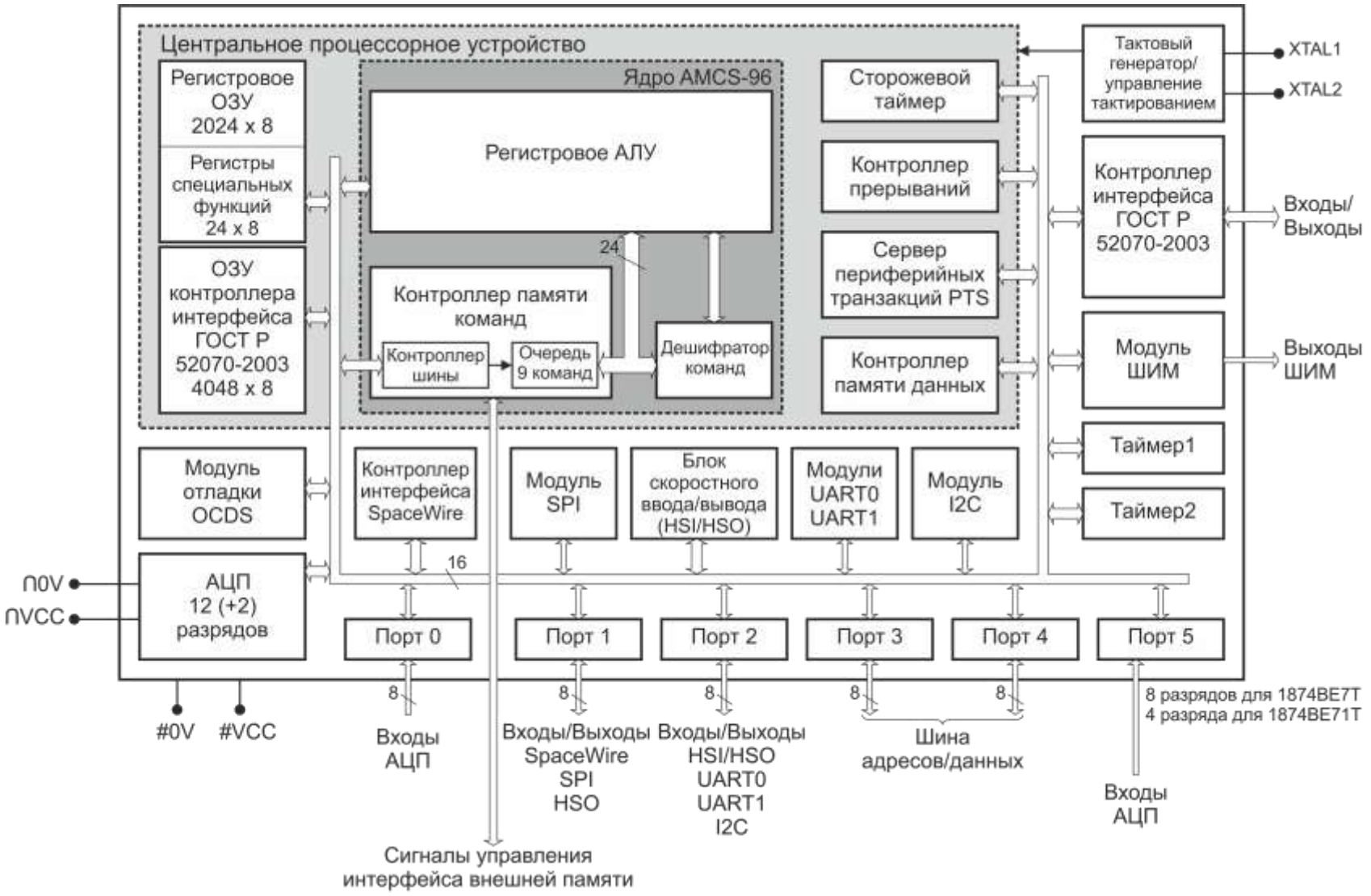


Рисунок 2.1 – Структурная схема микроконтроллеров 1874BE7T и 1874BE71T
(в микросхеме 1874BE7T интерфейс JTAG отсутствует)

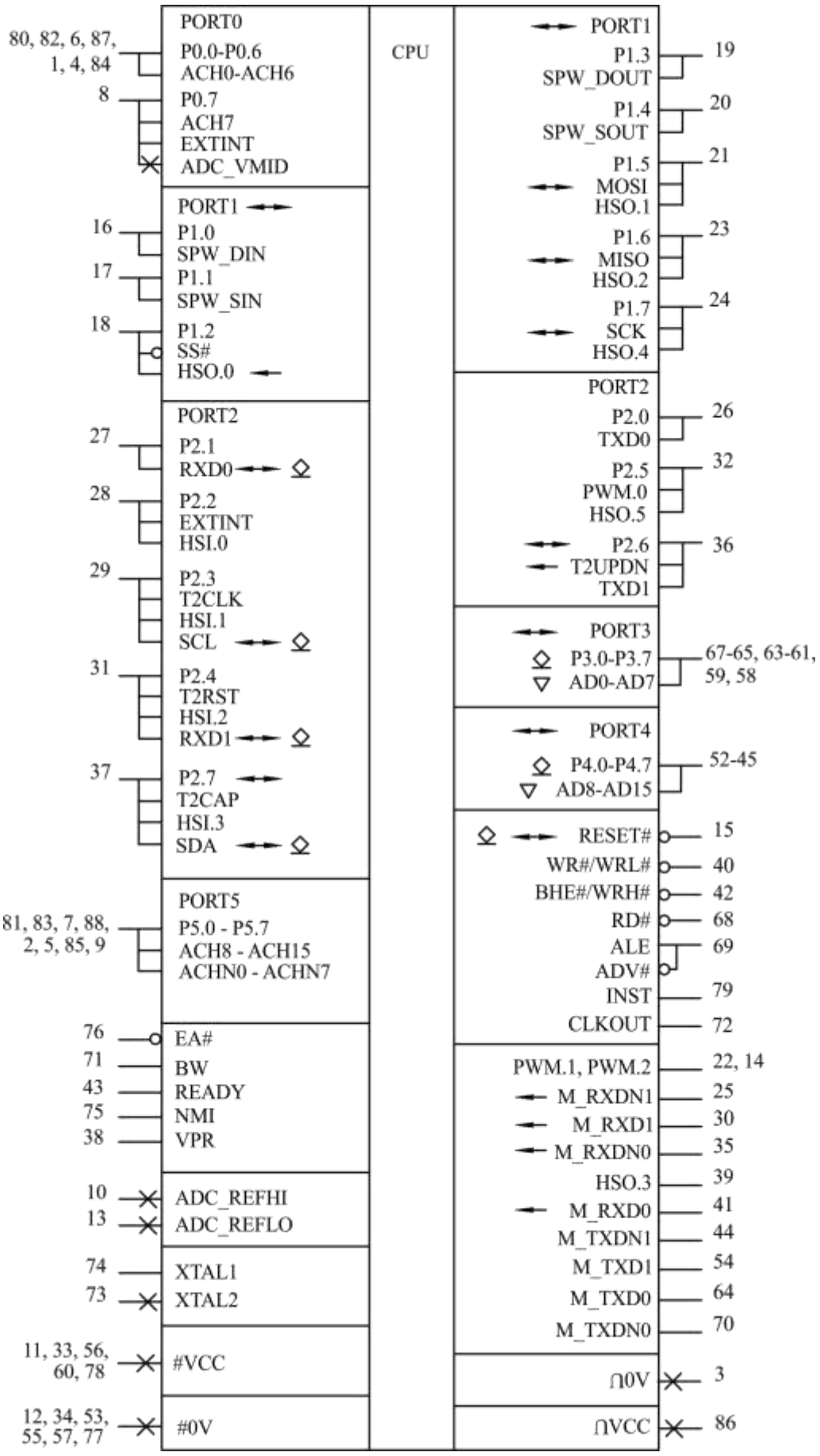


Рисунок 2.2 – Условное графическое изображение микросхемы 1874ВЕ7Т в корпусе 4235.88-1

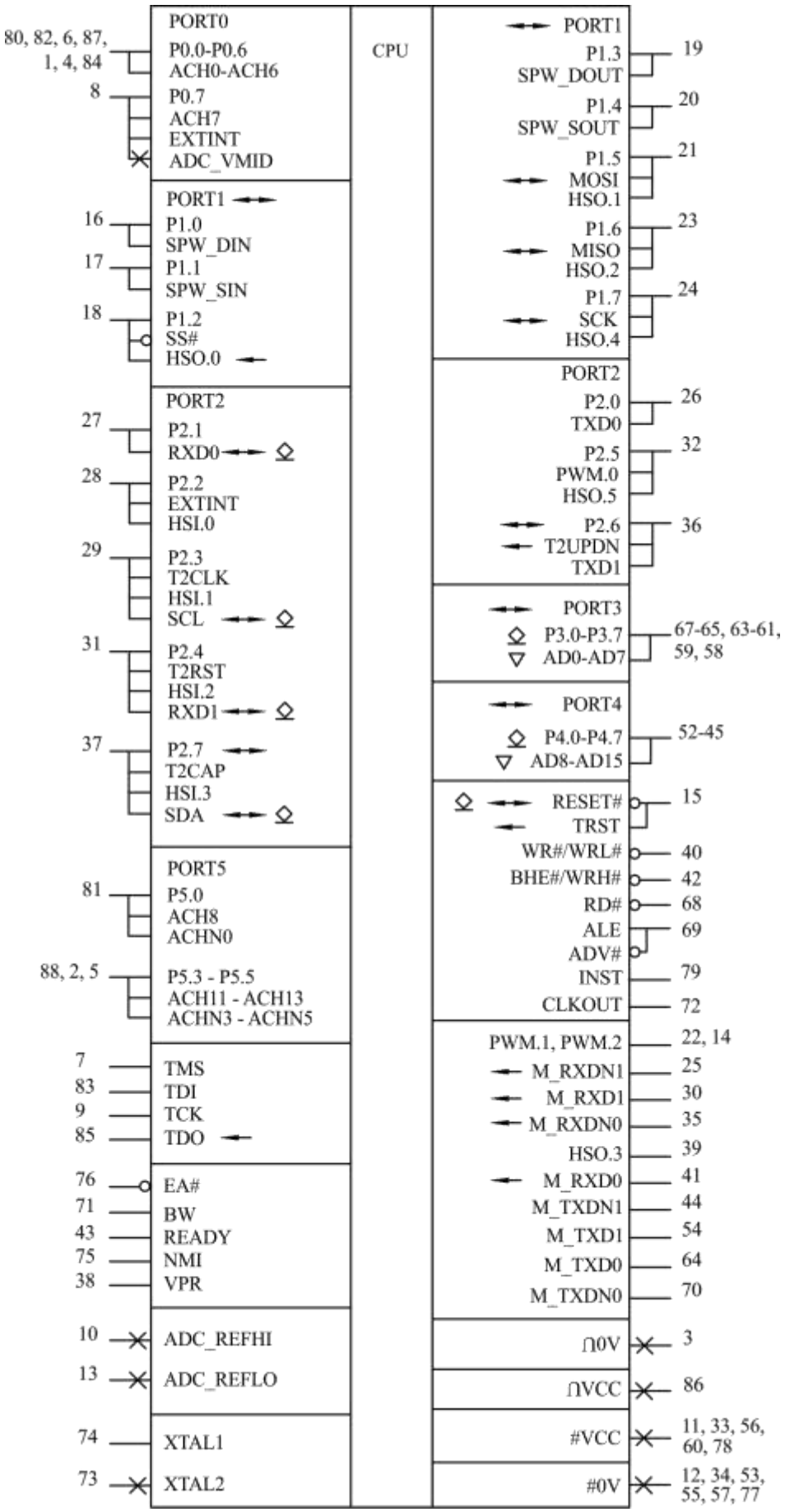


Рисунок 2.3 – Условное графическое изображение микросхемы 1874ВЕ71Т в корпусе 4235.88-1

Таблица 2.1 – Назначение выводов микроконтроллеров 1874ВЕ7Т, 1874ВЕ71Т

Обозначение вывода	Номер вывода	Функциональное назначение	Тип вывода
1	2	3	4
P0.0 ACH0	80	Вход «порт 0, 0 разряд» Вход АЦП, канал 0	I I
P0.1 ACH1	82	Вход «порт 0, 1 разряд» Вход АЦП, канал 1	I I
P0.2 ACH2	6	Вход «порт 0, 2 разряд» Вход АЦП, канал 2	I I
P0.3 ACH3	87	Вход «порт 0, 3 разряд» Вход АЦП, канал 3	I I
P0.4 ACH4	1	Вход «порт 0, 4 разряд» Вход АЦП, канал 4	I I
P0.5 ACH5	4	Вход «порт 0, 5 разряд» Вход АЦП, канал 5	I I
P0.6 ACH6	84	Вход «порт 0, 6 разряд» Вход АЦП, канал 6	I I
P0.7 ACH7 EXTINT ADC_VMID	8	Вход «порт 0, 7 разряд» Вход АЦП, канал 7 Вход «сигнал внешнего прерывания» Напряжение средней точки АЦП	I I I –
P1.0 SPW_DIN	16	Вход/выход «порт 1, 0 разряд» Вход последовательных данных SpaceWire	I/O I
P1.1 SPW_SIN	17	Вход/выход «порт 1, 1 разряд» Вход стробирующего сигнала SpaceWire	I/O I
P1.2 SS# HSO.0	18	Вход/выход «порт 1, 2 разряд» Инверсный вход выбора ведомого порта SPI Выход «быстрый вывод, канал 0»	I/O I O
P1.3 SPW_DOUT	19	Вход/выход «порт 1, 3 разряд» Выход последовательных данных SpaceWire	I/O O
P1.4 SPW_SOUT	20	Вход/выход «порт 1, 4 разряд» Выход стробирующего сигнала SpaceWire	I/O O
P1.5 MOSI HSO.1	21	Вход/выход «порт 1, 5 разряд» Вход данных ведомого/выход данных ведущего порта SPI Выход «быстрый вывод, канал 1»	I/O I/O O
P1.6 MISO HSO.2	23	Вход/выход «порт 1, 6 разряд» Выход данных ведомого/вход данных ведущего порта SPI Выход «быстрый вывод, канал 2»	I/O I/O O
P1.7 SCK HSO.4	24	Вход/выход «порт 1, 7 разряд» Вход/выход тактовой частоты порта SPI Выход «быстрый вывод, канал 4»	I/O I/O O
P2.0 TXD0	26	Выход «порт 2, 0 разряд» Выход последовательных данных UART0	O O

Продолжение таблицы 2.1

1	2	3	4
P2.1	27	Вход «порт 2, 1 разряд»	I
RXD0		Вход/выход последовательных данных UART0	I/O/2
P2.2	28	Вход «порт 2, 2 разряд»	I
EXTINT		Вход «сигнал внешнего прерывания»	I
HSI.0		Вход «быстрый ввод, канал 0»	I
P2.3	29	Вход «порт 2, 3 разряд»	I
T2CLK		Вход «синхронизация таймера 2»	I
HSI.1		Вход «быстрый ввод, канал 1»	I
SCL		Вход/выход синхронизации порта I2C	I/O/2
P2.4	31	Вход «порт 2, 4 разряд»	I
T2RST		Вход «сброс таймера 2»	I
RXD1		Вход/выход последовательных данных порта UART1	I/O/2
HSI.2		Вход «быстрый ввод, канал 2»	I
P2.5	32	Выход «порт 2, 5 разряд»	O
PWM.0		Выход «ШИМ0»	O
HSO.5		Выход «быстрый вывод, канал 5»	O
P2.6	36	Вход/выход «порт 2, 6 разряд»	I/O
T2UPDN		Вход «выбор режима таймера 2»	I
TXD1		Выход последовательных данных порта UART1	O
P2.7	37	Вход/выход «порт 2, 7 разряд»	I/O
T2CAP		Вход «выборка данных таймера 2»	I
HSI.3		Вход «быстрый ввод, канал 3»	I
SDA		Вход/выход данных шины I2C	I/O/2
P3.0	67	Вход/выход «порт 3, 0 разряд»	I/O/2
AD0		Вход/выход «адрес/данные, 0 разряд»	I/O/Z
P3.1	66	Вход/выход «порт 3, 1 разряд»	I/O/2
AD1		Вход/выход «адрес/данные, 1 разряд»	I/O/Z
P3.2	65	Вход/выход «порт 3, 2 разряд»	I/O/2
AD2		Вход/выход «адрес/данные, 2 разряд»	I/O/Z
P3.3	63	Вход/выход «порт 3, 3 разряд»	I/O/2
AD3		Вход/выход «адрес/данные, 3 разряд»	I/O/Z
P3.4	62	Вход/выход «порт 3, 4 разряд»	I/O/2
AD4		Вход/выход «адрес/данные, 4 разряд»	I/O/Z
P3.5	61	Вход/выход «порт 3, 5 разряд»	I/O/2
AD5		Вход/выход «адрес/данные, 5 разряд»	I/O/Z
P3.6	59	Вход/выход «порт 3, 6 разряд»	I/O/2
AD6		Вход/выход «адрес/данные, 6 разряд»	I/O/Z
P3.7	58	Вход/выход «порт 3, 7 разряд»	I/O/2
AD7		Вход/выход «адрес/данные, 7 разряд»	I/O/Z
P4.0	52	Вход/выход «порт 4, 0 разряд»	I/O/2
AD8		Вход/выход «адрес/данные, 8 разряд»	I/O/Z
P4.1	51	Вход/выход «порт 4, 1 разряд»	I/O/2
AD9		Вход/выход «адрес/данные, 9 разряд»	I/O/Z
P4.2	50	Вход/выход «порт 4, 2 разряд»	I/O/2
AD10		Вход/выход «адрес/данные, 10 разряд»	I/O/Z

Продолжение таблицы 2.1

1	2	3	4
P4.3 AD11	49	Вход/выход «порт 4, 3 разряд» Вход/выход «адрес/данные, 11 разряд»	I/O/2 I/O/Z
P4.4 AD12	48	Вход/выход «порт 4, 4 разряд» Вход/выход «адрес/данные, 12 разряд»	I/O/2 I/O/Z
P4.5 AD13	47	Вход/выход «порт 4, 5 разряд» Вход/выход «адрес/данные, 13 разряд»	I/O/2 I/O/Z
P4.6 AD14	46	Вход/выход «порт 4, 6 разряд» Вход/выход «адрес/данные, 14 разряд»	I/O/2 I/O/Z
P4.7 AD15	45	Вход/выход «порт 4, 7 разряд» Вход/выход «адрес/данные, 15 разряд»	I/O/2 I/O/Z
P5.0 ACH8 ACHN0	81	Вход «порт 5, 0 разряд» Вход АЦП, канал 8 Инверсный вход АЦП, канал 0	I I I
P5.3 ACH11 ACHN3	88	Вход «порт 5, 3 разряд» Вход АЦП, канал 11 Инверсный вход АЦП, канал 3	I I I
P5.4 ACH12 ACHN4	2	Вход «порт 5, 4 разряд» Вход АЦП, канал 12 Инверсный вход АЦП, канал 4	I I I
P5.5 ACH13 ACHN5	5	Вход «порт 5, 5 разряд» Вход АЦП, канал 13 Инверсный вход АЦП, канал 5	I I I
HSO.3	39	Выход «быстрый вывод, канал 3»	O
PWM.1	22	Выход «ШИМ1»	O
PWM.2	14	Выход «ШИМ2»	O
M_TXD0	64	Прямой выход последовательных данных основного канала передачи MIL_STD	O
M_TXD1	54	Прямой выход последовательных данных резервного канала передачи MIL_STD	O
M_TXDN0	70	Инверсный выход последовательных данных основного канала передачи MIL_STD	O
M_TXDN1	44	Инверсный выход последовательных данных резервного канала передачи MIL_STD	O
M_RXD0	41	Прямой вход последовательных данных основного канала приема MIL_STD	I
M_RXD1	30	Прямой вход последовательных данных резервного канала приема MIL_STD	I
M_RXDN0	35	Инверсный вход последовательных данных основного канала приема MIL_STD	I
M_RXDN1	25	Инверсный вход последовательных данных резервного канала приема MIL_STD	I
ALE ADV#	69	Выход «разрешение записи адреса» Выход «адрес действителен»	O O
BHE# WRH#	42	Выход «разрешение записи старшего байта» Выход «запись старшего байта»	O O
WR# WRL#	40	Выход «запись» Выход «запись младшего байта»	O O

Окончание таблицы 2.1

1	2	3	4
RD#	68	Выход «чтение»	O
EA#	76	Вход «внешний доступ»	I
BW	71	Вход «разрядность внешней шины»	I
READY	43	Вход «готовность»	I
INST	79	Выход «чтение команды»	O
CLKOUT	72	Выход «системный тактовый сигнал»	O
NMI	75	Вход «немаскируемое прерывание»	I
XTAL1	74	Вывод подключения кварцевого резонатора/ Вход тактового сигнала	– I
XTAL2	73	Вывод подключения кварцевого резонатора	–
VPR	38	Вход «старт с альтернативного адреса» Вход «возврат из режима пониженного потребления»	I I
ADC_REFHI	10	Верхнее опорное напряжение АЦП	–
ADC_REFLO	13	Нижнее опорное напряжение АЦП	–
∩0V	3	Общий вывод АЦП	–
∩VCC	86	Питание 3,3 В аналоговой части микросхемы	–
#VCC	11, 33, 56, 60, 78	Питание 3,3 В цифровой части микросхемы	–
#0V	12, 34, 53, 55, 57, 77	Общий вывод цифровой части микросхемы	–

Примечания

1 В графе «Тип вывода» используются обозначения: I – вход, O – выход, Z – третье состояние, 2 – режим открытого стока.

2 Выводы P1.0 – P1.7, P2.6, P2.7 являются квазидвунаправленными. Выводы NMI и RESET# микроконтроллера имеют схемы «Pull-down» и «Pull-up», соответственно. Выводы портов P3 и P4 микроконтроллера имеют схемы «Pull-up», включающиеся только в течение времени, когда активен сигнал RESET# (подробнее – см. таблицу 7.3).

Таблица 2.2 – Назначение выводов микроконтроллера 1874ВЕ7Т

Номер вывода	Обозначение вывода	Функциональное назначение	Тип вывода
83	P5.1 ACH9 ACHN1	Вход «порт 5, 1 разряд» Вход АЦП, канал 9 Инверсный вход АЦП, канал 1	I I I
7	P5.2 ACH10 ACHN2	Вход «порт 5, 2 разряд» Вход АЦП, канал 10 Инверсный вход АЦП, канал 2	I I I
85	P5.6 ACH14 ACHN6	Вход «порт 5, 6 разряд» Вход АЦП, канал 14 Инверсный вход АЦП, канал 6	I I I
9	P5.7 ACH15 ACHN7	Вход «порт 5, 7 разряд» Вход АЦП, канал 15 Инверсный вход АЦП, канал 7	I I I
15	RESET#	Вход/выход «сброс»	I/O/2

Примечание – В графе «Тип вывода» используются обозначения: I – вход, O – выход, 2 – режим открытого стока.

Таблица 2.3 – Назначение выводов микроконтроллера 1874ВЕ71Т

Номер вывода	Обозначение вывода	Функциональное назначение	Тип вывода
83	TDI	Вход данных порта JTAG	I
7	TMS	Вход режима порта JTAG	I
85	TDO	Выход данных порта JTAG	O
9	TCK	Вход тактового сигнала JTAG	I
15	RESET# TRST	Вход/выход «сброс» Вход сброса порта JTAG	I/O/2 I
Примечание – В графе «Тип вывода» используются обозначения: I – вход, O – выход, 2 – режим открытого стока.			

2.2 Электрические параметры

Основные параметры:

- номинальное значение напряжения питания микросхемы – 3,3 В;
- допустимое отклонение напряжения питания ± 10 %;
- амплитуда пульсаций напряжения питания – не более 30 мВ;
- напряжение источника опорного напряжения – от 2,6 (– 1 %) до 3,6 В (+ 1 %);
- номинальная температура рабочей среды – (25 ± 10) ºС;
- диапазон температур рабочей среды – от (минус 60 ± 3) до (85 ± 3) ºС.

Порядок подачи напряжения питания и входных сигналов на выводах микросхемы следующий:

- подать напряжения питания U_{CC1}, U_{CC2} до достижения номинального напряжения U_{CC1};
- подать напряжения входных сигналов U_{IL} и U_{IH}.

Порядок снятия напряжений обратный.

Микросхемы 1874ВЕ7Т, 1874ВЕ71Т сохраняют свои параметры в процессе и после воздействия на нее:

- климатических факторов по таблице 4 ОСТ В 11 0998–99;
- механических факторов по ОСТ В 11 0998–99;
- технологических факторов при изготовлении радиоэлектронной аппаратуры по ОСТ В 11 0998–99.

Микросхемы устойчивы к воздействию статического электричества с потенциалом не менее 2 кВ.

Полный перечень электрических параметров микросхемы при приёмке и поставке и предельно допустимые значения параметров приведены в таблицах 2.4 и 2.5 соответственно.

Таблица 2.4 – Электрические параметры при приёмке и поставке в диапазоне рабочих температур окружающей среды от минус 60 до плюс 85 ºС

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 RD#, WR#, ALE, BHE#, INST, CLKOUT, RESET#, P1.0 – P1.7, P2.0, P2.1/RXD0 (в режиме 0), P2.3/SCL, P2.4/RXD1 (в режиме 0), P2.5 – P2.7, P3.0 – P3.7, P4.0 – P4.7, PWM.1, PWM.2, HSO.3, M_TXD0, M_TXDN0, M_TXD1, M_TXDN1, TDO[1], В, U_{CC1} = 3,0 В, I_{OL} = 2,8 мА		U_{OL}	–	0,4	–60 ± 3 25 ± 10 85 ± 3
2 Выходное напряжение высокого уровня по выводам RD#, WR#, ALE, BHE#, INST, CLKOUT, P2.0, P2.5, P3.0 – P3.7, P4.0 – P4.7, PWM.1, PWM.2, HSO.3, M_TXD0, M_TXDN0, M_TXD1, M_TXDN1, TDO[1], В, U_{CC1} = 3,0 В, I_{OH1} = –3,2 мА		U_{OH1}	U_{CC1}–0,4	–	
3 Выходное напряжение высокого уровня по выводам P1.0 – P1.7, P2.6, P2.7, В, U_{CC1} = 3,0 В, I_{OH2} = –0,03 мА		U_{OH2}	U_{CC1}–0,4	–	
4 Ток утечки низкого уровня по входам EA#, BW, READY, NMI, VPR, P0.0 – P0.7, P2.1 – P2.4, P3.0 – P3.7, P4.0 – P4.7, M_RXD0, M_RXDN0, M_RXD1, M_RXDN1, P5.0, P5.1[2], P5.2[2], P5.3, P5.4, P5.5, P5.6[2], P5.7[2], TDI[1], TCK[1], TMS[1], мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В		I_{ILL}	–10	–	
5 Ток утечки высокого уровня по входам EA#, BW, READY, VPR, P0.0 – P0.7, P1.0 – P1.7, P2.1 – P2.4, P2.6, P2.7, P3.0 – P3.7, P4.0 – P4.7, M_RXD0, M_RXDN0, M_RXD1, M_RXDN1, P5.0, P5.1[2], P5.2[2], P5.3, P5.4, P5.5, P5.6[2], P5.7[2], TDI[1], TMS[1], RESET#, мкА, U_{CC1} = 3,6 В, U_{IH} = U_{CC1}		I_{ILH}	–	10	
6 Входной ток низкого уровня по выводам P1.0 – P1.7, P2.6, P2.7, мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В		I_{IL1}	–300	–	
7 Входной ток высокого уровня по выводам P1.0 – P1.7, P2.6, P2.7, мкА, U_{CC1} = 3,6 В, U_{IH} = 2,0 В		I_{IH1}	–960	–	
8 Входной ток низкого уровня по выводу RESET#, мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В		I_{IL2}	–800	–50	
9 Входной ток высокого уровня по выводу NMI, мкА, U_{CC1} = 3,6 В, U_{IH} = 2,4 В		I_{IH2}	10	200	
10 Входной ток низкого уровня по выводу XTAL1, мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В		I_{IL3}	–40	–	
11 Входной ток высокого уровня, мкА, U_{CC1} = 3,6 В, U_{IH} = U_{CC1}	по выводу XTAL1	I_{IH3}	–	40	
	по выводу TCK[1]	I_{IH4}[1]	–	400	

Окончание таблицы 2.4

1	2	3	4	5
12 Динамический ток потребления цифровой части в режиме сброса по выводам #VCC, мА, $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_{CI} = 24$ МГц	I_{OCC1}	–	160	
13 Ток потребления АЦП по выводу ∩VCC, мА, $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_S = 75$ кГц, $f_{IN} = 7{,}5$ кГц	I_{CC2}	–	20	
14 Общие гармонические искажения канала АЦП, дБ, $U_{CC1} = U_{CC2} = 3{,}3$ В, $f_S = 75$ кГц, $f_{IN} = 7{,}5$ кГц	THD	–	–50	–60 ± 3 25 ± 10 85 ± 3
15 Отношение сигнал/(шум + искажения) в канале АЦП, дБ, $U_{CC1} = U_{CC2} = 3{,}3$ В, $f_S = 75$ кГц, $f_{IN} = 7{,}5$ кГц	SINAD	50	–	
16 Функциональный контроль, $U_{CC1} = (3{,}0; 3{,}6)$ В, $f_{CI} = 24$ МГц	ФК	–	–	
Примечание – Параметры I_{ILL}, I_{ILH} при температуре минус 60 ºС не измеряются, а гарантируются нормами при температуре (25 ± 10) ºС.				

[1)] Только для ИС 1874ВЕ71Т.
[2)] Только для ИС 1874ВЕ7Т.

Таблица 2.5 – Значения предельно допустимых электрических режимов эксплуатации в диапазоне рабочих температур

Наименование параметра режима, единица измерения	Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
1 Напряжение питания цифровой части, В[1)]	U_{CC1}	3,0	3,6	–	5,0
2 Напряжение питания АЦП, В[1)]	U_{CC2}	3,0	3,6	–	5,0
3 Входное напряжение низкого уровня, В[1)]	U_{IL}	–0,5	0,8	–0,6	–
4 Входное напряжение высокого уровня на входах, кроме выводов XTAL1 и RESET#, В[1)]	U_{IH}	$0{,}2U_{CC1}+1{,}0$	U_{CC1}	–	$U_{CC1}+0{,}6$
5 Входное напряжение высокого уровня на выводе XTAL1, В[1)]	U_{IHCI}	$0{,}7U_{CC1}$	U_{CC1}	–	$U_{CC1}+0{,}6$
6 Входное напряжение высокого уровня на входе RESET#, В[1)]	U_{IHRST}	$0{,}6U_{CC1}$	U_{CC1}	–	$U_{CC1}+0{,}6$
7 Диапазон преобразования АЦП, В	U_{FSR}	0,6	3,0	0	U_{CC2}
8 Выходной ток низкого уровня, мА[1)]	I_{OL}	–	2,8	–	10
9 Выходной ток высокого уровня, мА[1)]	I_{OH1}	–3,2	–	–10	–
	I_{OH2}	–0,03	–	–0,1	–
10 Частота следования импульсов тактового сигнала, МГц	f_{CI}	0,001	24	–	–
11 Частота дискретизации АЦП, кГц	f_S	10	75	–	–
12 Частота входного сигнала АЦП, кГц	f_{IN}	0	7,5	–	–
13 Длительность фронта и спада входных сигналов, нс	t_{LH}, t_{HL}	–	5	–	–
14 Емкость нагрузки выходов, пФ	C_L	–	100	–	200

[1)] Время работы в одном из предельных режимов должно быть не более 5 с.

3 Архитектура изделия

16-разрядные микроконтроллеры ИС 1874ВЕ7Т, 1874ВЕ71Т предназначены для выполнения вычислительных команд с повышенным быстродействием и высокоточных аналогово-цифровых преобразований в условиях радиационного воздействия. Они имеют общие архитектуру и набор инструкций с другими микросхемами серии 1874, но базируются на новом ядре AMCS-96 разработки ОАО «НИИЭТ», производительность которого в среднем на 35 % выше производительности ядер других контроллеров серии 1874, реализующих архитектуру MCS-96, при одинаковой частоте, что было достигнуто благодаря введению конвейеризации процесса выборки-выполнения команд, полной оптимизации процесса выполнения инструкций, наличию аппаратных сдвигателя, умножителя и делителя, а также другим конструктивным особенностям. Реализуемая ядром микроконтроллера архитектура AMCS-96 полностью программно совместима с ядрами MCS-96 архитектуры.

Контроллеры обладают регистровым ОЗУ объемом 2 024 байт, доступным быстрым способам адресации, имеют конфигурируемую 16/8-разрядную внешнюю шину адреса/данных и позволяют программировать начальный адрес выполнения программ из внешней памяти.

Сервер периферийных транзакций PTS позволяет передавать данные в моменты простоя шины данных, что обеспечивает транзакции без остановки ЦПУ и повышает общее быстродействие системы. Для выполнения высокоточных аналого-цифровых преобразований может использоваться 12-разрядный конфигурируемый АЦП с двумя дополнительными аппаратными битами повышения точности. Подсистема высокоскоростного ввода-вывода с разрешающей способностью в один машинный цикл позволяет использовать микроконтроллер в быстродействующих системах управления. Входящие в состав микроконтроллеров контроллеры интерфейсов SPI, I2C, SpaceWire, магистральный последовательный интерфейс (ГОСТ Р 52070–2003) и последовательные порты ввода-вывода типа UART позволяют встраиваться в системы, которые обмениваются информацией по соответствующим протоколам.

Микроконтроллеры имеют гибкую систему управления энергопотреблением. Помимо режимов пониженного энергопотребления Powerdown и Idle, а также режима медленной работы Slow, поддерживается режим электрической изоляции Once. Кроме того, реализована возможность выборочного отключения неиспользуемых периферийных устройств.

Процессорное ядро

Микроконтроллеры имеют архитектуру Фон Неймана с внешней системной магистралью для обмена данными с внешней памятью и дополнительными периферийными устройствами. Система команд поддерживает широкий набор методов адресации, в т. ч. битовую адресацию.

Архитектура микроконтроллера, называемая архитектурой типа «регистр-регистр», принципиально отличается от архитектуры микроконтроллеров других серий. Такая архитектура обеспечивает достижение более высокой производительности и упрощает работу с периферией. Этим, в первую очередь, объясняются возможности эффективного решения на базе микроконтроллера достаточно сложных задач управления в реальном времени.

Главным отличительным признаком архитектуры является полностью обновленное ядро, базирующееся на регистровом файле. Большое число универсальных легкодоступных регистров исключает «узкие места», свойственные архитектуре, использующей специальные регистры-аккумуляторы, и обеспечивает быстрое переключение контекста. Все устройства микроконтроллера реализуют операции с байтами, словами, несколько операций с 32-битовыми операндами, а также команды перехода по битам.

Структурная схема ядра AMCS-96 разработки ОАО «НИИЭТ» представлена на рисунке 3.1.

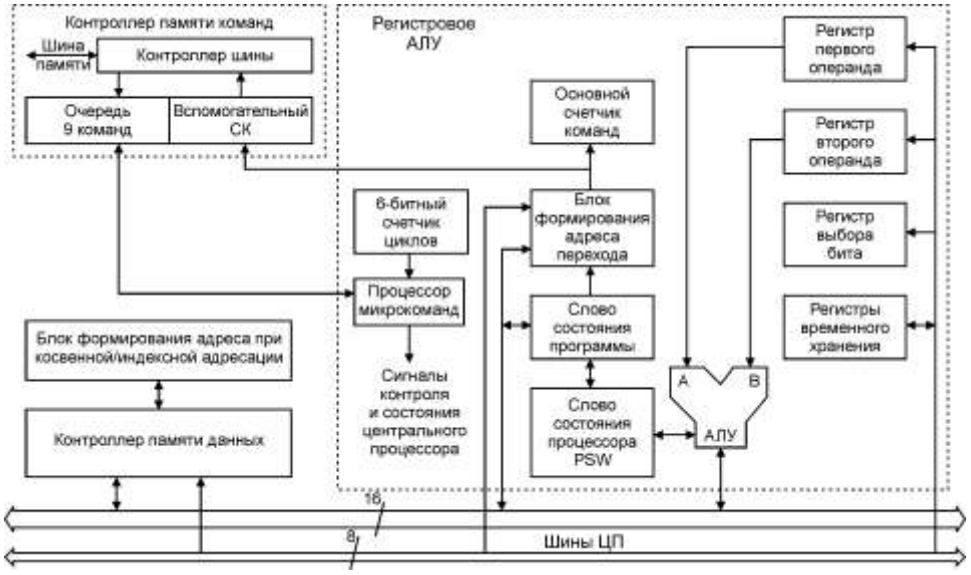


Рисунок 3.1 – Структурная схема ядра микроконтроллера

Команды поступают в блок очереди команд, который может хранить до девяти команд длиной до 7 байт каждая. Процесс выборки и выполнения команд независимый, что позволяет минимизировать время простоя как контроллера команд, так и ЦПУ. Выборка команд может осуществляться или из внутреннего ПЗУ, или через интерфейс из внешней памяти в режиме 16/8 бит. По сигналу команды попадают в процессор микрокоманд, где происходит их декодирование и выполнение. Выполнение команды состоит из последовательности выполнений микроинструкций. Нужная инструкция выбирается счетчиком циклов, обнуляющимся при старте выполнения каждой новой команды. Контроллер памяти данных выполняет выборку нужных данных из ОЗУ или внешней памяти.

Блок формирования адреса при косвенной/индексной адресации позволяет формировать адрес без использования функций АЛУ. В своем составе содержит сумматор/вычитатель. Основной счетчик команд содержит адрес команды, следующей за выполняемой. После сброса загружается начальным адресом, с которого начинается выполнение программы. По выбору значения начального адреса – 2080h, 9000h. Если переход, прерывание, вызов подпрограммы или возврат из подпрограммы изменяет адресную последовательность, то АЛУ загружает соответствующий адрес в PC и очищает очередь. ЦПУ выполняет вычисления в АЛУ. Слова вводятся в АЛУ через входы A и B. АЛУ выполняет все логические и арифметические операции за один (для деления – два) машинных цикла. В своем составе содержит встроенные аппаратные сумматоры/вычитатели, сдвигатели, умножитель, делитель и др.

Регистр слова состояния программы является 16-разрядным и состоит из двух байт. Старший байт – это непосредственно регистр PSW, младший – регистр INT_MASK. Описание всех регистров микроконтроллера приводится в приложении А. Слово состояния содержит бит I, который разрешает или запрещает обработку всех маскируемых прерываний, бит PSE, который разрешает или запрещает программный сервер обмена PTS, и шесть двоичных флагов, которые отражают состояние программы пользователя.

Слово состояния программы PSW прямо не доступно. Для доступа следует записать его значение в стек (команда PUSHF), а затем считать это значение в регистр (команда POP).

Команды EI и DI устанавливают и сбрасывают бит I, команды EPTS и DPTS устанавливают и сбрасывают PSE, различные команды тестируют, устанавливают и сбрасывают двоичные флаги. Команды PUSHF и PUSHA сохраняют значение PSW в системном стеке, команды POPF и POPA – восстанавливают его.

Система команд

Микроконтроллер содержит полный набор команд, включающий операции с битами, байтами, словами, двойными словами (беззнаковые 32 бит), длинные операции (32 бит со знаком), работу с флагами, а также переходы и вызовы подпрограмм. Все стандартные логические и арифметические команды работают как с байтами, так и со словами. Команды перехода по установке бита (JBS) и очистке бита (JBC) могут работать с регистрами SFR и другими байтами регистрового файла. Эти быстрые битовые операции позволяют ускорить функции ввода-вывода.

Операции над байтами и словами составляют основу системы команд. Ассемблер ASM-96 использует суффикс «B» в мнемонике для команд операций над байтами, иначе мнемоника относится к командам операций над словами.

Длинные и двухсловные операции включают операции сдвигов, нормализацию, умножения и деления. В операции «Деление» за два машинных цикла 32-битное число делится на 16-битное число, что порождает 16-битное частное и 16-битный остаток. В операции «Умножение» за один машинный цикл 16-битное число умножается на 16-битное число с образованием 32-битного результата. Обе операции могут выполняться с числами со знаком или без знака. Команды нормализации и соответствующий флаг обеспечивают аппаратную поддержку пакета программ для выполнения операций над числами с плавающей запятой (FPAL-96). Подробное описание системы команд микроконтроллера приводится в приложении Б.

Прерывания

Основной функцией микроконтроллеров является обеспечение управления устройствами в реальном времени. Схема контроллера прерываний позволяет событиям реального времени управлять выполнением программы. Когда событие вызывает прерывание, ядро обслуживает это прерывание перед выполнением следующей команды. Встроенная периферия, внешний сигнал или команда могут выставить запрос на прерывание. В простейшем случае микроконтроллер получает запрос прерывания, осуществляет обслуживание и возвращается к прерванной программе.

В микроконтроллерах реализованы 44 источника прерываний, для которых выделены 17 векторов прерываний, а также два дополнительных вектора – пошаговое выполнение программы (SWTRAP) и неподдерживаемый код операции (UNIOPCODE), которые могут использоваться в системе отладки и платах-прототипах. Когда контроллер прерываний определит одно из семнадцати прерываний, он устанавливает соответствующий бит в один из двух регистров ожидания прерываний INT_PEND и INT_PEND1. Отдельные прерывания разрешаются, либо запрещаются установкой или очисткой соответствующих битов в регистрах масок прерываний INT_MASK и INT_MASK1.

Когда контроллер прерываний производит обработку прерывания, он делает вызов программы обслуживания прерывания, на которую указывает соответствующий вектор прерывания. Далее контроллер прерываний очищает соответствующий бит ожидания.

Имеются две возможности обслуживания прерываний: программное – через контроллер прерываний – и микропрограммное – через сервер периферийных транзакций PTS. Можно выбрать вариант обслуживания прерываний для каждого из маскируемых прерываний. Немаскируемые прерывания (NMI, программное прерывание, прерывание по неподдерживаемому коду) всегда обрабатываются контроллером прерываний.

4 Типы данных и адресация

В данном разделе рассматриваются типы операндов и способы адресации, обеспечиваемые архитектурой микроконтроллера, а также приводятся краткие рекомендации по программированию.

4.1 Типы операндов

Система команд поддерживает множество типов данных, которые обычно используются в управляющих алгоритмах.

Типы переменных операндов указаны заглавными буквами, например, BYTE – 8-битная беззнаковая переменная в инструкции.

В таблице 4.1 даётся общий обзор этих типов операндов.

Таблица 4.1 – Типы операндов

Тип операнда	Наличие знака и число бит		Возможные значения	Адресация
Бит (BIT)	без-знаковые	1	1 или 0	Как к компоненту байта
Байт (BYTE)		8	от 0 до (2^8–1) (от 0 до 255)	К байту
Слово (WORD)		16	от 0 до (2^{16}–1) (от 0 до 65 535)	К младшему байту (адрес должен быть кратным двум)
Двойное слово (DOUBLE WORD)		32	от 0 до (2^{32}–1) (от 0 до 4 294 967 295)	К младшему байту младшего слова в регистровом файле (адрес должен быть кратным четырем)
Короткое целое (SHORT INTEGER)	знаковые	8	от $(-2)^7$ до (2^7–1) (от –128 до 127)	К байту
Целое (INTEGER)		16	от $(-2)^{15}$ до (2^{15}–1) (от –32 768 до 32 767)	К младшему байту (адрес должен быть кратным двум)
Длинное целое (LONG INTEGER)		32	от $(-2)^{31}$ до (2^{31}–1) (от –2 147 483 648 до 2 147 483 647)	К младшему байту младшего слова в регистровом файле (адрес должен быть кратным четырем)
Примечание – 32-битные переменные поддерживаются только как операнды в инструкциях сдвига, как делимое при инструкции деления 32 на 16 и как результат действия умножения 16 на 16. 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 бит (BIT)

BIT – одноразрядная переменная, которая может иметь булевы значения «TRUE» и «FALSE». Архитектура требует, чтобы к биту обращались как к компонентам байта или слова, так как прямая адресация бита не поддерживается.

Операнд байт (BYTE)

BYTE – 8-битная переменная без знака, которая может иметь значения от 0 до 255 (2^8–1). Арифметические операторы и операторы сравнения могут быть применены к операндам байтам, но результат должен интерпретироваться по модулю 256 арифметически. Логические действия с байтами осуществляются побитно. Биты в пределах байта нумеруются от 0 до 7, бит 0 – младший бит. Нет никаких ограничений выравнивания для байт, так что они могут быть помещены по любому адресу памяти.

Операнд слово (WORD)

WORD – 16-битная беззнаковая переменная, состоящая из двух байт (старшего и младшего), которая может иметь значения от 0 до 65535 (2^{16}–1). Арифметические операции и операции сравнения могут быть применены к операндам словам, но результат должен интерпретироваться по модулю 65536 арифметически. Логические действия со словами осуществляются побитно. Биты в пределах слова нумеруются от 0 до 15, бит 0 – младший бит.

Слова должны быть выровнены по границе четного байта адреса пространства. Младший байт слова находится по четному адресу, а старший байт находится в следующем (нечетном) адресе. Адрес слова – это адрес его младшего байта. Действия над словами, расположенными по нечетным адресам, не гарантируются.

Операнд двойное слово (DOUBLE WORD)

DOUBLE WORD – 32-битная переменная без знака, состоящая из двух слов (старшего и младшего), которая может принимать значения от 0 до 4 294 967 295 (2^{32}–1). Архитектура непосредственно поддерживает операнды двойные слова только как операнды в командах сдвига, делимого при делении 32/16 и произведения при умножении 16×16. Для этих действий переменная двойное слово должна находиться в младшем файле регистров и должна быть выровнена по адресу, который является кратным четырем. Адрес двойного слова – это адрес его самого младшего байта (четный адрес). Младшее слово двойного слова находится всегда в младшем адресе, даже когда данные находятся в стеке. Это означает, что старшее слово должно быть выдвинуто в стек первым.

Действия над двойным словом, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя словами. Например, следующие последовательности инструкций с 16 битами исполняют 32-битное сложение и 32-битное вычитание соответственно.

```
ADD   REG1, REG3      ; (2-операндное сложение)
ADDC  REG2, REG4
SUB   REG1, REG3      ; (2-операндное вычитание)
SUBC  REG2, REG4
```

Операнд короткое целое (SHORT INTEGER)

SHORT INTEGER – 8-битная переменная со знаком, которая может принимать значения от –128 (–2^7) до 127 (2^7–1). Арифметические действия, результат которых оказывается вне допустимого диапазона, устанавливают флаги переполнения в регистре PSW. Числовой результат тот же самый, что и результат эквивалентного действия над байтами. Нет никаких ограничений выравнивания для коротких целых, так что они могут быть помещены по любому адресу памяти.

Операнд целое (INTEGER)

INTEGER – 16-битная переменная со знаком, состоящая из двух байт (старшего и младшего), которая может принимать значения от –32 768 (-2^{15}) до 32 767 ($2^{15}-1$). Арифметические действия, результат которых оказывается вне допустимого диапазона, устанавливают флаги переполнения в регистре PSW. Числовой результат тот же самый, что и результат эквивалентного действия над словами.

Целое должно быть выровнено по границе четного байта в адресном пространстве. Младший байт целого находится по четному адресу, а старший байт находится в следующем (нечетном) адресе. Адрес целого – адрес его младшего байта. Действия над целыми, расположенными по нечетным адресам, не гарантируются.

Операнд длинное целое (LONG INTEGER)

LONG INTEGER – 32-битная переменная со знаком, состоящая из двух слов (старшего и младшего), которая может принимать значения от –2 147 483 648 (-2^{31}) до 2 147 483 647 ($2^{31}-1$). Архитектура непосредственно поддерживает операнды длинные слова только как операнды в командах сдвига, как делимое при делении 32/16 и как произведение при умножении 16×16. Для этих действий переменная длинное слово должна быть в младшем файле регистров и должна быть выровнена по адресу, который является кратным четырем. Адрес длинного целого – это адрес его самого младшего байта (четный адрес).

Действия над длинными целыми, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя целыми. Например, следующие последовательности инструкций с 16 битами исполняют 32-битное сложение и 32-битное вычитание соответственно.

```
ADD    REG1, REG3        ; (2-операндное сложение)
ADDC   REG2, REG4
SUB    REG1, REG3        ; (2-операндное вычитание)
SUBC   REG2, REG4
```

4.2 Способы адресации

Система команд микроконтроллера допускает следующие способы адресации: прямая регистровая, непосредственная, косвенная, косвенная с автоинкрементом, короткая индексная, длинная индексная, индексная с использованием нулевого регистра и с использованием указателя стека. Каждая команда использует, по крайней мере, один из способов адресации. Имеются расширенные версии команд прерываемой и непрерываемой передачи блока. Команды ассемблера состоят из мнемоники, за которой следуют адрес или данные. В представленных ниже примерах используются обозначения:

AX, BX, CX	–	16-разрядные регистры;
DX	–	8-разрядный регистр;
AH, AL	–	старший и младший байты регистра AX;
BH, BL	–	старший и младший байты регистра BX;
SP	–	регистр указателя стека;
MEM_WORD(reg)	–	содержимое ячейки памяти, адрес которой указан в регистре reg, в качестве которого может использоваться любой 8/16-разрядный регистр файла регистров;

Прямая регистровая адресация

Наиболее простой и быстрый способ адресации. Предполагает размещение операнда в файле регистров. Операнд имеет короткий байтовый адрес. С использованием оконной адресации памяти прямая регистровая адресация может быть распространена на операнды с адресом, превышающим FFh. Для удобства целесообразно работать с символическими именами переменных, предварительно определив их.

Пример:

```
ADD CX, AX, BX          ; CX ← AX + BX
INCB DX                 ; DX ← DX + 1
```

Непосредственная адресация

Непосредственная адресация позволяет брать значение операнда непосредственно из поля команды (указывается после символа «#»). Для операций с переменными байтами и короткими целыми это будет 8-разрядное поле. Для операций с переменными типа слово и целое – 16-разрядное поле. Команда может содержать только один непосредственный операнд; для других операндов необходимо использовать прямую регистровую адресацию.

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

Пример:

```
ADD AX, #340            ; AX ← AX + 340
PUSH #1234h             ; SP ← SP – 2, SP ← 1234H
DIVB AX, #10            ; AL ← AX/10, AH ← AX MOD 10
```

Косвенная адресация

Косвенная адресация осуществляет доступ к операнду, адрес которого размещен в переменной типа слово регистрового файла. Вычисляемый адрес должен соответствовать правилам выравнивания. Следует заметить, что косвенная адресация позволяет обращаться к операнду в любом месте адресного пространства, включая файл регистров. 8-битное поле 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:

```
LD AX, [AX]             ; AX ← MEM_WORD(AX)
ADDB AL, BL, [CX]       ; AL ← BL + MEM_BYTE(CX) (ADDB_3op)
POP [AX]                ; MEM_WORD(AX) ← MEM_WORD(SP), SP ← SP + 2
```

Косвенная адресация с автоинкрементом

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

Пример:

```
LD AX, [BX]+            ; AX ← MEM_WORD(BX), BX ← BX + 2
ADDB AL, BL, [CX]+      ; AL ← BL + MEM_BYTE(CX)
                        ; CX ← CX + 1 (ADDB_3op)
PUSH [AX]+              ; SP ← SP – 2
                        ; MEM_WORD(SP) ← MEM_WORD(AX)
                        ; AX ← AX + 2
```

Короткая индексная адресация

При короткоиндексной адресации адрес одного из операндов вычисляется с помощью двух 8-битных полей. Одно 8-битное поле выбирает в файле регистров переменную типа слово, содержащую адрес. Второе 8-битное поле в команде имеет знаковое расширение и суммируется с переменной типа слово для формирования исполнительного адреса операнда. Исполнительный адрес может находиться на 128 байт ранее от текущего адреса или на 127 байт после него. Команда может содержать только один такой операнд, другие операнды задаются с помощью прямой регистровой адресации.

Пример:

```
LD AX, 12[BX]           ; AX ← MEM_WORD(BX+12)
MULB AX, BL, 3[CX]      ; AX ← BL * MEM_BYTE(CX+3), (MULB_3op)
```

Длинная индексная адресация

Длинноиндексная адресация похожа на короткоиндексную адресацию, за исключением того, что 16-битное поле берётся из команды и добавляется к переменной типа слово для формирования адреса операнда. Такая адресация может применяться только для одного операнда в команде, для других операндов необходимо использовать прямую регистровую адресацию.

Пример:

```
AND AX, BX, TABLE[CX]      ; AX ← BX AND MEM_WORD(TABLE+CX)
                           ; (AND_3op)
ST AX, TABLE[BX]           ; MEM_WORD(TABLE_BX) ← AX
ADDB AL, BL, LOOKUP[CX]    ; AL ← BL + MEM_BYTE(LOOKUP+CX)
                           ; (ADDB_3op)
```

Адресация с использованием регистра нуля

Кроме источника фиксированного нуля для расчетов и сравнения, 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) (ADD_2op)
POP 5678[ZERO_REG]         ; MEM_WORD(5678) ← MEM_WORD(SP)
                           ; SP ← SP + 2
```

Адресация с использованием указателя стека

Байты с адресами 18h и 19h файла регистров содержат указатель стека SP, к которому адресуются по адресу 18h. Кроме обычных операций указатель стека SP может также использоваться как переменная типа слово в 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
```

5 ОЗУ и внешняя память

Микроконтроллер имеет внутреннее ОЗУ объемом 2 Кбайта, память сообщений контроллера интерфейса ГОСТ Р 52070–2003 объемом 4 Кбайта, которая может использоваться как дополнительное внутреннее ОЗУ, и 64 Кбайт доступного адресного пространства внешней памяти.

5.1 ОЗУ и память сообщений

ОЗУ микроконтроллера, или так называемый файл регистров (или регистровый файл), состоит из двух равных по объему областей – регистрового и расширенного ОЗУ.

В свою очередь файл регистров состоит из младшего файла и старшего файла.

Структура ОЗУ показана на рисунке 5.1, дополнительная информация – в таблице 5.1.

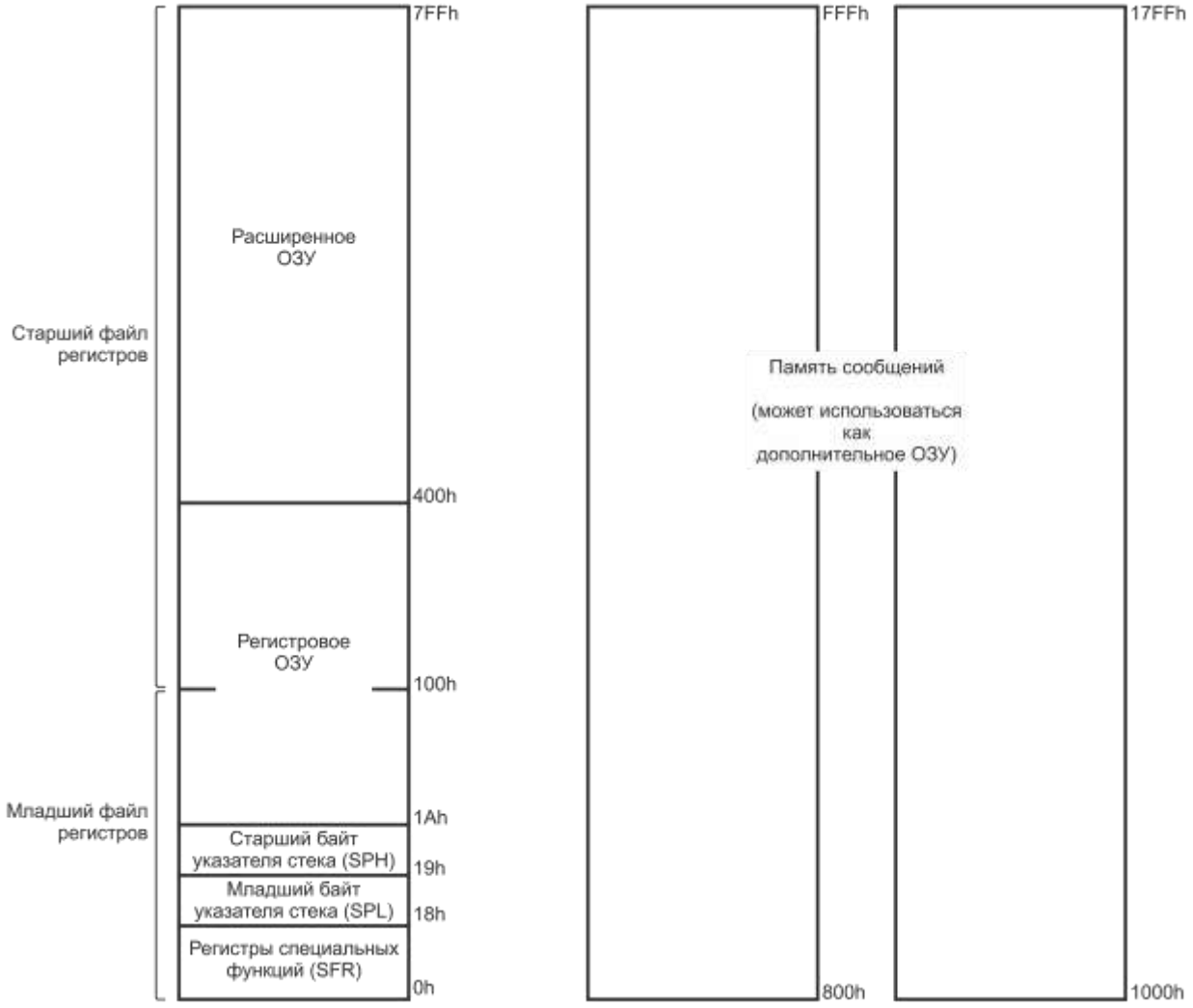


Рисунок 5.1 – Карта адресов ОЗУ

Младший файл регистров ОЗУ

Занимает адресное пространство с 00h по FFh.

С 000h по 017h располагаются 24 регистра специальных функций (SFR). Доступ к регистрам осуществляется посредством трех горизонтальных окон. Через регистры SFR РАЛУ осуществляет прямое управление всеми периферийными модулями, кроме портов 3 и 4.

При использовании SFR в качестве базового или индексного регистра при косвенной или индексной адресациях следует помнить, что содержимое SFR непредсказуемо.

Внешние события могут менять содержимое SFR, и некоторые SFR очищаются при считывании. Функции многих SFR различны, в зависимости от того, считывается из них или записывается в них информация. По этой причине никогда не следует использовать SFR в качестве операнда в команде чтение-модификация-запись.

Не следует использовать зарезервированные биты и байты области SFR, в противном случае, в них надо записывать нули или оставлять в исходном состоянии. Значения, возвращаемые при считывании зарезервированных ячеек, не определены.

Регистр указателя стека занимает ячейки с адресами 018h (младший байт) и 019h (старший байт). SP содержит адрес вершины стека. SP должен указывать на слово с четным адресом, который на два больше, чем желаемый стартовый адрес. Перед тем, как ЦПУ выполнит вызов подпрограммы или программы обслуживания прерывания, оно дважды декрементирует содержимое SP и копирует (командой PUSH) адрес следующей команды из программного счётчика в стек. Затем загружает адрес подпрограммы или программы обслуживания прерываний в программный счётчик. После завершения выполнения подпрограммы или программы обслуживания прерывания, ЦПУ возвращается из подпрограммы (командой RET) и загружает (командой POP) содержимое вершины стека (т. е. адрес возврата) в программный счётчик.

Подпрограммы могут быть вложенными. Это значит, что каждая подпрограмма может вызывать другую подпрограмму. ЦПУ засылает содержимое программного счётчика в стек каждый раз при выполнении вызова подпрограммы. Стек растёт вниз по мере добавления содержимого. Глубину стека ограничивает только величина доступной памяти. Каждый раз при возврате из подпрограммы ЦПУ выгружает адрес из вершины стека, и потом вершина стека перемещается на следующий адрес возврата. Когда операции со стеком не проводятся, ячейки SP могут использоваться в качестве регистров общего назначения.

При инициализации стека следует помнить:

- значение адреса, загружаемое в стек, должно быть на две единицы больше значения желаемого адреса;
- стек растёт вниз, что требует достаточного количества ячеек для максимального числа адресов, заносимых в стек;
- стек может находиться в файле регистров, либо во внешней памяти.

С 01Ah по 0FFh располагаются 230 регистров общего назначения регистрового ОЗУ. РАЛУ может обращаться к этой памяти, используя прямую регистровую адресацию.

Таблица 5.1 – Адреса областей ОЗУ

Название области		Адресный диапазон	Тип адресации
Младший файл регистров	Регистры специальных функций (SFR)	000h – 017h	Прямая, регистровая, косвенная, индексная
	Регистр указателя стека SP	018h – 019h	
	Регистровое ОЗУ (регистры общего назначения)	01Ah – 0FFh	
Старший файл регистров (регистры общего назначения)	Регистровое ОЗУ	100h – 7FFh	Косвенная, индексная. При использовании вертикальных окон также еще прямая регистровая

Старший файл регистров ОЗУ

Занимает адресное пространство с 100h по 7FFh и содержит ячейки регистров общего назначения. Как правило, РАЛУ обращается к этой области памяти, используя косвенную или индексную адресацию. Тем не менее, доступна и прямая регистровая адресация, которая реализуется посредством создания вертикальных окон. Создание вертикальных окон обеспечивает быстрое переключение на задачи по прерываниям и ускоряет выполнение программ.

Память сообщений

Занимает адресное пространство с 800h по 17FFh и содержит ячейки регистров общего назначения, которые используются контроллером интерфейса ГОСТ Р 52070–2003 во время его работы. Между тем, независимо от активности контроллера интерфейса, память сообщений всегда доступна для обращения (посредством косвенной адресации). Это следует учитывать при одновременном использовании памяти сообщений по прямому назначению и как дополнительной оперативной памяти.

5.2 Внешняя память

Область выше ОЗУ всегда назначается на внешнюю память. Обращение к ней осуществляется через шину внешних адресов/данных.

Значение вывода внешнего доступа EA# во время переднего фронта сигнала RESET# определяет доступность внешней памяти. Для доступа к внешней памяти, на выводе EA# должен удерживаться низкий уровень сигнала. Это значение запоминается и не может быть изменено до следующего сброса устройства.

Примечание – Если в момент сброса на выводе EA# будет уровень логической единицы, то доступ к внешней памяти будет запрещен. Так как микроконтроллер не имеет внутренней памяти программ, поведение микроконтроллера в этом режиме никак не регламентируется.

Структура внешней памяти представлена на рисунке 5.2.

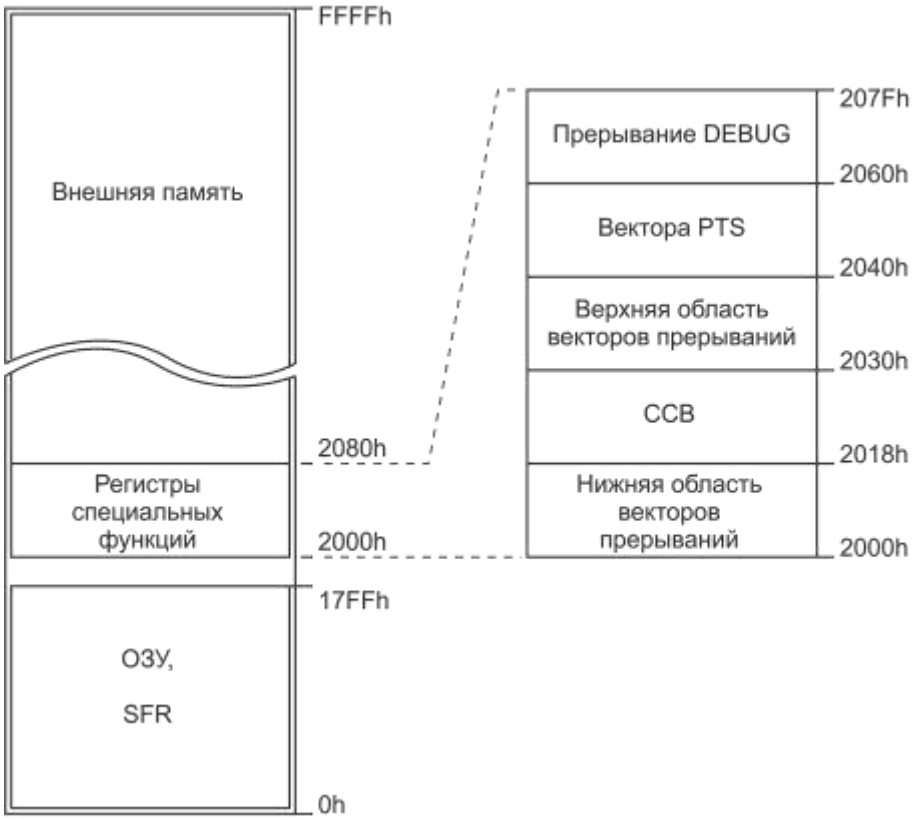


Рисунок 5.2 – Карта адресов внешней памяти

Стоит уделить внимание области памяти под названием «Память специального назначения», которая расположена в адресном пространстве с 2000h по 207Fh и содержит зарезервированные ячейки памяти и векторы прерываний. Назначение областей и соответствующие адресные диапазоны указаны в таблице 5.2.

Таблица 5.2 – Назначение ячеек памяти специального назначения

Назначение	Диапазон адресов
Младшие векторы прерываний	2000h – 2013h
Байт конфигурации кристалла CCB	2018h
Старшие векторы прерываний	2030h – 203Fh
Векторы прерываний PTS	2040h – 205Dh
Вектор прерывания DEBUG	2060h
Векторы прерываний общего назначения	2062h – 207Fh, 205Eh – 205Fh, 2019h – 202Fh, 2014h – 2017h

Регистр конфигурации кристалла CCR

Первый байт, выбираемый из внешней памяти (EA# = 0) из ячейки с адресом 2018h, после сброса микроконтроллера является байтом конфигурации кристалла CCB. Этот байт загружается в регистр CCR, который является регистром конфигурации кристалла. Однажды загруженный после сброса регистр CCR не может быть изменен до следующего сброса микроконтроллера.

Регистр CCR управляет режимом READY работы внешних устройств, разрядностью и режимом работы внешней шины управления, а также контролирует возможность перехода микроконтроллера в режим пониженного энергопотребления Powerdown.

Если на выводе READY удерживается низкий уровень сигнала во время загрузки регистра CCR, контроллер шины автоматически выставляет максимум три состояния ожидания в шинном цикле, что позволяет производить выборку байта CCB из медленной внешней памяти.

После сброса микроконтроллера выполнение программы начинается с адреса, загружаемого в счетчик команд. По умолчанию значение адреса – 2080h.

Кроме этого, реализована возможность старта программы с адреса 9000h. Для этого во время сброса микроконтроллера следует удерживать на выводе VPR низкий уровень сигнала.

5.3 Горизонтальные окна адресации

В микроконтроллере применяется технология горизонтальных окон для обращения к регистрам специального назначения. Используются три окна, каждое из которых позволяет «видеть» 24 байта блока памяти с адресами от 00h до 17h. Это горизонтальное окно 0, горизонтальное окно 1 и горизонтальное окно 15 (см. рисунок 5.3).

Рисунок 5.3 – Горизонтальные окна

Для управления переключением окон используется регистр WSR. Этот регистр «виден» в каждом окне по одному адресу (14h) и всегда доступен как для записи, так и для чтения. Доступ к остальным регистрам SFR осуществляется иначе. Так, например, регистр сторожевого таймера WATCHDOG, расположенный по адресу 0Ah, доступен для записи только при включенном нулевом горизонтальном окне, а для чтения – при включенном 15-м.

В таблице 5.3 указано, какие регистры SFR доступны для записи/чтения в зависимости от активного горизонтального окна. Если регистр 16-разрядный и занимает две ячейки в одном окне, то младший байт регистра всегда расположен по четному адресу, а старший – по нечетному. Серым цветом выделены регистры, всегда доступные для записи и чтения, вне зависимости от того, какое окно активно.

Таблица 5.3 – Горизонтальные окна адресации

<table>
<tr><td rowspan="3">Адрес</td><td colspan="5">Активное горизонтальное окно</td><td rowspan="3">Адрес</td></tr>
<tr><td colspan="2">0</td><td colspan="2">15</td><td>1</td></tr>
<tr><td>Запись</td><td>Чтение</td><td>Запись</td><td>Чтение</td><td>Запись/Чтение</td></tr>
<tr><td>17h</td><td>PWM0_
CONTROL</td><td colspan="2">IOS2</td><td>PWM0_
CONTROL</td><td>PWM2_
CONTROL</td><td>17h</td></tr>
<tr><td>16h</td><td>IOC1</td><td colspan="2">IOS1</td><td>IOC1</td><td>PWM1_
CONTROL</td><td>16h</td></tr>
<tr><td>15h</td><td>IOC0</td><td colspan="2">IOS0</td><td>IOC0</td><td></td><td>15h</td></tr>
<tr><td>14h</td><td colspan="5">WSR</td><td>14h</td></tr>
<tr><td>13h</td><td colspan="5">INT_MASK1</td><td>13h</td></tr>
<tr><td>12h</td><td colspan="5">INT_PEND1</td><td>12h</td></tr>
<tr><td>11h</td><td>SP_CON0</td><td colspan="2">SP_STAT0</td><td>SP_CON0</td><td rowspan="3">–</td><td>11h</td></tr>
<tr><td>10h</td><td colspan="2">IOPORT2</td><td rowspan="3" colspan="2">–</td><td>10h</td></tr>
<tr><td>0Fh</td><td colspan="2">IOPORT1</td><td>0Fh</td></tr>
<tr><td>0Eh</td><td>BAUD_RATE0</td><td>IOPORT0</td><td>0Eh</td></tr>
<tr><td>0Dh</td><td rowspan="2" colspan="2">TIMER2</td><td rowspan="2" colspan="2">T2CAPTURE</td><td></td><td>0Dh</td></tr>
<tr><td>0Ch</td><td>IOC3</td><td>0Ch</td></tr>
<tr><td>0Bh</td><td>IOC2</td><td colspan="2" rowspan="2">TIMER1</td><td>IOC2</td><td rowspan="2">–</td><td>0Bh</td></tr>
<tr><td>0Ah</td><td>WATCHDOG</td><td>WATCHDOG</td><td>0Ah</td></tr>
<tr><td>09h</td><td colspan="5">INT_PEND</td><td>09h</td></tr>
<tr><td>08h</td><td colspan="5">INT_MASK</td><td>08h</td></tr>
<tr><td>07h</td><td>SBUF_TX0</td><td colspan="2">SBUF_RX0</td><td>SBUF_TX0</td><td rowspan="2">PTSSRV</td><td>07h</td></tr>
<tr><td>06h</td><td>HSO_
COMMAND</td><td colspan="2">HSI_STATUS</td><td>HSO_
COMMAND</td><td>06h</td></tr>
<tr><td>05h</td><td rowspan="2">HSO_TIME</td><td rowspan="2" colspan="2">HSI_TIME</td><td rowspan="2">HSO_TIME</td><td rowspan="2">PTSSEL</td><td>05h</td></tr>
<tr><td>04h</td><td>04h</td></tr>
<tr><td>03h</td><td>HSI_MODE</td><td colspan="2">–</td><td>HSI_MODE</td><td>–</td><td>03h</td></tr>
<tr><td>02h</td><td>–</td><td colspan="2">–</td><td>–</td><td>–</td><td>02h</td></tr>
<tr><td>01h</td><td colspan="5" rowspan="2">ZERO_REG</td><td>01h</td></tr>
<tr><td>00h</td><td>00h</td></tr>
</table>

5.4 Вертикальные окна

Технология вертикальных окон заключается в том, чтобы создать в области младшего файла регистров «окно» – область, на которую можно отобразить какую-либо другую область памяти (другое «окно»), в том числе с 16-разрядными адресами, и иметь возможность использовать прямую регистровую адресацию для обращения к регистрам, к которым стандартно можно обратиться только косвенно или посредством индексной адресации. «Окно» в области младшего файла регистров – это базовое вертикальное окно.

Всю область ОЗУ можно разделить на 64 окна объемом по 32 байта, 32 окна объемом по 64 байта или 16 окон объемом по 128 байт (см. рисунок 5.4). Нумерация окон начинается с области, начальный адрес которой 0000h и используется для указания окна, которое будет «просматриваться» посредством базового вертикального окна. Начальный адрес базового окна зависит от количества окон. Так при 64 окнах базовое окно будет располагаться в области адресов с 00E0h по 00FFh, при 32 – с 00C0h по 00FFh, при 16 – с 0080h по 00FFh.

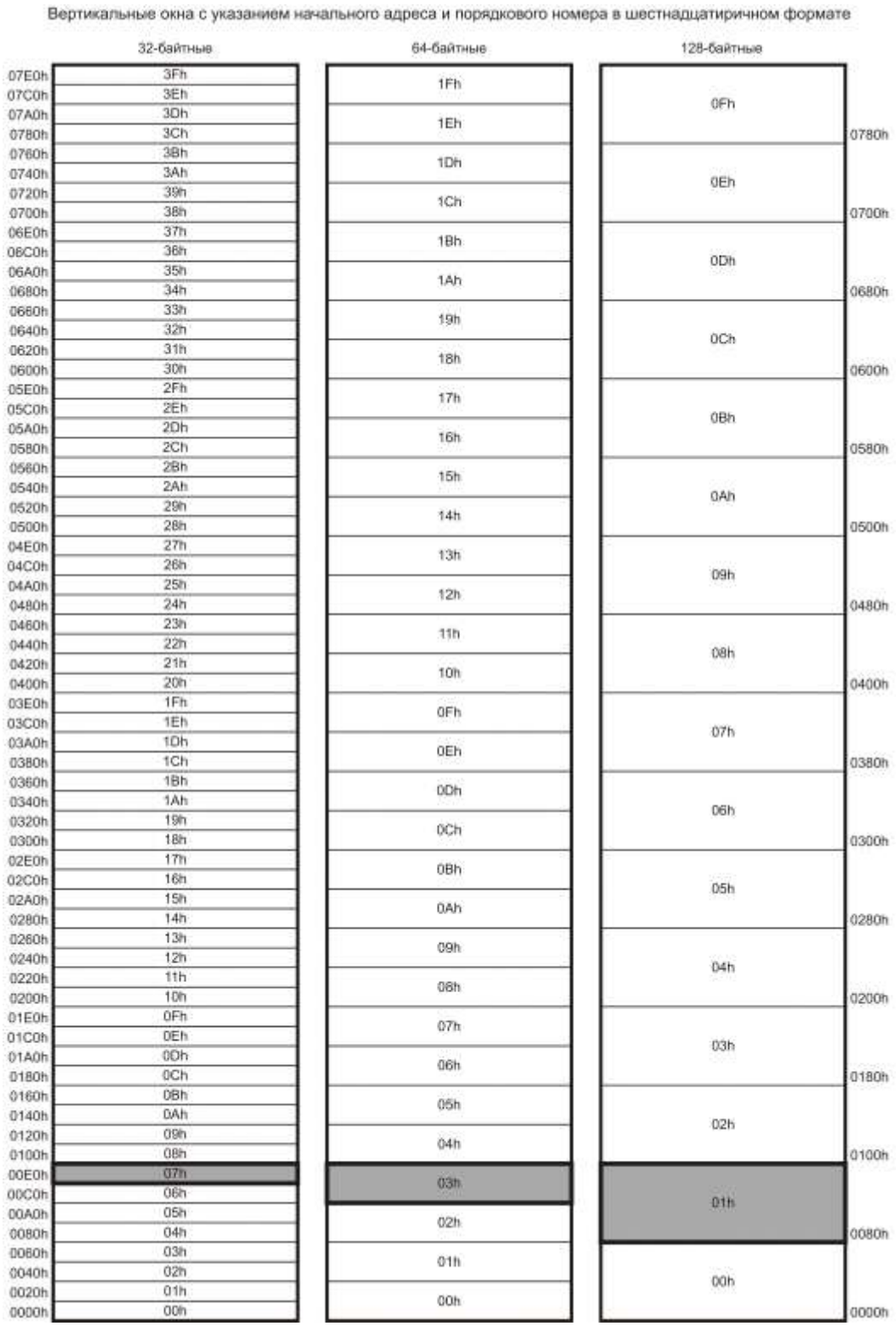


Рисунок 5.4 – Три варианта разделения ОЗУ на окна

Для управления включением/переключением вертикальных окон используется регистр WSR. Так, для включения режима 32-байтных вертикальных окон старшие два бита регистра WSR должны быть в состоянии 01b, при этом оставшиеся шесть бит задают порядковый номер окна, которое будет спроецировано на область базового вертикального окна. Для примера выбрано окно с номером 2Eh, расположенное в области адресов с 05C0h по 05DFh, которое проецируется в базовое окно. Теперь для обращения к ячейке памяти, расположенной по адресу 05C0h, достаточно указывать адрес 00E0h (см. рисунок 5.5).

