МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1874ВЕ96Т

Руководство пользователя

2011

Содержание

1 Введение

В настоящем руководстве КФДЛ.431295.042 приведено описание архитектуры, функционального построения, системы команд и особенностей применения микросхемы 1874ВЕ96Т. Микросхема представляет собой СБИС однокристального 16-разрядного микроконвертера с тактовой частотой до 33 МГц, ОЗУ (2024×8) бит, расширенным ОЗУ (2048×8) бит, восемью 16-разряднымыми АЦП с возможностью дифференциального включения входов и блоком цифровых компараторов, 14-разрядным ЦАП, 3-канальным ШИМ, двумя последовательными портами ввода-вывода (UART0 и UART1), синхронным последовательным интерфейсом SPI, интерфейсом I2C, с внутренней памятью программ типа EEPROM (16K×16) бит, устройством высокоскоростного ввода-вывода, отладочным модулем и сторожевым таймером.

Изделие служит цели развития отечественной элементной базы для применения в различных цифровых системах управлениях, радиосвязи и изделиях, требующих точные аналогово-цифровые и цифро-аналоговые преобразования.

Настоящее руководство может служить практическим пособием по применению микроконвертера для разработчиков систем на основе ИС 1874ВЕ96Т.

2 Назначение и область применения

Архитектура микроконвертера ориентирована на создание цифровых управляющих систем, функционирующих в режиме реального времени с возможностью адаптации и модификации под конкретные приложения. Изделие может служить элементной базой для цифровых систем управления различной аппаратурой, силовой электроникой, автомобильной техникой и т. д. Наличие встроенных аппаратных сдвигателя, умножителя и делителя и возросшая производительность ядра позволят использовать микроконвертер при решении задач обработки сигналов.

Наличие средств инструментальной отладки и встроенного отладочного модуля обеспечивает как эффективное проектирование систем на основе микроконвертера, так и возможность смены алгоритма работы при создании модификаций систем.

Возможность гибкого управления энергопотреблением микроконвертера (три режима пониженного энергопотребления, возможность отключения неиспользуемых блоков) позволит использовать микросхему в критичных к потреблению приложениях.

3 Краткое техническое описание ИС 1874ВЕ96Т

3.1 Функциональные параметры микросхемы

Функциональные параметры ИС 1874ВЕ96Т:

разрядность данных, бит	16
частота следования импульсов тактового сигнала, МГц	до 33
динамически конфигурируемая шина данных, бит	8 или 16
встроенная память программ типа EEPROM, бит	16К×16
регистровое ОЗУ, бит	2024×8
адресуемая память, бит	64К×8
число источников прерывания	44
число параллельных 8-разрядных портов ввода-вывода	5
разрядность сторожевого таймера	16
число 16-разрядных таймеров/счетчиков	2
разрядность таймеров/счетчиков	16
универсальный последовательный порт UART	2
синхронный последовательный интерфейс SPI	1
интерфейс I2C	1
количество АЦП	8
максимальное количество используемых АЦП при дифференциальном включении входов	4
число разрядов аналого-цифрового преобразователя	16
число разрядов цифро-аналогового преобразователя	14
число каналов блока ШИМ	3
количество режимов энергопотребления	4
генератор псевдослучайных последовательностей	1
модуль отладки OCDS	1
периферийный сервер PTS	1

Микросхема выполнена в металлокерамическом планарном корпусе с четырехсторонним расположением выводов 4235.88-1 и предназначена для ручной и автоматической сборки в соответствии с ГОСТ РВ 20.39.412-97. Масса микросхемы – не более 4,5 г.

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

Микросхемы не имеют собственных резонансных частот ниже 100 Гц.

Структурная схема микросхемы 1874ВЕ96Т приведена на рисунке 3.1.

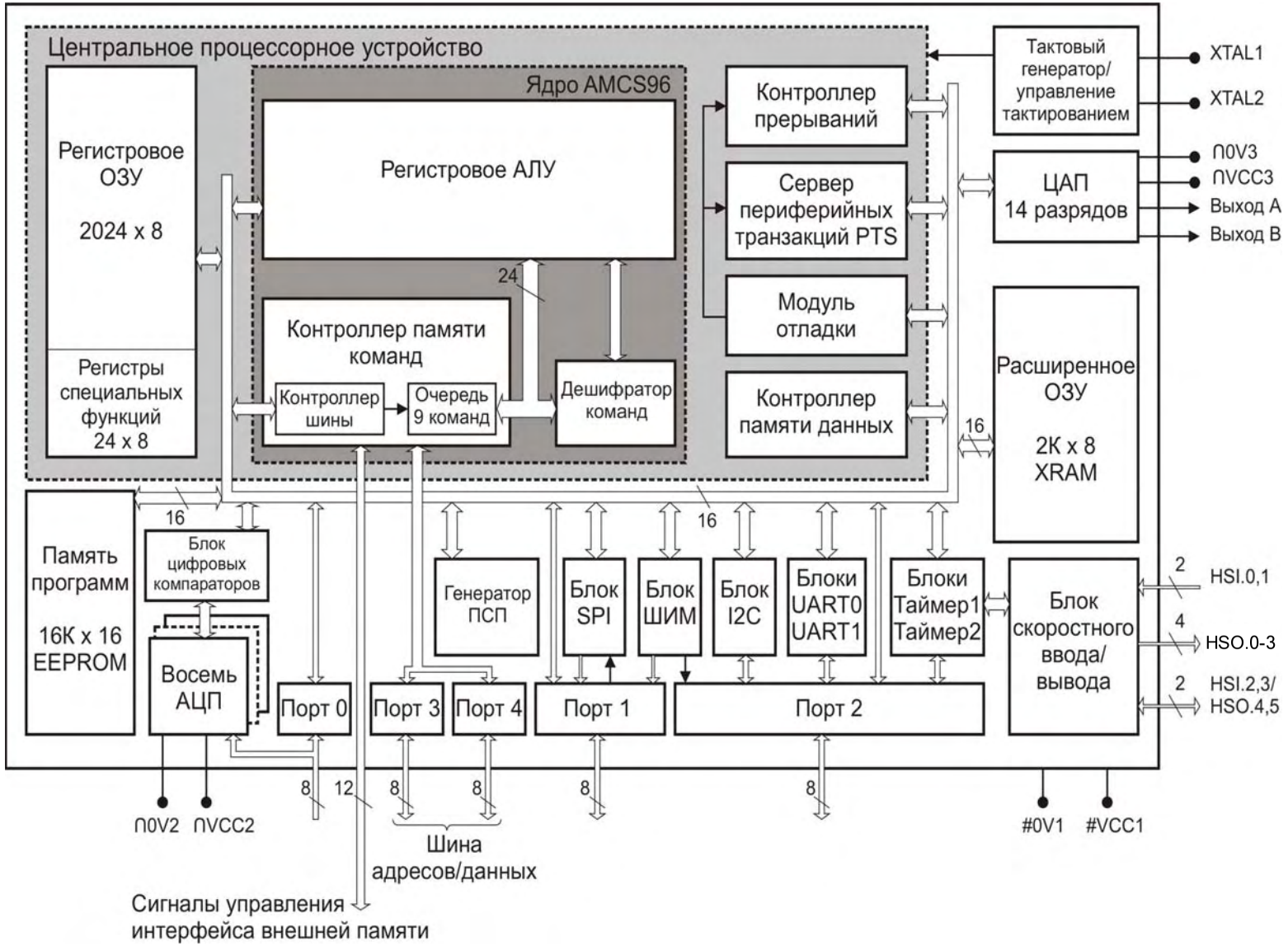


Рисунок 3.1 – Структурная схема микроконвертера 1874ВЕ96Т

Условное графическое обозначение микросхемы приведено на рисунке 3.2. Функциональное назначение выводов микросхем приведено в таблице 3.1.

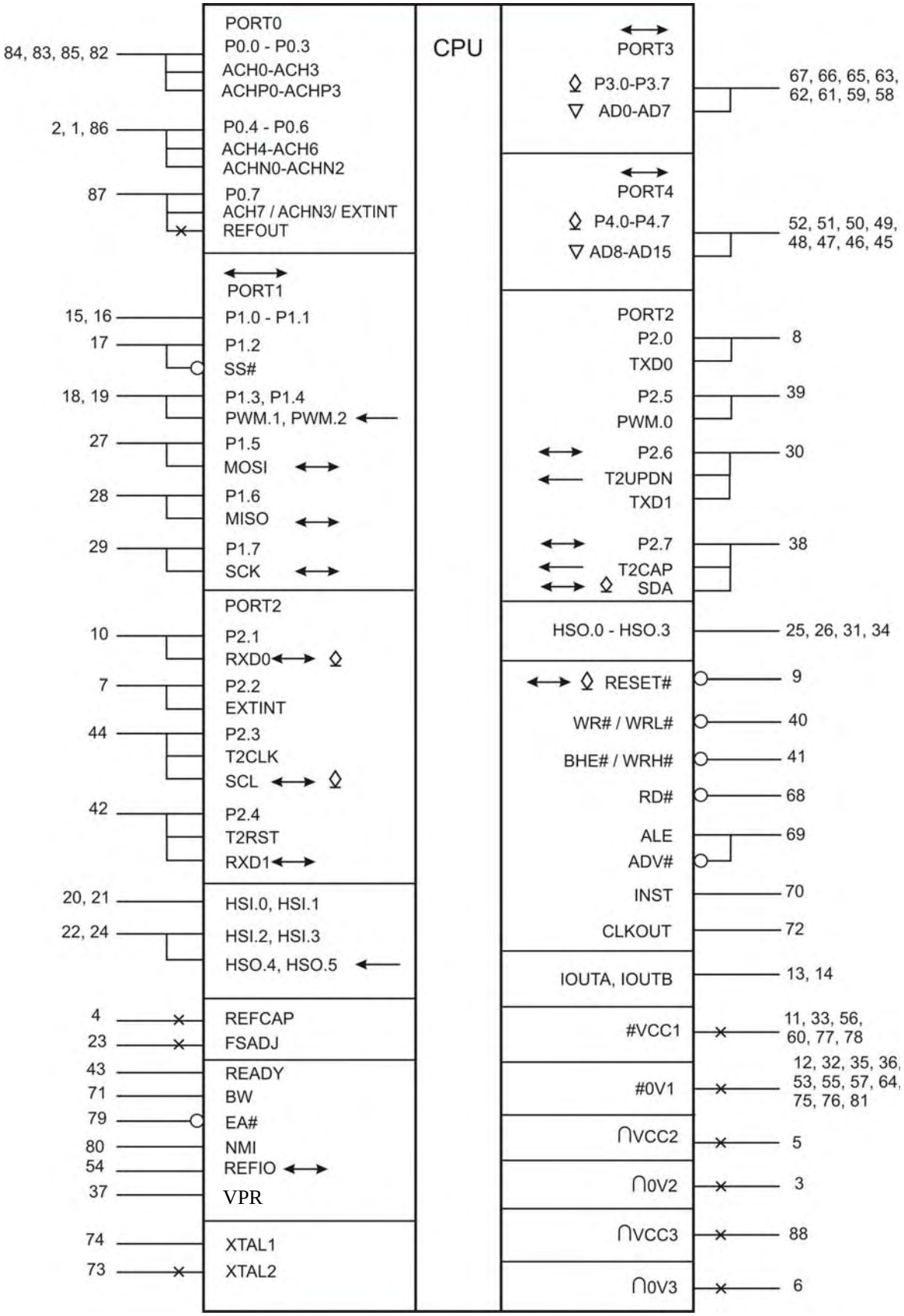


Рисунок 3.2 – Условное графическое изображение микросхемы 1874BE96T

Таблица 3.1 – Функциональное назначение выводов микросхем

Обозначение вывода	Номер вывода	Функциональное назначение	Тип вывода	Обозначение альтернативной функции вывода
1	2	3	4	5
P0.0	84	Вход «порт 0, 0 разряд» Вход АЦП, канал 0 Прямой вход АЦП, канал 0	I I I	 ACH0 ACHP0
P0.1	83	Вход «порт 0, 1 разряд» Вход АЦП, канал 1 Прямой вход АЦП, канал 1	I I I	 ACH1 ACHP1
P0.2	85	Вход «порт 0, 2 разряд» Вход АЦП, канал 2 Прямой вход АЦП, канал 2	I I I	 ACH2 ACHP2
P0.3	82	Вход «порт 0, 3 разряд» Вход АЦП, канал 3 Прямой вход АЦП, канал 3	I I I	 ACH3 ACHP3
P0.4	2	Вход «порт 0, 4 разряд» Вход АЦП, канал 4 Инверсный вход АЦП, канал 0 Выбор режима программирования*	I I I I	 ACH4 ACHN0 PMODE0
P0.5	1	Вход «порт 0, 5 разряд» Вход АЦП, канал 5 Инверсный вход АЦП, канал 1 Выбор режима программирования*	I I I I	 ACH5 ACHN1 PMODE1
P0.6	86	Вход «порт 0, 6 разряд» Вход АЦП, канал 6 Инверсный вход АЦП, канал 2 Выбор режима программирования*	I I I I	 ACH6 ACHN2 PMODE2
P0.7	87	Вход «порт 0, 7 разряд» Вход АЦП, канал 7 Вход «сигнал внешнего прерывания» Инверсный вход АЦП, канал 3 Буферизированный вывод опорного напряжения АЦП Выбор режима программирования*	I I I I – I	 ACH7 EXTINT ACHN3 REFOUT PMODE3
P1.0	15	Вход/выход «порт 1, 0 разряд»	I/O	
P1.1	16	Вход/выход «порт 1, 1 разряд»	I/O	
P1.2	17	Вход/выход «порт 1, 2 разряд» Вход выбора ведомого порта SPI	I/O I	 SS#
P1.3	18	Вход/выход «порт 1, 3 разряд» Выход ШИМ, 1 канал	I/O O	 PWM.1
P1.4	19	Вход/выход «порт 1, 4 разряд» Выход ШИМ, 2 канал	I/O O	 PWM.2
P1.5	27	Вход/выход «порт 1, 5 разряд» Вход данных ведомого/выход данных ведущего порта SPI	I/O I/O	 MOSI
P1.6	28	Вход/выход «порт 1, 6 разряд» Выход данных ведомого/вход данных ведущего порта SPI	I/O I/O	 MISO

Продолжение таблицы 3.1

1	2	3	4	5
P1.7	29	Вход/выход «порт 1, 7 разряд» Вход/выход тактовой частоты порта SPI	I/O I/O	 SCK
P2.0	8	Выход «порт 2, 0 разряд» Выход последовательных данных UART0 Выход «верификации»*	O O O	 TXD0 PVER
P2.1	10	Вход «порт 2, 1 разряд» Вход/выход последовательных данных UART0 Вход фиксации адреса/команды*	I I/O/2 I	 RXD0 PALE#
P2.2	7	Вход «порт 2, 2 разряд» Вход «сигнал внешнего прерывания» Вход фиксации данных*	I I I	 EXTINT PROG#
P2.3	44	Вход «порт 2, 3 разряд» Вход «синхронизация таймера 2» Вход-выход синхронизации порта I2C	I I I/O/2	 T2CLK SCL
P2.4	42	Вход «порт 2, 4 разряд» Вход «сброс таймера 2» Вход/выход последовательных данных порта UART1 Вход «автоинкремент»*	I I I/O I	 T2RST RXD1 AINC#
P2.5	39	Выход «порт 2, 5 разряд» Выход ШИМ, 0 канал	O O	 PWM.0
P2.6	30	Вход/выход «порт 2, 6 разряд» Вход «выбор режима таймера 2» Выход последовательных данных порта UART1 Выход «ошибка программирования»*	I/O I O O	 T2UPDN TXD1 CPVER
P2.7	38	Вход/выход «порт 2, 7 разряд» Вход «выборка данных таймера 2» Вход/выход данных шины I2C Выход «подтверждение программирования»*	I/O I I/O/2 O	 T2CAP SDA PACT#
P3.0	67	Вход/выход «порт 3, 0 разряд» Вход/выход «адрес-данные, 0 разряд» Вход/выход «адрес-команда-данные, 0 разряд»*	I/O/2 I/O/Z I/O/Z	 AD0 PBUS0
P3.1	66	Вход/выход «порт 3, 1 разряд» Вход/выход «адрес-данные, 1 разряд» Вход/выход «адрес-команда-данные, 1 разряд»*	I/O/2 I/O/Z I/O/Z	 AD1 PBUS1
P3.2	65	Вход/выход «порт 3, 2 разряд» Вход/выход «адрес-данные, 2 разряд» Вход/выход «адрес-команда-данные, 2 разряд»*	I/O/2 I/O/Z I/O/Z	 AD2 PBUS2
P3.3	63	Вход/выход «порт 3, 3 разряд» Вход/выход «адрес-данные, 3 разряд» Вход/выход «адрес-команда-данные, 3 разряд»*	I/O/2 I/O/Z I/O/Z	 AD3 PBUS3