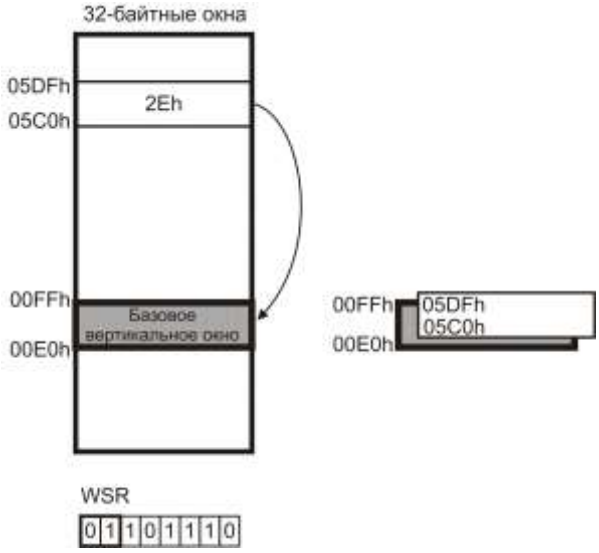


Рисунок 5.5 – Выбор вертикального окна при работе с 32-байтными окнами

Для включения режима 128-байтных вертикальных окон старшие четыре бита регистра WSR должны быть в состоянии 0001b, при этом оставшиеся четыре бита задают порядковый номер окна, которое будет спроецировано на область базового вертикального окна. Для примера выбрано окно с номером 0Dh, расположенное в области адресов с 0680h по 06FFh, которое проецируется в базовое окно. Теперь для обращения к ячейке памяти, расположенной по адресу 0680h достаточно указывать адрес 0080h (см. рисунок 5.6).

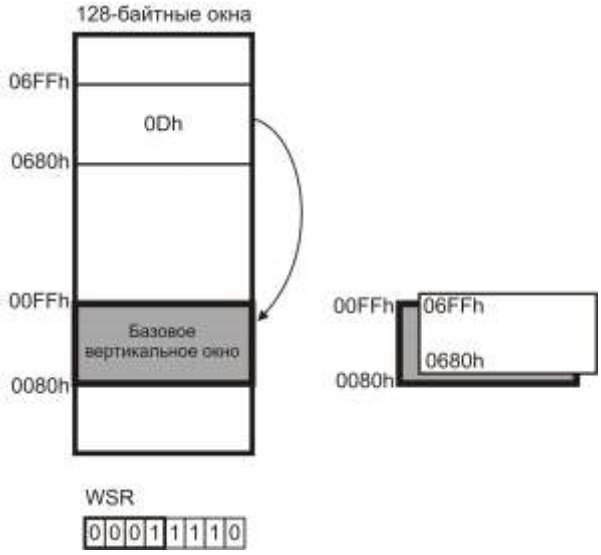


Рисунок 5.6 – Выбор вертикального окна при работе с 128-байтными окнами

Важно помнить:

- при работе с вертикальными окнами (в старшем нибле регистра WSR находится значение, отличное от 0000b) механизм горизонтальных окон не работает. По умолчанию активно горизонтальное окно 0;

- при работе с горизонтальными окнами (в старшем нибле регистра WSR находится значение 0000b) механизм вертикальных окон не работает.

Команда PUSHA автоматически сохраняет содержимое WSR в стеке, а команда POPA восстанавливает прежнее значение WSR из стека. В процедуре обслуживания прерывания после команды PUSHA можно выбрать необходимое вертикальное окно и быстро записать в него результаты расчетов или, наоборот, считать значения переменных.

Восстановление прежнего значения WSR произойдет автоматически после команды POPA в конце процедуры обслуживания прерывания (перед RET).

Ниже приведен фрагмент программы, иллюстрирующий один из возможных механизмов определения переменной в верхнем регистровом файле и способы доступа к этой переменной посредством прямой регистровой и индексной адресации. В режиме вертикальных окон выбирается окно с номером 07h, которому принадлежит ячейка памяти, расположенная по адресу 0382h.

```
        VAR1 EQU   0382H :WORD ; 16-разрядный адрес ячейки памяти

        PUSHA                  ; загрузка содержимого WSR в стек
        LDB   WSR, #17H        ; режим 128-байтных окон, выбрано окно с номером 07h

;Команда с обычной индексной адресацией
        ADD AX, VAR1[0]        ; AX ← AX + VAR1 + 0

;Те же действия, но с использованием окна
        ADD AX, 82H            ; AX ← AX + VAR1

;Команды с обычной индексной адресацией (после сложения переменных регистров AX и
;VAR1 результат нужно сохранить в регистре VAR1)

        ADD AX, VAR1[0]        ; AX ← AX + VAR1 + 0
        ST AX, VAR1[0]         ; VAR1 ← AX

;Те же действия, но с использованием окна
        ADD 82H, AX            ;VAR1 ← VAR1 + AX

        POPA                   ; перегрузка первоначальной информации из стека в WSR
```

6 Прерывания

Основной функцией микроконтроллера является обеспечение управления устройствами в режиме реального времени. В состав микроконтроллера входят два блока обработки прерываний – контроллер прерываний, позволяющий осуществлять программное обслуживание прерываний, и сервер периферийных транзакций (далее PTS), являющийся аппаратным обработчиком прерываний без задействования ЦПУ.

В микроконтроллере выделены 17 векторов прерываний для 44 источников прерывания. Встроенная периферия, внешний сигнал или команда могут сформировать запрос на прерывание. В простейшем случае центральный процессор получает запрос на прерывание, приостанавливает выполнение текущей программы, обслуживает запрос и возвращается к выполнению программы.

6.1 Блоки обслуживания прерываний

Для маскируемых прерываний может быть выбран любой из двух блоков обработки прерываний. Немаскируемые прерывания (в том числе специальные – пошагового исполнения, NMI, DEBUG и по неподдерживаемому коду) всегда обрабатываются контроллером прерываний (т.е. только программно).

Контроллер прерываний

Контроллер прерываний обслуживает прерывания совместно с подпрограммами обработки прерываний. Когда аппаратура обнаруживает прерывание, она генерирует и выполняет специальный вызов подпрограммы обслуживания прерывания. При этом в стек помещается содержимое программного счётчика, а затем счетчик команд PC загружается значением, соответствующим вектору прерывания (верхние и нижние области векторов прерываний, расположенные в специально отведённой области внешней памяти, содержат адреса подпрограмм обслуживания прерываний). Далее ЦПУ выполняет подпрограмму обслуживания прерывания, после чего программный счетчик PC загружается значением из стека, и продолжается выполнение прерванной программы.

Сервер периферийных транзакций PTS

PTS представляет собой блок аппаратной обработки прерываний (только маскируемых), который может использоваться вместо стандартных программ обработки прерываний и содержащий набор встроенных алгоритмов, исходные данные для которых должны быть размещены во внутреннем ОЗУ и/или во внешней памяти.

PTS предназначен для обеспечения некоторой последовательности событий, выполняемых на микропрограммном уровне в заданное время без участия процессора. Такими событиями могут быть:

- передача блока информации из одного места памяти (или устройства ввода-вывода) в другое;
- выгрузка данных из блока HSI;
- загрузка данных в блок HSO.

Алгоритмы PTS охватывают, в основном, пересылки данных. Пересылка может осуществляться и в режиме IDLE.

PTS обслуживает прерывания параллельно работе основной программы, не модифицирует стек или PSW и осуществляет передачу данных только во время простоя шины данных (моменты, когда осуществляются арифметические операции, переходы, простои вследствие пустой очереди команд и т.п.). Пример передачи данных представлен на рисунке 6.1. Более подробное описание блока PTS представлено ниже.

Рисунок 6.1 – Пример передачи PTS данных

PTS работает в пяти специальных микропрограммных режимах, которые позволяют ему выполнять отдельные задачи гораздо быстрее, чем подпрограммам обслуживания прерываний. Если возникает запрос на прерывание, то специальный декодировщик приоритетов выбирает соответствующий вектор (из таблицы векторов PTS) и вызывает специальный блок управления (PTSCB). Каждое прерывание PTS генерирует один цикл, показанный на рисунке 6.2.

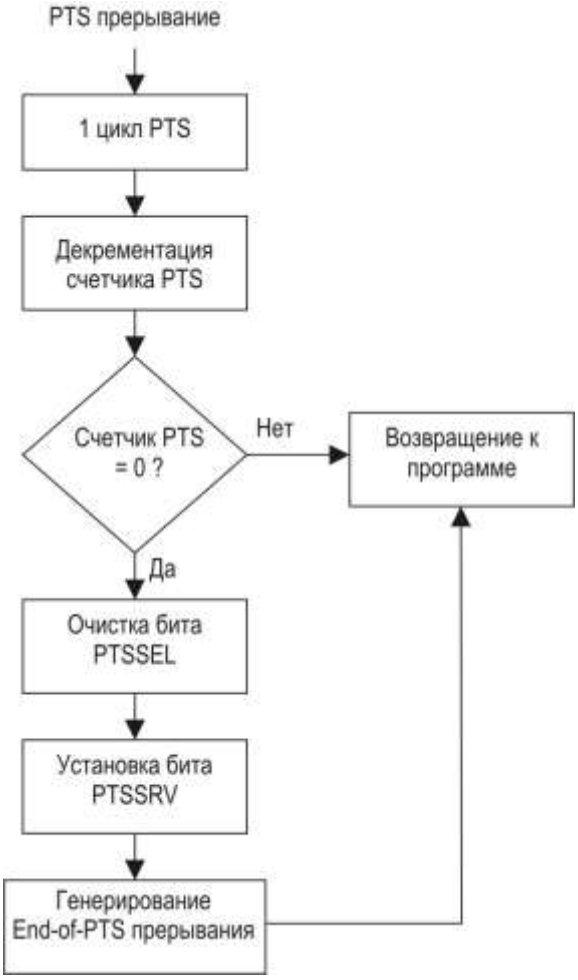


Рисунок 6.2 – Цикл прерывания PTS

На рисунке 6.3 представлен алгоритм обслуживания стандартного прерывания и прерывания PTS.

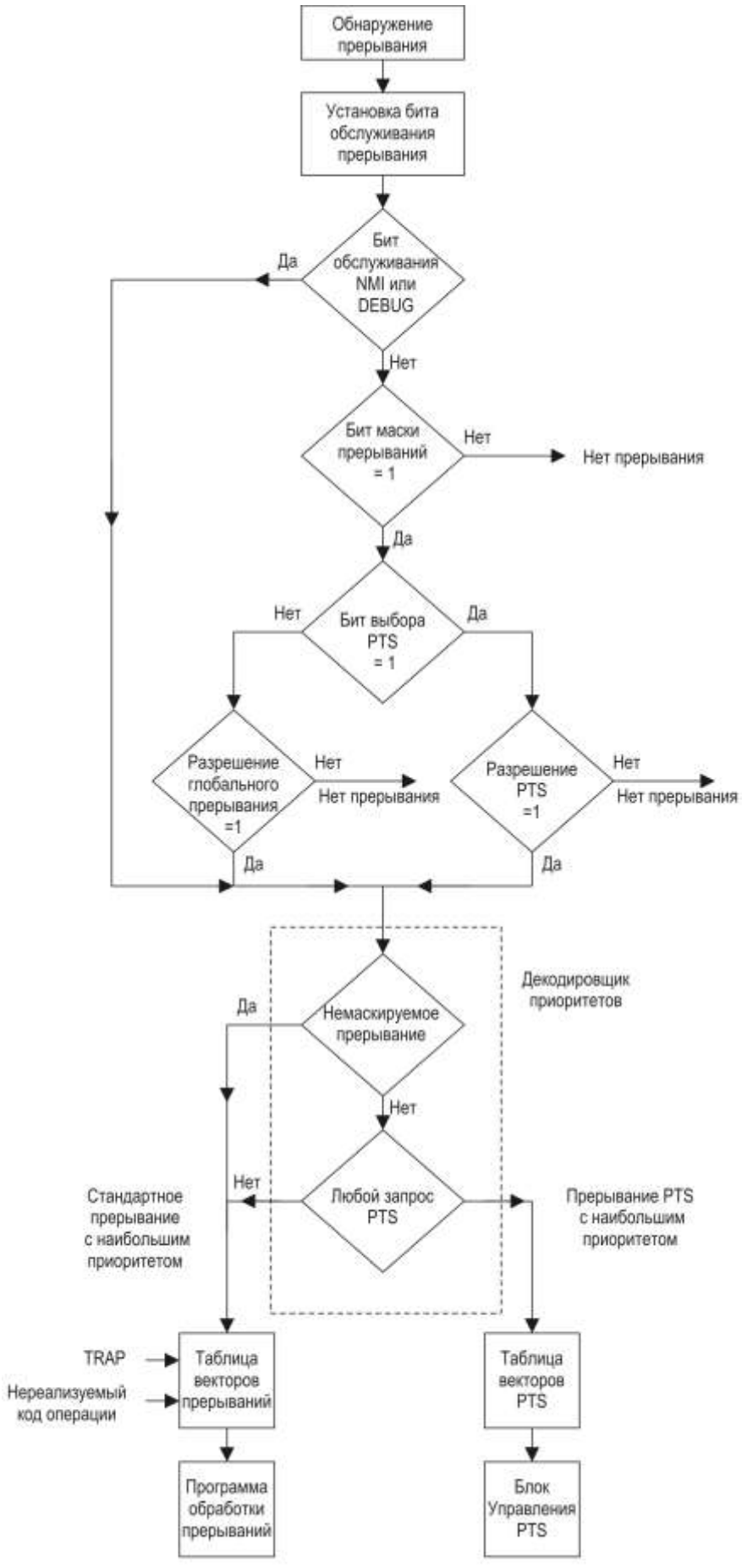


Рисунок 6.3 – Алгоритм обслуживания стандартного прерывания и прерывания PTS

6.2 Управление прерываниями

Приоритеты прерываний

Прерывания по неподдерживаемому коду и программные прерывания не имеют приоритетов и напрямую проходят в контроллер прерываний для обслуживания. Контроллер прерываний выбирает соответствующий вектор (ячейку памяти), расположенный в памяти специального назначения (см. рисунок 6.2). Вектор (ячейка памяти) содержит стартовый адрес программы обработки прерываний.

Все остальные прерывания имеют собственные уровни приоритета, определяемые декодировщиком приоритетов (уровень 16 является высшим, 0 – низшим).

В таблице 6.1 представлены все векторы прерываний с указанием их номеров, адресов и приоритетов.

Таблица 6.1 – Векторы прерываний

Источник прерывания	Номер	Вектор прерывания	Ячейка вектора прерывания	Ячейка вектора PTS	При-ори-тет
Неподдерживаемый код	–	UNIOPCODE	2012h	–	–
Команда TRAP	–	SWTRAP	2010h	–	–
Модуль отладки	INT16	DEBUG	2060h	–	16
NMI	INT15	NMI	203Eh	–	15
Буфер FIFO HSI полный	INT14	HSIFIFOFULL	203Ch	205Ch	14
Внешнее прерывание по выводу P2.2	INT13	EXTINT1	203Ah	205Ah	13
Переполнение таймера 2	INT12	T2OVF	2038h	2058h	12
Захват значения таймера 2	INT11	T2CAP	2036h	2056h	11
Четвёртая запись в FIFO модуля HSI	INT10	HSIFIFO4	2034h	2054h	10
Контроллер магистрального последовательного интерфейса ГОСТ Р 52070–2003	INT09	BSIINT	2032h	2052h	9
Флаги RI и TI модулей UART0 и UART1	INT08	RECTRANS	2030h	2050h	8
Внешнее прерывание по выводу P2.2 или по выводу P0.7	INT07	EXTINT	200Eh	204Eh	7
Прерывание SPI, I2C, SpaceWire	INT06	SERPORT	200Ch	204Ch	6
Программное прерывание 0/1/2/3, сброс таймера 2, запуск АЦП	INT05	SWTIMER	200Ah	204Ah	5
Вход 0 модуля HSI	INT04	HSI0	2008h	2048h	4
Выходы 0 – 5 модуля HSO	INT03	HSOINT	2006h	2046h	3
Заполнение FIFO или загрузка фиксирующего регистра модуля HSI	INT02	HSIDATAV	2004h	2044h	2
Завершение аналого-цифрового преобразования или срабатывание цифровых компараторов	INT01	ADCOM	2002h	2042h	1
Таймер 1 или таймер 2	INT00	TOVF	2000h	2040h	0
Примечание – Каждое из маскируемых прерываний с номерами INT14 – INT00 может быть назначено на PTS. Любое из PTS прерываний имеет приоритет над всеми другими маскируемыми прерываниями.					

Любой запрос прерывания PTS имеет больший приоритет, чем запрос маскируемого прерывания. Если нет запроса прерывания от модуля отладки и немаскируемого прерывания (NMI), то декодировщик приоритетов устанавливает высший приоритет для запроса прерываний PTS, и контроллер прерываний выбирает соответствующий вектор PTS (в области адресов 2040h – 205Dh внешней памяти), который содержит стартовый адрес специального блока управления PTS. Если нет и прерывания PTS, то декодировщик приоритетов устанавливает наивысший приоритет запросам стандартных прерываний, и контроллер прерываний выбирает соответствующий вектор (из памяти специального назначения внешней памяти), который содержит стартовый адрес соответствующей подпрограммы обработки прерывания.

Изменение приоритетов прерываний

Приоритеты маскируемых прерываний, определенные по умолчанию, можно перепрограммировать посредством регистров масок прерываний INT_MASK и INT_MASK1. Можно задать, какие прерывания будут обслуживаться подпрограммой обработки прерывания, а какие – в цикле PTS. Ниже представлен вариант подпрограммы обработки запроса прерывания от модуля UART0, которая запрещает все прерывания, кроме EXTINT (вектор EXTINT с приоритетом 7, вектор REC с приоритетом 9).

```
Serial_int:          ; Вход в подпрограмму обработки прерывания
PUSHA                ; Сохранение регистров PSW, INT_MASK и WSR
DI                   ; Запрет обслуживания всех прерываний
LDB INT_MASK1, #20h  ; Разрешение внешнего прерывания
EI                   ; Разрешение обслуживания всех прерываний
POPA                 ; Восстановление регистров PSW, INT_MASK и WSR
RET                  ; Выход из подпрограммы обработки прерывания
```

Вектор прерывания REC, соответствующий прерыванию модуля UART0, расположен по адресу 2032h, и должен содержать адрес, по которому расположена метка Serial_int.

Пояснение к приведенной выше программе

После обнаружения прерывания и определения его приоритета, аппаратная часть запускает механизм обслуживания прерывания. Значение программного счетчика PC помещается в стек, а сам он загружается значением из ячейки вектора, соответствующего немаскируемому прерыванию с наибольшим приоритетом. Далее никакое другое прерывание не будет обслужено до тех пор, пока не будет выполнена первая команда подпрограммы обработки прерывания (в примере это команда PUSHA).

Команда PUSHA, которая гарантированно будет выполнена, помещает в стек значения регистров PSW, INT_MASK1 и WSR, после чего очищает регистры PSW и INT_MASK1. Очистка регистра PSW, помимо сброса арифметических флагов, сбрасывает флаги I и PSE, что в совокупности с очисткой регистра INT_MASK1 маскирует все маскируемые прерывания, запрещает стандартную процедуру обслуживания прерываний и PTS.

Команда DI запрещает обслуживание всех прерываний, после чего программирование регистра INT_MASK1 (например, посредством команды LDB), позволяет разрешить и/или запретить прерывания от разных источников, тем самым создавая возможность программного распределения приоритетов (в примере разрешается только внешнее прерывание с вектором EXTINT). После этого обслуживание всех прерываний разрешается командой EI, но запросы на прерывания начнут обслуживаться только после выполнения команды, следующей за командой EI (в примере после команды POPA). В приведенном примере между командами EI и POPA не выполняется никаких действий, но фактически в этом месте может располагаться программа пользователя.

В конце программы обработки прерывания находится команда POPA, которая загружает регистры PSW, INT_MASK1 и WSR их исходными значениями, которые хранились в стеке (естественно, все значения, записанные в эти регистры в течение выполнения программы обработки прерывания, теряются). Запросы на прерывания начнут обслуживаться только после выполнения команды, следующей за командой POPA, т.е. после выполнения команды RET, что автоматически исключает ситуацию запуска механизма обслуживания следующего прерывания до завершения обслуживания текущего, которая могла бы привести к переполнению стека.

Временные ограничения прерываний

Микроконтроллер поддерживает пять внешних источников прерываний (см. таблицу 6.2).

Таблица 6.2 – Внешние источники прерываний

Вывод микроконтроллера	Альтернативная функция вывода	Номер прерывания	Вектор прерывания
NMI	–	INT15	NMI
P0.7	EXTINT	INT07	EXTINT
P2.2	EXTINT	INT07	EXTINT
		INT13	EXTINT1
	HSI.0	INT04	HSI0

На всех выводах, которые используются в качестве источников прерываний, активным уровнем сигнала является уровень логической единицы, поэтому аппаратно обнаруженный переход сигнала из «нуля» в «единицу» интерпретируется как запрос прерывания. Для схемы мониторинга внешних прерываний важно, чтобы высокий уровень сигнала на выводе сохранялся в течение не менее двух тактов синхросигнала микроконтроллера.

В системе команд микроконтроллера есть шесть особых команд – DI, EI, POPA, POPF, PUSHA и PUSHF. Каждая их этих команд (помимо своего основного назначения) запрещает обслуживание всех прерываний до тех пор, пока не будет выполнена следующая за ней команда.

Прерывания также запрещаются после:

- прерывания по неподдерживаемому коду;
- прерывания по команде TRAP (программное прерывание).

Отклик на прерывание

От момента обнаружения запроса на прерывание до выборки первой команды подпрограммы обработки прерывания (или цикла PTS) проходит некоторое время, которое напрямую зависит от того, в какой фазе выполнения команды был сформирован запрос. Если запрос на прерывание возникает не позднее, чем за четыре машинных цикла до окончания выполнения команды, то его обслуживание начнется по окончании выполнения команды. В случае более позднего появления запроса отклик на него будет получен только по окончании выполнения следующей команды.

Если прерывание обрабатывается контроллером прерываний, то после запуска механизма обслуживания требуется 16 машинных циклов для выбора соответствующего вектора прерывания, загрузки стека и перехода к выборке первой команды. Если используется стек, расположенный во внешней памяти, то добавляются еще несколько машинных циклов, что обусловлено работой внешней шины. В случае если прерывание поступает на блок PTS, то сразу же запускается цикл PTS.

Таким образом, время реакции на прерывание обозначается в машинных циклах (мц):

- для контроллера прерываний время отклика t_1 вычисляется следующим образом:

4 мц + t_2 (время выполнения команды) + 16 мц;

- для блока PTS время отклика t_1 составит: 4 мц + t_2 (время выполнения команды).

Программирование прерываний

Любое из прерываний с номерами от INT00 до INT14 может быть направлено для обслуживания как в контроллер прерываний, так и в блок PTS. Задать вариант обслуживания для каждого прерывания можно посредством регистра PTSSEL. Биты регистра с 14 по 0 соответствуют прерываниям с номерами INT14 – INT00 в том же порядке. По умолчанию все биты регистра PTSSEL сброшены, и все запросы на прерывания направляются в контроллер прерываний.

Для глобального разрешения обслуживания прерываний следует установить бит I регистра PSW. Бит устанавливается командой EI. Для запрета обслуживания всех прерываний, т.е. для сброса бита I используется команда DI.

Помимо глобального разрешения обслуживания, каждое прерывание должно быть индивидуально разрешено посредством соответствующего бита. Биты управления прерываниями находятся в регистрах масок INT_MASK и INT_MASK1. Прерываниям с номерами INT15 – INT8 соответствуют биты с седьмого по нулевой регистра INT_MASK1, а прерываниям с номерами INT7 – INT0 соответствуют биты с седьмого по нулевой регистра INT_MASK в том же порядке (см. рисунок 6.4).

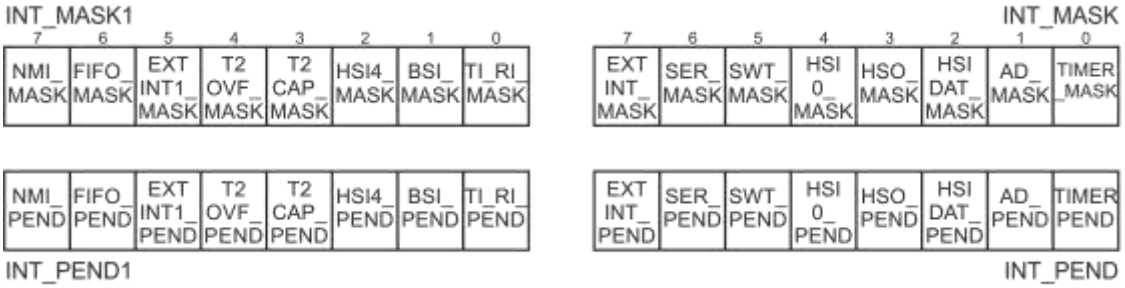


Рисунок 6.4 – Регистры масок прерываний и ждущих прерываний

Дополнительными регистрами обслуживания прерываний являются регистры ждущих прерываний INT_PEND и INT_PEND1. Как только система мониторинга прерываний обнаруживает запрос, устанавливается соответствующий бит регистра, являющийся индикатором. Прерываниям с номерами INT15 – INT08 соответствуют биты с седьмого по нулевой регистра INT_PEND1, а прерываниям с номерами INT07 – INT00 соответствуют биты с седьмого по нулевой регистра INT_PEND в том же порядке.

Индикаторы обнаруженных запросов прерываний устанавливаются всегда, независимо от состояния бита I регистра PSW как для немаскированных, так и для замаскированных прерываний. Регистры INT_PEND и INT_PEND1 всегда доступны для чтения и записи, что позволяет не только следить за возникающими запросами прерываний, но и отменять или формировать их программно, обнуляя или устанавливая соответствующие биты. Аппаратно каждый индикатор запроса на прерывание сбрасывается сразу после запуска механизма обслуживания соответствующего прерывания.

Для некоторых прерываний, помимо глобального и индивидуального разрешения, требуется дополнительное разрешение, которое осуществляется посредством регистра IOC1:

- бит HSI_INT задает источник прерывания для вектора HSIDATAV (INT02);
- биты T2_OVF_INT и T1_OVF_INT – для вектора TOVF (INT00);
- бит EXTINT_SRC – для вектора EXTINT (INT07).

На рисунке 6.5 показаны источники прерываний и соответствующие им векторы, а также управляющие биты. Непоказанные на рисунке 6.5 линии прерываний не имеют управляющих бит, т.е. при возникновении запроса на прерывание сразу осуществляется переход по вектору прерывания.

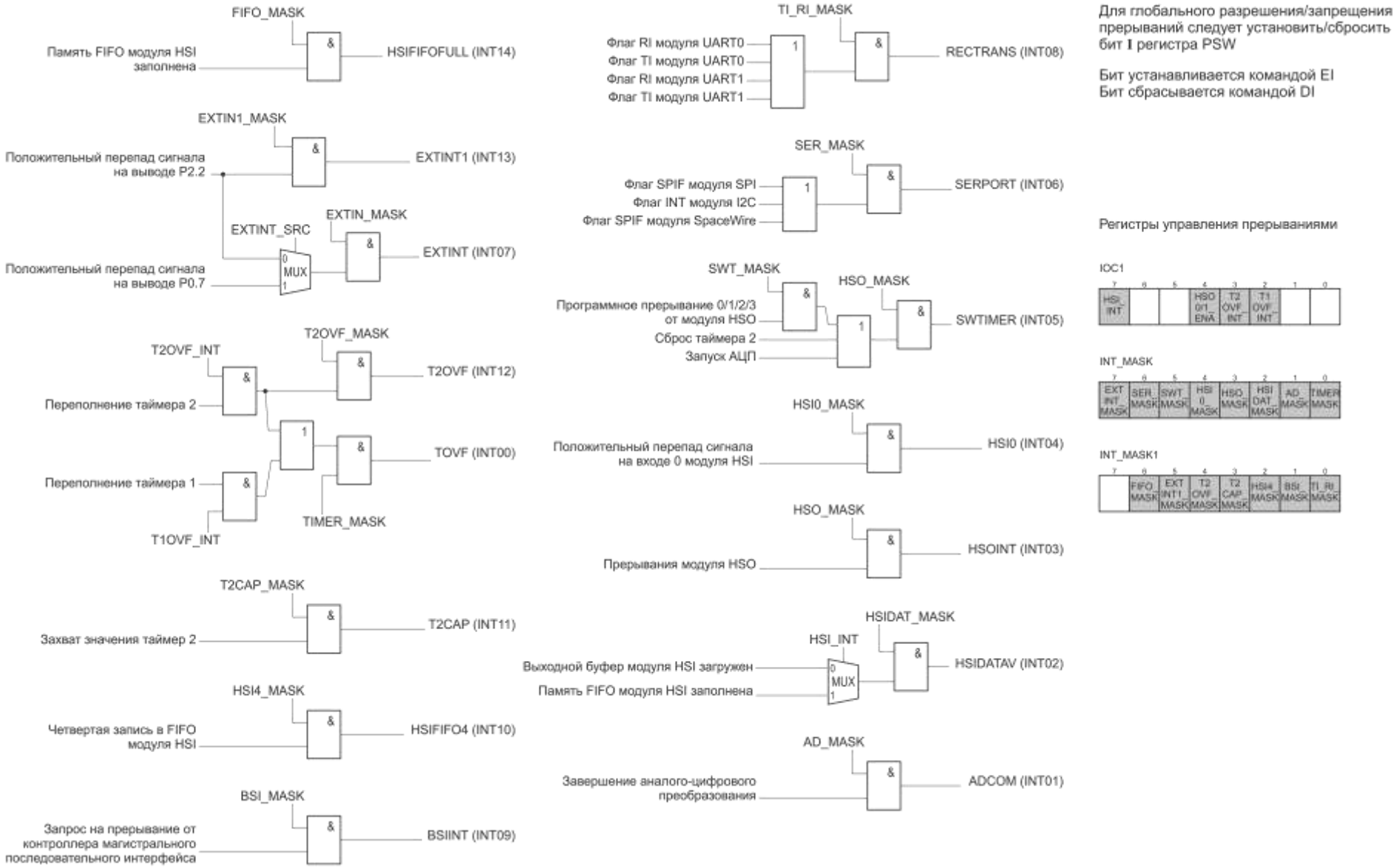


Рисунок 6.5 – Источники прерываний, управляющие биты и векторы прерываний

6.3 Специальные прерывания

Микроконтроллер поддерживает четыре специальных прерывания:
- по неподдерживаемому коду (вектор UNIOPCODE);
- по команде TRAP (программное прерывание с вектором SWTRAP);
- модуля отладки (вектор DEBUG);
- внешнее по выводу NMI микроконтроллера (вектор NMI).

Эти прерывания обслуживаются только контроллером прерываний (не могут быть назначены на PTS), не могут быть замаскированы и на них не влияет состояние бита I регистра PSW. Только прерывания с векторами NMI и DEBUG проходят через детектор передачи и декодировщик приоритетов, остальные два – напрямую в контроллер прерываний.

Неподдерживаемый код. При попытке ЦПУ выполнить неподдерживаемый код, появившийся в результате программной или аппаратной ошибки, произойдет переход по вектору UNIOPCODE с адресом 2012h. Вектор должен содержать стартовый адрес подпрограммы обработки ошибки.

Программное прерывание. Используется при отладке программ и генерации программных прерываний и позволяет вырабатывать прерывания после выполнения каждой команды. Вектор SWTRAP, расположенный по адресу 2010h, должен содержать стартовый адрес подпрограммы обслуживания прерывания.

Прерывание модуля отладки. Вектор прерывания DEBUG расположен по адресу 2060h. Прерывание имеет наивысший приоритет и используется при отладке программ (подробнее в разделе 17).

Немаскируемое прерывание. Внешнее прерывание. Формируется сигналом, приходящим на вывод NMI микроконтроллера. Имеет 15 уровень приоритета и используется для экстренного прерывания микроконтроллера. Программа обслуживания прерывания вызывается косвенно через вектор NMI с адресом 203Eh.

Если вывод NMI не используется, его необходимо заземлить.

6.4 Блок PTS

Разрешением работы блока PTS управляет бит PSE регистра PSW. Бит устанавливается командой EPTS (разрешение PTS) и сбрасывается командой DPTS (запрещение PTS).

Для того, чтобы запрос на прерывание был направлен в блок PTS, следует установить соответствующий бит в регистре PTSSEL.

Прежде чем направлять запросы прерываний в блок PTS, следует сформировать управляющие блоки. Для каждого прерывания формируется отдельный блок (PTSCB), состоящий из восьми байт, каждый из которых становится управляющим регистром. Задаются регистры: PTSCOUNT, PTSCON, PTSSRC (формируется двумя регистрами PTSSRC_H и PTSSRC_L), PTSDST (формируется двумя регистрами PTSDST_H и PTSDST_L) и PTSBLOCK (см. рисунок 6.6). Блок определяет режим, общее число передач, общее число циклов и адреса источника и приемника передач данных.

Все блоки размещаются в регистровом ОЗУ по четным адресам. Адрес начала блока записывается в ячейку вектора, который соответствует прерыванию. Если в управляющем блоке имеются неиспользуемые байты, то они могут использоваться, как регистры общего назначения ОЗУ.

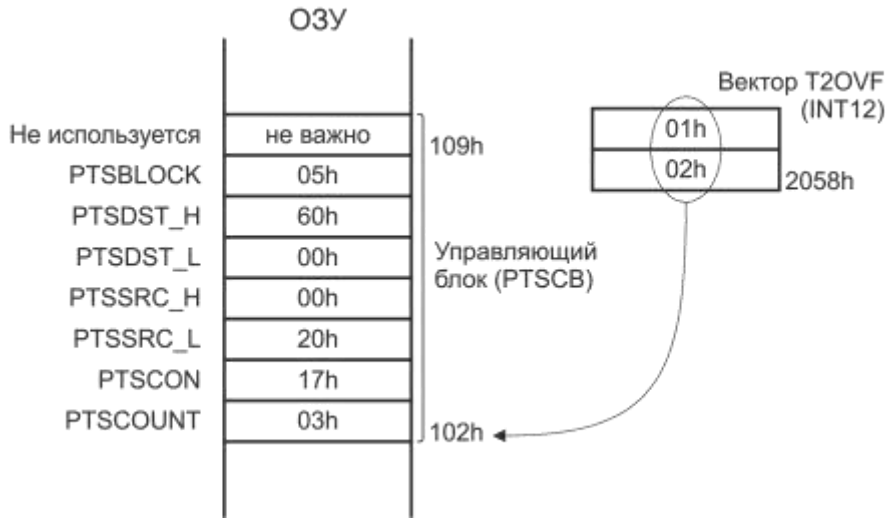


Рисунок 6.6 – Структура управляющего блока (PTSCB) обслуживания прерывания от таймера 2 с вектором T2OVF (INT12)

Блок PTS поддерживает четыре режима функционирования – одиночной передачи, блочной передачи, режимы HSO и HSI. Конфигурация управляющих блоков меняется в зависимости от режима работы. На рисунке 6.7 показаны варианты структур управляющих блоков для каждого режима.

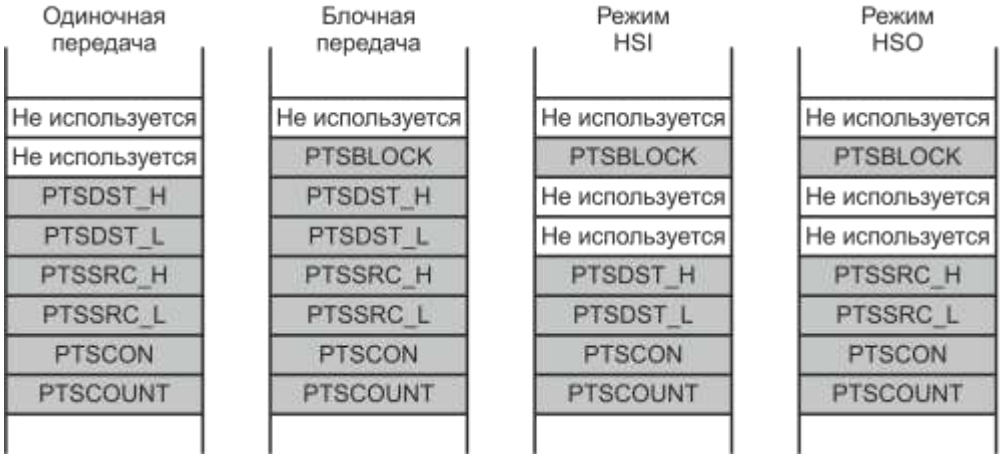


Рисунок 6.7 – Структура управляющего блока в зависимости от режима работы

Регистр PTSCOUNT

В начальном байте каждого управляющего блока всегда располагается счетный регистр PTSCOUNT, который задает число циклов PTS (максимально – 256), которые будут последовательно выполнены без участия центрального процессора. Регистр PTSCOUNT декрементируется в конце каждого цикла PTS и как только содержимое регистра станет равно нулю, соответствующий бит в регистре PTSSEL сбросится, а в регистре PTSSRV установится индикатор запроса на прерывание END-OF-PTS. Прерывание END-OF-PTS является стандартным прерыванием и обрабатывается контроллером прерываний.

Пример. Чтобы направить запрос прерывания от таймера 2 с вектором T2OVF (INT12) на блок PTS, следует установить бит T2OVF_SEL регистра PTSSEL. При возникновении запроса на прерывание аппаратная часть обратится к ячейке памяти с адресом 2058h (см. таблицу 6.1), чтобы получить адрес начала соответствующего управляющего блока (на рисунке 6.6 этот адрес равен 102h).

Когда счетчик циклов обнулится, аппаратная часть сгенерирует прерывание END-OF-PTS и в регистре PTSSRV установится индикатор T2OVF_SRV, при этом бит T2OVF_SEL сбросится. Далее прерывание END-OF-PTS будет направлено на обслуживание в контроллер прерываний с вектором T2OVF, которому соответствует ячейка с адресом 2038h (адрес, используемый контроллером прерываний при стандартной процедуре обслуживания прерывания от таймера 2). Как только контроллер прерываний запустит механизм обслуживания прерывания END-OF-PTS, бит T2OVF_SRV будет сброшен.

Для того, чтобы последующие прерывания от таймера 2 направлялись на PTS, бит T2OVF_SEL следует снова установить.

Регистр PTSSRV фактически является аналогом регистров INT_PEND и INT_PEND1 и позволяет не только следить за возникающими запросами прерываний END-OF-PTS, но и отменять или формировать их программно, обнуляя или устанавливая соответствующие биты. Каждому прерыванию, которое может быть направлено в блок PTS, соответствует один бит регистра PTSSRV. Прерываниям с номерами INT14 – INT00 соответствуют биты регистра с 14 по 0 в том же порядке.

Регистр PTSCON

Основной регистр управляющего блока, предназначенный для задания режима его работы и конфигурации. В режимах одиночной и блочной передачи этот регистр имеет один формат, а в режимах HSO/HSI – другой.

Регистр PTSSRC

Регистр используется в режимах одиночной и блочной передач и в режиме HSO и содержит адрес ячейки памяти источника.

В режиме одиночной передачи биты SI и SU регистра PTSCON определяют, будет ли PTSSRC инкрементироваться.

В режиме блочной передачи бит SI определяет, будет ли PTSSRC инкрементироваться после каждой передачи, а бит SU – будет ли PTSSRC сохранять свое последнее значение или восстанавливать первоначальное.

В режиме HSO бит SU определяет, будет ли PTSSRC обновляться в конце цикла обмена.

Регистр PTSDST

Регистр используется в режимах одиночной и блочной передач и в режиме HSI и содержит адрес ячейки памяти приемника.

В режиме одиночной передачи биты DI и DU регистра PTSCON определяют, будет ли PTSDST инкрементироваться.

В режиме блочной передачи бит DI определяет, будет ли PTSDST инкрементироваться после каждой передачи, а бит DU – будет ли PTSDST сохранять свое последнее значение или восстанавливать первоначальное.

В режиме HSO бит DU определяет, будет ли PTSDST обновляться в конце цикла обмена.

Регистр PTSBLOCK

Регистр используется во время блочной передачи и в режимах HSI и HSO и управляет числом передач.

Режим одиночной передачи

В режиме одиночной передачи в течение цикла PTS передается один байт (если установлен бит BW регистра PTSCON) или одно слово (если бит BW сброшен) из одной ячейки памяти в другую. Этот режим обычно используется с прерываниями по последовательному порту ввода-вывода или концу аналого-цифрового преобразования. Количество передач задается регистром PTSCOUNT. Адреса ячеек памяти источника и приемника байта/слова задаются регистрами PTSSRC и PTSDST, которые могут указывать на любую ячейку памяти (в случае передачи слова адрес ячейки должен быть четным). Пары бит SU-SI и DU-DI управляют состояниями регистров PTSSRC и PTSDST

(могут инкрементироваться независимо друг от друга). Оба бита каждой пары должны иметь одинаковые значения. Запись 0b-1b или 1b-0b запрещена.

Пример режима одиночной передачи.

На рисунке 6.8 показано состояние управляющего блока в начале обслуживания прерывания и в конце. Счетный регистр задает восемь циклов PTS. Каждый цикл пересылает одно слово (бит BW очищен) из ячейки с адресом 20h в ячейку внешней памяти. После первой передачи слова в ячейку с адресом 6000h значение регистра PTSDST инкрементируется на 2 (установлен бит DI), значение счетного регистра уменьшается на единицу. Состояние регистра PTSSRC не изменяется, поскольку биты SU и SI не установлены. Далее цикл повторяется восемь раз.

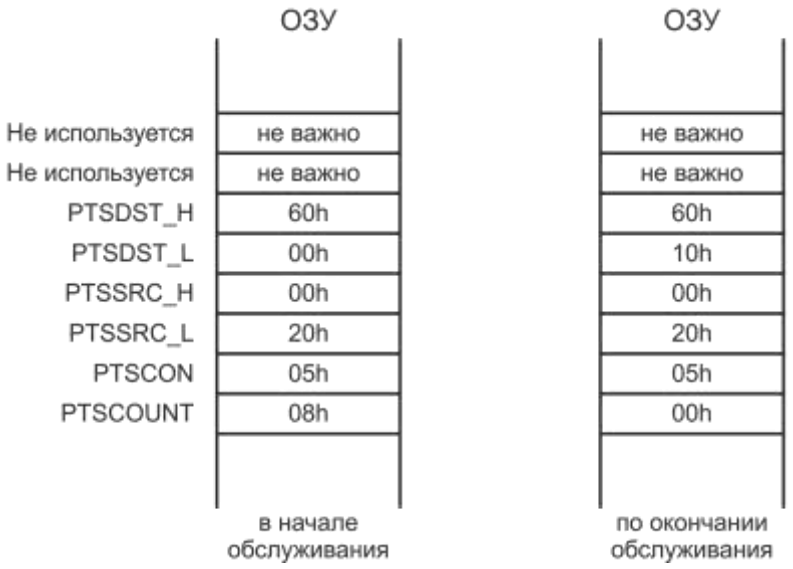


Рисунок 6.8 – Управляющий блок в режиме одиночной передачи

По окончании восьми передач будет заполнена область внешней памяти с адреса 6000h по 600Fh. Счетный регистр обнулится, и сформируется прерывание END-OF-PTS. Состояние регистра PTSDST не сбросится в начальное, поскольку установлен бит DU.

Режим блочной передачи

В режиме блочной передачи в течение цикла PTS может быть передано от одного до 32 байт или слов (в зависимости от состояния бита BW) из одной области памяти в другую. Количество передаваемых байт/слов задается регистром количества блоков PTSBLOCK. Количество циклов PTS задается регистром PTSCOUNT. Адреса ячеек памяти источника и приемника байт/слов задаются регистрами PTSSRC и PTSDST, которые могут указывать на любую ячейку памяти (в случае передачи слова адрес ячейки должен быть четным). В отличие от режима одиночной передачи, биты SU, SI, DU и DI могут устанавливаться независимо друг от друга. Установка бита SI/DI включает автоинкремент регистра PTSSRC/PTSDST, т.е. после каждой передачи байта/слова значение регистра будет увеличиваться на один/два. Установка бита SU/DU задает режим работы регистра PTSSRC/PTSDST, когда этот регистр сохраняет значение адреса, которое было в нем на момент завершения цикла PTS. Таким образом, одновременная установка битов SU-SI/DU-DI задает режим последовательного увеличения содержимого регистра PTSSRC/PTSDST.

Пример режима блочной передачи.

На рисунке 6.9 показано состояние управляющего блока в начале обслуживания прерывания и в конце. Счетный регистр задает три цикла PTS. В каждом цикле должно быть передано пять байт, что задается регистром PTSBLOCK (значение 05h) и битом BW (установлен).

Первый байт передается из ячейки с адресом 20h в ячейку с адресом 6000h, после чего регистры PTSSRC и PTSDST инкрементируются (биты SI и DI установлены). Далее передаются остальные четыре байта. Таким образом, заполняются ячейки с адресами 6000h – 6004h. На этом заканчивается первый цикл PTS. Регистр PTSCOUNT декрементируется. Поскольку бит DU установлен, то в регистре PTSDST сохраняется значение 6005h, которое было в нем на момент окончания первого цикла PTS. В тоже время регистр PTSSRC восстанавливает свое исходное значение 0020h, поскольку бит SU не установлен. Далее цикл PTS повторяется.

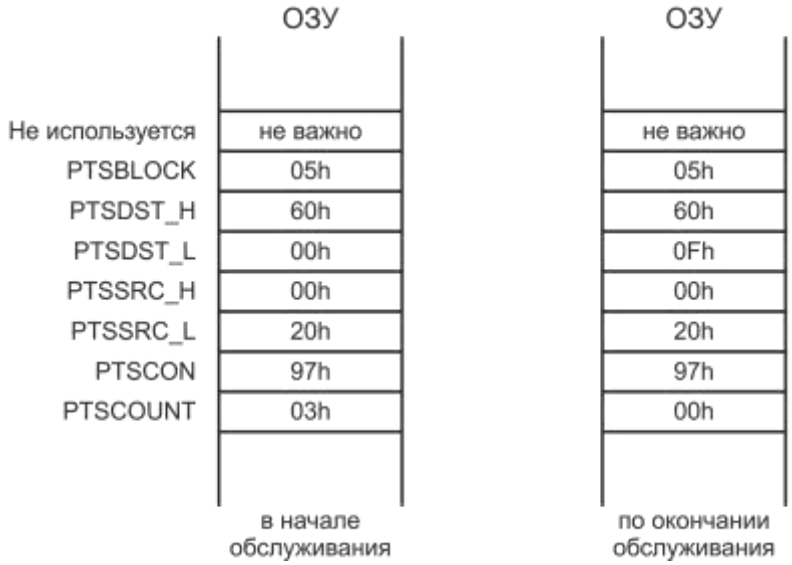


Рисунок 6.9 – Управляющий блок в режиме блочной передачи

Таким образом, по окончании трех циклов PTS значения из области памяти с адресами 20h – 24h будут последовательно скопированы в области с адресами 6000h – 6004h, 6005h – 6009h, 600Ah – 600Eh.

Режим HSI

В режиме HSI блок PTS загружает содержимое памяти FIFO модуля HSI во внутреннюю, либо внешнюю память, формируя так называемую «Таблицу». Адрес начала «Таблицы» находится в регистре PTSDST. Любое прерывание модуля HSI может быть использовано для запуска цикла PTS. Значение, записанное в регистр PTSBLOCK, указывает, сколько слов из памяти FIFO будет передано в «Таблицу» в течение каждого цикла PTS (максимально семь). Необходимо ввести значение, соответствующее прерыванию, которое запускает цикл PTS. Например, четвертая запись в память FIFO может явиться источником прерывания модуля HSI с вектором HSIFIFO4 (INT10). Если это прерывание использовалось для запуска цикла PTS, то в регистр PTSBLOCK следует записать значение 04h, таким образом, четыре значения из памяти FIFO будут переписаны в «Таблицу».

Данные памяти FIFO доступны посредством последовательного чтения регистров HSI_STATUS и HSI_TIME. Регистр HSI_STATUS содержит биты событий и текущее состояние выводов HSI. Регистр HSI_TIME содержит время обнаружения события.

С каждой передачей значения регистров HSI_STATUS и HSI_TIME последовательно копируются в «Таблицу» (см. рисунок 6.10, где «xxx» – адрес начала «Таблицы»). Для того, чтобы регистр PTSDST сохранял значение адреса, в конце каждого цикла PTS должен быть установлен бит UPDT регистра PTSCON (см. конфигурацию регистра для режима HSI/HSO).

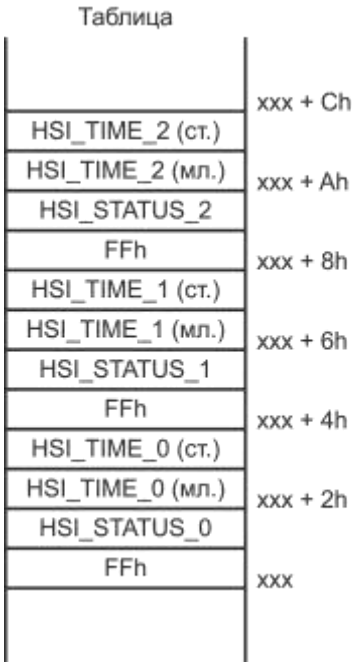


Рисунок 6.10 – Порядок заполнения «Таблицы» для режимов HSI и HSO

На рисунке 6.11 показано состояние управляющего блока в начале обслуживания прерывания и в конце. Задано 10 циклов PTS, в каждом из которых будет передано семь слов данных (значения регистров HSI_STATUS и HSI_TIME) из памяти FIFO модуля HSI в «Таблицу» с начальным адресом 100h. Начальный адрес инкрементируется в конце каждой передачи и обновляется по окончании каждого цикла PTS.

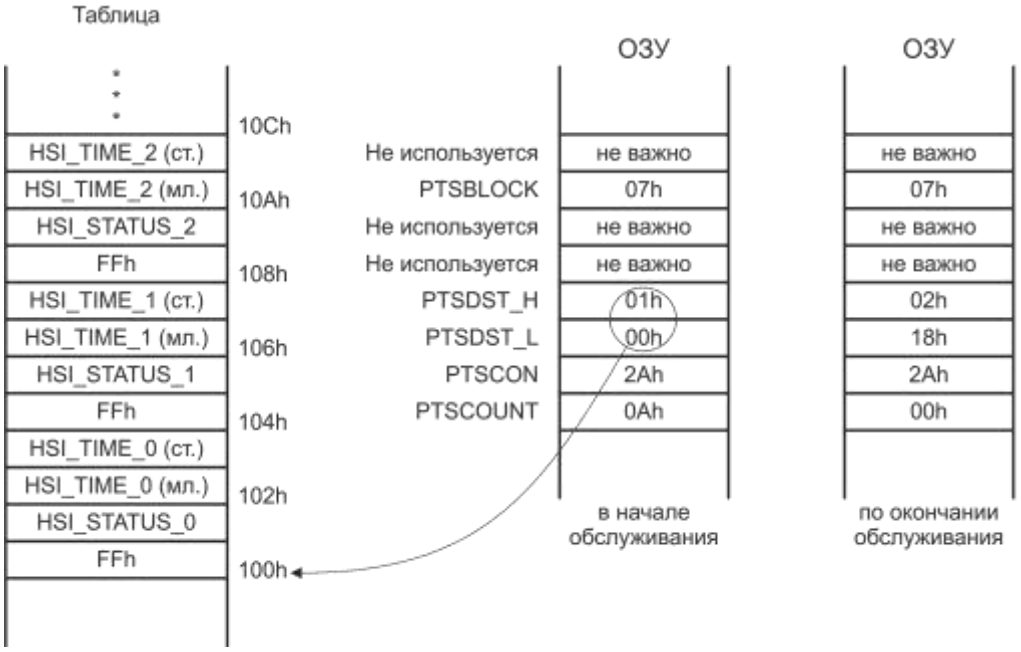


Рисунок 6.11 – Управляющий блок обслуживания прерывания модуля HSI

Режим HSO

В режиме HSO блок PTS загружает содержимое «Таблицы», расположенной во внутренней или внешней памяти, в ассоциативное запоминающее устройство (АЗУ) модуля HSO, т.е. выполняет действия, обратные тем, что выполняются в режиме HSI. Адрес начала «Таблицы» находится в регистре PTSSRC.

Все восемь слов АЗУ являются 24-разрядными. 16 бит каждого слова задают время выполнения команды, а оставшиеся 8 бит – код команды и ее параметры. Эти данные следует разместить в «Таблице» в последовательности, указанной на рисунке 6.10.

В одном цикле PTS от одного до восьми слов данных загружается последовательно через регистры HSI_COMMAND и HSI_TIME в АЗУ модуля HSO. Если в следующем цикле загружается девятое слово, а АЗУ заполнено, то это слово помещается в предварительный буфер, где находится до тех пор, пока не освободится место в АЗУ.

Любое прерывание модуля HSO может быть использовано для запуска цикла PTS. Значение, записанное в регистр PTSBLOCK, указывает, сколько записей будет скопировано из «Таблицы» в модуль HSO в течение каждого цикла PTS (максимально восемь). Для того, чтобы регистр PTSSRC сохранял значение адреса в конце каждого цикла PTS, бит UPDT регистра PTSCON должен быть установлен.

Пример режима HSO.

На рисунке 6.12 показано состояние управляющего блока в начале обслуживания прерывания и в конце. Задано 10 циклов PTS, в каждом из которых будет передано восемь слов данных из «Таблицы» с начальным адресом 100h в модуль HSO. Начальный адрес инкрементируется в конце каждой передачи и обновляется по окончании каждого цикла PTS.

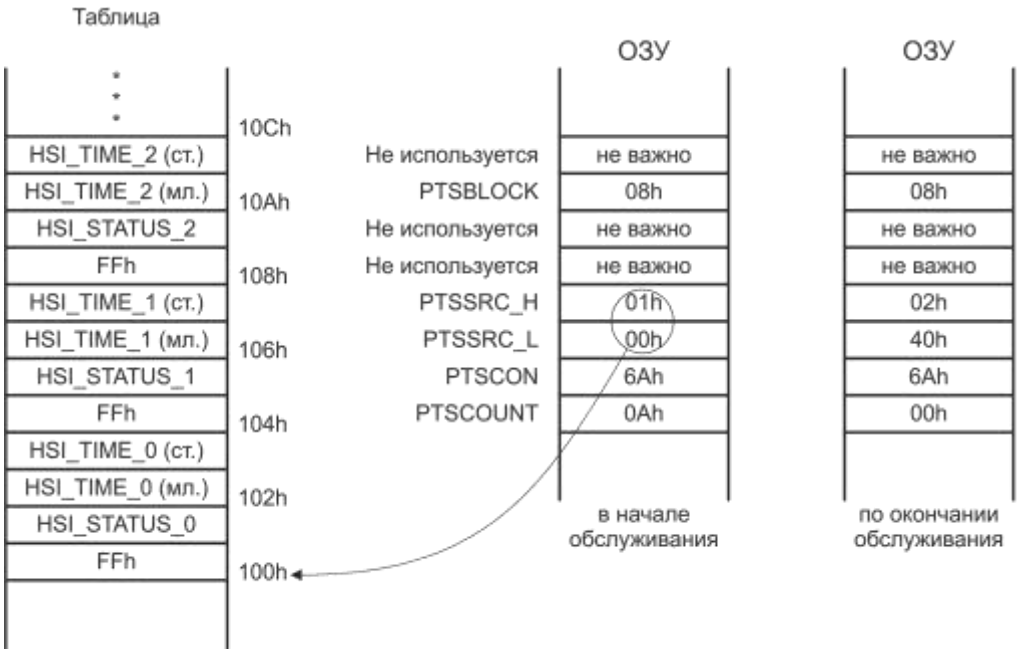


Рисунок 6.12 – Управляющий блок обслуживания прерывания модуля HSO

7 Порты ввода-вывода

Порты ввода-вывода служат для передачи информации между микроконтроллером и окружающими его устройствами. Посредством портов конфигурируется микроконтроллер, формируются управляющие сигналы для внешних устройств и считывается информация о состоянии окружения микроконтроллера.

7.1 Функциональные возможности

ИС 1874ВЕ7Т и ИС 1874ВЕ71Т имееют соответственно по шесть (P0 – P5) и пять (P0 – P4) 8-битных портов ввода-вывода. В ИС 1874ВЕ71Т шестой порт P5 реализован как 4-разрядный. Выводы портов выполняют функции ввода и/или вывода информации (входы и выходы). Имеются квазидвунаправленные выводы и двунаправленные с открытым стоком.

Вывод вход

Любой вывод порта, определенный как вход, может быть только считан. Любая операция записи недопустима или игнорируется. Функциональная схема входа показана на рисунке 7.1.

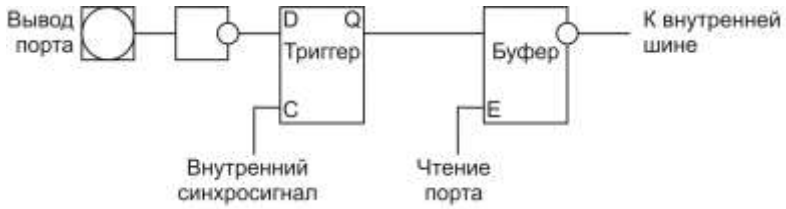


Рисунок 7.1 – Функциональная схема входа

Входной сигнал с вывода порта инвертируется и запоминается в триггере, который синхронизируется внутренним сигналом тактирования микроконтроллера. По сигналу чтения порта (пока сигнал CLOCKOUT имеет низкий уровень) содержимое триггера заносится в буфер и вместе с этим выставляется на внутреннюю шину.

Следует помнить, что частота изменения сигнала на выводе должна быть как минимум в два раза меньше рабочей частоты микроконтроллера.

Входные схемы не имеют выходных драйверов, ток утечки на входе и емкостная нагрузка очень малы.

Вывод выход

Любой вывод порта, определенный как выход, может быть только записан. Выходная схема не имеет буфера для чтения, поэтому при считывании будет получено неопределенное значение. Функциональная схема выхода с защелкой и драйверами показана на рисунке 7.2.

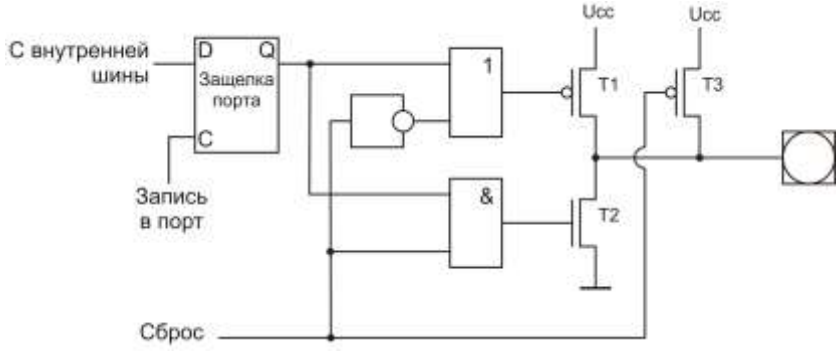
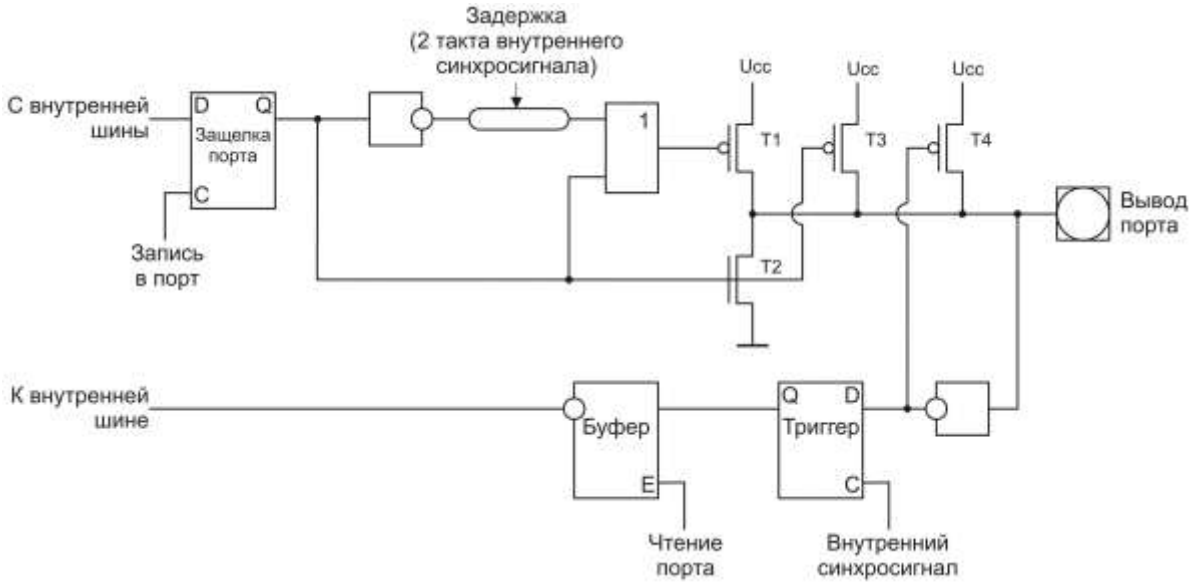


Рисунок 7.2 – Функциональная схема выхода

По сигналу записи в порт (пока сигнал CLOCKOUT имеет низкий уровень) новое значение запоминается в защелке порта и выставляется на выводе. Состояние вывода будет сохраняться до тех пор, пока не будет записано новое значение в защелку порта или не произойдет сброс микроконтроллера.

Квазидвунаправленный вывод

Квазидвунаправленный вывод может одновременно функционировать как в режиме входа, так и в режиме выхода, при этом управлять переключением не нужно. Такой вывод имеет активный низкий уровень и пассивный высокий уровень сигнала. При высоком уровне сигнала на выводе последний может функционировать как вход. В случае удержания на выводе низкого уровня сигнала внешнее устройство не может перевести вывод в состояние логической единицы. Функциональная схема квазидвунаправленного вывода показана на рисунке 7.3.



Принятые условные обозначения:
T1, T2 – сильные драйверы
T3 – очень слабый драйвер
T4 – слабый драйвер.

Рисунок 7.3 – Функциональная схема квазидвунаправленного вывода

Схема квазидвунаправленного вывода состоит из двух схем – схемы ввода сигналов и схемы вывода. Для приема сигналов с вывода используются триггер и буфер, для вывода – защелка порта и специальная логическая схема драйверов.

Запись единицы в защелку порта закрывает сильный драйвер T2 и открывает очень слабый драйвер T3. Для быстрого переключения вывода порта в единицу сильный драйвер T1 открывается на один такт внутреннего сигнала синхронизации микроконтроллера, а затем закрывается, оставляя открытым только T3. До тех пор, пока нагрузка на выводе мала, драйвер T3 и дополнительный драйвер T4 удерживают на выводе логическую единицу.

Для уменьшения суммарного тока, протекающего через вывод при подаче внешним устройством логического нуля, слабый драйвер T4 закрывается (при достижении уровня примерно 2 В).

На рисунке 7.4 приведена переходная характеристика вывода при открытых драйверах T3 и T4. По мере уменьшения напряжения U_I уменьшается ток I_I. Как только величина тока достигнет порогового значения I_{TL}, драйвер T4 закроется и включенным останется только очень слабый драйвер T3.

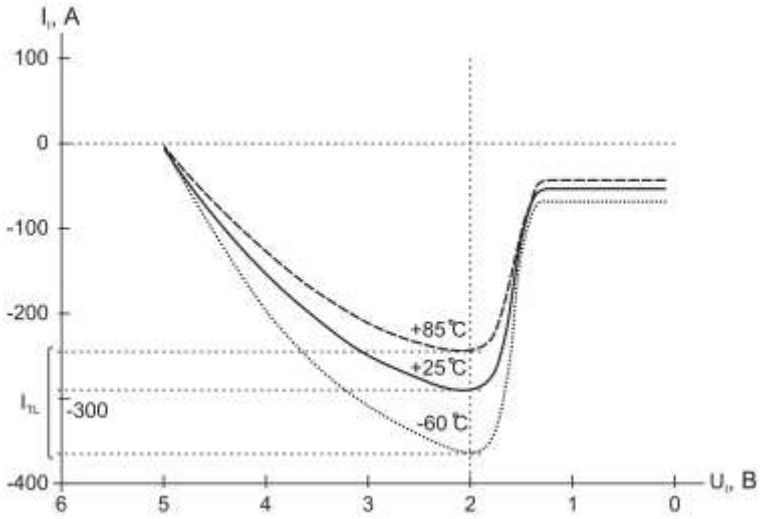


Рисунок 7.4 – Передаточная характеристика квазидвунаправленного выхода

В случае, если вывод не подключен, драйвер Т3 является подтяжкой и поддерживает на выводе маленький ток, который формирует логическую единицу.

Временные характеристики квазидвунаправленного вывода аналогичны характеристикам выводов-выходов и выводов-входов, т.е есть запись/чтение происходят, пока сигнал CLOCKOUT имеет низкий уровень.

Двунаправленный вывод с открытым стоком

Работа двунаправленного вывода с открытым стоком аналогична работе квазидвунаправленного вывода, за исключением того, что отсутствуют сильные и слабые драйверы. Функциональная схема вывода показана на рисунке 7.5.

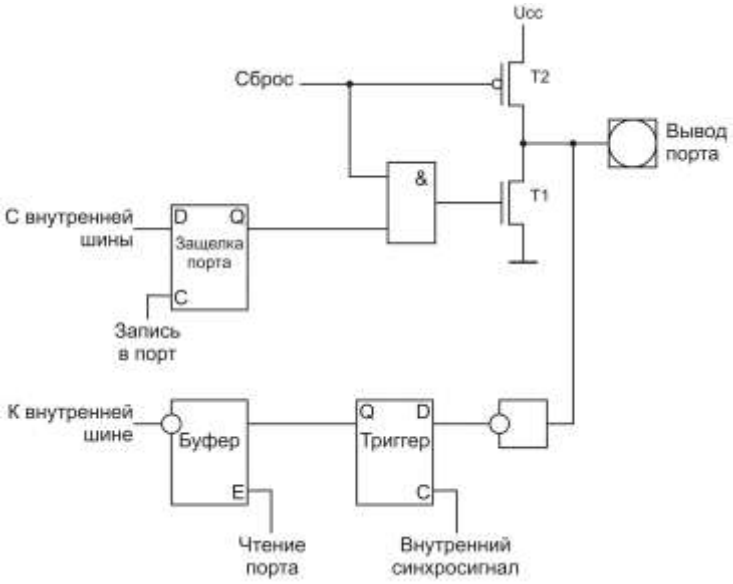


Рисунок 7.5 – Функциональная схема двунаправленного вывода с открытым стоком

Запись нуля в защелку порта открывает сильный драйвер Т1, и на выводе устанавливается логический ноль. В этом случае внешнее устройство не может перевести вывод в состояние логической единицы. Состояние вывода будет сохраняться до тех пор, пока не будет записано новое значение в защелку порта или не произойдет сброс микроконтроллера.

Запись единицы в защелку порта закроет драйвер T1, и вывод окажется в неподключенном состоянии. Снаружи уровень сигнала на выводе может быть подтянут до уровня логической единицы использованием подтягивающего резистора.

Порты 0 и 5

Выводы портов 0 и 5 могут функционировать только как входы. Каждый вывод одновременно выполняет функции как цифрового, так и аналогового входа (порты 0 и 5 выполняют функции цифрового порта, также и при включенном АЦП). Состояние портов может быть прочитано посредством регистров IOPORT0 и IOPORT5, которые всегда доступны для чтения.

Следует помнить, что чтение регистров IOPORT0 и IOPORT5 во время аналого-цифрового преобразования возвращает неопределенное значение.

Седьмой разряд порта 0 дополнительно является входом внешнего прерывания EXTINT.

Порт 1

Все выводы порта 1 квазидвунаправленные и по умолчанию функционируют, как входы-выходы общего назначения. Получать информацию о состоянии выводов порта, а также управлять их состоянием можно посредством регистра IOPORT1 (всегда доступен для записи/чтения), каждый бит которого соответствует одному выводу. Для того, чтобы вывод мог функционировать как вход, в соответствующем бите регистра IOPORT1 должна находиться единица. Для конфигурирования порта можно воспользоваться командой LDB. Например, установить на младших разрядах единицы, а на старших – нули:

LDB IOPORT1, #00001111b.

Когда на состояние выводов оказывает влияние внешнее устройство, то следует внимательнее относиться к операциям чтения, изменения и записи регистра IOPORT1 (т.е. считыванию состояния порта и его изменению). Если какой-либо вывод удерживается внешним устройством в состоянии логического нуля, то невозможно установить на этом выводе высокий уровень записью единицы в соответствующий бит регистра IOPORT1.

Низкий уровень сигнала на выводе может быть установлен записью нуля в соответствующий бит регистра IOPORT1. Поскольку в этом случае низкий уровень сигнала поддерживается сильным драйвером (см. рисунок 7.3), то подача высокого уровня от внешнего устройства приведет к протеканию большого тока короткого замыкания, что может негативно сказаться на работе как микроконтроллера, так и внешнего устройства.

Вывод 0 порта одновременно является входом-выходом общего назначения и входом SPW_DIN модуля SpaceWire. При использовании модуля SpaceWire нулевой бит регистра IOPORT1 должен быть установлен.

Вывод 1 порта одновременно является входом-выходом общего назначения и входом SPW_SIN модуля SpaceWire. При использовании модуля SpaceWire первый бит регистра IOPORT1 должен быть установлен.

Вывод 2 порта одновременно является входом-выходом общего назначения и входом выбора ведомого SS# модуля SPI и мультиплицирован с выходом HSO.0 модуля HSO. При использовании модуля SPI в качестве ведомого устройства второй бит регистра IOPORT1 должен быть установлен, чтобы позволить внешнему мастеру активировать и дезактивировать ведомого. Если требуется постоянное активное состояние ведомого, можно записать ноль в указанный бит (не рекомендуется). При использовании модуля SPI в качестве мастера состояние второго бита регистра IOPORT1 не оказывает влияния на работу модуля. Для включения альтернативной функции следует установить бит HSO.0/1_ENA регистра IOC1.

Вывод 3 порта является входом-выходом общего и мультиплицирован с выходом SPW_DOUT модуля SpaceWire. Для включения альтернативной функции следует установить бит SPWD_SEL регистра IOC3.

Вывод 4 порта является входом-выходом общего и мультиплицирован с выходом SPW_SOUT модуля SpaceWire. Для включения альтернативной функции следует установить бит SPWS_SEL регистра IOC3.

Вывод 5 порта одновременно является входом-выходом общего назначения и входом-выходом данных MOSI модуля SPI и мультиплицирован с выходом HSO.1 модуля HSO. При использовании модуля SPI пятый бит регистра IOPORT1 должен быть установлен. Для включения альтернативной функции следует установить бит HSO.0/1_ENA регистра IOC1.

Вывод 6 порта одновременно является входом-выходом общего назначения и входом-выходом данных MISO модуля SPI и мультиплицирован с выходом HSO.2 модуля HSO. При использовании модуля SPI шестой бит регистра IOPORT1 должен быть установлен. Для включения альтернативной функции следует установить бит HSO.2/4_ENA регистра IOC1.

Вывод 7 порта одновременно является входом-выходом общего назначения и входом-выходом синхросигнала SCK модуля SPI и мультиплицирован с выходом HSO.4 модуля HSO. При использовании модуля SPI седьмой бит регистра IOPORT1 должен быть установлен. Для включения альтернативной функции следует установить бит HSO.2/4_ENA регистра IOC1.

Порт 2

Два вывода порта (0 и 5) являются выходами, четыре вывода порта (с 1 по 4) являются входами и оставшиеся два – входами/выходами. Получать информацию о состоянии выводов порта, а также управлять их состоянием можно посредством регистра IOPORT2, но следует помнить, что выводы-входы не могут быть запрограммированы, а состояние выводов-выходов не может быть прочитано.

Вывод 0 порта мультиплицирован с выходом TXD0 модуля UART0. По умолчанию вывод функционирует как выход общего назначения. Для включения альтернативной функции следует установить бит TXD_SEL регистра IOC1.

Вывод 1 порта является входом общего назначения, а также является выводом RXD0 модуля UART0 (при приеме данных функционирует как обычный вход, при передаче – как выход с открытым стоком).

Вывод 2 порта одновременно является входом общего назначения, входом внешнего прерывания EXTINT и входом HSI.0 модуля HSI.

Вывод 3 порта одновременно является входом общего назначения, входом сигнала синхронизации для таймера 2 и входом HSI.1 модуля HSI, а также выводом SCL модуля I2C (при приеме синхросигнала функционирует как обычный вход, при передаче – как выход с открытым стоком).

Вывод 4 порта одновременно является входом общего назначения, входом сигнала сброса для таймера 2 и входом HSI.2 модуля HSI, а также функционирует как вывод RXD1 модуля UART1 (обычный вход-выход).

Вывод 5 порта одновременно является выходом общего назначения и выходом HSO.5 модуля HSO (объединены по И) и мультиплицирован с выходом PWM.0 модуля ШИМ. Для включения альтернативной функции следует установить бит PWM_SEL регистра IOC1.

Вывод 6 порта – квазидвунаправленный вывод. Одновременно является входом-выходом общего назначения и входом сигнала задания направления счета для таймера 2. Вывод мультиплицирован с выходом TXD1 модуля UART1. Для включения альтернативной функции следует установить бит TXD_SEL регистра IOC1.

Вывод 7 порта – квазидвунаправленный вывод. Одновременно является входом -выходом общего назначения, входом сигнала захвата состояния таймера 2, входом HSI.3 модуля HSI, а также выводом SDA модуля I2C (при приеме синхросигнала функционирует как обычный вход, при передаче – как выход с открытым стоком).

Порты 3 и 4

Выводы портов 3 и 4 представляют собой двунаправленные выводы и используются как системная шина адрес/данные для работы с внешней памятью. Во время системного сброса на выводе EA# должен удерживаться низкий уровень сигнала, который конфигурирует микроконтроллер на режим работы с внешней памятью.

Младший байт адреса/данных передается через порт 3, старший – через порт 4.

Если используются подтягивающие (до уровня логической единицы) резисторы, то в случае считывания кода команды из несуществующей ячейки памяти на шине устанавливается значение FFFFh, что в свою очередь приводит к выборке значения FFh и дальнейшему сбросу микроконтроллера. Выборка команды из несуществующей ячейки памяти может указывать как на программную, так и на аппаратную ошибку, и сброс устройства в этом случае предотвращает дальнейшую некорректную работу.

Аппаратное подключение к квазидвунаправленным портам

Когда выводы квазидвунаправленных портов используются как входы, то уровень сигнала на них задается внешним устройством. В тоже время состояние битов регистра порта также оказывает влияние на соответствующие выводы. Если внешнее устройство удерживает на выводе высокий уровень сигнала, а в соответствующий этому выводу бит регистра порта записывается ноль, то возникает высокий ток короткого замыкания.

Решение этой проблемы может быть как программным, так и аппаратным. Программно достаточно не записывать нули в соответствующие биты регистра порта. Аппаратное решение заключается в том, чтобы последовательно с выводом соединить резистор в 1 кОм.

7.2 Программирование портов

После сброса микроконтроллера каждый вывод порта переходит в режим, который назначен по умолчанию (в таблице 2.1 в колонке «Обозначение вывода» основная функция указана левее, а альтернативная – правее, соответственно, в колонке «функциональное назначение» основная функция прописана в первой строке). Изменить режим работы можно, если включить альтернативную функцию. Так, например, вывод 1 порта 2 по умолчанию является входом, но при альтернативном функционировании в качестве вывода RXD0 модуля UART0 становится двунаправленным выводом с открытым стоком.

Большинство выводов микроконтроллера выполняют основную и альтернативную функцию одновременно. Например, выводы 5, 6 и 7 порта 1, которые являются выводами общего назначения и выводами модуля SPI или выводы портов 0 и 5, которые являются входами общего назначения и входами АЦП.

Все выводы портов микроконтроллера программируются посредством регистров портов. В таблице 7.1 указаны выводы микроконтроллера, управление которыми дополнительно задействует регистры IOC1 и IOC3 (дополнительно см. таблицу 2.1).

Таблица 7.1 – Выводы, дополнительно контролируемые регистром IOC1

Вывод порта	Бит регистра IOC1	Состояние бита	Режим работы вывода порта
P1.2 P1.5	HSO0/1_ENA	0	Выводы общего назначения и выводы модуля SPI
		1	Выходы модуля HSO
P1.6 P1.7	HSO2/4_ENA	0	Выводы общего назначения и выводы модуля SPI
		1	Выходы модуля HSO
P2.0 P2.6	TXD_SEL	0	Выходы общего назначения
		1	Выходы модулей UART0 и UART1
P2.5	PWM_SEL	0	Вывод общего назначения и вывод HSO.5
		1	Выход модуля ШИМ

Таблица 7.2 – Выводы, дополнительно контролируемые регистром IOC3

Вывод порта	Бит регистра IOC3	Состояние бита	Режим работы вывода порта
P1.3	SPWD_SEL	0	Вывод общего назначения
		1	Вывод модуля SpaceWire
P1.4	SPWS_SEL	0	Вывод общего назначения
		1	Вывод модуля SpaceWire
CLOCK OUT	CLKOUT DIS	1	Вывод системного тактового сигнала всегда разрешен. Бит не може быть сброшен

В таблице 7.3 указано состояние выводов во время и после сброса.

Таблица 7.3 – Состояние выводов во время и после сброса

Название вывода	Состояние вывода во время сброса	Состояние вывода после сброса
P0.0 – P0.7	Входы	Входы
P5.0 – P5.7	Входы	Входы
P1.0 – P1.7	«Pull-up»	«Pull-up» (I_{IL1})
P2.0	«Pull-up»	Выход
P2.1	Вход	Вход
P2.2	Вход	Вход
P2.3	Вход	Вход
P2.4	Вход	Вход
P2.5	«Pull-down»	Выход
P2.6 – P2.7	«Pull-up»	«Pull-up» (I_{IL1})
P3.0 – P4.7	«Pull-up»	Шина адрес/данные
HSO.3	Выход	Выход
PWM.1	Выход	Выход
PWM.2	Выход	Выход
M_TXD0	Выход	Выход
M_TXD1	Выход	Выход
M_TXDN0	Выход	Выход
M_TXDN1	Выход	Выход
M_RXD0	Вход	Вход
M_RXD1	Вход	Вход
M_RXDN0	Вход	Вход
M_RXDN1	Вход	Вход
ALE	«Pull-up»	Выход
BHE#	«Pull-up»	Выход
BW	Вход	Вход
CLKOUT	Выход	Выход
EA#	Вход	Вход
INST	«Pull-up»	Выход
NMI	«Pull-down» (I_{IH2})	«Pull-down» (I_{IH2})
RD#	«Pull-up»	Выход
READY	Вход	Вход
RESET#	«Pull-up» (I_{IL2})	«Pull-up» (I_{IL2})
WR#	«Pull-up»	Выход
VPR	Вход	Вход

8 Управление синхронизацией периферийных устройств

Микроконтроллер имеет функцию управления синхронизацией периферийных устройств, а также возможность дополнительного деления на два частоты внешнего синхросигнала. Если какое-либо периферийное устройство не используется, его можно отключить, тем самым уменьшив общее потребление энергии.

Управление синхронизацией осуществляется посредством регистра CLKC. Каждому периферийному устройству (за исключением модуля SpaceWire, который не может быть отключен) соответствует один бит разрешения. Запись нуля в бит отключает соответствующее устройство. Для включения – следует снова установить бит.

По умолчанию частота внешнего тактового сигнала делится на два. Один машинный такт (цикл) равен одному такту полученного синхросигнала (т. е. двум тактам внешнего тактового сигнала Fosc). Для включения дополнительного делителя на два используется бит SLOW регистра CLKC.

Структурная схема блока синхронизации представлена на рисунке 8.1.

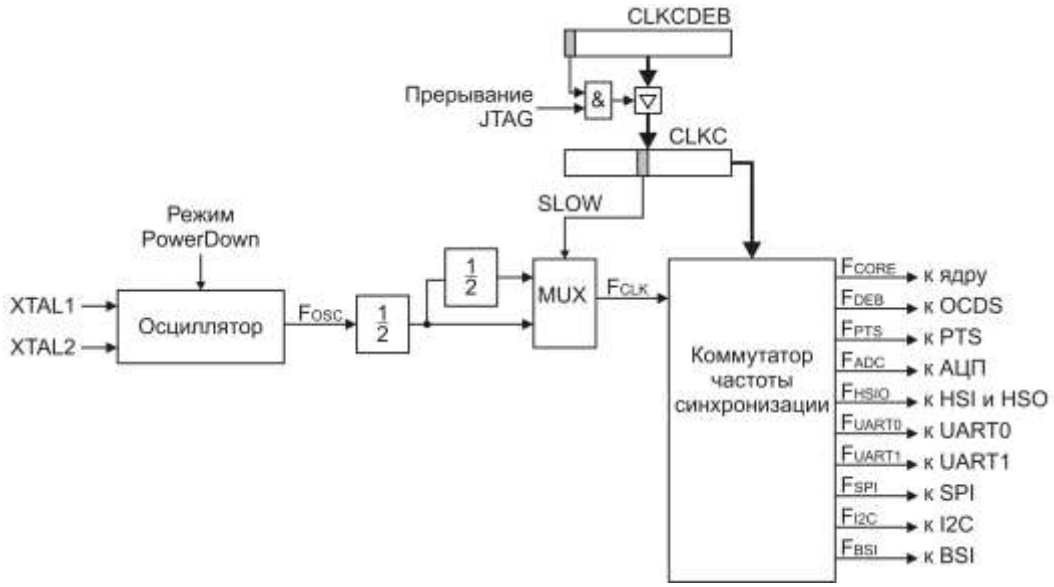


Рисунок 8.1 – Структурная схема синхронизации

Дополнительный регистр CLKCDEB используется во время отладки (при работе с модулем JTAG) и содержит значение, загружаемое в регистр CLKC по прерыванию JTAG. Его структура идентична структуре регистра CLKC, за исключением бита 15, который является битом разрешения загрузки (не сбрасывается аппаратно!). Такая структура позволяет отключать синхронизацию тех устройств, регистры которых нужно сохранить в неизменном виде. Следует помнить, что перезагрузка значения регистра CLKC в регистр CLKCDEB происходит только в момент прерывания JTAG. Тем не менее, оба регистра всегда доступны для записи и чтения. Регистры отключенного устройства не доступны для записи, а при чтении этих регистров будет выдаваться последнее, записанное до отключения, значение.

Время включения/отключения всех периферийных устройств составляет один машинный такт. Исключением является включение АЦП.

Включение/выключение режима дополнительного деления частоты битом SLOW регистра CLKC составляет один машинный такт.

9 Последовательные порты UART0 и UART1

Микроконтроллер содержит два идентичных расширенных универсальных асинхронных приемопередатчика типа UART с поддержкой обнаружения ошибок посылки. Через приемопередатчики осуществляется прием и передача информации, представленной последовательным кодом (младшими битами вперед), в полном дуплексном режиме обмена.

Условные названия приемопередатчиков – UART0 и UART1. Далее символы «0» и «1», указывающие на принадлежность регистров, выводов и др., к тому или иному приемопередатчику не будут использоваться, если это не затрудняет понимание текста. Дальнейшее описание применимо как к UART0, так и к UART1 вследствие их полной функциональной идентичности.

В состав приемопередатчика UART входят принимающий и передающий сдвиговые регистры (недоступные программно), а также специальные буферные регистры SBUF_TX и SBUF_RX для записи данных для передачи и чтения полученных данных.

Запись байта в регистр SBUF_TX приводит к автоматическому переносу этого байта в сдвиговый регистр передатчика и инициирует начало передачи (некоторые особенности указаны ниже). По окончании передачи полученные данные можно получить, прочитав регистр SBUF_RX. Наличие буферного регистра хранения полученного байта позволяет совмещать чтение ранее принятого байта с приемом очередного. Однако следует помнить, что если к моменту окончания приема очередного байта предыдущий байт не был считан из SBUF_RX, то он будет потерян (перезаписан новым байтом).

Блок UART имеет функцию фиксации ошибок посылки, при использовании которой отслеживаются все стоповые биты. При обнаружении пропущенного бита устанавливается флаг FE в регистре SCONx.

Последовательный порт UART может работать в четырех режимах – синхронном (режим 0) и асинхронных (режимы 1, 2, 3). Режим задается битовым полем SER_MODE регистра SP_CON.

9.1 Синхронный режим работы

Режим 0 (посылка 8 бит)

Синхронный режим, в основном используемый для записи и чтения внешнего параллельно-последовательного регистра. Посылки данных по 8 бит передаются или принимаются младшим битом вперед через вывод RxD микроконтроллера. Тактирование передачи/приема осуществляется синхросигналом, который генерируется блоком UART на выводе TxD (см. рисунок 9.1).

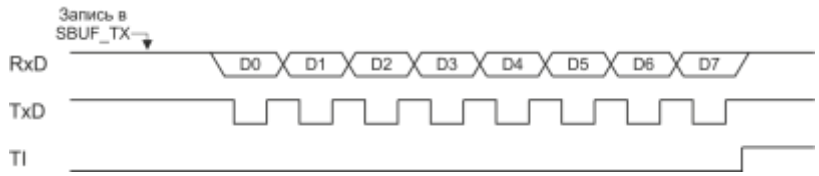


Рисунок 9.1 – Временная диаграмма передачи байта в режиме 0

Для инициализации передачи сначала следует очистить бит REN в регистре SP_CON (разрешение работы вывода RxD как выхода данных), а затем записать байт данных в регистр SBUF_TX. После записи данных в регистр SBUF_TX на аппаратном уровне происходит перезапись этого байта в передающий сдвиговый регистр, запуск генератора синхросигнала и инициализация передачи. С каждым тактом синхросигнала содержимое сдвигового регистра сдвигается вправо (в сторону младших битов) и поступает на вывод RxD. В освобождающиеся биты передающего сдвигового регистра записываются нули.

По окончании передачи восьмого бита устанавливается флаг TI в регистре SP_STAT, и формируется запрос на прерывание. Флаг TI не сбрасывается аппаратно с началом передачи очередного байта. Сброс флага происходит только при чтении регистра SP_STAT. Следует помнить, что в режиме 0 (и только в этом режиме) выход RxD функционирует как выход с открытым стоком. Поэтому на соответствующей линии передачи желательно использовать подтягивающий резистор номиналом 15 кОм (см. рисунок 9.2).

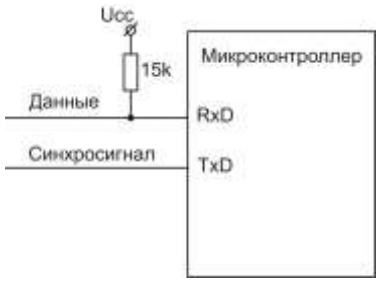


Рисунок 9.2 – Установка подтягивающего резистора

Для инициализации приема следует установить бит REN.

После обнаружения на входе RxD отрицательного импульса (длительностью не менее двух тактов XTAL1) запускается механизм приема данных. По окончании приема устанавливается флаг RI и формируется запрос на прерывание. Полученные данные могут быть прочитаны из регистра SBUF_RX.

Флаг RI не сбрасывается аппаратно. Сброс флага происходит только при чтении регистра SP_STAT.

Примечание – Пока флаг RI установлен, прием очередного байта не возможен. Если установлен бит REN, то сброс флага RI включает механизм приема данных. Для того, чтобы этого не произошло, следует очистить бит REN перед чтением регистра SP_STAT.

9.2 Асинхронные режимы работы

Режим 1 (посылка 10 бит)

В этом асинхронном режиме посылки данных по 10 бит передаются (младшим битом вперед) через вывод TxD микроконтроллера и принимаются (младшим битом вперед) через вывод RxD (см. рисунок 9.3).

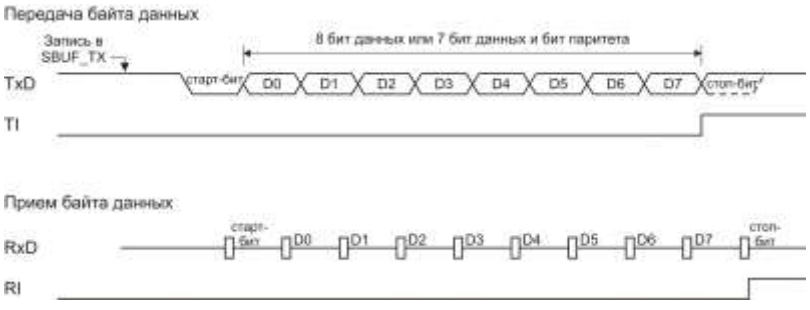


Рисунок 9.3 – Временная диаграмма передачи в режиме 1

Одна посылка состоит из старт-бита (всегда «0»), 8 бит данных и стоп-бита (всегда «1»). Если разрешен контроль по паритету (установлен бит PEN в регистре SP_CON), то посылка состоит из старт-бита, 7 бит данных, бита паритета (вместо восьмого бита данных) и стоп-бита.

Для инициализации передачи следует записать байт данных в регистр SBUF_TX. После записи байта в регистр SBUF_TX, на аппаратном уровне происходит перезапись

этого байта в передающий сдвиговый регистр, запуск генератора синхросигнала для задания скорости передачи и инициализация передачи. Содержимое сдвигового регистра сдвигается вправо (в сторону младших битов) и поступает на выход TxD. В освобождающиеся биты передающего сдвигового регистра записываются нули. По окончании передачи девятого бита посылки и перед началом передачи стоп-бита устанавливается флаг TI и генерируется прерывание. Передача следующего байта данных начнется только после того, как будет отправлен стоп-бит текущей передачи.

Если бит REN установлен и модуль UART находится в режиме ожидания, то появление возрастающего фронта сигнала на входе RxD вызовет запуск генератора синхросигнала для определения скорости передачи и инициализирования приема. Перед окончанием стоп-бита устанавливается флаг RI и формируется запрос на прерывание.

Значение принятого бита 9 всегда сохраняется в бите RPE/RB8 регистра SP_STAT. В случае, если разрешен контроль по паритету, то состояние бита RPE/RB8 будет зависеть от результата проверки. Полученный байт данных переписывается в регистр SBUF_RX, откуда может быть прочитан.

В режиме 1 тактовый сигнал не передается, поэтому для согласованной работы устройств они должны быть настроены на одну скорость передачи. Скорость передачи задается регистром BAUD_RATE.

Режимы 2 и 3 (посылка 11 бит)

В этом асинхронном режиме посылки данных по 11 бит передаются (младшим битом вперед) через вывод TxD микроконтроллера и принимаются (младшим битом вперед) через вывод RxD (см. рисунок 9.4). Одна посылка состоит из старт-бита (всегда «0»), 8 бит данных, программируемого девятого бита данных и стоп-бита (всегда «1»).

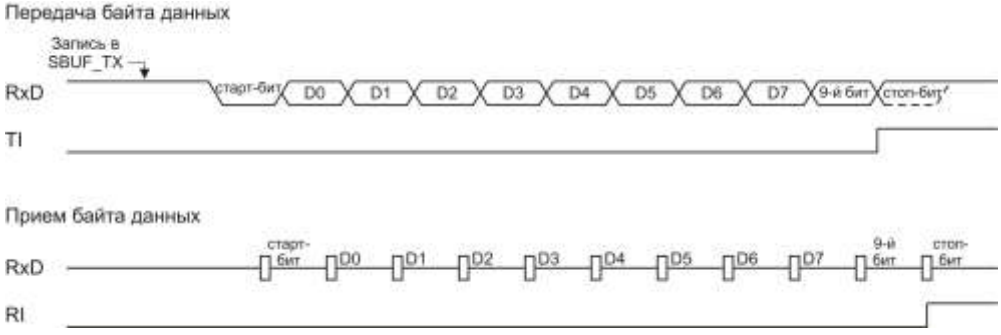


Рисунок 9.4 – Временная диаграмма передачи в режимах 2 и 3

Для инициализации передачи следует записать байт данных в регистр SBUF_TX. После записи байта в регистр SBUF_TX на аппаратном уровне происходит перезапись этого байта в передающий сдвиговый регистр, запуск генератора синхросигнала для задания скорости передачи и инициализация передачи. Содержимое сдвигового регистра сдвигается вправо (в сторону младших битов) и поступает на выход TxD. В освобождающиеся биты передающего сдвигового регистра записываются нули. По окончании передачи десятого бита посылки и перед началом передачи стоп-бита устанавливается флаг TI и генерируется прерывание. Передача следующего байта данных начнется только после того, как будет отправлен стоп-бит текущей передачи.

Десятый передаваемый бит – это бит, значение которого может быть задано посредством бита TB8 регистра SP_CON. Состояние бита TB8 должно быть задано до начала передачи (то есть до записи регистра SBUF_TX). По окончании передачи бит TB8 очищается аппаратно, поэтому его следует устанавливать (если необходимо) перед началом каждой передачи.

Если бит REN установлен и модуль UART находится в режиме ожидания, то появление возрастающего фронта сигнала на входе RxD вызовет запуск генератора

синхросигнала для определения скорости передачи и инициализирование приема. Время установки флага RI и формирование запроса на прерывание идентично режиму 1. Значение десятого принятого бита всегда сохраняется в бите RPE/RB8 регистра SP_STAT. Полученный байт данных переписывается в регистр SBUF_RX, откуда может быть прочитан.

Следует помнить, что в режиме 2 установка флага RI и формирование запроса на прерывание допускается только, если установлен бит TB8, а контроль по паритету не может быть включен.

По формату передачи режим 3 идентичен режиму 2, поэтому временная диаграмма, представленная на рисунке 9.4, справедлива для обоих режимов. В отличие от режима 2, в режиме 3 может быть включен контроль по паритету. Если бит разрешения контроля по паритету (PEN) регистра SP_CON очищен, то значение десятого передаваемого бита задается состоянием бита TB8. Если бит PEN установлен, то десятый передаваемый бит является битом паритета. Поведение флага TI и запроса на прерывание идентично режиму 2.

При приеме, если бит PEN очищен, то значение десятого принятого бита сохраняется в бите RPE/RB8 регистра SP_STAT; если бит PEN установлен, то десятый принятый бит является битом паритета, а бит RPE/RB8 функционирует как флаг ошибки паритета. В отличие от режима 2, в режиме 3 состояние бита RB8 не влияет на установку флага RI и формирование запроса на прерывание. По окончании приема без ошибок полученный байт данных переписывается в регистр SBUF_RX, откуда может быть прочитан.

В режимах 2 и 3 тактовый сигнал не передается, поэтому для согласованной работы устройств они должны быть настроены на одну скорость передачи. Скорость передачи задается регистром BAUD_RATE.

9.3 Скорость приема/передачи

Скорость передачи/приема данных задается посредством регистра BAUD_RATE. Регистр позволяет указать источник синхросигнала для внутреннего генератора блока UART и скорость обмена информацией между устройствами.

Источником синхросигнала может быть вход XTAL1 или вход T2CLK микроконтроллера (см. рисунок 9.5). Если установлен бит SOURCE регистра BAUD_RATE, то источником является вход XTAL1 и значение частоты синхросигнала будет fuart = 0,5 × fxtal1, в противном случае, fuart = ft2clk.

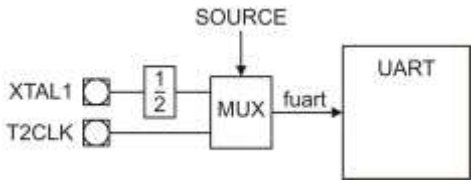


Рисунок 9.5 – Управление выбором источника синхронизации

Значение, записываемое в поле BR_VALUE регистра BAUD_RATE, связано со скоростью передачи данных, выраженной в бодах, следующими соотношениями:

$$BR\_VALUE_{DEC} = \left(\frac{fuart}{Bod} \right) - 1 \quad \text{(для режима 0);} \tag{9.1}$$

$$BR\_VALUE_{DEC} = \left(\frac{fuart}{Bod \times 8} \right) - 1 \quad \text{(для режимов 1, 2 и 3).} \tag{9.2}$$

В формулах 9.1 и 9.2 приняты обозначения:
- BR_VALUE_{DEC} – значение BR_VALUE в десятичном формате;
- fuart – частота синхросигнала, поступающего на блок (Гц);
- Bod – желаемая скорость передачи (бод).

Пример расчета значения BR_VALUE для желаемой скорости передачи данных, равной 9 600 бод, в режиме 1. В качестве источника синхросигнала выбран вход XTAL1 (SOURCE = 1b), на который подается сигнал с частотой 16 МГц.

$$BR\_VALUE_{DEC} = \left(\frac{8 \times 10^6}{9600 \times 8} \right) - 1 = 103.$$

При переводе в шестнадцатеричный формат получается значение 67h. Таким образом, в поле BR_VALUE следует записать значение 0067h. С учетом того, что бит SOURCE установлен, в регистр BAUD_RATE в итоге записывается значение 8067h.

Для режима 0 в поле BR_VALUE может быть записано максимальное значение 3FFFh и минимальное значение 0001h.

Для режимов 1, 2 и 3 в поле BR_VALUE может быть записано максимальное значение 3FFFh и минимальное значение 0000h (источник синхронизации XTAL1) или 0001h (источник синхронизации T2CLK).

Регистр BAUD_RATE является 16-разрядным с последовательной загрузкой. Значение, которое должно быть записано в регистр, разбивается на два байта – старший и младший. Первым по адресу регистра BAUD_RATE записывается значение младшего байта, а вторым, по тому же адресу – значение старшего байта.

9.4 Конфигурирование выводов TxD0, TxD1, RxD0 и RxD1

Функционирование нулевого и первого, четвертого и шестого выводов порта P2 микроконтроллера как входов/выходов модулей UART0 и UART1 соответственно является альтернативной функцией порта. Для включения альтернативной функции следует установить бит TXD_SEL в регистре IOC1. Этот бит одновременно разрешает работу выводов порта P2.0 и P2.6 как TxD0 и TxD1 соответственно. Для выключения альтернативной функции следует очистить бит TXD_SEL.

Альтернативные функции выводов P2.1 и P2.4 (RxD0 и RxD1) включаются битами REN регистров SP_CON0 и SP_CON1 соответственно. Когда эти биты установлены, выводы порта функционируют как входы модулей UART0 и UART1. Очистка бита REN запрещает прием данных на соответствующем выводе и переводит его в вывод общего назначения порта P2.

9.5 Прерывания

Управление прерываниями модулей UART0 и UART1 осуществляется посредством регистров INT_MASK, INT_MASK1, INT_PEND и INT_PEND1.

Если требуется формировать запрос на прерывание по окончании передачи или приема, то следует установить бит TI_RI_MASK в регистре INT_MASK1. При возникновении запроса на прерывание в регистре INT_PEND1 будет аппаратно установлен бит TI_RI_PEND. Также этот бит может быть установлен программно.

Следует помнить, что программная установка флагов RI и TI в регистрах SP_STAT0 и SP_STAT1 не приводит к формированию запросов прерываний, а сброс этих флагов при чтении SP_STAT0 и SPATAT1 не влияет на состояние регистров INT_PEND и INT_PEND1.

10 Контроллер интерфейса SPI

Последовательный периферийный интерфейс SPI предназначен для быстрого синхронного побайтного обмена информацией между микроконтроллером и периферийными устройствами или между двумя микроконтроллерами. Четырехпроводной интерфейс SPI организуется по принципу «Мастер» - «Ведомый», т.е. между одним ведущим и одним или несколькими ведомыми устройствами, не требует дополнительного оборудования для подключения SPI-совместимых устройств и обладает широкими возможностями для конфигурирования.

Модуль SPI может функционировать и как мастер, и как ведомое устройство. В режиме мастера модуль сам генерирует синхросигнал и таким образом управляет скоростью обмена информацией. Все подключенные устройства могут быть только ведомыми и должны выдавать и принимать данные синхронно. В режиме ведомого модуль ожидает появление синхросигнала и при его получении начинает выдавать и принимать данные.

Модуль SPI имеет четыре вывода SCK, MISO, MOSI и SS#, которые объединены логически по «И» с выводами микроконтроллера P1.7, P1.6, P1.5 и P1.2 соответственно. Когда модуль выключен, он не влияет на работу выводов. В свою очередь, для нормального функционирования модуля следует записывать единицы в биты регистра IOPORT1, которые управляют указанными выводами микроконтроллера. Вывод P1.2 является входом выбора устройства SS#. Когда модуль SPI работает в режиме ведомого, подача низкого уровня сигнала на вход P1.2 активирует модуль. Как правило, управление уровнем сигнала на входе SS# осуществляется внешним мастером, который активирует ведомого для обмена данными и отключает его в остальное время. Тем не менее, установить низкий уровень сигнала на входе P1.2 можно записью нуля в соответствующий бит регистра IOPORT1. В режиме мастера состояние вывода P1.2 не оказывает влияния на работу модуля. Посредством выводов P1.7, P1.6, P1.5 и P1.2 модуль SPI может коммутироваться с внешними SPI-совместимыми устройствами.

Модуль SPI имеет два дополнительных буфера данных – один для передаваемых, а второй для принимаемых данных, что позволяет осуществлять непрерывную передачу и прием неограниченного числа байт с непрерывной синхронизацией.

Структура модуля

В состав модуля SPI входят два 8-разрядных сдвиговых регистра для передачи и приема данных, два буфера данных и регистр данных, блок синхронизации, регистры управления и блок коммутации и управления выводами. На рисунке 10.1 показана структурная схема взаимодействия блоков модуля SPI.

Синхронизирующим сигналом модуля SPI является сигнал Fspi, поступающий от коммутатора частоты синхронизации (см. рисунок 8.1). Частота сигнала синхронизации микроконтроллера Fosc, подающегося на вывод XTAL1, аппаратно делится на два. Дополнительное влияние на частоту сигнала синхронизации Fspi оказывает бит SLOW регистра CLKC:

- если бит SLOW сброшен, то fspi = fosc/2;
- если бит SLOW установлен, то fspi = fosc/4.

Это следует учитывать при задании/расчете скорости передачи и приема данных.

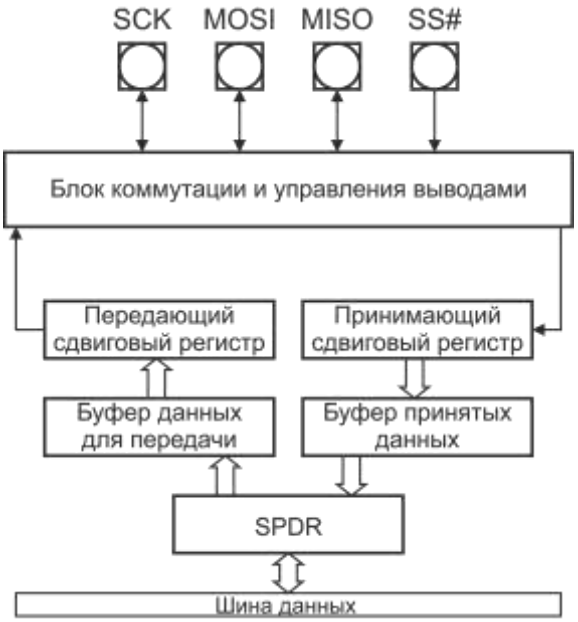


Рисунок 10.1 – Структурная схема взаимодействия блоков модуля SPI

10.1 Управляющие регистры

Управление и контроль состояния модуля SPI, загрузка и чтение полученных данных осуществляются посредством трех регистров: SPCR, SPSR и SPDR.

Регистр SPCR

По умолчанию, модуль SPI выключен. Регистр управления SPCR позволяет задать все основные параметры работы модуля.

Если бит SPIE установлен, то по окончании передачи каждого байта будет генерироваться запрос на прерывание с вектором SERPORT (INT06). Бит SER_MASK регистра INT_MASK является маскирующим битом прерывания. Бит SER_PEND регистра INT_PEND является индикатором сформированного запроса на прерывание.

Бит SPE является битом включения модуля SPI. Этот бит может быть сброшен в любой момент, что приведет к немедленному выключению модуля. Состояние всех битов регистра SPCR должно быть задано одновременно с установкой бита SPE. Если в процессе работы требуется изменить состояние какого-либо бита регистра SPCR (т.е. изменить конфигурацию модуля SPI), то сначала обязательно (!) нужно сбросить бит SPE.

Бит DORD задает порядок передачи и приема битов при обмене байтами. По умолчанию, передача и прием каждого байта осуществляется старшим битом вперед. Установка бита DORD меняет порядок передачи и приема на противоположный – младшим битом вперед.

Бит MSTR позволяет конфигурировать модуль SPI на работу в режиме мастера. По умолчанию, включен режим ведомого.

Бит CPOL задает полярность тактового сигнала. Если бит сброшен, то в отсутствие передачи на линии SCK мастером удерживается низкий уровень сигнала, в противном случае – высокий (см. рисунок 10.2).

Передним всегда считается тот фронт сигнала, который является началом передачи первого бита (т.е. передним может быть и положительный, и отрицательный фронт).

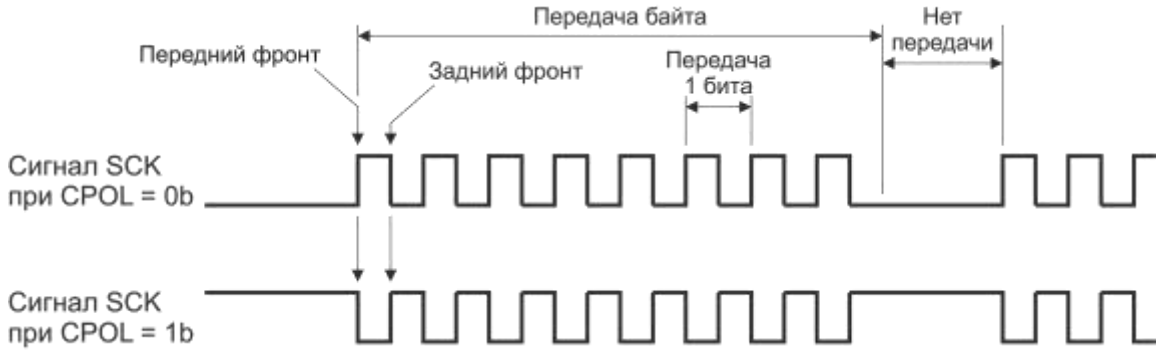


Рисунок 10.2 – Сигнал синхронизации SCK при разных состояниях бита CPOL

Бит CPHA задает порядок считывания и выставления данных. По умолчанию бит CPHA установлен, и выставление данных на линиях MISO и MOSI происходит по переднему фронту сигнала синхронизации, а считывание (выборка) – по заднему. Сброс бита CPHA меняет порядок на обратный. Комбинации битов CPOL и CPHA задают четыре режима обмена данными (см. рисунок 10.3).

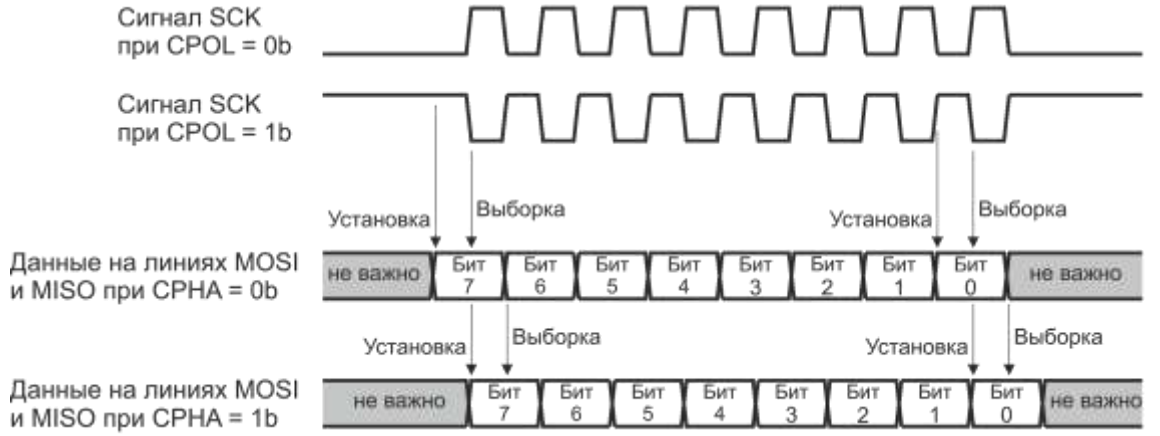


Рисунок 10.3 – Временные диаграммы передач данных в разных режимах

По умолчанию данные передаются старшим битом вперед. Установка и считывание данных выполняется мастером и ведомым одновременно (поэтому на рисунке 10.3 линии MOSI и MISO условно объединены). При CPHA = 0b установка первого бита на линию передачи данных происходит до появления переднего фронта сигнала SCK. К моменту появления переднего фронта сигнала SCK данные должны быть стабильными, чтобы мастер и ведомый смогли произвести корректную выборку. При CPHA = 0b выборка данных производится по переднему фронту сигнала синхронизации, а установка – по заднему. При CPHA = 1b установка первого бита (и последующих бит) на линию передачи данных происходит только тогда, когда появляется передний фронта сигнала SCK. Выборка данных происходит по заднему фронту. Во всех режимах до момента установки первого бита на линию передачи данных и после последней (восьмой) выборки не важно, какой уровень сигнала удерживается на линии.

Оставшиеся два бита SPR1 и SPR0 регистра задают скорость передачи данных. Если модуль SPI функционирует в режиме мастера, то именно он формирует синхросигнал SCK для передачи и приема данных. Синхросигнал формируется на основе сигнала тактовой частоты Fspi (см. рисунок 8.1). Четыре комбинации состояний битов SPR1 и SPR0 позволяют программировать четыре скорости передачи данных (см. таблицу 10.1).

Таблица 10.1 – Комбинации битов SPR1, SPR0

SPR1	SPR0	Частота синхросигнала SCK	
		При SLOW = 0b	При SLOW = 1b
0	0	fspi/4	fspi/2
0	1	fspi/16	fspi/8
1	0	fspi/64	fspi/32
1	1	fspi/128	fspi/64

Если модуль SPI функционирует в режиме ведомого, то состояния битов SPR1 и SPR0 игнорируются. Ведомый передает данные на частоте сигнала SCK, который задается внешним мастером, при этом частота входного сигнала синхронизации fsck не должна превышать частоту fspi более, чем в четыре раза.

Регистр SPDR

Регистр данных SPDR позволяет загружать данные для передачи и считывать принятые данные (обращение по одному адресу). В режиме мастера запись в регистр SPDR автоматически инициализирует передачу и прием байта. В режиме ведомого запись в SPDR только подготавливает данные для передачи и должна производиться до начала передачи, т.е. до появления переднего фронта синхросигнала SCK.

В нормальном режиме работы данные передаются побайтно. И в режиме мастера, и в режиме ведомого запись очередного байта в регистр SPDR во время передачи игнорируется, передача не прерывается, но выставляется флаг WCOL, который является индикатором конфликта процессов записи и передачи.

В режиме буферизации данные могут передаваться непрерывным потоком с непрерывной синхронизацией. Режим включается установкой бита ENH. И в режиме мастера, и в режиме ведомого запись очередного байта в регистр SPDR во время передачи приводит к загрузке этого байта в буфер, передача не прерывается, и выставляется флаг WCOL, который в данном режиме является индикатором того, что очередной байт загружен и готов для передачи. Если по окончании передачи байта обнаруживается установленный флаг WCOL, то байт из буфера переносится в передающий регистр и передача данных продолжается, а флаг WCOL сбрасывается. Буфер может хранить только один байт данных – повторная запись приведет к потере ранее записанного байта.

В обоих режимах (нормальном и буферизации) по окончании передачи и приема каждого байта устанавливается флаг SPIF и формируется запрос на прерывание (если установлен бит SPIE регистра SPCR). Каждый принятый байт сохраняется и доступен для чтения посредством регистра SPDR до тех пор, пока не будет перезаписан очередным принятым байтом.

Регистр SPSR

По умолчанию модуль SPI функционирует в нормальном режиме, и данные передаются побайтно. Для включения режима буферизации следует установить бит ENH.

Бит SPIF является флагом окончания передачи и приема и функционирует, если установлен бит SPIE регистра SPCR. В обоих режимах (нормальном и буферизации) флаг SPIF устанавливается по окончании передачи и приема каждого байта. Для сброса флага необходимо выполнить последовательно действия – прочитать регистр SPSR, а затем прочитать/записать регистр SPDR. При этом между командой чтения регистра SPSR и командой чтения/записи регистра SPDR допускается размещение других команд. При выключении модуля SPI сбросом бита SPE флаг SPIF не сбрасывается. При включении модуля флаг SPIF всегда сбрасывается аппаратно.

Бит WCOL является флагом записи в регистр SPDR во время передачи байта. В нормальном режиме работы флаг сбрасывается одновременно с флагом SPIF. В режиме буферизации данных флаг сбрасывается одновременно с загрузкой очередного байта из буфера в передающий регистр. При выключении модуля SPI флаг WCOL сбрасывается аппаратно.

Бит LDEN является битом разрешения загрузки буфера в режиме буферизации данных. Во время каждого обмена байтами бит LDEN установлен в течение передачи первых четырех бит, а затем сбрасывается. Наиболее подходящее время для записи очередного байта для передачи – это время, когда бит LDEN установлен.

На состояние информационных битов SPIF, WCOL и LDEN оказывают влияния только аппаратные процессы. Чтение регистра SPSR не модифицирует эти биты.

Дополнительным битом управления передачей данных является бит DISSO. В режиме ведомого установка этого бита переводит выход MISO в высокоимпедансное состояние. Таким образом, модуль SPI может принимать данные, но ответной передачи не происходит. Это может использоваться в системах с одним мастером и несколькими параллельно соединенными и постоянно активными ведомыми (каждый со своим адресом). Выходы MISO всех ведомых заблокированы. Мастер передает байт адреса, все ведомые принимают его и к началу следующей передачи ведомый, распознавший свой адрес, сбрасывает бит DISSO, тем самым разрешая передачу данных к мастеру через вывод MISO. Регистр SPSR всегда доступен для записи, и в связи с этим бит DISSO может быть установлен/сброшен в любой момент.

10.2 Особенности функционирования

1 Мастер всегда должен включаться (битом SPE) раньше ведомого. Особенно это важно, если вход SS# ведомого заземлен (т.е. ведомый активен сразу после включения).

2 Деактивация ведомого вследствие появления высокого уровня сигнала на линии SS# сразу останавливает прием и передачу данных ведомым устройством, но не сбрасывает его передающие регистры. Для корректного продолжения работы следует вновь записать данные в регистр SPDR ведомого и только потом активировать его.

3 При выключении модуля SPI текущая передача прерывается, сдвиговые регистры обнуляются.

11 Контроллер интерфейса I2C

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

Функциональные возможности модуля:

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

Особые возможности SMBus:

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

11.1 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 11.1).

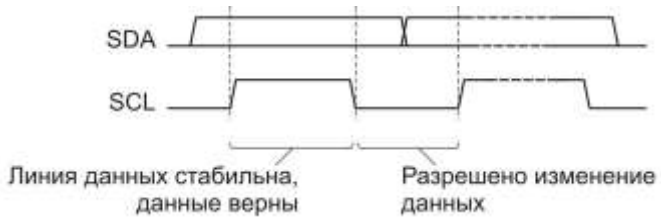


Рисунок 11.1 – Передача бита данных

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 11.2).

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 11.2).

Состояния старта и стопа формирует только мастер.

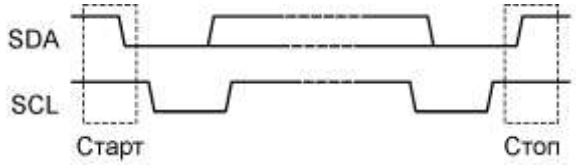


Рисунок 11.2 – Состояния старта и стопа

После того, как сформировано состояние старта, шина считается занятой и другие устройства не должны пытаться управлять ей. Шина считается занятой до тех пор, пока не будет сформировано состояние стопа. В середине передачи может быть сформировано состояние повторного старта, если мастеру нужно обратиться к другому ведомому или если требуется изменение направления передачи данных без потери контроля над шиной (см. рисунок 11.3).

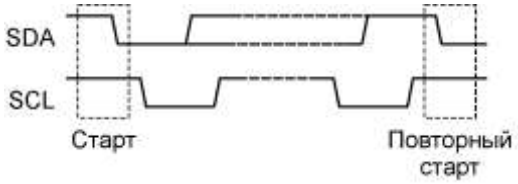


Рисунок 11.3 – Состояние повторного старта

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 11.4 приведен пример арбитража.

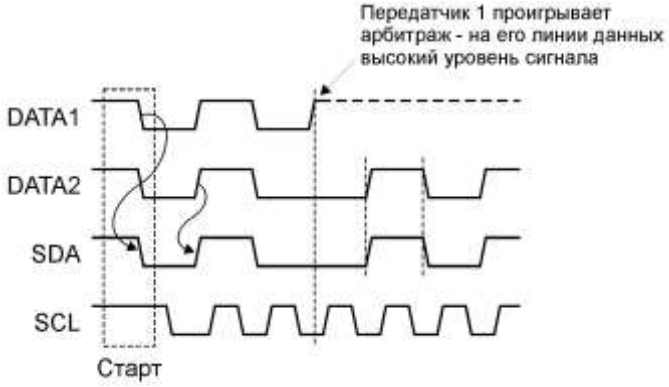


Рисунок 11.4 – Арбитраж на линии SDA

Два устройства передают свои данные DATA1 и DATA2 на линию SDA. В момент времени, когда очередной бит данных DATA1 равен «1», а бит данных DATA2 равен «0», второе устройство выигрывает арбитраж и продолжает передачу своих данных, а первое устройство прекращает передачу.

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

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

Синхронизация

Синхронизация тактовых сигналов разных устройств, подключенных к шине I2C, реализуется в случаях, когда несколько устройств являются мастерами, и выполняется с использованием той особенности, что линия SCL реализована как монтажное «И» линий тактовых сигналов этих устройств. Для примера рассмотрим синхронизацию двух мастеров с линиями тактовых сигналов CLK1 и CLK2 (см. рисунок 11.5).

Рисунок 11.5 – 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 11.5). 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 11.5). 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 11.6).

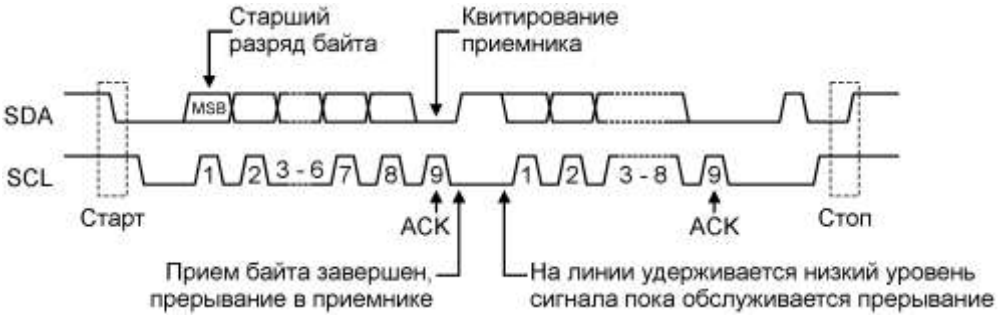


Рисунок 11.6 – Передача данных

Квитирование

Каждый байт посылки должен быть завершен квитированием, т.е. ответом на прием сигнала запроса подтверждения приема (ACK). На рисунках 11.6 и 11.7 показано положение момента квитирования в пределах посылки.

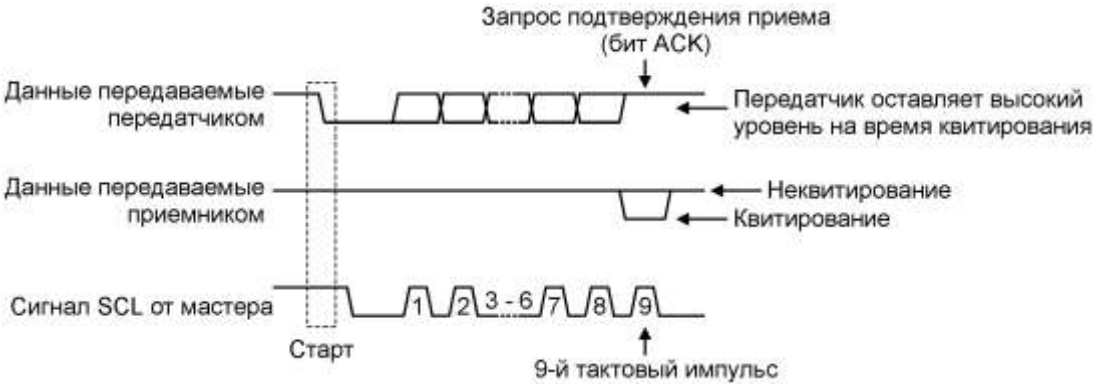


Рисунок 11.7 – Квитирование и неквитирование бита подтверждения ACK

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 11.7). 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 11.8 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 11.8 – Передача данных с 7-битной адресацией

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

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

Протокол SMBus/I2C поддерживает уникальную функцию – распознавание адреса отклика на сигнал предупреждения (Alert Response Address – ARA). В системах с несколькими ведомыми каждое устройство может послать мастеру сигнал предупреждения. Для этого используется дополнительная третья линия ALERT#, физически идентичная линиям SDA и SCL, реализованная по принципу монтажное «И». К этой линии также подключаются все устройства. Когда какому-то ведомому (или нескольким ведомым) необходимо обратиться к мастеру, он (или они) выставляет на линии ALERT# низкий уровень сигнала – это сигнал предупреждения (см. рисунок 11.9).

Рисунок 11.9 – Передача адреса отклика на сигнал предупреждения

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

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

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

Формат передачи данных с 10-битной адресацией

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

Основной идеей формата является передача 10-битного адреса в двух первых байтах, следующих сразу после старта. В первом байте передается значение 1111_0xxb, где «xx» – это два старших бита адреса и бит R/W# (на рисунке 11.10 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 11.10).

Рисунок 11.10 – Передача данных ведомому с 10-битным адресом (для расшифровки обозначений, применяемых на рисунке, следует обратиться к таблице 11.1)

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 11.11).

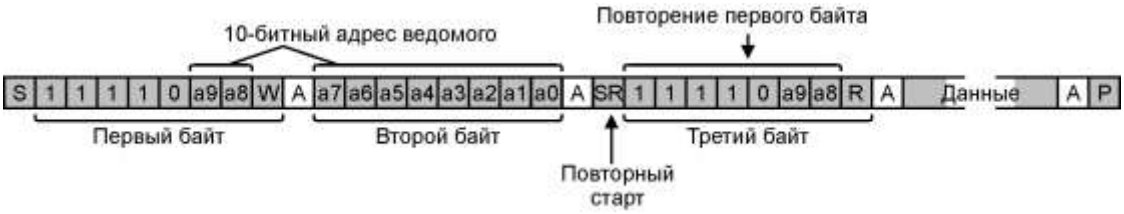


Рисунок 11.11 – Получение данных от ведомого с 10-битным адресом

На рисунках 11.10 и 11.11 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 11.1 с подробными пояснениями.

Таблица 11.1 – Условные обозначения, принятые на рисунках, показывающих содержимое посылок данных или адресов на линии SDA

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

11.2 Функциональное описание

Структурная схема модуля I2C представлена на рисунке 11.12. Далее приводится краткое описание назначения блоков модуля.

Входные и выходные каскады линий SDA и SCL

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

Управление режимом работы и опрос состояния

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

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

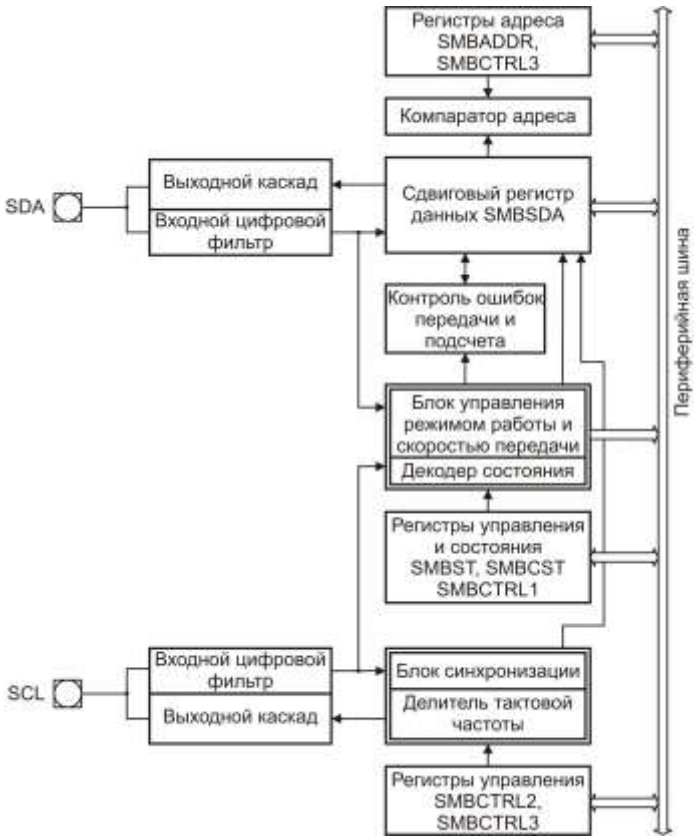


Рисунок 11.12 – Структурная схема модуля I2C

Регистры адреса и компаратор адреса

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

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

Если включен режим 10-битной адресации (одновременно установлены биты SAEN и S10EN регистров SMBADDR и SMBCTRL3, соответственно), компаратор сравнивает старшие пять битов первого полученного байта со значением 1111_0b, а следующие два бита со значением второго и первого битов поля S10ADR регистра SMBCTRL3. Старший бит второго полученного байта сравнивается со значением нулевого бита поля S10ADR, а оставшиеся семь битов – со значением битового поля ADDR регистра SMBADDR (см. рисунок 11.13).

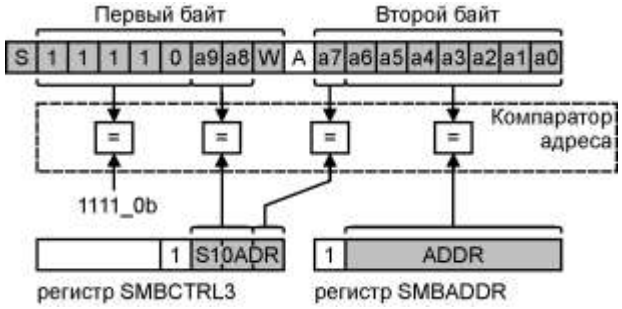


Рисунок 11.13 – Компаратор адреса в режиме 10-битной адресации

Сдвиговый регистр данных

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

Генерация тактового сигнала и синхронизация

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

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

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

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

Определяемое спецификацией протокола SMBus наименьшее время ожидания на линии SCL составляет 25 мс. Если пауза между двумя тактовыми импульсами превысила 25 мс, то устройство должно прервать текущую передачу. Мастер должен сформировать состояние старта в процессе передачи или после ее окончания. Ведомый должен

освободить шину. Устройства, обнаружившие данное состояние, должны восстановить свои соединения и ожидать формирования состояния старта в пределах 10 мс.

Для отслеживания периодов ожиданий на шине в модуле I2C имеется счетчик времени ожидания. Функциональная схема счетчика показана на рисунке 11.14.

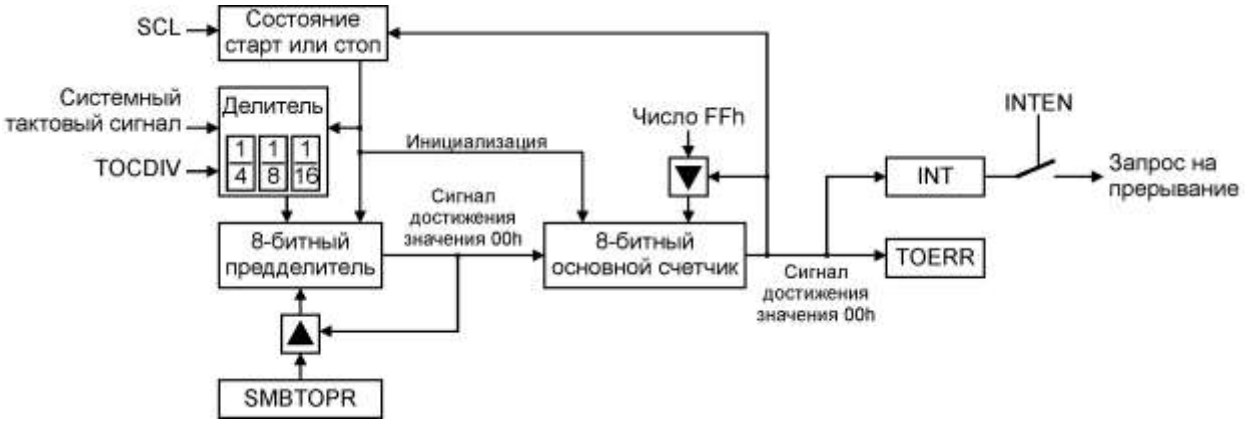


Рисунок 11.14 – Функциональная схема счетчика времени ожидания

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

Период времени ожидания определяется следующим выражением:

$$Тожид = Tosc \times TOCDIV \times (SMBTOPR + 1) \times 256, \quad (11.1)$$

где Tosc – период системного тактового сигнала с частотой fosc.

Арбитраж и обнаружение ошибок на шине

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

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

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

Обнаружение и исправление ошибок на шине

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

- в поле MODE регистра SMBST записывается код ошибки 1Fh;

- генерируется прерывание (если разрешено);
- модуль I2C переходит в режим безадресного ведомого;
- линии SDA и SCL освобождаются.

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

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

Режим IDLE

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

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

11.3 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 кГц 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 МГц 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, поддерживающих режим HS, переключаются в этот режим. Мастер генерирует состояние повторного старта (SR), а затем передает адрес ведомого и бит направления передачи R/W# (см. рисунок 11.15. Расшифровка обозначений, принятых на рисунке, приведена в таблице 11.1).

Рисунок 11.15 – Переход в режим HS и обратно в режим FS

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

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

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

Далее все режимы работы модуля I2C будут рассмотрены подробно.

Инициализация

Для начала работы следует произвести инициализацию:

1 Включить модуль I2C установкой бита ENABLE в регистре SMBCTRL2.

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

3 Если активен режим ведомого, необходимо:

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

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

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

Функционирование

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

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

Передача информации по шине состоит из последовательности различных действий (начало передачи, прием данных и др.). Каждое действие называется состоянием (состояние старта, состояние останова и др.). После того, как то или иное состояние сформировано, его код аппаратно записывается в регистр SMBST в битовое поле MODE и может быть прочитано программно. В таблицах 11.2 и 11.3 приводятся все возможные состояния, их мнемонические обозначения и коды. На квитирование или неквитирование приема указывает запись «ACK» или «NACK», соответственно. Так, например, если мастер отправил байт адреса ведомому, который после получения квитировал прием, то на это будет указывать «ACK», а в поле MODE регистра SMBST будет записан код 04h, соответствующий состоянию с мнемоническим обозначением «MTADPA». Более подробно каждый режим работы модуля I2C будет рассмотрен далее. На рисунках 11.16 –

11.23, поясняющих работу модуля в том или ином режиме, приняты обозначения, расшифровка которых приводится в таблице 11.1. Для получения дополнительной информации и понимания работы модуля I2C можно воспользоваться приложением В.

Таблица 11.2 – Коды функционирования модуля I2C в режиме FS

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

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

Таблица 11.3 – Коды функционирования модуля I2C в режиме HS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра SMBST	ACK/ NACK
Мастер в режиме HS	–	21h	HMTMCOK	Код мастера передан успешно, переход в режим HS	–
		22h	HRSDONE	Сформировано состояние повторного старта	–
		23h	HIDLARL	Потеря арбитража, переход в режим HS безадресного ведомого	–
	Передача	24h	HMTADPA	Отправлен адрес ведомого	ACK
		25h	HMTADNA	Отправлен адрес ведомого	NACK
		26h	HMTDAPA	Отправлен байт данных	ACK
		27h	HMTDANA	Отправлен байт данных	NACK
	Прием	28h	HMRADPA	Отправлен адрес ведомого	ACK
		29h	HMRADNA	Отправлен адрес ведомого	NACK
		2Ah	HMRDAPA	Принят байт данных	ACK
		2Bh	HMRDANA	Принят байт данных	NACK
Ведомый в режиме HS	Прием	30h	HSRADPA	Принят адрес	ACK
		32h	HSRDAPA	Принят байт данных	ACK
		33h	HSRDANA	Принят байт данных	NACK
	Передача	34h	HSTADPA	Принят адрес	ACK
		36h	HSTDAPA	Отправлен байт данных	ACK
		37h	HSTDANA	Отправлен байт данных	NACK
Примечание – Диапазоны значений кодов 2Ch–2Fh и 38h–3Fh, а также коды 20h, 31h и 35h зарезервированы и не доступны для использования. Дополнительная информация находится в приложении 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 SMBCTRL1. Если бит BB в регистре SMBCST сброшен, т.е. шина свободна, будет сгенерировано состояние старта. Если бит BB = 1b, то бит START останется установленным, а состояние старта будет сгенерировано по истечении времени, равного одному такту сигнала тактирования на линии SCL, после освобождения шины.

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

Передача адреса и данных:

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

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

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

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

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

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

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

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

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

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

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

1 Установить бит START.

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

3 Сбросить флаг прерывания INT.

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

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

1 Установить бит STOP в регистре SMBCTRL1.

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

3 Сбросить флаг INT.

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

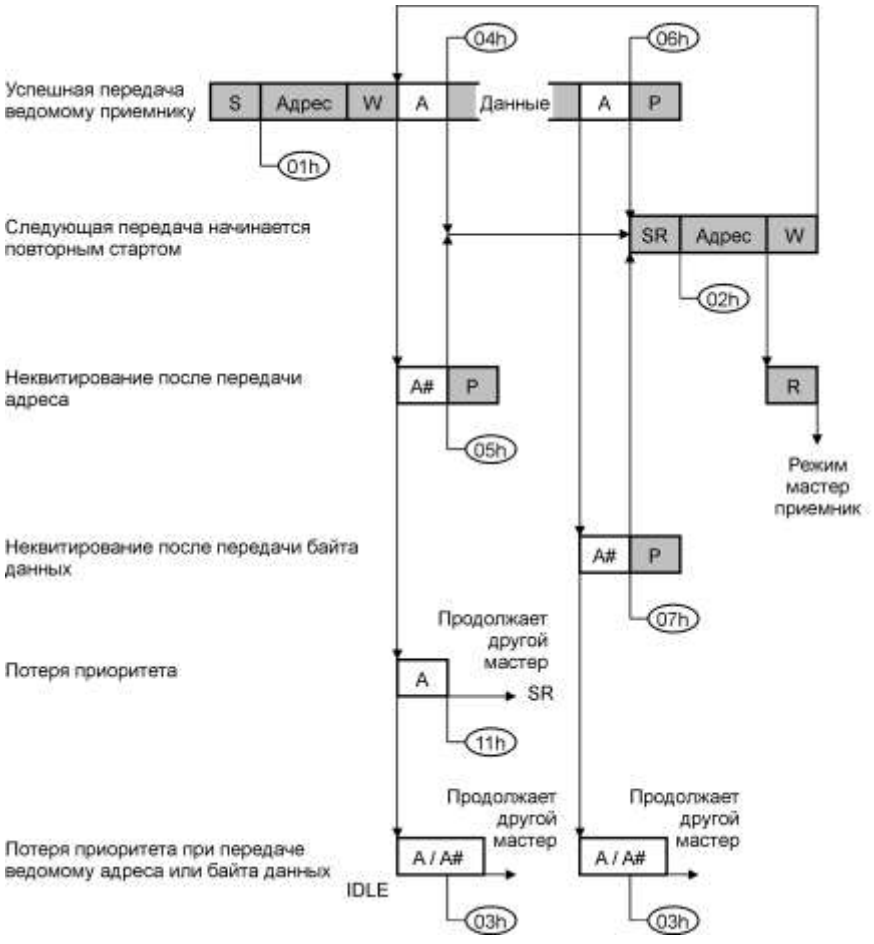


Рисунок 11.16 – Режим FS мастера передатчика

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.16 представлено графическое пояснение к описанию режима.

Режим HS мастера передатчика

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

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

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.17 представлено графическое пояснение к описанию режима.

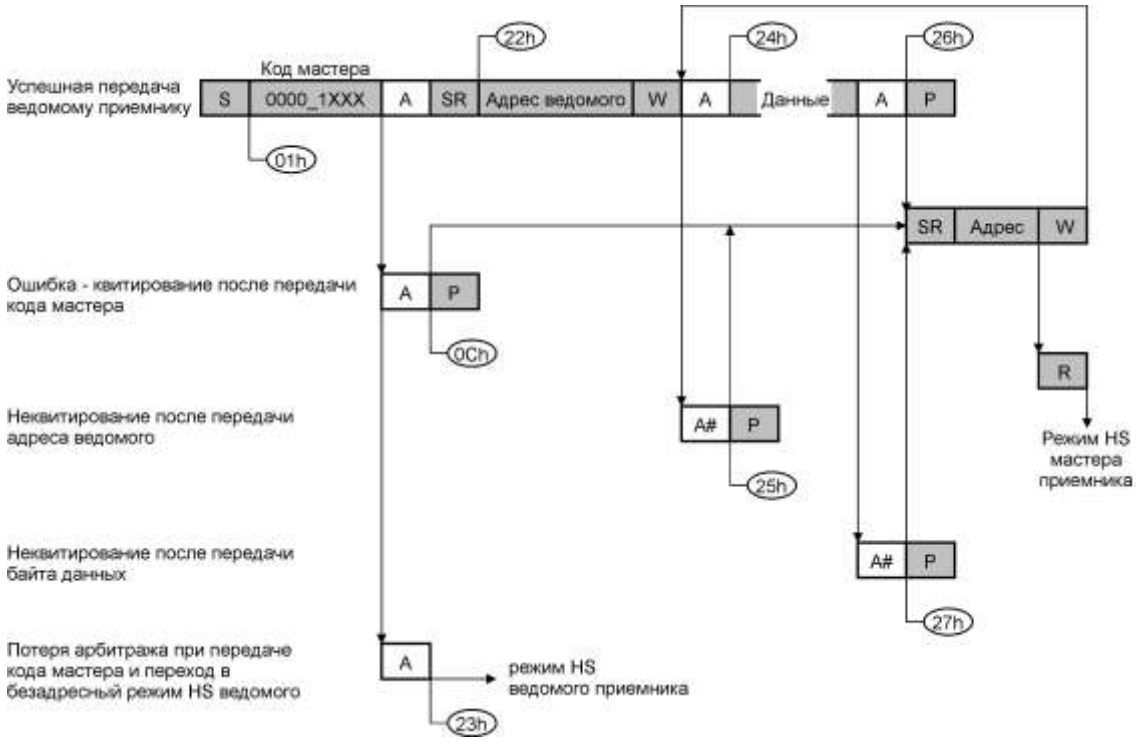


Рисунок 11.17 – Режим HS мастера передатчика

Режим FS мастера приемника

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

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

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

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

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.18 представлено графическое пояснение к описанию режима.

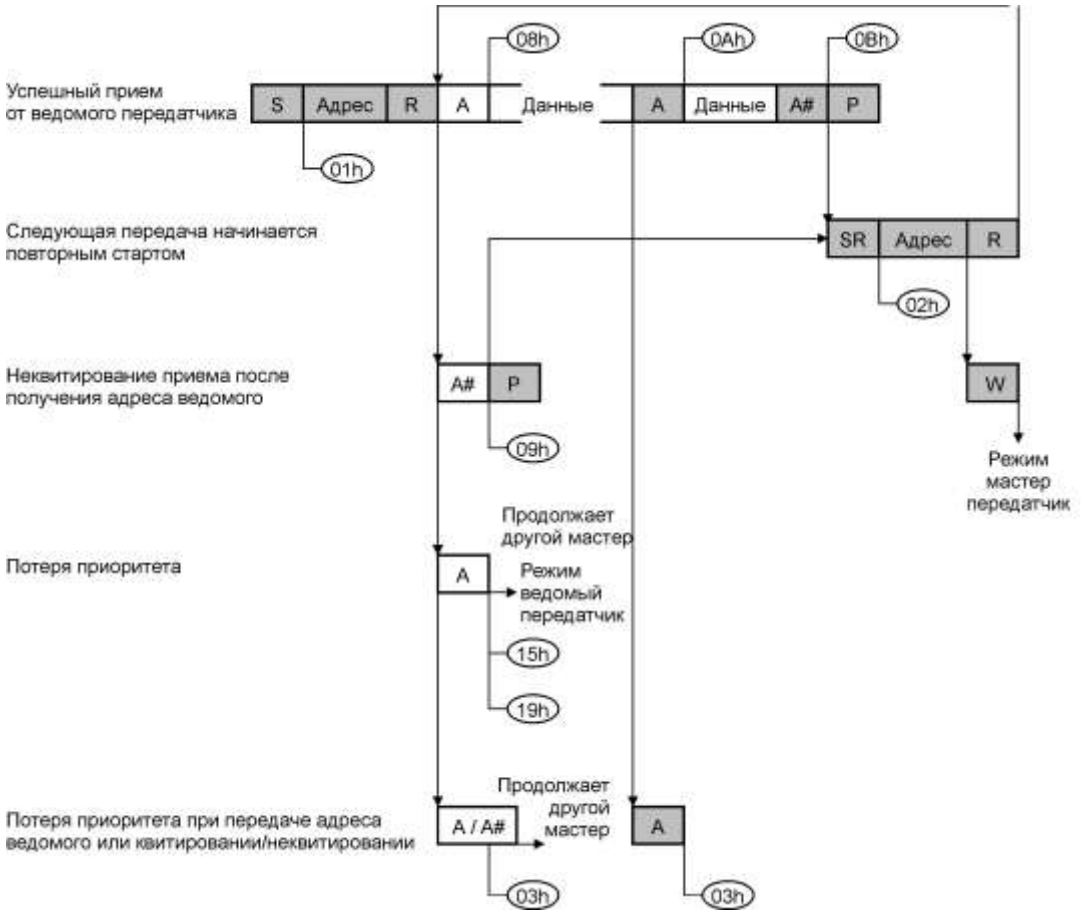


Рисунок 11.18 – Режим FS мастера приемника

Режим HS мастера приемника

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.19 представлено графическое пояснение к описанию режима.

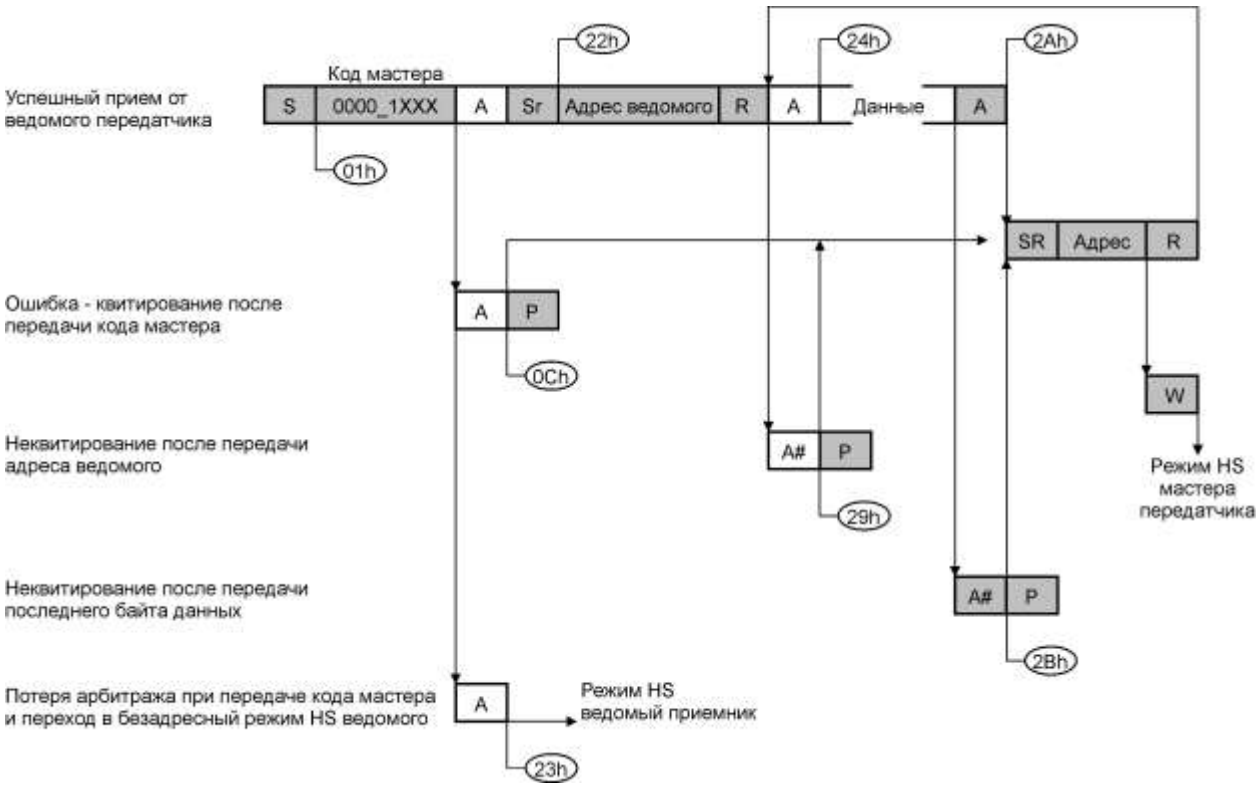


Рисунок 11.19 – Режим HS мастера приемника

Режим FS ведомого приемника

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

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

После получения байта адреса ведомый сравнивает полученный адрес с:

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

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

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

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

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

Последовательность передачи бит в посылке при 10-битной адресации была рассмотрена ранее в подразделе 11.1 настоящего ТО.

Механизм распознавания адреса изложен в подразделе 11.2 настоящего ТО и показан на рисунке 11.13.

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

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

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.20 представлено графическое пояснение к описанию режима.

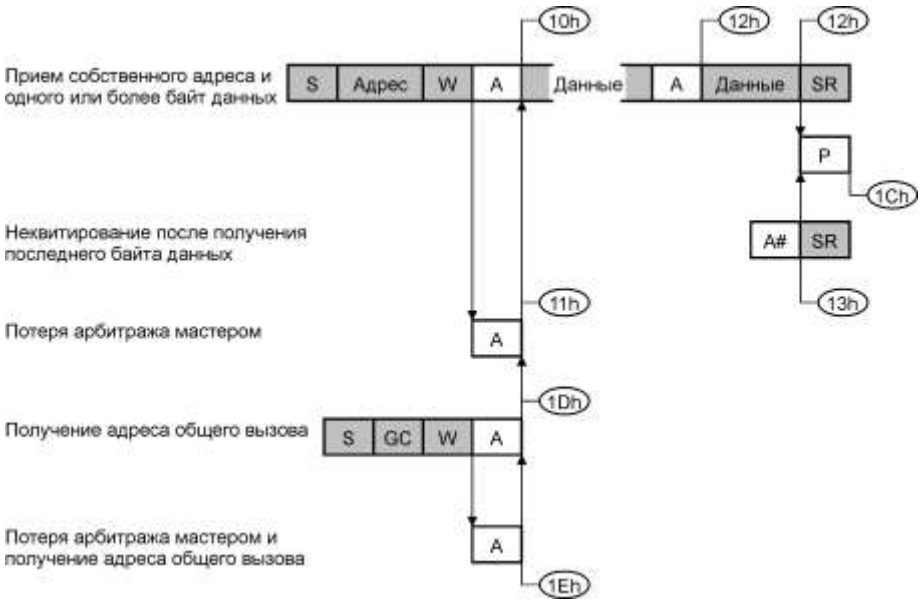


Рисунок 11.20 – Режим FS ведомого приемника

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 11.21 представлено графическое пояснение к описанию режима.

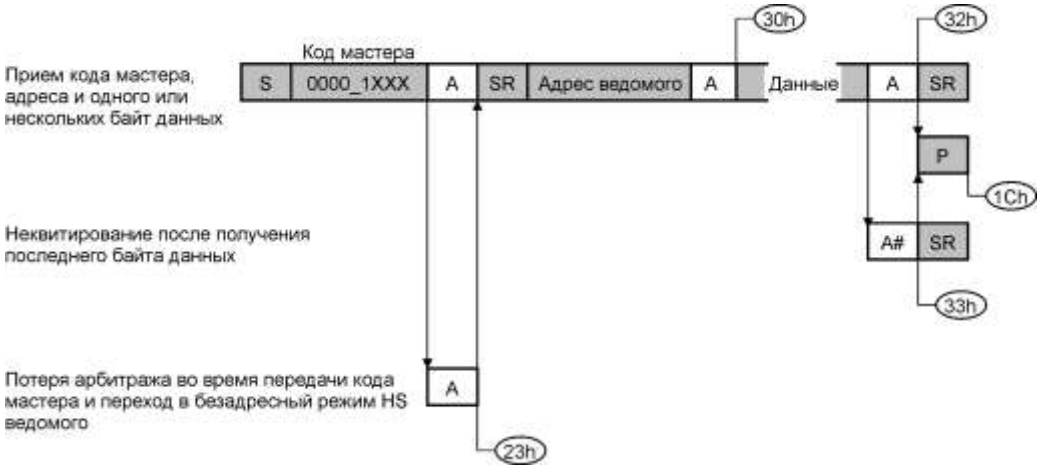


Рисунок 11.21 – Режим HS ведомого приемника

Режим FS ведомого передатчика

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

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

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

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

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

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

Если включен механизм распознавания ошибок, то последний отправленный ведомым передатчиком байт будет байтом CRC. Программа должна «знать» количество байт, посылаемых в пакете данных, и после отправки всех байт устанавливать бит PECNEXT (вместо записи очередных данных в регистр SMBSDA) для того, чтобы в регистр SMBSDA записался байт контрольной суммы.

В модуле I2C поддерживается функция распознавания адреса отклика, который передается мастером шины ко всем ведомым. Ведомое устройство, получившее адрес отклика (0001_100b), переключается в режим передатчика и начинает передавать свой собственный адрес (подробнее – см. подраздел 11.1, «Формат передачи данных с 7-битной адресацией»).

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

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

Дополнительно можно обратиться к приложению В.

На рисунке 11.22 представлено графическое пояснение к описанию режима.

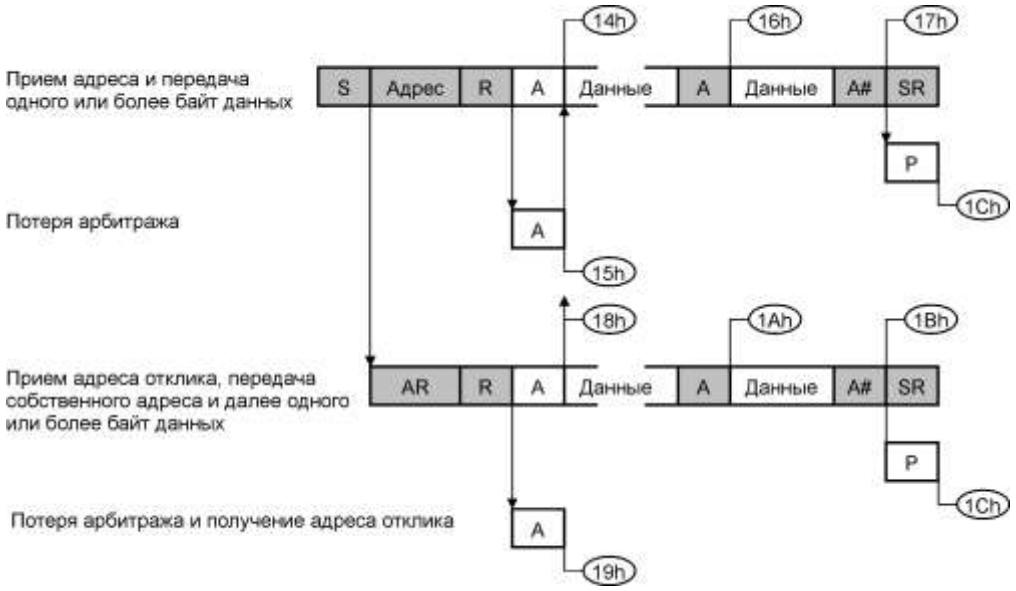


Рисунок 11.22 – Режим FS ведомого передатчика

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 11.23 представлено графическое пояснение к описанию режима.