Продолжение таблицы 3.1

1	2	3	4	5
P3.4	62	Вход/выход «порт 3, 4 разряд» Вход/выход «адрес-данные, 4 разряд» Вход/выход «адрес-команда-данные, 4 разряд»*	I/O/2 I/O/Z I/O/Z	 AD4 PBUS4
P3.5	61	Вход/выход «порт 3, 5 разряд» Вход/выход «адрес-данные, 5 разряд» Вход/выход «адрес-команда-данные, 5 разряд»*	I/O/2 I/O/Z I/O/Z	 AD5 PBUS5
P3.6	59	Вход/выход «порт 3, 6 разряд» Вход/выход «адрес-данные, 6 разряд» Вход/выход «адрес-команда-данные, 6 разряд»*	I/O/2 I/O/Z I/O/Z	 AD6 PBUS6
P3.7	58	Вход/выход «порт 3, 7 разряд» Вход/выход «адрес-данные, 7 разряд» Вход/выход «адрес-команда-данные, 7 разряд»*	I/O/2 I/O/Z I/O/Z	 AD7 PBUS7
P4.0	52	Вход/выход «порт 4, 0 разряд» Вход/выход «адрес-данные, 8 разряд» Вход/выход «адрес-команда-данные, 8 разряд»*	I/O/2 I/O/Z I/O/Z	 AD8 PBUS8
P4.1	51	Вход/выход «порт 4, 1 разряд» Вход/выход «адрес-данные, 9 разряд» Вход/выход «адрес-команда-данные, 9 разряд»*	I/O/2 I/O/Z I/O/Z	 AD9 PBUS9
P4.2	50	Вход/выход «порт 4, 2 разряд» Вход/выход «адрес-данные, 10 разряд» Вход/выход «адрес-команда-данные. 10 разряд»*	I/O/2 I/O/Z I/O/Z	 AD10 PBUS10
P4.3	49	Вход/выход «порт 4, 3 разряд» Вход/выход «адрес-данные, 11 разряд» Вход/выход «адрес-команда-данные, 11 разряд»*	I/O/2 I/O/Z I/O/Z	 AD11 PBUS11
P4.4	48	Вход/выход «порт 4, 4 разряд» Вход/выход «адрес-данные, 12 разряд» Вход/выход «адрес-команда-данные, 12 разряд»*	I/O/2 I/O/Z I/O/Z	 AD12 PBUS12
P4.5	47	Вход/выход «порт 4, 5 разряд» Вход/выход «адрес-данные, 13 разряд» Вход/выход «адрес-команда-данные, 13 разряд»*	I/O/2 I/O/Z I/O/Z	 AD13 PBUS13
P4.6	46	Вход/выход «порт 4, 6 разряд» Вход/выход «адрес-данные, 14 разряд» Вход/выход «адрес-команда-данные, 14 разряд»*	I/O/2 I/O/Z I/O/Z	 AD14 PBUS14
P4.7	45	Вход/выход «порт 4, 7 разряд» Вход/выход «адрес-данные, 15 разряд» Вход/выход «адрес-команда-данные, 15 разряд»*	I/O/2 I/O/Z I/O/Z	 AD15 PBUS15
HSI.0	20	Вход «быстрый ввод, канал 0»	I	
HSI.1	21	Вход «быстрый ввод, канал 1»	I	

Продолжение таблицы 3.1

1	2	3	4	5
HSI.2	22	Вход «быстрый ввод, канал 2» Выход «быстрый вывод, канал 4»	I O	 HSO.4
HSI.3	24	Вход «быстрый ввод, канал 3» Выход «быстрый вывод, канал 5»	I O	 HSO.5
HSO.0	25	Выход «быстрый вывод, канал 0»	O	
HSO.1	26	Выход «быстрый вывод, канал 1»	O	
HSO.2	31	Выход «быстрый вывод, канал 2»	O	
HSO.3	34	Выход «быстрый вывод, канал 3»	O	
EA#	79	Вход «внешний доступ»	I	
BW	71	Вход «разрядность внешней шины»	I	
READY	43	Вход «готовность»	I	
NMI	80	Вход «немаскируемое прерывание»	I	
RESET#	9	Вход/выход «сброс»	I/O/2	
INST	70	Выход «чтение команды»	O	
ALE	69	Выход «разрешение записи адреса» Выход «адрес действителен»	O O	 ADV#
WR#	40	Выход «запись» Выход «запись младшего байта»	O O	 WRL#
BHE#	41	Выход «разрешение записи старшего байта» Выход «запись старшего байта»	O O	 WRH#
RD#	68	Выход «чтение»	O	
FSADJ	23	Вывод подстройки выходного тока ЦАП	–	
REFCAP	4	Вывод подключения внешнего источника опорного напряжения АЦП/Вывод подключения фильтрующего конденсатора источника опорного напряжения АЦП	–	
REFIO	54	Вход/выход опорного напряжения ЦАП Вывод подключения фильтрующего конденсатора источника опорного напряжения ЦАП	I/O –	
CLKOUT	72	Выход «системный тактовый сигнал»	O	
IOUTA	13	Выход тока ЦАП	O	
IOUTB	14	Комплементарный выход тока ЦАП	O	
XTAL1	74	Вывод подключения кварцевого резонатора Вход тактового сигнала	 – I	
XTAL2	73	Вывод подключения кварцевого резонатора	–	
VPR	37	Вход в режим программирования* Вход «старт с альтернативного адреса» Вход «возврат из режима пониженного потребления»	I I I	
#VCC1	11, 33, 56, 60, 77, 78	Вывод питания 3,3 В цифровой части микросхемы	–	
∩VCC2	5	Вывод питания 3,3 В аналого-цифрового преобразователя	–	

Окончание таблицы 3.1

1	2	3	4	5
∩VCC3	88	Вывод питания 3,3 В цифро-аналогового преобразователя	–	
#0V1	12, 32, 35, 36, 53, 55, 57, 64, 75, 76, 81	Общий вывод цифровой части микросхемы	–	
∩0V2	3	Общий вывод аналого-цифрового преобразователя	–	
∩0V3	6	Общий вывод цифро-аналогового преобразователя	–	

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

_____

* Функция реализуется только при использовании стандартной программы-монитора НИИЭТ, записанной в ПЗУ.

3.2 Электрические параметры микросхемы

Электрические параметры микросхемы при приёмке и поставке и предельно допустимые значения параметров приведены в таблицах 3.2 и 3.3, соответственно.

Номинальное значение напряжения питания микросхемы 1874ВЕ96Т – 3,3 В. Допустимое отклонение напряжения питания ± 10 %. Амплитуда пульсаций напряжения питания – не более 30 мВ. Напряжение источника опорного напряжения – от 2,6 до 3,6 В. Допустимое отклонение напряжения питания от крайних значений – минус 1 % для напряжения 2,6 В и 1 % для напряжения 3,6 В.

Микросхема должна быть стойкой к климатическим воздействиям и сохранять свои параметры в процессе и после воздействия на них климатических факторов, приведенных в таблице 4 ОСТ В 11 0998-99, в том числе:

- пониженной рабочей температуры среды – минус 60 ºС;
- повышенной рабочей температуры среды – 125 ºС;
- повышенной предельной температуры – 150 ºС.

Разработанная микросхема 1874ВЕ96Т должна быть стойкой к воздействию механических факторов по ОСТ В 11 0998-99, к технологическим воздействиям при изготовлении радиоэлектронной аппаратуры по ОСТ В 11 0998-99.

Таблица 3.2 – Электрические параметры при приёмке и поставке в диапазоне рабочих температур окружающей среды от минус 60 до 125 ºС

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С
		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/TXD0, P2.1/RXD0 (в режиме 0), P2.3/SCL, P2.4/RXD1 (в режиме 0), P2.5/PWM.0, P2.6/TXD1, P2.7/SDA, P3.0 – P3.7, P4.0 – P4.7, HSO.0 – HSO.3, HSI.2/HSO.4, HSI.3/HSO.5, В, I_{OL} = 6 мА, U_{CC1} = 3,0 В	U_{OL}	–	0,4	–60 ± 3 25 ± 10 125 ± 3

Продолжение таблицы 3.2

1	2	3	4	5
2 Выходное напряжение высокого уровня по выводам P1.0 – P1.7, P2.6/TXD1, P2.7/SDA, В, I_{OH} = –100 мкА, U_{CC1} = 3,0 В	U_{OH1}	U_{CC1} – 0,7	–	
3 Выходное напряжение высокого уровня по выводам RD#, WR#, ALE, BHE#, INST, CLKOUT, P2.0/TXD0, P2.4/RXD1, P2.5/PWM0, P3.0 – P3.7, P4.0 – P4.7, HSO.0 – HSO.3, HSI.2/HSO.4, HSI.3/HSO.5, В, I_{OH} = – 6 мА, U_{CC1} = 3,0 В	U_{OH2}	U_{CC1} – 0,7	–	
4 Ток утечки низкого уровня по входам P2.2 – P2.4, P3.0 – P3.7, P4.0 – P4.7, HSI.0 – HSI.3, EA#, BW, READY, VPR, NMI, мкА[1)], U_{IL} = 0 В, U_{CC1} = U_{CC2} = 3,6 В	I_{LIL}	–10	–	
5 Ток утечки высокого уровня по входам P2.2 – P2.4, P3.0 – P3.7, P4.0 – P4.7, HSI.0 – HSI.3, EA#, BW, READY, VPR, RESET#, мкА[1)], U_{IH} = U_{CC1}, U_{CC1} = U_{CC2} = 3,6 В	I_{LIH}	–	10	
6 Входной ток низкого уровня по выводу XTAL1[1)], мкА, U_{IL} = 0 В, U_{CC1} = 3,6 В	I_{IL1}	–35	–	
7 Входной ток высокого уровня по выводу XTAL1[1)], мкА, U_{CI} = U_{CC1} = 3,6 В	I_{IH1}	–	35	
8 Входной ток низкого уровня по выводам P1.0 – P1.7, P2.6, P2.7, мкА, U_{IL} = 0,4 В, U_{CC1} = 3,6 В	I_{IL2}	–300	–	
9 Входной ток высокого уровня по выводам P1.0 – P1.7, P2.6, P2.7, мкА, U_{IH} = 2,0 В, U_{CC1} = 3,6 В	I_{IH2}	–1 000	–	
10 Входной ток низкого уровня по выводу RESET#, мкА, U_{IL} = 0 В, U_{CC1} = 3,6 В	I_{IL3}	–800	–200	-60 ± 3 25 ± 10 125 ± 3
11 Входной ток высокого уровня по выводу NMI, мкА, U_{IH} = 2,4 В, U_{CC1} = 3,6 В	I_{IH3}	50	250	
12 Динамический ток потребления цифровой части в режиме RESET по выводам #VCC1, мА, U_{CC1} = 3,6 В, f_{CI} = 33 МГц	I_{OCC1}	–	50	
13 Ток потребления цифровой части в режиме холостого хода по выводам #VCC1, мА, U_{CC1} = 3,6 В, f_{CI} = 33 МГц	I_{CCI1}	–	40	
14 Ток потребления цифровой части в режиме хранения по выводам #VCC1, мА, U_{CC1} = 3,6 В, f_{CI} = 33 МГц	I_{CCS1}	–	3	
15 Динамический ток потребления АЦП по выводу ∩VCC2, мА,[2)] U_{CC2} = U_{CC1} = 3,6 В, f_S = 64 кГц	I_{OCC2}	–	15	

Окончание таблицы 3.2

1	2	3	4	5
16 Динамический ток потребления ЦАП по выводу ∩VCC3, мА, $U_{CC3} = U_{CC1} = 3{,}6$ В, $f_{CI} = 33$ МГц, $I_{OUTFS} = 20$ мА	I_{OCC3}	–	50	
17 Интегральная нелинейность ЦАП, LSB, $U_{CC3} = U_{CC1} = 3{,}3$ В, $I_{OUTFS} = 20$ мА	E_L	–7	7	125 ± 3
		–5	5	–60 ± 3
18 Дифференциальная нелинейность ЦАП, LSB, $U_{CC3} = U_{CC1} = 3{,}3$ В, $I_{OUTFS} = 20$ мА	E_{LD}	–4	4	25 ± 10 85 ± 3 [3]
		–6	6	125 ± 3
19 Погрешность коэффициента усиления ЦАП, % от полной шкалы, $U_{CC3} = U_{CC1} = 3{,}3$ В, $I_{OUTFS} = 20$ мА	GE	–0,5	0,5	–60 ± 3 25 ± 10 125 ± 3
20 Погрешность смещения ЦАП, % от полной шкалы, $U_{CC1} = U_{CC3} = 3{,}3$ В, $I_{OUTFS} = 20$ мА	OE	–0,02	0,02	
21 Общие гармонические искажения ЦАП, дБ, $U_{CC1} = U_{CC3} = 3{,}3$ В, $I_{OUTFS} = 20$ мА, $f_{OUT} = 125$ кГц	THD_1	–	–70	
22 Выходной шум ЦАП, pA $\sqrt{\text{Hz}}$, $U_{CC1} = U_{CC3} = 3{,}3$ В, $I_{OUTFS} = 20$ мА, $f_{OUT} = 125$ кГц	ON	–	20	
23 Общие гармонические искажения АЦП, дБ, [2] $U_{CC2} = U_{CC1} = 3{,}3$ В, PGA = 0 дБ, $f_{IN} = (0 – 4)$ кГц, $f_S = (8; 64)$ кГц	THD_2	–	–76	
24 Отношение сигнал/(шум + искажения) в каналах АЦП, дБ, [2] $U_{CC2} = U_{CC1} = 3{,}3$ В, PGA = 0 дБ, $f_{IN} = (0 – 4)$ кГц, $f_S = (8; 64)$ кГц	SINAD	73	–	
25 Функциональный контроль $U_{CC1} = (3{,}0; 3{,}6)$ В, $f_{CI} = 33$ МГц	ФК	–	–	

_____

[1] Параметры I_{LIL}, I_{LIH}, I_{IL1}, I_{IH1} при температуре минус 60 ºС не измеряются, а гарантируются нормой при температуре (25 ± 10) ºС.

[2] Параметры I_{OCC2}, THD_2, SINAD измеряются в режиме дифференциальных входов.

[3] Параметры E_L и E_{LD} ЦАП при повышенной температуре 85 ºС не измеряются, а гарантируются измерениями при температуре 125 ºС и подтверждаются периодическими испытаниями.

Таблица 3.3 – 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	2	3	4	5	6
1 Напряжение питания цифровой части ИС, В	U_{CC1}	3,0	3,6	–0,3	6,0
2 Напряжение питания АЦП, В	U_{CC2}	3,0	3,6	–0,3	6,0
3 Напряжение питания ЦАП, В	U_{CC3}	3,0	3,6	–0,3	6,0
4 Входное напряжение низкого уровня, В	U_{IL}	0	0,8	–0,3	–
5 Входное напряжение высокого уровня тактового сигнала XTAL1, В	U_{CI}	$0,7U_{CC1}$	U_{CC1}	–	U_{CC1} + 0,3
6 Входное напряжение высокого уровня сигнала RESET#, В	U_{IHRST}	$0,6U_{CC1}$	U_{CC1}	–	U_{CC1} + 0,3
7 Входное напряжение высокого уровня, В	U_{IH}	$0,2U_{CC1}$ + 1	U_{CC1}	–	U_{CC1} + 0,3
8 Напряжение между выводами #0V1 и ∩0V2, В	U_{0V1}– U_{0V2}	–0,01	0,01	–0,3	0,3
9 Входное опорное напряжение АЦП по выводу REFCAP, В	U_{REF}	1,125	1,6	0	U_{CC2}
10 Диапазон преобразования каналов АЦП (на входах ACH.0 – ACH.7), В, * U_{CC2} = 3,3 В, PGA = 0 дБ, U_{REF} = 1,25 В	U_{FSR}	1,44	*	–	–
11 Выходной ток полной шкалы ЦАП, мА U_{CC3} = 3,3 В	I_{OUTFS}	2	20	–	–
12 Выходной ток высокого уровня по выводам P1.0 – P1.7, P2.6, P2.7, мкА	I_{OH1}	–100	–	–200	–
13 Выходной ток высокого уровня по выводам P3.0 – P3.7, P4.0 – P4.7, TXD0/P2.0, RXD0/P2.1, мА	I_{OH2}	–6	–	–10	–
14 Выходной ток низкого уровня, мА	I_{OL}	–	6	–	10
15 Частота следования импульсов тактового сигнала, МГц	f_{CI}	–	33	–	–
16 Длительность фронта и спада входных сигналов, нс	t_{LH}, t_{HL}	–	5	–	–
17 Емкость выходной нагрузки, пФ	C_L	–	70	–	200

Примечание – Не допускается одновременная подача двух и более предельных режимов эксплуатации.

_____

* Верхнее значение практического диапазона преобразования U_{FSR} определяется потребителем, исходя из заданного U_{FSR} и диапазона рабочих температур, с учетом п. 16.1.1 и рисунка16.4а.