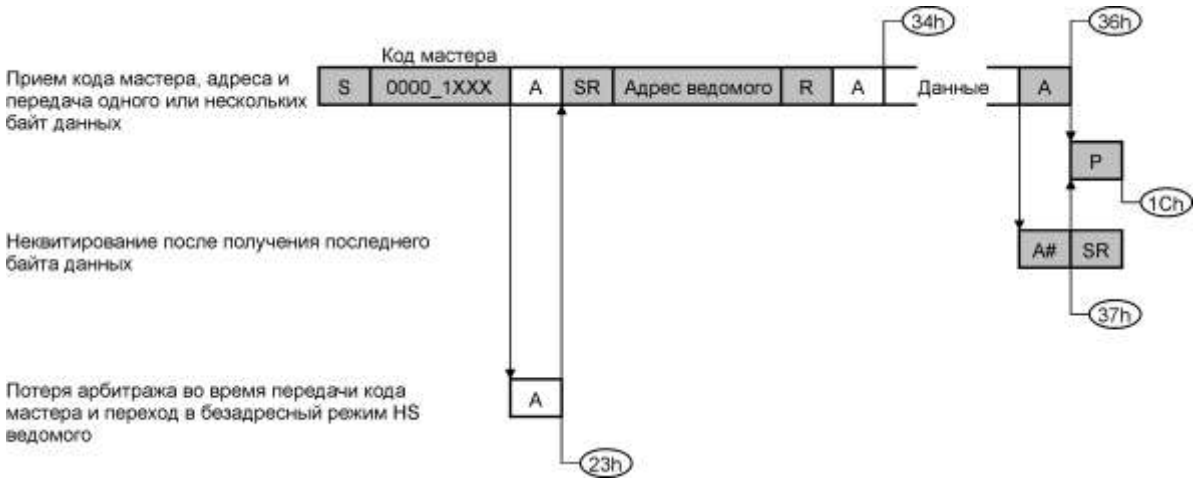


Рисунок 11.23 – Режим HS ведомого передатчика

Дополнительная информация о работе модуля

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

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

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

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

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

- выключить и включить модуль I2C для перевода его в режим безадресного ведомого;
- установить бит START для создания состояния старта;
- проверить, удерживается ли линия SDA в «0» (активное состояние) чтением бита TSDA регистра SMBCST. Если линия активна, отправить одиночный импульс по линии SCL, установив бит TGSCL в регистре SMBCST;
- проверить, что в поле MODE записан код 01b (состояние STDONE), который укажет на то, что состояние старта сформировано. Если нет, то повторять предыдущий и этот шаги до тех пор, пока линия SDA не освободится.

12 Таймеры и модуль высокоскоростного ввода-вывода HSIO

Два 16-разрядных счетчика – таймер 1 и таймер 2 – и 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 (в том числе с ШИМ) и генерировать прерывания.

12.1 Таймер 1

Независимый 16-разрядный однонаправленный счетчик, инкрементирующийся с каждым восьмым тактом синхросигнала Fhsio. Регистр TIMER1 содержит значение счетчика таймера и всегда доступен для записи/чтения. Инициализация таймера 1 происходит после записи в регистр TIMER1 значения, отличного от нуля. Если в дальнейшем таймер функционирует совместно с модулем HSI и/или HSO, то следует уделять большое внимание моменту программирования регистра таймера, поскольку это может вносить изменения во временные соотношения между ожидаемыми и/или запрограммированными событиями. Запущенный счетчик таймера 1 может быть остановлен только сбросом микроконтроллера. Функциональная схема блока таймера 1 показана на рисунке 12.1.

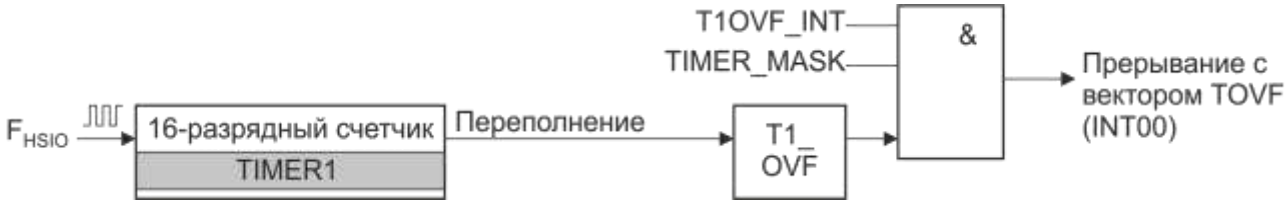


Рисунок 12.1 – Функциональная схема блока таймера 1

Бит T1OVF_INT регистра IOC1 и бит TIMER_MASK регистра INT_MASK управляют формированием запросов на прерывания по переполнению таймера. Индикатором запроса на прерывание является бит TIMER_PEND регистра INT_PEND. Флагом переполнения является бит T1_OVF регистра IOS1 (следует помнить, что чтение регистра IOS1 сбрасывает его биты с пятого по нулевой). Прерыванию соответствует вектор TOVF (INT00).

12.2 Таймер 2

Независимый 16-разрядный реверсивный счетчик с блоком выбора источника синхронизации и регистром захвата. Таймер может функционировать в режиме нормального счета, переключаясь с каждым восьмым тактом синхросигнала, или в режиме скоростного счета – с каждым тактом. Синхросигналом таймера 2 может быть один из трех сигналов: сигнал Fhsio, сигнал с вывода HIS.1, сигнал с вывода T2CLK.

Регистр TIMER2 содержит значение счетчика таймера и всегда доступен для записи/чтения. Инициализация таймера происходит после записи в регистр TIMER2 значения, отличного от нуля. Если в дальнейшем таймер функционирует совместно с модулем HSO, то следует уделять большое внимание моменту программирования регистра таймера, поскольку это может вносить изменения во временные соотношения между запрограммированными событиями. Запущенный счетчик таймера 2 может быть остановлен только сбросом микроконтроллера, в тоже время обнуление счетчика можно контролировать. Счетчик таймера обнуляется аппаратно по переполнению, при возникновении запрограммированного события или может быть обнулен в нужный момент программно. Регистр захвата T2CAPTURE служит регистром сохранения

значения таймера 2, которое находится в его счетчике на момент появления положительного фронта сигнала на выводе P2.7 (T2CAP) микроконтроллера. Сигнал на входе T2CAP должен оставаться без изменений не менее длительности двух тактов сигнала CLOCKOUT. Регистр T2CAP всегда доступен для чтения и записи. Функциональная схема блока таймера 2 показана на рисунке 12.2.

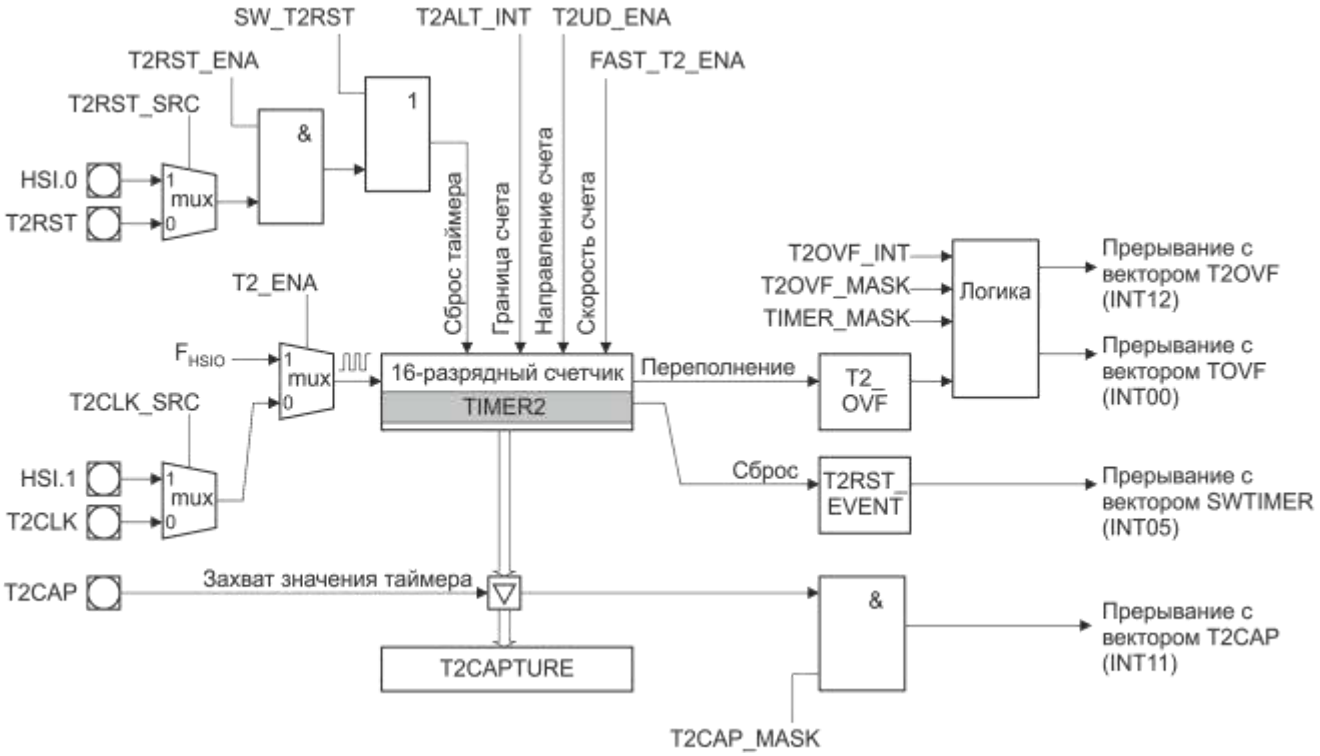


Рисунок 12.2 – Функциональная схема блока таймера 2

Источник синхронизации таймера 2 задается битами T2_ENA и T2CLK_SRC регистров IOC3 и IOC0 соответственно.

Граница переполнения/опустошения таймера (FFFFh/0000h или 7FFFh/8000h), направление и режим счета задаются битами T2ALT_INT, T2UD_ENA и FAST_T2_ENA регистра IOC2.

Биты T2RST_ENA, T2RST_SRC и SW_T2RST регистра IOC0 управляют сбросом таймера. Кроме того, таймер может быть сброшен командой T2RST (код 0Eh), выполняемой модулем HSO.

Бит T2OVF_INT регистра IOC1, бит T2OVF_MASK регистра INT_MASK1 и бит TIMER_MASK регистра INT_MASK управляют формированием запросов на прерывания по переполнению таймера. Бит T2CAP_MASK регистра INT_MASK1 управляет формированием запросов на прерывания по захвату значения таймера. Индикаторами запросов на прерывания являются биты T2OVF_PEND и T2CAP_PEND регистра INT_PEND1 и бит TIMER_PEND регистра INT_PEND. Флагом переполнения является бит T2_OVF регистра IOS1. Флагом сброса является бит T2RST_EVENT регистра IOS2.

Векторы прерываний:

- по переполнению – T2OVF (INT12) и TOVF (INT00);
- по захвату значения – T2CAP (INT11);
- по сбросу – SWTIMER (INT05).

Примечание – Если генерирование прерывания по переполнению таймера 2 разрешено, то вектор прерывания будет выбран в зависимости от состояния битов T2OVF_MASK и TIMER_MASK. Если оба бита установлены, то сначала будет обслужено прерывание с вектором T2OVF, а затем – с вектором TOVF, в соответствии с приоритетом (см. таблицу 6.1 и рисунок 6.5). Таким образом, одно прерывание может быть дважды обслужено двумя разными подпрограммами, что может быть нежелательным. Это следует учитывать при программировании.

Состояния входов микроконтроллера T2CLK, T2UP_DN, T2CAP и T2RST фиксируются в моменты, когда внутренний сигнал синхронизации CLKOUT имеет низкий уровень. Для гарантированного и правильного детектирования состояние сигналов на входах должно оставаться стабильным по крайней мере в течение одного такта сигнала CLOCKOUT. Хотя захват состояний выводов происходит в каждом такте сигнала CLOCKOUT, опрос портовых защелок производится каждые 8 тактов, если таймер 2 работает в режиме нормального счета. Поэтому время реакции на изменение уровня того или иного сигнала на входе зависит от момента этого изменения в границах периода переключения счетчика таймера и может составлять от одного до восьми тактов сигнала CLOCKOUT. В режиме скоростного счета время реакции на изменения сигналов T2CLK, T2UP_DN и T2CAP составляет один такт сигнала CLOCKOUT.

В отличие от других входных сигналов, сигнал T2RST обрабатывается всегда одинаково – только каждый восьмой такт сигнала CLOCKOUT – независимо от режима работы таймера 2.

При одновременной установке сигналов T2CLK, T2CAP и T2RST аппаратная последовательность действий всегда следующая:

- захват значения таймера 2;
- сброс таймера 2 (обнуление счетчика таймера);
- переключение счетчика таймера.

При одновременной установке сигналов T2CLK и T2CAP (T2RST не установлен) сначала всегда производится захват значения таймера 2, а потом переключение счетчика таймера.

12.3 Модуль высокоскоростного ввода HSI

Основным назначением модуля является мониторинг состояния своих выводов HSI.0 – HSI.3 и фиксирование времени обнаружения до восьми запрограммированных событий с использованием таймера 1 (и только его) с возможностью последующей выдачи сохраненной информации. Функциональная схема модуля показана на рисунке 12.3.

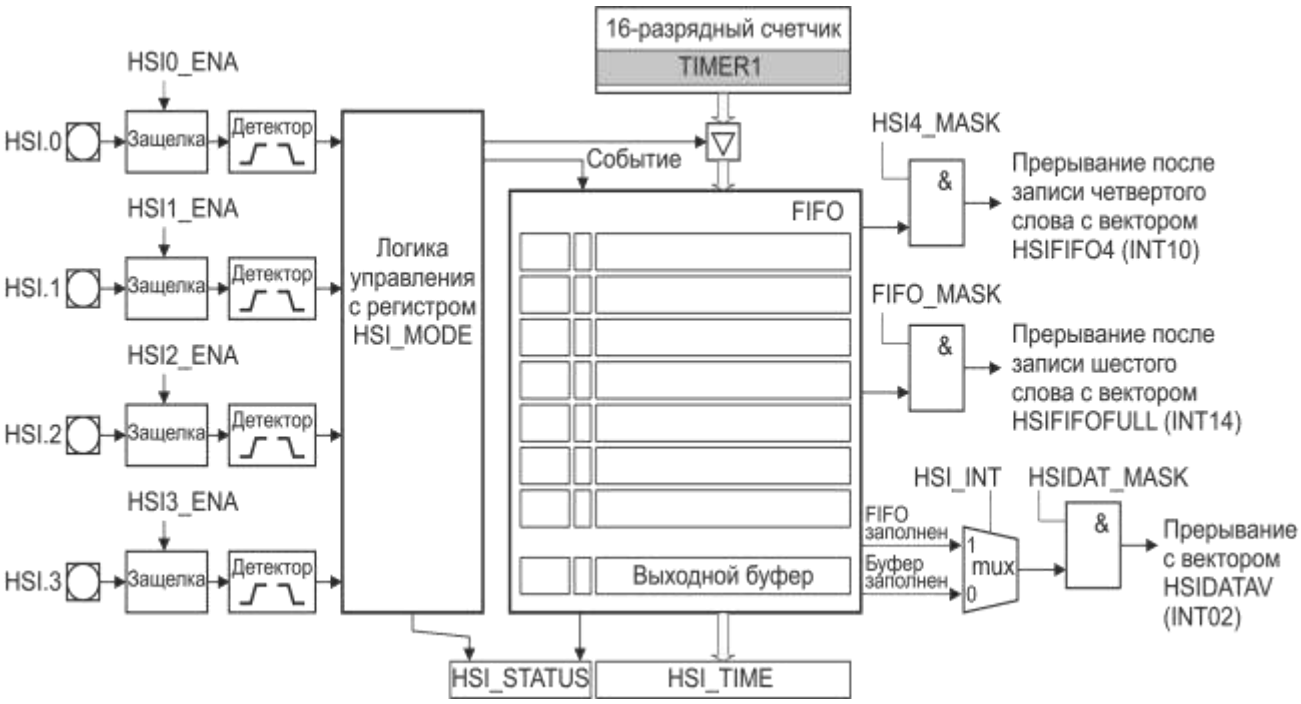


Рисунок 12.3 – Функциональная схема модуля HSI

Функции выводов P2.2, P2.3, P2.4 и P2.7 микроконтроллера как входов HSI.0, HSI.1, HSI.2 и HSI.3 модуля HSI являются дополнительными и требуют включения посредством битов HSI0_ENA, HSI1_ENA, HSI2_ENA и HSI3_ENA регистра IOC0 соответственно. Если вывод запрограммирован как вход HSI, то остальные дополнительные функции не реализуются.

Каждый вход оснащен защелкой, которая с каждым тактом синхросигнала CLOCKOUT фиксирует состояние соответствующего входа, детектором фронта входного сигнала и счетчиком восьми фронтов. Для гарантированного и правильного детектирования длительность стабильного состояния сигнала на входе должна быть не менее длительности одного такта сигнала CLOCKOUT. Таким образом реализованы четыре канала захвата модуля HSI. Для каждого канала независимо может быть задан один из четырех возможных режимов мониторинга состояния соответствующего вывода:

- детектирование каждого восьмого положительного фронта входного сигнала;
- детектирование каждого положительного фронта входного сигнала;
- детектирование каждого отрицательного фронта входного сигнала;
- детектирование каждого фронта входного сигнала.

Режимы задаются посредством регистра HSI_MODE. Как только на выводе обнаруживается запрограммированное событие, в накопительный блок FIFO загружается 16-разрядное значение таймера 1 (время события), бит-индикатор события и три служебных бита. Время загрузки в FIFO составляет один такт сигнала CLOCKOUT. Если два события обнаруживаются одновременно, они записываются с одинаковыми временами.

Если задан режим детектирования каждого восьмого положительного фронта входного сигнала, то активируется счетчик восьми фронтов. Через каждые восемь переключений счетчик подает сигнал на логику управления блока HSI. Особенность счетчика заключается в том, что при смене режима мониторинга счетчик не сбрасывается. Следует помнить, что при повторном включении режима отслеживания восьми фронтов счетчик может иметь ненулевое значение.

Накопительный блок FIFO состоит из запоминающего устройства для семи 20-разрядных расширенных слов и выходного 20-разрядного буфера. Таким образом, заполненный блок FIFO может хранить информацию о восьми событиях. Как только FIFO заполняется полностью, он перестает принимать информацию до тех пор, пока не появится свободное место. Информация о событии, обнаруженном первым, т.е. расширенное слово, записанное первым, находится в выходном буфере, остальные слова – в памяти FIFO в порядке, соответствующем порядку детектирования событий.

По мере заполнения выходного буфера и далее памяти блока FIFO модуль HSI может генерировать прерывания (в скобках указан соответствующий вектор прерывания):

- слово из памяти FIFO (в том числе и первое слово) загружено в выходной буфер или память FIFO заполнена (HSIDATAV); источник определяется битом HSI_INT;
- в память FIFO записано шестое слово, при этом состояние выходного буфера не учитывается (HSIFIFOFULL);
- в память FIFO записано четвертое слово, при этом состояние выходного буфера не учитывается (HSIFIFO4);

В таблице 12.1 представлены комбинации битов регистров IOC1, INT_MASK1 и INT_MASK для программирования прерываний модуля HSI.

В зависимости от того, какие биты управления и/или масок были установлены, при формировании запросов прерываний устанавливаются биты ожидания в регистрах INT_PEND1 и INT_PREND. Регистр IOS1 содержит два флага прерываний модуля HSI – флаг HSI_RDY выходного буфера и флаг FIFO_FULL заполнения памяти FIFO.

Таблица 12.1 – Связь источников прерываний и векторов прерываний

Запись в память FIFO	Запись в выходной буфер	Регистры и биты				Вектор прерывания
		IOC1	INT_MASK1		INT_MASK	
		HSI_INT	FIFO_ MASK	HSI4_ MASK	HSIDAT_ MASK	
–	Да	0	–	–	1	HSIDATAV (INT02)
Шестая	–	1	–	–	1	
Шестая	–	–	1	–	–	HSIFIFOFULL (INT14)
Четвертая	–	–	–	1	–	HSIFIFO4 (INT10)
Примечание – Символ «–» указывает на то, что состояние выходного буфера и состояние бита не важно.						

Чтение информации из блока FIFO происходит через выходной буфер. Начиная с первого записанного слова (которое уже находится в буфере), все остальные слова в том же порядке, в каком были записаны в блок FIFO, последовательно загружаются в буфер и затем становятся доступными для чтения.

Для чтения информации из блока FIFO последовательно выполняются два действия:

1 Чтение регистра HSI_STATUS.

В регистре находятся четыре пары битов – по одной для каждого канала. В паре один бит с приставкой «_EVENT» в названии указывает на номер канала, в котором было зафиксировано событие. Второй бит с приставкой «_STAT» в названии отражает текущее состояние сигнала на входе канала (именно текущее, а не то, что было на

момент обнаружения события). Таким образом, считывая состояние регистра HSI_STATUS можно точно определить, к какому каналу относится слово, находящееся в выходном буфере – соответствующий бит будет установлен в то время, как остальные будут сброшены.

2 Чтение регистра HSI_TIME.

Регистр содержит значение, которое соответствует времени фиксирования события в блоке FIFO.

Сразу после прочтения регистра HSI_TIME очередное слово из памяти FIFO загружается в выходной регистр, и значения в регистрах HSI_STATUS и HSI_TIME обновляются. Поэтому важно соблюдать указанную последовательность чтения регистров.

После того, как хотя бы одно слово будет прочитано из выходного буфера, блок FIFO возобновит фиксацию времени событий до следующего своего заполнения.

12.4 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 HSO.0 – HSO.5, генерировать программные прерывания, сбрасывать таймер 2 и запускать аналого-цифровое преобразование. Модуль HSO позволяет формировать независимые импульсные последовательности, в том числе и с ШИМ. Функциональная схема модуля показана на рисунке 12.4.

Функции выводов P1.2, P1.5, P1.6 и P1.7 микроконтроллера, как выходов HSO.0, HSO.1, HSO.2 и HSO.4 модуля HSO, являются дополнительными и требуют включения посредством битов HSO0/1_ENA и HSO2/4_ENA регистра IOC1 соответственно. Если вывод запрограммирован как выход HSO, то остальные дополнительные функции не реализуются.

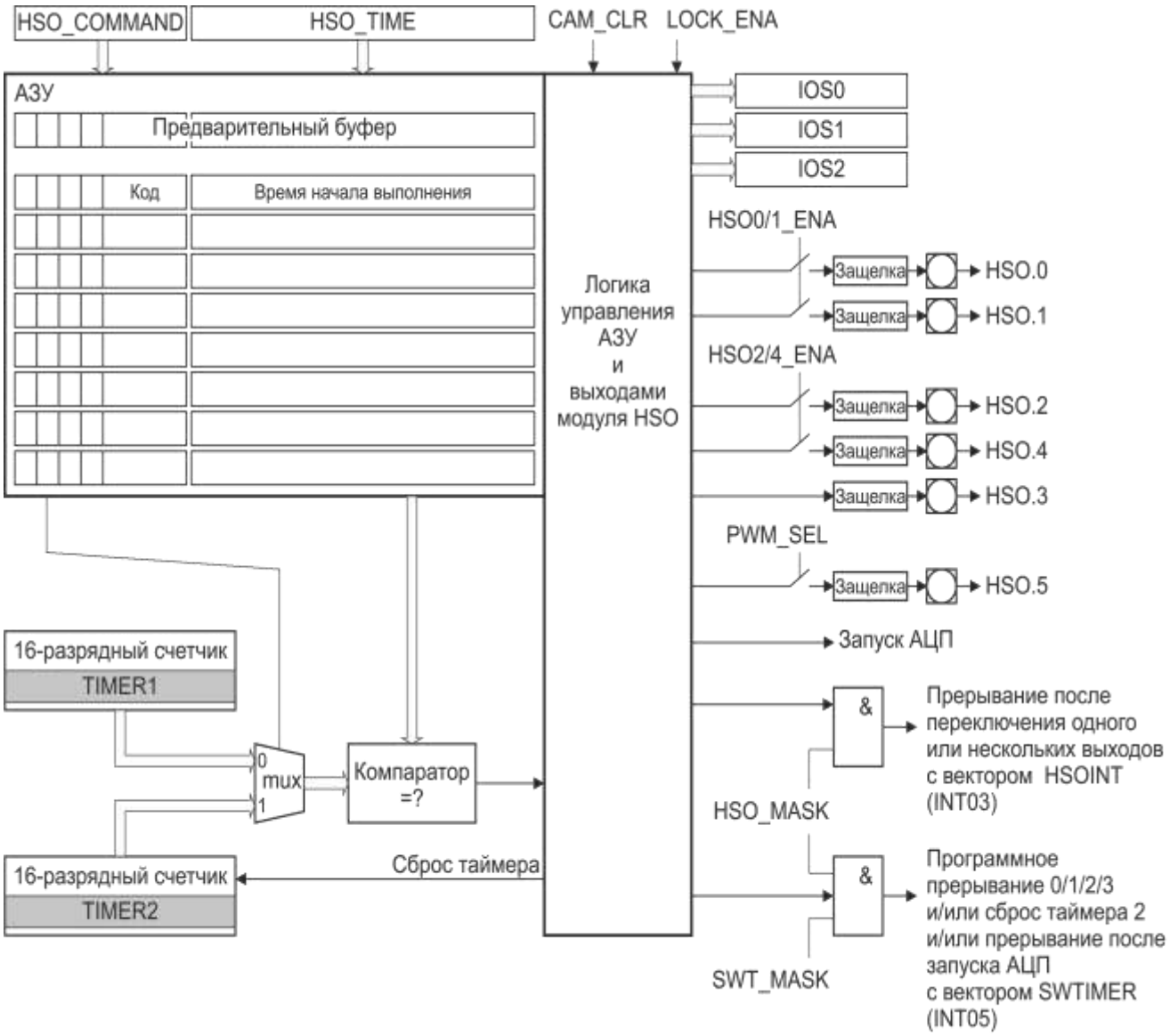


Рисунок 12.4 – Функциональная схема модуля HSO

Выход HSO.3 является независимым выходом микроконтроллера.

Вывод P2.5 микроконтроллера объединен логически по «И» с выходом HSO.5 (включено по умолчанию). Выход PWM.0 модуля ШИМ является дополнительной функцией вывода P2.5, включением которой управляет бит PWM_SEL регистра IOC1. Для того, чтобы функционировал выход HSO.5, бит PWM_SEL должен быть сброшен.

Являются ли указанные выводы микроконтроллера выходами модуля HSO или функционируют как выводы общего назначения, можно определить по состоянию флагов HSO0_STAT, HSO1_STAT, HSO2_STAT, HSO4_STAT и HSO5_STAT регистра IOS0.

Основным блоком модуля HSO является ассоциативное запоминающее устройство (АЗУ) для восьми 24-разрядных расширенных слов с 24-разрядным предварительным буфером хранения.

Каждое слово АЗУ включает в себя 16-разрядное значение времени для задания момента выполнения команды, четыре бита команды и четыре бита параметров.

Для записи каждого слова в АЗУ последовательно выполняются действия:

- проверка состояния флагов HR_STAT и HRCAM_STAT регистра IOS0;
- запись регистра HSO_COMMAND;
- запись регистра HSO_TIME.

1 Проверка состояния флагов HR_STAT и HRCAM_STAT регистра IOS0.

При программировании модуля HSO все слова сначала размещаются в АЗУ, минуя предварительный буфер. В АЗУ последовательно можно записать максимум восемь слов. После того, как АЗУ будет заполнено, установится флаг HRCAM_STAT. Девятое записанное слово автоматически будет размещено в предварительном буфере, после чего установится флаг HR_STAT. При дальнейшей работе каждый раз, когда будет освобождаться ячейка АЗУ, в нее будет переписываться слово из предварительного буфера. Цель проверки указанных флагов заключается в том, чтобы определить, свободен ли буфер или нет, поскольку неконтролируемая запись в буфер (при заполненном АЗУ) может привести к потере ранее записанной и подготовленной для загрузки в АЗУ информации.

2 Запись регистра HSO_COMMAND.

Если проверка флагов HR_STAT и HRCAM_STAT показала, что хотя бы одна ячейка АЗУ свободна и/или свободен предварительный буфер, то можно загрузить очередное слово. В первую очередь записываются параметры команды и ее код.

Бит CAM_LOCK позволяет зафиксировать команду в АЗУ, в результате чего команда будет выполняться столько раз, сколько подключенный таймер будет достигать значения времени выполнения. Незафиксированная команда будет выполнена всего один раз, после чего ячейка АЗУ, в которой она была размещена, полностью будет очищена и готова для загрузки нового слова. Для глобального разрешения фиксирования всех команд следует установить бит LOCK_ENA регистра IOC2.

В зависимости от состояния бита TIMER_SEL, может быть подключен как таймер 1, так и таймер 2.

В зависимости от выполняемой команды (команд) модуль HSO может оказывать влияние на свои выходы HSO.0 – HSO.5. Влияние заключается в том, что модуль может устанавливать на этих выходах как высокий, так и низкий уровень сигнала. Бит PIN_CMD задает уровень сигнала, который будет установлен на выходе/выходах в результате выполнения команды.

Установленный бит HSOINT_ENA разрешает прерывания, генерируемые ячейкой АЗУ. Прерывания маскируются битом HSO_MASK регистра INT_MASK. Индикатором сгенерированного запроса на прерывание является бит HSO_PEND регистра INT_PEND.

Оставшиеся четыре бита – поле CMD_TAG – задают код одной из 14 доступных команды.

Восемь команд с кодами от 0h до 7h и команда с кодом Ch управляют выходами HSO.0 – HSO.5. Результат выполнения любой из девяти команд – переключение соответствующего выхода (или выходов) в состояние, заданное битом PIN_CMD. Выполнение команды сопровождается генерированием прерывания с вектором HSOINT (INT03). Поскольку для всех девяти команд выделен только один вектор прерывания, то при обслуживании прерывания нужно считывать регистр IOS2, чтобы определить источник/источники прерывания. Комбинация состояний битов HSO0_EVENT, HSO1_EVENT, HSO2_EVENT, HSO3_EVENT, HSO4_EVENT и HSO5_EVENT будет указывать на источник/источники прерывания.

Четыре команды с кодами от 8h до Bh позволяют формировать программные прерывания. Результат выполнения любой из четырех команд – формирование запроса на прерывание с вектором SWTIMER (INT05). Поскольку для всех четырех команд выделен только один вектор прерывания, то при обслуживании прерывания нужно считывать регистр IOS1, чтобы определить источник/источники прерывания. Комбинация состояний битов SWTF0, SWTF1, SWTF2 и SWTF3 будет указывать на источник/источники прерывания.

Результат выполнения команды с кодом Eh – сброс таймера 2 и формирование запроса на прерывание с вектором SWTIMER (INT05). Для подтверждения источника прерывания следует проверять состояние флага T2RST_EVENT регистра IOS2.

Результат выполнения команды с кодом Fh – запуск аналого-цифрового преобразования и формирование запроса на прерывание с вектором SWTIMER (INT05). Для подтверждения источника прерывания следует проверять состояние флага AD_EVENT регистра IOS2.

3 Запись регистра HSO_TIME.

Запись в регистр завершает процесс заполнения одной ячейки (одного 24-разрядного слова) памяти АЗУ. Регистр HSO_TIME позволяет задать момент времени начала выполнения команды. Каждой команде в АЗУ соответствует определенное значение времени, при этом несколько команд могут иметь одинаковые значения времени.

Работа блока HSO начинается сразу после запуска таймера. Все значения времени, записанные в АЗУ, сравниваются с таймером одновременно в момент его переключения (для каждого слова может быть задан любой из двух таймеров). В этом процессе участвуют только те значения времени, которые расположены непосредственно в ячейках АЗУ. Информация, находящаяся в предварительном буфере хранения, не используется до тех пор, пока не будет переписана в освободившуюся ячейку АЗУ.

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.

Если результаты выполнения двух или нескольких команд, или одной команды, записанной несколько раз, являются противоположными, то выходы остаются в неизменном состоянии. Например, если в одной ячейке АЗУ записана команда HSO0, выполнение которой переключает выход HSO.0 в единицу, а в другой ячейке записана та же команда, но выполнение которой переключает выход HSO.0 в ноль, а значения времени совпадают, то состояние выхода HSO.0 не изменится. После этого, если команды не зафиксированы, они будут удалены из АЗУ.

Пример:

```
LDB     HSO_COMMAND, #00100000b
LDB     HSO_TIME, #2000h
LDB     HSO_COMMAND, #00000000b
LDB     HSO_TIME, #2000h
```

При программировании времени выполнения команды можно задавать значение времени непосредственно

```
LDB     HSO_COMMAND, #[параметры и команда]
LDB     HSO_TIME, #[время]
```

или как смещение относительно текущего значения таймера (в примере – таймера 1)

```
LDB     HSO_COMMAND, #[параметры и команда]
ADD     HSO_TIME, TIMER1, #[смещение]
```

Следует помнить, что загружаемое значение времени должно как минимум на две единицы превышать текущее значение счетчика таймера (т.е. «смещение» ≥ 02h), в

противном случае их совпадение произойдет только через период таймера (65 535/32 767 переключений). Это ограничение справедливо как для таймера 1, так и для таймера 2.

Для выполнения одной команды из АЗУ требуется один такт синхросигнала таймера. Если несколько команд имеют одинаковое значение времени, то для выполнения этих команд требуется несколько тактов. Таким образом, если значение времени одно для всех команд в АЗУ, то нужно восемь тактов. По этой причине подключенный к модулю HSO таймер должен переключаться каждый восьмой такт синхросигнала. Это важно учитывать при работе с таймером 2, поскольку он имеет возможность функционировать в режиме ускоренного счета.

Допускается использование режима ускоренного счета таймера 2 в случае необходимости быстрого выполнения одной (!) команды, при условии отсутствия в АЗУ других команд с совпадающими значениями времени. В случае наличия нескольких команд с одинаковыми значениями времени будет выполнена только одна команда. Кроме того, если эта команда зафиксирована, то она и только она будет выполняться в каждом периоде таймера. Если команда не была зафиксирована, то после выполнения она будет удалена из АЗУ. Оставшиеся команды будут выполняться аналогично.

Направление счета таймера 2 может быть любым и может меняться в процессе работы, но желательно программировать таймер на счет только в одном направлении.

Если таймер 2 запрограммирован на сброс внешним сигналом (установлен бит T2RST_SRC регистра IOC0), то не следует задавать значение времени выполнения команды, равным «0000h». Поскольку внешний сброс таймера является асинхронным по отношению к работе модуля HSO, то переключение таймера после сброса может произойти до того, как компаратор успеет произвести сравнение и есть вероятность того, что состояние «0000h» таймера не будет обнаружено, а, следовательно, не будет выполнена запрограммированная команда.

13 Аналого-цифровой преобразователь

Аналого-цифровой преобразователь реализован по схеме последовательного приближения и имеет структурную схему, представленную на рисунке 13.1.

Примечание – Схема на рисунке 13.1 соответствует ИС 1874ВЕ7Т и имеет 16 каналов (АСН0 – АСН15). Схема АЦП для ИС 1874ВЕ71Т аналогична представленной, за исключением количества каналов – их 12 (АСН0 – АСН8, АСН11 – АСН13). Дополнительная информация в таблицах 2.1 и 2.2.

АЦП может быть запрограммирован на работу в режиме однократного/непрерывного преобразования выбранного канала или в режиме последовательного сканирования каналов.

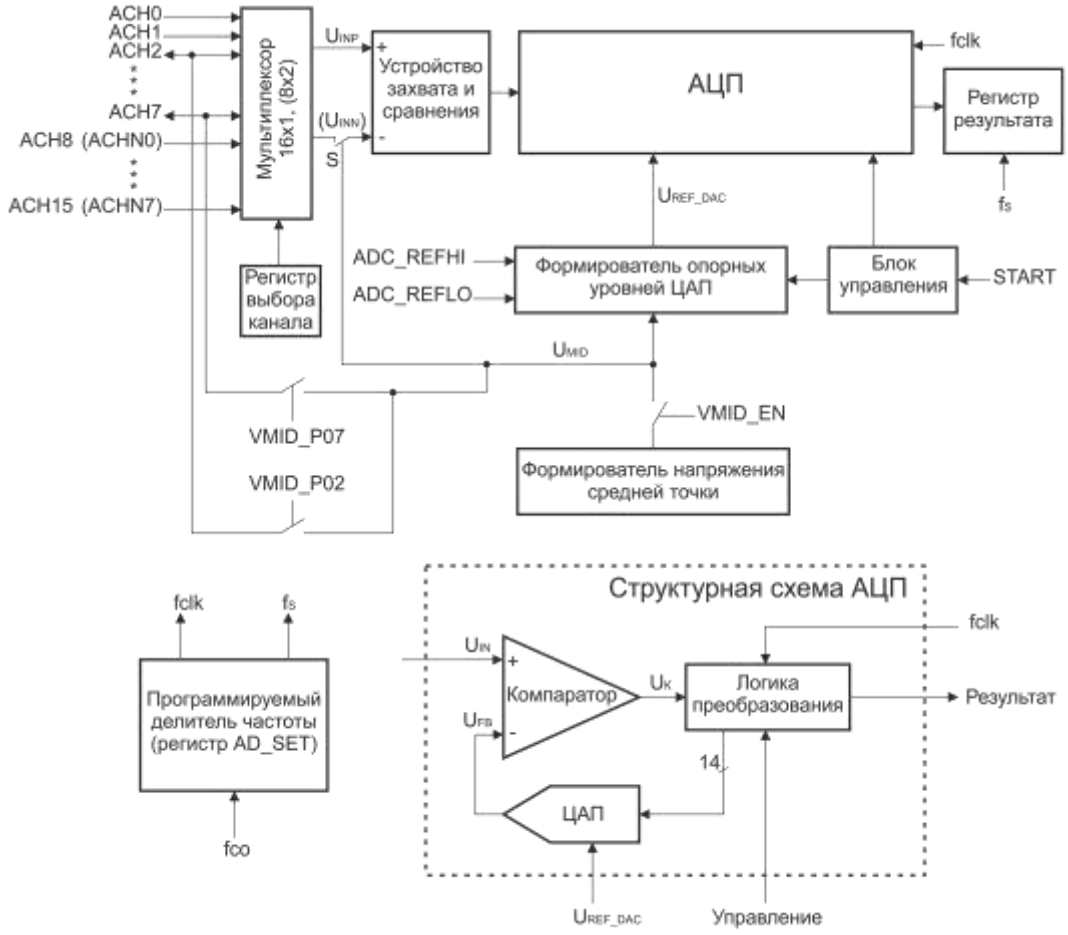


Рисунок 13.1 – Структурная схема модуля АЦП

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.).

Примечание – Для ИС 1874ВЕ71Т организуются 12 каналов, а в режиме дифференциального включения входов – 4 пары каналов.

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 P0.2 или P0.7, 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 ADC_SET.

13.1 Функциональные возможности

АЦП преобразует аналоговые уровни напряжений на входе в цифровые значения в режиме однополярного/дифференциального включения входов (режим задается битом EN_DIF регистра ADC_CON). Передаточная характеристика АЦП для режима однополярного включения входа показана на рисунке 13.2.

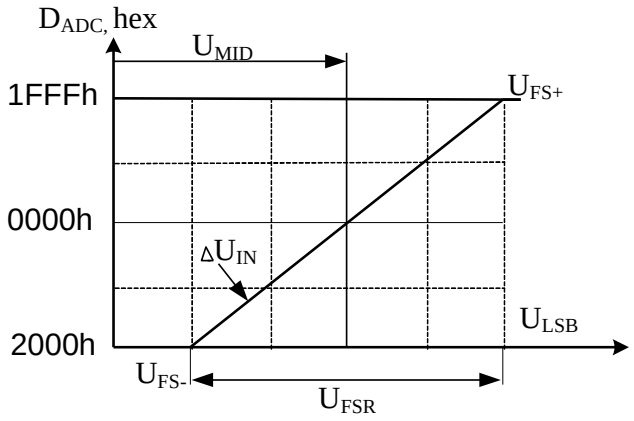


Рисунок 13.2 – Передаточная характеристика АЦП для режима однополярного включения входа (в общем виде)

Передаточная характеристика АЦП для режима дифференциального включения входов показана на рисунке 13.3.

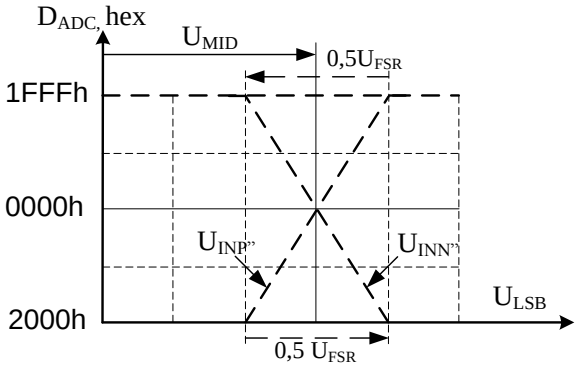


Рисунок 13.3 – Передаточная характеристика АЦП для режима дифференциального включения входов U_{INP} и U_{INN} относительно средней точки U_{MID} (дополнительно см. рисунок 13.6)

Соотношение режимных параметров в канале АЦП

Термины, обозначения и соотношения режимных параметров АЦП приведены в соответствии с ГОСТ 29109–91.

Приведенные ниже соотношения (13.1) – (13.5) 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:

$$U_{LSB} = \frac{U_{FSR}}{2^N}, \tag{13.1}$$

где U_{FSR} –диапазон напряжений преобразования АЦП, при выходном коде 2^N;

N – количество разрядов.

Диапазон преобразования определяется отношением:

$$U_{FSR} = U_{REF} \cdot K_G, \tag{13.2}$$

где U_{REF} – опорное напряжение АЦП;

K_G – коэффициент преобразования АЦП (тангенс угла наклона прямой преобразования).

Значение опорного напряжения АЦП U_{REF} расчитывается как:

$$U_{REF} = U_{REFH} - U_{REFL}. \tag{13.3}$$

Коэффициент преобразования указан на графике, представленном на рисунке 13.4.

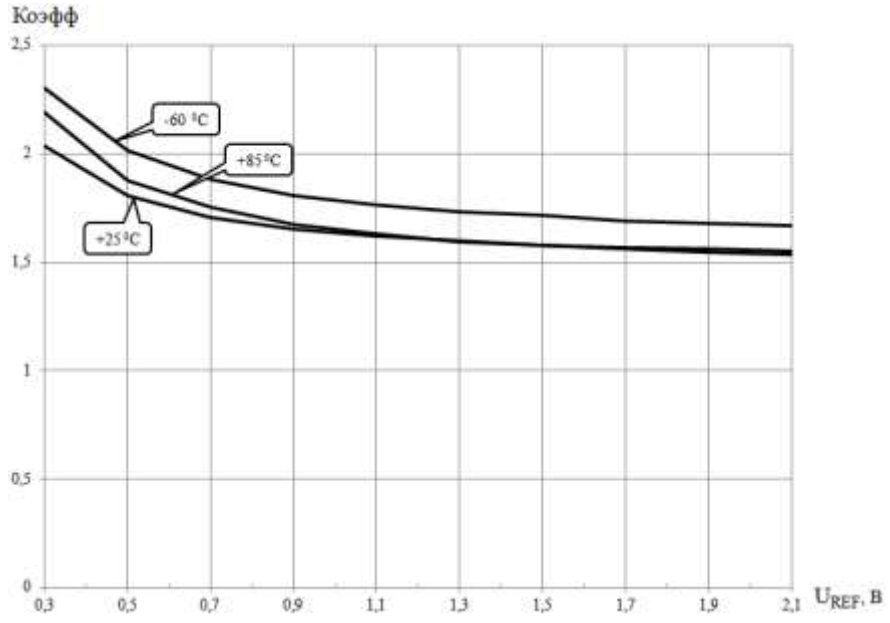


Рисунок 13.4 – Зависимость коэффициента преобразования АЦП от значения опорного напряжения

Тогда передаточная характеристика преобразования напряжения канала АЦП, показанная на рисунках 13.2 и 13.3 в режиме дифференциального включения входа, описывается формулой 13.4, а в режиме однополярного включения – формулой 13.5:

$$D_{ADC} = \frac{(U_{INP} - U_{INN}) \cdot 8192}{(U_{REFH} - U_{REFL}) \cdot K_G}. \tag{13.4}$$

$$D_{ADC} = 2 \times \frac{(U_{INP} - U_{INN}) \cdot 8192}{(U_{REFH} - U_{REFL}) \cdot K_G}. \tag{13.5}$$

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 13.5.

Напряжения внешних опорных источников U_{REFH} и U_{REFL} устанавливается, 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 13.5).

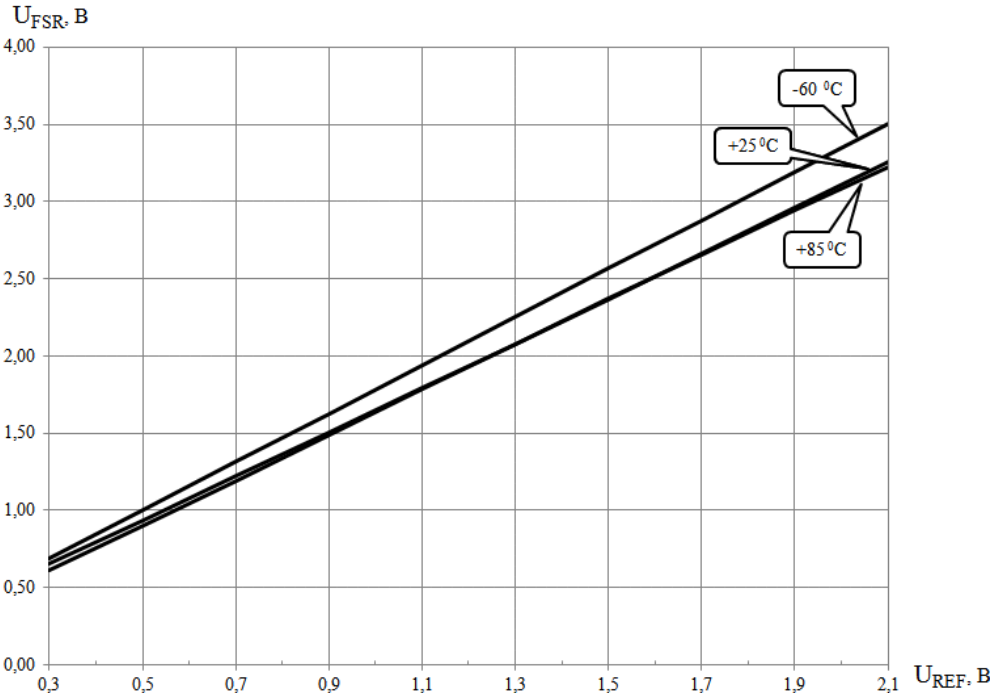


Рисунок 13.5 – Типовая зависимость U_{FSR} АЦП от опорного напряжения U_{REF}

Диаграмма входных сигналов при измерении динамических параметров АЦП SINAD и THD приведена на рисунке 13.6. Динамические и статические параметры АЦП приведены в таблицах 2.2 и 2.3.

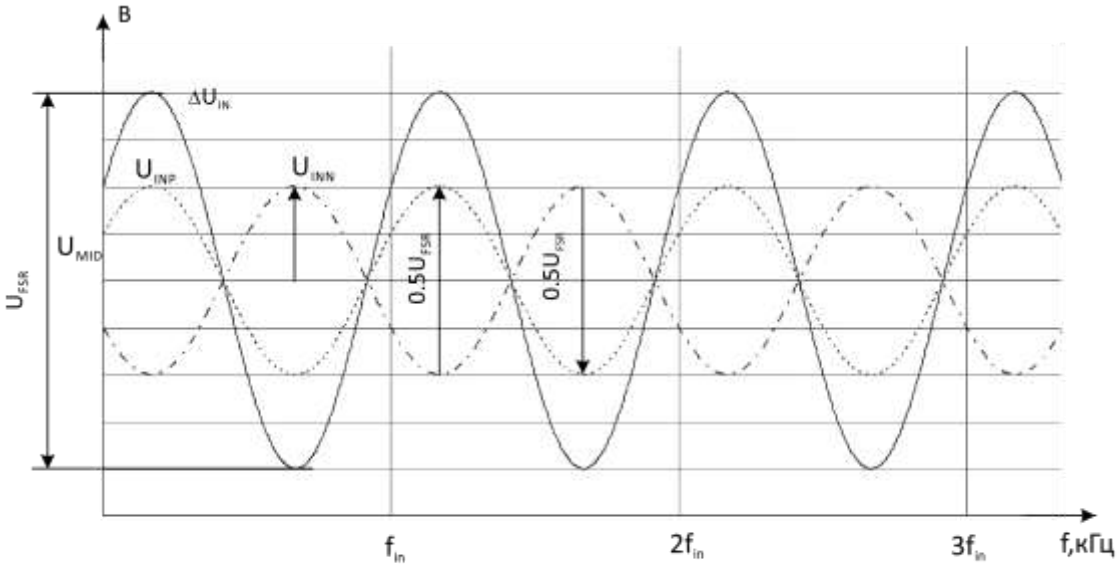


Рисунок 13.6 − Временные диаграммы входных напряжений АЦП при измерении динамических параметров

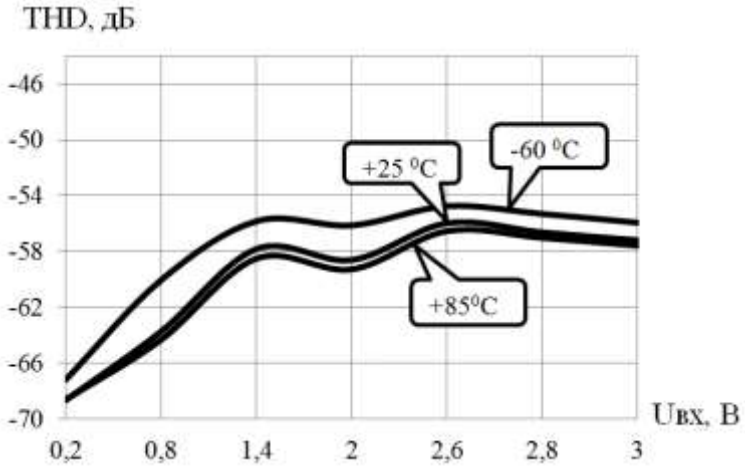


Рисунок 13.7 – Зависимость общих гармонических искажений THD канала АЦП в режиме одиночного входа в диапазоне температур от минус 60 до 85 °С при U_{CC2} = 3,3 В, U_{REF} = 2,1 В, f_s = 75 кГц

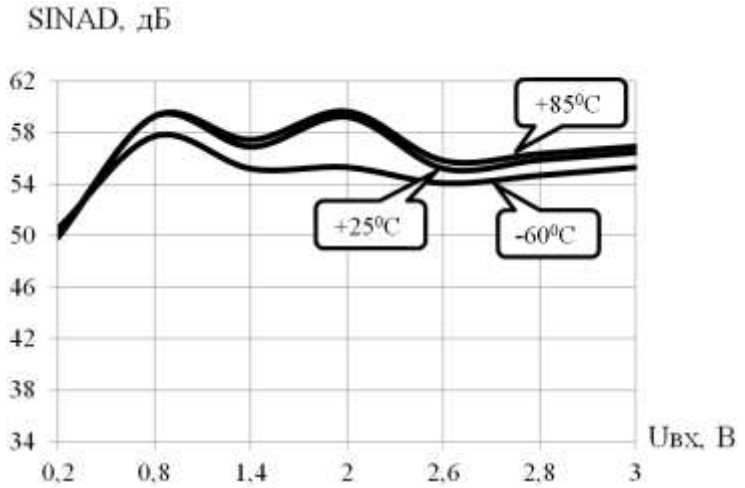


Рисунок 13.8 – Зависимость отношения сигнал/(шум+искажения) SINAD в канале АЦП в режиме одиночного входа в диапазоне температур от минус 60 до 85 °С при U_{CC2} = 3,3 В, U_{REF} = 2,1 В, f_s = 75 кГц

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 ADC_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_P07 и VMID_P02 регистра ADC_EN должны быть сброшены.

Внешний источник средней точки подключается к выводу P0.7 или P0.2, а его использование разрешается установкой бита VMID_P07 или VMID_P02, соответственно. При этом внутренний источник напряжения средней точки должен быть выключен (сброшен бит VMID_EN регистра ADC_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 ADC_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}.

Примечание – Часть выводов микроконтроллера реализуют в качестве альтернативных функций и ACHx и ACHNx, поэтому они должны быть сконфигурировны соответствующим образом, в зависимости от режима работы АЦП.

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 13.9). Для расчета RC-фильтра следует использовать соотношение:

$$f = \frac{1}{2 \cdot \pi \cdot R \cdot C}, \tag{13.6}$$

где f – частота среза, Гц;
R – сопротивление резистора, Ом;
C – емкость конденсатора, мкФ.

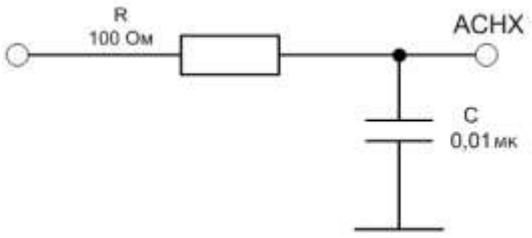


Рисунок 13.9 – Входной RC-фильтр АЦП при f_S = 150 кГц, f_{IN} = 15 кГц

13.2 Программирование АЦП

После подачи напряжения питания на выводы питания микроконтроллера модуль АЦП выключен, что обеспечивает снижение общей потребляемой мощности от источников питания.

Регистр ADC_EN

Инициализация АЦП запускается посредством регистра ADC_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 P0.7 или P0.2 установкой бита VMID_P07 или VMID_P02, 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 ADC_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 (или 12) каналов, независимо от режима включения входов (однополярного или дифференциального), поэтому режим является полнофункциональным 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 ADC_RESULT, являющийся индикатором выполнения преобразования.

Регистр ADC_RESULT

Регистр результата преобразования. Перезагружается новым значением по окончании каждого преобразования, поэтому должен своевременно считываться программно.

Регистр ADC_TRSCHAN

Регистр номера канала, для которого выполнялось преобразование. Перезагружается по окончании каждого преобразования (одновременно с регистром ADC_RESULT), поэтому должен своевременно считываться программно.

14 Широтно-импульсный модулятор

Встроенный трехканальный широтно-импульсный модулятор предназначен для генерации ШИМ сигналов на выходах микросхемы без участия процессора. ШИМ позволяет формировать импульсы с переменной скважностью и периодом следования.

Функциональные особенности

В состав ШИМ входит селектор опорного сигнала MUX, 8-битный счетчик, три идентичных формирователя сигналов pwm0, pwm1, pwm2 с блоком управления выводами (см. рисунок 14.1). Каждый формирователь состоит из двух регистров и компаратора.

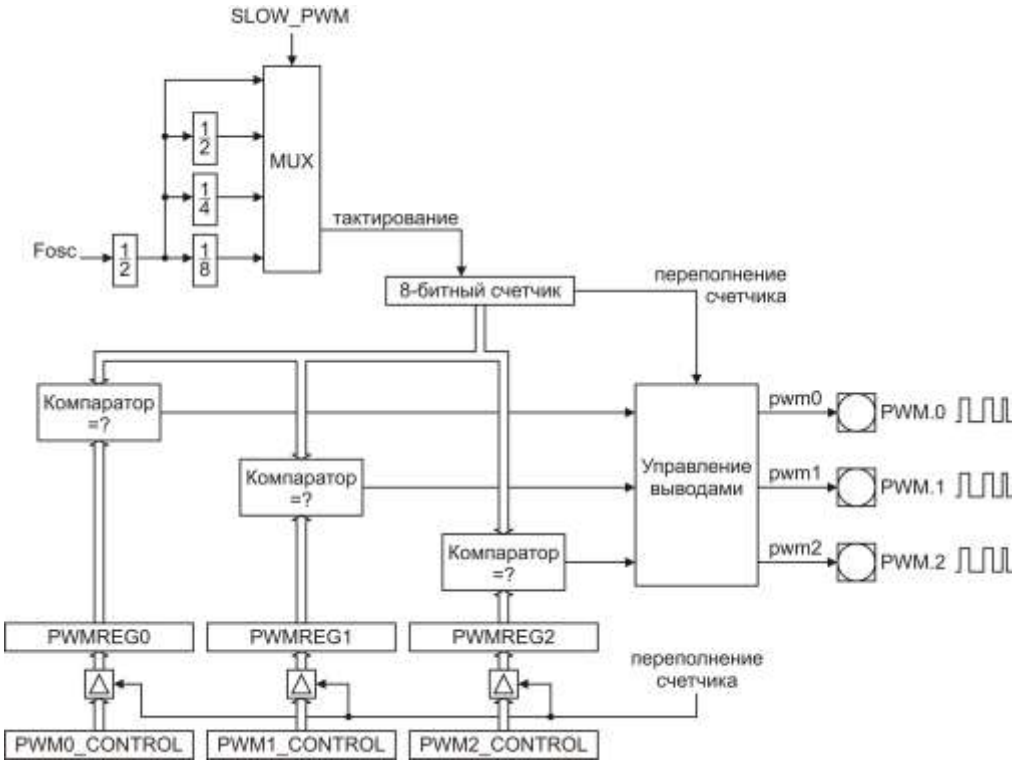


Рисунок 14.1 – Блок-схема модуля PWM

8-битный счетчик тактируется сигналом синхронизации, поступающим с селектора MUX. Счетчик инкрементируется с каждым тактом синхросигнала и по достижении значения FFh обнуляется. Одновременно с этим вырабатывается сигнал переполнения счетчика, который подается на логику управления выводами.

Регистры PWM0_CONTROL, PWM1_CONTROL и PWM2_CONTROL содержат запрограммированные значения длительностей импульсов формируемых сигналов pwm0, pwm1 и pwm2. Каждый раз при переполнении счетчика значения регистров PWM0_CONTROL, PWM1_CONTROL и PWM2_CONTROL загружаются в программно недоступные регистры PWMREG0, PWMREG1 и PWMREG2 соответственно. Такой механизм позволяет точно и легко управлять скважностью формируемых сигналов.

Посредством компараторов, содержимое регистров PWMREG0, PWMREG1 и PWMREG2 постоянно сравнивается с содержимым счетчика. С каждым компаратором связан триггер, который удерживает соответствующую линию вывода формируемого сигнала в том или ином логическом состоянии.

Начальное состояние сигналов на выводах PWM.0, PWM.1 и PWM.2 – логическая единица. Компаратор, обнаруживший совпадение значений регистра и счетчика, вырабатывает сигнал, поступающий на логику управления выводами, и на соответствующем выводе устанавливается логический ноль. По сигналу переполнения счетчика на всех выводах устанавливается логическая единица.

На рисунке 14.2 показан пример формирования импульсов на выводах PWM.0, PWM.1 и PWM.2 при постоянных значениях регистров PWM0_CONTROL, PWM1_CONTROL и PWM2_CONTROL.

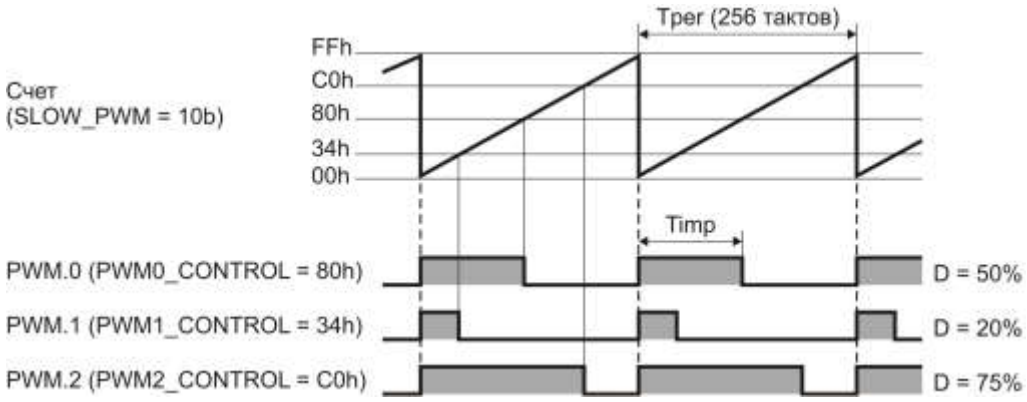


Рисунок 14.2 – Формирование импульсов

Управление модуляцией сигналов

Базовым сигналом тактирования счетчика является сигнал Fpwm, поступающий на блок ШИМ. Fpwm = Fosc/2. По умолчанию счетчик будет инкрементироваться с частотой Fosc/2. Для понижения частоты переключения счетчика следует задать соответствующий коэффициент деления k посредством битов SLOW_PWM в регистре IOC2.

Период счетчика Tper, мкс, составляет 256 тактов синхросигнала (см. рисунок 14.2) и, при заданном коэффициенте деления k и известной частоте синхросигнала Fpwm, МГц, может быть рассчитан по формуле

$$Tper = 256 \times \frac{k}{Fpwm} . \qquad (14.1)$$

Для задания длительности импульса Timp (см. рисунок 14.2) каждого сигнала используются регистры PWM0_CONTROL, PWM1_CONTROL и PWM2_CONTROL.

Регистры длительности импульса позволяют управлять коэффициентами заполнения сигналов и таким образом осуществлять широтно-импульсную модуляцию. На рисунке 14.2 для каждого сигнала указан его коэффициент заполнения D (в процентах). Соотношение между длительностью импульса Timp и коэффициентом D следующее

$$D = \frac{IMP_{DEC}}{256} \times 100\ \%, \qquad (14.2)$$

где IMP_{DEC} – значение поля IMP в десятичном формате.

Если в регистр длительности импульса будет записано значение 00h, то при очередном переполнении счетчика на соответствующем выводе будет установлен логический ноль и будет удерживаться до тех пор, пока в регистр длительности импульса не будет записано значение, отличное от нуля.

Если значение IMP равно FFh, то переключение соответствующего вывода в ноль будет происходить каждый раз, когда счетчик будет достигать значения FFh. После сброса счетчика вывод будет переключаться в единицу. Как видно, длительность удержания уровня логического нуля в этом случае составит один такт синхросигнала счетчика.

На рисунке 14.3 показана форма выходного сигнала на выводе PWM.0 для случая, когда значение регистра PWM0_CONTROL равно FFh. Из рисунка также видно, что счетчик переключается с частотой, которая в два раза меньше частоты сигнала Fpwm (обусловлено значением 01b в поле SLOW_PWM).

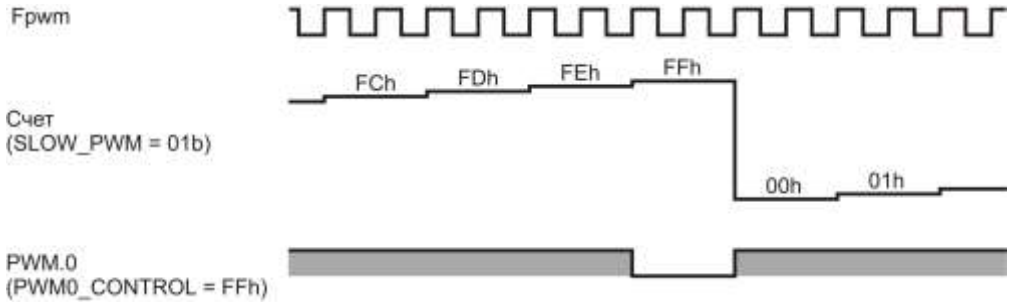


Рисунок 14.3 – Форма выходного сигнала при IMP = FFh

Пример программирования

fosc = 20 МГц и, как следствие, Fpwm = fosc/2 = 10 МГц.

В SLOW_PWM записывается значение 01b, что соответствует выбору делителя «2» для частоты сигнала тактирования счетчика. Для расчета длительности периода счетчика используется формула (14.1):

$$Tper = 256 \times k / Fpwm = 256 \times 2 / 10 = 51{,}2 \text{ мкс}$$

Коэффициенты заполнения D0, D1, D2 сигналов:
- на выводе PWM.0 – 0,5;
- на выводе PWM.1 – 0,2;
- на выводе PWM.2 – 0,75.

Длительности импульсов:
- Timp0 = Tper × D0 = 51,2 × 0,5 = 25,6 мкс;
- Timp1 = Tper × D1 = 51,2 × 0,2 = 10,24 мкс;
- Timp2 = Tper × D2 = 51,2 × 0,75 = 38,4 мкс.

Согласно формуле (14.2):

- $IMP_{DECPWM0}$ = 256 × 0,5 = 128 и, следовательно, в регистр PWM0_CONTROL следует записать значение 80h;
- $IMP_{DECPWM1}$ = 256 × 0,2 = 51,2 и, следовательно, в регистр PWM1_CONTROL следует записать значение 34h;
- $IMP_{DECPWM2}$ = 256 × 0,75 = 192 и, следовательно, в регистр PWM2_CONTROL следует записать значение C0h.

Этому примеру соответствует поведение сигналов, представленное на рисунке 14.2.

Управление выводами

Передача выходного сигнала нулевого канала модулятора является альтернативной функцией вывода 5 порта P2. Для включения альтернативной функции следует установить бит PWM_SEL в регистре IOC1. Выходные сигналы первого и второго каналов передаются напрямую через отдельные выводы, не имеющие альтернативных функций.

15 Контроллер интерфейса ГОСТ Р 52070–2003

Модуль представляет собой устройство, поддерживающее обмен данными с другими устройствами (контроллерами) посредством магистрального последовательного интерфейса (ГОСТ Р 52070–2003), совместимого со стандартом MIL-STD-1553B.

На 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.

Модуль магистрального последовательного интерфейса имеет два канала приема-передачи – основной с выводами M_TxD0, M_TxD_N0, M_RxD0, M_RxD_N0 и резервный с выводами M_TxD1, M_TxD_N1, M_RxD1, M_RxD_N1. Каналы полностью идентичны. Модуль может функционировать в одном из трех режимов:

- контроллер шины (КШ);
- удаленный терминал (УТ);
- монитор шины (МШ).

Установка режима, а также других дополнительных параметров работы осуществляется посредством регистра конфигурации BSICONFIG.

Для безотказной работы модуля следует использовать следующие частоты синхронизации микроконтроллера: 12, 16, 20 и 24 МГц.

Примечание – При использовании частот 12 и 16 МГц входные сигналы M_RxD и M_RxD_N от приемника рекомендуется синхронизировать по фронту сигнала CLKOUT с использованием D-триггера.

15.1 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. Контроллер шины может посылать и принимать как одиночные сообщения, так и цепочки сообщений. Одно сообщение может состоять из одного или более слов.

Используются следующие типы слов:

- командное;
- данных;
- ответное.

Командное слово позволяет указать устройства на шине, между которыми нужно произвести передачу слова/слов данных, а также направление передачи. Ответное слово передается последним и несет информацию о состоянии устройства и корректности передачи.

Для задания параметров передачи сообщения, отслеживания и исправления ошибок и формирования запроса на прерывание используется так называемое управляющее слово. Оно не передается по шине и программируется для каждого сообщения индивидуально. Для задания последовательности передачи сообщений используется слово указателя. Это слово также не передается по шине.

Объем области ОЗУ микроконтроллера, которая выделяется модулю магистрального последовательного интерфейса во время его работы в режиме контроллера шины,

составляет 4 Кбайта. Область расположена в диапазоне адресов от 0800h до 17FFh и имеет условное название – память сообщений. Если модуль не используется, указанная область может использоваться как обыкновенное ОЗУ.

Перед началом передачи сообщения оно должно быть размещено в памяти сообщений. Управляющее слово, слово указателя и все слова сообщения должны быть записаны строго в установленном порядке, образовав таким образом блок сообщения. Адрес начала блока может быть любым при условии, что значение адреса его последнего слова не превышает 17FEh. Порядок записи слов определяется форматом сообщения.

Допускаются 10 форматов сообщений, которые будут подробнее рассмотрены ниже. Здесь для примера приведен формат 1.

Порядок формирования блока сообщения формата 1 посредством последовательной записи слов в память сообщений:

1 Управляющее слово.

2 Слово указателя (адрес управляющего слова следующего сообщения).

3 Командное слово.

4 Последовательно все слова данных (не более 32 слов).

5 Резерв для ответного слова (для записи принятого слова).

На рисунке 15.1 представлен пример размещения слов в памяти сообщений.

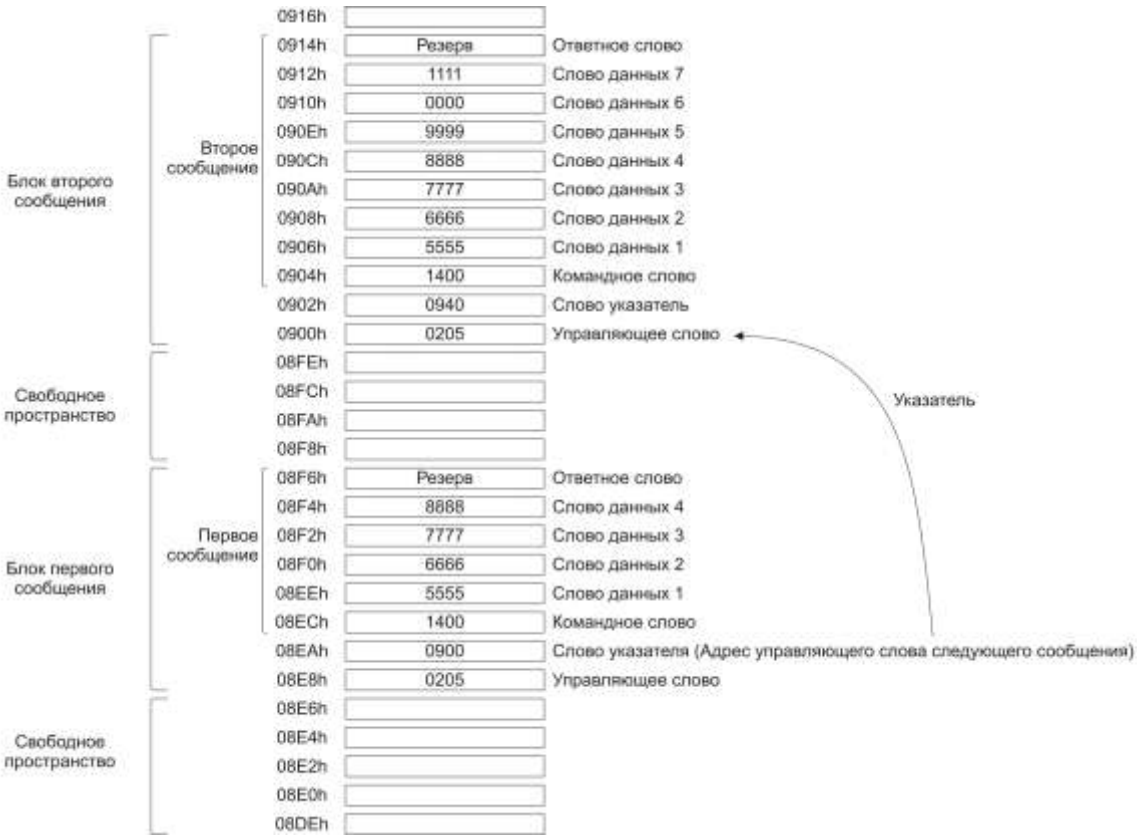


Рисунок 15.1 – Пример размещения двух сообщений в формате 1

Пояснение к рисунку 15.1.

Блок первого сообщения размещен в диапазоне с 08E8h по 08F6h и имеет в своем составе указатель на начало следующего блока сообщения. Блок второго сообщения размещен с 0900h по 0914h. В каждом блоке сообщение имеет формат 1.

В свободном пространстве может находиться любая информация.

Порядок и последовательность размещения блоков сообщений в пределах памяти не важен, поскольку все они связаны посредством указателей. Описание слов приведено в таблицах 15.1 – 15.5.

Таблица 15.1 – Формат и назначение битов управляющего слова

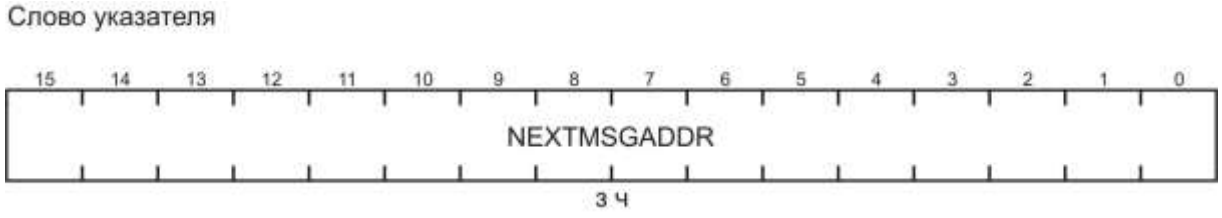
Управляющее слово (УС)

Поле	Бит	Тип	Описание		
1	2	3	4		
CHNUM	15	Запись Чтение	Выбор канала для передачи/приема		
			0	Основной	
			1	Резервный	
FRAME	14	Запись Чтение	Указатель формата сообщения		
			0	Используется любой формат, за исключением форматов 3 и 8	
			1	Используется формат 3 или формат 8	
SWITCH	13	Запись Чтение	Бит переключения канала при ошибке		
			0	Канал передачи/приема не меняется при повторной передаче	
			1	Канал передачи/приема меняется после каждого повторения (если установлен бит JUMPEN в регистре BSICON)	
REPNUM	12 - 10	Запись Чтение	Поле количества повторений. Если во время передачи была обнаружена ошибка, сообщение может быть передано повторно. Поле REPNUM устанавливает, следует ли повторить попытку передачи или нет, а также количество повторов		
			000	Повтор запрещен	Разрешено, если установлен бит REPEN регистра BSICON
			001	1	
			010	2	
			011	4	
			100	8	
			101	16	
			110	31	
			111	Повтор передачи до тех пор, пока сообщение не будет передано без ошибок (не будет ошибок в словах данных и установленных битов ошибок в ответном слове)	
NOP	9	Запись Чтение	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		
MSGERR MASK	8	Запись Чтение	Маска бита MSGERR ответного слова		

Окончание таблицы 15.1

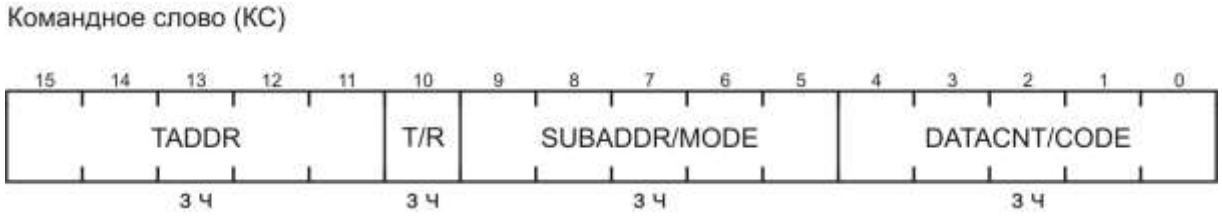
1	2	3	4
SERVREQ MASK	7	Запись Чтение	Маска бита SERVREQ ответного слова
ABUSY MASK	6	Запись Чтение	Маска бита ABUSY ответного слова
AERR MASK	5	Запись Чтение	Маска бита AERR ответного слова
TERR MASK	4	Запись Чтение	Маска бита TERR ответного слова
ASK MISS MASK	3	Запись Чтение	Игнорирование битов ошибок ответного слова. Если бит установлен, то прием недостоверного ответного слова или отсутствие его вообще не является ошибкой
MSGINT EN	2	Запись Чтение	Бит разрешения прерывания (только для этого сообщения). Если этот бит установлен, то по окончании передачи сообщения будет сформирован запрос на прерывание. Состояние этого бита не влияет на формирование общего запроса на прерывание, который всегда возникает по окончании передачи всех сообщений одной группы, т.е. после сброса бита START в регистре BSICON
NOT LASTMSG	0	Запись Чтение	Индикатор последнего сообщения. Сообщение, в управляющем слове которого этот бит не установлен, является последним в цепочке сообщений
-	1	-	Зарезервировано
Примечание – Если бит маски установлен, то маска считается включенной. Бит ответного слова, закрытый соответствующей маской, считается сброшенным.			