4 Архитектура изделия

16-разрядный микроконвертер 1874ВЕ96Т предназначен для выполнения вычислительных инструкций с повышенным быстродействием и высокоточных аналогово-цифровых и цифро-аналоговых преобразований. Он имеет общие архитектуру и набор инструкций с другими микросхемами серии 1874, но базируется на новом ядре AMCS-96 (Advanced MCS-96) разработки НИИЭТ. Данное ядро отличается увеличенной производительностью (в среднем на 30 – 40 % относительно других контроллеров серии 1874, реализующих архитектуру MCS-96, при одинаковой частоте) благодаря введению конвейеризации процесса выборки – выполнения команд, полной оптимизации процесса выполнения инструкций, наличию аппаратных сдвигателя, умножителя, делителя и других конструктивных особенностей. Реализуемая ядром микроконвертера архитектура AMCS-96 полностью программно совместима с ядрами MCS-96 архитектуры. Контроллер обладает ОЗУ 2024×8 бит, доступным быстрым способам адресации, и отключаемым расширенным ОЗУ 2048×8 бит, доступным при косвенном способе адрессации. Контроллер имеет ПЗУ типа EEPROM объемом 16К×16 бит с защитой и удобным механизмом программирования.

Конвертер имеет в своем составе сервер периферийных транзакций PTS, позволяющий передавать данные в моменты простоя шины данных. Это обеспечивает транзакции без остановки ЦПУ, что повышает общее быстродействие системы.

Контроллер имеет возможность начала выполнения программы из разных областей внутренней и внешней памяти, имеет конфигурируемую внешнюю шину адреса/данные.

Для выполнения высокоточных аналого-цифровых и цифро-аналоговых преобразований микроконвертер имеет в своем составе восемь 16-битных АЦП с возможностью параллельного преобразования каналов и дифференциального включения входов и 14-битный высокоскоростной ЦАП. Для более эффективного использования ресурсов ИС имеет блок цифровых компараторов, позволяющий отслеживать выход параметров процессов за границы допустимых значений.

Конвертер имеет гибкую систему управления энергопотреблением. Помимо режимов пониженного энергопотребления (POWERDOWN), холостого хода (IDLE) и медленной работы (SLOW), существует возможность отключения неиспользуемых периферийных устройств.

Микроконвертер имеет подсистему высокоскоростного ввода-вывода с разрешающей способностью в один машинный цикл, позволяющую использовать ИС в быстродействующих системах управления.

ИС имеет в своем составе блок отладки программ с возможностью перезагрузки микросхемы.

Контроллер имеет четыре порта последовательного ввода-вывода: UART0, UART1, SPI и I2C.

В настоящем разделе р у к о в о д с т в а приводится краткое описание устройства и принципов его работы. Полное описание, включая описание системы команд, встроенных функциональных блоков и особенностей программирования, приведено в последующих разделах технического описания.

4.1 Особенности архитектуры

Структурная схема микросхемы 1874ВЕ96Т приведена на рисунке 3.1.

Микроконвертер реализуют архитектуру AMCS-96, полностью программно совместимую с MCS-96 фирмы Intel с изменениями, направленными на увеличение производительности.

Это 16-разрядная быстродействующая ИС высокой степени интеграции, ориентированная на решение задач управления процессами в реальном масштабе времени. В результате применения микросхемы ожидается увеличение функциональности, надежности и производительности встроенных систем управления за счёт использования

полностью нового быстродействующего 16-разрядного ядра, высокоточных 16-разрядного аналого-цифрового и 14-разрядного цифро-аналогового преобразователей, большого объема внутрикристального ОЗУ (4 Кбайт), энергонезависимой памяти программ типа EEPROM (32 Кбайт), устройства высокоскоростного ввода-вывода, четырех последовательных портов приема/передачи. Гибкое управление энергопотреблением позволит использовать ИС 1874ВЕ96Т в критичных к потреблению приложениях, а использование высокоточных АЦП и ЦАП – уменьшить стоимость разрабатываемых систем.

Архитектура микроконвертеров идеально ориентирована для создания модификаций систем для реализации функций управления и вычисления в реальном режиме времени под конкретные приложения. Этому способствует увеличенная производительность микроконвертеров, развитая система встроенных функциональных блоков и мощная система команд. Микросхемы могут служить элементной базой для систем управления различной аппаратурой.

Ключевые особенности микроконвертера

Функционирование на частотах до 33 МГц.

Быстродействующая архитектура типа «регистр-регистр».

Аппаратный умножитель 16×16 за один машинный цикл.

Аппаратный делитель 32×16 за два машинных цикла.

Регистровое ОЗУ емкостью 2024 байта.

Расширенное отключаемое ОЗУ емкостью 2048 байт.

Восемь 16-разрядных аналого-цифровых преобразователей с возможностью дифференциального включения входов и цифровым компаратором на каждый канал.

14-разрядный цифро-аналоговый преобразователь.

Внутренняя программируемая память команд типа EEPROM емкостью 32 Кбайт.

Динамически конфигурируемая разрядность шины данных – 8 или 16 бит.

Устройство высокоскоростного ввода-вывода с разрешающей способностью один машинный цикл.

Два 16-разрядных таймера/счетчика с предделителями и режимами квадратурного счета.

16-разрядный сторожевой таймер.

Два последовательных порта UART.

Синхронный последовательный интерфейс SPI.

Интерфейс I2C.

Пять 8-разрядных портов ввода-вывода.

Режимы холостого хода IDLE, пониженного энергопотребления POWERDOWN, медленной работы SLOW.

Сервер периферийных транзакций PTS.

Модуль отладки OCDS.

Микроконвертеры содержат 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 (Jump Bit Set) и очистке бита (Jump Bit Clear) могут работать с каким-либо регистром SFR или с 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 операций над байтами, иначе мнемоника относится к операциям над словами.

Длинные и двухсловные операции включают операции сдвигов, нормализацию, умножения и деления. В операции “Деление” делится 32-битное число на 16-битное число, что порождает 16-битное частное и 16-битный остаток. В операции «Умножение»

умножается 16-битное 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).

4.2 Краткое описание функционирования микросхемы

4.2.1 Процессорное ядро

Микроконвертеры построены по архитектуре Фон Неймана. Они имеют внешнюю системную магистраль для обмена данными с внешней памятью и дополнительными периферийными устройствами. Система команд микроконвертеров поддерживает широкий набор методов адресации, в т. ч. битовую адресацию (смотри приложение А).

Архитектура микроконвертера, 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 микроконвертера достаточно сложных задач управления в реальном времени.

Главным отличительным признаком архитектуры микроконвертера 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 микроконвертера реализуют операции с байтами, словами, несколько операций с 32-битовыми операндами, а также команды перехода по битам.

Структурная схема ядра AMCS-96 разработки НИИЭТ представлена на рисунке 4.1.

Команды поступают в блок очереди команд, который может хранить до 9 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, расширенного ОЗУ, области периферийных устройств, внутреннего ПЗУ 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, 0A000H (см. 6.4.2). 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 процессора PSW содержит бит PSW.1, который производит общее разрешение обслуживания всех маскируемых прерываний, бит PSW.2,

разрешающий или запрещающий работу сервера периферийного обмена, и шесть битовых флагов, отражающих состояние программы пользователя.

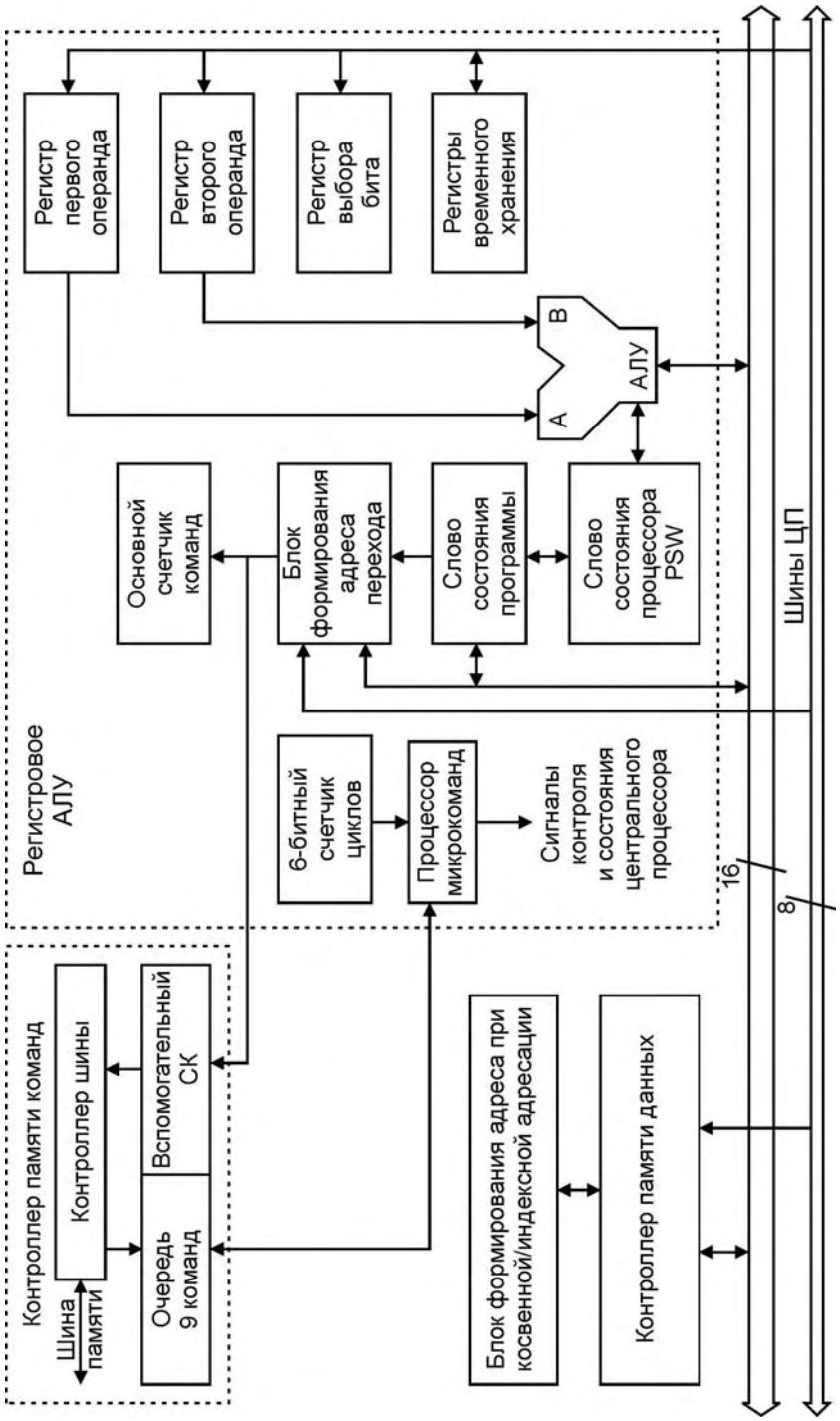


Рисунок 4.1 – Структурная схема ядра микроконвертера

4.2.2 Способы адресации

Система команд микроконвертера содержит следующие типы адресации: прямая регистровая (register direct), косвенная (indirect), косвенная с автоинкрементом (indirect with autoincrement), непосредственная (immediate), короткая индексная (short-indexed) и длинная индексная (long-indexed). Эти типы адресации увеличивают гибкость и скорость выполнения команд устройствами микроконвертера. Каждая команда использует, по крайней мере, один из способов адресации.

Прямая регистровая адресация и непосредственная адресация выполняются наиболее быстро. Прямая регистровая адресация обеспечивает доступ к файлу регистров и SFR. Непосредственная адресация использует информацию, следующую за кодом команды, как операнд.

Оба режима косвенной адресации используют значение слова в регистре как адрес операнда. Косвенная адресация с автоинкрементом увеличивает адресное слово на единицу после операции с байтом и на два – после операции со словом. Этот способ адресации обеспечивает легкий доступ к справочным таблицам.

Длинная индексная адресация обеспечивает прямой доступ к любой ячейке 64К адресного пространства. Этот способ формирует адрес операнда добавлением 16-битного значения к регистровому слову. Индексирование с нулевым регистром позволяет иметь прямую адресацию к любой ячейке. Короткая индексная адресация формирует адрес операнда добавлением 8-битного значения к регистровому слову.

Кроме того, имеются расширенные версии команд прерываемой и непрерываемой передачи блока. Множество способов адресации делает легким программирование на языке ассемблера и обеспечивает отличную взаимосвязь с языками высокого уровня. Команды ассемблера состоят из мнемоники, за которой следуют адрес или данные.

4.2.3 Прерывания

Основной функцией микроконвертера 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 микроконвертер получает запрос прерывания, осуществляет обслуживание и возвращается к прерванной программе.

В разработанном микроконвертере – 44 источника прерывания и 17 векторов прерывания, а также два дополнительных вектора прерывания Software Trap (пошаговое выполнение программ) и Unimplemented Opcodes (неопознанные коды операций), используемых в системах отладки или платах-прототипах. Когда контроллер прерывания определит одно из семнадцати прерываний, он устанавливает соответствующий бит в один из двух регистров ожидания прерываний. Отдельные прерывания разрешаются либо запрещаются установкой или очисткой бит в регистре маски прерываний. Когда контроллер прерывания производит обработку прерывания, он делает вызов программы обслуживания прерываний (ISR). Соответствующий вектор прерывания содержит адрес ISR. Затем контроллер прерывания очищает соответствующий бит ожидания.

Имеются две возможности обслуживания прерываний: программное обслуживание прерываний через контроллер прерываний и микропрограммное обслуживание прерываний через PTS. Можно выбрать вариант обслуживания прерываний для каждого из маскируемых прерываний. Немаскируемые прерывания (NMI, программное прерывание, прерывание по неподдерживаемому коду, прерывание отладочного модуля) всегда обрабатываются подпрограммами обработки прерываний.

4.2.4 Периферийные устройства

Стандартные порты ввода-вывода

Микроконвертеры имеют пять 8-разрядных портов ввода-вывода. Порт 0 – входной, он же порт аналоговых входов для АЦП. Порт 1 – квазидвунаправленный, он разделяет выводы с двумя выходами модуля ШИМ. Порт 2 содержит три типа линий – квазидвунаправленные, входные и выходные. Другие функции МК используют входные и выходные линии совместно с портом 2. Порты 3 и 4 являются двунаправленными портами с открытым стоком, они могут использоваться как интерфейс шины адресов/данных.

Таймеры-счетчики

Микроконвертер имеет два независимых 16-разрядных многофункциональных таймера-счетчика. В одном из таймеров используется внутренняя синхронизация (режим работы – таймер реального времени), в другом – внешняя (счетчик внешних событий). Содержимое таймеров может быть в любое время считано или программно модифицировано, а также сброшено программно или внешним сигналом. Таймер-счетчик внешних событий подсчитывает положительные или отрицательные фронты входных сигналов и может работать как в режиме прямого счета (инкремента), так и обратного (декремента).

Источником счетных импульсов может быть либо системный синхросигнал с фиксированной частотой (возможно предварительно деленной в заданное число раз), либо один из входов микроконвертера. В первом случае устройство выполняет функции таймера, так как фактически считает интервалы времени постоянной длительности.

Во втором случае устройство выполняет функции счетчика событий (отрицательных или положительных перепадов) на входе микросхемы. Счет может производиться либо в одном, либо в обоих направлениях. В последнем случае направление счета определяется уровнем сигнала на соответствующем входе микросхемы. При переходе содержимого счетчика из наибольшего состояния в наименьшее, и наоборот, могут генерироваться соответствующие внутренние запросы на прерывания. Конкретные режимы работы и структура таймеров устанавливаются программно.

Внутренняя синхронизация увеличивает значение таймера 1 и таймера 2 каждый восьмой такт процессора, равный трем периодам тактовой частоты. Таймер 2 может иметь внешнюю синхронизацию. При внешней синхронизации значение таймера 2 увеличивается при каждом положительном или отрицательном перепаде. Любой внешний или внутренний источник может сбросить таймер 2. Таймер 1 и таймер 2 могут сформировать прерывание при переходе границы 0FFFFH/0000H. Микроконвертер также позволяет использовать таймеры для задания скорости передачи информации последовательным портом и для управления работой сторожевого таймера (Watchdog timer). Это внутренний таймер для сброса системы в случае, если программное обеспечение не может нормально функционировать. Два 16-битных таймера могут работать от внутреннего генератора или внешнего источника. Внешний режим «quadrature clocking» пригоден для наблюдения за скоростью и направлением при позиционном кодировании.

Сторожевой таймер

Сторожевой таймер WDT позволяет восстанавливать нормальную работу при сбоях программ. Если WDT разрешен, он будет вызывать аппаратный сброс, если программа не очищает его каждые 64К машинных циклов.

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

Устройства высокоскоростного ввода HSI

HSI может записывать время внешних событий с разрешающей способностью в один машинный цикл. HSI может следить за четырьмя независимыми линиями HSI и определять значение таймера 1, когда произойдет событие. Четыре типа событий могут инициализировать захват: нарастающий фронт, спад сигнала, нарастание и спад или каждый восьмой нарастающий фронт. HSI может хранить восемь элементов (значений таймера 1), семь в семиуровневой очереди FIFO и один – в регистре хранения HSI. Чтение регистра хранения HSI не перегружает значения, размещенные в FIFO ранее. Устройства HSI могут сформировать прерывание при загрузке регистра хранения HSI, или когда загружаемое значение является четвертым или шестым элементом FIFO.

Высокоскоростное устройство вывода HSO

HSO может инициализировать события в определенные моменты, базирующиеся на значениях таймера 1 или таймера 2. Эти программируемые события заключаются в запуске аналого-цифрового преобразования, сбросе таймера 2, формировании до четырех программных временных задержек и установке или очистке одной или нескольких линий из шести выходных линий HSO. HSO хранит ожидаемые события и соответствующее время в блоке памяти Content Addressable Memory (CAM). Этот блок хранит до восьми команд. Каждая команда определяет время действия, «природу» действия, имеет ли место прерывание, идет ссылка на таймер 1 или таймер 2. Каждый такт HSO сравнивает значение ячеек CAM на совпадение с текущим временем. HSO инициализирует определенные события при совпадении времени. Команда стирается из CAM после выполнения или остается в CAM как сохранённая точка входа CAM, тогда команда будет постоянно выполняться при совпадении значения времени для этой точки со значением текущего времени в соответствующем таймере. Сохранённая точка входа полезна в применениях, требующих периодических или повторяющихся событий, таких как повторяющаяся ШИМ последовательность.

Последовательные порты UART

Последовательные порты имеют один синхронный режим (Mode0) и три асинхронных (1, 2 и 3). Асинхронные режимы полностью дуплексные, т. е. они могут передавать и принимать данные одновременно. Приемник на МК буферизован так, что прием второго байта может начаться до считывания первого. Передатчик также дважды буферизован. Наиболее общее использование синхронного режима 0 – расширение возможностей устройств ввода-вывода микроконвертеров, использующих регистры сдвигов. Режим 1 – стандартный асинхронный режим, используемый для нормальных последовательных коммуникаций. Структура данных для режима 1 состоит из 10 бит: стартовый, 8 бит данных (LSB – первый) и стоповый бит. Если передача бита четности разрешена (PEN=1), бит дополнения до четности посылается вместо восьмого бита данных. Режимы 2 и 3 – 9-битные режимы общего пользования для межпроцессорных коммуникаций. Структура данных, используемых в этих режимах, состоит из 11 бит: стартовый, 9 бит данных (LSB первый) и стоповый бит. Устройства, работающие в режиме 2, будут вырабатывать прерывание при приеме только тогда, когда девятый информационный бит будет установлен. Устройства, работающие в режиме 3, всегда будут вырабатывать прерывание при приеме. Режим 3 позволяет передавать восемь информационных бит плюс бит дополнения до четности.

Синхронный последовательный интерфейс SPI

Последовательный периферийный интерфейс SPI предназначен для быстрого синхронного обмена информацией между микроконтроллером и периферией или между двумя микроконтроллерами. Кроме того, SPI-пользователи могут использовать его для программирования ПЗУ или отладки программы при использовании отладочного модуля. В расширенном режиме буфер записи SPI хранит следующий передаваемый байт. Это позволяет передать несколько байт при минимальных задержках между передачами байт, при условии, что ЦПУ имеет возможность поддерживать буфер записи заполненным.

Обмен информацией может быть полностью дуплексным и осуществляться с использованием трех линий. Максимальная частота, на которой происходит передача, равна четверти значения частоты генератора (для 33 МГц частоты микроконтроллера – 10 МГц). Завершение передачи сопровождается установкой в «1» соответствующего флага.

Основные особенности SPI:

- полнодуплексная, трехпроводная синхронная передача данных;
- работа в режиме Master или Slave;
- максимальная частота передачи равна f/4;

- передача данных либо по принципу «младший бит вперед», либо «старший бит вперед»;

- четыре программируемых скорости передачи данных в режиме задающего устройства (Master Mode);

- флаг прерывания окончания передачи.

Интерфейс I2C

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

Функциональные возможности модуля:

- совместимость с SMBus (Version 1.1 и 2.0), ACCESS.Bus, I2C (Version 2.1);

- поддержка стандартного (Standard), скоростного (F/S) и высокоскоростного (Hs) режимов;

- программирование действий Master или Slave;

- возможность подключения к шине нескольких ведущих устройств (режим Multi-Master);

- один определяемый программно 7/10-битный адрес для Slave;

- возможность одновременного обращения ко всем устройствам шины, так называемый «общий вызов» (global call).

Особые возможности SMBus:

- отслеживание времени ожидания линии SCL;

- наличие функции PEC (Packet Error Checking – отслеживание ошибок в пакетах данных) с использованием стандартного полинома SMBus;

- возможность обращения одного или нескольких Slave к Master с получением отклика от последнего;

- возможность последовательного опроса;

- функционирование под управлением прерываний.

Блок ШИМ (PWM)

Встроенный ШИМ предназначен для генерации широтно-модулированного сигнала на выходе микросхемы без участия процессора и имеет три выхода PWM.

Скважность импульса на выходе PWM является переменной величиной, импульсы повторяются каждые 256 или 512 тактов. PWM может использоваться в различных применениях. Различные типы моторов требуют использование PWM – формирователя импульсов для наиболее эффективной работы. Кроме того, фильтрация этой ШИМ последовательности будет создавать постоянный уровень, который изменяется с изменением скважности.

Аналого-цифровые преобразователи АЦП

Восемь АЦП конвертируют аналоговые напряжения на входе в цифровой эквивалент. Разрешающая способность – 16 бит с программируемым временем преобразованиями. Любой преобразователь может начать преобразование немедленно, или HSO может инициализировать преобразование всех каналов в запрограммированное время. При завершении каждого преобразования АЦП вырабатывает прерывание. МК 1874BE96T имеет отдельные выводы питания ∩VCC2 и ∩0V2, что исключает влияние помех по линиям питания на аналого-цифровое преобразование. Имеется возможность выдачи опорного напряжения на вывод P0.7. Для уменьшения действия синфазных помех возможно дифференциальное включение входов, при этом максимальное количество одновременно работающих каналов равно четырем. Для каждого из каналов существует возможность организации режима цифрового компаратора, при котором выдача прерываний происходит при выходе результата преобразования за заданные границы.

Цифро-аналоговый преобразователь ЦАП

ЦАП конвертирует цифровой код в аналоговый эквивалент. Разрешающая способность – 14 бит. После записи кода в управляющий регистр на выходах ИС появляется ток, соответствующий этому коду. Возможен перевод ЦАП в режим пониженного энергопотребления. Микроконвертеры имеют отдельный вход FSADJ для подстройки тока преобразователя.

Сервер периферийных транзакций PTS

Этот функциональный блок предназначен для аппаратной обработки прерываний. Он содержит набор встроенных алгоритмов, исходные данные для которых должны быть размещены программой пользователя как в памяти кристалла, так и во внешней памяти. Алгоритмы PTS охватывают, в основном, пересылки данных. Прерывания, обслуживаемые PTS, отрабатываются параллельно работе ЦПУ в моменты простоя шины данных, тем самым не мешая выполнению основной программы. Пересылка может осуществляться и в режиме IDLE.

Этот блок предназначен для обеспечения некоторой последовательности событий в заданное время без участия процессора. Заданные события выполняются на микропрограммном уровне. Такими блоками событий могут быть:

- передача блока информации из одного места памяти (или устройства ВВ) в другое;
- последовательный опрос нескольких каналов АЦП;
- загрузка данных в блок HSO;
- выгрузка данных из блока HSI.

Сервер периферийных транзакций PTS поддерживает обработку запросов на прерывание на микропрограммном уровне, не требуя вмешательства процессора.

Модуль отладки (OCDS)

Этот функциональный блок предназначен для упрощения поцесса отладки программного обеспечения. По выбранному типу события (требуемые данные на шине данных, требуемый адрес операнда на шине адреса, значение счетчика команд, выход в область внешнего ПЗУ) этот блок генерирует слудующие виды действий:

- немаскируемое прерывание;
- режим IDLE (режим ожидания, выход по прерыванию или сбросу);
- аппаратный сброс.

Аппаратный сброс вместе с настройкой контроля выхода в область внешнего ПЗУ может использоваться для контроля выполнения программ и защиты от несанкционированного доступа к внешней памяти.

Энергосберегающие режимы работы

Реализованы режимы работы с пониженным энергопотреблением.

Режим IDLE – когда работают только встроенные функциональные устройства, а микроконвертер находится в режиме ожидания разрешенного прерывания от внешнего или внутреннего устройства.

Режим POWERDOWN – вся внутренняя синхронизация замораживается в состоянии логического нуля, и генератор отключается. Внутреннее ОЗУ и большинство периферийной памяти сохраняет своё значение, если сохраняется напряжение питания. В этом случае ток потребления – всего несколько микроампер.

Режим SLOW. В этом режиме замедляется работа как ЦПУ, так и периферийных устройств за счет увеличения длительности машинного цикла в два раза. Может включаться/выключаться программно без сброса микроконвертера.

Помимо этих трех режимов существует возможность независимого отключения/включения тактового сигнала периферийных устройств (смотреть раздел 9). Это позволяет гибко управлять общим потреблением, используя наиболее оптимальный режим в каждый момент времени работы системы.

5 Типы данных и адресация

В данном разделе приведены типы операндов и способы адресации, обеспечиваемые архитектурой микроконвертера 1874ВЕ96Т.

5.1 Типы операндов

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

Типы переменных операндов указаны заглавными буквами во избежание путаницы. Например, BYTE – 8-битная беззнаковая переменная в инструкции, в то время как byte – любая 8-битная единица данных (со знаком или без знака).

В таблице 5.1 даётся общий обзор этих типов операндов. Далее в этом разделе детально рассматривается каждый из них.

Таблица 5.1 – Определения типа операнда

Тип операнда	Число бит	Знак	Возможные значения	Ограничения адресации
BIT	1	нет	TRUE (1) или FALSE (0)	как компоненты байтов
BYTE	8	нет	от 0 до (2^8–1) (от 0 до 255)	нет
SHORT INTEGER	8	да	от $–2^7$ до (2^7–1) (от –128 до 127)	нет
WORD	16	нет	от 0 до (2^{16}–1) (от 0 до 65 535)	адрес четного байта
INTEGER	16	да	от $–2^{15}$ до (2^{15}–1) (от –32 768 до 32 767)	адрес четного байта
DOUBLE WORD [1)]	32	нет	от 0 до (2^{32}–1) (от 0 до 4 294 967 295)	адрес в младшем регистровом файле, кратный четырем [2)]
LONG INTEGER [1)]	32	да	от $–2^{31}$ до (2^{31}–1) (от –2 147 483 648 до 2 147 483 647)	адрес в младшем регистровом файле, кратный четырем [2)]

_____

[1)] 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.

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

Таблица 5.2 – Эквивалентные типы операндов для ассемблера и языка «Си»

Тип операнда	Эквивалент ассемблера	Эквивалент на языке «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

Операнды BIT

BIT – одноразрядная переменная, которая может иметь булевы значения «true» и «false». Архитектура требует, чтобы к BIT обращались как к компонентам BYTE или WORD, так как прямая адресация BIT не поддерживается.

Операнды BYTE

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

Операнды SHORT INTEGER

SHORT INTEGER – 8-битная переменная со знаком, которая может принимать значения от минус 128 (–2^7) до 127 (2^7–1). Арифметические действия, которые производят результаты вне диапазона SHORT INTEGER, устанавливают флаги переполнения в слове состояния процессора (PSW). Числовой результат тот же самый, что и результат эквивалентного действия с переменной BYTE. Нет никаких ограничений выравнивания для SHORT INTEGER, так что они могут быть помещены по любому адресу памяти.

Операнды WORD

WORD – 16-битная беззнаковая переменная, которая может иметь значения от 0 до 6 535 (2^{16}–1). Арифметические и сравнение операторы могут быть применены к операндам WORD, но результат должен интерпретироваться по модулю 65 536 арифметически. Логические действия с WORD осуществляются побитно. Биты в пределах WORD помечены от 0 до 15, бит 0 – младшего значения бит.

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

Операнды INTEGER

INTEGER – 16-битная переменная со знаком, которая может принимать значения от минус 32 768 (–2^{15}) до 32 767 (2^{15}–1). Арифметические действия, которые производят результаты вне диапазона INTEGER, устанавливают флаги переполнения в слове состояния процессора (PSW). Числовой результат тот же самый, как результат эквивалентного действия на переменных WORD.

INTEGER должен быть выровнен по границе четного байта в адресном пространстве. Младший байт INTEGER находится по четному адресу байта, а старший байт находится в следующем выше нечетном адресе. Адрес INTEGER – адрес его младшего байта. Действия INTEGER по нечетным адресам не гарантируются.

Операнды DOUBLE WORD

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

Действия DOUBLE WORD, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя операндами WORD. 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
```

Операнды LONG INTEGER

LONG INTEGER – 32-битная переменная со знаком, которая может принимать значения от минус 2 147 483 648 (-2^{31}) до 2 147 483 647 ($2^{31}-1$). Архитектура непосредственно поддерживает операнды LONG INTEGER только как операнды в командах сдвига, как делимое при делении 32/16 и как произведение при умножении 16×16. Для этих действий переменная LONG INTEGER должна быть в младшем файле регистров и должна быть выровнена по адресу, который является кратным четырем. Адрес LONG INTEGER – это его младший байт (адрес четного байта).

Инструкции с LONG INTEGER, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя операндами INTEGER. См. пример в «Операнды DOUBLE WORD».

5.2 Режимы адресации

Косвенная адресация осуществляет доступ к операнду, адрес которого размещен в переменной типа WORD регистрового файла. Вычисляемый адрес должен соответствовать правилам выравнивания. Следует заметить, что косвенная адресация позволяет обращаться к операнду в любом месте адресного пространства MCS-96, включая регистровый файл. 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 косвенной адресации:

```
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
```

Рабочие регистры:

AX, CX – 16-и битные регистры.

AL, SL – младшие байты регистров AX, BX.

5.2.1 Косвенная адресация с автоинкрементом

Косвенная адресация с автоинкрементом – это тоже самое, что и косвенная адресация, но в дополнение переменная типа WORD, содержащая косвенный адрес, инкрементируется после использования для адресации операнда. Младший значащий бит регистра типа WORD по-разному интерпретируется в зависимости от наличия или отсутствия автоинкремента в косвенной адресации. Если команда работает с переменными типа BYTE или SHORT INTEGER, адрес инкрементируется на 1, а если команда работает с переменными типа WORD или INTEGER, то адрес инкрементируется на 2.

Пример косвенной адресации с автоинкрементом:

```
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
```

Рабочие регистры:

AX, BX, CX – 16-битные регистры;

AL, BL – младшие байты регистров AX, BX.

5.2.2 Непосредственная адресация

Непосредственная адресация позволяет брать операнд непосредственно из поля команды. Для операций с переменными типа BYTE или SHORT INTEGER это будет

8-разрядное поле. Для операций с переменными типа WORD или INTEGER – 16-разрядное поле. Команда может содержать только один непосредственный операнд; для других операндов необходимо использовать прямую регистровую адресацию.

Пример непосредственной адресации:

```
ADD AX, #340              ; AX ← AX + 340 (ADD)
PUSH #1234H               ; SP ← SP - 2
                          ; MEM_WORD(SP) ← 1234H
DIVB AX, #10              ; AL ← AX/10
                          ; AH ← AX MOD 10
```

Рабочие регистры:

AX – 16-битный регистр.

AL – младший байт регистра AX.

AH – старший байт регистра AX.

5.2.3 Короткая индексная адресация

При короткой индексной адресации адрес одного из операндов вычисляется с помощью двух 8-битных полей. Одно 8-битное поле выбирает переменную типа WORD в регистровом файле, которая содержит адрес. Второе 8-битное поле в команде имеет знаковое расширение и суммируется с переменной типа WORD для формирования исполнительного адреса операнда. Исполнительный адрес может быть на 128 байт впереди от текущего адреса переменной типа WORD и на 127 байт после него. Команда может содержать только один такой операнд, другие операнды задаются с помощью прямой регистровой адресации.

Пример короткой индексной адресации:

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

Рабочие регистры:

AX, BX, CX – 16-битные регистры.

BL – младший байт регистра BX.

5.2.4 Длинная индексная адресация

Длинная индексная адресация похожа на короткую индексную адресацию за исключением того, что 16-битное поле берётся из команды и добавляется к переменной типа WORD для формирования адреса операнда. Нет необходимости в знаковом расширении. Такая адресация может применяться только для одного операнда в команде, для других операндов необходимо использовать прямую регистровую адресацию.

Пример длинной индексной адресации:

```
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)
```

Рабочие регистры:

AX, BX, CX – 16-битные регистры.

AL, BL – младшие байты регистров AX, BX.

5.2.5 Адресация с использованием нулевого регистра

Два первых байта в регистровом файле образуют нулевой регистр (ZERO_REG). Эти байты всегда равны нулю. Кроме источника фиксированного нуля для расчётов и сравнения, нулевой регистр может использоваться как переменная типа WORD в длинной индексной адресации. Эта комбинация способа адресации и регистра позволяет адресоваться к любому участку памяти. Так как этот способ использует индексную адресацию, то доступ осуществляется медленнее, чем при прямой регистровой адресации.

Пример адресации с использованием нулевого регистра:

```
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
```

Рабочие регистры:

AX – 16-битный регистр.

5.2.6 Адресация с использованием указателя стека

Байты 18H и 19H нижнего регистрового файла содержат указатель стека SP, к которому адресуются по адресу 18H. Кроме обычных операций указатель стека SP может также использоваться как переменная типа WORD в косвенной адресации для доступа к вершине стека или в короткой индексной адресации для доступа к данным внутри стека.

Пример адресации с использованием указателя стека:

```
PUSH [SP]                   ; Duplicate TOP_OF_STACK
                            ; (скопировать вершину стека)
LD      AX, 2[SP]           ; AX ← NEXT_TO_TOP
```

Рабочие регистры:

AX – 16-битный регистр.

5.3 Выборы способов адресации на языке ассемблера

Ассемблер упрощает выбор режимов адресации. Это упрощает задачу программирования, и эти возможности ассемблера следует применять там, где только это возможно.

5.3.1 Прямая адресация

Ассемблер выбирает между прямой регистровой адресацией и адресацией с использованием нулевого регистра, в зависимости от местоположения операнда в памяти. Необходимо обратиться к операнду с его символическим именем. Если операнд находится в младшем файле регистров, ассемблер выбирает прямую адресацию. Если операнд находится в другой области памяти, ассемблер выбирает адресацию с нулевым индексом.

5.3.2 Индексная адресация

Ассемблер выбирает между короткой индексной и длинной индексной адресациями в зависимости от значения индекса. Если значение может быть выражено в восьми битах, язык ассемблера выбирает короткую индексную адресацию. Если значение больше 8-битного, выбирается длинная индексная адресация.

5.4 Стандарты и соглашения программного обеспечения

При разработке программного обеспечения рекомендуется, чтобы использовались соглашения, принятые для редактирования на основе языка «Си». Эти стандарты применимы и для ассемблера, и для программной среды языка «Си», и они обеспечивают совместимость между этими средами.

5.4.1 Использование регистров

В 256-байтном младшем файле регистров содержатся регистры специальных функций центрального процессора и указатель стека. Остаток младшего файла регистров и старший файл регистров доступны для использования. Периферийные регистры специальных функций (SFRs) и распределенные в карте памяти SFR расположены в верхней области адресного пространства. Периферийные SFR могут быть перемещены через «окна» в младший файл регистров для прямого доступа. Распределенные в карте памяти SFR не могут быть использованы в режиме «окон». Можно использовать косвенную или индексную адресацию, чтобы получить доступ к ним. Все SFR оперируют как BYTE или WORD, если не определено иначе.

Язык программирования «Си» адаптирован к простой эффективной стратегии, размещает восемь байтов, начинающихся в адресе 1CH, во временное хранение и

рассматривает оставшуюся область в файле регистров как сегмент запоминающего устройства, размещённый там, где необходимо.

Примечание – Использование любого SFR как базового или индексного регистра для косвенных или индексных инструкций может вызвать непредсказуемые результаты, потому что внешние события могут изменить содержимое SFRs. Так как некоторые SFRs очищаются при чтении, допустимо использование SFR как операнда в инструкциях «чтение-модификация-запись» (например, XORB).

5.4.2 Адресация 32-битных операндов

32-битные операнды (DOUBLE WORD и LONG INTEGER) сформированы двумя смежными 16-битными словами в запоминающем устройстве. Младшее слово DOUBLE WORD находится всегда в младшем адресе, даже когда данные находятся в стеке (это означает, что старшее слово должно быть выдвинуто в стек первым). Адрес 32-битного операнда – адрес его младшего байта.

Аппаратные средства поддерживают 32-разрядные типы данных как операнды в инструкциях сдвига, делимое в делении 32/16 и произведение в умножении 16×16. Для этих действий 32-битный операнд должен располагаться в младшем файле регистров и должен быть выровнен по адресу, кратному четырем.

5.4.3 Соединение подпроцедур

Параметры передаются к подпроцедурам через стек. Параметры выдвигаются в стек от самого правого параметра налево. Параметры на восемь битов выдвигаются в стек со старшим неопределенным байтом. 32-битные параметры выдвигаются в стек как два 16-битных; старшая часть параметра выдвигается в стек первой. Как пример рассмотрим следующую процедуру:

Void example_procedure (char param1, long param2, int param3)

Когда эта процедура выполняется, стек будет содержать параметры в следующем порядке:

param3
low word of param2
high word of param2
undefined; param1
return address ← Stack Pointer

Если процедура возвращает значение к коду запроса (в противоположность изменению более глобальных переменных), результат возвращается в область временного хранения (TMPREG0 в этом примере), начинающуюся в 1CH. TMPREG0 рассматривается или как 8-, 16- или 32-битная переменная, в зависимости от типа процедуры.

Стандарт вызова, принятый языком «Си», имеет несколько следующих ключевых особенностей.

Процедуры могут всегда предполагать, что восемь байтов файла регистров, начинающиеся в 1CH, могут использоваться для временного хранения в пределах тела процедуры.

Код, который вызывает процедуру, должен учитывать, что процедура изменяет восемь байтов файла регистров, начинающихся в 1CH.

Код, который вызывает процедуру, должен предполагать, что процедура изменяет слово состояния процессора PSW, потому что процедуры не сохраняют и не восстанавливают PSW.

Результат процедур всегда возвращается в переменную TMPREG0.

Язык «Си» позволяет определять прерывание процедур, которые выполняются. Прерывание процедур не соответствует правилам нормальных процедур. Параметры нельзя передать к этим процедурам, и они не могут возвратить результаты. Так как прерывание процедуры можно выполнить по существу в любое время, они должны сохранить и восстанавливать значения PSW и TMPREG0.

5.5 Защита и руководящие принципы программного обеспечения

В микроконвертере реализовано несколько механизмов восстановления после ошибок программного обеспечения и аппаратных средств. Прерывание по невыполнимому коду инструкции обеспечивает защиту от выполнения некорректного кода инструкции. Команда аппаратного сброса RST может вызвать сброс, если программный счетчик выходит за границы. Код инструкции RST – FFH, поэтому процессор сбросит себя, если попробует выполнить инструкцию от незапрограммированных ячеек в энергонезависимом запоминающем устройстве или из шины, линии которой подключены к высокому уровню. Сторожевой таймер WDT может сбросить микроконвертер в случае аппаратной или программной ошибки. Блок отладки может инициализировать сброс в случае выхода в недопустимые области памяти.

Рекомендуется заполнить неиспользованные области кодом с NOP и периодическими переходами к процедуре обслуживания ошибок или RST инструкции. Это особенно важно для кодов окружения таблиц поиска. Везде, где место позволяет, необходимо окружить каждую таблицу семью NOP (потому что самая длинная инструкция устройства имеет семь байтов) и RST или переходами к процедуре обслуживания ошибок. Так как RST – однобайтовая инструкция, NOP не нужны, если RST используются вместо переходов к процедуре ошибки.

При использовании сторожевого таймера WDT для защиты программного обеспечения рекомендуется сбрасывать WDT только в одном месте программы, сокращая возможность нежелательного сброса WDT. Секция кода, который сбрасывает WDT, должна контролировать другие секции кода для правильного выполнения инструкций. Это может быть сделано проверкой переменных, чтобы удостовериться, что они – в пределах разумных значений.

6 Распределение памяти

В этом разделе описано адресное пространство памяти микроконвертера. Изделие имеет 64 Кбайт адресного пространства, большая часть которого предназначена для программ или запоминания данных. ИС имеет общее пространство для адресов и команд, но выполнение команд из некоторых областей памяти ограничено. Так невозможно выполнение команд из областей регистрового файла и расширенного ОЗУ. При попытке чтения команд из этих адресов будет происходить обращение к внешней памяти.

На рисунке 6.1 показано разделение памяти для данных.

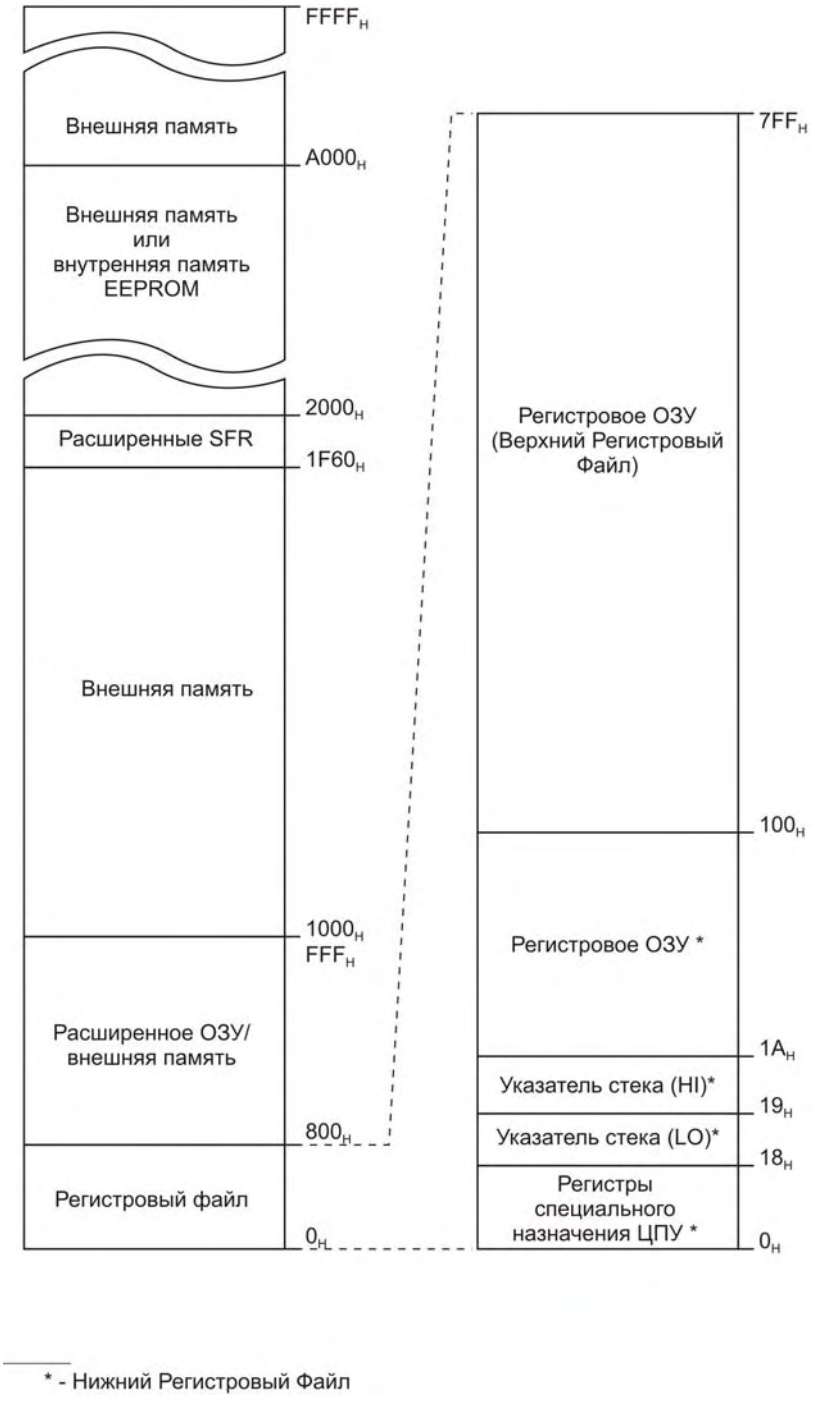


Рисунок 6.1 – Карта адресов памяти для данных

Распределение памяти для команд показано на рисунке 6.2.

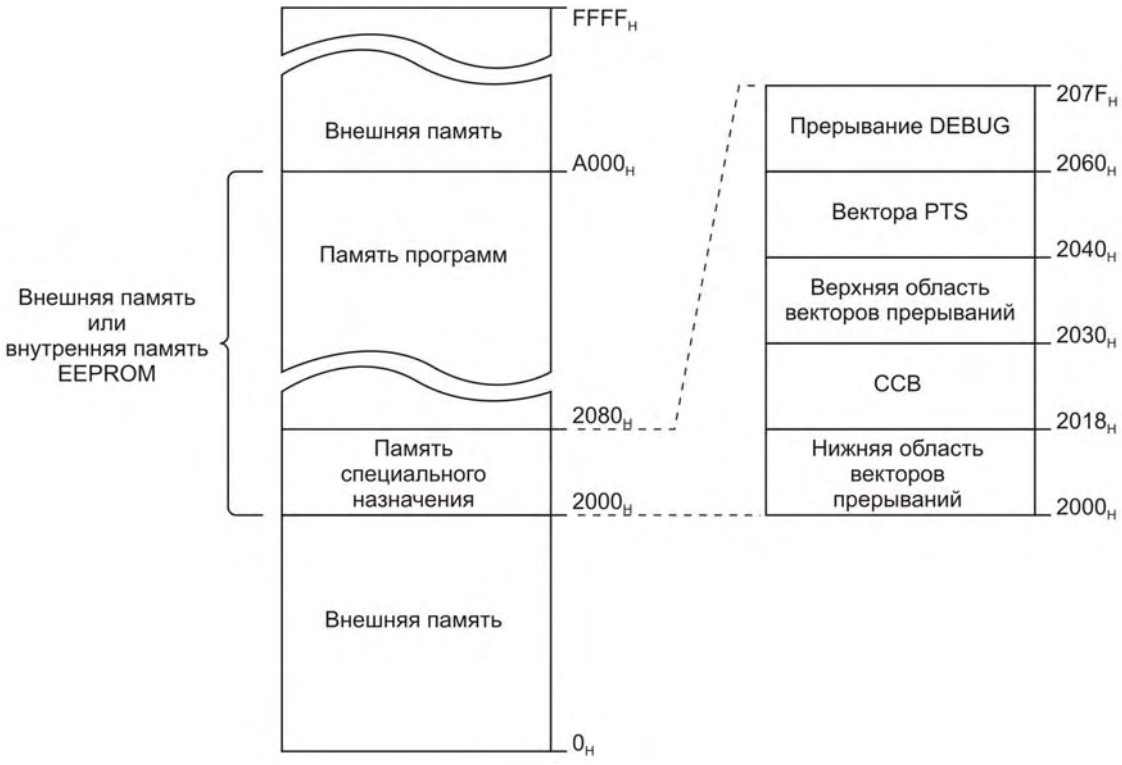


Рисунок 6.2 – Карта адресов памяти для команд

В таблице 6.1 приводятся начальные и конечные адреса памяти при обращении к данным в шестнадцатеричной и десятичной системах исчисления, в таблице 6.2 – при обращении к командам. Команды могут располагаться либо в области внутренней EEPROM, либо в области внешней памяти. В последующих подразделах описывается каждый раздел памяти.

Таблица 6.1 – Старшие адреса памяти при обращении к данным данных МК

Обозначение	Диапазон адресов	
	Шестнадцатеричные	Десятичные
Внешняя память	0A000H – 0FFFFH	40960 – 65635
Внешняя или внутренняя память EEPROM	2000H – 9FFFH	8192 – 40959
Расширенные SFR	1F70H – 1FFFH	8048 – 8191
Внешняя память	1000H – 1F6FH	4098 – 8047
Расширенное ОЗУ/внешняя память	800H – FFFH	2048 – 4097
Регистровый файл (включая SFR)	0H – 7FFH	0 – 2047

Таблица 6.2 – Старшие адреса памяти при обращении к командам МК

Обозначение	Диапазон адресов	
	Шестнадцатеричные	Десятичные
Внешняя память	0A000H – 0FFFFH	40960 – 65635
Программная память *	2080H – 9FFFH	8320 – 40959
Память специального назначения *	2000H – 207FH	8192 – 8319
Внешняя память	0H – 1FFFH	0 – 8191

* Может находиться как в области внутренней EEPROM, так и во внешней памяти.

6.1 Разделы внешней памяти для данных

Область выше расширенных SFR или внутренней EEPROM и область выше расширенного ОЗУ или регистрового файла всегда назначается на внешнюю память данных. Обращение к ней осуществляется через шину внешних адресов/данных. Разделение памяти данных и памяти команд осуществляется сигналом INST.

6.2 Расширенные SFR

Область расширенных SFR содержит регистры управления программированием ПЗУ, тактированием периферийных блоков, расширенного управления АЦП, отладочным модулем и порты 3 и 4. Порты 3 и 4 читаются и записываются как слово по адресу 1FFEH (порт 3 – младший байт; порт 4 – старший байт). Эти порты работают либо как порты ввода-вывода, либо как мультиплексированная шина адрес/данные, либо их комбинация, в зависимости от EA# сигнала при сбросе.

6.3 Расширенное ОЗУ

Область расширенного ОЗУ предназначена для хранения больших массивов данных, доступ к которым осуществляется косвенными методами адресации. Обращение к этой области занимает больше машинных циклов, чем обращение к регистровому файлу прямым методом, но намного меньше чем к внешней ОЗУ. Расширенное ОЗУ имеет функцию отключения (смотреть раздел 9). При этом область 800H – FFFH адресуется в область внешней памяти.

6.4 Программная память и память специального назначения

Программная память и память специального назначения занимают раздел памяти 32 Кбайт (рисунок 6.2 и таблица 6.2), который может постоянно располагаться либо во внутренней EEPROM либо во внешней памяти. Эта область является общей для команд и данных в случае работы ИС в режиме включенного ПЗУ, что позволяет программировать ПЗУ путем записи нужных данных в нужные ячейки.

6.4.1 Выбор внутренней или внешней памяти