Таблица 15.2 – Формат и назначение битов слова указателя



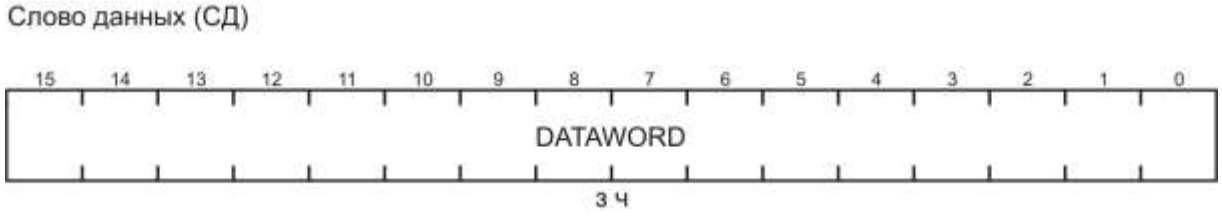
Поле	Бит	Тип	Описание
NEXTMSG ADDR	15-0	Запись Чтение	Адрес управляющего слова сообщения, которое должно быть передано следующим

Таблица 15.3 – Формат и назначение битов командного слова



Поле	Бит	Тип	Описание	
TADDR	15-11	Запись Чтение	Адрес удаленного терминала. В поле записывается адрес удаленного терминала, к которому обращается контроллер шины. Значение 1Fh является групповым адресом и служит для обращения ко всем удаленным терминалам одновременно.	
T/R	10	Запись Чтение	Направление передачи данных	
			1	Передача
			0	Прием
SUBADDR/ MODE	9-5	Запись Чтение	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	Запись Чтение	Количество данных/Код команды управления. Режим работы битового поля устанавливается полем SUBADDR/MODE	

Таблица 15.4 – Формат и назначение битов слова данных



Поле	Бит	Тип	Описание
DATAWORD	15-0	Запись Чтение	Данные

Инициализация

После того, как блоки сообщений сформированы и размещены в памяти сообщений, механизм передачи может быть запущен. Для запуска, установки и задания параметров передачи используются три регистра BSISADDR, BSISPACE и BSICON.

Примечание – В состав модуля магистрального последовательного интерфейса входит один регистр управления с названием BSICON (адрес 1F32h). Назначение его битов меняется в зависимости от режима, в котором функционирует модуль.

В регистр BSISADDR записывается стартовый адрес блока сообщения, которое должно быть передано первым. Фактически, стартовый адрес блока – это адрес, по которому расположено управляющее слово (в примере на рисунке 15.1 стартовый адрес – 08E8h).

Управление осуществляется посредством регистра BSICON.

Для инициализации передачи следует установить бит START в регистре BSICON.

Далее на аппаратном уровне выполняются следующие действия:

1 После инициализации:

а) чтение управляющего слова сообщения, на которое указывает регистр BSISADDR;

б) проверка состояния бита NOP управляющего слова; если бит установлен, то выполняется переход к управляющему слову следующего сообщения; если бит не установлен, то выполняется чтение командного слова сообщения и загрузка его в передающий регистр выбранного канала (выбирается битом CHNUM);

в) передача/получение всех остальных слов сообщения в порядке, установленном форматом сообщения.

2 После успешного окончания передачи/приема одного сообщения из цепочки, проверяется состояние бита NOTLASTMSG управляющего слова сообщения:

а) если бит установлен, то выполняется переход к управляющему слову следующего сообщения цепочки и далее по описанному выше алгоритму, начиная с 1.б);

б) если бит не установлен, то передача сообщений завершается, бит START сбрасывается и модуль переходит в режим ожидания.

3 В случае обнаружения ошибки проверяется состояния бита REPEN регистра BSICON:

а) если бит установлен, то в зависимости от значения поля REPNUM управляющего слова, осуществляется повтор передачи сообщения или, если REPNUM = 000b, бит START сбрасывается и модуль переходит в режим ожидания. Если значение поля REPNUM не равно нулю, то сообщение будет повторно передаваться указанное количество раз или до тех пор, пока не будет выполнена передача без ошибок. При этом при каждом повторе передачи сообщения будет происходить переключение канала (основной и резервный каналы активируются по очереди) или будет использоваться только один канал, в зависимости от состояния битов SWITCHEN (регистр BSICON) и SWITCH (управляющее слово);

б) если бит не установлен, то бит START сбрасывается и модуль переходит в режим ожидания.

Прерывания

В режиме контроллера шины формируются запросы на прерывания:

1 По окончании успешной передачи сообщения, если установлен бит MSGINTEN в управляющем слове.

2 По окончании успешной передачи всей цепочки сообщений (т.е. по окончании передачи сообщения, в управляющем слове которого был сброшенный бит NOTLASTMSG).

3 По ошибке, если:

- время непрерывной передачи превысило 800 мкс;
- обнаружена ошибка, а бит REPEN и/или поле REPNUM обнулены;

- после выполнения всех запрограммированных повторений снова обнаруживается ошибка.

Примечание – Если запрос на прерывание формируется по ошибке, то одновременно с ним в регистре BSISTAT устанавливается флаг DATAERR.

Поскольку причин для формирования запроса на прерывание несколько, а вектор прерывания один, то при обслуживании запроса следует считывать состояние регистра BSISTAT для точного выявления события, вызвавшего прерывание.

Примечание – В состав модуля магистрального последовательного интерфейса входит один регистр состояния с названием BSISTAT и адресом 1F38h. Назначение его битов меняется в зависимости от режима, в котором функционирует модуль.

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 сообщений.

Форматы основных сообщений

Применяются для передачи информации, предназначенной одному из удаленных терминалов с получением от него соответствующего ответа. Описания форматов приведено в таблице 15.5.

Таблица 15.5 – Форматы основных сообщений

Название формата	Алгоритм передачи	Порядок размещения слов в блоке сообщения
1	2	3
Формат 1 (см. пример на рисунке 15.1)	Передача данных от контроллера шины к удаленному терминалу	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Командное слово. 4 Последовательно все слова данных (не более 32 слов). 5 Резерв для ответного слова
Формат 2	Передача данных от удаленного терминала к контроллеру шины	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Командное слово. 4 Резерв для ответного слова. 5 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 Командное слово для приемника. 4 Командное слово для передатчика. 5 Резерв для ответного слова от передатчика. 6 Резерв для слов данных (согласно запросу). 7 Резерв для ответного слова от приемника
Формат 4	Передача команды управления от контроллера шины к удаленному терминалу	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Командное слово. 4 Резерв для ответного слова

Окончание таблицы 15.5

1	2	3
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 Командное слово. 4 Резерв для ответного слова. 5 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 Командное слово. 4 Слово данных. 5 Резерв для ответного слова

Форматы групповых сообщений

Применяются для передачи информации, предназначенной нескольким удаленным терминалам без получения от них ответных слов. Групповые сообщения всегда начинаются с передачи команды общего вызова с кодом 1Fh в командном слове. Каждый удаленный терминал, который может принять команду общего вызова, после ее обнаружения устанавливает соответствующий флаг в регистре, где хранится ответное слово, но не передает ответное слово.

Описания форматов приведено в таблице 15.6.

Таблица 15.6 – 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 Командное слово. 4 Последовательно все слова данных (не более 32 слов)
Формат 8	Передача данных (в групповом сообщении) от удаленного терминала к удаленному терминалу	1 Управляющее слово. 2 Адрес управляющего слова следующего сообщения. 3 Командное слово для приемника. 4 Командное слово для передатчика. 5 Резерв для ответного слова от передатчика. 6 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 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 Командное слово. 4 Слово данных

Команды управления

Команды управления передаются в поле DATACNT/CODE командного слова контроллера шины и применяются только для управления удаленными терминалами (не для обмена данными!). Наличие команд управления позволяет контроллеру шины не только контролировать работу сети, но и обрабатывать, и исправлять обнаруженные ошибки.

Команды управления с кодами 00h – 08h применяются без слов данных, а 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 15.7.

Таблица 15.7 – Команды управления, передаваемые в командном слове

Код команды	Название команды управления	Бит T/R	Возможность применения		Прерывание*
			в групповых сообщениях	со словом данных	
01h	Синхронизация	1	Да	Нет	Да
02h	Передать ответное слово	1	Нет		Нет
03h	Начать самоконтроль	1	Да		Да
04h	Блокировать передатчик	1	Да		Да
05h	Разблокировать передатчик	1	Да		Да
06h	Блокировать признак неисправности удаленного терминала	1	Да		Нет
07h	Разблокировать признак неисправности удаленного терминала	1	Да		Нет
08h	Установить удаленный терминал в исходное состояние	1	Да		Нет
10h	Передать векторное слово	1	Нет	Да	Да
11h	Синхронизация со словом данных	0	Да		Да
12h	Передать последнюю команду	1	Нет		Нет
13h	Передать слово встроенной системы контроля (ВСО) удаленного терминала к контроллеру шины	1	Нет		Да
14h	Блокировать выбранный передатчик	0	Да		Да
15h	Разблокировать выбранный передатчик	0	Да		Да
00h, 09h–0Fh, 16h–1Fh	Зарезервировано. Не использовать!				

_____

* В случае получения управляющей команды удаленный терминал не формирует запрос на прерывание, а только аппаратно выполняет предписанное командой действие.

15.2 Удаленный терминал

Удаленный терминал выполняет прием и дешифрирование командных слов, выполняет команды управления, формирует и выдает через выбранный основной/ резервный канал ответные слова, отслеживает ошибки в сообщениях. Каждому удаленному терминалу присваивается адрес в диапазоне от 00h до 1Eh.

Объем области ОЗУ микроконтроллера, которая выделяется модулю магистрального последовательного интерфейса во время его работы в режиме удаленного терминала, составляет 4 Кбайта. Область расположена в диапазоне адресов от 0800h до 17FFh и имеет

условное название – 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. Адресация блоков начинается с 0800h.

Примечание – Блоки данных с адресами 0800h и 1000h не используются, поэтому области 0800h – 083Fh и 1000h – 103Fh могут быть заполнены сторонними данными, размещаемыми в ОЗУ микроконтроллера.

Половина области сообщений с адресами от 0840h до 0FFFh выделена для записи и хранения принятых слов данных (структура показана на рисунке 15.2). Вторая половина – для хранения слов данных, записанных пользователем и предназначенных для передачи (структура показана на рисунке 15.3).

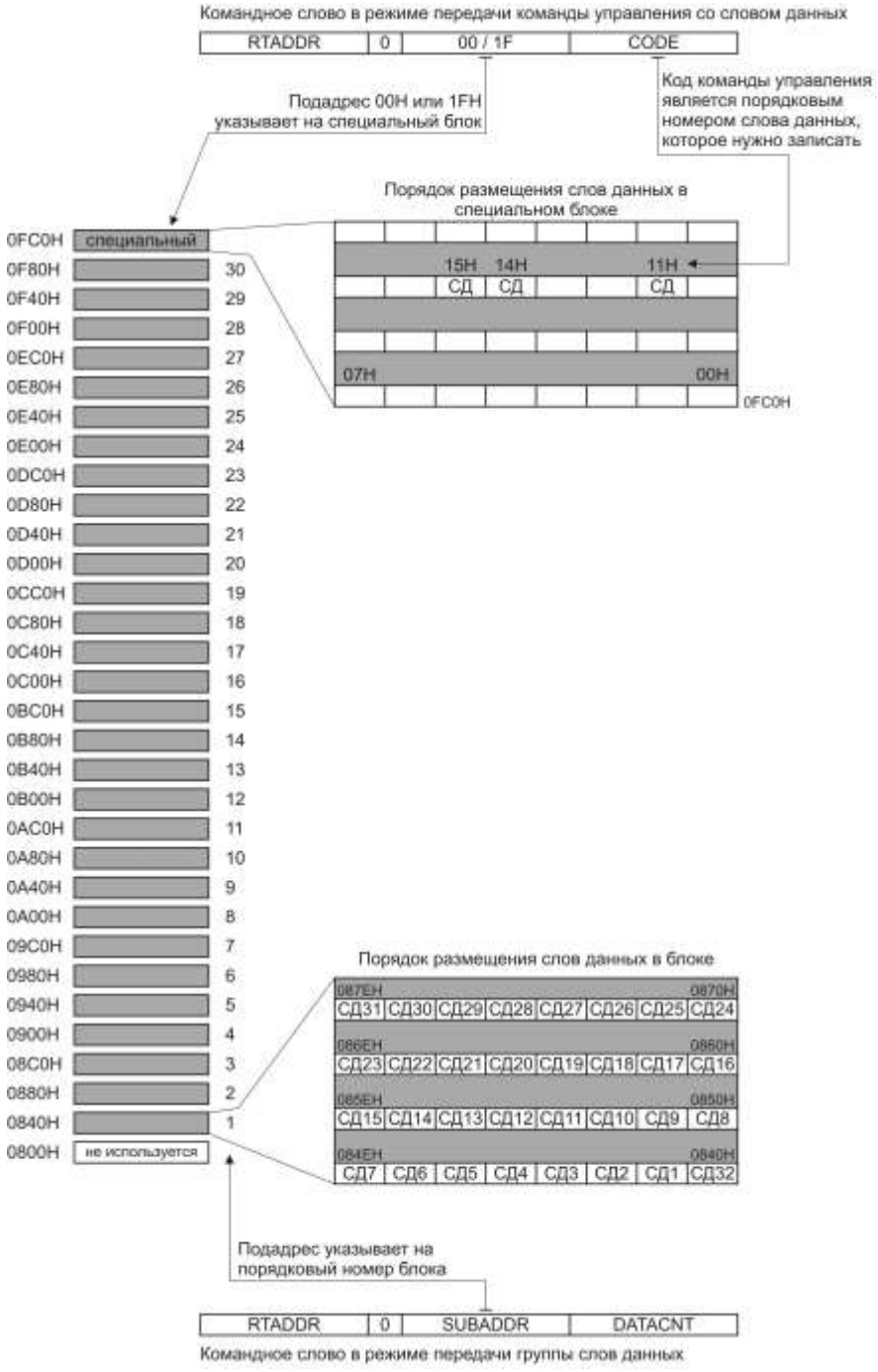


Рисунок 15.2 – Структура области памяти сообщения для принятых слов данных

Пояснение к рисунку 15.2. Память поделена на 32 блока данных, каждый со своим порядковым номером от 0 до 31. Нулевой блок данных не используется. 31-й блок данных является специальным. Остальные 30 блоков могут заполняться принятыми данными. Структура всех блоков идентична. В пределах блока располагаются 32 слова данных.

Получив командное слово, удаленный терминал анализирует его.

T/R = 0 указывает на то, что нужно принять данные.

SUBADDR указывает на номер блока, в который следует записать слова данных.

DATACNT указывает, какое количество данных будет передано. DATACNT = 00h означает, что будут переданы 32 слова данных.

Принимаемые слова данных записываются в блок последовательно, начиная со второй ячейки, условно обозначенной СД1, а после – в СД2, СД3 и т.д. 32-е слово данных записывается в первую ячейку блока, обозначенную СД32. Если значение поля 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.

Таблица 15.8 – Соответствие кодов команд управления и адресов ОЗУ

Код команды управления, с которой передается одно слово данных	Физический адрес ячейки памяти, в которую записывается принятое слово данных
11h	0FE2h
14h	0FE8h
15h	0FEAh

Пояснение к рисунку 15.3. Память поделена на 32 блока данных, каждый со своим порядковым номером от 0 до 31. Нулевой блок данных не используется. 31-й блок данных является специальным. Остальные 30 блоков могут хранить данные. Структура всех блоков идентична. В пределах блока располагаются 32 слова данных.

Данные, предназначенные для передачи, должны быть предварительно записаны в память. Пользователь записывает данные, обращаясь к физическим адресам ячеек блока как к ОЗУ. Запись данных может происходить в любом порядке. В тоже время следует помнить, что порядок чтения слов данных из блока всегда один. Первым читается слово из ячейки СД1, а после из СД2, СД3 и т.д. Из ячейки СД32 слово данных будет прочитано последним.

Получив командное слово, удаленный терминал анализирует его.

T/R = 1 указывает на то, что нужно передать данные.

SBADDR указывает на номер блока, из которого следует прочитать слова данных.

DATACNT указывает, какое количество данных требуется прочитать. DATACNT = 00h означает, что требуется прочитать все 32 слова данных.

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 один блок, называемый специальным. В пределах этого блока каждая ячейка имеет свой порядковый номер от 00h до 1Fh. Код полученной команды управления и будет указывать на ячейку, данные из которой следует передать.

Специальные данные, предназначенные для передачи, должны быть предварительно записаны в память также, как обыкновенные слова данных (см. таблицу 15.9).

Таблица 15.9 – Соответствие кодов команд управления и адресов ОЗУ

Код команды управления, запрашивающей специальное слово данных	Физический адрес ячейки памяти, из которой будет прочитано слово данных
10h	17E0h
12h	17E4h
13h	17E6h

В режиме удаленного терминала можно включить функцию распознавания адреса общего вызова и механизм работы с групповыми сообщениями. Для этого следует установить бит BCMSGEN в регистре BSICON.

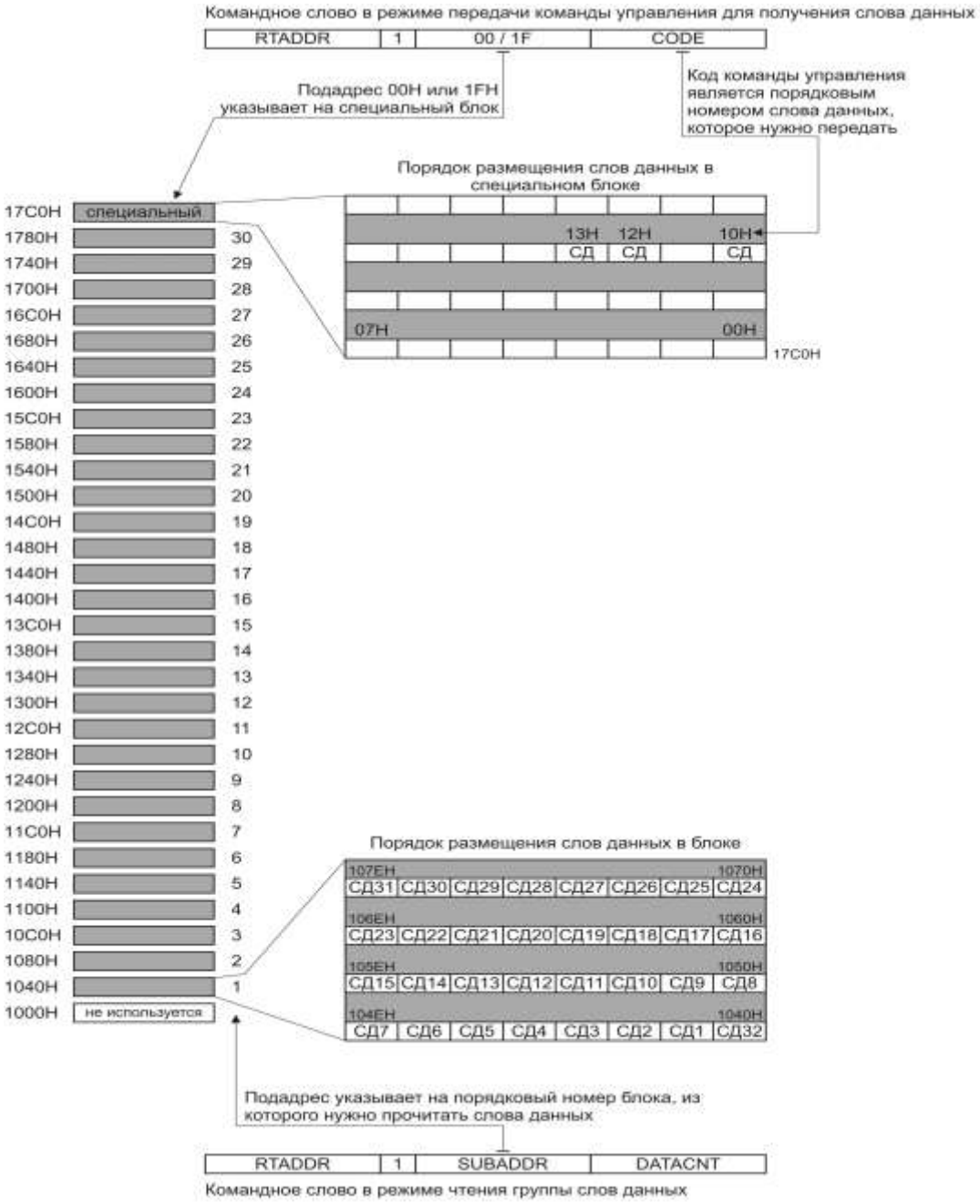


Рисунок 15.3 – Структура области памяти сообщения для передачи слов данных

Инициализация

Перед включением терминала необходимо задать его конфигурацию. В регистре BSICONFIG:

- в поле MODE записать значение 10h;
- установить битом TXDLEV желаемый уровень сигнала;
- в поле DIV записать значение делителя.

При необходимости сброса удаленного терминала можно установить бит RST. Состояние поля ASKDELAY не важно.

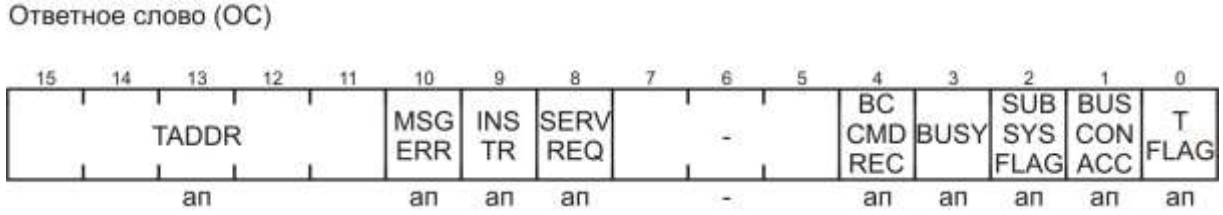
Управление осуществляется посредством регистра BSICON.

После конфигурирования устройства, размещения данных в памяти сообщений и указания собственного адреса, удаленный терминал готов к работе.

После установки бита TON удаленный терминал переходит в режим ожидания командного слова.

Для контроля работы удаленного терминала со стороны контроллера шины используется ответное слово, которое отправляется контроллеру в конце передачи каждого сообщения или по прямому запросу. Описание ответного слова приведено в таблице 15.10.

Таблица 15.10 – Формат и назначение битов ответного слова ОС



Поле	Бит	Тип	Описание
1	2	3	4
TADDR	15-11	Аппаратное влияние	Собственный адрес удаленного терминала
MSGERR	10	Аппаратное влияние	Ошибка в сообщении. Установленный бит указывает на то, что удаленный терминал не смог осуществить корректный прием
INSTR	9	Аппаратное влияние	Бит распознавания (всегда ноль)
SERV REQ	8	Аппаратное влияние	Запрос на обслуживание Устанавливается посредством бита SERVREQ регистра BSICON в режиме удаленного терминала
BCCMD REC	4	Аппаратное влияние	Получена групповая команда. Устанавливается после получения команды с широковещательным адресом
BUSY	3	Аппаратное влияние	Абонент занят. Устанавливается посредством бита ABUSY регистра BSICON в режиме удаленного терминала. Указывает на то, что подчиненное устройство (абонент) удаленного терминала занят

Окончание таблицы 15.10

1	2	3	4
SUBSYS FLAG	2	Аппаратное влияние	Неисправность абонента. Устанавливается посредством бита AERR регистра BSICON в режиме удаленного терминала. Указывает на ошибки в работе подчиненного устройства (абонента) удаленного терминала
BUSCON ACC	1	Аппаратное влияние	Бит подтверждения принятия управления. Устанавливается аппаратно в случае, если удаленный терминал не был занят и получил команду управления «Принять управление интерфейсом» с кодом 00h
TFLAG	0	Аппаратное влияние	Неисправность удаленного терминала. Устанавливается посредством бита TERR регистра BSICON в режиме удаленного терминала. Указывает на ошибки в работе удаленного терминала
–	7-5	–	Зарезервировано
Примечание – Ответное слово не передается после приема группового сообщения и в случае обнаружения ошибки в принятых данных.			

Прерывания

В режиме удаленного терминала формируются запросы на прерывания:

1 По окончании выполнения действий, предписанных полученным командным словом.

2 По ошибке:

- если возникла непрерывная генерация сигнала в канале передачи;
- если обнаружена ошибка длины команды, паритета или кода Манчестер II слова данных.

Примечание – Если запрос на прерывание формируется по ошибке, то одновременно с ним в регистре BSISTAT устанавливается флаг DATAERR.

Поскольку причин для формирования запроса на прерывание несколько, а вектор прерывания один, то при обслуживании запроса следует считывать состояние регистра BSISTAT для точного выявления события, вызвавшего прерывание.

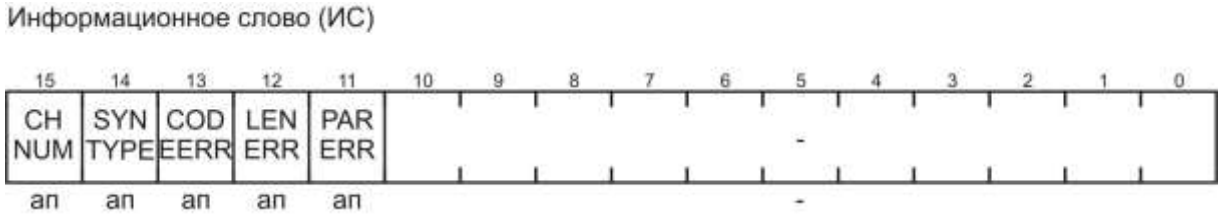
15.3 Монитор шины

Монитор шины непрерывно «слушает» оба канала (основной и резервный) и сохраняет в памяти результаты мониторинга. Монитор шины имеет собственный 32-разрядный таймер для отсчета времени от момента включения.

Объем области ОЗУ микроконтроллера, которая выделяется модулю магистрального последовательного интерфейса во время его работы в режиме монитора шины, составляет 4 Кбайта. Область расположена в диапазоне адресов от 0800h до 17FFh и имеет условное название – память слов. Если модуль не используется, указанная область может использоваться, как обыкновенное ОЗУ.

Каждое обнаруженное на шине слово запоминается монитором. Одновременно с этим монитор считывает состояние своего таймера и формирует информационное слово, которое содержит информацию о работе канала (см. таблицу 15.11).

Таблица 15.11 – Формат и назначение битов информационного слова



Поле	Бит	Тип	Описание	
CHNUM	15	Аппаратное влияние	Обнаруженный канал	
			0	Основной
			1	Резервный
SYNTYPE	14	Аппаратное влияние	Форма синхронизации	
			0	Командное слово
			1	Слово данных или ответное слово
CODE ERR	13	Аппаратное влияние	Ошибка кода Манчестер II. Флаг устанавливается в случае обнаружения ошибок кодирования передаваемой информации	
LEN ERR	12	Аппаратное влияние	Ошибка длины. Бит был установлен вследствие того, что длина принятого слова оказалась менее 20 бит	
PAR ERR	11	Аппаратное влияние	Ошибка паритета. Число, сформированное полученными битами, четное	
–	10-0	–	Зарезервировано	

Таким образом, формируется информационный блок, состоящий из четырех слов:

- старшее слово таймера;
- младшее слово таймера;
- «услышанное» слово;
- информационное слово.

Каждый такой блок из четырех слов последовательно записывается в область слов. Объем памяти позволяет записать 512 блоков. Пример заполнения памяти представлен на рисунке 15.4.

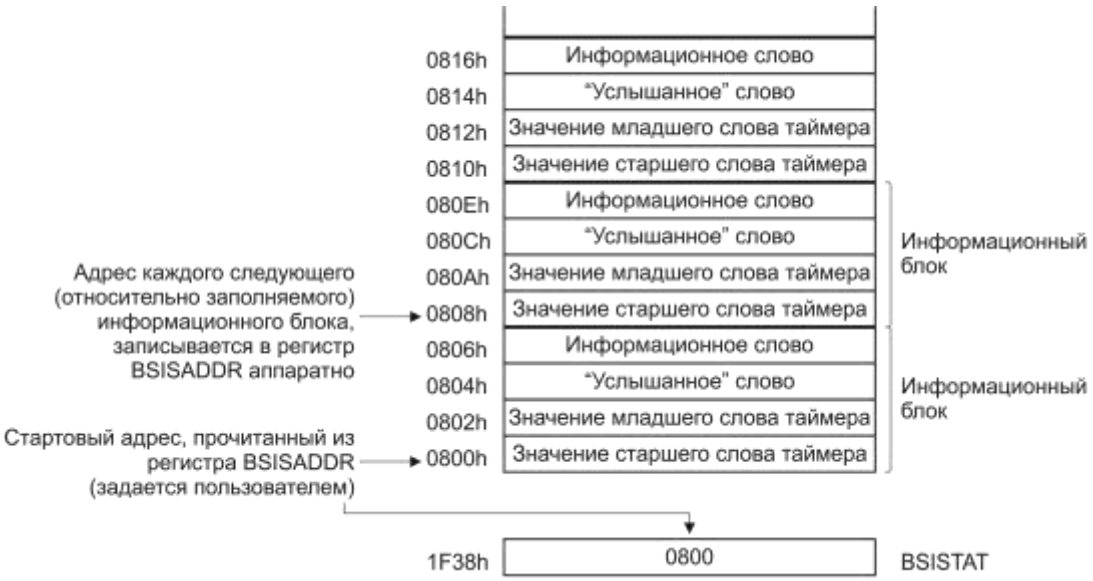


Рисунок 15.4 – Пример заполнения ОЗУ в режиме монитора шины

Пользователь может прочитать записанные данные, обращаясь к физическим адресам ячеек памяти слов, как к ОЗУ.

Инициализация

Перед включением монитора необходимо в регистре BSICONFIG в поле DIV задать значение делителя. При необходимости сброса монитора можно установить бит RST. Состояние поля ASKDELAY и бита TXDLEV не важно.

По умолчанию монитор записывает информацию в ОЗУ, начиная с адреса 0800h, но можно задать другой желаемый стартовый адрес (в пределах памяти слов) посредством регистра BSISADDR.

Монитор шины не имеет отдельного бита для включения, поэтому запись значения 11h в поле MODE сразу же запускает процесс мониторинга шины.

Регистр управления монитора шины содержит два битовых поля WORDNUM и FREESPACE.

Монитор записывает «услышанные» слова непрерывно. В некоторых случаях может потребоваться разделение всего записанного объема информации на группы слов. Например, для периодического чтения записанных данных из памяти слов без остановки монитора. Для этого используется поле WORDNUM. Число, записанное в битовое поле, является максимальным значением, до которого считает счетчик слов. Счетчик запускается сразу после включения монитора.

Как только счетчик достигает указанного в WORDNUM значения, формируется запрос на прерывание, а сам счетчик обнуляется. Одновременно с этим в регистр стартового адреса BSISADDR аппаратно загружается адрес начала следующего информационного блока. При желании можно изменить начальный адрес следующего информационного блока, записав в регистр BSISADDR другое значение. После этого в обработчике прерываний или основной программе нужно прочитать регистр статуса монитора BSISTAT. Это регистр содержит значение стартового адреса, с которого монитор начал заполнение памяти. Как только регистр состояния BSISTAT прочитан, в него аппаратно загружается значение из регистра BSISADDR, в противном случае, его состояние не меняется.

Примечание – Если требуется изменить стартовый адрес следующего информационного блока, запись в регистр BSISADDR следует производить до начала чтения регистра BSISTAT.

Далее можно прочитать записанные данные из памяти слов.

Следует помнить, что если режим монитора не выключен, то отслеживание слов и их запись не прерываются. Т.е. процессы чтения данных и записи новых идут

параллельно. И каждый раз, когда счетчик будет достигать значения, указанного в поле WORDNUM, будет формироваться запрос на прерывание, указывающий на то, что получена очередная группа слов. Желательно, чтобы к этому моменту чтение ранее записанных данных было закончено.

Для примера, можно обратиться к рисунку 15.5. В поле WORDNUM записано значение 04h. Поэтому через каждые четыре принятых слова будет формироваться запрос на прерывание.

Если в поле WORDNUM записано значение 00h, то подсчет принятых слов не производится.

Примечание – Описанный выше алгоритм работы с данными достаточно легко реализуется программно, поскольку чтение одного информационного блока в несколько раз выше скорости приема одного слова даже на невысоких частотах работы микроконтроллера.

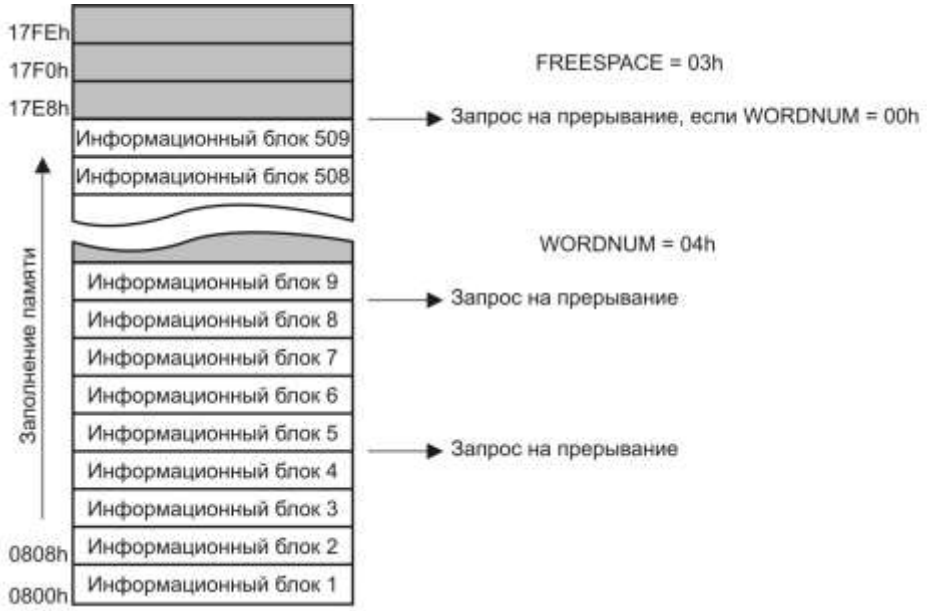


Рисунок 15.5 – Формирование запросов на прерывания при заполнении памяти

Может возникнуть ситуация, когда верхнюю границу памяти слов нужно изменить, определив новую границу с адресом, имеющим значение меньше, чем 17FFh. Например, для контроля заполнения памяти или сохранения каких-либо данных, записанных ранее.

Битовое поле FREESPACE позволяет задать количество слов, для записи которых еще остается место, т.е. оставшееся свободное пространство. Фактически, указанное количество слов будет определять количество свободных информационных блоков. Это следует помнить при расчете адресов размещения данных.

Механизм отслеживания свободного пространства запускается сразу после включения монитора. Неважно, с какого адреса началось заполнение памяти. Как только счетчик адреса достигнет значения, которое было аппаратно вычислено на основе заданного в FREESPACE значения, сформируется запрос на прерывание (см. рисунок 15.6).

Далее возможны два варианта:

1 Монитор не отключается и продолжает считывать информацию с шины. Каждое последующее «услышанное» слово записывается в память вместе с тремя дополнительными словами в очередной информационный блок. При каждой такой записи формируется запрос на прерывание (см. рисунок 15.6).

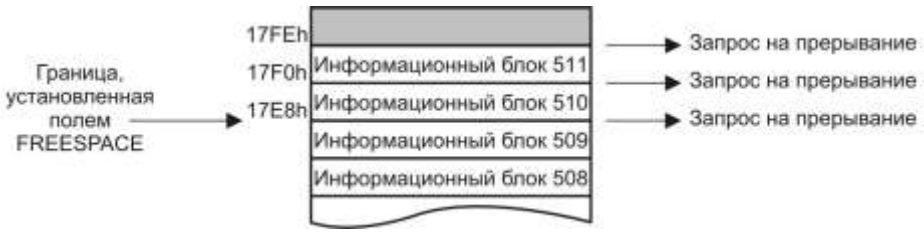


Рисунок 15.6 – Заполнение памяти слов

Поведение регистров BSISADDR и BSISTAT и управление ими аналогично тому, как описано ранее (для поля WORDNUM). После достижения верхней границы памяти, ее заполнение начнется сначала.

2 Монитор выключается записью значения 00h в поле MODE регистра BSICONFIG. Вся информация, расположенная выше значения, которое было аппаратно вычислено на основе заданного в FREESPACE значения, сохраняется.

Примечание – Битовые поля WORDNUM и FREESPACE регистра BSISTAT не могут использоваться одновременно. Если в поле WORDNUM записано любое значение, неравное 00h, то состояние второго поля игнорируется, в противном случае применяется значение поля FREESPACE. Если состояния обоих полей равны 00h, то подсчет «услышанных» слов не ведется, а запрос на прерывание формируется только при достижении верхней границы памяти слов.

Прерывания

Если значение поля WORDNUM не равно нулю, то запрос на прерывание будет формироваться только тогда, когда счетчик принятых слов будет достигать значения, записанного в поле WORDNUM, и ни при каких других обстоятельствах.

Если значение поля WORDNUM равно нулю, то запрос на прерывание будет формироваться по одной из причин:

а) счетчик адреса достиг значения, вычисленного на основе записанного в поле FREESPACE, числа слов;

б) обнаружена ошибка длины команды;

в) обнаружена ошибка паритета;

г) обнаружена ошибка кода Манчестер II принятого слова.

Если причиной прерывания является обнаруженная ошибка, то в информационном слове устанавливается флаг, соответствующий ошибке.

При обслуживании запроса на прерывание следует считывать состояние регистра BSISTAT для контроля полученной информации.

16 Контроллер интерфейса 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 31.07.2008г.

16.1 Инициализация и функционирование

Установление соединения инициируется записью единицы в бит LINKEN регистра управления SWCON. 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 не нормирован стандартом. Пакет включает поле заголовка, содержащее адрес назначения, и поле данных, замыкаемое маркером конца пакета.

Маршрутизаторы, которые обрабатывают 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 SWDATAT. Каждый байт, записанный в регистр SWDATAT, 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 SWCAPSTAT.

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 SWCAPSTAT.

Для чтения данных используется регистр SWDATAR. 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 SWTIMET. Код времени, который нужно передать, декрементируется на единицу и записывается в поле TRANSTIME. Бит TICK при этом устанавливать не следует. Т.е. в регистр SWTIMET 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 SWTIMET.

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 SWTIMER. 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 SWTIMER. Если разница значений последовательно принятых кодов превышает единицу, код времени считается ошибочным и 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 SWSTAT. Для управления прерываниями используется регистр SWINTMASK.

17 Средства отладки

17.1 Программные и аппаратные средства отладки

Для создания программного обеспечения рекомендуется использовать программный продукт CodeMaster-96.

Программный продукт CodeMaster-96 представляет собой набор программно-аппаратных средств, предназначенный для разработки и отладки систем на базе микроконтроллеров семейства AMCS-96 ОАО «НИИЭТ».

При работе с микроконтроллером 1874ВЕ7Т, 1874ВЕ71Т отладочная программа-монитор (в случае использования интерфейсов отладки SPI или UART) автоматически добавляется средой CodeMaster-96 в код проекта на этапе компиляции (в опциях проекта необходимо выбрать микроконтроллер, название которого заканчивается на «with Debug Monitor»). После этого достаточно записать сформированный файл с расширением .HEX во внешнюю память и подключить адаптер JEM-96 к микроконтроллеру.

Адаптер JEM-96 обеспечивает взаимодействие между интегрированной средой разработки CodeMaster-96, установленной на персональном компьютере, и отладочными ресурсами, встроенными в микроконтроллер 1874ВЕ7Т, 1874ВЕ71Т, а также выполнение отладочных функций.

Адаптер взаимодействует с микроконтроллером через интерфейсы:

- UART, SPI (в случае использования должна выполняться резидентная отладочно-загрузочная программа, располагающаяся во внешней памяти);
- JTAG (в случае использования резидентная программа не требуется).

Отладочная программа-монитор обеспечивает поддержку отладочных функций комплекса, для выполнения которых используется модуль отладки OCDS, входящий в состав микроконтроллера, а также поддерживает аппаратные и программные средства ограничения доступа к внутренним ресурсам микроконтроллера.

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.

При работе с 1874ВЕ7Т, 1874ВЕ71Т в режиме отладки посредством интерфейсов SPI или UART выводы микроконтроллеров P0.5 и P0.4 (PMODE) должны всегда находиться в состоянии, соответствующем используемому отладочному порту, и эти выводы не могут быть использованы пользовательской программой. Вывод P0.6 (DBG#) также не может быть задействован пользовательской программой, т.к. он используется отладчиком для входа в отладочную программу-монитор во время сброса.

В зависимости от выбранного интерфйса отладки, контакты разъема адаптера JEM-96 подключаются к микроконтроллеру в соответствии с таблицей 17.1.

Таблица 17.1 – Интерфейс отладки

Контакт адаптера JEM-96		Вывод микроконтроллера1874BE7T, 1874BE71T при отладке через интерфейс		
№	Название	SPI	UART0/1	JTAG
3	DCS - разрешение работы по SPI	P1.2 (SS#)	Не используется	Не используется
4	DBG# - выбор стартового адреса в памяти программ	P0.6	Не используется	Не используется
5	DIN/TDI - вход последовательных данных	P1.5 (MOSI)	P2.1 (RXD0) для UART0 или P2.4 (RXD1) для UART1	P5.1 (TDI)
7	NMI/TMS - прерывание пользовательской программы в произвольный момент времени	NMI	NMI	P5.2 (TMS)
9	DCLK/TCK - тактовая частота	P1.7 (SCK)	Не используется	P5.7 (TCK)
11	DOUT/TDO - выход последовательных данных	P1.6 (MISO)	P2.0 (TXD0) для UART0 или P2.6 (TXD1) для UART1	P5.6 (TDO)
1, 2	VCCtrg - напряжение питания +3.3 В	#VCC1		
13	RST# - сброс	RESET#		
6, 8, 10, 12, 14	GND - общий	#0V1		
Примечание – Выбор режима отладочной программы при работе через SPI и UART определяется состоянием выводов P0.5 и P0.4 (PMODE): - P0.5 = 0, P0.4 = 1 – режим DEBUG-SPI; - P0.5 = 1, P0.4 = 0 – режим DEBUG-UART0; - P0.5 = 1, P0.4 = 1 – режим DEBUG-UART1.				

Ниже приводится алгоритм формирования HEX-файла с отладочным монитором для микроконтроллеров 1874BE7T, 1874BE71T (при отладке по SPI или UART):

1 В свойствах проекта выбрать микроконтроллер «K1874BE7T with Debug Monitor», ниже в поле «Физическая память отлаживаемой системы» добавить область памяти, соответствующую используемой внешней памяти, в области «External Flash» (например, 0x2000–0xFFFF).

2 Выбрать в качестве отладчика – симулятор.

3. В меню «Компиляция» выбрать «Собрать проект и запустить отладку [Shift+F7]».

4 Во внешнюю флеш записать полученный HEX-файл и установить микросхему памяти на отладочную плату.

5 Установить конфигурацию на выводах P0.5 и P0.4 в соответствии с выбранным режимом интерфейса отладки (SPI/UART0/UART1).

6 Подключить JEM-96 к разъему USB персонального компьютера.

7 Включить питание на отладочной плате.

8 В среде CodeMaster-96 в качестве отладчика выбрать JEM-96.

9 В опциях отладки указать выбранный интерфейс отладки (SPI или UART).

17.2 Модуль отладки OCDS

Модуль 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 того или иного из указанных события может формироваться отклик:

- прерывание DEBUG (2060h);
- переход в режим IDLE;
- аппаратный сброс микроконтроллера;

Для задания адресов, байт/слов данных и значений счетчика команд используются регистры модуля отладки.

Регистр DEBADDR

Используется, если требуется отследить появление на шине адресов желаемого значения. Как только значение слова адреса операнда на шине и значение, записанное в поле DEBADDRVAL, совпадут, это будет считаться событием. Для программирования отклика используется поле DEBADDREN регистра DEBCTRL.

Регистр DEBDATA

Используется, если требуется отследить появление на внутренней шине данных желаемого значения. Как только значение слова данных на шине и значение, записанное в поле DEBDATAWORD, совпадут, это будет считаться событием. Для программирования отклика используется поле DEBDATAEN регистра DEBCTRL.

Регистр DEBDATH и DEBDATL

Используются, если требуется отследить появление на внутренней шине данных желаемого значения только старшего или только младшего байта данных, что будет являться событием. Для программирования откликов используются поля DEBDATHEN и DEBDATLEN регистра DEBCTRL.

Регистры DEBPCEQ и DEBPCEQ2

Используются, если требуется отследить момент, когда счетчик команд достигнет желаемого значения. Как только значение счетчика команд и значение, записанное в поле DEBPCEQUAL, совпадут, это будет считаться событием. Для программирования отклика используется поле DEBPCEQEN регистра DEBCTRL.

Регистр DEBPC

Используется, если требуется отследить момент, когда счетчик команд превысит желаемое значение. Как только значение счетчика команд станет больше значения, записанного в поле DEBPCLESS, это будет считаться событием. Для программирования отклика используется поле DEBPCEN регистра DEBCTRL.

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 EXTMEMEN регистра DEBCTRL. Поскольку по умолчанию значение поля EXTMEMEN равно 00b, то логика модуля отладки не будет воспринимать обращение к внешней памяти как событие до тех пор, пока состояние поля не будет перепрограммировано.

Функционирование модуля отладки и управление его работой

Для программирования откликов на события используется регистр DEBCTRL. По умолчанию состояние регистра равно 0000h, что запрещает любые действия модуля отладки. Логика модуля OCDS начинает отслеживать желаемое событие сразу же после того, как для этого события будет запрограммирован отклик (т.е. значение соответствующего поля не равно 00b).

Для наглядности на рисунке 17.1 представлена структурная схема логики модуля OCDS, позволяющая формировать программируемые отклики на желаемые события.

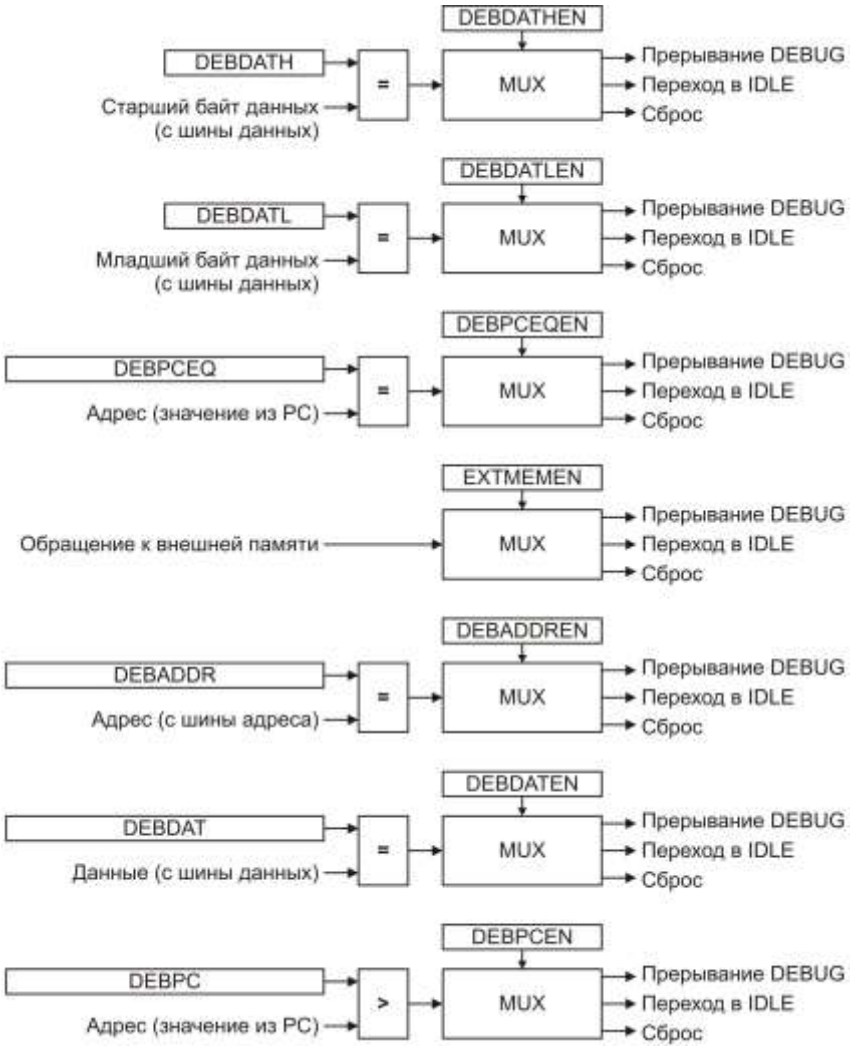


Рисунок 17.1 – Структурная схема работы модуля отладки

Семь линий откликов «прерывание DEBUG» далее объединяются по ИЛИ и формируют, таким образом, одну линию прерывания DEBUG, показанную на рисунке 17.2. Аналогично объединяются между собой линии «переход в IDLE» и «Сброс».

Прерывание DEBUG имеет самый высший приоритет и является немаскируемым. Оно не может назначаться на PTS. Время отклика на запрос определяется так же, как и

для стандартных прерываний. Данное прерывание имеет бит ожидания, но он не доступен для записи и чтения.

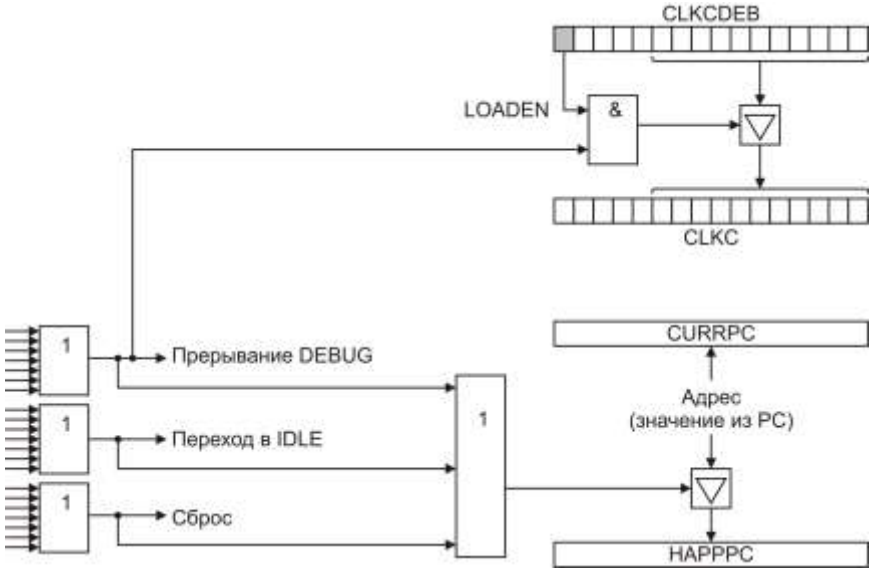


Рисунок 17.2 – Дополнительная структурная схема работы модуля отладки

Регистр HAPPPC

Из схемы на рисунке 17.2 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

На рисунке 17.2 также указан регистр CURRPC. Значение, находящееся в этом регистре, всегда совпадает со значением счетчика команд.

Управление синхронизацией периферийных устройств

Одновременно с возникновением прерывания DEBUG модуля отладки может осуществляться перезапись регистра CLKC (регистр управления тактовыми сигналами периферийных устройств), что позволяет отключать/включать периферию. При этом управляющие регистры и регистры состояний будут сохранять свои значения, которые были на момент отключения.

Следует помнить, что после отключения блока его регистры доступны только для чтения.

Для того, чтобы осуществить перезапись регистра CLKC, необходимо установить бит LOADEN регистра CLKCDEB. Далее, как только будет сформировано прерывание DEBUG, логика модуля OCDS побитно перезагрузит регистр CLKC значением, которое хранится в регистре CLKCDEB (см. рисунок 17.2).

18 Сторожевой таймер

Сторожевой таймер (или WDT) позволяет восстанавливать нормальную работу при сбоях программ. Если работа WDT разрешена, он будет вызывать аппаратный сброс, если программа не очищает его до истечения времени выполнения 64К машинных циклов.

Когда таймер переполняется, он переводит линию RESET в состояние низкого уровня не менее, чем на два машинных цикла, сбрасывая микроконтроллер и другие устройства, подключенные к его выводу RESET#.

Для очистки и включения сторожевого таймера следует последовательно друг за другом записать значения 1Eh и E1h в регистр WATCHDOG. Регистр счетчика таймера является 16-разрядным, но для пользователя доступен только его старший байт, получить который можно посредством чтения регистра WATCHDOG.

Сразу после записи значений 1Eh и E1h в регистр WATCHDOG, в счетчик сторожевого таймера будет аппаратно загружено значение 0000h, которое далее будет инкрементироваться каждый машинный цикл. Для программного сброса сторожевого таймера следует своевременно записывать значения 1Eh и E1h в регистр WATCHDOG.

После инициализации сторожевой таймер может быть выключен только аппаратно, после сброса микроконтроллера.

19 Специальные режимы работы микроконтроллера

Микроконтроллер поддерживает четыре специальных режима 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 – замедляется работа как ЦПУ, так и периферийных устройств за счет увеличения длительности машинного цикла в два раза. Может включаться и выключаться программно без сброса микроконтроллера.

Режим Once – микроконтроллер электрически изолирован от других устройств на печатной плате.

Помимо этих режимов, существует возможность независимого отключения тактовых сигналов периферийных устройств (см. раздел 8). Это позволяет гибко управлять общим потреблением, используя наиболее оптимальный режим в каждый момент времени работы системы.

19.1 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, последовательный порт, сервер периферийных транзакций и система прерываний продолжают функционировать. Потребление энергии уменьшается по сравнению с нормальным режимом работы.

Выводы управления шиной (ALE, RD#, WR#, INST и BHE#) переведены в неактивное состояние. Если порты 3 и 4 использовались, как порты ввода-вывода, они будут сохранять значения, записанные в их защелках. Если эти порты использовались как шина адрес/данные, то выводы будут «плавающими». Исключение – использование PTS передач во внешнюю память.

Если таймер WDT разрешен, то он продолжает работать, поэтому микроконтроллер должен выходить из режима Idle каждые 64К такта, чтобы сбросить таймер.

Вход в режим Idle

Осуществляется по команде IDLPD #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 (в том числе во внешнюю память), не выходя из режима Idle. При этом во время передачи выводы управления внешней шиной 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 IDLPD #1.

19.2 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 (SFR) и регистры ОЗУ сохраняют данные.

Выводы управления шиной (ALE, RD#, WR#, INST и BHE#) переведены в неактивное состояние. Все выходы сохраняют значения, находящиеся в их защелках. Если порты 3 и 4 использовались как порты ввода-вывода (EA# = 0), их выводы будут удерживать последнее выведенное значение. Если порты 3 и 4 использовались как шина адрес/данные (EA# = 1), их выводы будут «плавающими».

Запрещение режима Powerdown

Режим запрещен, когда бит PD регистра CCR сброшен.

Переход в режим Powerdown

До перехода в режим обязательно должен быть завершен обмен через последовательный порт 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 WDT может сбросить микроконтроллер до стабилизации тактового генератора. В режиме Powerdown счетчик WDT не меняет своего значения.

Переход в режим Powerdown осуществляется по команде IDLPD #2.

Примечание – На выводе EXTINT должен удерживаться низкий уровень сигнала, пока устройство находится в режиме Powerdown.

Выход из режима Powerdown

Выход из режима Powerdown обеспечивается одним из трех способов.

1 Удержание низкого уровня сигнала на выводе VPR в течение как минимум 50 нс приведет к разблокированию внутреннего генератора и возобновлению тактирования микроконтроллера.

2 Удержание высокого уровня сигнала на выводе EXTINT в течение как минимум 50 нс. Вывод EXTINT обычно используется как вход внешнего прерывания. В режиме Powerdown он используется как вход, чувствительный к уровню. Бит EXTINT_SRC регистра IOC1 определяет, какой из двух входов EXTINT будет использоваться. Сброшенный бит EXTINT_SRC выбирает вывод P2.2, а установленный – P0.7.

В обоих случаях не будет сброса микроконтроллера, а продолжится выполнение программы с команды, следующей за командой IDLPD #2.

3 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.

19.3 Режим Slow

Микроконтроллер имеет дополнительный режим уменьшения энергопотребления, при котором внутренняя частота уменьшается в два раза и длительность машинного цикла становится равной четырем тактам сигнала XTAL. Это требует пересчета временных соотношений для периферийных устройств, поскольку длительности всех тактовых сигналов увеличатся в два раза.

Вход в режим Slow

Осуществляется установкой бита SLOW регистра CLKC.

Выход из режима Slow

Осуществляется сбросом бита SLOW регистра CLKC**.**

Данная функция может быть использована на определенных этапах выполнения программы или в случае, если все вычислительные возможности необходимы только в определенные моменты времени.

В режиме Slow доступны режимы Idle и Powerdown, механизмы работы всех периферийных блоков не меняются. Вход и выход из режима Slow занимает один машинный цикл и может осуществляться в любой момент времени.

19.4 Режим Once

Режим тестирования, который электрически изолирует микроконтроллер от других устройств на печатной плате. Все выводы находятся в режиме высокого сопротивления (кроме квазидвунаправленных выводов, которые находятся в режиме слабой единицы), а схема – в режиме Powerdown. В режиме Once на выводе RESET# должен удерживаться высокий уровень сигнала.

Вход в режим Once

Осуществляется, если во время нарастающего фронта сигнала #RESET удерживать низкий уровень сигнала на выводе P2.0.

Выход из режима Once

Осуществляется подачей сигнала RESET#. Вывод P2.0 при этом может быть неподключенным или находиться в состоянии логической единицы.

Для уменьшения энергопотребления в режиме Once одновременно с электрической изоляцией происходит прекращение внутреннего тактирования и выключение генератора (включается режим Powerdown).

Следует помнить, что контроллер может выйти из режима Powerdown, но при этом оставаться в режиме электрической изоляции. Чтобы этого не произошло, необходимо удерживать вывод VPR в состоянии логической единицы, а вывод P2.2 в состоянии логического нуля.

20 Интерфейс внешней памяти

Микроконтроллеры допускают подключение различных устройств, реализующих функцию внешней памяти по 8/16-разрядному программируемому интерфейсу с внутренним контролем готовности для медленных устройств.

Подключение внешней памяти и контроль ее функционирования осуществляется посредством выводов и сигналов, указанных в таблице 20.1. Название сигнала совпадает с названием вывода микроконтроллера. Если сигнал имеет бит управления (бит регистра CCR), то этот бит и его состояние указаны под названием вывода микроконтроллера в скобках.

Таблица 20.1 – Сигналы интерфейса внешней памяти

Вывод микро-контроллера	Дополни-тельная функция	Описание сигнала
1	2	3
P4.7 – P4.0 P3.7 – P3.0	AD15 –AD8 AD7 – AD0	Системная 8/16-разрядная мультиплексированная шина адреса/данных. В течение адресной фазы шинного цикла адрес выставляется на шину и фиксируется во внешних устройствах по сигналу ALE/ADV#
ALE (ALE = 1b)		Выходной сигнал с высоким активным уровнем. Информирует о том, что на шине установлен и доступен верный адрес и начался шинный цикл. Сигнал ALE активен только в начале шинного цикла, а потом он переходит в неактивное состояние. Сигнал ALE может использоваться внешним устройством как стробирующий сигнал захвата адреса в регистр-защелку
	ADV# (ALE = 0b)	Выходной сигнал с низким активным уровнем. Информирует о том, что на шине установлен и доступен верный адрес и начался шинный цикл. В отличие от ALE, сигнал ADV# активен в течение всего шинного цикла. Сигнал ADV# может использоваться внешним устройством как стробирующий сигнал захвата адреса в регистр-защелку
BHE# (WR = 1b)		Выходной сигнал с низким активным уровнем. Становится активным только во время записи слова или отдельно старшего байта во внешнюю память. Сигнал BHE# используется одновременно с нулевым разрядом адреса A0
	WRH# (WR = 0b)	Выходной сигнал с низким активным уровнем. В режиме 16-разрядной шины становится активным во время записи слова или отдельно старшего байта. В режиме 8-разрядной шины становится активным во время записи слова или отдельно старшего/младшего байта
BW (BW0 = 1b)	–	Входной сигнал задания разрядности внешней шины, если установлен бит BW0 регистра CCR. Высокий уровень сигнала BW устанавливает 16-разрядную шину, низкий – 8-разрядную. Если бит BW0 сброшен, то шина автоматически устанавливается как 8-разрядная независимо от сигнала BW

Окончание таблицы 20.1

1	2	3
EA#	–	Входной сигнал с низким активным уровнем. В активном состоянии разрешает доступ к внешней памяти. Поскольку микроконтроллер не имеет внутренней памяти программ, вывод EA# желательно заземлить
INST	–	Выходной сигнал с высоким активным уровнем. Становится активным только во время цикла чтения внешней памяти. Высокий уровень сигнала INST информирует о том, что происходит выборка команды, низкий – чтение данных. Сигнал INST может использоваться в приложениях, когда требуется независимость областей памяти команд и данных (байт конфигурации CCB и векторы прерываний считаются данными)
RD#	–	Выходной сигнал с низким активным уровнем. Становится активным во время чтения внешней памяти
READY	–	Входной сигнал с высоким активным уровнем, используемый при работе с медленными внешними устройствами. В активном состоянии информирует микроконтроллер о том, что внешняя память готова для передачи или получения данных
WR# (WR = 1b)		Выходной сигнал с низким активным уровнем. Становится активным во время записи внешней памяти
	WRL# (WR = 0b)	Выходной сигнал с низким активным уровнем. В режиме 16-разрядной шины становится активным во время записи слова или отдельно младшего байта. В режиме 8-разрядной шины становится активным во время записи слова или отдельно старшего/младшего байта

20.1 Регистр CCR и конфигурирование шины

Первый байт, выбираемый из внешней памяти (EA# = 0) из ячейки с адресом 2018h после сброса микроконтроллера, является байтом конфигурации кристалла CCB (байт обычно программируется однократно, когда пользовательская программа откомпилирована и не переопределяется во время обычных операций). Этот байт загружается в регистр CCR, который является регистром конфигурации кристалла. Однажды загруженный, после сброса, регистр CCR не может быть изменен до следующего сброса микроконтроллера. Регистр CCR управляет режимом READY работы внешних устройств, разрядностью и режимом работы внешней шины управления, а также контролирует возможность перехода микроконтроллера в режим пониженного энергопотребления Powerdown.

Если на выводе READY удерживается низкий уровень сигнала во время загрузки регистра CCR, контроллер шины автоматически вставляет три цикла ожидания, что позволяет производить выборку байта CCB из медленной внешней памяти. Количество циклов ожидания можно запрограммировать посредством битов IRC1 и IRC0.

При загрузке байта CCB неважно, какой используется режим включения внешней памяти, так как временная диаграмма загрузки байта с четного адреса и в 16-разрядном, и в 8-разрядном включении одинакова.

Конфигурирование шины

Внешняя шина микроконтроллера является динамической и позволяет изменять разрядность во время работы. Возможностью изменения разрядности шины управляет бит

BW0 регистра CCR. Если этот бит сброшен, то шина аппаратно конфигурируется как фиксированная 8-разрядная. В дальнейшем разрядность шины не может быть изменена до сброса микроконтроллера и изменения состояния бита BW0 (см. рисунок 20.1).

Если бит BW0 установлен, то конфигурация шины будет зависеть от состояния сигнала на входе BW. Низкий уровень сигнала будет включать 8-разрядный режим шины с использованием выводов порта 3. Порт 4 в этом случае может функционировать как порт общего назначения. Высокий уровень сигнала на входе BW будет включать 16-разрядный режим шины (см. рисунок 20.1). Переключение уровня сигнала BW нужно производить во время неактивного уровня сигнала RD#/WR#.

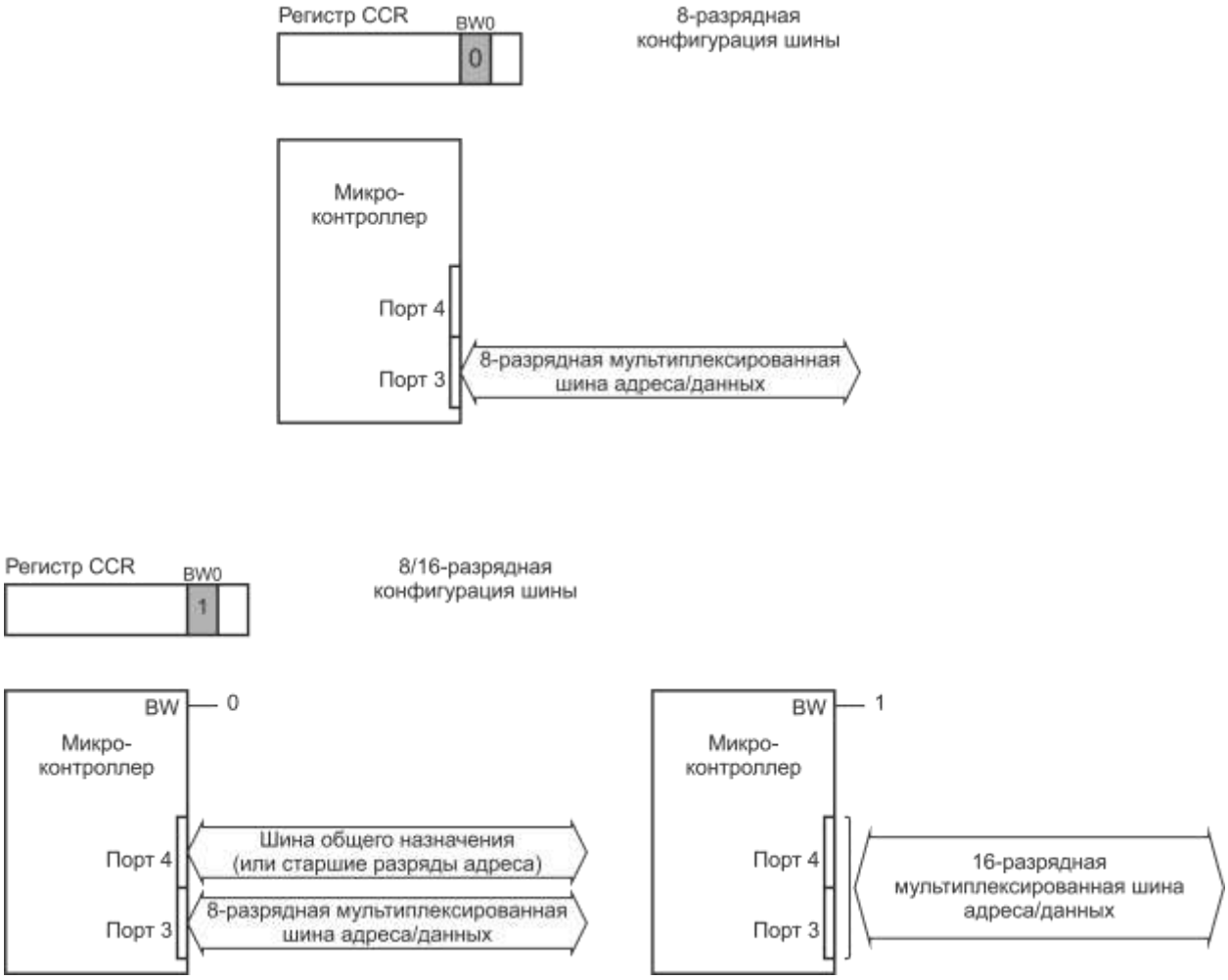


Рисунок 20.1 – Возможные варианты конфигурации шины адреса/данных

Динамическая конструкция шины расширяет границы аппаратного взаимодействия устройств. Например, если микроконтроллер взаимодействует с двумя внешними устройствами памяти – 16-разрядной памятью программ и 8-разрядной памятью данных, – а сигнал BW дополнительно заведен на входы «выбора» этих устройств, то переключение сигнала BW будет попеременно активировать память программ и память данных и соответственно менять разрядность шины.

Следует помнить, что в случае работы с 8-разрядной шиной будет происходить снижение производительности, поскольку для записи/чтения слова потребуется дополнительный шинный цикл, а предварительная выборка в очередь команд будет недоиспользована.

16-разрядная шина

Используются выводы порта 4 для передачи старших разрядов адреса/данных и выводы порта 3 для передачи младших разрядов адреса/данных мультиплексированной шины AD15-0. Упрощенные временные диаграммы для внешних циклов записи и чтения показаны на рисунке 20.2а.

По сигналу ALE на шине выставляется адрес, который запоминается в регистре-защелке. До тех пор, пока сигнал ALE в активном состоянии, на шине находится верный адрес.

В цикле чтения после сброса сигнала ALE контроллер шины переключает сигнал RD# в активное состояние. По ниспадающему фронту сигнала RD# устройство внешней памяти должно выставить данные на шину. Активный уровень сигнала INST во время чтения указывает на то, что идет выборка команды.

В цикле записи после сброса сигнала ALE контроллер шины переключает сигнал WR# в активное состояние. После обнаружения ниспадающего фронта сигнала WR# устройство внешней памяти должно начать считывание данных с шины.

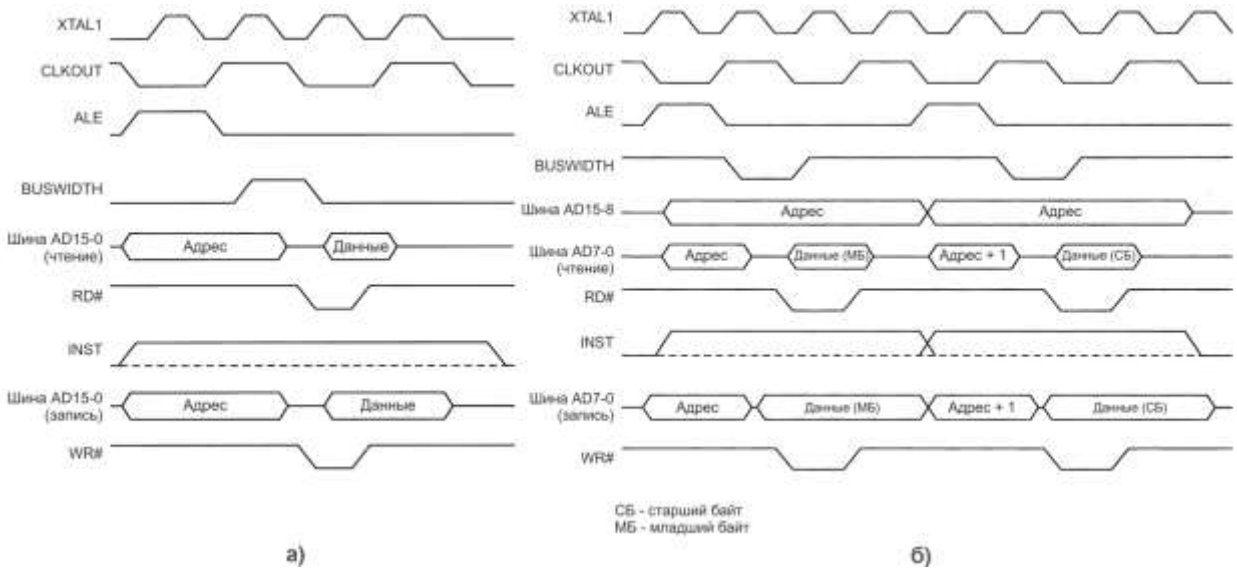


Рисунок 20.2 – Упрощенные временные диаграммы обмена: а) по 16-разрядной внешней шине, б) по 8-разрядной внешней шине

8-разрядная шина

Используются выводы порта 3 для передачи младших разрядов адреса/данных мультиплексированной шины AD7-0. Линии AD15-8 не мультиплексируются. Старшие разряды адреса не меняются в течение всего шинного цикла. Упрощенные временные диаграммы для внешних циклов записи и чтения показаны на рисунке 20.2б.

По сигналу ALE на шине выставляется адрес, который запоминается в регистре-защелке. До тех пор, пока сигнал ALE в активном состоянии, на шине находится верный адрес.

В цикле чтения после сброса сигнала ALE контроллер шины переключает сигнал RD# в активное состояние. По ниспадающему фронту сигнала RD# устройство внешней памяти должно выставить данные на шину. При чтении слова контроллер шины выполняет два цикла чтения последовательно – сначала принимается младший байт, а затем старший. Активный уровень сигнала INST во время чтения указывает на то, что идет выборка команды.

В цикле записи после сброса сигнала ALE контроллер шины переключает сигнал WR# в активное состояние. После обнаружения ниспадающего фронта сигнала WR# устройство внешней памяти должно начать считывание данных с шины. При записи слова

контроллер шины выполняет два цикла записи последовательно – сначала передается младший байт, а затем старший.

Длительность нормального шинного цикла чтения/записи составляет два машинных такта (два такта сигнала CLKOUT).

Управление длительностью шинного цикла

При работе с медленными внешними устройствами памяти может потребоваться увеличение длительности шинного цикла записи/чтения. Микроконтроллер имеет специальный вывод READY, который соединяется с внешним устройством памяти или другим устройством, контролирующим работу памяти. Когда внешнему устройству требуется дополнительное время для готовности к доступу, оно переводит сигнал READY в низкий уровень и удерживает его в таком состоянии до полной готовности. Как только на выводе READY обнаруживается низкий уровень сигнала, контроллер шины начинает вставлять дополнительные циклы ожидания (см. рисунок 20.3). Каждый цикл ожидания равен одному машинному циклу. Количество циклов задается битами IRC1 и IRC0 регистра конфигурации кристалла CCR. Если заранее неизвестно, какое количество циклов ожидания может потребоваться, то следует установить оба бита. В этом случае контроллер шины будет вставлять дополнительные циклы ожидания до тех пор, пока на выводе READY будет удерживаться низкий уровень сигнала.

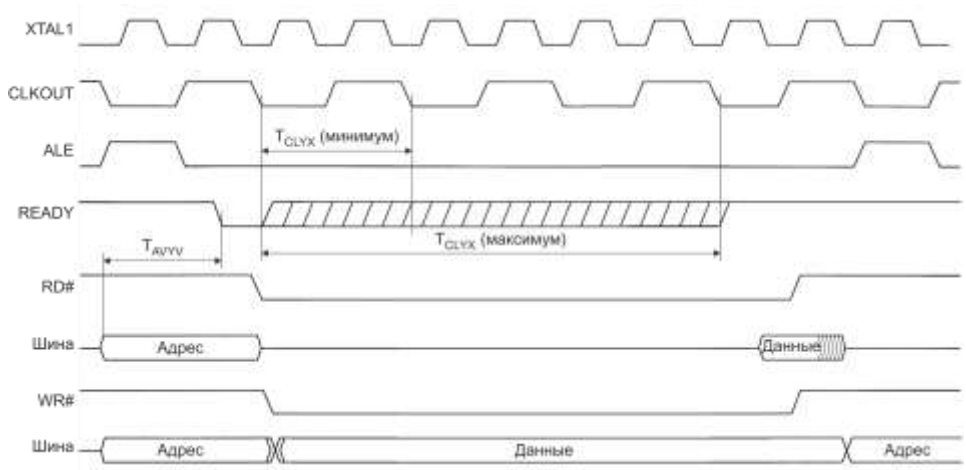


Рисунок 20.3 – Временные диаграммы для сигнала READY

Если необходимо, чтобы в каждый цикл записи/чтения вставлялась задержка в один, два или три цикла ожидания (в зависимости от состояния битов IRC1 и IRC0), то постоянное переключение сигнала READY не требуется – достаточно постоянно удерживать низкий уровень. В противном случае на сигнал READY накладывается ограничение. Проверка состояния вывода READY происходит после выставления адреса на шину по сигналу ALE в промежуток времени T_{AVYV} (см. рисунок 20.3), поэтому переключение в низкий уровень сигнала READY должно происходить не позднее окончания этого временного промежутка. Далее низкий уровень сигнала должен удерживаться до тех пор, пока не будут вставлены все циклы ожидания, т.е. переключение в высокий уровень должно происходить только по окончании временного промежутка T_{CLYX} (см. рисунок 20.3).

20.2 Управление шиной

В зависимости от того, какие группы сигналов управляют шинными циклами чтения/записи, реализуются четыре режима управления шиной. Каждому режиму соответствует своя комбинация состояний битов ALE и WR регистра CCR (см. таблицу 20.2).

Таблица 20.2 – Режимы управления шиной

Режим	Биты регистра CCR		Активные сигналы управления шиной			
	ALE	WR				
Стандартный	1	1	ALE	RD#	WR#	BHE#
Строб записи	1	0	ALE	RD#	WRL#	WRH#
Строб достоверного адреса	0	1	ADV#	RD#	WR#	BHE#
Строб достоверного адреса и записи	0	0	ADV#	RD#	WRL#	WRH#

Стандартный режим

По сигналу ALE на шине устанавливается адрес, сигнал BHE# выбирает область памяти, которая адресуется старшим байтом адреса. Для записи используются сигналы BHE# и WR# (см. рисунок 20.4). Для чтения используется сигнал RD#.

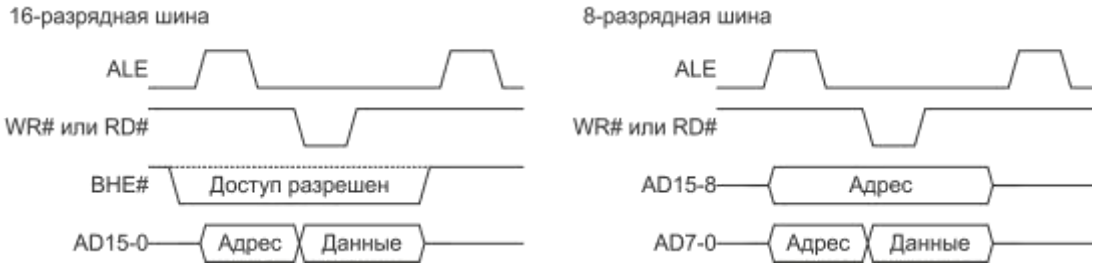


Рисунок 20.4 – Стандартный режим управления шиной

При 16-разрядной конфигурации шины возможна как запись слова данных, так и отдельно младшего или старшего байта. Для этого используются два сигнала записи WRH# и WRL#, передаваемые через выводы BHE# и WR# соответственно. Схема формирования этих сигналов показана на рисунке 20.5.

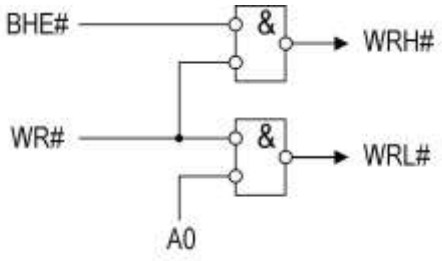


Рисунок 20.5 – Дешифрация сигналов WRL# и WRH#

Для чтения данных двух сигналов не требуется – достаточно одного сигнала RD#. При чтении на шину всегда выставляется слово, а в случае, если нужен только один байт, то второй просто отбрасывается. На рисунке 20.6а показана схема взаимодействия микроконтроллера и внешней памяти с использованием 8-разрядной шины.

Старший разряд адреса (A15) является сигналом выбора устройства. Когда сигнал A15 имеет высокий уровень, выбирается ОЗУ, иначе – ПЗУ. Сигнал ALE дополнительно является стробирующим сигналом для защелкивания адреса (A7-0) в регистре.

На рисунке 20.6б показана схема взаимодействия микроконтроллера и внешней памяти с использованием динамической16-разрядной шины.

Старший разряд адреса (A15) является сигналом выбора устройства. Когда сигнал A15 имеет высокий уровень, выбирается ОЗУ, иначе – ПЗУ. Кроме того, он управляет разрядностью шины, поскольку подключен на вход BW. Сигнал ALE дополнительно является стробирующим сигналом для защелкивания адреса (A15-0) в регистрах.

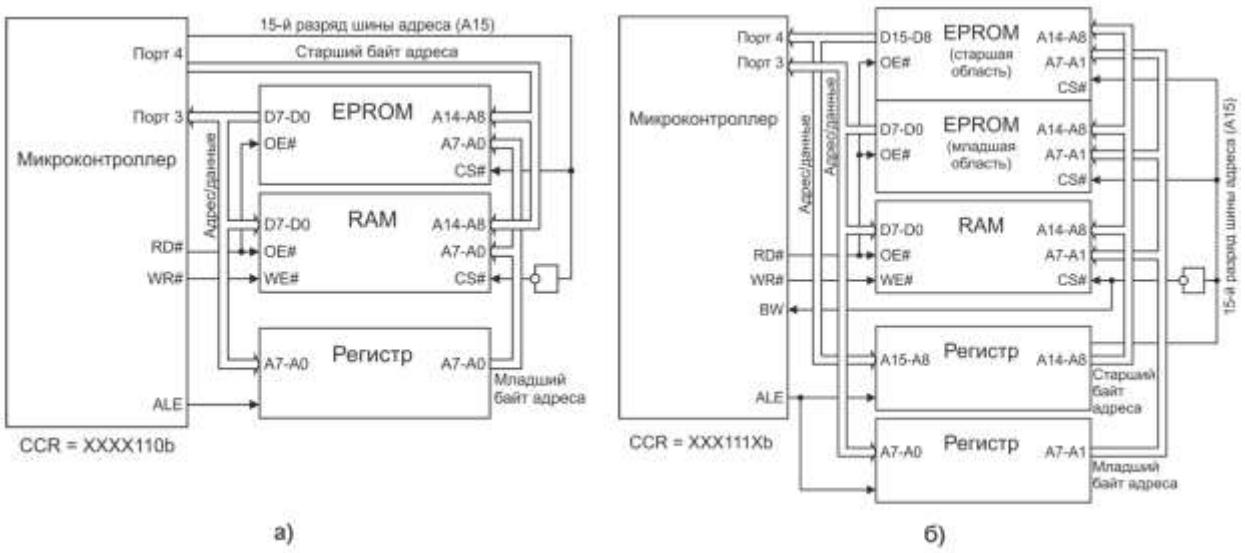


Рисунок 20.6 – Взаимодействие с внешней памятью с использованием: а) 8-разрядной шины, б) динамической 16-разрядной шины

Режим строба записи

В этом режиме отсутствует внешняя дешифрация адреса при записи старшего и младшего байт данных во внешнюю память при 16-разрядной конфигурации шины. По сигналу ALE на шине устанавливается адрес. Для записи слова или только младшего байта (с четным адресом) используется сигнал WRL#. Для записи слова или только старшего байта (с нечетным адресом) используется сигнал WRH#. При 8-разрядной конфигурации шины сигналы WRL# и WRH# устанавливаются одновременно как для четных, так и для нечетных адресов (см. рисунок 20.7). Для чтения используется сигнал RD#.

Рисунок 20.7 – Режим строба записи

На рисунке 20.8 показана схема взаимодействия микроконтроллера и внешней памяти с использованием динамической 16-разрядной шины.

Старший разряд адреса (A15) является сигналом выбора устройства. Когда сигнал A15 имеет высокий уровень, выбирается ОЗУ, иначе – ПЗУ. Сигнал ALE дополнительно является стробирующим сигналом для защелкивания адреса (A15-0) в регистрах. Младший разряд адреса (A0) не используется ОЗУ, поскольку записью старшего и младшего байт управляют сигналы WRH# и WRL#.

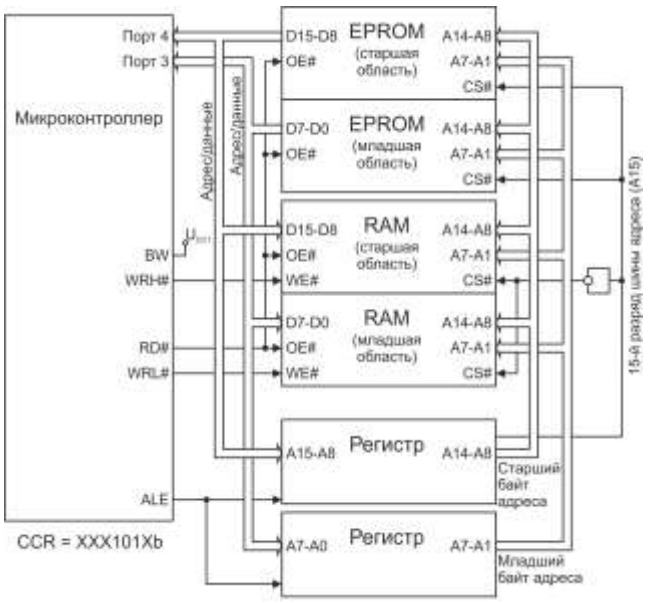


Рисунок 20.83 – Взаимодействие с внешней памятью с использованием динамической 16-разрядной шины

Режим строба достоверного адреса

В этом режиме вместо сигнала ALE используется сигнал ADV#, который становится активным после того, как на шине установится адрес (см. рисунок 20.9). Сигнал BHE# выбирает область памяти, которая адресуется старшим байтом адреса. Для записи используются сигналы BHE# и WR#. Для чтения используется сигнал RD#.

Рисунок 20.9 – Режим строба достоверного адреса

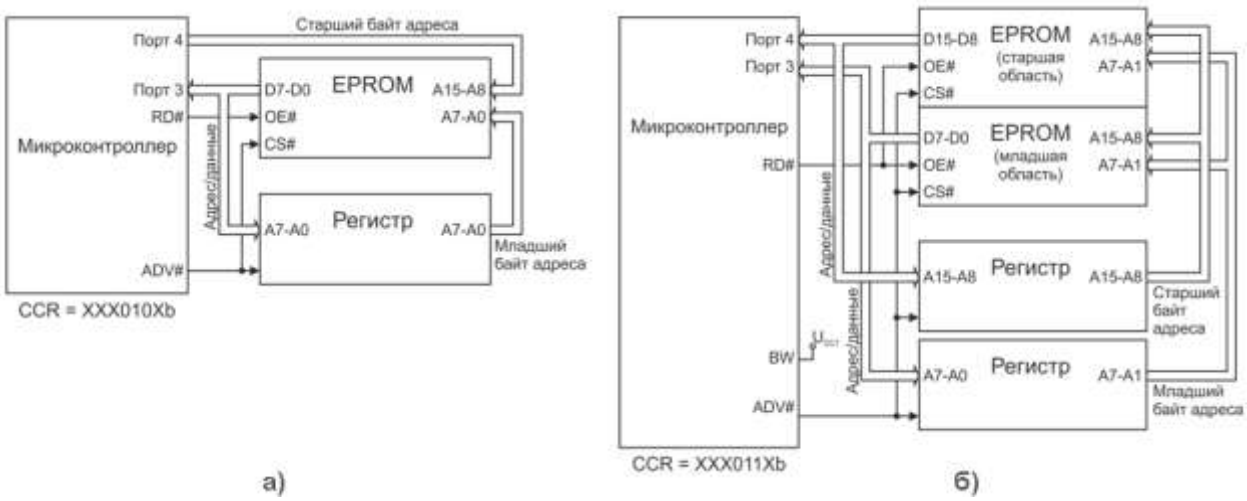


Рисунок 20.10 – Взаимодействие с внешним ПЗУ с использованием: а) 8-разрядной шины, б) динамической 16-разрядной шины.

Основное отличие работы сигнала ADV# от сигнала ALE в том, что он активен в течение всего шинного цикла. Это позволяет использовать его не только как стробирующий сигнал для защелкивания адреса, а также как сигнал выбора устройства внешней памяти. На рисунке 20.10 показаны схемы взаимодействия микроконтроллера и внешнего ПЗУ с использованием 8-разрядной шины и 16-разрядной динамической шины.

Когда шинные циклы взаимодействия с внешней памятью следуют последовательно друг за другом, вид сигналов ALE и ADV# идентичен. Различие становится очевидным во время простоя шины, поскольку в это время сигнал ADV# имеет высокий неактивный уровень (см. рисунок 20.11).

Рисунок 20.41 – Поведения сигналов ALE и ADV#

Режим строба достоверного адреса и записи

В этом режиме используются сигналы ADV#, WRH# и WRL#, RD# (см. рисунок 20.12). Режим применяется в простых системах взаимодействия микроконтроллера и внешних устройств с использованием 16-разрядной шины.

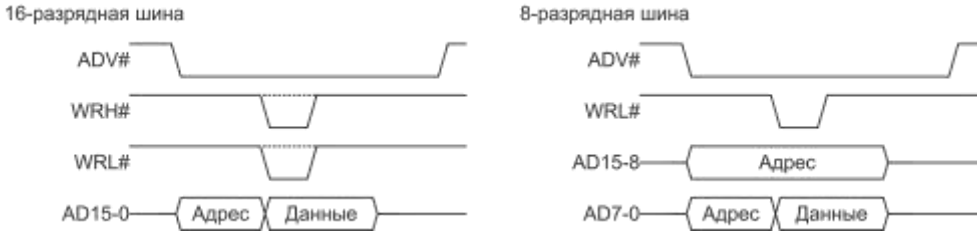


Рисунок 20.52 – Режим строба достоверного адреса и записи

На рисунке 20.13 показана схема взаимодействия микроконтроллера и внешнего ОЗУ с использованием 16-разрядной шины.

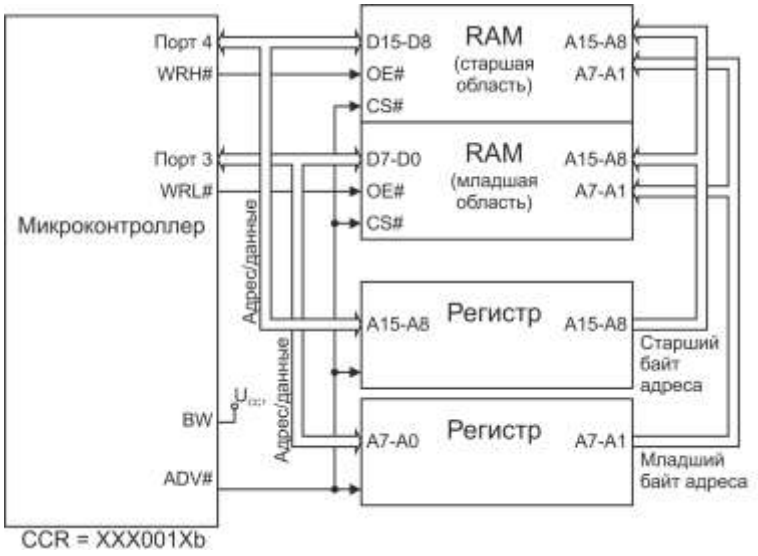


Рисунок 20.63 – Взаимодействие микроконтроллера и внешней памяти

Заключение

В настоящем руководстве пользователя приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения ИС 1874ВЕ7Т, 1874ВЕ71Т, которые представляют собой СБИС одно-кристального 16-разрядного микроконтроллера с тактовой частотой до 24 МГц, регистровым ОЗУ объемом 2 024 байта, 3-канальным ШИМ, двумя портами ввода-вывода UART, интерфейсами SPI, I2C, SpaceWire, магистральным последовательным интерфейсом по ГОСТ Р 52070–2003 и 16-канальным (12-канальным для 1874ВЕ71Т) 12-разрядным АЦП с возможностью работы с дифференциальными сигналами и двумя дополнительными аппаратными битами повышения точности.

Основные области применения микроконтроллера – это различные системы управления, системы автоматизации технологических процессов, системы автоматизированного управления электроприводом, робототехнические комплексы, телекоммуникационная и другая техника.

Все значения электрических параметров ИС приведены в АЕЯР.431280.903ТУ; значения параметров, приведенные в настоящем техническом описании, являются справочными.

Руководстве пользователя может служить практическим руководством по применению микроконтроллеров для разработчиков систем на основе ИС 1874ВЕ7Т, 1874ВЕ71Т и программистов.

Приложение А
(обязательное)

Список регистров микроконтроллера

В таблице А.1 представлен список регистров микроконтроллера, расположенных в алфавитном порядке их мнемонических обозначений. Далее в таблицах А.2 – А.90 представлены форматы регистров и информация о функциональном назначении их бит.

Таблица А.1 – Регистры микроконтроллера

Мнемоническое обозначение	Название	Адрес	Состояние после сброса
1	2	3	4
ADC_CON	Регистр управления АЦП	1FC0h	00h
ADC_EN	Регистр инициализации АЦП	1FC2h	00h
ADC_RESULT	Регистр результата преобразования АЦП	1FC4h	0000h
ADC_SET	Регистр настроек АЦП	1FC1h	00h
ADC_TRSCHAN	Регистр номера канала АЦП	1FC6h	0000h
BAUD_RATE0	Регистр скорости передачи порта UART0	0Eh (0/-)	00h
BAUD_RATE1	Регистр скорости передачи порта UART1	1FA6h	00h
BSICONFIG	Регистр конфигурации	1F30h	000Dh
BSISADDR	Регистр начального адреса	1F34h	0800h
BSISPACE	Регистр паузы между сообщениями	1F36h	0004h
BSICON	Регистр управления	1F32h	0000h
BSISTAT	Регистр состояния	1F38h	0000h
CCR	Регистр конфигурации кристалла	–	1Fh
CLKC	Регистр управления тактированием	1FF0h	FF3Fh
CLKCDEB	Регистр загрузки CLKC при отладке	1FF6h	0000h
CURRPC	Регистр текущего значения счетчика команд	1FE8h	XXXXh
DEBADDR	Регистр ожидаемого значения адреса	1FE4h	0000h
DEBCTRL (старший байт)	Регистр управления модулем отладки	1FE7h	00h
DEBCTRL (младший байт)	Регистр управления модулем отладки	1FE6h	00h
DEBDATA	Регистр ожидаемого значения данных	1FE2h	0000h
DEBDATH	Регистр ожидаемого значения старшего байта данных	1FEFh	00h
DEBDATL	Регистр ожидаемого значения младшего байта данных	1FEEh	00h
DEBPC	Регистр ожидаемого значения счетчика команд	1FE0h	0000h
DEBPCEQ	Регистр ожидаемого значения счетчика команд	1FECh	0000h
DEBPCEQ2	Дополнительный регистр ожидаемого значения счетчика команд	1FACh	0000h
HAPPPC	Регистр значения счетчика команд	1FEAh	XXXXh

Продолжение таблицы А.1

1	2	3	4
HSI_MODE	Регистр режима HSI	03h (0/15)	FFh
HSI_STATUS	Регистр состояния HSI	06h (15/0)	X0X0X0X0b
HSI_TIME	Регистр времени HSI	04h (15/0)	XXXXh
HSO_COMMAND	Регистр управления HSO	06h (0/15)	XXh
HSO_TIME	Регистр времени HSO	04h (0/15)	XXXXh
INT_MASK	Регистр маски прерываний	08h	00h
INT_MASK1	Регистр маски прерываний 1	13h	00h
INT_PEND	Регистр ждущих прерываний	09h	00h
INT_PEND1	Регистр ждущих прерываний 1	12h	00h
IOC0	Регистр 0 управления вводом/выводом	15h (0/15)	000000X0b
IOC1	Регистр 1 управления вводом/выводом	16h (0/15)	21h
IOC2	Регистр 2 управления вводом/выводом	0Bh (0/15)	X0000000b
IOC3	Регистр 3 управления вводом/выводом	0Ch (1)	F0h
IOPORT0	Регистр ввода/вывода порта 0	0Eh (-/0) или 1FF8h	XXh
IOPORT1	Регистр ввода/вывода порта 1	0Fh (0)	FFh
IOPORT2	Регистр ввода/вывода порта 2	10h (0)	C1h
IOPORT3	Регистр ввода/вывода порта 3	1FFEh	FFh
IOPORT4	Регистр ввода/вывода порта 4	1FFFh	FFh
IOPORT5	Регистр ввода/вывода порта 5	1FF9h	FFh
IOS0	Регистр 0 состояния ввода/вывода	15h (15/0)	00h
IOS1	Регистр 1 состояния ввода/вывода	16h (15/0)	00h
IOS2	Регистр 2 состояния ввода/вывода	17h (15/0)	00h
PSW	Слово состояния программы	–	–
PTSBLOCK	Регистр числа передач PTS	–	XXh
PTSCON	Регистр управления PTS	–	00h/02h
PTSCOUNT	Регистр количества циклов PTS	–	XXh
PTSDST	Регистр приемника PTS	–	XXXXh
PTSSRC	Регистр источника PTS	–	XXXXh
PTSSEL	Регистр выбора PTS	04h (1)	0000h
PTSSRV	Регистр обслуживания PTS	06h (1)	0000h
PWM0_CONTROL	Регистр длительности импульса канала 0	17h (0/15)	00h
PWM1_CONTROL	Регистр длительности импульса канала 1	16h (1)	00h
PWM2_CONTROL	Регистр длительности импульса канала 2	17h (1)	00h
SBUF_RX0	Регистр принятых данных UART0	07h (15/0)	00h
SBUF_TX0	Регистр передаваемых данных UART0	07h (0/15)	00h
SBUF_RX1	Регистр принятых данных UART1	1FABh	00h
SBUF_TX1	Регистр передаваемых данных UART1	1FA9h	00h
SMBADDR	Регистр собственного адреса I2C	1FB8h	00h
SMBCST	Регистр управления и состояния I2C	1FB4h	00h
SMBCTRL1	Регистр 1 управления I2C	1FB6h	00h
SMBCTRL2	Регистр 2 управления I2C	1FBAh	00h
SMBCTRL3	Регистр 3 управления I2C	1FBEh	00h
SMBSDA	Сдвиговый регистр данных I2C	1FB0h	XXh
SMBST	Регистр состояния I2C	1FB2h	00h
SMBTOPR	Регистр загрузки предделителя	1FBCh	00h

Окончание таблицы А.1

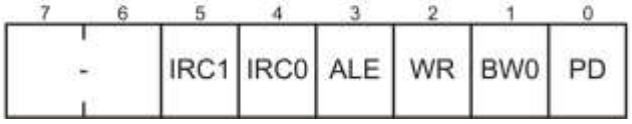
1	2	3	4
SWCAPSTAT	Регистр состояния буферов	1F4Ch	0000h
SWCON	Регистр управления	1F40h	FF00h
SWDATAT	Регистр отправляемых данных	1F44h	0000h
SWDATAR	Регистр принятых данных	1F46h	0000h
SWINTMASK	Регистр маски прерываний	1F4E	0000h
SWSTAT	Регистр состояния	1F42h	3001h
SWTIMET	Регистр отправляемого кода времени	1F48h	0000h
SWTIMER	Регистр принятого кода времени	1F4Ah	0000h
SP	Регистр указателя стека	18h	1D18h
SPCR	Регистр управления SPI	1FA0h	04h
SPDR	Регистр данных SPI	1FA4h	00h
SPSR	Регистр состояния SPI	1FA2h	00h
SP_CON0	Регистр управления UART0	11h (0/15)	0Bh
SP_CON1	Регистр управления UART1	1FAFh	0Bh
SP_STAT0	Регистр состояния UART0	11h (15/0)	0Bh
SP_STAT1	Регистр состояния UART1	1FADh	0Bh
T2CAPTURE	Регистр захвата таймера 2	0Ch	XXXXh
TIMER1	Регистр таймера 1	0Ah (15/0)	0000h
TIMER2	Регистр таймера 2	0Ch (0)	0000h
WATCHDOG	Регистр сторожевого таймера	0Ah (0/15)	XXh
WSR	Регистр выбора окна	14h	X0h
ZERO_REG	Регистр нуля	00h	0000h

Примечание – Для регистров, адреса которых лежат в диапазоне от 00h до 17h, в скобках после адреса указаны номера горизонтальных окон, посредством которых осуществляется запись/чтение. Если цифра одна, значит запись/чтение производится через одно окно. Отсутствие скобок указывает на то, что регистр доступен для записи/чтения по указанному адресу в независимости от того, какое окно активно.

Таблица А.2 – Регистр CCR

CCR
регистр конфигурации кристалла

значение после Сброса: 1Fh



Поле	Бит	Тип	Описание	
IRC1 IRC0	5,4	Однократная аппаратная загрузка при сбросе микросхемы	Биты управления готовностью внешних устройств. Определяют число циклов ожидания, которые могут быть введены, пока на выводе READY удерживается низкий уровень сигнала. Длительность одного цикла ожидания составляет один такт сигнала CLOCKOUT	
			00	1 цикл ожидания
			01	2 цикла ожидания
			10	3 цикла ожидания
			11	Вставка циклов ожидания до тех пор, пока сигнал READY в низком состоянии
ALE, WR	3, 2		Биты выбора режима. Определяют, какие сигналы будут генерироваться на шине управления в циклах записи и чтения внешней памяти (подробнее см. таблицу А.3)	
BW0	1		Бит управления разрядностью шины	
			0	8-разрядная шина; состояние вывода BW не важно
			1	8-разрядная шина, при BW = 0 16-разрядная шина, при BW = 1
PD	0		Бит разрешения включения режима PowerDown	
			0	Режим запрещен
			1	Режим разрешен
–	7,6	–	Зарезервировано	

Таблица А.3 – Варианты конфигурации сигналов шины управления

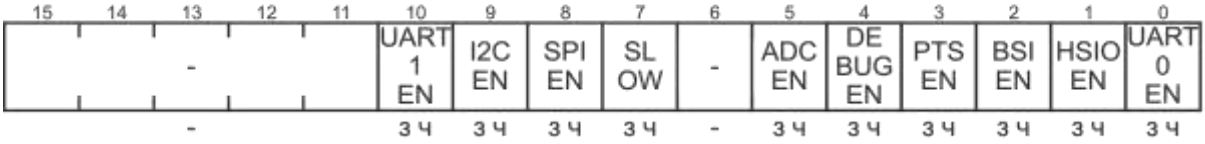
Биты регистра CCR		Режим шины управления	Активный сигнал вывода на шине управления			
ALE	WR					
0	0	Строба записи и достоверного адреса	ADV#	RD#	WRH#	WRL#
0	1	Строба достоверного адреса	ADV#	RD#	BHE#	WR#
1	0	Строба записи	ALE	RD#	WRH#	WRL#
1	1	Стандартный	ALE	RD#	BHE#	WR#

Таблица А.4 – Регистр CLKC

CLKC
регистр управления тактированием

значение после Сброса: FF3Fh

1FF0h



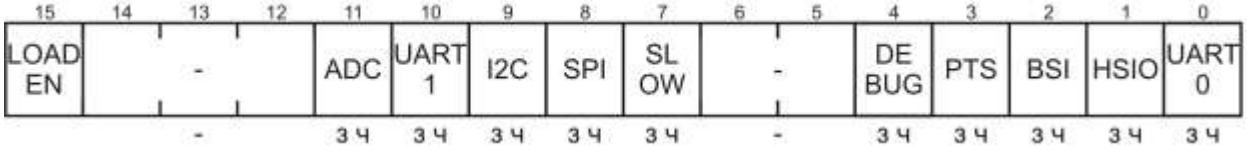
Поле	Бит	Тип	Описание	
UART1EN	10	Запись Чтение	Бит разрешения тактирования UART1	
			0	Запрещено
			1	Разрешено
I2CEN	9	Запись Чтение	Бит разрешения тактирования I2C	
			0	Запрещено
			1	Разрешено
SPIEN	8	Запись Чтение	Бит разрешения тактирования SPI	
			0	Запрещено
			1	Разрешено
SLOW	7	Запись Чтение	Бит включения режима медленного тактирования SLOW	
			0	Выключен (Fclk = Fosc/2)
			1	Включен (Fclk = Fosc/4)
ADCEN	5	Запись Чтение	Бит разрешения тактирования АЦП	
			0	Запрещено
			1	Разрешено
DEBUGEN	4	Запись Чтение	Бит разрешения тактирования модуля отладки	
			0	Запрещено
			1	Разрешено
PTSEN	3	Запись Чтение	Бит разрешения тактирования PTS	
			0	Запрещено
			1	Разрешено
BSIEN	2	Запись Чтение	Бит разрешения тактирования магистрального интерфейса	
			0	Запрещено
			1	Разрешено
HSIOEN	1	Запись Чтение	Бит разрешения тактирования HSIO	
			0	Запрещено
			1	Разрешено
UART0EN	0	Запись Чтение	Бит разрешения тактирования UART0	
			0	Запрещено
			1	Разрешено
–	15-11, 6	–	Зарезервировано	

Таблица А.5 – Регистр CLKCDEB

CLKCDEB
регистр загрузки CLKC при отладке

значение после Сброса: 0000h

1FF6h



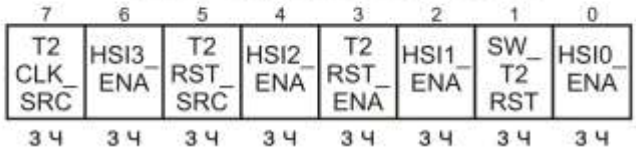
Поле	Бит	Тип	Описание	
LOADEN	15	Запись Чтение	Бит разрешения загрузки регистра CLKC	
			0	Запрещена
			1	Разрешена. Загрузка CLKC по прерыванию DEBUG
Биты с 11 по 7 и с 4 по 0		Запись Чтение	Биты, которые загружаются в соответствующие им биты регистра CLKC по прерыванию DEBUG	
–	14-12, 6, 5	–	Зарезервировано	

Таблица А.6 – Регистр IOC0

IOC0
регистр 0 управления вводом/выводом

15h значение после Сброса: 0000 00X0b

запись - горизонтальное окно 0
чтение - горизонтальное окно 15



Поле	Бит	Тип	Описание	
1	2	3	4	
T2CLK_SRC	7	Запись Чтение	Бит выбора внешнего источника синхронизации для таймера 2 (вывод P2.3)	
			0	T2CLK
			1	HSI.1
HSI3_ENA	6	Запись Чтение	Бит разрешения захвата состояний вывода HSI.3	
			0	Запрещено
			1	Разрешено
T2RST_SRC	5	Запись Чтение	Бит выбора источника внешнего сброса таймера 2	
			0	Передний фронт сигнала на выводе T2RST
			1	Передний фронт сигнала на выводе HSI.0
HSI2_ENA	4	Запись Чтение	Бит разрешения захвата состояний вывода HSI.2	
			0	Запрещено
			1	Разрешено

Окончание таблицы А.6

1	2	3	4	
T2RST_ENA	3	Запись Чтение	Бит разрешения внешнего сброса таймера 2	
			0	Запрещено
			1	Разрешено
HSI1_ENA	2	Запись Чтение	Бит разрешения захвата состояний вывода HSI.1	
			0	Запрещено
			1	Разрешено
SW_T2RST	1	Запись Чтение	Программный сброс таймера 2. При чтении этого бита всегда возвращается единица	
			0	Нет действий
			1	Запись единицы вызывает сброс таймера 2
HSI0_ENA	0	Запись Чтение	Бит разрешения захвата состояний вывода HSI.0	
			0	Запрещено
			1	Разрешено

Таблица А.7 – Регистр IOC1

IOC1
регистр 1 управления вводом/выводом

16h значение после Сброса: 21h

запись - горизонтальное окно 0
чтение - горизонтальное окно 15

7	6	5	4	3	2	1	0
HSI_INT	HSO 2/4_ ENA	TXD_ SEL	HSO 0/1_ ENA	T2 OVF_ INT	T1 OVF_ INT	EXT INT_ SRC	PWM_ SEL
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Поле	Бит	Тип	Описание	
1	2	3	4	
HSI_INT	7	Запись Чтение	Бит выбора источника прерывания HSI	
			0	Выходной буфер загружен
			1	Память FIFO заполнена
HSO2/4_ENA	6	Запись Чтение	Бит выбора режима работы выводов P1.6 и P1.7	
			0	Выводы общего назначения / выводы модуля SPI (объединены по И)
			1	Выводы HSO.2 и HSO.4 модуля HSO
TXD_SEL	5	Запись Чтение	Бит выбора режима работы выводов P2.0 и P2.6	
			0	Выводы общего назначения
			1	Выходы TXD0 и TXD1 последовательных портов
HSO0/1_ENA	4	Запись Чтение	Бит выбора режима работы выводов P1.2 и P1.5	
			0	Выводы общего назначения / выводы модуля SPI (объединены по И)
			1	Выводы HSO.0 и HSO.1 модуля HSO
T2OVF_INT	3	Запись Чтение	Бит разрешения прерывания по переполнению от таймера 2	
			0	Запрещено
			1	Разрешено

Окончание таблицы А.7

1	2	3	4	
T1OVF_INT	2	Запись Чтение	Бит разрешения прерывания по переполнению от таймера 1	
			0	Запрещено
			1	Разрешено
EXTINT_SRC	1	Запись Чтение	Бит выбора источника внешнего прерывания	
			0	Вывод P2.2
			1	Вывод P0.7
PWM_SEL	0	Запись Чтение	Бит выбора режима работы вывода P2.5	
			0	Вывод общего назначения / вывод HSO.5 (объединены по И)
			1	Выход PWM.0 модуля ШИМ

Таблица А.8 – Регистр IOC2

IOC2
регистр 2 управления вводом/выводом

0Bh значение после Сброса: X0000000b

запись - горизонтальное окно 0
чтение - горизонтальное окно 15

7	6	5	4	3	2	1	0
CAM_CLR	LOCK_ENA	T2ALT_INT	-	SLOW_PWM		T2UD_ENA	FAST_T2_ENA
З Ч	З Ч	З Ч	-	З Ч		З Ч	З Ч

Поле	Бит	Тип	Описание	
1	2	3	4	
CAM_CLR	7	Запись Чтение	Бит очистки АЗУ модуля HSO. При чтении этого бита всегда возвращается единица	
			0	Нет действий
			1	Запись единицы вызывает очистку всего содержимого АЗУ
LOCK_ENA	6	Запись Чтение	Бит разрешения фиксации команд в АЗУ модуля HSO	
			0	Фиксация всех команд запрещена
			1	Фиксация всех команд разрешена. Параметры фиксации для каждой команды устанавливаются индивидуально посредством бита CAM_LOCK регистра HSO_COMMAND при загрузке в АЗУ
T2ALT_INT	5	Запись Чтение	Бит выбора границы переполнения таймера 2	
			0	FFFFh/0000h
			1	7FFFh/8000h
SLOW_PWM	3-2	Запись Чтение	Поле делителя выходной частоты каналов модуля ШИМ	
			00	Период выходного сигнала 256 машинных тактов
			01	512 машинных тактов
			10	1024 машинных такта
			11	2048 машинных тактов

Окончание таблицы А.8

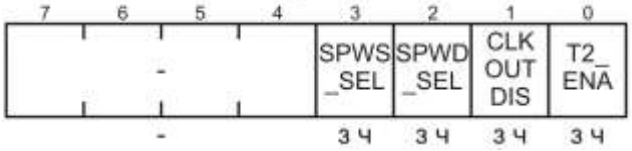
1	2	3	4	
T2UD_ENA	1	Запись Чтение	Бит задания направления счета таймера 2	
			0	Таймер считает только вверх
			1	Если на выводе P2.6 (T2UPDN) ноль, то таймер считает вверх; Если на выводе P2.6 единица, то таймер считает вниз
FAST_T2_ ENA	0	Запись Чтение	Бит выбора режима работы таймера 2	
			0	Режим нормального счета. Счетчик таймера переключается каждые 8 тактов синхросигнала
			1	Режим скоростного счета. Счетчик таймера переключается с каждым тактом синхросигнала
–	4	–	Зарезервировано	

Таблица А.9 – Регистр IOC3

IOC3
регистр 3 управления вводом/выводом

0Ch значение после Сброса: F0h

запись/чтение - горизонтальное окно 1



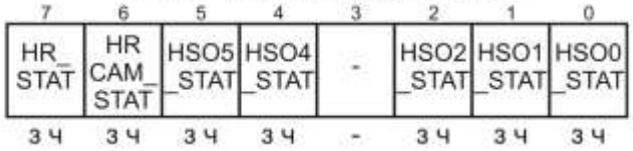
Поле	Бит	Тип	Описание	
SPWS_SEL	3	Запись Чтение	Бит выбора режима работы вывода P1.4	
			0	Вывод общего назначения
			1	Вывод SPW_SOUT модуля SpaceWire
SPWD_SEL	2	Запись Чтение	Бит выбора режима работы вывода P1.3	
			0	Вывод общего назначения
			1	Вывод SPW_DOUT модуля SpaceWire
CLKOUT DIS	1	Запись Чтение	Бит разрешения вывода сигнала CLKOUT. Бит аппаратно установлен и не может быть сброшен	
T2_ENA	0	Запись Чтение	Бит выбора источника синхронизации таймера 2	
			0	Внешний источник (см. бит T2CLK_SRC регистра IOC0)
			1	Внутренний источник (внутренняя тактовая частота микроконтроллера)
–	7-4	–	Зарезервировано	

Таблица А.10 – Регистр IOS0

IOS0
регистр 0 состояния ввода/вывода

15h значение после Сброса: 00h

запись - горизонтальное окно 15
чтение - горизонтальное окно 0



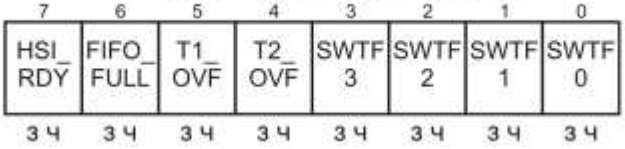
Поле	Бит	Тип	Описание
HR_STAT	7	Запись Чтение	Флаг состояния предварительного буфера модуля HSO. Устанавливается, когда производится запись в предварительный буфер
		0	Буфер пуст
		1	Буфер не пуст
HRCAM_ STAT	6	Запись Чтение	Флаг состояния предварительного буфера и АЗУ модуля HSO
		0	Буфер пуст и по крайней мере одно слово АЗУ пусто
		1	Буфер заполнен и/или заполнено АЗУ
HSO5_STAT	5	Запись Чтение	Бит режима работы вывода P2.5
		0	Вывод общего назначения
		1	Выход HSO.5
HSO4_STAT	4	Запись Чтение	Бит режима работы вывода P1.7
		0	Вывод общего назначения
		1	Выход HSO.4
HSO2_STAT	2	Запись Чтение	Бит режима работы вывода P1.6
		0	Вывод общего назначения
		1	Выход HSO.2
HSO1_STAT	1	Запись Чтение	Бит режима работы вывода P1.5
		0	Вывод общего назначения
		1	Выход HSO.1
HSO0_STAT	0	Запись Чтение	Бит режима работы вывода P1.2
		0	Вывод общего назначения
		1	Выход HSO.0
–	3	–	Зарезервировано

Таблица А.11 – Регистр IOS1

IOS1
регистр 1 состояния ввода/вывода

16h значение после Сброса: 00h

запись - горизонтальное окно 15
чтение - горизонтальное окно 0



Поле	Бит	Тип	Описание	
HSI_RDY	7	Запись Чтение	Флаг загрузки выходного буфера модуля HSI. Устанавливается всегда, когда в выходной буфер загружается слово	
			0	Состояние буфера не изменялось
			1	Буфер загружен
FIFO_FULL	6	Запись Чтение	Флаг заполнения памяти FIFO модуля HSI	
			0	В памяти менее шести записанных слов
			1	В памяти шесть или более записанных слов
T1_OVF	5	Запись Чтение	Флаг переполнения таймера 1	
			0	Нет действий
			1	Таймер переполнился
T2_OVF	4	Запись Чтение	Флаг переполнения таймера 2	
			0	Нет действий
			1	Таймер переполнился
SWTF3	3	Запись Чтение	Флаг программного прерывания по команде SWT3 от модуля HSO	
			0	Нет действий
			1	Сформирован запрос на прерывание
SWTF2	2	Запись Чтение	Флаг программного прерывания по команде SWT2 от модуля HSO	
			0	Нет действий
			1	Сформирован запрос на прерывание
SWTF1	1	Запись Чтение	Флаг программного прерывания по команде SWT1 от модуля HSO	
			0	Нет действий
			1	Сформирован запрос на прерывание
SWTF0	0	Запись Чтение	Флаг программного прерывания по команде SWT0 от модуля HSO	
			0	Нет действий
			1	Сформирован запрос на прерывание

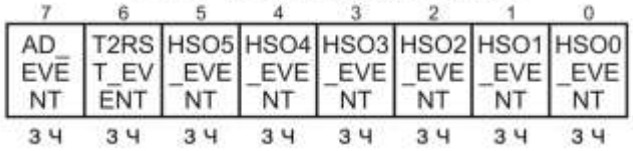
Примечание – Операция чтения регистра IOS1 всегда сбрасывает его биты с пятого по нулевой.

Таблица А.12 – Регистр IOS2

IOS2
регистр 2 состояния ввода/вывода

17h значение после Сброса: 00h

запись - горизонтальное окно 15
чтение - горизонтальное окно 0



Поле	Бит	Тип	Описание	
AD_EVENT	7	Запись Чтение	Флаг запуска АЦП	
			0	Нет действий
			1	Команда модуля HSO запустила АЦ-преобразование
T2RST_EVENT	6	Запись Чтение	Флаг сброса таймера 2	
			0	Нет действий
			1	Команда модуля HSO сбросила таймер 2
HSO5_EVENT	5	Запись Чтение	Флаг переключения вывода HSO.5	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
HSO4_EVENT	4	Запись Чтение	Флаг переключения вывода HSO.4	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
HSO3_EVENT	3	Запись Чтение	Флаг переключения вывода HSO.3	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
HSO2_EVENT	2	Запись Чтение	Флаг переключения вывода HSO.2	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
HSO1_EVENT	1	Запись Чтение	Флаг переключения вывода HSO.1	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
HSO0_EVENT	0	Запись Чтение	Флаг переключения вывода HSO.0	
			0	Нет действий
			1	Команда модуля HSO переключила вывод
Примечание – Операция чтения регистра IOS2 всегда сбрасывает все его биты.				

Таблица А.13 – Регистр INT_MASK1

INT_MASK1
регистр 1 маски прерываний

13h значение после Сброса: 00h

запись/чтение - любое горизонтальное окно

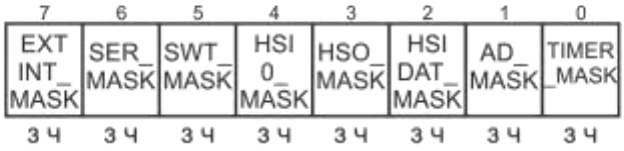
Поле	Бит	Тип	Описание	
NMI_MASK	7	Запись Чтение	Нефункционирующий бит. Немаскируемое прерывание NMI разрешено при любом состоянии этого бита	
FIFO_MASK	6	Запись Чтение	Бит разрешения прерывания по окончании заполнения памяти FIFO модуля HSI	
			0	Запрещено
			1	Разрешено
EXTINT1_MASK	5	Запись Чтение	Бит разрешения прерывания с вывода EXINT1	
			0	Запрещено
			1	Разрешено
T2OVF_MASK	4	Запись Чтение	Бит разрешения прерывания по переполнению от таймера 2	
			0	Запрещено
			1	Разрешено
T2CAP_MASK	3	Запись Чтение	Бит разрешения прерывания при захвате значения таймера 2	
			0	Запрещено
			1	Разрешено
HSI4_MASK	2	Запись Чтение	Бит разрешения прерывания после четвертой записи в память FIFO модуля HSI	
			0	Запрещено
			1	Разрешено
BSI_MASK	1	Запись Чтение	Бит разрешения прерывания от контроллера магистрального последовательного интерфейса	
			0	Запрещено
			1	Разрешено
TI_RI_MASK	0	Запись Чтение	Бит разрешения прерывания по окончании передачи/приема от модулей UART0 и UART1	
			0	Запрещено
			1	Разрешено

Таблица А.14 – Регистр INT_MASK

INT_MASK
регистр маски прерываний

08h значение после Сброса: 00h

запись/чтение - любое горизонтальное окно



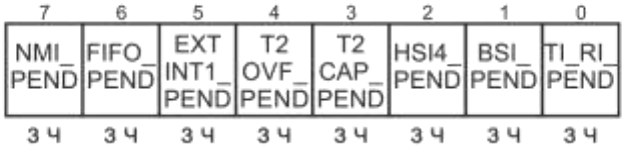
Поле	Бит	Тип	Описание	
EXTINT_ MASK	7	Запись Чтение	Бит разрешения внешнего прерывания с вывода EXTINT	
			0	Запрещено
			1	Разрешено
SER_MASK	6	Запись Чтение	Бит разрешения прерываний от последовательных портов SPI, I2C и SpaceWire	
			0	Запрещено
			1	Разрешено
SWT_MASK	5	Запись Чтение	Бит разрешения программных прерываний по командам модуля HSO	
			0	Запрещено
			1	Разрешено
HSI0_MASK	4	Запись Чтение	Бит разрешения внешнего прерывания с вывода P2.2	
			0	Запрещено
			1	Разрешено (если сброшен бит HSI0_ENA регистра IOC0)
HSO_MASK	3	Запись Чтение	Бит глобального разрешения прерываний от HSO	
			0	Запрещено
			1	Разрешено
HSIDAT_ MASK	2	Запись Чтение	Бит разрешения прерывания по окончании формирования данных от HSI	
			0	Запрещено
			1	Разрешено
AD_MASK	1	Запись Чтение	Бит разрешения прерывания по окончании преобразования от АЦП	
			0	Запрещено
			1	Разрешено
TIMER_ MASK	0	Запись Чтение	Бит разрешения прерывания по переполнению от таймеров 1 и 2	
			0	Запрещено
			1	Разрешено

Таблица А.15 – Регистр INT_PEND1

INT_PEND1
регистр 1 ждущих прерываний

12h значение после Сброса: 00h

запись/чтение - любое горизонтальное окно



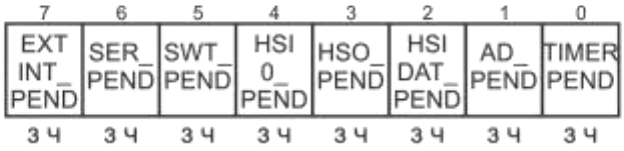
Поле	Бит	Тип	Описание	
NMI_PEND	7	Запись Чтение	Индикатор запроса на прерывание NMI	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
FIFO_PEND	6	Запись Чтение	Индикатор запроса на прерывание по окончании заполнения памяти FIFO модуля HSI	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
EXTINT1_ PEND	5	Запись Чтение	Индикатор запроса на прерывание с вывода EXINT1	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
T2OVF_PEND	4	Запись Чтение	Индикатор запроса на прерывание по переполнению таймера 2	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
T2CAP_PEND	3	Запись Чтение	Индикатор запроса на прерывание при захвате значения таймера 2	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
HSI4_PEND	2	Запись Чтение	Индикатор запроса на прерывание после четвертой записи в память FIFO модуля HSI	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
BSI_PEND	1	Запись Чтение	Индикатор запроса на прерывание от модуля BSI	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
TI_RI_PEND	0	Запись Чтение	Индикатор запроса на прерывание по окончании передачи/приема от модулей UART0 и UART1	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания

Таблица А.16 – Регистр INT_PEND

INT_PEND
регистр ждущих прерываний

09h значение после Сброса: 00h

запись/чтение - любое горизонтальное окно



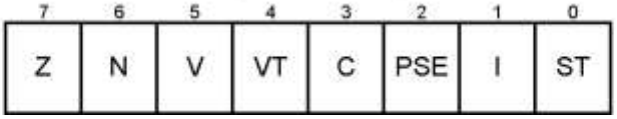
Поле	Бит	Тип	Описание	
EXTINT_ PEND	7	Запись Чтение	Индикатор запроса на прерывание с вывода EXINT	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
SER_PEND	6	Запись Чтение	Индикатор запроса на прерывание от последовательных портов SPI, I2C и SpaceWire	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
SWT_PEND	5	Запись Чтение	Индикатор запроса программного прерывания по команде модуля HSO	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
HSI0_PEND	4	Запись Чтение	Индикатор запроса на прерывание с вывода P2.2	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
HSO_PEND	3	Запись Чтение	Индикатор запроса на прерывание от HSO	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
HSIDAT_ PEND	2	Запись Чтение	Индикатор запроса на прерывание по окончании формирования данных от HSI	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
AD_PEND	1	Запись Чтение	Индикатор запроса на прерывание по окончании преобразования от АЦП	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания
TIMER PEND	0	Запись Чтение	Индикатор запроса на прерывание по переполнению от таймеров 1 и 2	
			0	Нет запроса на прерывание
			1	Запрос на прерывание ожидает обслуживания

Примечание – Каждый индикатор сбрасывается аппаратно сразу после того, как происходит переход по соответствующему вектору прерывания.

Таблица А.17 – Регистр PSW

PSW
слово состояния программы

Нет прямого доступа значение после Сброса: --

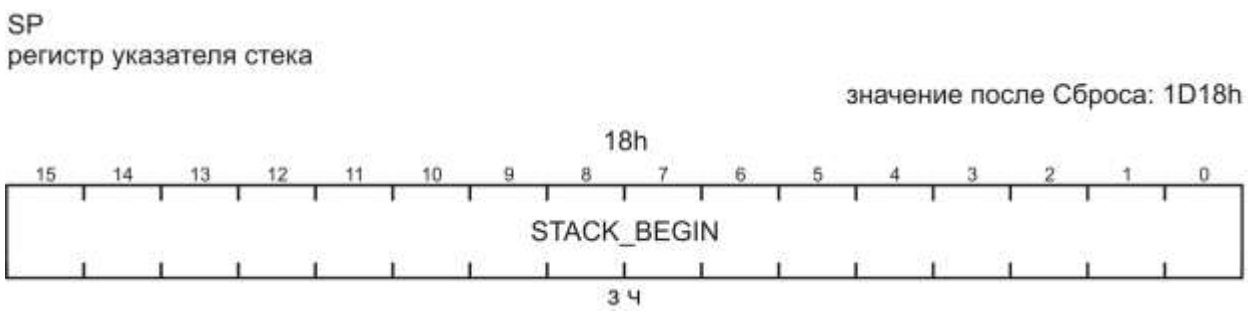


Поле	Бит	Описание
1	2	3
Z	7	Флаг нуля. Устанавливается, если результат операции равен нулю. При операциях сложения с переносом и вычитания с заемом этот флаг никогда не устанавливается, но он сбрасывается, если результат не нулевой. Таким образом, флаг нулевого результата показывает наличие нулевого 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
V	5	Флаг переполнения. Устанавливается, если после выполнения операции получен результат, выходящий за границы диапазона значений типа данных приёмника. При сдвиговых операциях (SHL, SHLB, SHLL) этот флаг устанавливается, если старший значащий бит операнда изменился в процессе сдвига. При операциях деления этот флаг устанавливается, если при выполнении команды: – DIVUB результат > 255 (FFh); – DIVU результат > 65 535 (FFFFh); – DIVB результат < –127 (81h) или > 127 (7Fh); – DIV результат < –32 767 (8001h) или > 32 767 (7FFFh)
VT	4	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
C	3	Флаг переноса. Устанавливается для индикации арифметического переноса из старшего разряда ALU или состояния, 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; если ≥ 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: а) 0 и бит ST = 0, то C = 0; б) > 0 и < 1/2 LSB и ST = 1, то C = 0; в) = 1/2 LSB и ST = 0, то C = 1; г) > 1/2 LSB и < 1 LSB и ST = 1, то C = 1

Окончание таблицы А.17

1	2	3	
PSE	2	Бит разрешения PTS. Устанавливается командой EPTS. Сбрасывается командой DPTS. Бит PSE глобально разрешает или запрещает работу сервера периферийных транзакций (PTS)	
		0	Запрещено
		1	Разрешено
I	1	Бит глобального разрешения прерываний. Устанавливается командой EI. Сбрасывается командой DI. Бит I разрешает или запрещает обслуживание всех прерываний, кроме прерываний NMI, TRAP и неподдерживаемого кода	
		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 C, а затем сброшен (т.е. через перенос прошла единица). Флаг ST может использоваться вместе с флагом переноса C для более точного округления.	

Таблица А.18 – Регистр SP



Поле	Бит	Тип	Описание
STACK_ BEGIN	15 – 0	Запись Чтение	Поле адреса вершины стека. Записанное значение должно быть четным и на два больше, чем желаемый адрес вершины стека

Таблица А.19 – Регистр WATCHDOG

WATCHDOG
регистр сторожевого таймера

0Ah значение после Сброса: XXh

сброс/разрешение - горизонтальное окно 0
чтение - горизонтальное окно 15

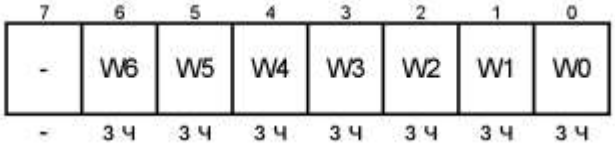
Поле	Бит	Тип	Описание
WATCHDOG	7–0	Запись Чтение	Старший байт счетчика сторожевого таймера

Таблица А.20 – Формат и функциональные назначения полей регистра WSR

WSR
регистр выбора окна

14H значение после Сброса: X0H

(все горизонтальные окна - чтение, запись)



Поле	Бит	Тип	Описание	
W6	6	Запись Чтение	Установка этого бита выбирает 32-байтные вертикальные окна. В других случаях этот бит должен быть равен нулю	
W5	5	Запись Чтение	Установка этого бита выбирает 64-байтные вертикальные окна. В других случаях этот бит должен быть равен нулю	
W4	4	Запись Чтение	Установка этого бита выбирает 128-байтные вертикальные окна. В других случаях этот бит должен быть равен нулю	
W3-W0	3-0	Запись Чтение	Поле указания номера окна. В эти биты записывается номер желаемого горизонтального или вертикального окна. Размер окна выбирается установкой WSR.4, WSR.5 или WSR.6. Действующие горизонтальные окна – 0, 1 и 15. Действующие вертикальные окна – 0 – 15. Для выбора вертикальных окон 16 – 31 необходимо установить WSR.4 и WSR.6	
			0000	Горизонтальное окно 0
			0001	Горизонтальное окно 1
			1111	Горизонтальное окно 15
			Остальные значения зарезервированы	
–	7	–	Зарезервировано	

Таблица А.21– Допустимые состояния регистра WSR

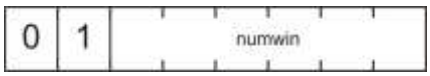
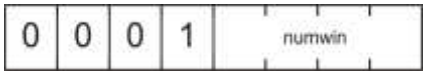
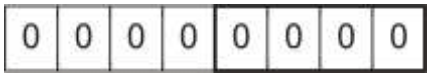
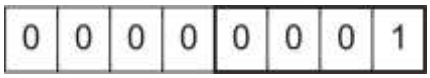
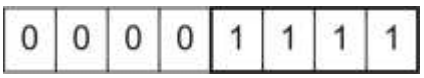
Допустимое состояние регистра WSR	Выбранный режим работы
	32-байтные вертикальные окна. В битовом поле numwin задается порядковый номер окна от 00h до 3Fh (см. рисунок 5.4)
	64-байтные вертикальные окна. В битовом поле numwin задается порядковый номер окна 00h до 1Fh (см. рисунок 5.4)
	128-байтные вертикальные окна. В битовом поле numwin задается порядковый номер окна 00h до 0Fh (см. рисунок 5.4)
	Горизонтальное окно 0
	Горизонтальное окно 1
	Горизонтальное окно 15

Таблица А.22 – Регистр ZERO_REG

Поле	Бит	Тип	Описание
ZERO	15-0	Запись Чтение	Нулевое слово (все биты всегда нули). При использовании регистра ZERO_REG в качестве источника при выполнении команды CMPL, двойное слово приемника сравнивается с 32-битным нулем

Регистры блока PTS

Таблица А.23 – Регистр PTSSEL

PTSSEL
регистр выбора PTS

04h значение после Сброса: 0000h

запись/чтение - горизонтальное окно 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	FIFO_SEL	EXT INT1_SEL	T2 OVF_SEL	T2 CAP_SEL	HSI4_SEL	BSI_SEL	TI_RI_SEL	EXT INT_SEL	SER_SEL	SWT_SEL	HSI0_SEL	HSO_SEL	HSI DAT_SEL	AD_SEL	T1 OVF_SEL
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

Бит	Тип	Описание	
14, 13, …, 1, 0	Запись Чтение	Бит задания варианта обслуживания прерывания	
		0	Прерывание обслуживается контроллером прерываний
		1	Прерывание обслуживается блоком PTS
15	–	Зарезервировано	

Таблица А.24 – Регистр PTSSRV

PTSSRV
регистр обслуживания PTS

06h значение после Сброса: 0000h

запись/чтение - горизонтальное окно 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	FIFO_SRV	EXT INT1_SRV	T2 OVF_SRV	T2 CAP_SRV	HSI4_SRV	BSI_SRV	TI_RI_SRV	EXT INT_SRV	SER_SRV	SWT_SRV	HSI0_SRV	HSO_SRV	HSI DAT_SRV	AD_SRV	T1 OVF_SRV
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

Бит	Тип	Описание	
14, 13, …, 1, 0	Запись Чтение	Индикатор запроса на прерывание END-OF-PTS от блока PTS	
		0	Нет запроса на прерывание
		1	Запрос на прерывание ожидает обслуживания
15	–	Зарезервировано	

Таблица А.25 – Регистр PTSCOUNT

PTSCOUNT
регистр количества циклов PTS

значение после Сброса: XXh

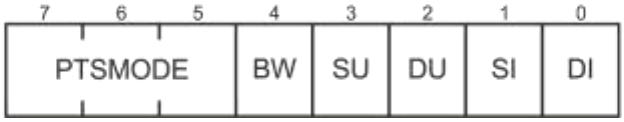
7	6	5	4	3	2	1	0
PTS_COUNT							

Поле	Бит	Тип	Описание
PTS_COUNT	7-0	Запись	Поле задания количества циклов PTS (количество передач), которые будут выполняться блоком PTS без участия центрального процессора

Таблица А.26 – Регистр PTSCON (режим одиночной/блочной передачи)

PTSCON
регистр управления PTS в режиме одиночной/блочной передачи

значение после Сброса: 00h

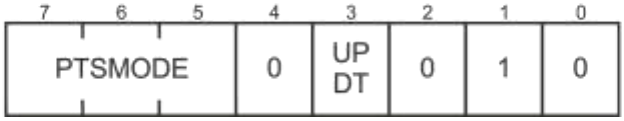


Поле	Бит	Тип	Описание	
PTS MODE	7-5	Запись	Выбор режима	
			000	Блочная передача
			100	Одиночная передача
BW	4	Запись	Бит выбора формата передачи	
			0	Передача слова
			1	Передача байта
SU	3	Запись	Бит изменения значения регистра адреса источника PTSSRC	
			0	В регистре сохраняется значение, которое было в начале цикла PTS
			1	В регистре сохраняется значение, которое возникло на момент окончания цикла PTS
DU	2	Запись	Бит изменения значения регистра адреса приемника PTSDST	
			0	В регистре сохраняется значение, которое было в начале цикла PTS
			1	В регистре сохраняется значение, которое возникло на момент окончания цикла PTS
SI	1	Запись	Бит автоинкремента регистра адреса источника PTSSRC	
			0	Нет действий
			1	Значение регистра инкрементируется в конце каждого цикла PTS: - на единицу при передаче байта; - на два при передаче слова
DI	0	Запись	Бит автоинкремента регистра адреса приемника PTSDST	
			0	Нет действий
			1	Значение регистра инкрементируется в конце каждого цикла PTS: - на единицу при передаче байта; - на два при передаче слова

Таблица А.27 – Регистр PTSCON (режим HSI/HSO)

PTSCON
регистр управления PTS в режиме HSI/HSO

значение после Сброса: 02h



Поле	Бит	Тип	Описание	
PTS MODE	7-5	Запись	Выбор режима обмена	
			001	HSI (соответствующий регистр PTSDST)
			011	HSO (соответствующий регистр PTSSRC)
UPDT	3	Запись	Бит изменения значения в регистре адреса источника/приемника: - PTSSRC (в режиме HSO); - PTSDST (в режиме HSI)	
			0	В регистре сохраняется значение, которое было в начале цикла обмена
			1	В регистр сохраняется значение, которое возникло на момент окончания цикла обмена
–	4, 2-0	–	Зарезервировано При чтении возвращаются указанные значения	

Таблица А.28 – Регистр PTSSRC

PTSSRC
регистр адреса источника PTS

значение после Сброса: XXXXh

Поле	Бит	Тип	Описание
PTSSRC_H	15-8	Запись	Старший байт адреса источника обмена
PTSSRC_L	7-0	Запись	Младший байт адреса источника обмена

Таблица А.29 – Регистр PTSDST

PTSDST
регистр адреса приемника PTS

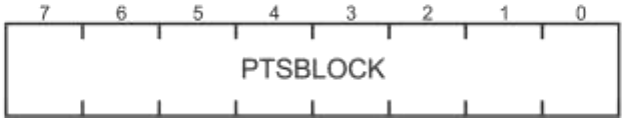
значение после Сброса: XXXXh

Поле	Бит	Тип	Описание
PTSDST_H	15-8	Запись	Старший байт адреса приемника PTS
PTSDST_L	7-0	Запись	Младший байт адреса приемника PTS

Таблица А.30 – Регистр PTSBLOCK

PTSBLOCK
регистр числа передач PTS

значение после Сброса: XXh

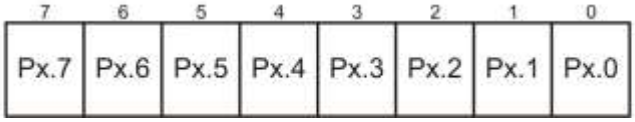


Поле	Бит	Тип	Описание
PTS BLOCK	7-0	Запись	Поле задания количества передаваемых байт/слов за один цикл PTS. Максимально возможное число, которое может быть записано в PTSBLOCK, зависит от режима работы, задаваемого полем PTSMODE регистра PTSCON. Так для: - режима блочной передачи это число 32 (20h); - режима HSI – 7 (07h); - режима HSO – 8 (08h). Во всех режимах запись значения 00h в поле PTSBLOCK автоматически задает максимальное число, соответствующее режиму

Регистры портов

Таблица А.31 – Формат регистра порта

IOPORTx
регистр ввода/вывода порта x



Поле	Бит	Тип	Описание
Px.7	7	См. таблицу А.32	Бит управления состоянием седьмого вывода порта x
Px.6	6		Бит управления состоянием шестого вывода порта x
Px.5	5		Бит управления состоянием пятого вывода порта x
Px.4	4		Бит управления состоянием четвертого вывода порта x
Px.3	3		Бит управления состоянием третьего вывода порта x
Px.2	2		Бит управления состоянием второго вывода порта x
Px.1	1		Бит управления состоянием первого вывода порта x
Px.0	0		Бит управления состоянием нулевого вывода порта x
Примечание – Символ x заменяет номер порта (см. таблицу А.32)			

Таблица А.32 – Регистры IOPORT0, IOPORT1, IOPORT2, IOPORT3, IOPORT4 и IOPORT5

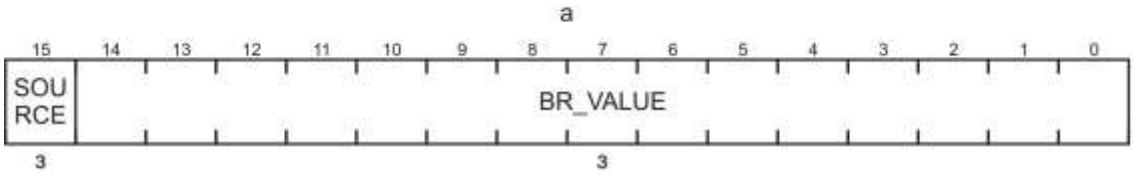
Порт (x)	Регистр	Адрес	Значение после сброса
0	IOPORT0	0Eh (только чтение – горизонтальное окно 0) или 1FF8h (только чтение).	XXh
1	IOPORT1	0Fh (чтение/запись – горизонтальное окно 0)	FFh
2	IOPORT2	10h (чтение/запись – горизонтальное окно 0)	C3h
3	IOPORT3	1FFEh	FFh
4	IOPORT4	1FFFh	FFh
5	IOPORT5	1FF9h	FFh

Регистры портов UART0 и UART1

Таблица А.33 – Регистры BAUD_RATE0 и BAUD_RATE1

BAUD_RATEx
регистр скорости передачи порта UARTx

значение после Сброса: 0000h



x	0	1
Регистр	BAUD_RATE0	BOUD_RATE1
a	0Eh (запись - горизонтальное окно 0)	1FA6h

Поле	Бит	Тип	Описание	
SOU RCE	15	Запись	Бит выбора источника синхросигнала	
			0	Сигнал с вывода T2CLK
			1	Сигнал с вывода XTAL1
BR_ VALUE	14-0	Запись	Скорость передачи. Для расчета значения, которое следует записать в это поле, следует воспользоваться нижеприведенными формулами. Для синхронного режима (Режим 0): $\mathrm{BR\_VALUE_{DEC}}\left(\frac{\mathrm{fuart}}{\mathrm{Bod}}\right)-1,$ (A.1) где $\mathrm{BR\_VALUE_{DEC}}$ – скорость передачи в десятичном формате, fuart – частота синхросигнала, поступающего на блок, Гц, Bod – желаемая скорость передачи, бод. Для асинхронных режимов (Режимы 1, 2, 3): $\mathrm{BR\_VALUE_{DEC}}\left(\frac{\mathrm{fuart}}{\mathrm{Bod}\times 8}\right)-1.$ (A.2)	

Примечание – Следует помнить, что при использовании синхросигнала с вывода T2CLK значение его частоты подставляется в формулы в неизменном виде, а при использовании синхросигнала с вывода XTAL1 значение частоты предварительно уменьшается вдвое. В таблице А.34 приведены значения BR_VALUE, рассчитанные для некоторых стандартных скоростей.

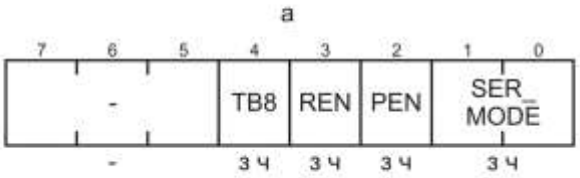
Таблица А.34 – Значения BR_VALUE в зависимости от источника синхросигнала

Скорость обмена	Источник синхросигнала и режим работы			
	XTAL1 (16МГц)		T2CLK (8 МГц)	
	Режим 0	Режимы 1, 2, 3	Режим 0	Режимы 1, 2, 3
9600 бод	8340h	8067h	0340h	0067h
4800 бод	8682h	80CFh	0682h	00CFh
2400 бод	8D04h	81A0h	0D04h	01A0h
1200 бод	9A0Ah	8340h	1A0Ah	0340h
600 бод	B415h	8682h	3415h	0682h

Таблица А.35 – Регистры SP_CON0 и SP_CON1

SP_CONx
регистр управления UARTx

значение после Сброса: 0Bh



x	0	1
Регистр	SP_CON0	SP_CON1
a	11h (запись - горизонтальное окно 0, чтение - горизонтальное окно 15)	1FAFh

Поле	Бит	Тип	Описание	
TB8	4	Запись Чтение	Бит четности. 9-й бит данных, передаваемый в режимах 2 и 3	
REN	3	Запись Чтение	Бит разрешения приема	
			0	Прием данных на входе RxD запрещен
			1	Прием данных на входе RxD разрешен
PEN	2	Запись Чтение	Бит разрешения контроля по паритету	
			0	Запрещен
			1	Разрешен
SER_ MODE	1, 0	Запись Чтение	Поле выбор режима работы	
			00	Режим 0
			01	Режим 1
			10	Режим 2
			11	Режим 3
–	7-5	–	Зарезервировано	

Таблица А.36 – Регистры SP_STAT0 и SP_STAT1

SP_STATx
регистр состояния UARTx

значение после Сброса: 0Bh

a

7	6	5	4	3	2	1	0
RPE/ RB8	RI	TI	FE	TXE	OE	-	
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	-	

x	0	1
Регистр	SP_STAT0	SP_STAT1
a	11h (запись - горизонтальное окно 15, чтение - горизонтальное окно 0)	1FADh

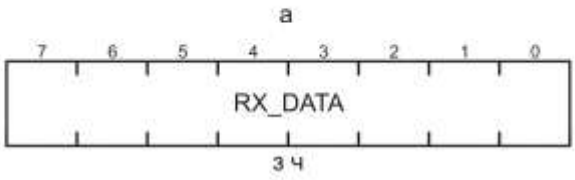
Поле	Бит	Тип	Описание
RPE/ RB8	7	Запись Чтение	Флаг ошибки паритета/принятый девятый бит. Если контроль по паритету разрешен (установлен бит PEN в регистре SP_CON), то флаг RPE установится, если будет обнаружена ошибка паритета в принятом байте. В случае отсутствия проверки по паритету, в бит RB8 будет записано значение девятого бита принятого байта (режимы 2 и 3)
RI	6	Запись Чтение	Флаг окончания приема. В режиме 0 – устанавливается аппаратно после приема 8-ого бита. В режимах 1, 2, 3 – устанавливается аппаратно в течение приема стоп-бита
TI	5	Запись Чтение	Флаг окончания передачи. В режиме 0 – устанавливается после передачи восьмого бита. В режимах 1, 2, 3 – устанавливается в течение передачи стоп-бита
FE	4	Запись Чтение	Флаг ошибки фрейма. Устанавливается, если при приеме байта не был получен стоп-бит
TXE	3	Запись Чтение	Флаг пустого буфера. Устанавливается, если буфер SBUF_TX и передающий сдвиговый регистр пусты. Флаг сбрасывается сразу же после записи в регистр SBUF_TX
OE	2	Запись Чтение	Флаг потери данных. Устанавливается, если регистр SBUF_RX не был прочитан до того, как в него был записан очередной принятый байт
–	1, 0	–	Зарезервировано

Примечание – Следует помнить, что чтение регистра SP_STAT очищает все биты, кроме TXE. В связи с этим рекомендуется копировать содержимое регистра в любой другой регистр и только потом осуществлять проверку состояния битов.

Таблица А.37 – Регистры SBUF_RX0 и SBUF_RX1

SBUF_RXx
регистр принятых данных UARTx

значение после Сброса: 00h



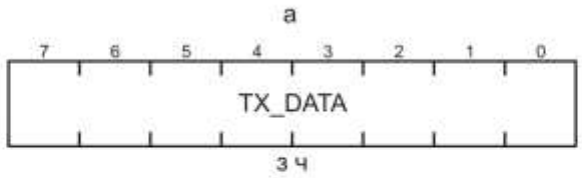
x	0	1
Регистр	SBUF_RX0	SBUF_RX1
a	07h (запись - горизонтальное окно 15, чтение - горизонтальное окно 0)	1FABh

Поле	Бит	Тип	Описание
RX_DATA	7-0	Запись Чтение	Поле принятых данных

Таблица А.38 – Регистры SBUF_TX0 и SBUF_TX1

SBUF_TXx
регистр передаваемых данных UARTx

значение после Сброса: 00h



x	0	1
Регистр	SBUF_TX0	SBUF_TX1
a	07h (запись - горизонтальное окно 0, чтение - горизонтальное окно 15)	1FA9h

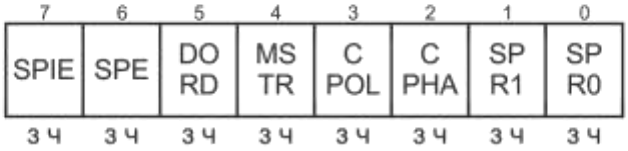
Поле	Бит	Тип	Описание
TX_DATA	7-0	Запись Чтение	Поле данных для передачи

Регистры контроллера интерфейса SPI

Таблица А.39 – Регистр SPCR

SPCR
регистр управления SPI

1FA0h значение после Сброса: 04h

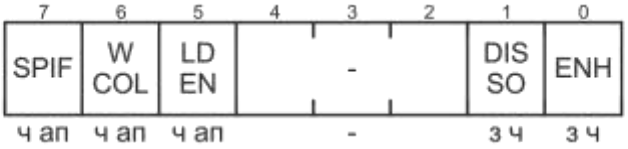


Поле	Бит	Тип	Описание			
1	2	3	4			
SPIE	7	Запись Чтение	Бит разрешения запроса прерывания. Разрешает установку флага SPIF в регистре SPSR. Вместе с битом ES в регистре IE разрешает формирование запроса на прерывание от модуля SPI			
			0	Запрещено		
			1	Разрешено		
SPE	6	Запись Чтение	Бит разрешения работы модуля SPI			
			0	Выключен. Все выводы в высокоимпедансном состоянии		
			1	Включен		
DORD	5	Запись Чтение	Бит управления порядком передачи и приема данных			
			0	Старшим битом вперед		
			1	Младшим битом вперед		
MSTR	4	Запись Чтение	Бит конфигурации			
			0	Режим ведомого		
			1	Режим мастера		
CPOL	3	Запись Чтение	Бит полярности сигнала тактирования			
			0	В отсутствии передачи на выводе SCK удерживается сигнал низкого уровня		
			1	В отсутствии передачи на выводе SCK удерживается сигнал высокого уровня		
CPHA	2	Запись Чтение	Бит фазы тактового сигнала. Бит CPHA вместе с битом CPOL управляет соотношением фаз тактового сигнала и данных на линиях MOSI и MISO			
SPR1 SPR0	1-0	Запись Чтение	Биты управления скоростью передачи данных			
			SPR1	SPR0	Частота синхросигнала SCK	
					При SLOW = 0b	При SLOW = 1b
			0	0	fspi/4	fspi/2
			0	1	fspi/16	fspi/8
			1	0	fspi/64	fspi/32
			1	1	fspi/128	fspi/64

Таблица А.40 – Регистр SPSR

SPSR
регистр состояния SPI

1FA2h значение после Сброса: 00h



Поле	Бит	Тип	Описание
1	2	3	4
SPIF	7	Чтение Аппаратное влияние	Флаг прерывания модуля SPI. Устанавливается (если установлен бит SPIE) по окончании приема/передачи каждого байта данных и является индикатором того, что данные получены, загружены в регистр SPDR и могут быть прочитаны. Одновременно с битом SPIF, если это разрешено битом SPIE регистра SPCR, формируется запрос на прерывание. В режиме буферизации данных (ENH = 1), если передается посылка из нескольких байт данных, флаг SPIF выставляется в конце передачи каждого байта, но запрос на прерывание формируется только один раз – по окончании передачи всей посылки. Для программного сброса флага следует сначала прочитать регистр SPSR. Только после этого чтение или запись регистра SPDR сбросит флаг SPIF. Аппаратно бит SPIF сбрасывается только при включении модуля SPI. При выключении модуля SPI значение бита SPIF не изменяется
WCOL	6	Чтение Аппаратное влияние	Флаг конфликта записи. Поведение флага WCOL зависит от режима работы модуля SPI, который устанавливается битом ENH. 1 В нормальном режиме (ENH = 0) флаг WCOL устанавливается при попытке записи данных в регистр SPDR во время текущей передачи. При этом передача не прерывается, а запись данных блокируется. В этом режиме флаг WCOL сбрасывается аппаратно, одновременно со сбросом флага SPIF. 2 В режим буферизации передаваемых данных (ENH = 1) флаг WCOL устанавливается при попытке записи данных в регистр SPDR во время текущей передачи. При этом передача не прерывается, а данные записываются в буфер данных. По окончании передачи логика проверяет состояние флага WCOL, и если он установлен, загружает байт из буфера данных в сдвиговый передающий регистр и продолжает передачу. Одновременно с этим флаг WCOL аппаратно сбрасывается

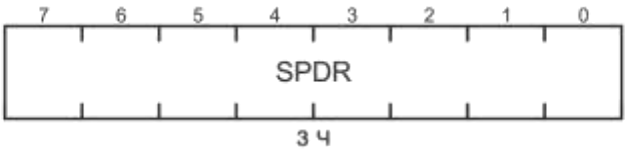
Окончание таблицы А.40

1	2	3	4	
LDEN	5	Чтение Аппаратное влияние	Разрешение загрузки в буфер данных. 1 В нормальном режиме (ENH = 0) бит LDEN не изменяет своего значения и остается равным нулю. 2 В режиме буферизации данных (ENH = 1) бит LDEN установлен в течение передачи первых четырех бит каждого передаваемого байта данных и сброшен во время передачи следующих четырех бит. Фактически бит LDEN определяет промежуток времени в течение передачи, когда желательно записать байт, который будет передан следующим по окончании текущей передачи	
DISSO	1	Запись Чтение	Независимое запрещение передачи данных. 1 В режиме мастера значение бита DISSO игнорируется. 2 В режиме ведомого бит DISSO контролирует выход данных. Пока бит DISSO остается равным нулю, ведомый передает и принимает данные. Установка бита DISSO блокирует передачу данных и переводит вывод MISO в состояние слабой логической единицы, при этом прием тактового сигнала и прием данных не блокируются	
ENH	0	Запись Чтение	Включение режима буферизации передаваемых данных	
			0	Модуль SPI работает в нормальном режиме. Передачи осуществляются по одному байту. Каждый следующий байт данных может быть записан в регистр SPDR и передан только по окончании текущей передачи
			1	Модуль SPI работает в режиме буферизации передаваемых данных. Байт, записываемый в регистр SPDR во время текущей передачи, попадает в буфер данных и загружается аппаратно в передающий регистр сразу же по окончании текущей передачи. В случае, если модуль SPI функционирует как мастер, то формируемый сигнал тактирования SCK в течение времени передачи всех байт не прерывается и его период остается стабильным. Таким образом организуется непрерывная передача неограниченного числа байт
–	4-2	–	Зарезервировано	

Таблица А.41 – Регистр SPDR

SPDR
регистр данных SPI

1FA4h значение после Сброса: 00h



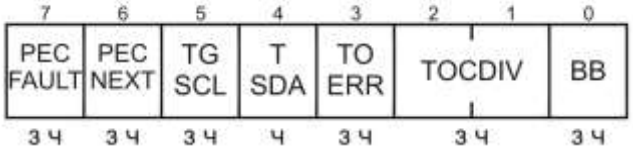
Поле	Бит	Тип	Описание
DATA	7-0	Запись Чтение	Поле данных

Регистры контроллера интерефейса I2C

Таблица А.42 – Регистр SMBCST

SMBCST
регистр управления и состояния I2C

1FB4h — значение после Сброса: 00h



Поле	Бит	Тип	Описание
1	2	3	4
PECFAULT	7	Запись Чтение	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной, значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Запись Чтение	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, что следующий передаваемый байт будет байтом CRC (байт контрольной суммы). Реакция на установку бита PECNEXT зависит от режима работы. В режиме мастера передатчика установка бита PECNEXT вызовет загрузку результата вычисления CRC в регистр SMBSDA. После сброса флага INT начнется передача байта CRC. В режиме приемника установка этого бита будет указывать логике управления на то, что следующий байт, который будет принят, будет байтом CRC. В режиме ведомого приемника модуль I2C автоматически будет квитировать или не квитировать прием байта CRC, в зависимости от того, будет ли выявлена ошибка пакета данных или нет. В режиме мастера приемника по окончании приема байта CRC, будет отправлено значение бита ACK регистра SMBCTRL1
TGSCL	5	Запись Чтение	Бит переключения SCL. Бит позволяет переключать вывод SCL во время восстановления после ошибки. Когда на выводе SDA низкий уровень сигнала, запись «1» в бит TGSCL переключит вывод SCL на один такт. Когда на SDA высокий уровень сигнала, запись «1» в бит TGSCL игнорируется. Бит очищается аппаратно по окончании такта.
TSDA	4	Чтение	Бит тестирования SDA. Содержит текущее значение SDA. Этот бит можно использовать для отслеживания окончания процесса восстановления после ошибки, в течение которого ведомый постоянно поддерживает низкий уровень сигнала на выводе SDA

Окончание таблицы А.42

1	2	3	4	
TOERR	3	Запись Чтение	Флаг ошибки простоя на шине. Если TOERR = 1b, это указывает на то, что на линии SCL был обнаружен простой. Флаг TOERR выставляется по обнулению основного счетчика времени простоя и может быть сброшен записью «1» в бит CLRST регистра SMBCTRL1	
TOCDIV	2-1	Запись Чтение	Поле коэффициента делителя. Устанавливает коэффициент деления системного тактового сигнала, подаваемого на предделитель времени простоя линии SCL	
			00	Тактовый сигнал отсутствует
			01	Деление на 4
			10	Деление на 8
			11	Деление на 16
BB	0	Запись Чтение	Флаг занятости шины. Если BB = 1b, это указывает на то, что шина занята. Устанавливается, как только шина переходит в активное состояние (одновременное появление низкого уровня сигнала на выводах SDA и SCL или хотя бы на одном из них), или в стартовое состояние. Сбрасывается при выключении интерфейса I2C, либо при обнаружении состояния останова.	