Доступ к адресам программной памяти и адресам памяти специального назначения осуществляется в зависимости от того, где находится данный раздел: во внутреннем EEPROM или во внешней памяти, причём он не может находиться там и там. Значение вывода внешнего доступа EA# во время переднего фронта сигнала RESET# определяет вид доступа: внутренний или внешний. Это значение запоминается в устройстве и не может быть изменено до сброса устройства. Если EA# имеет низкий уровень, то внутреннее EEPROM недоступно, и все обращения к этому адресному диапазону направляются прямо во внешнюю память. Если EA# имеет высокий уровень, то обращение к этому адресному диапазону осуществляется во внутреннее EEPROM. Время включения EEPROM составляет 5 мкс, поэтому необходимо обеспечить соответствующую длительность сигнала RESET при первом использовании режима работы с внешней памятью или переключении в этот режим. Время выполнения программы из внутренней памяти равно времени выполнения программы из внешней памяти, имеющей 16-битный доступ с одним дополнительным циклом ожидания.

Примечание – Вывод EA# должен иметь высокий уровень для надежного функционирования ЦПУ без внешней памяти.

6.4.2 Программная память

Программная память располагается по адресам 2080H – 9FFFH. Существует возможность запуска программы с различных начальных адресов.

Стандартный режим работы. Начальный адрес – 2080H.

Режим старта из области внешней памяти 0A000H. Для этого надо удерживать сигналы VPR и ALE в низком уровне во время сброса. Может использоваться только при работе в режиме внутренней памяти команд.

Режим старта с начального адреса 9000H. Для этого надо удерживать VPR в низком уровне, а ALE в высоком во время сброса. Если выбран режим работы с внутренней

памятью, то начнется выполнение программы – монитора, реализующей функции программирования ПЗУ.

Примечание – Так как исходное значение ячеек внутренней памяти EEPROM равно 00H, то рекомендуется записывать FFH в неиспользуемые ячейки программной памяти, так как при неправильном ходе выполнения программы это приведет к перезагрузке схемы.

6.4.3 Память специального назначения

Память специального назначения расположена по адресам 2000H – 207FH (см. рисунок 6.2). Она содержит несколько зарезервированных ячеек памяти, векторы для периферийного сервера PTS и стандартных прерываний. В таблице 6.3 приведен список адресов памяти специального назначения а также начальные и конечные адреса в шестнадцатеричной и десятичной системах.

Векторы прерываний

Верхняя и нижняя области векторов прерываний содержат адреса подпрограмм обслуживания прерываний. PTS-векторы содержат адреса управляющих блоков PTS.

Таблица 6.3 – Адреса памяти специального назначения

Обозначение	Адресный диапазон	
	Шестнадцатеричные	Десятичные
Общего назначения	2062H – 207FH	8290 – 8319
Вектор прерывания DEBUG	2060H	8288
Общего назначения	205EH – 205FH	8286 – 8287
Векторы PTS	2040H – 205DH	8256 – 8285
Старшие векторы прерывания	2030H – 203FH	8240 – 8255
Общего назначения	2019H – 202FH	8217 – 8239
CCB	2018H	8216
Общего назначения	2014H – 2017H	8212 – 8215
Младшие векторы прерывания	2000H – 2013H	8192 – 8211

Байт конфигурации кристалла CCB

Байт конфигурации кристалла CCB – первый байт, выбранный из памяти после сброса устройства. Последовательность сброса загружает CCB во внутренний регистр, называемый регистром конфигурации кристалла (CCR). CCR управляет защитой внутренней памяти, режимом READY, сигналом управления шиной, разрядностью шины и режимом POWERDOWN (режимом пониженного энергопотребления).

6.5 Регистровый файл

Регистровый файл разделён на верхний и нижний регистровые файлы (см. рисунок 6.1). Верхний регистровый файл содержит регистровое ОЗУ общего назначения. Нижний регистровый файл содержит регистровое ОЗУ общего назначения, указатель стека (SP) и регистры специальных функций (SFR). В таблице 6.4 приведены начальные и конечные адреса в шестнадцатеричной и десятичной системах.

Таблица 6.4 – Адреса памяти регистрового файла

Обозначение	Адресный диапазон	
	Шестнадцатеричные	Десятичные
Старший регистровый файл *	100H – 7FFH	256 – 2047
Регистровое RAM (ОЗУ) общего назначения **	1AH – 0FFH	26 – 255
Указатель стека SP **	18H – 19H	24 – 25
Регистры специального назначения SFR **	0H – 17H	0 – 23

_____

* Содержит регистровое RAM (ОЗУ) общего назначения (доступно с помощью косвенной или индексной адресации, если только нет использования вертикальных окон).

** Расположены в нижнем регистровом файле (доступны с помощью прямой регистровой, косвенной или индексной адресации).

6.5.1 Регистровое ОЗУ общего назначения

Ячейки 1AH – 0FFH содержат регистровое RAM (ОЗУ) общего назначения. Ячейки указателя стека 18H – 19H могут также использоваться в качестве регистрового ОЗУ общего назначения, когда операции со стеком не проводятся. РАЛУ может прямо обращаться к этой памяти, используя прямую регистровую адресацию. Верхний Регистровый Файл также содержит регистровое ОЗУ общего назначения (100H – 7FFH). Обычно РАЛУ обращается к этой памяти, используя косвенную или индексную адресацию. Технология Vertical windowing (создание вертикальных окон) позволяет РАЛУ использовать прямую регистровую адресацию для доступа к ОЗУ в верхнем регистровом файле (см. раздел 3, где описаны различия между прямой регистровой и индексной адресациями). Создание вертикальных окон обеспечивает быстрое переключение на задачи по прерываниям и ускоряет выполнение программ (см. подраздел 6.7).

6.5.2 Указатель стека SP

Ячейки памяти 18H и 19H содержат указатель стека SP. SP содержит адрес вершины стека. SP должен указывать на адрес слова (четный), который на два байта больше, чем желаемый стартовый адрес. Перед тем, как ЦПУ выполнит вызов подпрограммы или программы обслуживания прерывания, оно дважды декрементирует содержимое SP и копирует (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 регистрового ОЗУ общего назначения.

Инициализация указателя стека

Пользователь программ должен загрузить двухбайтный адрес в указатель стека. Выбор адреса, на два байта большего, чем желаемый стартовый адрес, делается потому, что указатель стека автоматически декрементируется перед тем, как ЦПУ засылает первый байт адреса возврата в стек. Следует помнить, что стек растёт вниз, что требует достаточного количества ячеек для максимального числа адресов, занесённых в стек. Стек может находиться во внутреннем Регистровом Файле, расширенном ОЗУ, либо во внешнем ОЗУ.

6.5.3 Регистры специальных функций SFRs

Ячейки 00H – 17H обеспечивают доступ к регистрам специальных функций SFR ЦПУ через три горизонтальных окна (HWindow 0, 1 и 15, смотри подраздел 6.5.). РАЛУ осуществляет прямое управление всеми периферийными модулями, кроме портов 3 и 4, через SFR.

При использовании SFR в качестве базового или индексного регистра при косвенной или индексной адресациях следует отдавать отчёт в том, 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 SFR; надо записать в них 0 или оставить их в исходном состоянии. При считывании зарезервированные биты и SFR могут возвращать неопределённые значения.

6.6 Создание горизонтальных окон

Микроконвертреры используют технологию horizontal windowing (создания горизонтальных окон), которая переключает три 24-байтных блока памяти (HWindow 0, 1 и 15) внутри младших 24 байтов в нижнем регистровом файле (рисунок 6.3). Каждое HWindow (горизонтальное окно) обеспечивает чтение или запись в определенные SFR. Некоторые регистры доступны как отдельный байт; другие – как слово (два байта). Некоторые регистры, такие как регистр выбора окна (WSR, 14H), доступны во всех трех окнах, другие могут быть доступны для записи в одном окне, а для считывания – в другом.

Информация о регистрах приведена в приложении В.

Рисунок 6.3 – Создание горизонтальных окон

6.6.1 Выбор горизонтального окна

Регистр выбора окна (WSR,14H) обеспечивает доступ к горизонтальным и вертикальным окнам (подраздел 6.7). Для выбора HWindow необходимо занести номер желаемого окна в WSR.0 – WSR.3 и очистить WSR.4 – WSR.6. Доступны только окна HWindow 0, 1 и 15. Все другие окна HWindow зарезервированы. В таблице 6.5 показано необходимое содержимое WSR для выбора каждого HWindow.

Таблица 6.5 – Выбор горизонтальных окон

HWindow	Содержимое WSR
0	X000 0000B = 00H
1	X000 0001B = 01H
15	X000 1111B = 0FH

6.6.2 Горизонтальное окно 0 (HWindow 0)

HWindow 0 – начальное окно. Оно обеспечивает доступ к чтению из 19 регистров и записи в 21 регистр (таблица 6.6). Некоторые регистры (например, INT_MASK1)

доступны для чтения и для записи внутри HWindow 0. Другие (например, IOS1) могут быть доступны или только для чтения, или только для записи внутри HWindow 0. Для проведения недостающих функций в этих регистрах следует выбирать HWindow 15.

6.6.3 Горизонтальное окно 1 (HWindow 1)

HWindow 1 обеспечивает доступ для чтения и для записи в 12 регистров (таблица 6.6). Некоторые регистры также доступны в двух окнах – HWindow 0 и HWindow 15.

Таблица 6.6 – окна HWindow 0 и HWindow 1

Адрес регистра в окне	HWindow 0 чтение	HWindow 0 запись	HWindow 1 чтение/запись
17H	IOS2		
16H	IOS1	IOC1	
15H	IOS0	IOC0	Зарезервировано
14H	WSR	WSR	WSR
13H			
12H			
11H			DACVAL(HI)
10H	IOPORT2	IOPORT2	DACVAL(LO)
0FH	IOPORT1	IOPORT1	Зарезервировано
0EH	IOPORT0		Зарезервировано
0DH	TIMER2(HI)	TIMER2(HI)	Зарезервировано
0CH	TIMER2(LO)	TIMER2(LO)	IOC3
0BH	TIMER1(HI)	IOC2	Зарезервировано
0AH	TIMER1(LO)	WATCHDOG	Зарезервировано
09H			
08H			
07H	SBUF(RX)	SBUF(TX)	PTSSRV(HI)
06H			PTSSRV(LO)
05H			PTSSEL (HI)
04H			PTSSEL (LO)
03H			
02H			Зарезервировано
01H			
00H			

6.6.4 Горизонтальное окно 15 (HWindow 15)

HWindow 15 обеспечивает доступ к тем же регистрам, что и HWindow 0, кроме байтов 0CH – 10H. Через байты 0CH – 0DH в HWindow 0 осуществляется доступ к 16-разрядному регистру TIMER2, а в HWindow 15 – к 16-разрядному регистру T2CAPTURE. Через байты 0EH – 10H в HWindow 0 осуществляется доступ к регистрам IOPORT0, IOPORT1, IOPORT2, а в HWindow 15 они зарезервированы. Регистры, доступные только для чтения в HWindow 0, доступны только для записи в HWindow 15 и наоборот (таблица 6.7).

Таблица 6.7 – Окно HWindow 15

Адрес регистра в окне	HWindow 15 чтение	HWindow 15 запись
17H	PWM0_CONTROL	IOS2
16H	IOC1	IOS1
15H	IOC0	IOS0
14H	WSR	WSR
13H	INT_MASK1	INT_MASK1
12H	INT_PEND1	INT_PEND1
11H	SP_CON	SP_STAT
10H	Зарезервировано	Зарезервировано
0FH	Зарезервировано	Зарезервировано
0EH	Зарезервировано	Зарезервировано
0DH	T2CAPTURE(HI)	T2CAPTURE(HI)
0CH	T2CAPTURE(LO)	T2CAPTURE(LO)
0BH	IOC2	TIMER1 (HI)
0AH	WATCHDOG	TIMER1 (LO)
09H	INT_PEND	INT_PEND
08H	INT_MASK	INT_MASK
07H	SBUF(TX)	SBUF(RX)
06H	HSO_COMMAND	HSI_STATUS
05H	HSO_TIME (HI)	HSI_TIME (HI)
04H	HSO_TIME (LO)	HSI_TIME (LO)
03H	HSI_MODE	
02H		
01H	ZERO_REG (HI)	ZERO_REG (HI)
00H	ZERO_REG (LO)	ZERO_REG (LO)

6.7 Создание вертикальных окон

Создание вертикальных окон преобразует области верхнего регистрового файла в блоки старших ячеек нижнего регистрового файла, размером в 32, 64 или 128 байт, известные как вертикальные окна – VWindows. Микроконвертер имеет шестьдесят четыре 32-байтных VWindows, тридцать два 64-байтных VWindows и шестнадцать 128-байтных VWindows. Пример 128-байтного окна показан на рисунке 6.4.

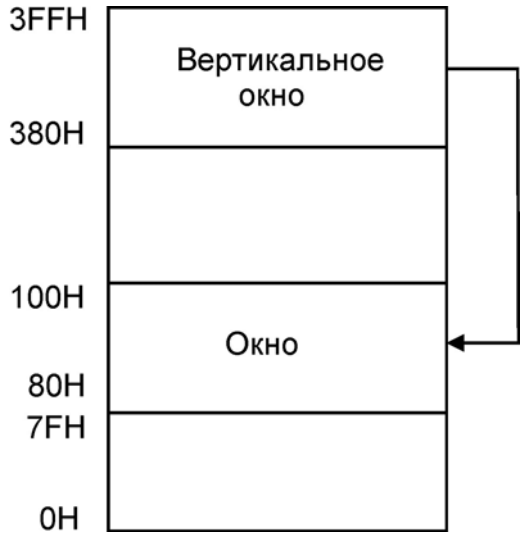


Рисунок 6.4 – Создание вертикальных окон

6.7.1 Выбор вертикального окна

Регистр выбора окна (WSR, 14H) обеспечивает доступ к горизонтальным – HWindow и вертикальным – VWindow окнам. Необходимо установить WSR.4, WSR.5 или WSR.6 для выбора 128-, 64- или 32-байтного VWindows соответственно (рисунок 6.5). Номер VWindows записывается в младшие биты регистра WSR. Для выбора вертикального окна, показанного на рисунке 6.4, нужно загрузить 17H в WSR.

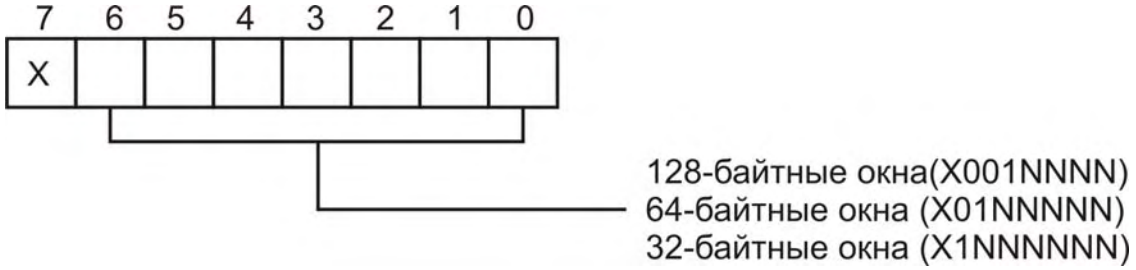


Рисунок 6.5 – Установка битов регистра выбора окна

6.7.2 Создание вертикальных окон и способы адресации

При создании вертикальных окон возможны команды:

- с прямой регистровой адресацией. При этом используется адрес внутри окна в нижнем регистровом файле для получения доступа к ячейкам памяти в верхнем регистровом файле;
- с косвенной или индексной адресацией, использующие адрес внутри окна в нижнем регистровом файле. Они получают доступ к действительной ячейке памяти.

Приведенные ниже команды иллюстрируют различие между прямой регистровой и индексной адресациями при использовании вертикальных окон.