Таблица А.43 – Регистр SMBCTRL1

SMBCTRL1
регистр 1 управления I2C

1FB6h значение после Сброса: 00h

7	6	5	4	3	2	1	0
CLR ST	SMB ARE	GCM EN	ACK	-	INT EN	STOP	STA RT
З	З Ч	З Ч	З Ч	-	З Ч	З Ч	З Ч

Поле	Бит	Тип	Описание	
1	2	3	4	
CLRST	7	Запись	Бит сброса флага прерывания INT. Запись «0» в бит CLR игнорируется. Запись «1» в бит CLR сбросит флаг INT в регистре SMBST. Чтение этого бита всегда возвращает «0»	
SMBARE	6	Запись Чтение	Бит управления реакцией на получение адреса отклика	
			0	Полученный адрес не проверяется на совпадение с адресом отклика
			1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом отклика (0001_100b)
			Бит очищается при выходе ведомого из режима IDLE	

Окончание таблицы А.43

1	2	3	4	
GCMEN	5	Запись Чтение	Бит управления реакцией на получение адреса общего вызова	
			0	Полученный адрес не проверяется на совпадение с адресом общего вызова
			1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом общего вызова (0000_000b)
			Бит очищается при выходе ведомого из режима IDLE	
ACK	4	Запись Чтение	Бит квитирования приема. В режиме передатчика не используется. В режиме приемника (мастера/ведомого) содержит значение, которое передается в течение цикла отклика на запрос передатчика подтвердить прием. Передача нуля по окончании передачи байта (квитирование) означает, что данные успешно получены. Передача единицы (неквитирование) означает, что приемник не может продолжать работу по каким-либо причинам. Бит ACK очищается аппаратно по окончании цикла отклика.	
INTEN	2	Запись Чтение	Бит разрешения прерывания	
			0	Запрещено
			1	Разрешено
STOP	1	Запись Чтение	Бит останова. В режиме мастера установка бита STOP генерирует состояние останова, которое завершает или прерывает текущую передачу. После прекращения передачи бит STOP очищается аппаратно	
START	0	Запись Чтение	Бит старта. Этот бит устанавливается, когда требуется сформировать стартовое состояние на шине. Бит START очищается аппаратно по окончании цикла стартового состояния, а также при обнаружении ошибки на шине (состояние с кодом 1Fh)	
–	3	–	Зарезервировано	

Таблица А.44 – Регистр SMBCTRL2

SMBCTRL2
регистр 2 управления I2C

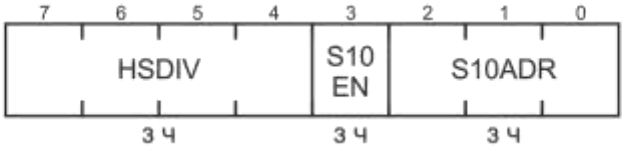
1FBAh значение после Сброса: 00h

Поле	Бит	Тип	Описание	
SCLFRQ	7-1	Запись Чтение	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме мастера. Длительности высокого(T_{SCLH}) и низкого(T_{SCLL}) уровней сигнала SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формуле: $T_{SCLH} = T_{SCLL} = 2 \times SCLFRQ \times (1/fosc)$. (A.3) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{SCLH} + T_{SCLL})$. (A.4) В поле SCLFRQ можно записать любое значение в диапазоне от 04h до 7Fh. При попытке записи любого значения меньше «04h», оно будет записано со смещением 04h. Например, при записи числа 02h, к нему будет аппаратно добавлено смещение 04h и, в итоге, в поле SCLFRQ окажется значение «06h»	
ENABLE	0	Запись Чтение	Бит включения модуля I2C	
			0	Модуль выключен. Тактирование не осуществляется. Регистры SMBCTRL1, SMBST, SMBCST сброшены
			1	Модуль включен

Таблица А.45 – Регистр SMBCTRL3

SMBCTRL3
регистр 3 управления I2C

1FBEh — значение после Сброса: 00h



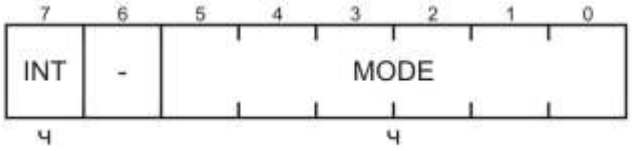
Поле	Бит	Тип	Описание
1	2	3	4
HSDIV	7-4	Запись Чтение	Поле выбора частоты f_{SCL} сигнала на выводе SCL в режиме HS мастера. Длительности высокого(T_{HSCLH}) и низкого(T_{HSCLL}) уровней сигнала на выводе SCL зависят от тактовой частоты fosc модуля I2C и рассчитываются по формулам $T_{HSCLH} = HSDIV \times (1/fosc)$, (A.5) $T_{HSCLL} = 2 \times HSDIV \times (1/fosc)$. (A.6) Таким образом, частота сигнала на выводе SCL равна $f_{SCL} = 1/(T_{HSCLH} + T_{HSCLL})$. (A.7) В поле HSDIV можно записать любое значение в диапазоне от 2h до Fh. При попытке записи любого значения меньше «2h» в поле HSDIV, оно будет записано со смещением 2h. Например, при записи числа 1h к нему будет аппаратно добавлено смещение 2h и, в итоге, в поле SCLFRQ окажется значение «3h».
S10EN	3	Запись Чтение	Бит разрешения 10-битной адресации ведомого 0 – Запрещена 1 – Разрешена при условии, что установлен бит SAEN в регистре SMBADDR.
S10ADR	2-0	Запись Чтение	Поле старших битов 10-битного адреса ведомого. Поле содержит старшие три разряда адреса ведомого при 10-битной адресации. Первый принятый байт адреса сравнивается со значением [11110b, S10ADR[2:1]], второй байт адреса – со значением [S10ADR[0], ADDR]*

* Скобки указывают на то, что заключенное в них число получается конкатенацией значений чисел, разделенных запятой; S10ADR[2:1] и S10ADR[0] – соответственно значения старших двух битов и младшего бита поля S10ADR; ADDR – значение поля регистра SMBADDR.

Таблица А.46 – Регистр SMBST

SMBST
регистр состояния I2C

1FB2h значение после Сброса: 00h

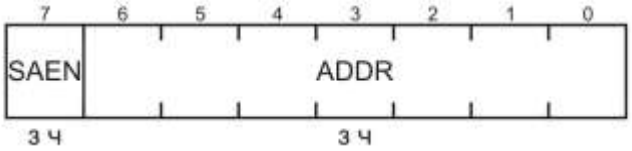


Поле	Бит	Тип	Описание
INT	7	Чтение	Флаг прерывания. Устанавливается после 9-го такта сигнала SCL (когда SCL = 0) в любое запрограммированное время. Условия выставления флага INT: - во время приема/передачи как в режиме мастера, так и в режиме ведомого; - при совпадении адреса (адреса ведомого, адреса отклика или адреса общего вызова) содержимое регистра SMBSDA должно контролироваться программно для определения типа полученного адреса; - после успешного формирования стартового состояния или состояния повторного старта; - в случае неквитирования переданной информации; - при потере арбитража во время передачи последнего бита; - при обнаружении валидного состояния останова или состояния повторного старта; - при обнаружении ошибки на шине. Пока установлен флаг INT, на линии SCL удерживается низкий уровень сигнала. Флаг INT может быть сброшен установкой бита CLRST в регистре SMBCTRL1 или выключением модуля I2C (обнуление бита ENABLE в регистре SMBCTRL2) Условия выставления флага INT (не влияющие на уровень сигнала на линии SCL): - простой на линии SCL; - состояние останова в режиме ведомого (MODE = 1Ch); - потеря арбитража, вследствие чего ведомый переключился в безадресный режим (MODE = 03h или MODE = 23h); - неквитированная передача байта данных (MODE = 17h)
MODE	5-0	Чтение	Код состояния. Возникновение того или иного состояния в течение функционирования модуля I2C сопровождается записью соответствующего кода в поле MODE. Возможные коды и описание состояний приведены в таблицах 11.2 и 11.3
–	6	–	Зарезервировано

Таблица А.47 – Регистр SMBADDR

SMBADDR
регистр собственного адреса I2C

1FB8h значение после Сброса: 00h

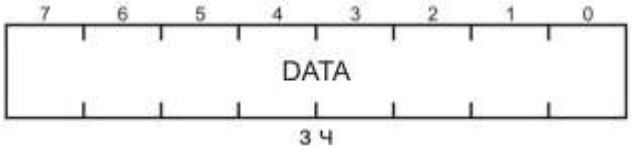


Поле	Бит	Тип	Описание	
SAEN	7	Запись Чтение	Бит разрешения распознавания адреса	
			0	Безадресный режим
			1	Включена функция распознавания принятого адреса
ADDR	6-0	Запись Чтение	Поле собственного 7-битного адреса. При работе в режиме ведомого первые 7 бит, принятые после стартового состояния, сравниваются со значением ADDR. Если обнаружено совпадение и установлен бит SAEN, ведомый переходит в режим приемника или передатчика (в зависимости от состояния бита направления R/W#)	

Таблица А.48 – Регистр SMBSDA

SMBSDA
сдвиговый регистр данных I2C

1FB0h значение после Сброса: XXh

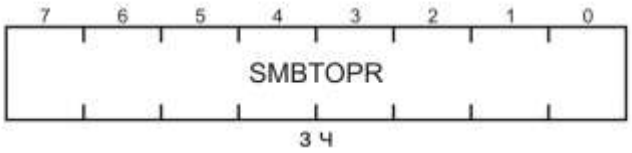


Поле	Бит	Тип	Описание
DATA	7-0	Запись Чтение	Поле данных

Таблица А.49 – Регистр SMBTOPR

SMBTOPR
регистр загрузки предделителя

1FBCh значение после Сброса: 00h



Поле	Бит	Тип	Описание
SMBTOPR	7-0	Запись Чтение	Поле значения перезагрузки предделителя

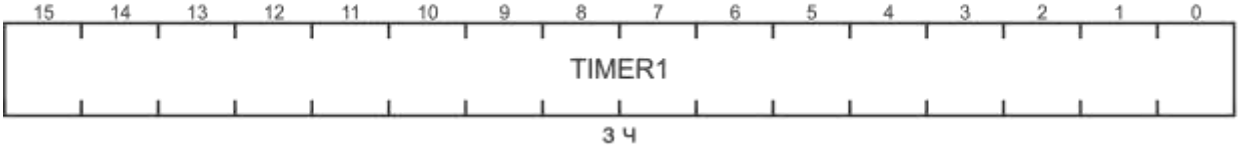
Регистры модуля HSIO

Таблица А.50 – Регистр TIMER1

TIMER1
регистр таймера 1

0Ah значение после Сброса: 0000h

запись - горизонтальное окно 15
чтение - горизонтальное окно 0



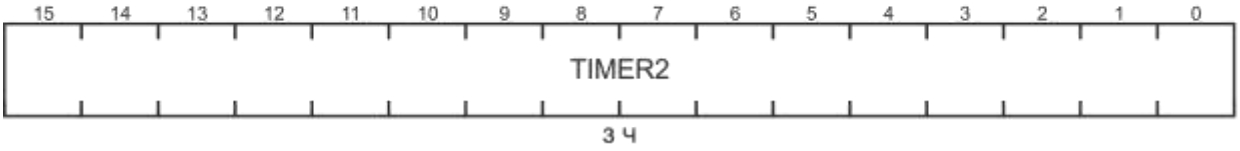
Поле	Бит	Тип	Описание
TIMER1	15-0	Запись Чтение	Поле счетчика таймера

Таблица А.51 – Регистр TIMER2

TIMER2
регистр таймера 2

0Ch значение после Сброса: 0000h

запись/чтение - горизонтальное окно 0



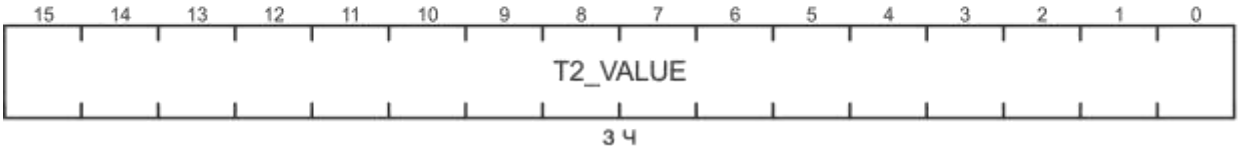
Поле	Бит	Тип	Описание
TIMER2	15-0	Запись Чтение	Поле счетчика таймера

Таблица А.52 – Регистр T2CAPTURE

T2CAPTURE
регистр захвата таймера 2

0Ch значение после Сброса: XXXXh

запись/чтение - горизонтальное окно 15



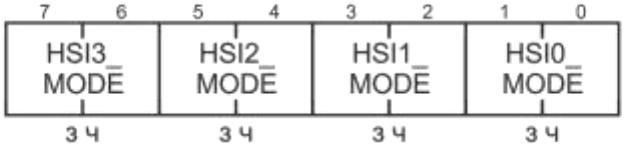
Поле	Бит	Тип	Описание
T2_VALUE	15-0	Запись Чтение	Захваченное значение таймера 2. Захват состояния счетчика таймера происходит после обнаружения положительного фронта сигнала на выводе P2.7 (T2CAP) микроконтроллера

Таблица А.53 – Регистр HSI_MODE

HSI_MODE
регистр режима HSI

03h значение после Сброса: FFh

запись - горизонтальное окно 0
чтение - горизонтальное окно 15



Поле	Бит	Тип	Описание
HSI3_MODE	7-6	Запись Чтение	Поле задания режима канала 3
HSI2_MODE	5-4	Запись Чтение	Поле задания режима канала 2
HSI1_MODE	3-2	Запись Чтение	Поле задания режима канала 1
HSI0_MODE	1-0	Запись Чтение	Поле задания режима канала 0

Примечание – Для каждого канала может быть задан один из четырех режимов работы. Для задания режима следует использовать комбинации:
00 – фиксация каждого восьмого положительного фронта;
01 – фиксация каждого положительного фронта;
10 – фиксация каждого отрицательного фронта;
11 – фиксация каждого фронта.

Таблица А.54 – Регистр HSI_STATUS

HSI_STATUS
регистр состояния HSI

06h значение после Сброса: X0X0X0X0b

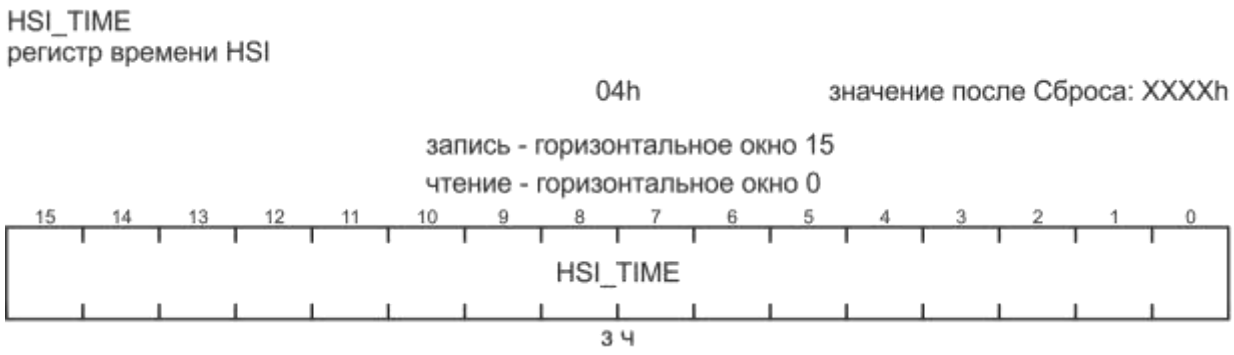
запись битов 0, 2, 4, 6 - горизонтальное окно 15
чтение - горизонтальное окно 0

Поле	Бит	Тип	Описание	
1	2	3	4	
HSI3_STAT	7	Запись Чтение	Индикатор текущего состояния канала 3	
HSI3_EVENT	6	Запись Чтение	Индикатор события в канале 3	
			0	Нет события
			1	Обнаружено событие
HSI2_STAT	5	Запись Чтение	Индикатор текущего состояния канала 2	

Окончание таблицы А.54

1	2	3	4	
HSI2_EVENT	4	Запись Чтение	Индикатор события в канале 2	
			0	Нет события
			1	Обнаружено событие
HSI1_STAT	3	Запись Чтение	Индикатор текущего состояния канала 1	
HSI1_EVENT	2	Запись Чтение	Индикатор события в канале 1	
			0	Нет события
			1	Обнаружено событие
HSI0_STAT	1	Запись Чтение	Индикатор текущего состояния канала 0	
HSI0_EVENT	0	Запись Чтение	Индикатор события в канале 0	
			0	Нет события
			1	Обнаружено событие

Таблица А.55 – Регистр HSI_TIME



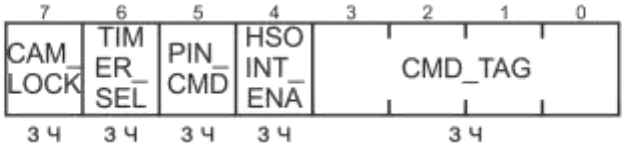
Поле	Бит	Тип	Описание
HSI_TIME	15-0	Запись Чтение	Поле времени события

Таблица А.56 – Регистр HSO_COMMAND

HSO_COMMAND
регистр управления HSO

06h значение после Сброса: XXh

запись - горизонтальное окно 0
чтение - горизонтальное окно 15

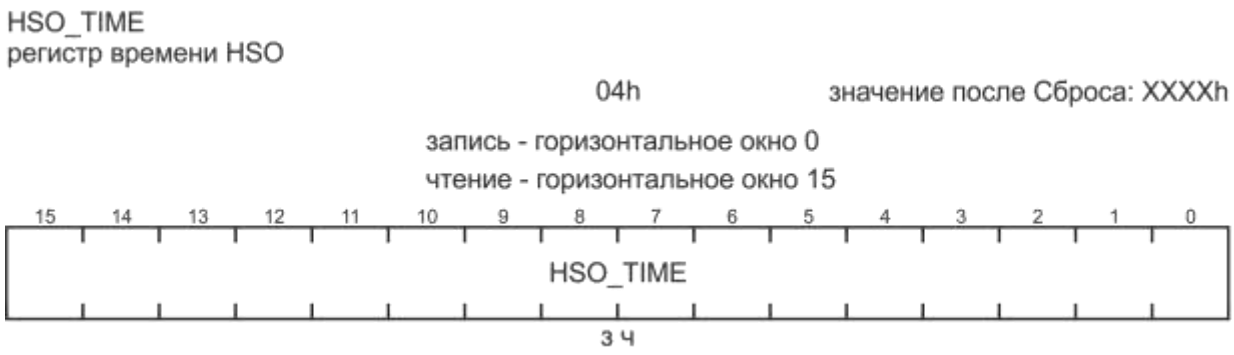


Поле	Бит	Тип	Описание	
CAM_LOCK	7	Запись Чтение	Бит фиксации команды в АЗУ. На работу бита влияет состояние бита LOCK_ENA регистра IOC2	
			0	Ячейка, в которой хранится команда, полностью очищается после выполнения команды
			1	Ячейка, в которой хранится команда, фиксируется, и команда будет выполняться каждый раз, когда запрограммированное значение времени будет совпадать со значением таймера
			Независимо от состояния бита CAM_LOCK, все (!) ячейки АЗУ очищаются, если устанавливается бит CAM_CLR регистра IOC2	
TIMER_SEL	6	Запись Чтение	Бит выбора таймера	
			0	Таймер 1
			1	Таймер 2
PIN_CMD	5	Запись Чтение	Бит выбора воздействия команды на вывод/выводы модуля HSO	
			0	Результатом выполнения команды является установка логического нуля на выходе/выходах
			1	Результатом выполнения команды является установка логической единицы на выходе/выходах
HSOINT_ ENA	4	Запись Чтение	Бит разрешения прерывания при выполнении действия. Дополнительно прерывание маскируется битом HSO_MASK регистра INT_MASK	
			0	Запрещено
			1	Разрешено
CMD_TAG	3-0	Запись Чтение	Код команды. Определяет, какое действие будет активировано во время, заданное в HSO_TIME (см. таблицу А.58)	

Таблица А.57 – Коды команд и соответствующие им действия

CMD_TAG	Мнемоника команды	Действие
0h	HSO0	Переключение выхода HSO.0
1h	HSO1	Переключение выхода HSO.1
2h	HSO2	Переключение выхода HSO.2
3h	HSO3	Переключение выхода HSO.3
4h	HSO4	Переключение выхода HSO.4
5h	HSO5	Переключение выхода HSO.5
6h	HSO01	Переключение выхода HSO.0 и HSO.1
7h	HSO23	Переключение выхода HSO.2 и HSO.3
8h	SWT0	Программное прерывание 0
9h	SWT1	Программное прерывание 1
Ah	SWT2	Программное прерывание 2
Bh	SWT3	Программное прерывание 3
Ch	HSOALL	Переключение выходов HSO.0 – HSO.5
Dh	–	Зарезервировано
Eh	T2RST	Сброс таймера 2
Fh	A_D	Начало аналого-цифрового преобразования

Таблица А.58 – Регистр HSO_TIME



Поле	Бит	Тип	Описание
HSO_TIME	15-0	Запись Чтение	Поле задания времени выполнения команды

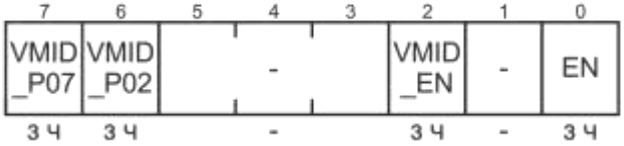
Регистры АЦП

Таблица А.59 – Регистр ADC_EN

ADC_EN
регистр инициализации АЦП

значение после Сброса: 00h

1FC2h



Поле	Бит	Тип	Описание	
VMID_P07	7	Запись Чтение	Бит подключения внешнего источника напряжения средней точки с вывода P0.7 к цепи U_{MID} (см. рисунок 13.1)	
			0	Отключен
			1	Подключен. Биты VMID_P02 и VMID_EN должны быть сброшены
VMID_P02	6	Запись Чтение	Бит подключения внешнего источника напряжения средней точки с вывода P0.2 к цепи U_{MID} (см. рисунок 13.1)	
			0	Отключен
			1	Подключен. Биты VMID_P07 и VMID_EN должны быть сброшены
VMID_EN	2	Запись Чтение	Бит подключения внутреннего источника напряжения средней точки	
			0	Отключен
			1	Подключен. Биты VMID_P02 и VMID_P07 должны быть сброшены
EN	0	Запись Чтение	Бит включения АЦП	
			0	Выключен
			1	Включен
–	5-3, 1	_	Зарезервировано	

Таблица А.60 – Регистр ADC_CON

ADC_CON
регистр управления АЦП

значение после Сброса: 00h

1FC0h

Поле	Бит	Тип	Описание	
1	2	3	4	
EN_DIF	7	Запись Чтение	Бит выбора режима входов АЦП	
			0	Однополярное включение
			1	Дифференциальное включение

Окончание таблицы А.60

1	2	3	4	
CHANNEL	6-3	Запись Чтение	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 (только для ИС 1874ВЕ7Т)
			1010	Канал 10 (только для ИС 1874ВЕ7Т)
			1011	Канал 11
			1100	Канал 12 (только для ИС 1874ВЕ7Т)
			1101	Канал 13 (только для ИС 1874ВЕ7Т)
			1110	Канал 14
			1111	Канал 15
MODE	2-1	Запись Чтение	Поле выбора режима работы АЦП	
			00	Единичное преобразование выбранного канала
			01	Многократное преобразование выбранного канала
			10	Последовательное сканирование каналов
			11	Зарезервировано
START	0	Запись Чтение	Бит запуска аналого-цифрового преобразования	
			0	Нет действий/остановка преобразования
			1	Запуск преобразования

Таблица А.61 – Регистр ADC_RESULT

ADC_RESULT
регистр результата преобразования АЦП

значение после Сброса: 0000h

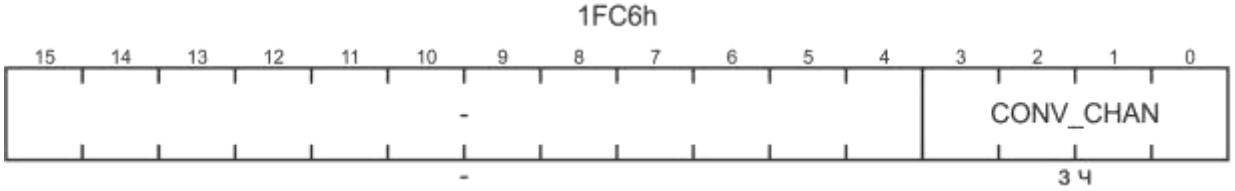
1FC4h

Поле	Бит	Тип	Описание	
BUSY	15	Чтение	Индикатор занятости АЦП	
			0	Нет действий. АЦП в режиме ожидания
			1	Выполняется преобразование
DATA	13-0	Чтение Аппаратное влияние	Поле результата преобразования	
–	14	–	Зарезервировано	

Таблица А.62 – Регистр ADC_TRSCHAN

ADC_TRSCHAN
регистр номера канала АЦП

значение после Сброса: 0000h



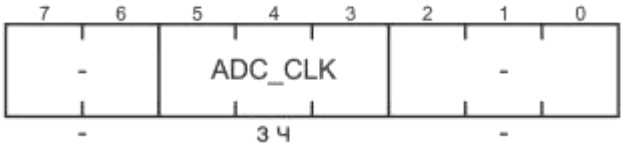
Поле	Бит	Тип	Описание	
CONV_ CHAN	3-0	Запись Чтение	Поле номера канала, над которым выполняется преобразование	
			0000	Канал 0
			0001	Канал 1
			…	
			1111	Канал 15
–	15-4	–	Зарезервировано	

Таблица А.63 – Регистр ADC_SET

ADC_SET
регистр настроек АЦП

значение после Сброса: 20h

1FC1h



Поле	Бит	Тип	Описание		
ADC_CLK	5-3	Запись Чтение	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 XTAL = 24 МГц)
			100	1/16	0,75 МГц (частота XTAL = 24 МГц)
			101	1/32	–
			110	1/64	–
			111	1/128	–
–	7, 6, 2-0	–	Зарезервировано		

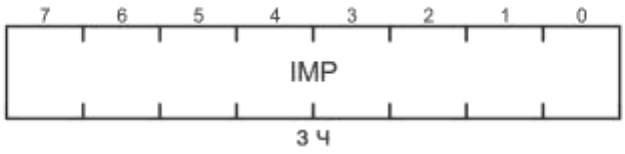
Регистры модуля ШИМ

Таблица А.65 – Регистры PWM0_CONTROL, PWM1_CONTROL, PWM2_CONTROL

PWMx_CONTROL
регистр длительности импульса канала x

значение после Сброса: 00h

a

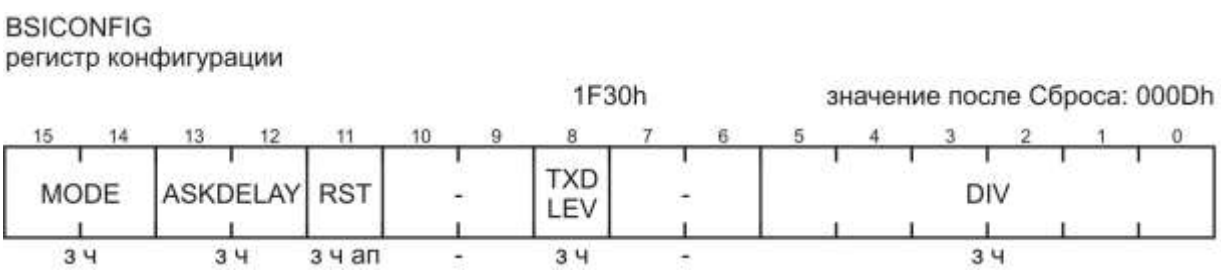


x	0	1	2
Регистр	PWM0_CONTROL	PWM1_CONTROL	PWM2_CONTROL
a	17h (запись - горизонтальное окно 0, чтение - горизонтальное окно 15)	16h (запись/чтение - горизонтальное окно 1)	17h (запись/чтение - горизонтальное окно 1)

Поле	Бит	Тип	Описание
IMP	7-0	Запись Чтение	Длительность импульса формируемого сигнала. Задается в количестве тактов (от 0 до 256)

Регистры контроллера интерфейса ГОСТ Р 52070–2003

Таблица А.66 – Регистр BSICONFIG

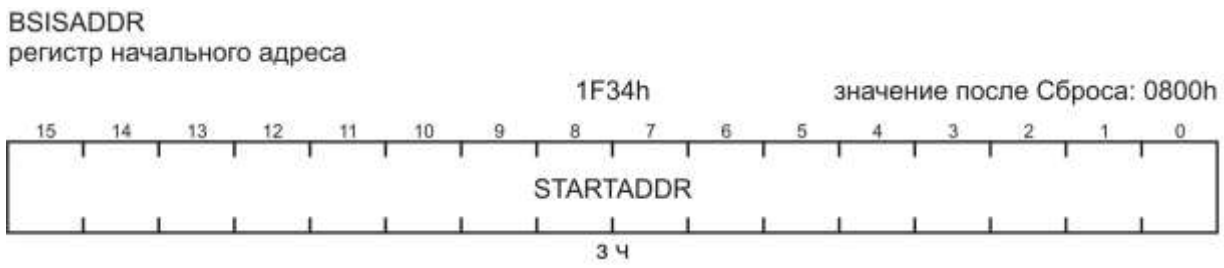


Поле	Бит	Тип	Описание	
1	2	3	4	
MODE	15-14	Запись Чтение	Режимы работы	
			00	Контроллер шины (КШ)
			01	Удаленный терминал (УТ)
			10	Зарезервировано. Не использовать!
			11	Монитор шины (МШ)
ASK DELAY	13-12	Запись Чтение	Поле выбора значения времени ожидания ответного слова	
			00	14 мкс
			01	28 мкс
			10	56 мкс
			11	112 мкс
RST	11	Запись Чтение Аппаратное влияние	Сброс устройства. Установка бита приводит к сбросу модуля и очистки всех его регистров. На данные, хранящиеся в ОЗУ, этот бит не влияет. Бит сбрасывается аппаратно	
TXD LEV	8	Запись Чтение	Бит выбора уровня сигнала. Бит управляет уровнем сигнала на выводах TxD, TxD_n при их неактивном состоянии	
			0	На выводах удерживается низкий уровень сигнала
			1	На выводах удерживается высокий уровень сигнала
DIV	5-0	Запись Чтение	Делитель входной частоты. ГОСТ Р 52070–2003 устанавливает частоту передачи информации, равную 1 МГц. Для получения этой частоты передачи в модуле предусмотрен делитель, значение которого может быть получено из формулы: $DIV_{DEC} = F_{XTAL1}/k$, (А.8) где DIV_{DEC} – значение делителя в десятичном формате, F_{XTAL1} – значение внешней частоты в МГц, k – коэффициент деления. При выключенном внутреннем делителе внешней частоты (сброшен бит SLOW регистра CLKC) k = 4, при включенном (бит SLOW установлен) – k = 8. Минимально допустимое значение k = 3. Так, при F_{XTAL1} = 24 МГц и SLOW = 0 DIV_{DEC} = 24/4 = 6. Значит, для получения частоты передачи 1 МГц в поле DIV следует записать значение 6h.	

Окончание таблицы А.66

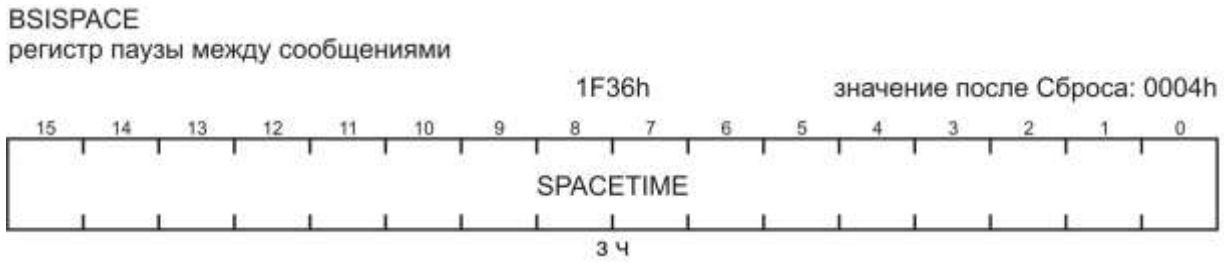
1	2	3	4
–	10-9, 7-6	–	Зарезервировано

Таблица А.67 – Регистр BSISADDR



Поле	Бит	Тип	Описание
STARTADDR	15-0	Запись Чтение	Стартовый адрес. Адрес в памяти сообщений, по которому расположено управляющее слово первого блока сообщения. По умолчанию, стартовый адрес 0800h – начало памяти сообщений

Таблица А.68 – Регистр BSISPACE

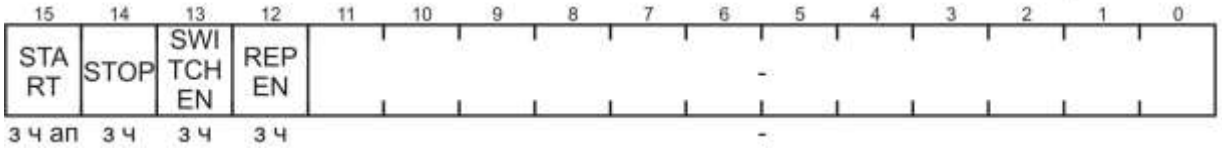


Поле	Бит	Тип	Описание
SPACETIME	15-0	Запись Чтение	Пауза между сообщениями. Это битовое поле задает временной промежуток, которым контроллер шины разделяет передаваемые сообщения. Единицей измерения является величина, равная времени передачи по шине одного бита информации. По умолчанию значение паузы составляет четыре бита, что является минимальным значением. Так, на скорости передачи 1 МГц минимальная пауза будет равна 4 мкс, а максимальная – 65,5 мс.

Таблица А.69 – Регистр BSICON в режиме контроллера шины

BSICON
регистр управления в режиме контроллера шины

1F32h значение после Сброса: 0000h

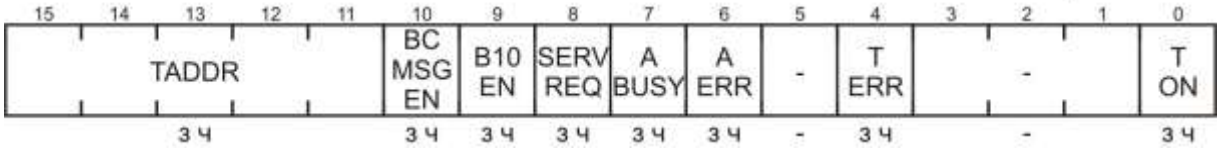


Поле	Бит	Тип	Описание	
START	15	Запись Чтение Аппаратное влияние	Бит запуска Установка бита инициирует передачу сообщения. Сбрасывается аппаратно: - после успешной передачи сообщения, если не был установлен бит NOTLASTMSG управляющего слова; - после установки бита STOP, но только по окончании передачи сообщения; - в результате обнаруженной ошибки	
STOP	14	Запись Чтение	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	
SWITCHEN	13	Запись Чтение	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
REPEN	12	Запись Чтение	Бит разрешения повтора передачи. Управляет механизмом повторной отправки сообщения	
			0	Повтор в случае ошибки запрещен
			1	Количеством повторов управляет поле REPNUM управляющего слова
–	11-0	–	Зарезервировано	

Таблица А.70 – Регистр BSICON в режиме удаленного терминала

BSICON
регистр управления в режиме удаленного терминала

1F32h — значение после Сброса: 0000h

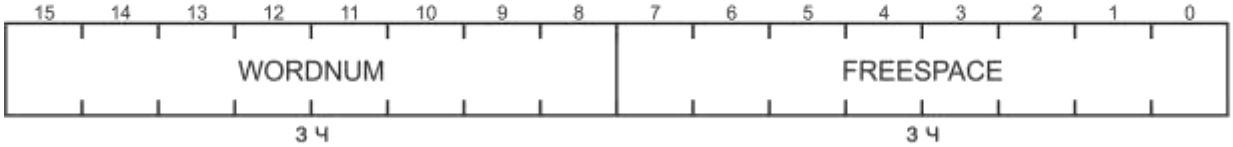


Поле	Бит	Тип	Описание
TADDR	15-11	Запись Чтение	Собственный адрес терминала
BCMSGEN	10	Запись Чтение	Бит разрешения работы с групповыми сообщениями
			0 Запрещено. Адрес 1Fh игнорируется
			1 Разрешено. Адрес 1Fh распознается как групповой
B10EN	9	Запись Чтение	Бит разрешения контроля признака слова
			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
SERVREQ	8	Запись Чтение	Бит запроса на обслуживание. Запись единицы в этот бит установит соответствующий бит в ответном слове
ABUSY	7	Запись Чтение	Бит занятости абонента. Запись единицы в этот бит установит соответствующий бит в ответном слове
AERR	6	Запись Чтение	Бит неисправности абонента. Запись единицы в этот бит установит соответствующий бит в ответном слове
TERR	4	Запись Чтение	Бит неисправности терминала. Запись единицы в этот бит установит соответствующий бит в ответном слове
TON	0	Запись Чтение	Бит включения терминала. Пока этот бит установлен, собственный адрес терминала не может быть изменен
–	5, 3-1	–	Зарезервировано

Таблица А.71 – Регистр BSICON в режиме монитора шины

BSICON
регистр управления в режиме монитора шины

1F32h — значение после Сброса: 0000h



Поле	Бит	Тип	Описание
1	2	3	4
WORDNUM	15-8	Запись Чтение	Поле количества слов

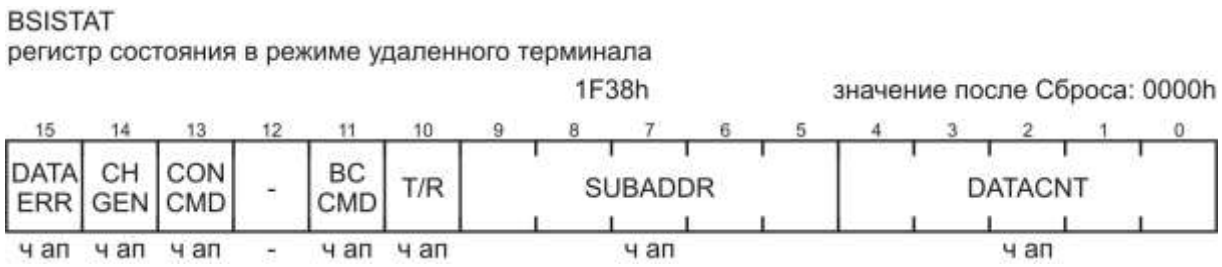
Окончание таблицы А.71

1	2	3	4
FREESPACE	7-0	Запись Чтение	Поле определения объема свободного пространства (в количестве слов). Значение этого поля учитывается только, если значение поля WORDNUM равно 00h

Таблица А.72 – Регистр BSISTAT в режиме контроллера шины

Поле	Бит	Тип	Описание
DATAERR	15	Чтение Аппаратное влияние	Любая ошибка
CHGEN	14	Чтение Аппаратное влияние	Генерация в канале передачи
TEMPTRANEND	13	Чтение Аппаратное влияние	Флаг промежуточного сообщения цепочки. Устанавливается по окончании передачи сообщения, в управляющем слове которого бит MSGINTEN был установленным
ALLMSGEND	12	Чтение Аппаратное влияние	Флаг последнего сообщения цепочки. Устанавливается по окончании передачи сообщения, в управляющем слове которого бит NOTLASTMSG не был установлен
BCCMD	11	Чтение Аппаратное влияние	Индикатор общего вызова. Бит устанавливается, если последней отправленной командой была команда с групповым адресом
T/R	10	Чтение Аппаратное влияние	Бит направления передачи. Состояние этого бита всегда копирует состояние бита T/R отправленного командного слова
SUBADDR	9-5	Чтение Аппаратное влияние	Подадрес. Состояние этого поля всегда копирует состояние поля SUBADDR/MODE отправленного командного слова
DATACNT	4-0	Чтение Аппаратное влияние	Длина последней команды. Состояние этого поля всегда копирует состояние поля DATACNT/CODE отправленного командного слова

Таблица А.73 – Регистр BSISTAT в режиме удаленного терминала



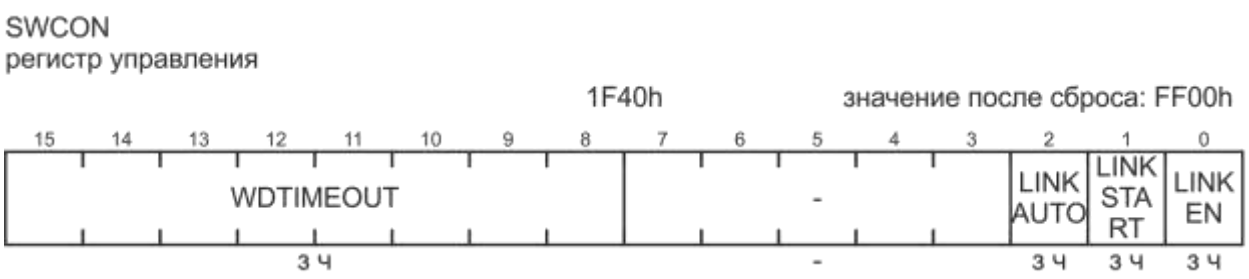
Поле	Бит	Тип	Описание
DATAERR	15	Чтение Аппаратное влияние	Любая ошибка
CHGEN	14	Чтение Аппаратное влияние	Генерация в канале
CONCMD	13	Чтение Аппаратное влияние	Индикатор получения команды управления
BCCMD	11	Чтение Аппаратное влияние	Индикатор общего вызова. Устанавливается, если получен групповой адрес 1Fh и если установлен бит BCMSGEN в регистре BSICON
T/R	10	Чтение Аппаратное влияние	Бит направления передачи. Состояние этого бита всегда копирует состояние бита T/R полученного командного слова
SUBADDR	9-5	Чтение Аппаратное влияние	Подадрес. Состояние этого поля всегда копирует состояние поля SUBADDR/MODE полученного командного слова
DATACNT	4-0	Чтение Аппаратное влияние	Количество данных. Состояние этого поля всегда копирует состояние поля DATACNT/CODE полученного командного слова
–	12	–	Зарезервировано

Таблица А.74 – Регистр BSISTAT в режиме монитора шины

Поле	Бит	Тип	Описание
MSGSTARTADDR	15-0	Чтение Аппаратное влияние	Адрес начала блока

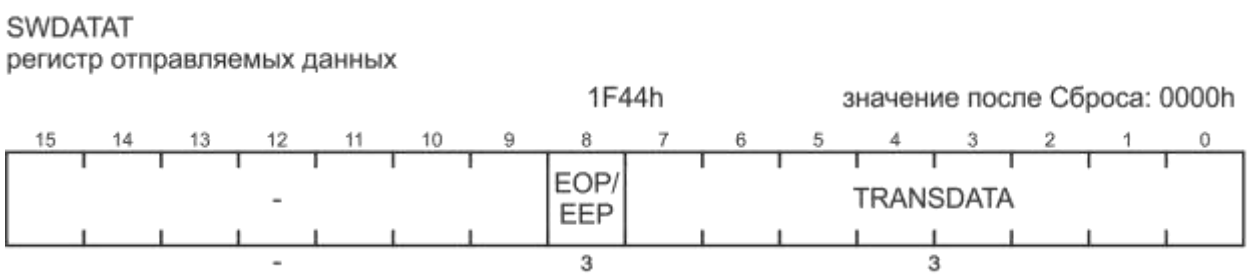
Регистры контроллера интерфейса SpaceWire

Таблица А.75 – Регистр SWCON



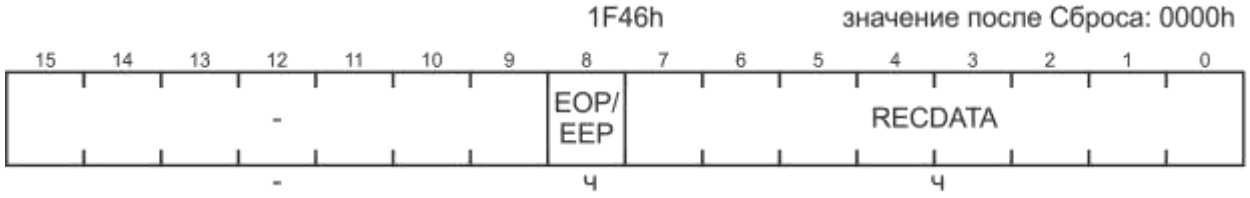
Поле	Бит	Тип	Описание
WDTIMEOUT	15-8	Запись Чтение	Временной коэффициент. Поле устанавливает значение времени потери соединения 800 нс, состояния сброса 6,4 мкс и других состояний 12,8 мкс. Значение временного коэффициента может быть получено из формулы: $WDTIMEOUT_{DEC} = k \times F_{XTAL1}$, (A.9) 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 CLKC) 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	Запись Чтение	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	Запись Чтение	Бит установления связи. Устанавливается, если модуль является инициатором передачи
		0	Нет связи
		1	Инициация передачи
LINKEN	0	Запись Чтение	Бит установки соединения
		0	Модуль отключен от сети
		1	Модуль подключен к сети
–	7-3	–	Зарезервировано

Таблица А.76 – Регистр SWDATAT



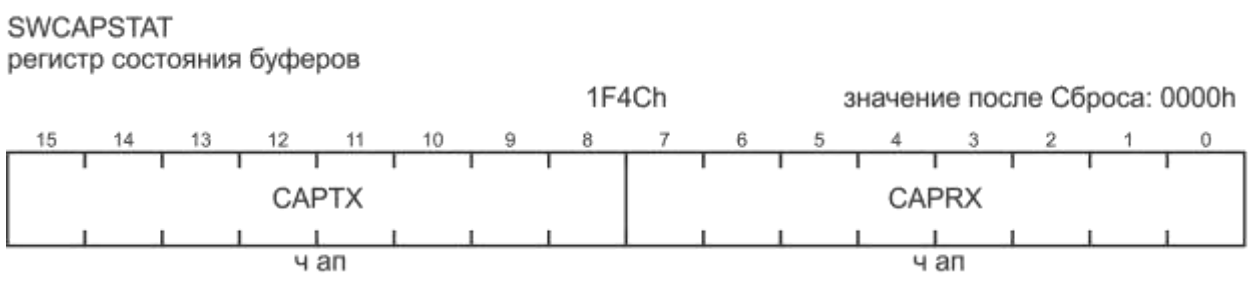
Поле	Бит	Тип	Описание
EOP/EEP	8	Запись	Маркер конца/ошибки пакета. Этот бит используется совместно с полем TRANSDATA как указатель окончания передачи пакета. Таким образом, если в регистр SWDATAT было записано значение 00xxh, то это указывает на байт данных («xx» – байт данных). Если в регистр SWDATAT было записано значение: – 0100h, то это указатель конца пакета; – 0101h, то это указатель ошибки, обнаруженной передатчиком в переданных данных
TRANSDATA	7-0	Запись	Байт данных для передачи При записи байт из этого поля загружается в передающий FIFO-буфер
–	15-9	–	Зарезервировано

Таблица А.77 – Регистр SWDATAR



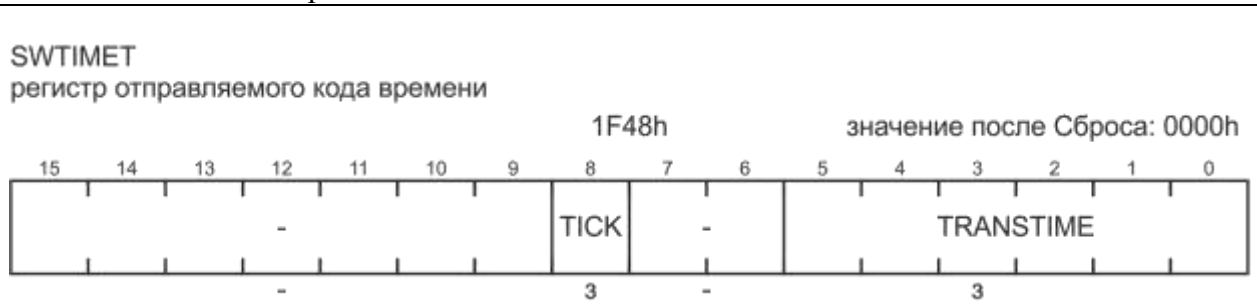
Поле	Бит	Тип	Описание
EOP/EEP	8	Чтение	Маркер конца/ошибки пакета. Этот бит используется совместно с полем RECDATA как указатель окончания передачи пакета. Таким образом, если при обращении к регистру SWDATAR было прочитано значение 00xxh, то это указывает на байт данных («xx» – байт данных). Если прочитано значение: – 0100h, то это указатель конца пакета; – 0101h, то это указатель ошибки, обнаруженной передатчиком в передаваемых данных
RECDATA	7-0	Чтение	Байт принятых данных. При чтении в это поле переписывается байт из принимающего FIFO-буфера
–	15-9	–	Зарезервировано

Таблица А.78 – Регистр SWCAPSTAT



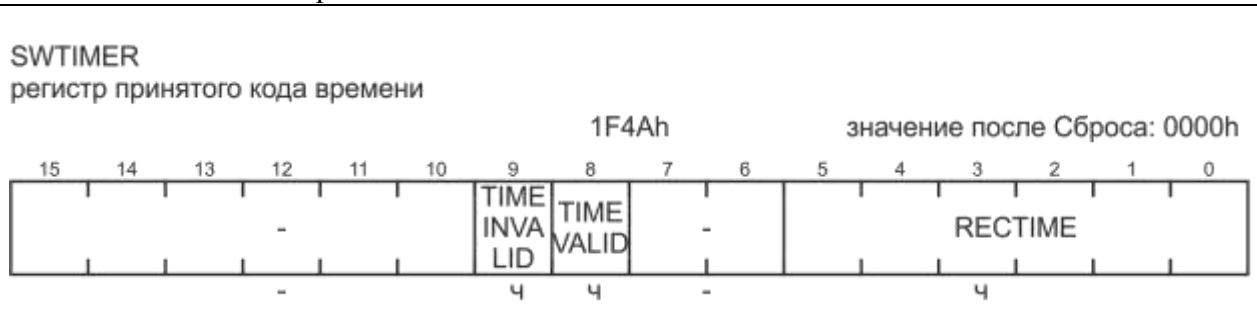
Поле	Бит	Тип	Описание
CAPTX	15-8	Чтение Аппаратное влияние	Число занятых ячеек (байт) передающего буфера
CAPRX	7-0	Чтение Аппаратное влияние	Число занятых ячеек (байт) принимающего буфера

Таблица А.79– Регистр SWTIMET



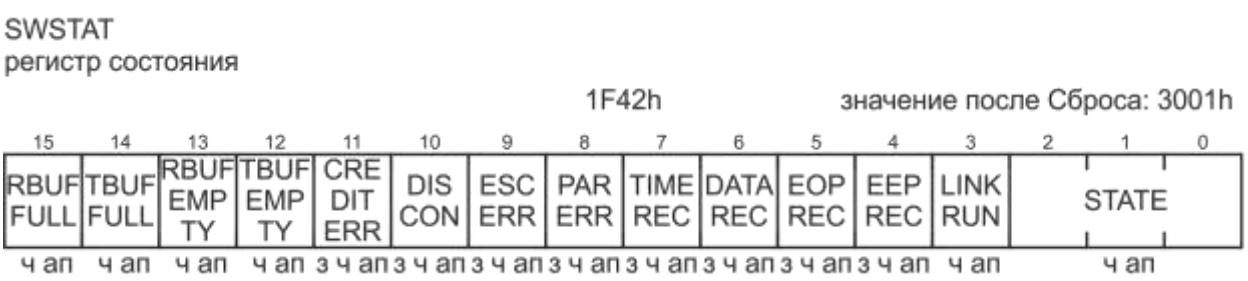
Поле	Бит	Тип	Описание
TICK	8	Запись	Бит инициализации отправки кода времени
TRANSTIME	5-0	Запись	Поле кода времени
–	15-9, 7-6	–	Зарезервировано

Таблица А.80 – Регистр SWTIMER



Поле	Бит	Тип	Описание
TIME INVALID	9	Чтение	Флаг некорректного принятого кода времени
TIME VALID	8	Чтение	Флаг корректного принятого кода времени
RECTIME	5-0	Чтение	Принятый код времени
–	15-10,7-6	–	Зарезервировано

Таблица А.81 – Регистр SWSTAT



Поле	Бит	Тип	Описание	
1	2	3	4	
RBUF FULL	15	Чтение Аппаратное влияние	Индикатор заполнения принимающего FIFO-буфера	
			0	В буфере есть как минимум одна свободная ячейка
			1	Буфер заполнен
TBUF FULL	14	Чтение Аппаратное влияние	Индикатор заполнения передающего FIFO-буфера	
			0	В буфере есть как минимум одна свободная ячейка
			1	Буфер заполнен
RBUF EMPTY	13	Чтение Аппаратное влияние	Индикатор пустоты принимающего FIFO-буфера	
			0	Как минимум одна ячейка буфера заполнена
			1	Буфер пуст
TBUF EMPTY	12	Чтение Аппаратное влияние	Индикатор пустоты передающего FIFO-буфера	
			0	Как минимум одна ячейка буфера заполнена
			1	Буфер пуст
CREDIT ERR	11	Запись Чтение Аппаратное влияние	Бит устанавливается при ошибке кредитования буфера приемника или передатчика (неправильном определении свободного места в устройстве). Флаг сбрасывается программно	
DIS CON	10	Запись Чтение Аппаратное влияние	Флаг потери соединения. Устанавливается при обнаружении потери соединения с сетью SpaceWire. Маскируется битом DISCONINT регистра SWINTMASK. Сбрасывается программно	
ESC ERR	9	Запись Чтение Аппаратное влияние	Флаг ошибки последовательности управляющих символов Устанавливается при обнаружении ошибки следования управляющих символов во время приема. Маскируется битом ESCINT регистра SWINTMASK. Сбрасывается программно	
PAR ERR	8	Запись Чтение Аппаратное влияние	Флаг ошибки паритета. Устанавливается при обнаружении ошибки четности принятой информации. Маскируется битом PARINT регистра SWINTMASK. Сбрасывается программно	
TIME REC	7	Запись Чтение Аппаратное влияние	Флаг приема кода времени. Устанавливается, если принятый байт является кодом времени. Маскируется битом TIMEINT регистра SWINTMASK. Флаг сбрасывается программно	

Окончание таблицы А.81

1	2	3	4
DATA REC	6	Запись Чтение Аппаратное влияние	Флаг приема данных. Устанавливается по окончании приема байта данных. Маскируется битом DATAINT регистра SWINTMASK. Сбрасывается программно
EOP REC	5	Запись Чтение Аппаратное влияние	Флаг приема маркера EOP. Устанавливается, если принят код 0100h, соответствующий маркеру конца пакета. Маскируется битом EOPINT Сбрасывается программно
EEP REC	4	Запись Чтение Аппаратное влияние	Флаг приема маркера EEP. Устанавливается, если принят код 0101h, соответствующий маркеру ошибки пакета. Маскируется битом EEPINT. Сбрасывается программно
LINK RUN	3	Чтение Аппаратное Влияние	Индикатор соединения с сетью SpaceWire
			0 — Связь не установлена
			1 — Связь установлена
STATE	2-0	Чтение Аппаратное влияние	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

Таблица А.82 – Регистр SWINTMASK

SWINTMASK
регистр маски прерываний

1F4Eh — значение после Сброса: 0000h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF FULL INT	TBUF FULL INT	RBUF EMP INT	TBUF EMP INT	CRE DIT INT	DIS CON INT	ESC INT	PAR INT	TIME INT	DATA INT	EOP INT	EEP INT	RUN INT	-		
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	-		

Поле	Бит	Тип	Описание
1	2	3	4
RBUF FULL INT	15	Запись Чтение	Разрешает прерывание при полном заполнении буфера приемника
TBUF FULL INT	14	Запись Чтение	Разрешает прерывание при полном заполнении буфера передатчика

Окончание таблицы А.82

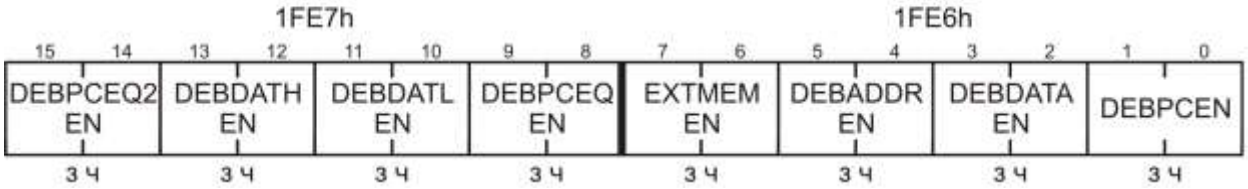
1	2	3	4
RBUF EMP INT	13	Запись Чтение	Разрешает прерывание, когда пустой приемный буфер принимает хотя бы один байт данных (прерывание по выходу из пустого состояния)
TBUF EMP INT	12	Запись Чтение	Разрешает прерывание, когда передающий буфер переходит из частично заполненного в пустое состояние (прерывание по опустошению)
CREDIT INT	11	Запись Чтение	Разрешает прерывание при ошибке кредитования буферов приемника и передатчика
DIS CON INT	10	Запись Чтение	Разрешает прерывание при потере связи
ESC INT	9	Запись Чтение	Разрешает прерывание при получении неверной последовательности управляющих символов
PARINT	8	Запись Чтение	Разрешает прерывание по ошибке четности в полученном символе
TIMEINT	7	Запись Чтение	Разрешает прерывание по получению кода времени (временной метки)
DATAINT	6	Запись Чтение	Разрешает прерывание по получению одного байта данных
EOPINT	5	Запись Чтение	Разрешает прерывание по получению пакета без ошибки
EEPINT	4	Запись Чтение	Разрешает прерывание по получению пакета с ошибкой
RUNINT	3	Запись Чтение	Разрешает прерывание по входу в состояние RUN
–	2-0	–	Зарезервировано

Регистры модуля OCDS

Таблица А.83 – Регистр DEBCTRL

DEBCTRL
регистр управления модулем отладки

значение после Сброса: 0000h



Поле	Бит	Тип	Описание*
DEBPCEQ2 EN	15-14	Запись Чтение	Совпадение текущего значения счетчика команд со значением регистра DEBPCEQ2
DEBDATH EN	13-12	Запись Чтение	Появление значения, заданного в регистре DEBDATH на старших разрядах внутренней шины данных
DEBDATL EN	11-10	Запись Чтение	Появление значения, заданного в регистре DEBDATL на младших разрядах внутренней шины данных
DEBPCEQ EN	9-8	Запись Чтение	Совпадение текущего значения счетчика команд со значением регистра DEBPCEQ
EXTMEM EN	7-6	Запись Чтение	Выборка команды из области внешней памяти
DEBADDR EN	5-4	Запись Чтение	Появление значения, заданного в регистре DEBADDR на внутренней шине адресов операндов
DEBDATA EN	3-2	Запись Чтение	Появление значения, заданного в регистре DEBDATA на внутренней шине данных
DEBPCEN	1-0	Запись Чтение	Текущее значение счетчика команд больше значения регистра DEBPC

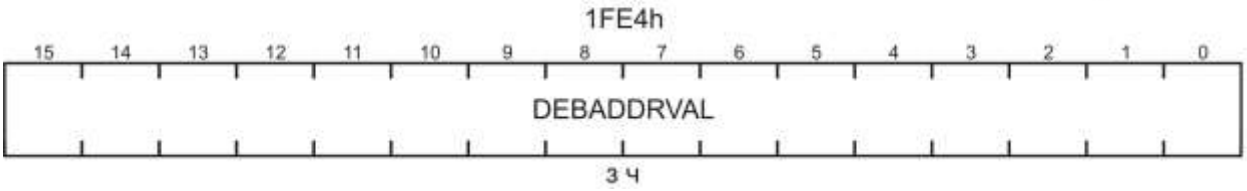
* Здесь указано событие, после обнаружения которого система отладки выполнит запрограммированное действие. Действие зависит от кода, записанного в соответствующем поле:

00 – нет действий;
01 – прерывание DEBUG (2060h);
10 – переход в IDLE;
11 – сброс микроконтроллера.

Таблица А.84 – Регистр DEBADDR

DEBADDR
регистр ожидаемого значения адреса

значение после Сброса: 0000h

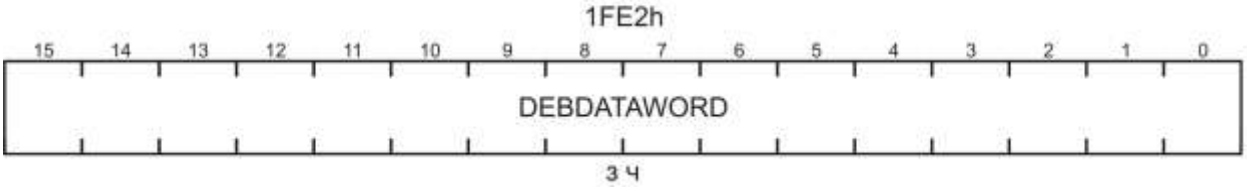


Поле	Бит	Тип	Описание
DEBADDRVAL	15-0	Запись Чтение	Значение адреса, при появлении которого на внутренней шине адресов выполняется действие, заданное полем DEBADDREN регистра DEBCTRL

Таблица А.85 – Регистр DEBDATA

DEBDATA
регистр ожидаемого значения данных

значение после Сброса: 0000h



Поле	Бит	Тип	Описание
DEBDATAWORD	15-0	Запись Чтение	Значение данных, при появлении которых на внутренней шине данных выполняется действие, заданное полем DEBDATAEN регистра DEBCTRL

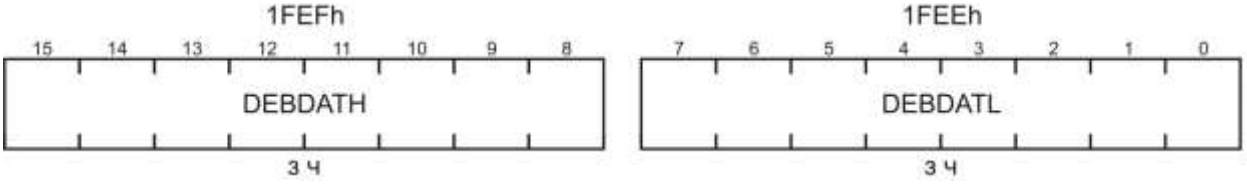
Таблица А.86 – регистры DEBDATH и DEBDATL

DEBDATH
регистр ожидаемого значения
старшего байта данных

значение после Сброса: 00h

DEBDATL
регистр ожидаемого значения
младшего байта данных

значение после Сброса: 00h

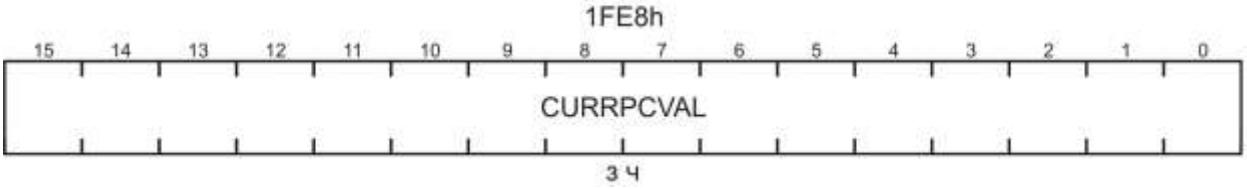


Поле	Бит	Тип	Описание
DEBDATH	15-8	Запись Чтение	Значение старшего байта данных, при появлении которого на старших разрядах внутренней шины данных выполняется действие, заданное полем DEBDATHEN регистра DEBCTRL
DEBDATL	7-0	Запись Чтение	Значение младшего байта данных, при появлении которого на младших разрядах внутренней шины данных выполняется действие, заданное полем DEBDATLEN регистра DEBCTRL

Таблица А.87 – Регистр CURRPC

CURRPC
регистр текущего значения счетчика команд

значение после Сброса: XXXXh

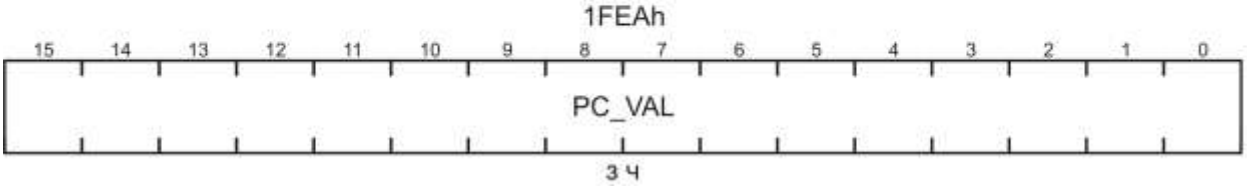


Поле	Бит	Тип	Описание
CURRPCVAL	15-0	Запись Чтение	Текущее значение основного счетчика команд

Таблица А.88 – Регистр HAPPPC

HAPPPC
регистр значения счетчика команд

значение после Сброса: XXXXh

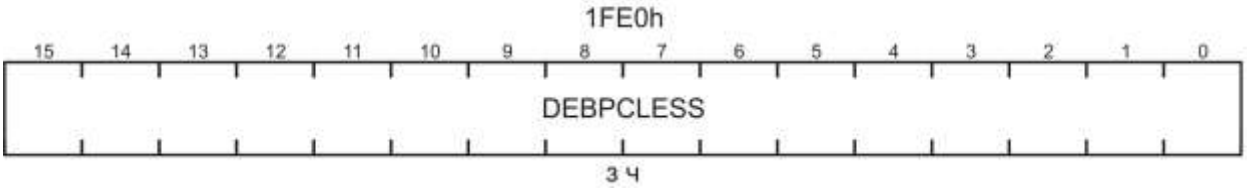


Поле	Бит	Тип	Описание
PC_VAL	15-0	Запись Чтение	Значение счетчика команд, которое было на момент возникновения ожидаемого события. Не сбрасывается во время системного сброса, сохраняя таким образом значение адреса команды, вызвавшей сброс микроконтроллера

Таблица А.89 – Регистр DEBPC

DEBPC
регистр ожидаемого значения счетчика команд

значение после Сброса: 0000h



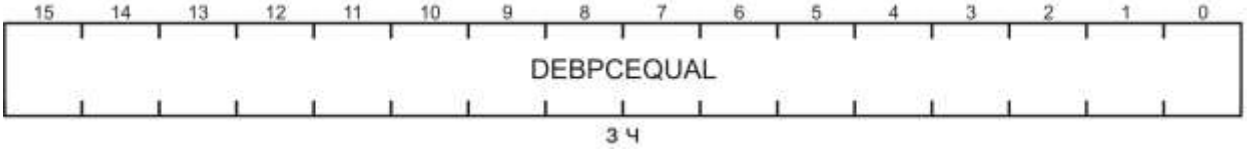
Поле	Бит	Тип	Описание
DEBPCLESS	15-0	Запись Чтение	Значение, при превышении которого счетчиком команд выполняется действие, заданное полем DEBPCEN регистра DEBCTRL

Таблица А.90 – Регистры DEBPCEQ и DEBPCEQ2

DEBPCEQ
регистр ожидаемого значения счетчика команд

значение после Сброса: 0000h

1FECh

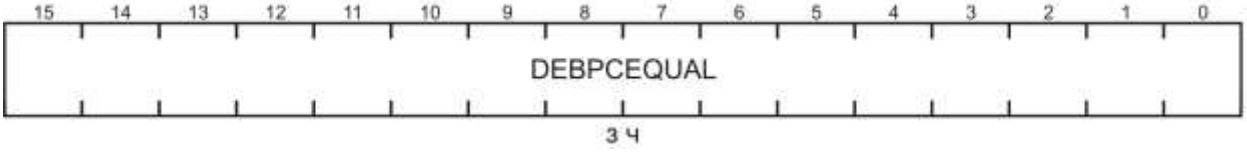


3 Ч

DEBPCEQ2
дополнительный регистр ожидаемого значения счетчика команд

значение после Сброса: 0000h

1FACh



3 Ч

Поле	Бит	Тип	Описание
DEBPCEQUAL	15-0	Запись Чтение	Значение, при достижении которого счетчиком команд выполняется действие, заданное полем DEBPCEQEN регистра DEBCTRL

Приложение Б
(обязательное)

Система команд микроконтроллера

Система команд микроконтроллера включает в себя 111 команд, поддерживаемых стандартными трансляторами с языка ассемблера для микроконтроллеров семейства MCS-196, и команду программного прерывания TRAP, используемую в инструментальных средствах разработки. Команды сложения, вычитания, умножения, деления, логических операций, сравнения, загрузки, а также команды работы с битами и стеком поддерживают до четырех типов адресации.

Краткую информацию о системе команд можно получить из таблиц Б.1 и Б.2.

В таблице Б.3 представлена информация о командах микроконтроллера и вариантах их использования. Для каждой команды указано:

- мнемоническое название;
- название выполняемой операции 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 Б.3 приняты обозначения:

DEST – операнд-приемник;
SRC – операнд-источник;
SRC1 – первый операнд-источник;
SRC2 – второй операнд-источник;
Ireg – регистр двойного слова (32 бита) во внутреннем регистровом файле;
wreg – регистр слова (16 бит) во внутреннем регистровом файле;
breg – регистр байта (8 бит) во внутреннем регистровом файле;
#data – 8/16-разрядная константа;
reg – 8/16-разрядный регистр во внутреннем регистровом файле, используемый для косвенной адресации;
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-разрядная константа для задания количества сдвигов (от 0 до 15) 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.

Далее в настоящем приложении приводится дополнительная информация по всем командам микроконтроллера. В описании используются следующие обозначения:

aa – два младших бита опкода команды, определяющие используемый режим адресации:
00 – прямая регистровая;
01 – непосредственная;
10 – косвенная (в том числе с автоинкрементом);
11 – индексная (короткая и длинная);

DIreg – регистр двойного слова (32 бита) во внутреннем регистровом файле, использующийся в командах как операнд-приемник (DEST). Должен выравниваться по адресу, кратному четырем;

Dwreg – регистр слова (16 бит) во внутреннем регистровом файле, использующийся в командах как операнд-приемник (DEST). Должен выравниваться по адресу, кратному двум;

Dbreg – регистр байта (8 бит) во внутреннем регистровом файле, использующийся в командах как операнд-приемник (DEST);

SIreg – регистр двойного слова (32 бита) во внутреннем регистровом файле, использующийся в командах как операнд-источник (SRC). Должен выравниваться по адресу, кратному четырем;

Swreg – регистр слова (16 бит) во внутреннем регистровом файле, использующийся в командах как операнд-источник (SRC). Должен выравниваться по адресу, кратному двум;

Sbreg – регистр байта (8 бит) во внутреннем регистровом файле, использующийся в командах как операнд-источник (SRC);

wreg – регистр слова (16 бит) во внутреннем регистровом файле. Должен выравниваться по адресу, кратному двум. В командах с двумя и тремя операндами используется как операнд-источник;

breg – регистр байта (8 бит) во внутреннем регистровом файле. В командах с двумя и тремя операндами используется как операнд-источник;

waop – регистр слова (16 бит), адресуемый любым, допускаемым командой способом адресации, и всегда использующийся как операнд-источник (исключение: в командах ST и POP используется как приемник). Должен выравниваться по адресу, кратному двум;

baop – регистр байта (8 бит), адресуемый любым, допускаемым командой способом адресации, и всегда использующийся как операнд-источник (исключение: в команде STB используется как приемник);

cadd – адрес в программном коде, указанный непосредственно, например, JC 20C0h, или как название метки перехода, например, JC label;

disp – смещение, являющееся расстоянием между концом команды и адресом (меткой) перехода. Вычисляется аппаратно;

scount – количество сдвигов, указываемое в командах логического сдвига и командах арифметического сдвига. Количество сдвигов от 0 до 15 может быть задано непосредственно, например, SHR AX, #5, а количество сдвигов от 0 до 31 может быть задано как содержимое какого-либо регистра, в котором хранится значение от 00h до 1Fh, например, SHR AX, BX.