```
PUSHA              ; пересылка содержимого WSR в стек
LDB WSR, #17H      ; выбор VWindow 7, 128-байтный блок

                   ; следующая команда использует прямую регистровую
                   ; адресацию

ADD 40H, 380H      ; слово (40H) ← слово (40H) + слово (380H)

                   ; следующие две команды используют косвенную
                   ; адресацию

ADD 40H, 80H       ; слово (40H)← слово (40H) + слово (80H+0)
ADD 40H, 380H[0]   ; слово (40H)← слово (40H) + слово (380H+0)
POPA               ; перегрузка  в WSR первоначальной информации
```

7 Прерывания

Основной функцией микроконвертера является обеспечение управления устройствами в режиме реального времени. Схема контроллера прерываний позволяет событиям управлять выполнением программы. Когда событие вызывает прерывание, ЦПУ обслуживает это прерывание перед выполнением следующей команды. Встроенная периферия, внешний сигнал или команда могут выставить запрос на прерывание. В простейшем случае МК получает запрос прерывания, осуществляет обслуживание и возвращается к прерванной программе. В данном разделе описываются схема управления прерываниями, схема приоритетов и синхронизация. Также дано описание трех специальных прерываний и объясняется программирование и управление прерываниями.

7.1 Обработка прерываний

В микроконвертере 1874BE96T имеются две возможности обслуживания прерываний: программное обслуживание прерываний через контроллер прерываний и микропрограммное обслуживание прерываний через PTS. Можно выбрать вариант обслуживания прерываний для каждого из маскируемых прерываний (пункт 7.5.1). Немаскируемые прерывания (NMI, прерывание отладочного модуля, программное прерывание, прерывание по неподдерживаемому коду) всегда обрабатываются подпрограммами обработки прерываний. На рисунке 7.1 показан процесс обработки прерываний.

7.1.1 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 памяти (раздел 6), содержат адреса подпрограмм обслуживания прерываний. ЦПУ выполняет подпрограмму обслуживания прерывания. После выполнения этой подпрограммы PC перезагружается значением из стека, и выполнение прерванной программы продолжается.

7.1.2 Сервер периферийных транзакций PTS

PTS является микропрограммным аппаратным обработчиком прерываний. Он может использоваться вместо стандартных подпрограмм обработки прерываний для любых маскируемых прерываний. PTS обслуживает прерывания параллельно работе основной программы, он не модифицирует стек или PSW. Сервер осуществляет передачу данных только во время простоя шины данных. Это могут быть моменты, когда осуществляются арифметические операции, переходы, простои вследствие пустой очереди команд и т.д. Данное поведение отражено на рисунке 7.1.

Рисунок 7.1 – Пример передачи PTS данных

PTS работает в пяти специальных микропрограммных режимах, которые позволяют PTS выполнять отдельные задачи гораздо быстрее, чем подпрограммы обслуживания прерывания (смотри описание режимов PTS в подразделе 7.6).

Каждое прерывание PTS требует блока данных, называемого блоком управления PTS (PTSCB). Если имеет место прерывание PTS, то декодировщик приоритетов выбирает соответствующий вектор и вызывает PTSCB.

PTSCB определяет режим, общее число передач (при необходимости), общее число циклов, которые будут выполняться, пока PTS не потребует дополнительного обслуживания и источник и/или приёмник данных при передаче (при необходимости). Каждое прерывание PTS генерирует один цикл PTS. На рисунке 7.2 показан цикл работы PTS.

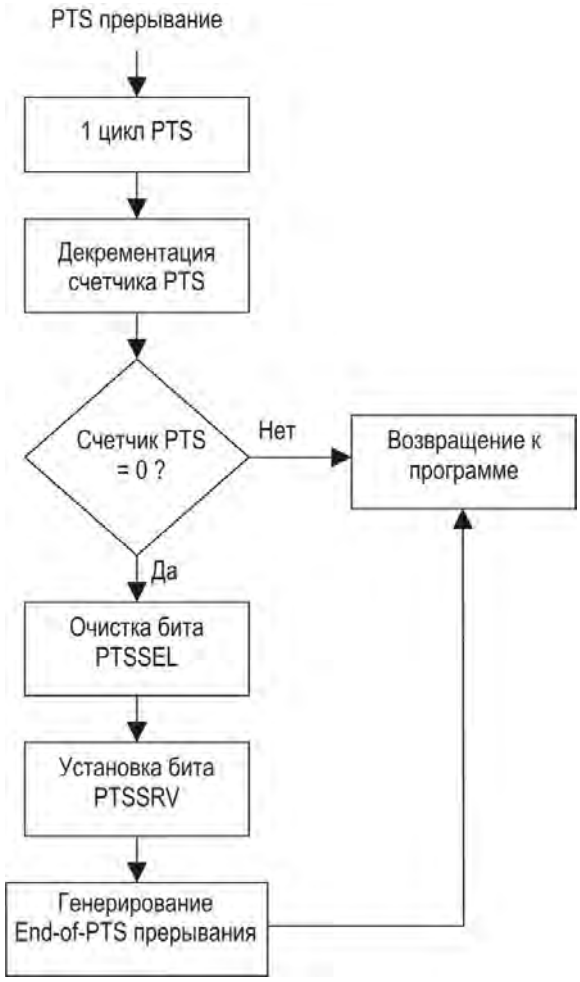


Рисунок 7.2 – Блок-схема цикла PTS

На рисунке 7.3 представлена блок-схема стандартного прерывания и PTS прерывания.

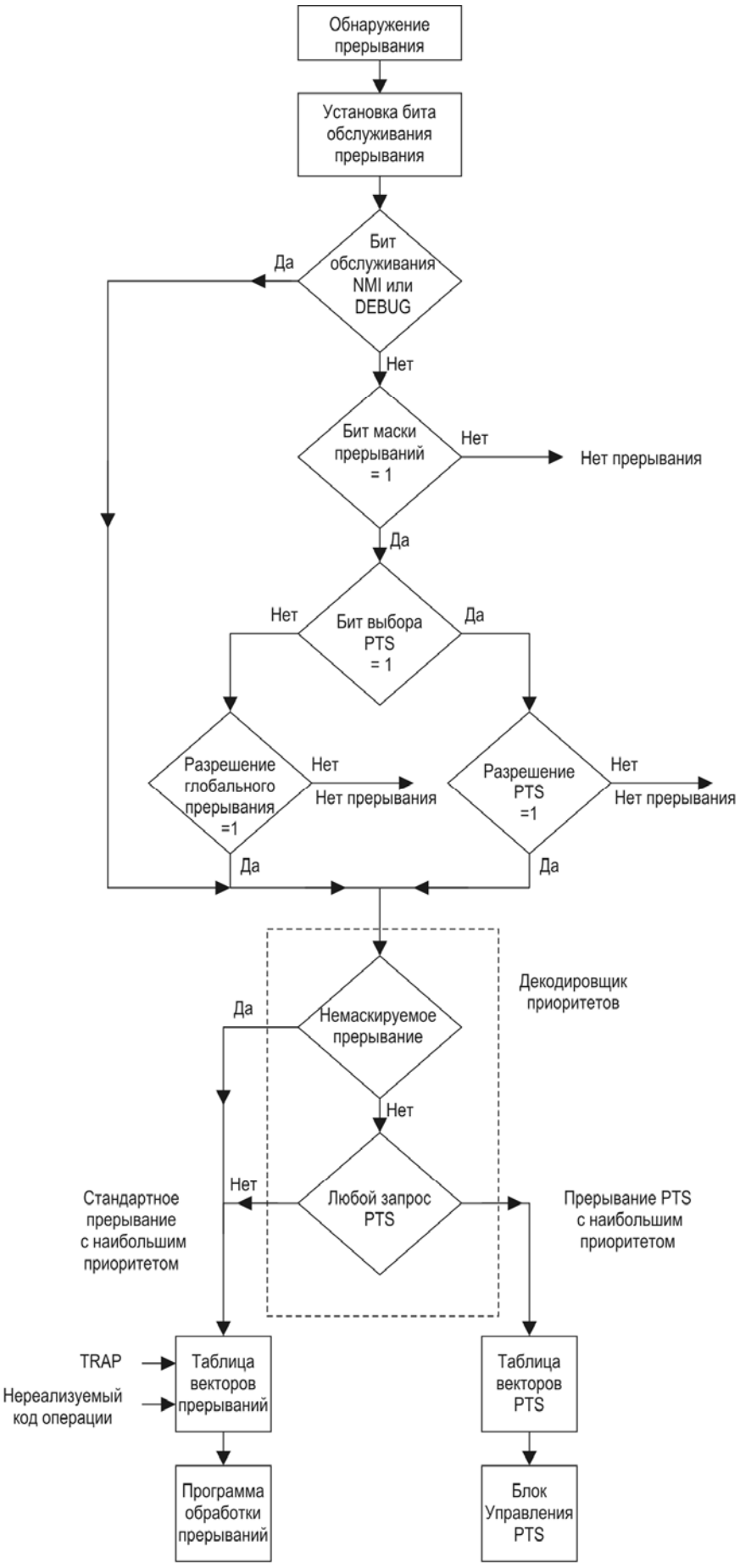


Рисунок 7.3 – Блок-схема стандартного прерывания и PTS прерывания

7.2 Приоритеты прерываний

Прерывания по неподдерживаемым командам и программное прерывание не имеют приоритетов; они напрямую проходят в контроллер прерываний для обслуживания. Контроллер прерываний выбирает соответствующий вектор, расположенный в специально отведённой памяти (раздел 6). Вектор содержит стартовый адрес программы обработки прерываний.

Декодировщик приоритетов определяет приоритеты всех других ожидаемых запросов прерывания. В таблице 7.1 показаны установленные по умолчанию приоритеты прерываний (16 – высший и 1 – низший). DEBUG имеет высший приоритет над всеми прерываниями, имеющими приоритет. Если произошло событие DEBUG модуляI, декодировщик приоритетов определяет его как запрос с высшим приоритетом, и контроллер прерываний выбирает соответствующий вектор, расположенный в специально отведённой памяти (раздел 6). Вектор содержит стартовый адрес соответствующей подпрограммы обработки прерываний.

Любой запрос прерывания PTS имеет больший приоритет, чем любой запрос маскируемого прерывания. Если DEBUG событие не произошло и NMI не установлен, декодировщик приоритетов устанавливает высший приоритет для запроса прерываний PTS, и контроллер прерываний выбирает соответствующий вектор PTS, расположенный в специально отведённой памяти. Вектор содержит стартовый адрес соответствующего блока управления PTS (PTSCB).

Если нет запроса ни от DEBUG модуля, ни от NMI, ни от PTS, декодировщик приоритетов устанавливает наивысший приоритет запросу стандартных прерываний, и контроллер прерываний выбирает соответствующий вектор, расположенный в специально отведённой памяти. Вектор содержит стартовый адрес соответствующей подпрограммы обработки прерываний.

7.2.1 Модификация приоритетов прерываний

Программное обеспечение может модифицировать установленные по умолчанию приоритеты маскируемых прерываний через регистры маски прерываний (INT_MASK и INT_MASK1). Например, можно определить, какие прерывания, если это необходимо, могут обслуживаться подпрограммой обработки прерывания, а какие – в цикле PTS. Следующая программа показывает возможный вариант запрещения всех прерываний, кроме EXTINT (приоритет 7), в подпрограмме обработки прерывания при приёме в последовательном канале (приоритет 9).