Таблица Б.1 – Команды микроконтроллера для работы со словами и байтами

Мнемоника команды, определяющей операцию над		Название операции
словами	байтами	
Арифметические команды		
ADD	ADDB	Сложение
ADDC	ADDCB	Сложение с переносом
CMP	CMPB	Сравнение
CMPL	–	Сравнение длинных слов
SUB	SUBB	Вычитание
SUBC	SUBCB	Вычитание с переносом
DIV	DIVB	Знаковое деление
DIVU	DIVUB	Беззнаковое деление
MUL	MULB	Знаковое умножение
MULU	MULUB	Беззнаковое умножение
Логические команды		
AND	ANDB	Умножение («И»)
OR	ORB	Сложение («ИЛИ»)
XOR	XORB	Сложение по модулю два (исключающее «ИЛИ»)
Команды работы с данными		
LD	LDB	Загрузка
–	LDBSE	Загрузка со знаковым расширением
–	LDBZE	Загрузка с беззнаковым расширением
ST	STB	Сохранение
CLR	CLRB	Очистка
INC	INCB	Инкремент
DEC	DECB	Декремент
NOT	NOTB	Инверсия
NEG	NEGB	Изменение знака
EXT	EXTB	Знаковое расширение
XCH	XCHB	Обмен
Блочные команды		
BMOV		Непрерываемое перемещение блоков данных
BMOVI		Прерываемое перемещение блоков данных
Команды сдвига		
NORML	–	Нормализация двойного слова
SHL	SHLB	Логический сдвиг влево
SHR	SHRB	Логический сдвиг вправо
SHLL	–	Логический сдвиг двойного слова влево
SHRL	–	Логический сдвиг двойного слова вправо
SHRA	SHRAB	Арифметический сдвиг вправо
SHRAL	–	Арифметический сдвиг двойного слова вправо

Таблица Б.2 – Команды микроконтроллера для работы с битами, стеком и специальные

Мнемоника команды	Название операции
1	2
Команды управления PTS	
DPTS	Запрет PTS

Окончание таблицы Б.2

1	2
EPTS	Разрешение PTS
Команды работы со стеком	
PUSH	Загрузка слова в стек
PUSHA	Загрузка всего в стек
PUSHF	Загрузка флагов в стек
POP	Чтение слова из стека
POPA	Чтение всего из стека
POPF	Чтение флагов из стека
Команды вызова	
LCALL	Длинный вызов
SCALL	Короткий вызов
RET	Возврат из подпрограммы
TRAP	Программное прерывание (не поддерживается транслятором с языка)
Команды безусловного и условного перехода	
BR	Косвенный переход
LJMP	Длинный вызов
SJMP	Короткий вызов
TIJMP	Косвенный табличный переход
DJNZ	Декремент байта и переход при отсутствии нуля
DJNZW	Декремент слова и переход при отсутствии нуля
JBC	Переход, если бит очищен
JBS	Переход, если бит установлен
JC	Переход, если C = 1
JNC	Переход, если C = 0
JLT	Переход, если N = 1
JGE	Переход, если N = 0
JE	Переход, если Z = 1
JNE	Переход, если Z = 0
JV	Переход, если V = 1
JNV	Переход, если V = 0
JVT	Переход, если VT = 1
JNVT	Переход, если VT = 0
JST	Переход, если ST = 1
JNST	Переход, если ST = 0
JGT	Переход, если N = 0 и Z = 0
JLE	Переход, если N = 1 или Z = 1
JH	Переход, если C = 1 и Z = 0
JNH	Переход, если C = 0 и Z = 1
Специальные команды	
DI	Запрещение обслуживания всех прерываний
EI	Разрешение обслуживания всех прерываний
CLRC	Очистка флага переноса
SETC	Установка флага переноса
CLRVT	Очистка дополнительного флага переполнения
IDLPD	Включение режима Idle или Powerdown
NOP	Нет операции
SKIP	Нет операции (двухбайтная)
RST	Сброс системы

Таблица Б.3 – Система команд микроконтроллера

Мнемо-ника	Операция и варианты использования команды	Опкод	Длина	Время выполнения*
1	2	3	4	5
ADD	**Сложение слов** (DEST, SRC)			Z N C V VT ST √ √ √ √ ↑ –
	ADD wreg, wreg	64h	3 байта	4 мт
	ADD wreg, #data	65h	4 байта	
	ADD wreg, [reg]	66h	3 байта	5/9 мт
	ADD wreg, [reg]+			6/10 мт
	ADD wreg, boffset[reg]	67h	4 байта	6/10 мт
	ADD wreg, woffset[reg]		5 байт	
ADD	**Сложение слов** (DEST, SRC1, SRC2)			Z N C V VT ST √ √ √ √ ↑ –
	ADD wreg, wreg, wreg	44h	4 байта	4 мт
	ADD wreg, wreg, #data,	45h	5 байт	
	ADD wreg, wreg, [reg]	46h	4 байта	5/9 мт
	ADD wreg, wreg, [reg]+			6/10 мт
	ADD wreg, wreg, boffset[reg]	47h	5 байт	6/10 мт
	ADD wreg, wreg, woffset[reg]		6 байт	
ADDB	**Сложение байт** (DEST, SRC)			Z N C V VT ST √ √ √ √ ↑ –
	ADDB breg, breg	74h	3 байта	4 мт
	ADDB breg, #data,	75h		
	ADDB breg, [reg]	76h		5/9 мт
	ADDB breg, [reg]+			6/10 мт
	ADDB breg, boffset[reg]	77h	4 байта	6/10 мт
	ADDB breg, woffset[reg]		5 байт	
ADDB	**Сложение байт** (DEST, SRC1, SRC2)			Z N C V VT ST √ √ √ √ ↑ –
	ADDB breg, breg, breg	54h	4 байта	4 мт
	ADDB breg, breg, #data,	55h		
	ADDB breg, breg, [reg]	56h		5/9 мт
	ADDB breg, breg, [reg]+			
	ADDB breg, breg, boffset[reg]	57h	5 байт	6/10 мт
	ADDB breg, breg, woffset[reg]		6 байт	
ADDC	**Сложение слов с переносом** (DEST, SRC)			Z N C V VT ST ↓ √ √ √ ↑ –
	ADDC wreg, wreg	A4h	3 байта	4 мт
	ADDC wreg, #data	A5h	4 байта	
	ADDC wreg, [reg]	A6h	3 байта	5/9 мт
	ADDC wreg, [reg]+			
	ADDC wreg, boffset[reg]	A7h	4 байта	6/10 мт
	ADDC wreg, woffset[reg]		5 байт	

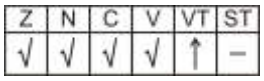
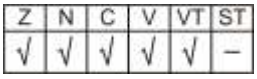
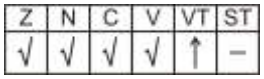
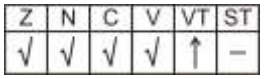
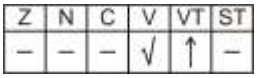
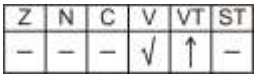
Продолжение таблицы Б.3

1	2	3	4	5
ADDCB	**Сложение байт с переносом** (DEST, SRC)		Z N C V VT ST ↓ √ √ √ ↑ –	
	ADDCB breg, breg	B4h	3 байта	4 мт
	ADDCB breg, #data,	B5h		
	ADDCB breg, [reg]	B6h		5/9 мт
	ADDCB breg, [reg]+			6/10 мт
	ADDCB breg, boffset[reg]	B7h	4 байта	6/10 мт
	ADDCB breg, woffset[reg]		5 байт	
AND	**Логическое «И» слов** (DEST, SRC)		Z N C V VT ST √ √ 0 0 – –	
	AND wreg, wreg	60h	3 байта	4 мт
	AND wreg, #data	61h	4 байта	
	AND wreg, [reg]	62h	3 байта	5/9 мт
	AND wreg, [reg]+			6/10 мт
	AND wreg, boffset[reg]	63h	4 байта	6/10 мт
	AND wreg, woffset[reg]		5 байт	
AND	**Логическое «И» слов** (DEST, SRC1, SRC2)		Z N C V VT ST √ √ 0 0 – –	
	AND wreg, wreg, wreg	40h	4 байта	4 мт
	AND wreg, wreg, #data,	41h	5 байт	
	AND wreg, wreg, [reg]	42h	4 байта	5/9 мт
	AND wreg, wreg, [reg]+			6/10 мт
	AND wreg, wreg, boffset[reg]	43h	5 байт	6/10 мт
	AND wreg, wreg, woffset[reg]		6 байт	
ANDB	**Логическое «И» байт** (DEST, SRC)		Z N C V VT ST √ √ 0 0 – –	
	ANDB breg, breg	70h	3 байта	4 мт
	ANDB breg, #data,	71h		
	ANDB breg, [reg]	72h		5/9 мт
	ANDB breg, [reg]+			6/10 мт
	ANDB breg, boffset[reg]	73h	4 байта	6/10 мт
	ANDB breg, woffset[reg]			
ANDB	**Логическое «И» байт** (DEST, SRC1, SRC2)		Z N C V VT ST √ √ 0 0 – –	
	ANDB breg, breg, breg	50h	4 байта	4 мт
	ANDB breg, breg, #data,	51h		
	ANDB breg, breg, [reg]	52h		5/9 мт
	ANDB breg, breg, [reg]+			6/10 мт
	ANDB breg, breg, boffset[reg]	53h	5 байт	6/10 мт
	ANDB breg, breg, woffset[reg]			

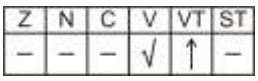
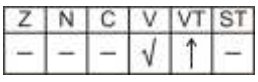
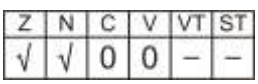
Продолжение таблицы Б.3

1	2	3	4	5
BMOV	**Непрерываемое перемещение блоков данных** (DEST, SRC)			
	BMOV Ireg, woffset[reg]	C1h	3 байта	11мт (Ex/In) 16 мт (Ex/Ex)
BMOVI	**Прерываемое перемещение блоков** (DEST, SRC)			
	BMOVI Ireg, woffset[reg]	CDh	3 байта	11мт (Ex/In) 16 мт (Ex/Ex)
BR	**Косвенный переход** (DEST)			
	BR [reg]	E3h	2 байта	2 мт
CLR	**Очистка слова** (DEST)		Z N C V VT ST 1 0 0 0 – –	
	CLR wreg	01h	2 байта	2 мт
CLRB	**Очистка байта** (DEST)		Z N C V VT ST 1 0 0 0 – –	
	CLRB breg	11h	2 байта	2 мт
CLRC	**Очистка флага переноса**		Z N C V VT ST – – 0 – – –	
	CLRC (прямая адресация)	F8h	1 байт	1 мт
CLRVT	**Очистка дополнительного флага переполнения**		Z N C V VT ST – – – – 0 –	
	CLRVT (прямая адресация)	FCh	1 байт	1 мт
CMP	**Сравнение слов** (DEST, SRC)		Z N C V VT ST √ √ √ √ ↑ –	
	CMP wreg, wreg	88h	3 байта	4 мт
	CMP wreg, #data	89h	4 байта	
	CMP wreg, [reg]	8Ah	3 байта	5/9 мт
	CMP wreg, [reg]+			6/10 мт
	CMP wreg, boffset[reg]	8Bh	4 байта	6/10 мт
	CMP wreg, woffset[reg]		5 байт	

Продолжение таблицы Б.3

1	2	3	4	5
CMPB	**Сравнение байт** (DEST, SRC)			
	CMPB breg, breg	98h	3 байта	4 мт
	CMPB breg, #data,	99h		
	CMPB breg, [reg]	9Ah		5/9 мт
	CMPB breg, [reg]+			6/10 мт
	CMPB breg, boffset[reg]	9Bh	4 байта	6/10 мт
	CMPB breg, woffset[reg]		5 байт	
CMPL	**Сравнение длинных слов** (DEST, SRC)			
	CMPL Ireg, Ireg	C5h	3 байта	6 мт
DEC	**Декремент слова** (DEST)			
	DEC wreg	05h	2 байта	3 мт
DECB	**Декремент байта** (DEST)			
	DEC breg	15h	2 байта	3 мт
DI	**Запрещение обслуживания всех прерываний**			
	DI	FAh	1 байт	1 мт
DIV	**Знаковое деление слов** (DEST, SRC)			
	DIV Ireg, wreg	FE8Ch	4 байта	8 мт
	DIV Ireg, #data	FE8Dh	5 байт	
	DIV Ireg, [reg]	FE8Eh	4 байта	9/13 мт
	DIV Ireg, [reg]+		4 байта	10/14 мт
	DIV Ireg, boffset[reg]	FE8Fh	5 байт	10/14 мт
	DIV Ireg, woffset[reg]		6 байт	
DIVB	**Знаковое деление байт** (DEST, SRC)			
	DIVB wreg, breg	FE9Ch	4 байта	6 мт
	DIVB wreg, #data	FE9Dh		
	DIVB wreg, [reg]	FE9Eh		7/11 мт
	DIVB wreg, [reg]+			8/12 мт
	DIVB wreg, boffset[reg]	FE9Fh	5 байт	8/12 мт
	DIVB wreg, woffset[reg]		6 байт	

Продолжение таблицы Б.3

1	2	3	4	5
DIVU	**Беззнаковое деление слов** (DEST, SRC) 			
	DIVU Ireg, wreg	8Ch	3 байта	8 мт
	DIVU Ireg, #data	8Dh	4 байта	
	DIVU Ireg, [reg]	8Eh	3 байта	9/13 мт
	DIVU Ireg, [reg]+			10/14 мт
	DIVU Ireg, boffset[reg]	8Fh	4 байта	10/14 мт
	DIVU Ireg, woffset[reg]		5 байт	
DIVUB	**Беззнаковое деление байт** (DEST, SRC) 			
	DIVUB wreg, breg	9Ch	3 байта	6 мт
	DIVUB wreg, #data	9Dh		
	DIVUB wreg, [reg]	9Eh		7/11 мт
	DIVUB wreg, [reg]+			8/12 мт
	DIVUB wreg, boffset[reg]	9Fh	4 байта	8/12 мт
	DIVUB wreg, woffset[reg]		5 байт	
DJNZ	**Декремент байта и переход при отсутствии нуля** (COUNT,ADDR)			
	DJNZ breg, cadd (короткоиндексная адресация)	E0h	3 байта	4 мт
DJNZW	**Декремент слова и переход при отсутствии нуля** (COUNT,ADDR)			
	DJNZW wreg, cadd (короткоиндексная адресация)	E1h	3 байта	4 мт
DPTS	**Запрет PTS**			
	DPTS	ECh	1 байт	1 мт
EI	**Разрешение обслуживания всех прерываний**			
	EI	FBh	1 байт	1 мт
EPTS	**Разрешение PTS**			
	EPTS	EDh	1 байт	1 мт
EXT	**Знаковое расширение слова до двойного слова** (DEST) 			
	EXT (прямая адресация)	06h	2 байта	4 мт

Продолжение таблицы Б.3

1	2	3	4	5
EXTB	**Знаковое расширение байта до слова** (DEST)	Z N C V VT ST √ √ 0 0 – –		
	EXTB (прямая адресация)	16h	2 байта	4 мт
IDLPD	**Включение режима Idle или Powerdown**	Если параметр (key) задан неверно, то флаги в PSW сбрасываются Z N C V VT ST 0 0 0 0 0 0		
	IDLPD #1 (включение режима Idle) IDLPD #2 (включение режима Powerdown) IDLPD #key (сброс микроконтроллера)	F6h	2 байт	5 мт
INC	**Инкремент слова** (DEST)	Z N C V VT ST √ √ √ √ ↑ 0		
	INC wreg	07h	2 байта	3 мт
INCB	**Инкремент байта** (DEST)	Z N C V VT ST √ √ √ √ ↑ –		
	INC breg	17h	2 байта	3 мт
JBC	**Переход, если бит очищен**			
	JBC breg, 0, cadd	30h	3 байта	2 мт
	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 (короткоиндексная адресация)	37h		
JBS	**Переход, если бит установлен**			
	JBS breg, 0, cadd	38h	3 байта	2 мт
	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 (короткоиндексная адресация)	3Fh		
JC	**Переход, если C = 1**			
	JC cadd (короткоиндексная адресация)	DBh	2 байта	1 мт

Продолжение таблицы Б.3

1	2	3	4	5
JE	**Переход, если Z = 1**			
	JE cadd (короткоиндексная адресация)	DFh	2 байта	1 мт
JGE	**Переход, если N = 0**			
	JGE cadd (короткоиндексная адресация)	D6h	2 байта	1 мт
JGT	**Переход, если N = 0 и Z = 0**			
	JGT cadd (короткоиндексная адресация)	D2h	2 байта	1 мт
JH	**Переход, если C = 1 и Z = 0**			
	JH cadd (короткоиндексная адресация)	D9h	2 байта	1 мт
JLE	**Переход, если N = 1 или Z = 1**			
	JLE cadd (короткоиндексная адресация)	DAh	2 байта	1 мт
JLT	**Переход, если N = 1**			
	JLT cadd (короткоиндексная адресация)	DEh	2 байта	1 мт
JNC	**Переход, если C = 0**			
	JNC cadd (короткоиндексная адресация)	D3h	2 байта	1 мт
JNE	**Переход, если Z = 0**			
	JNE cadd (короткоиндексная адресация)	D7h	2 байта	1 мт
JNH	**Переход, если C = 0 и Z = 1**			
	JNH cadd (короткоиндексная адресация)	D1h	2 байта	1 мт
JNST	**Переход, если ST = 0**			
	JNST cadd (короткоиндексная адресация)	D0h	2 байта	1 мт
JNV	**Переход, если V = 0**			
	JNV cadd (короткоиндексная адресация)	D5h	2 байта	1 мт
JNVT	**Переход, если VT = 0**			
	JNVT cadd (короткоиндексная адресация)	D4h	2 байта	1 мт

Z	N	C	V	VT	ST
–	–	–	–	0	–

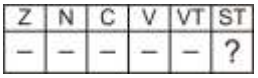
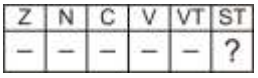
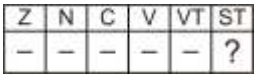
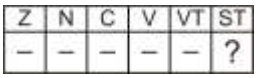
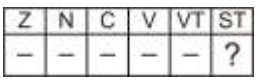
Продолжение таблицы Б.3

1	2	3	4	5
JST	**Переход, если ST = 1**			
	JST cadd (короткоиндексная адресация)	D8h	2 байта	1 мт
JV	**Переход, если V = 1**			
	JV cadd (короткоиндексная адресация)	DDh	2 байта	1 мт
JVT	**Переход, если VT = 1**		Z N C V VT ST – – – – 0 –	
	JVT cadd (короткоиндексная адресация)	DCh	2 байта	1 мт
LCALL	**Длинный вызов**			
	LCALL cadd (длинноиндексная адресация)	EFh	3 байта	3 (7) мт
LD	**Загрузка слова** (DEST, SRC)			
	LD wreg, wreg	A0h	3 байта	2 мт
	LD wreg, #data	A1h	4 байта	1 мт
	LD wreg, [reg]	A2h	3 байта	3/7 мт
	LD wreg, [reg]+			4/8 мт
	LD wreg, boffset[reg]	A3h	4 байта	4/8 мт
	LD wreg, woffset[reg]		5 байт	
LDB	**Загрузка байта** (DEST, SRC)			
	LDB breg, breg	B0h	3 байта	2 мт
	LDB breg, #data	B1h		1 мт
	LDB breg, [reg]	B2h		3/7 мт
	LDB breg, [reg]+			4/8 мт
	LDB breg, boffset[reg]	B3h	4 байта	4/8 мт
	LDB breg, woffset[reg]		5 байт	
LDBSE	**Загрузка байта со знаковым расширением** (DEST, SRC)			
	LDBSE wreg, breg	BCh	3 байта	2 мт
	LDBSE wreg, #data	BDh		1 мт
	LDBSE wreg, [reg]	BEh		3/7 мт
	LDBSE wreg, [reg]+			4/8 мт
	LDBSE wreg, boffset[reg]	BFh	4 байта	4/8 мт
	LDBSE wreg, woffset[reg]		5 байт	

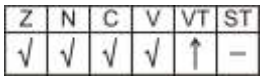
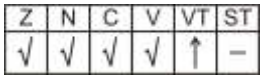
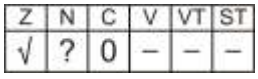
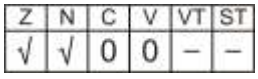
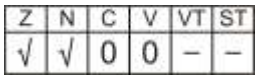
Продолжение таблицы Б.3

1	2	3	4	5
LDBZE	**Загрузка байта с беззнаковым расширением** (DEST, SRC)			
	LDBZE wreg, breg	ACh	3 байта	2 мт
	LDBZE wreg, #data	ADh		1 мт
	LDBZE wreg, [reg]	AEh		3/7 мт
	LDBZE wreg, [reg]+			4/8 мт
	LDBZE wreg, boffset[reg]	AFh	4 байта	4/8 мт
	LDBZE wreg, woffset[reg]		5 байт	
LJMP	**Длинный переход**	Z N C V VT ST – – – – – ?		
	LJMP cadd (длинноиндексная адресация)	E7h	3 байта	1 мт
MUL	**Знаковое умножение слов** (DEST, SRC)	Z N C V VT ST – – – – – ?		
	MUL Ireg, wreg	FE6Ch	4 байта	5 мт
	MUL Ireg, #data	FE6Dh	5 байт	
	MUL Ireg, [reg]	FE6Eh	4 байта	6/10 мт
	MUL Ireg, [reg]+			7/11 мт
	MUL Ireg, boffset[reg]	FE6Fh	5 байт	7/11 мт
	MUL Ireg, woffset[reg]		6 байт	
MUL	**Знаковое умножение слов** (DEST, SRC1, SRC2)	Z N C V VT ST – – – – – ?		
	MUL Ireg, wreg, wreg	FE4Ch	5 байт	5 мт
	MUL Ireg, wreg, #data	FE4Dh	6 байт	
	MUL Ireg, wreg, [reg]	FE4Eh	5 байт	6/10 мт
	MUL Ireg, wreg, [reg]+			7/11 мт
	MUL Ireg, wreg, boffset[reg]	FE4Fh	6 байт	7/11 мт
	MUL Ireg, wreg, woffset[reg]		7 байт	
MULB	**Знаковое умножение байт** (DEST, SRC)	Z N C V VT ST – – – – – ?		
	MULB wreg, breg	FE7Ch	4 байта	4 мт
	MULB wreg, #data	FE7Dh		
	MULB wreg, [reg]	FE7Eh		5/9 мт
	MULB wreg, [reg]+			6/10 мт
	MULB wreg, boffset[reg]	FE7Fh	5 байт	6/10 мт
	MULB wreg, woffset[reg]		6 байт	

Продолжение таблицы Б.3

1	2	3	4	5
MULB	**Знаковое умножение байт** (DEST, SRC1, SRC2) 			
	MULB wreg, breg, breg	FE5Ch	5 байт	4 мт
	MULB wreg, breg, #data	FE5Dh		
	MULB wreg, breg, [reg]	FE5Eh		5/9 мт
	MULB wreg, breg, [reg]+			6/10 мт
	MULB wreg, breg, boffset[reg]	FE5Fh	6 байт	6/10 мт
	MULB wreg, breg, woffset[reg]		7 байт	
MULU	**Беззнаковое умножение слов** (DEST, SRC) 			
	MULU Ireg, wreg	6Ch	3 байта	5 мт
	MULU Ireg, #data	6Dh	4 байта	
	MULU Ireg, [reg]	6Eh	3 байта	6/10 мт
	MULU Ireg, [reg]+			7/11 мт
	MULU Ireg, boffset[reg]	6Fh	4 байта	7/11 мт
	MULU Ireg, woffset[reg]		5 байт	
MULU	**Беззнаковое умножение слов** (DEST, SRC1, SRC2) 			
	MULU Ireg, wreg, breg	4Ch	4 байта	5 мт
	MULU Ireg, wreg, #data	4Dh	5 байт	
	MULU Ireg, wreg, [reg]	4Eh	4 байта	6/10 мт
	MULU Ireg, wreg, [reg]+			7/11 мт
	MULU Ireg, wreg, boffset[reg]	4Fh	5 байт	7/11 мт
	MULU Ireg, wreg, woffset[reg]		6 байт	
MULUB	**Беззнаковое умножение байт** (DEST, SRC) 			
	MULUB wreg, breg	7Ch	3 байта	4 мт
	MULUB wreg, #data	7Dh		
	MULUB wreg, [reg]	7Eh		5/9 мт
	MULUB wreg, [reg]+			6/10 мт
	MULUB wreg, boffset[reg]	7Fh	4 байта	6/10 мт
	MULUB wreg, woffset[reg]		5 байт	
MULUB	**Беззнаковое умножение байт** (DEST, SRC1, SRC2) 			
	MUL wreg, breg, breg	5Ch	4 байта	4 мт
	MUL wreg, breg, #data	5Dh		
	MUL wreg, breg, [reg]	5Eh		5/9 мт
	MUL wreg, breg, [reg]+			6/10 мт
	MUL wreg, breg, boffset[reg]	5Fh	5 байт	6/10 мт
	MUL wreg, breg, woffset[reg]		6 байт	

Продолжение таблицы Б.3

1	2	3	4	5
NEG	**Изменение знака слова**			
	NEG wreg	03h	2 байта	3 мт
NEGB	**Изменение знака байта**			
	NEGB breg	13h	2 байта	3 мт
NOP	**Нет операции**			
	NOP	FDh	1 байт	1 мт
NORML	**Нормализация двойного слова** (SRC, DEST)			
	NORML Ireg, breg	0Fh	3 байта	6 мт
NOT	**Инверсия слова**			
	NOT wreg	02h	2 байта	3 мт
NOTB	**Инверсия байта**			
	NOTB breg	12h	2 байта	3 мт
OR	**Логическое «ИЛИ» слов** (DEST, SRC)			
	OR wreg, wreg	80h	3 байта	4 мт
	OR wreg, #data	81h	4 байта	
	OR wreg, [reg]	82h	3 байта	5/9 мт
	OR wreg, [reg]+			6/10 мт
	OR wreg, boffset[reg]	83h	4 байта	6/10 мт
	OR wreg, woffset[reg]		5 байт	
ORB	**Логическое «ИЛИ» байт** (DEST, SRC)			
	ORB breg, breg	90h	3 байта	4 мт
	ORB breg, #data	91h		
	ORB breg, [reg]	92h		5/9 мт
	ORB breg, [reg]+			6/10 мт
	ORB breg, boffset[reg]	93h	4 байта	6/10 мт
	ORB breg, woffset[reg]		5 байт	

Продолжение таблицы Б.3

1	2	3	4	5
POP	**Чтение слова из стека**			
	POP wreg	CCh	2 байта	3 (7) мт
	POP [reg]	CEh	2 байта	8/13 (4/8) мт
	POP [reg]+			
	POP boffset[reg]	CFh	3 байта	
	POP woffset[reg]		4 байта	
POPA	**Чтение всего из стека**	Z N C V VT ST √ √ √ √ √ √		
	POPA	F5h	1 байт	3 (7) мт
POPF	**Чтение флагов из стека**	Z N C V VT ST √ √ √ √ √ √		
	POPF	F3h	1 байт	1 (5) мт
PUSH	**Загрузка слова в стек**			
	PUSH wreg	C8h	2 байта	2 (6) мт
	PUSH #data	C9h	3 байта	
	PUSH [reg]	CAh	2 байта	3/7 (7/12) мт
	PUSH [reg]+			4/8 (8/13) мт
	PUSH boffset[reg]	CBh	3 байта	4/8 (8/13) мт
	PUSH woffset[reg]		4 байта	
PUSHA	**Загрузка всего в стек**	Z N C V VT ST 0 0 0 0 0 0		
	PUSHA	F4h	1 байт	4 (8) мт
PUSHF	**Загрузка флагов в стек**	Z N C V VT ST 0 0 0 0 0 0		
	PUSHF	F2h	1 байт	2 (6) мт
RET	**Возврат из подпрограммы**			
	RET	F0h	1 байт	2 (4) мт
RST	**Сброс системы**			
	RST	FFh	1 байт	40 мт (включая выборку байта ССВ)

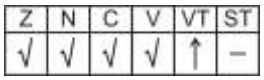
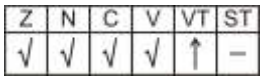
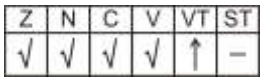
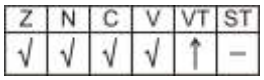
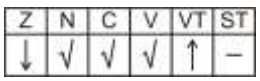
Продолжение таблицы Б.3

1	2	3	4	5
SCALL	**Короткий вызов**			
	SCALL cadd (короткоиндексная адресация)	28h – 2Fh	2 байта	3 (7) мт
SETC	**Установка флага переноса**	Z N C V VT ST – – 1 – – –		
	SETC	F9h	1 байт	1 мт
SHL	**Логический сдвиг слова влево**	Z N C V VT ST √ ? √ √ ↑ –		
	SHL wreg, #count SHL wreg, breg	09h	3 байта	4 мт (3 мт, если сдвига нет)
SHLB	**Логический сдвиг байта влево**	Z N C V VT ST √ ? √ √ ↑ –		
	SHLB breg, #count SHLB breg, breg	19h	3 байта	4 мт (3 мт, если сдвига нет)
SHLL	**Логический сдвиг двойного слова влево**	Z N C V VT ST √ ? √ √ ↑ –		
	SHLL lreg, #count SHLL lreg, breg	0Dh	3 байта	4 мт (3 мт, если сдвига нет)
SHR	**Логический сдвиг слова вправо**	Z N C V VT ST √ 0 √ 0 – √		
	SHR wreg, #count SHR wreg, breg	08h	3 байта	4 мт (3 мт, если сдвига нет)
SHRA	**Арифметический сдвиг слова вправо**	Z N C V VT ST √ √ √ 0 – √		
	SHRA wreg, #count SHRA wreg, breg	0Ah	3 байта	4 мт (3 мт, если сдвига нет)
SHRAB	**Арифметический сдвиг байта вправо**	Z N C V VT ST √ √ √ 0 – √		
	SHRAB breg, #count SHRAB breg, breg	1Ah	3 байта	4 мт (3 мт, если сдвига нет)

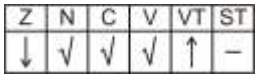
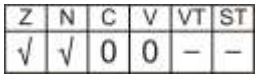
Продолжение таблицы Б.3

1	2	3	4	5
SHRAL	**Арифметический сдвиг двойного слова вправо**	Z N C V VT ST √ √ √ 0 – √		
	SHRAL Ireg, #count SHRAL Ireg, breg	0Eh	3 байта	6 мт (5 мт, если сдвига нет)
SHRB	**Логический сдвиг байта вправо**	Z N C V VT ST √ 0 √ 0 – √		
	SHRB breg, #count SHRB breg, breg	18h	3 байта	4 мт (3 мт, если сдвига нет)
SHRL	**Логический сдвиг двойного слова вправо**	Z N C V VT ST √ 0 √ 0 – √		
	SHRL Ireg, #count SHRL Ireg, breg	0Ch	3 байта	6 мт (5 мт, если сдвига нет)
SJMP	**Короткий переход**			
	SJMP cadd (короткоиндексная адресация)	20h – 27h	2 байта	1 мт
SKIP	**Нет операции (двухбайтная)**			
	SKIP	00h	2 байта	1 мт
ST	**Сохранение слова** (SRC, DEST)			
	ST wreg, wreg	C0h	3 байта	2 мт
	ST wreg, [reg]	C2h		3/7 мт
	ST wreg, [reg]+			4/8 мт
	ST wreg, boffset[reg]	C3h	4 байта	4/8 мт
	ST wreg, woffset[reg]		5 байт	
STB	**Сохранение байта** (SRC, DEST)			
	STB breg, breg	C4h	3 байта	2 мт
	STB breg, [reg]	C6h		3/7 мт
	STB breg, [reg]+			4/8 мт
	STB breg, boffset[reg]	C7h	4 байта	4/8 мт
	STB breg, woffset[reg]		5 байт	

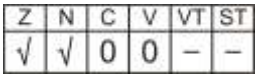
Продолжение таблицы Б.3

1	2	3	4	5
SUB	**Вычитание слова** (DEST, SRC)			
	SUB wreg, wreg	68h	3 байта	4 мт
	SUB wreg, #data	69h	4 байта	
	SUB wreg, [reg]	6Ah	3 байта	5/9 мт
	SUB wreg, [reg]+			6/10 мт
	SUB wreg, boffset[reg]	6Bh	4 байта	6/10 мт
	SUB wreg, woffset[reg]		5 байт	
SUB	**Вычитание слов** (DEST, SRC1, SRC2)			
	SUB wreg, wreg, wreg	48h	4 байта	4 мт
	SUB wreg, wreg, #data	49h	5 байт	
	SUB wreg, wreg, [reg]	4Ah	4 байта	5/9 мт
	SUB wreg, wreg, [reg]+			6/10 мт
	SUB wreg, wreg, boffset[reg]	4Bh	5 байт	6/10 мт
	SUB wreg, wreg, woffset[reg]		6 байт	
SUBB	**Вычитание байт** (DEST, SRC)			
	SUBB breg, breg	78h	3 байта	4 мт
	SUBB breg, #data	79h		
	SUBB breg, [reg]	7Ah		5/9 мт
	SUBB breg, [reg]+			6/10 мт
	SUBB breg, boffset[reg]	7Bh	4 байта	6/10 мт
	SUBB breg, woffset[reg]		5 байт	
SUBB	**Вычитание байт** (DEST, SRC1, SRC2)			
	SUBB breg, breg, breg	58h	4 байта	4 мт
	SUBB breg, breg, #data	59h		
	SUBB breg, breg, [reg]	5Ah		5/9 мт
	SUBB breg, breg, [reg]+			6/10 мт
	SUBB breg, breg, boffset[reg]	5Bh	5 байт	6/10 мт
	SUBB breg, breg, woffset[reg]		6 байт	
SUBC	**Вычитание слов с переносом** (DEST, SRC)			
	SUBC wreg, wreg	A8h	3 байта	4 мт
	SUBC wreg, #data	A9h	4 байта	
	SUBC wreg, [reg]	AAh	3 байта	5/9 мт
	SUBC wreg, [reg]+			6/10 мт
	SUBC wreg, boffset[reg]	ABh	4 байта	6/10 мт
	SUBC wreg, woffset[reg]		5 байт	

Продолжение таблицы Б.3

1	2	3	4	5
SUBCB	**Вычитание байт с переносом** (DEST, SRC)			
	SUBCB breg, breg	B8h	4 байта	4 мт
	SUBCB breg, #data	B9h	3 байта	
	SUBCB breg, [reg]	BAh	3 байта	5/9 мт
	SUBCB breg, [reg]+			6/10 мт
	SUBCB breg, boffset[reg]	BBh	4 байта	6/10 мт
	SUBCB breg, woffset[reg]		5 байт	
TIJMP	**Косвенный табличный переход**			
	TIJMP wreg, [wreg], #mask	E2h	4 байта	9 мт (Ex/In) 14 мт (Ex/Ex)
TRAP	**Программное прерывание**			
	TRAP	F7h	1 байт	1 (5) мт
XCH	**Обмен слов** (DEST, SRC)			
	XCH wreg, wreg	04h	3 байта	4 мт
	XCH wreg, boffset[reg]	0Bh	4 байта	6/10 мт
	XCH wreg, woffset[reg]		5 байт	
XCHB	**Обмен байт** (DEST, SRC)			
	XCHB breg, breg	14h	3 байта	4 мт
	XCHB breg, boffset[reg]	1Bh	4 байта	6/10 мт
	XCHB breg, woffset[reg]		5 байт	
XOR	**Логическое исключающее «ИЛИ» слов** (DEST, SRC)			
	XOR wreg, wreg	84h	3 байта	4 мт
	XOR wreg, #data	85h	4 байта	
	XOR wreg, [reg]	86h	3 байта	5/9 мт
	XOR wreg, [reg]+			6/10 мт
	XOR wreg, boffset[reg]	87h	4 байта	6/10 мт
	XOR wreg, woffset[reg]		5 байт	

Окончание таблицы Б.3

1	2	3	4	5
XORB	**Логическое исключающее «ИЛИ» байт** (DEST, SRC)  			
	XORB breg, breg	94h	3 байта	4 мт
	XORB breg, #data	95h		
	XORB breg, [reg]	96h		5/9 мт
	XORB breg, [reg]+			6/10 мт
	XORB breg, boffset[reg]	97h	4 байта	6/10 мт
	XORB breg, woffset[reg]		5 байт	

* Время выполнения команды указано в количестве машинных тактов (мт). Время выполнения команды с использованием косвенной и индексной адресации указано двумя цифрами, написанными через слеш. Первая цифра показывает время выполнения команды с использованием регистров специального назначения или регистров внутреннего ОЗУ с адресами 000h – 7FFh. Вторая цифра показывает время выполнения команды при работе с внешней памятью. Это время включает в себя время ожидания освобождения контроллера доступа к памяти команд (два машинных такта) и время одной выборки команды, так как приоритет отдается именно им (самая быстрая выборка – еще два машинных такта). Для некоторых команд (LCALL, POP, PUSH, RET и других, использующих стек) указано время выполнения при работе с внутренним стеком, а рядом в скобках – с внешним стеком.

Дополнительная информация о командах микроконтроллера

ADD Dwreg, waop
Dwreg, wreg, waop

Функция: **Сложение слов**

Код: (011001aa) (waop) (Dwreg)
(010001aa) (waop) (wreg) (Dwreg)

Описание: Сложение слова источника SRC и слова приемника DEST и сохранение суммы в приемнике.
Сложение слова первого источника SRC1 и слова второго источника SRC2 и сохранение суммы в приемнике DEST.

Алгоритм: (DEST) ← (DEST) + (SRC)
(DEST) ← (SRC1) + (SRC2)

ADDB Dbreg, baop
Dbreg, breg, baop

Функция: **Сложение байт**

Код: (011101aa) (baop) (Dbreg)
(010101aa) (baop) (breg) (Dbreg)

Описание: Сложение байта источника SRC и байта приемника DEST и сохранение суммы в приемнике.
Сложение байта первого источника SRC1 и байта второго источника SRC2 и сохранение суммы в приемнике DEST.

Алгоритм: (DEST) ← (DEST) + (SRC)
(DEST) ← (SRC1) + (SRC2)

ADDC Dwreg, waop

Функция: **Сложение слов с переносом**

Код: (101001aa) (waop) (Dwreg)

Описание: Сложение слова источника SRC, слова приемника DEST и флага переноса C и сохранение суммы в приемнике.

Алгоритм: (DEST) ← (DEST) + (SRC) + (C)

ADDCB Dbreg, baop

Функция: **Сложение слов с переносом**

Код: (101101aa) (baop) (Dbreg)

Описание: Сложение байта источника SRC, байта приемника DEST и флага переноса C и сохранение суммы в приемнике.

Алгоритм: (DEST) ← (DEST) + (SRC) + (C)

AND Dwreg, waop
Dwreg, wreg, waop

Функция: **Логическое «И» слов**

Код: (011000aa) (waop) (Dwreg)
(010000aa) (waop) (wreg) (Dwreg)

Описание: Логическое умножение слова источника SRC и слова приемника DEST и сохранение результата в приемнике.
Логическое умножение слова первого источника SRC1 и слова второго источника SRC2 и сохранение результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) AND (SRC)
(DEST) ← (SRC1) AND (SRC2)

ANDB Dbreg, baop
Dbreg, breg, baop

Функция: **Логическое «И» байт**

Код: (011100aa) (baop) (Dbreg)
(010100aa) (baop) (breg) (Dbreg)

Описание: Логическое умножение байта источника SRC и байта приемника DEST и сохранение результата в приемнике.
Логическое умножение байта первого источника SRC1 и байта второго источника SRC2 и сохранение результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) AND (SRC)
(DEST) ← (SRC1) AND (SRC2)

BMOV DIreg, wreg

Функция: **Непрерываемое перемещение блоков данных**

Код: (11000001) (wreg) (DIreg)

Описание: Перемещение блоков слов данных из одной зоны памяти в другую. Важно помнить, что никакое прерывание не будет обслужено до окончания выполнения команды BMOV, поэтому при пересылке больших массивов данных лучше использовать команду BMOVI. Адреса источника и приемника вычисляются при помощи режима косвенной адресации с автоинкрементом. Длинный регистр PTRS содержит указатели источника SRCPTR и приемника DSTPTR, которые хранятся в соседних регистрах слов. Блоки данных могут находиться в любом месте памяти, но не должны перекрываться. Регистр числа передач CNTREG определяет количество перемещений. Регистр CNTREG не декрементируется в процессе выполнения команды BMOV.

Алгоритм:

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

BMOVI DIreg, wreg

Функция: **Прерываемое перемещение блоков данных**

Код: (11001101) (wreg) (DIreg)

Описание: Перемещение блоков слов данных из одной зоны памяти в другую. Команда идентична команде BMOV, за исключением того, что она является прерываемой. Регистр числа передач CNTREG определяет количество перемещений. Регистр CNTREG изменяет свое значение только в том случае, если выполнение команды BMOVI прерывается. После обслуживания прерывания регистр CNTREG содержит число оставшихся пересылок, поэтому перед повторным вызовом команды BMOVI, этот регистр должен быть проинициализирован.

Алгоритм: Идентичен алгоритму команды BMOV

BR [Dwreg]

Функция: **Косвенный переход**

Код: (11100011) (Dwreg)

Описание: Загрузка в программный счетчик PC адреса, косвенно задаваемого операндом и продолжение выполнения программы с этого адреса.

Алгоритм: PC ← (DEST)

CLR Dwreg

Функция: **Очистка слова**

Код: (00000001) (Dwreg)

Описание: Запись нулей в приемник DEST.

Алгоритм: (DEST) ← 0000h

CLRB Dbreg

Функция: **Очистка байта**

Код: (00010001) (Dbreg)

Описание: Запись нулей в младший байт приемника DEST.

Алгоритм: (DEST (младший байт)) ← 00h

CLRC

Функция: **Очистка флага переноса**

Код: (11111000)

Описание: Запись нуля в бит C регистра PSW.

Алгоритм: C ← 0b

CLRVT

Функция: **Очистка дополнительного флага переполнения**

Код: (11111100)

Описание: Запись нуля в бит VT регистра PSW.

Алгоритм: VT ← 0b

CMP Dwreg, waop

Функция: **Сравнение слов**

Код: (100010aa) (waop) (Dwreg)

Описание: Вычитание слова источника SRC из слова приемника DEST. Флаги в PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается.

Алгоритм: (DEST) – (SRC)

CMPB Dbreg, baop

Функция: **Сравнение байт**

Код: (100110aa) (baop) (Dbreg)

Описание: Вычитание байта источника SRC из байта приемника DEST. Флаги в PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается.

Алгоритм: (DEST) – (SRC)

CMPL DIreg, Ireg

Функция: **Сравнение длинных слов**

Код: (11000101) (Ireg) (DIreg)

Описание: Вычитание двойного слова источника SRC из двойного слова приемника DEST. Операнды определяются с использованием режима прямой адресации. Флаги в PSW устанавливаются, но значения операндов остаются прежними. Если возникает заем, то флаг переноса сбрасывается, в противном случае устанавливается.

Алгоритм: (DEST) – (SRC)

DEC Dwreg

Функция: **Декремент слова**

Код: (00000101) (Dwreg)

Описание: Уменьшение величины операнда на единицу.

Алгоритм: (DEST) ← (DEST) – 1

DECB Dbreg

Функция: **Декремент байта**

Код: (00010101) (Dbreg)

Описание: Уменьшение величины операнда на единицу.

Алгоритм: (DEST) ← (DEST) – 1

DI

Функция: **Запрещение обслуживания всех прерываний**

Код: (11111010)

Описание: Запрещается обслуживание всех прерываний. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: Бит I регистра PSW ← 0

DIV DIreg, waop

Функция: **Знаковое деление слов**

Код: (11111110) (100011aa) (waop) (DIreg)

Описание: Деление целого двойного слова приемника DEST на целое слово источника SRC, с использованием знаковой арифметики. Частное сохраняется в младшем слове (т. е. в слове с меньшим адресом) приемника, а остаток – в старшем слове.

Алгоритм: (DEST (младшее слово)) ← (DEST)/(SRC)
(DEST (старшее слово))← Остаток от (DEST)/(SRC)

DIVB Dwreg, baop

Функция: **Знаковое деление байт**

Код: (11111110) (100111aa) (baop) (Dwreg)

Описание: Деление целого слова приемника DEST на целый байт источника SRC, с использованием знаковой арифметики. Частное сохраняется в младшем байте (т. е. в байте с меньшим адресом) приемника, а остаток – в старшем байте.

Алгоритм: (DEST (младший байт)) ← (DEST)/(SRC)
(DEST (старший байт))← Остаток от (DEST)/(SRC)

DIVU DIreg, waop

Функция: **Беззнаковое деление слов**

Код: (100011aa) (waop) (DIreg)

Описание: Деление целого двойного слова приемника DEST на целое слово источника SRC, с использованием беззнаковой арифметики. Частное сохраняется в младшем слове (т. е. в слове с меньшим адресом) приемника, а остаток – в старшем слове.

Алгоритм: (DEST (младшее слово)) ← (DEST)/(SRC)
(DEST (старшее слово))← Остаток от (DEST)/(SRC)
Обе операции выполняются одновременно

DIVUB Dwreg, baop

Функция: **Беззнаковое деление байт**

Код: (100111aa) (baop) (Dwreg)

Описание: Деление целого слова приемника DEST на целый байт источника SRC, с использованием беззнаковой арифметики. Частное сохраняется в младшем байте (т. е. в байте с меньшим адресом) приемника, а остаток – в старшем байте.

Алгоритм: (DEST (младший байт)) ← (DEST)/(SRC)
(DEST (старший байт))← Остаток от (DEST)/(SRC)
Обе операции выполняются одновременно

DJNZ Dbreg, cadd

Функция: **Декремент байта и переход при отсутствии нуля**

Код: (11100000) (Dbreg) (disp)

Описание: 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.

Алгоритм: (COUNT) ← (COUNT) – 1
Если (COUNT) ≠ 0, тогда PC ← PC + disp

DJNZW Dwreg, cadd

Функция: **Декремент слова и переход при отсутствии нуля**

Код: (11100001) (Dwreg) (disp)

Описание: 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.

Алгоритм: (COUNT) ← (COUNT) – 1
Если (COUNT) ≠ 0, тогда PC ← PC + disp

DPTS

Функция: **Запрет PTS**

Код: (11101100)

Описание: Блокирование сервера периферийных транзакций PTS.

Алгоритм: Бит PSE регистра PSW ← 0

EI

Функция: **Разрешение обслуживания всех прерываний**

Код: (11111011)

Описание: Разрешается обслуживание всех прерываний. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: Бит I регистра PSW ← 1

EPTS

Функция: **Разрешение PTS**

Код: (11101101)

Описание: Деблокирование сервера периферийных транзакций PTS.

Алгоритм: Бит PSE регистра PSW ← 1

EXT DIreg

Функция: **Знаковое расширение слова до двойного слова**

Код: (00000110) (DIreg)

Описание: Знаковое расширение младшего слова операнда до двойного слова.

Алгоритм: Если (DEST (младшее слово)) < 8000h, тогда
(DEST (старшее слово)) ← 0000h,
иначе
(DEST (старшее слово)) ← FFFFh

EXTB Dwreg

Функция: **Знаковое расширение байта до слова**

Код: (00010110) (Dwreg)

Описание: Знаковое расширение младшего байта операнда до слова

Алгоритм: Если (DEST (младший байт)) < 80h, тогда
(DEST (старший байт)) ← 00h,
иначе
(DEST (старший байт)) ← FFh

IDLPD #key

Функция: **Включение режима Idle или Powerdown**

Код: (11110110) (key)

Описание: Результат выполнения этой команды зависит от значения 8-битной величины операнда (key). Контроллер шины завершает цикл упреждающей выборки перед остановкой ЦПУ или сбросом. Следует помнить, что при неправильном задании величины операнда key, все флаги регистра PSW будут сброшены.

Алгоритм: Если (key) = 1, тогда вход в режим Idle,
иначе
если (key) = 2, тогда вход в режим Powerdown,
иначе
сброс микроконтроллера

INC Dwreg

Функция: **Инкремент слова**

Код: (00000111) (Dwreg)

Описание: Увеличение значения слова операнда на единицу.

Алгоритм: (DEST) ← (DEST) + 1

INCB Dbreg

Функция: **Инкремент байта**

Код: (00010111) (Dbreg)

Описание: Увеличение значения байта операнда на единицу.

Алгоритм: (DEST) ← (DEST) + 1

JBC Dbreg, bitno, cadd

Функция: **Переход, если бит очищен**

Код: (00110bbb) (Dbreg) (disp)

Описание: Тестирование бита с номером, задаваемым операндом (bitno) в указанном байте (Dbreg) и, в случае если бит очищен, переход на указанный адрес (метку). Если бит установлен, 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. Три младших бита (bbb) опкода команды – номер бита.

Алгоритм: Если бит = 0b, тогда PC ← PC + disp

JBS Dbreg, bitno, cadd

Функция: **Переход, если бит установлен**

Код: (00111bbb) (Dbreg) (disp)

Описание: Тестирование бита с номером, задаваемым операндом (bitno) в указанном байте (Dbreg) и, в случае если бит установлен, переход на указанный адрес (метку). Если бит очищен, 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. Три младших бита (bbb) опкода команды – номер бита.

Алгоритм: Если бит = 1b, тогда PC ← PC + disp

JC cadd

Функция: **Переход, если C = 1**

Код: (11011011) (disp)

Описание: Тестирование флага переноса C в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг очищен, 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.

Алгоритм: Если C = 1b, тогда PC ← PC + disp

JE cadd

Функция: **Переход, если Z = 1**
(Переход, если равно)

Код: (11011111) (disp)

Описание: Тестирование флага нуля Z в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг очищен, 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.

Алгоритм: Если Z = 1b, тогда PC ← PC + disp

JGE cadd

Функция: **Переход, если N = 0**
(Переход, если число со знаком больше или равно)

Код: (11010110) (disp)

Описание: Тестирование флага отрицательного результата N в регистре PSW и, в случае если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если N = 0b, тогда PC ← PC + disp

JGT cadd

Функция: **Переход, если N = 0 и Z = 0**
(Переход, если число со знаком больше)

Код: (11010010) (disp)

Описание: Тестирование флагов отрицательного результата N и нуля Z в регистре PSW и, в случае если оба флага сброшены, переход на указанный адрес (метку). Если хотя бы один из двух флагов установлен, 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.

Алгоритм: Если N = 0b и Z = 0b, тогда PC ← PC + disp

JH cadd

Функция: **Переход, если C = 1 и Z = 0**
(Переход, если беззнаковое число больше)

Код: (11011001) (disp)

Описание: Тестирование флагов переноса C и нуля Z в регистре PSW и, в случае если флаг C установлен, а флаг Z сброшен, переход на указанный адрес (метку). В противном случае 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.

Алгоритм: Если C = 1b и Z = 0b, тогда PC ← PC + disp

JLE cadd

Функция: **Переход, если N = 1 и Z = 1**
(Переход, если число со знаком меньше или равно)

Код: (11011010) (disp)

Описание: Тестирование флагов отрицательного результата N и нуля Z в регистре PSW и, в случае если оба флага установлены, переход на указанный адрес (метку). Если хотя бы один из двух флагов сброшен, 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.

Алгоритм: Если N = 1b и Z = 1b, тогда PC ← PC + disp

JLT cadd

Функция: **Переход, если N = 1**
(Переход, если число со знаком меньше)

Код: (11011110) (disp)

Описание: Тестирование флага отрицательного результата N в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг очищен, 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.

Алгоритм: Если N = 1b, тогда PC ← PC + disp

JNC cadd

Функция: **Переход, если C = 0**

Код: (11010011) (disp)

Описание: Тестирование флага переноса C в регистре PSW и, в случае, если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если C = 0b, тогда PC ← PC + disp

JNE cadd

Функция: **Переход, если Z = 0**
(Переход, если не равно)

Код: (11010111) (disp)

Описание: Тестирование флага нуля Z в регистре PSW и, в случае если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если Z = 0b, тогда PC ← PC + disp

JNH cadd

Функция: **Переход, если C = 0 и Z = 1**
(Переход, если беззнаковое число не больше)

Код: (11010001) (disp)

Описание: Тестирование флагов переноса C и нуля Z в регистре PSW и, в случае если флаг C сброшен, а флаг Z установлен, переход на указанный адрес (метку). В противном случае 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.

Алгоритм: Если C = 0b и Z = 1b, тогда PC ← PC + disp

JNST cadd

Функция: **Переход, если ST = 0**

Код: (11010000) (disp)

Описание: Тестирование дополнительного флага переноса ST в регистре PSW и, в случае если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если ST = 0b, тогда PC ← PC + disp

JNV cadd

Функция: **Переход, если V = 0**

Код: (11010101) (disp)

Описание: Тестирование флага переполнения V в регистре PSW и, в случае если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если V = 0b, тогда PC ← PC + disp

JNVT cadd

Функция: **Переход, если VT = 0**

Код: (11010100) (disp)

Описание: Тестирование дополнительного флага переполнения VT в регистре PSW и, в случае если флаг сброшен, переход на указанный адрес (метку). Если флаг установлен, 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.

Алгоритм: Если VT = 0b, тогда PC ← PC + disp

JST cadd

Функция: **Переход, если ST = 1**

Код: (11011000) (disp)

Описание: Тестирование дополнительного флага переноса ST в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг сброшен, 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.

Алгоритм: Если ST = 1b, тогда PC ← PC + disp

JV cadd

Функция: **Переход, если V = 1**

Код: (11011101) (disp)

Описание: Тестирование флага переполнения V в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг сброшен, 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.

Алгоритм: Если V = 1b, тогда PC ← PC + disp

JVT cadd

Функция: **Переход, если VT = 1**

Код: (11011100) (disp)

Описание: Тестирование дополнительного флага переполнения VT в регистре PSW и, в случае если флаг установлен, переход на указанный адрес (метку). Если флаг сброшен, 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.

Алгоритм: Если VT = 1b, тогда PC ← PC + disp

LCALL cadd

Функция: **Длинный вызов**

Код: (11101111) (disp-low) (disp-high)

Описание: 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 двумя байтами ((disp-low) (disp-high)), равное промежутку от конца команды до метки перехода. Операнд может иметь любой адрес во всем адресном пространстве. Подпрограмма должна заканчиваться командой RET.

Алгоритм: SP ← SP – 2
(SP) ← PC
PC ← PC + ((disp-high) (disp-low))

LD Dwreg, waop

Функция: **Загрузка слова**

Код: (101000aa) (waop) (Dwreg)

Описание: Загрузка значения слова источника SRC в слово приемника DEST.

Алгоритм: (DEST) ← (SRC)

LDB Dbreg, baop

Функция: **Загрузка байта**

Код: (101100aa) (baop) (Dbreg)

Описание: Загрузка значения байта источника SRC в байт приемника DEST.

Алгоритм: (DEST) ← (SRC)

LDBSE Dwreg, baop

Функция: **Загрузка байта со знаковым расширением**

Код: (101111aa) (baop) (Dwreg)

Описание: Знаковое расширение байта источника SRC и загрузка его в слово приемника DEST.

Алгоритм: (DEST (младший байт)) ← (SRC)
Если (SRC) < 80h, тогда
(DEST (старший байт)) ← 00h,
иначе
(DEST (старший байт)) ← FFh

LDBZE Dwreg, baop

Функция: **Загрузка байта с беззнаковым расширением**

Код: (101011aa) (baop) (Dwreg)

Описание: Беззнаковое расширение байта источника SRC и загрузка его в слово приемника DEST.

Алгоритм: (DEST (младший байт)) ← (SRC)
(DEST (старший байт)) ← 00h

LJMP cadd

Функция: **Длинный переход**

Код: (11100111) (disp-low) (disp-high)

Описание: Переход по указанному адресу (метке). Для вычисления перехода к текущему значению программного счетчика PC добавляется смещение, задаваемое двумя байтами ((disp-low) (disp-high)), равное промежутку от конца команды до метки перехода. Операнд может иметь любой адрес во всем адресном пространстве.

Алгоритм: PC ← PC + ((disp-high)(disp-low))

MUL DIreg, waop
DIreg, wreg, waop

Функция: **Знаковое умножение слов**

Код: (11111110) (010111aa) (waop) (DIreg)
(11111110) (010011aa) (waop) (wreg) (DIreg)

Описание: Перемножение двойного слова приемника DEST и слова источника SRC с использованием знаковой арифметики и размещение двойного слова результата в приемнике DEST.
Перемножение слова первого источника SRC1 и слова второго источника SRC2 с использованием знаковой арифметики и размещение двойного слова результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) × (SRC)
(DEST) ← (SRC1) × (SRC2)

MULB Dwreg, baop
Dwreg, breg, baop

Функция: **Знаковое умножение байт**

Код: (11111110) (011111aa) (baop) (Dwreg)
(11111110) (010111aa) (baop) (breg) (Dwreg)

Описание: Перемножение слова приемника DEST и байта источника SRC с использованием знаковой арифметики и размещение слова результата в приемнике DEST.
Перемножение байта первого источника SRC1 и байта второго источника SRC2 с использованием знаковой арифметики и размещение слова результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) × (SRC)
(DEST) ← (SRC1) × (SRC2)

MULU DIreg, waop
DIreg, wreg, waop

Функция: **Беззнаковое умножение слов**

Код: (011011aa) (waop) (DIreg)
(010011aa) (waop) (wreg) (DIreg)

Описание: Перемножение двойного слова приемника DEST и слова источника SRC с использованием беззнаковой арифметики и размещение двойного слова результата в приемнике DEST.
Перемножение слова первого источника SRC1 и слова второго источника SRC2 с использованием беззнаковой арифметики и размещение двойного слова результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) × (SRC)
(DEST) ← (SRC1) × (SRC2)

MULUB Dwreg, baop
Dwreg, breg, baop

Функция: **Беззнаковое умножение байт**

Код: (011111aa) (baop) (Dwreg)
(010111aa) (baop) (breg) (Dwreg)

Описание: Перемножение слова приемника DEST и байта источника SRC с использованием беззнаковой арифметики и размещение слова результата в приемнике DEST.
Перемножение байта первого источника SRC1 и байта второго источника SRC2 с использованием беззнаковой арифметики и размещение слова результата в приемнике DEST.

Алгоритм: (DEST) ← (DEST) × (SRC)
(DEST) ← (SRC1) × (SRC2)

NEG Dwreg

Функция: **Изменение знака слова**

Код: (00000011) (Dwreg)

Описание: Изменение знака слова операнда на противоположный.

Алгоритм: (DEST) ← – (DEST)

NEGB Dbreg

Функция: **Изменение знака байта**

Код: (00010011) (Dbreg)

Описание: Изменение знака байта операнда на противоположный.

Алгоритм: (DEST) ← – (DEST)

NOP

Функция: **Нет операции**

Код: (11111101)

Описание: Пустая однобайтная команда. Управление передается следующей по порядку команде.

Алгоритм: PC ← PC + 1

NORML SIreg, Dbreg

Функция: **Нормализация двойного слова**

Код: (00001111) (Dbreg) (SIreg)

Описание: Сдвиг двойного слова источника 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.

Алгоритм: (COUNT) ← 00h
Пока (старший бит SRC = 0b) и (COUNT)<31)
Выполнять
(SRC) ← (SRC) × 2
(COUNT) ← (COUNT) + 1
По окончании (DEST) ← (COUNT)

NOT Dwreg

Функция: **Инверсия слова**

Код: (00000010) (Dwreg)

Описание: Побитное инвертирование слова (каждая единица меняется на ноль, каждый ноль меняется на единицу).

Алгоритм: (DEST) ← NOT (DEST)

NOTB Dbreg

Функция: **Инверсия байта**

Код: (00010010) (Dbreg)

Описание: Побитное инвертирование байта (каждая единица меняется на ноль, каждый ноль меняется на единицу).

Алгоритм: (DEST) ← NOT (DEST)

OR Dwreg, waop

Функция: **Логическое «ИЛИ» слов**

Код: (100000aa) (waop) (Dwreg)

Описание: Логическое сложение слова источника SRC и слова приемника DEST и сохранение результата в приемнике.

Алгоритм: (DEST) ← (DEST) OR (SRC)

ORB Dbreg, baop

Функция: **Логическое «ИЛИ» байт**

Код: (100100aa) (baop) (Dbreg)

Описание: Логическое сложение байта источника SRC и байта приемника DEST и сохранение результата в приемнике.

Алгоритм: (DEST) ← (DEST) OR (SRC)

POP waop

Функция: **Чтение слова из стека**

Код: (110011aa) (waop)

Описание: Чтение слова из вершины стека и размещение его в приемнике DEST.

Алгоритм: (DEST) ← (SP)
SP ← SP + 2

POPA

Функция: **Чтение всего стека**

Код: (11110101)

Описание: Команда используется взамен команды POPF для поддержки восьми добавочных прерываний. Два слова читаются из стека и размещаются в сдвоенных регистрах INT_MASK1/WSR (первое слово) и PSW/INT_MASK (второе слово). Указатель стека SP инкрементируется на четыре. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: INT_MASK1/WSR ← (SP)
SP ← SP + 2
PSW/INT_MASK ← (SP)
SP ← SP + 2

POPF

Функция: **Чтение флагов из стека**

Код: (11110011)

Описание: Чтение слова из вершины стека и размещение его в регистре PSW. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: (PSW) ← (SP)
SP ← SP + 2

PUSH waop

Функция: **Загрузка слова в стек**

Код: (110010aa) (waop)

Описание: Загрузка значения операнда в стек.

Алгоритм: (SP) ← SP – 2
(SP) ← (DEST)

PUSHA

Функция: **Загрузка всего в стек**

Код: (11110100)

Описание: Команда используется взамен команды PUSHF для поддержки восьми добавочных прерываний. Два слова загружаются в стек из сдвоенных регистров PSW/INT_MASK (первое слово) и INT_MASK1/WSR (второе слово), после чего регистры PSW, INT_MASK и INT_MASK1 очищаются. Указатель стека SP декрементируется на четыре. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: SP ← SP – 2
(SP) ← PSW/INT_MASK
PSW/INT_MASK ← 0000h
SP ← SP – 2
(SP) ← INT_MASK1/WSR
INT_MASK1 ← 00h

PUSHF

Функция: **Загрузка флагов в стек**

Код: (11110010)

Описание: Загрузка значения сдвоенного регистра PSW/INT_MASK в вершину стека и очистка регистра PSW/INT_MASK. Все прерывания автоматически запрещаются. Запросы на прерывания не могут идти сразу после этой команды.

Алгоритм: SP ← SP – 2
(SP) ← PSW/INT_MASK
PSW/INT_MASK ← 0000h

RET

Функция: **Возврат из подпрограммы**

Код: (11110000)

Описание: Считывание значения адреса (адреса возврата) из вершины стека и размещение его в программном счетчике PC. Указатель стека SP инкрементируется на два.

Алгоритм: PC ← (SP)
SP ← SP + 2

RST

Функция: **Сброс системы**

Код: (11111111)

Описание: Очистка регистра PSW, загрузка в программный счетчик PC значения 2080h, загрузка в регистры SFR их начальных значений. В результате выполнения этой команды вывод RESET# микроконтроллера находится в состоянии логического нуля 16 машинных тактов.

Алгоритм: PSW ← 00h
PC ← 2080h
SFR ← начальное значение SFR
Вывод #RESET в состояние логического нуля

SCALL cadd

Функция: **Короткий вызов**

Код: (00101xxx) (disp)

Описание: 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.

Алгоритм: SP ← SP – 2
(SP) ← PC
PC ← PC + ((xxx)(disp))

SETC

Функция: **Установка флага переноса**

Код: (11111001)

Описание: Запись единицы в бит C регистра PSW.

Алгоритм: C ← 1b

SHL Dwreg, scount

Функция: **Логический сдвиг слова влево**

Код: (00001001) (scount) (Dwreg)

Описание: Сдвиг слова приемника 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 прямо, как содержимое какого-либо регистра.

Алгоритм:
Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
    C ← Вытесненный старший бит операнда DEST
    (DEST) ← (DEST) × 2
    Temp ← Temp – 1

SHLB Dbreg, scount

Функция: **Логический сдвиг байта влево**

Код: (00011001) (scount) (Dbreg)

Описание: Сдвиг байта приемника 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 прямо, как содержимое какого-либо регистра.

Алгоритм:
Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
    C ← Вытесненный старший бит операнда DEST
    (DEST) ← (DEST) × 2
    Temp ← Temp – 1

SHLL DIreg, scount

Функция: **Логический сдвиг двойного слова влево**

Код: (00001101) (scount) (DIreg)

Описание: Сдвиг двойного слова приемника 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 прямо, как содержимое какого-либо регистра.

Алгоритм:
Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
    C ← Вытесненный старший бит операнда DEST
    (DEST) ← (DEST) × 2
    Temp ← Temp – 1

SHR Dwreg, scount

Функция: **Логический сдвиг слова вправо**

Код: (00001000) (scount) (Dwreg)

Описание: Сдвиг слова приемника 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 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.

Алгоритм:
Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
    C ← Вытесненный младший бит операнда DEST
    (DEST) ← (DEST) / 2
    Temp ← Temp – 1

SHRA Dwreg, scount

Функция: **Арифметический сдвиг слова вправо**

Код: (00001010) (scount) (Dwreg)

Описание: Сдвиг слова приемника 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 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.

Алгоритм: Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1

SHRAB Dbreg, scount

Функция: **Арифметический сдвиг байта вправо**

Код: (00011010) (scount) (Dbreg)

Описание: Сдвиг байта приемника 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.

Алгоритм: Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1

SHRAL Dwreg, scount

Функция: **Арифметический сдвиг двойного слова вправо**

Код: (00001110) (scount) (Dwreg)

Описание: Сдвиг двойного слова приемника 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. 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.

Алгоритм: Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1

SHRB Dbreg, scount

Функция: **Логический сдвиг байта вправо**

Код: (00011000) (scount) (Dbreg)

Описание: Сдвиг байта приемника 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 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.

Алгоритм: Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1

SHRL DIreg, scount

Функция: **Логический сдвиг двойного слова вправо**

Код: (00001100) (scount) (DIreg)

Описание: Сдвиг двойного слова приемника 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 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.

Алгоритм: Temp ← (COUNT)
Пока Temp ≠ 0
Выполнять
C ← Вытесненный младший бит операнда DEST
(DEST) ← (DEST) / 2
Temp ← Temp – 1

SJMP cadd

Функция: **Короткий переход**

Код: (00100xxx) (disp)

Описание: 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.

Алгоритм: PC ← PC + ((xxx)(disp))

SKIP breg

Функция: **Нет операции**

Код: (00000000) (breg)

Описание: Пустая двухбайтная команда. Управление передается следующей по порядку команде. Значение операнда не важно.

Алгоритм: PC ← PC + 2

ST Swreg, waop

Функция: **Сохранение слова**

Код: (110000aa) (waop) (Swreg)

Описание: Сохранение слова источника SRC в слове приемника DEST.

Алгоритм: (DEST) ← (SRC)

STB Sbreg, baop

Функция: **Сохранение байта**

Код: (110001aa) (baop) (Sbreg)

Описание: Сохранение байта источника SRC в байте приемника DEST.

Алгоритм: (DEST) ← (SRC)

SUB Dwreg, waop
Dwreg, wreg, waop

Функция: **Вычитание слов**

Код: (011010aa) (waop) (Dwreg)
(011010aa) (waop) (wreg) (Dwreg)

Описание: Вычитание слова источника SRC из слова приемника DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW.
Вычитание слова второго источника SRC2 из слова первого источника SRC1 и сохранение результата в приемнике DEST. При заеме устанавливается флаг переноса C в регистре PSW.

Алгоритм: (DEST) ← (DEST) – (SRC)
(DEST) ← (SRC1) – (SRC2)

SUBB Dbreg, baop
Dbreg, breg, baop

Функция: **Вычитание байт**

Код: (010110aa) (baop) (Dbreg)
(010110aa) (baop) (breg) (Dbreg)

Описание: Вычитание байта источника SRC из байта приемника DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW.
Вычитание байта второго источника SRC2 из байта первого источника SRC1 и сохранение результата в приемнике DEST. При заеме устанавливается флаг переноса C в регистре PSW.

Алгоритм: (DEST) ← (DEST) – (SRC)
(DEST) ← (SRC1) – (SRC2)

SUBC Dwreg, waop

Функция: **Вычитание слов с переносом**

Код: (101010aa) (waop) (Dwreg)

Описание: Вычитание слова источника SRC и флага переноса из слова приемника DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW.

Алгоритм: (DEST) ← (DEST) – (SRC) – (C)

SUBCB Dbreg, baop

Функция: **Вычитание байт с переносом**

Код: (101110aa) (baop) (Dbreg)

Описание: Вычитание байта источника SRC и флага переноса из байта приемника DEST и сохранение результата в приемнике. При заеме устанавливается флаг переноса C в регистре PSW.

Алгоритм: (DEST) ← (DEST) – (SRC) – (C)

TIJMP tbase, [index], #mask

Функция: **Косвенный табличный переход**

Код: (11100010) (index) (mask) (tbase)

Описание: Продолжение выполнения программы по адресу, выбранному из таблицы адресов переходов. Операнд (tbase) содержит 16-битный адрес начала таблицы переходов. Регистр этого адреса может располагаться в диапазоне адресов до 0FEh без работы с окнами и выше 0FFh при оконной адресации. Таблица адресов переходов может размещаться на странице памяти размером до 0FFh в любой свободной области, выровненной по границе слова.
Операнд ([index]) содержит 16-битный адрес байта, содержащий 7-битный индекс перехода. Регистр для хранения значения (index) может быть в диапазоне адресов до 0FEh без работы с окнами и выше 0FFh при оконной адресации.
Байтовый непосредственный операнд (#mask) содержит 7-битную маску, которая накладывается по «И» на значение (index) при расчете величины смещения OFFSET.
Адрес вектора перехода DEST X определяется путем добавления к адресу начала таблицы переходов удвоенного значения смещения.
Для задания таблицы адресов переходов следует использовать псевдокоманду ассемблера DCW. При этом автоматически выполняется выравнивание таблицы по границе слова.

Алгоритм: OFFSET ← (index) AND (mask)
DEST X ← tbase + (2×OFFSET)
PC ← (DEST X)

TRAP

Функция: **Программное прерывание**

Код: (11110111)

Описание: Выполнение этой команды эквивалентно вызову процедуры обслуживания прерывания по вектору 2010h и не зависит от состояния флага I в регистре PSW. Запросы на прерывания не могут идти сразу после этой команды.
Следует помнить, что команда TRAP не поддерживается транслятором с языка ассемблер. Она используется исключительно в инструментальных системах разработки, в частности в мониторах-отладчиках для реализации режимов выполнения пользовательских программ с точками останова или в пошаговом режиме. Попытка ввода этой команды на языке ассемблера приведет к сообщению транслятора о синтаксической ошибке.

Алгоритм: SP ← SP – 2
(SP) ← PC
PC ← (2010h)

XCH Dwreg, waop

Функция: **Обмен слов**

Код: (00000100) (waop) (Dwreg)
(00001011) (waop) (Dwreg)

Описание: Обмен словами между источником SRC и приемником DEST.

Алгоритм: (DEST) ↔ (SRC)

XCHB Dbreg, baop

Функция: **Обмен байт**

Код: (00010100) (baop) (Dbreg)
(00011011) (baop) (Dbreg)

Описание: Обмен байтами между источником SRC и приемником DEST.

Алгоритм: (DEST) ↔ (SRC)

XOR Dwreg, waop

Функция: **Логическое исключающее «ИЛИ» слов**

Код: (100001aa) (waop) (Dwreg)

Описание: Логическое сложение по модулю два слова источника SRC и слова приемника DEST и сохранение результата в приемнике.

Алгоритм: (DEST) ← (DEST) XOR (SRC)

XORB Dbreg, baop

Функция: **Логическое исключающее «ИЛИ» байт**

Код: (100101aa) (baop) (Dbreg)

Описание: Логическое сложение по модулю два байта источника SRC и байта приемника DEST и сохранение результата в приемнике.

Алгоритм: (DEST) ← (DEST) XOR (SRC)

Особенности системы команд

1 Опкоды 10h, 1Ch, 1Dh, 1Eh, 1Fh, E4h, E5h, E6h, E8h, E9h, EAh, Ebh, F1h являются зарезервированными и при обнаружении вызывают генерирование прерывания невыполнимого кода.

2 Опкод EEh является зарезервированным, но при обнаружении не вызывает генерирование прерывания невыполнимого кода.

3 Опкод FEh используется в качестве старшего байта двухбайтного опкода команд знакового деления DIV, DIVB и знакового умножения MUL, MULB.

4 Каждый сдвиг на один бит при выполнении команды логического сдвига влево (SHL и SHLL) идентичен беззнаковому арифметическому умножению на два. Каждый сдвиг на один бит при выполнении команды логического сдвига вправо (SHR, SHRL) идентичен беззнаковому арифметическому делению на два, а арифметического сдвига вправо (SHRA, SHRAL) – знаковому арифметическому делению на два.

Приложение В
(обязательное)

Коды состояний функционирования модуля I2C

В таблицах В1 – В11 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 SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
00h	IDLE	–	–	–	–	–	Ожидать завершения текущей передачи байта
1Fh	Ошибка на шине	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)

Таблица В.2 – Режим FS мастера передатчика (дополнительно см. таблицу В.4)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
01h	Старт	Код мастера	1	0	0	0	Передать код мастера и перейти в режим HS (0Ch/ 21h)
		[ADR,0]					Передать адрес ведомого (04h/ 05h)
02h	Повторный старт	[ADR,0]	1	0	0	0	Передать адрес ведомого (04h/ 05h)
		[ADR,1]					Передать адрес ведомого, после чего перейти в режим приемника (08h/ 09h)

Окончание таблицы 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 SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
08h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)
09h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
0Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)

Окончание таблицы 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 В2)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
0Ch	Отправлен код мастера, обнаружена ошибка (ACK)	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица В.5 – Режим FS ведомого приемника (дополнительно см. таблицу В7)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
10h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
11h	Принят адрес после потери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
12h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
13h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица В.6 – Режим FS ведомого передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
14h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
15h	Принят адрес после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
16h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (16h/17h)
17h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
18h	Принят адрес отклика и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)
19h	Принят адрес отклика после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)
1Ah	Отправлен байт данных в ответ на получение адреса отклика с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (1Ah/1Bh)

Окончание таблицы 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 В5)

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
1Ch	Стоп	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
1Dh	Принят адрес общего вызова и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
1Eh	Принят адрес общего вызова после потери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)

Таблица В.8 – Режим HS мастера передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
21h	Успешно отправлен код мастера, мастер перешел в режим HS	–	1	0	0	1	Сделать повторный старт (22h)
22h	Повторный старт	[ADR,0]	1	0	0	0	Передать адрес ведомого (28h/29h)
		[ADR,1]					Передать адрес ведомого, после квитирования/не квитирования переключиться в режим мастера приемника (28h/29h)
23h	Потеря арбитража, мастер перешел в режим HS безадресного ведомого	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
24h	Отправлен адрес ведомого с ACK	DAT	1	0	0	0	Передать байт данных (26h/27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
25h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
26h	Отправлен байт данных с ACK	DAT	1	0	0	0	Передать байт данных (26h/ 27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
27h	Отправлен байт данных с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица В.9 – Режим HS мастера приемника

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
28h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
29h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
2Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
2Bh	Принят байт данных и не квити-рован	DAT	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица В.10 – Режим HS ведомого приемника

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
30h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
32h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
33h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица В.11 – Режим HS ведомого передатчика

Код	Описание состояния	Регистр SMBSDA	Биты регистра SMBCTRL1				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
34h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (36h/37h)
36h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/не квитировать (36h/37h)
37h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Подп.	Дата
	изме-ненных	замененных	новых	анну-лиро-ванных				