```
SERIAL_RI_ISR:                  ; Сохранение PSW, INT_MASK, & WSR
pusha                           ; Запрет всех прерываний, за исключением
                                ; EXTINT
di
ldb INT_MASK1,#00100000B        ; Разрешение обслуживания прерывания
                                ; Обслуживание прерываний RI
ei                              ; Восстановление PSW, INT_MASK, & WSR
popa
ret
```

Ячейка, которая расположена по адресу 2032H в таблице векторов прерываний, загружается значением метки SERIAL_RI_ISR, для разрешения прерывания приёма.

Таблица 7.1 – Источники, расположение и приоритеты векторов прерывания

Номер	Вектор прерывания	Источник	Ячейка вектора прерывания	Ячейка вектора PTS	Приоритеты *
Специальный	Неподдерживаемый код	Неподдерживаемый код	2012H		
Специальный	Программное прерывание	Команда TRAP	2010H		
INT16	DEBUG	Модуль отладки	2060H		16
INT15	NMI **	NMI	203EH		15
Каждое из следующих маскируемых прерываний может быть назначено на PTS. Любое из PTS прерываний имеет приоритет над всеми другими маскируемыми прерываниями.					
INT14	HSI FIFO Full	Буфер FIFO HSI полный	203CH	205CH	14
INT13	EXTINT **	P2.2	203AH	205AH	13
INT12	Timer 2 Overflow	Переполнение таймера 2	2038H	2058H	12
INT11	Timer 2 Capture **	Захват значения таймера 2	2036H	2056H	11
INT10	HSI FIFO 4	Четвёртый вход в FIFO HSI	2034H	2054H	10
INT09	Receive UART	RI флаг UART0, RI флаг UART1	2032H	2052H	9
INT08	Transmit UART	TI флаг UART0, RI флаг UART1	2030H	2050H	8
INT07	EXTINT **	P2.2 или P0.7	200EH	204EH	7
INT06	Serial Ports	RI флаг и TI флаг UART0 и UART1, SPIF флаг SPI, INT флаг I2C	200CH	204CH	6
INT05	Software Timer	Программный таймер 0 – 3, сброс таймера 2, запуск АЦП	200AH	204AH	5
INT04	HSI.0 **	Вход HSI.0	2008H	2048H	4
INT03	HSO	HSO.0 – HSO.5	2006H	2046H	3
INT02	HSI Data Available	Заполнение FIFO HSI или загрузка фиксирующего регистра HSI	2004H	2044H	2
INT01	A/D Conversion	Завершение АЦ преобразования или срабатывание цифровых компараторов	2002H	2042H	1
INT00	Timer Overflow	Таймер 1 или 2	2000H	2040H	0

* Прерывание по неподдерживаемым кодам и программное прерывание не имеют приоритетов. Они проходят напрямую в контроллер прерывания на обработку. DEBUG имеет наивысший приоритет из всех прерываний с приоритетом. Любое PTS-прерывание имеет больший приоритет, чем остальные маскируемые прерывания.

** Данные прерывания могут быть сконфигурированы как независимые внешние прерывания.

Далее приведена работа программы обработки прерываний.

После аппаратного детектирования и определения приоритетов запроса прерывания генерируется и выполняется вызов специального прерывания. Он помещает PC в стек и затем загружает его значением вектора прерывания, соответствующего наибольшему приоритету, если нет немаскируемого прерывания. Аппаратное обеспечение не может

вызвать следующее прерывание, пока не выполнится первая команда подпрограммы обработки прерывания.

Команда PUSHA, которая гарантированно выполняется, сохраняет значение PSW, INT_MASK1 и WSR в стеке, а затем очищает PSW и INT_MASK1. Дополнительно к арифметическим флагам PSW содержит регистр INT_MASK, бит глобального разрешения прерываний (I) и бит разрешения PTS (PSE). Обнуление регистров PSW и INT_MASK1 маскирует все маскируемые прерывания, запрещает обработку стандартных прерываний и запрещает PTS. Команда PUSHA препятствует вызову прерывания, пока не выполнится следующая команда.

Команда LDB INT_MASK1 разрешает те прерывания, которые выбраны в программе обработки прерывания. В приведенном примере только EXTINT может прерывать программу обработки прерывания по приему информации. Разрешением и запрещением прерываний программное обеспечение устанавливает свои приоритеты обработки прерываний.

Команда EI разрешает обработку прерываний после выполнения следующей за ней команды.

Действующая программа обработки прерываний выполняется в соответствии со структурой приоритетов, установленной программным обеспечением.

В конце программы обработки команда POPA восстанавливает исходное содержимое регистров PSW, INT_MASK1 и WSR. Так как вызов прерывания не может произойти сразу же за командой POPA, то последняя команда RET будет выполнена до того, как произойдет другой вызов прерывания. Рассмотрим, что будет, если команда POPA разрешает прерывания. Если контроллер прерываний отработал принятое прерывание, прежде чем команда RET выполнена, то адрес возврата в прерванную программу, которая выполнялась до того, как случилось данное прерывание, останется в стеке. Несмотря на то, что это не является проблемой для выполнения программы, всё же это может привести к переполнению стека, если прерывания происходят на большой частоте.

Следует заметить, что описанные операции с регистрами до и во время обслуживания прерывания не сохраняют и не восстанавливают регистры ОЗУ. Для программ обработки прерываний принято сохранять установки регистров пользователя из нижнего регистрового файла. Это возможно благодаря доступности 232 байт ОЗУ в нижнем регистровом файле. В дополнение к этому ОЗУ из верхнего регистрового файла доступно через режим вертикальных окон (раздел 6).

7.3 Синхронизация прерываний

Пять внешних источников могут прервать микроконвертер. Входная схема фиксирует прерывания по нарастающему фронту на входе. Сигнал на входе прерывания должен поддерживаться в высоком состоянии на время минимального импульса для уверенного распознавания. В таблице 7.2 приведена длительность минимального импульса для каждого внешнего прерывания.

Таблица 7.2 – Минимальная длительность импульса прерывания и фаза приёма

Источник прерывания	Вектор прерывания	Номер	Минимальная ширина импульса
HSI.0	Вывод HSI.0	INT04	≥2 тактов
NMI	NMI	INT15	≥2 тактов
P0.7	EXTINT	INT07	≥2 тактов
P2.2	EXTINT	INT07	≥2 тактов
P2.2	EXTINT1	INT13	≥2 тактов
Захват значения таймера 2	Timer 2 Capture	INT11	≥2 тактов

В таблице 7.3 описаны шесть команд, которые всегда блокируют прерывания до тех пор, пока не выполнится следующая за ней команда.

Таблица 7.3 – Команды, блокирующие прерывания

Команда	Описание
DI	Запрещает прерывания
EI	Разрешает прерывания после выполнения следующего состояния
POPA	Считывает два слова из верхушки стека и помещает первое в регистровую пару INT_MASK1/WSR, а второе – в регистровую пару PSW/INT_MASK
POPF	Считывает слово из стека и помещает в регистровую пару PSW/INT_MASK
PUSHA	Записывает содержимое регистров PSW, INT_MASK, INT_MASK1 и Window Select в верхушку стека, затем очищает регистры PSW, INT_MASK и INT_MASK1
PUSHF	Записывает содержимое пары регистров PSW/INT_MASK в верхушку стека, затем очищает их

Выполнение любой из следующих операций также блокирует прерывания, пока не выполнится следующая команда:

- прерывание по неподдерживаемым кодам;
- прерывание пошагового исполнения (программное прерывание).

7.3.1 Время ожидания прерываний

Время ожидания прерывания – это общая задержка между временем, когда произошла генерация прерывания, и временем, когда МК начнёт выборку первой команды подпрограммы обработки прерывания или цикла PTS. Задержка имеется между временем детектирования прерывания и временем его выполнения. Стандартное прерывание не начнет выполняться, пока текущая команда не будет выполнена. Если прерывание пришло менее чем за один машинный цикл до завершения текущей команды, то оно не будет обрабатываться ЦПУ.

После завершения текущей команды, когда стандартное прерывание начинает выполняться, аппаратно очищается бит ожидания прерывания и принудительно вызывается адрес, содержащийся в соответствующем векторе прерывания. Если стек находится во внешнем ОЗУ, дополнительно требуются циклы для ожидания шины. Если выбрано прерывание PTS, то запрос по окончании одного машинного цикла поступает в блок PTS, который работает параллельно ЦПУ, поэтому не требует окончания текущей выполняемой команды. Момент времени фактического начала выборки блоков управления сервером и передачи данных зависит от конкретной программы и текущего состояния ИС и не поддается вычислению.

7.3.2 Вычисление времени ожидания выборки первой команды подпрограммы обработки стандартного прерывания

Максимальное время ожидания получается, если прерывание пришло слишком поздно для подтверждения во время текущей команды и стек находится во внешней памяти. Для подсчёта времени ожидания суммируются следующие условия:

- цикл для формирования запроса в ЦПУ;
- общее количество тактов следующей команды; наибольшая команда (DIV с индексной адресацией) содержит 10 циклов;
- время ожидания шины для выборки вектора перехода третьего цикла;
- выборка вектора перехода (зависит от режима функционирования ИС – с внешней, внутренней памятью, от параметров внешней шины);
- время ожидания шины для записи стека, если стек находится во внешней памяти – пять циклов;
- время записи стека (зависит от режима функционирования ИС – с внешней, внутренней памятью, от параметров внешней шины). Если стек находится в ОЗУ – 1 цикл;

- время ожидания шины для выборки первого адреса – 2 цикла.

Самая быстрая обработка прерываний будет в случае, если стек находится в области внутреннего ОЗУ, и запрос поступил во время короткой команды или простоя ЦПУ. Стоит обратить внимание, что в последовательности выбор вектора – запись в стек, первым будет именно выбор вектора.

7.4 Специальные прерывания

МК поддерживают четыре специальных прерывания: по неподдерживаемым кодам, пошагового исполнения, NMI и DEBUG. Эти прерывания не используют бит разрешения прерывания (I) в PSW (PSW.1), и они не могут быть замаскированы. Все эти прерывания обслуживаются контроллером прерывания; они не могут быть назначены на PTS. Из них только NMI и DEBUG проходят через детектор передачи и декодировщик приоритетов. Остальные два специальных прерывания проходят напрямую в контроллер прерывания на обработку. Следует быть осторожными, так как данные прерывания часто назначаются на специальные функции в современных устройствах отладки.

7.4.1 Неподдерживаемые коды

Если ЦПУ попытается выполнить неподдерживаемый код, то произойдёт переход по вектору 2012H. Это предотвращает их случайное программное исполнение в случае аппаратного или программного сбоя. Вектор прерывания должен обязательно содержать стартовый адрес программы обработки ошибки, чтобы не усугубить и без того ошибочную ситуацию.

7.4.2 Программное прерывание

Команда TRAP (код 0F7H) вызывает прерывание по вектору, расположенному в ячейке 2010H. Команда TRAP позволяет выработать прерывание после каждой команды. Это полезно при отладке программы или генерации программных прерываний.

7.4.3 Немаскируемое прерывание (NMI)

Внешний вывод NMI генерирует немаскируемое прерывание для выполнения экстренных программ прерывания. NMI имеет высший приоритет над всеми прерываниями с приоритетом. Оно передается напрямую от детектора передачи к декодировщику приоритетов и вызывается косвенно через ячейку 203EH. Если ваша система не использует NMI-прерывание, необходимо заземлить вывод NMI для предотвращения неожиданных прерываний.

По аналогии с регистром INT_PEND1 в регистре INT_MASK1 существует бит маски NMI. Однако этот бит не используется; NMI разрешён как при установленном так и при сброшенном NMI_MASK.

Таблица 7.4 – Прерывание, управление PTS и регистры статуса

Мнемоника регистра	Имя регистра	Описание
1	2	3
INT_MASK INT_MASK1	Регистры маски прерывания	Данные регистры разрешают/запрещают все маскируемые прерывания (то есть все прерывания, кроме прерываний по неподдерживаемым кодам, пошагового исполнения и NMI)
INT_PEND INT_PEND1	Регистры ожидания прерывания	Биты этого регистра устанавливаются аппаратно для индикации ожидания прерываний
IOC1	Регистр управления вводом-выводом	Данный регистр указывает источник для прерываний INT00, INT02 и INT07
IOS1	Регистр статуса ввода-вывода	Этот регистр содержит флаги, индицирующие, какие события вызвали прерывания

Окончание таблицы 7.4

1	2	3
PSW	Слово состояния программы	Этот регистр содержит один бит, который глобально разрешает или запрещает обработку всех маскируемых прерываний, а также бит, который разрешает или запрещает PTS
PTSSEL	Регистр выбора PTS	Это регистр выбирает либо PTS цикл, либо программу стандартной обработки прерываний для каждого из 15 маскируемых запросов прерываний
PTSSRV	Регистр обработки PTS	Биты в этом регистре устанавливаются аппаратно по запросу
DINTC	Регистр управления прерыванием DEBUG	Данный регистр управляет источником прерывания модуля DEBUG

7.5 Программирование прерываний

В таблице 7.4 показаны программируемые регистры, которые воздействуют на работу и функции контроллера прерываний и PTS.

7.5.1 Выбор способа обработки прерываний: через PTS или стандартную процедуру обработки прерываний

Через регистр выбора PTS (PTSSEL) выбирается либо цикл PTS, либо программа стандартной обработки прерываний для каждого из 15 маскируемых запросов прерывания. Установкой бита выбирается цикл PTS; обнулением бита выбирается программа стандартной обработки прерываний.

7.5.2 Разрешение PTS прерываний

Если прерывания назначены на подпрограммы стандартной программной обработки, нужно разрешить как обработку прерываний, так и каждое прерывание в отдельности. Бит глобального разрешения прерываний (I) в PSW (PSW.1) глобально разрешает или запрещает обработку всех маскируемых прерываний. Команда EI устанавливает этот бит, который разрешает обработку прерываний. Команда DI очищает бит, чем запрещает обработку прерываний. Биты в INT_MASK и INT_MASK1 индивидуально разрешают или запрещают прерывания (таблица 7.5).

Таблица 7.5 – Источники стандартных прерываний, векторы и биты регистра

Источник прерывания	Вектор прерывания	Номер	Бит разрешения прерывания [1)]	Условия выбора источника [2)]
1	2	3	4	5
Завершение аналого-цифрового преобразования	A/D Convertion	INT01	INT_MASK.1	–
Срабатывание цифровых компараторов	A/D Convertion	INT01	INT_MASK.1	AD_COMP_INT
Запуск АЦ преобразования	Software Timer	INT05	INT_MASK.5	–
Четвёртая запись в FIFO HSI	HSI FIFO 4	INT10	INT_MASK1.2	–
Заполнение FIFO HSI	HSI FIFO Full	INT14	INT_MASK1.6	–
Заполнение FIFO HSI	HSI Data Available	INT02	INT_MASK.2	IOC1.7 = «1»
Регистр удержания HSI загружен	HSI Data Available	INT02	INT_MASK.2	IOC1.7 = «0»
HSI.0	Вывод HSI.0	INT04	INT_MASK.4	–

Окончание таблицы 7.5

1	2	3	4	5
HSO.0 – HSO.5	HSO	INT03	INT_MASK.3	–
NMI	NMI	INT15	–	–
События модуля отладки	DEBUG	INT16	–	–
P0.7	EXTINT	INT07	INT_MASK.7	IOC1.1 = «1»
P2.2	EXTINT	INT07	INT_MASK.7	IOC1.1 = «0»
P2.2	EXTINT1	INT13	INT_MASK1.5	–
RI UART0 или UART1	Receive UART	INT09	INT_MASK1.1	–
RI UART0 или UART1, флаг SPIF потра SPI, флаг INT порта I2C	Serial Ports	INT06	INT_MASK.6	–
Программные таймеры 0 – 3	Software Timer	INT05	INT_MASK.5	–
TI UART0 или UART1	Transmit	INT08	INT_MASK1.0	–
TI UART0 или UART1	Serial Port	INT06	INT_MASK.6	–
Переполнение таймера 1	Timer Overflow	INT00	INT_MASK.0	IOC1.2 = «1»
Захват значения таймера 2	Timer 2 Capture	INT11	INT_MASK1.3	–
Переполнение таймера 2	Timer Overflow	INT00	INT_MASK.0	IOC1.3 = «1»
Переполнение таймера 2	Timer 2 Overflow	INT12	INT_MASK1.4	–
Сброс таймера 2	Software Timer	INT05	INT_MASK.5	–

[1)] В этой колонке перечисляются для каждого из источников прерываний биты маски, которые необходимо установить для разрешения прерываний. Семь источников прерываний могут генерировать два отдельных прерывания – прерывания, совместимые с 8096BH, и новые специальные прерывания (HSI FIFO Full, EXTINT1, Receive UART0 и UART1, Transmit UART0 и UART1, Timer 2 Overflow). Во всех случаях только одно прерывание может быть разрешено для каждого источника. (Для этого бит маски должен быть установлен только для одного из двух возможных прерываний.)

[2)] Три из совместимых с 8096BH прерывания (HSI Data Available, EXTINT и Timer Overflow) могут генерироваться любым из двух источников. В колонке приведены биты регистра IOC1 и их значения, которые выбирают соответствующий источник.

Прерывания, имеющие место, когда обработка прерываний глобально запрещена (PSW.1 очищено), сохраняются в регистрах ожидания прерывания.

7.5.3 Выбор источников прерываний

На рисунке 7.4 показаны источники прерывания для каждого вектора прерывания. В таблице 7.5 приведены биты регистра IOC1 и их значения, которые выбирают соответствующий источник.

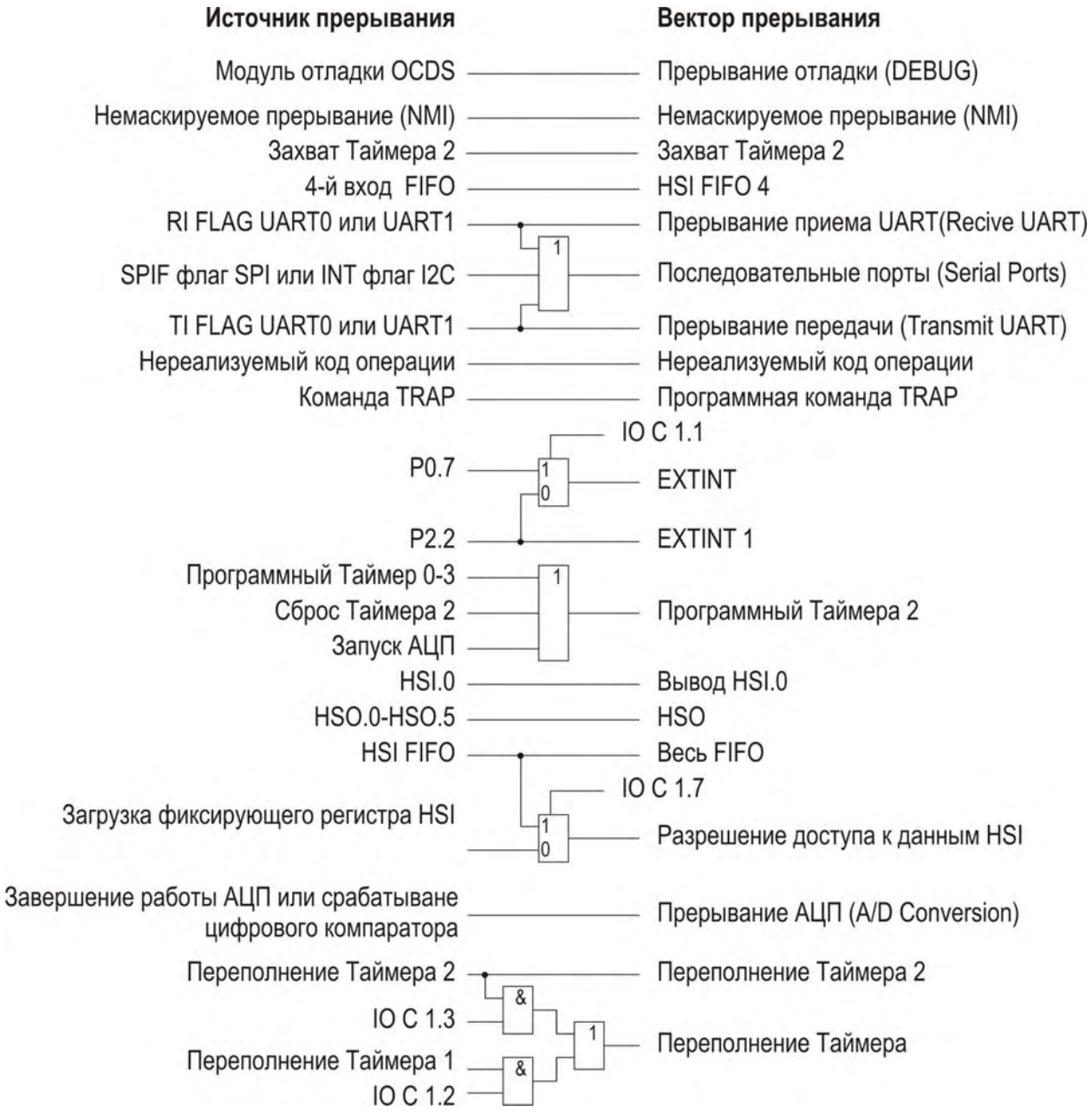


Рисунок 7.4 – Источники прерываний

7.5.4 Регистры масок прерываний

Регистры масок прерываний INT_MASK и INT_MASK1 разрешают или запрещают (маскируют) каждое из прерываний. За исключением бита немаскируемого прерывания (NMI) (INT_MASK1.7), установка бита разрешает соответствующий источник прерываний; очистка бита запрещает источник. Если устройство сбрасывается, то регистры масок прерываний очищаются (прерывания запрещены).

Для достижения соответствия с регистром INT_PEND1 в регистре INT_MASK1 также содержится бит маски NMI. Однако этот бит маски не используется; NMI разрешено как при установленном, так и при сброшенном INT_MASK. Для обеспечения совместимости с будущими версиями контроллера всегда следует записывать ноль в бит маски NMI.

7.5.5 Регистры ожидания прерываний

Когда детектор запросов обнаруживает прерывание, он устанавливает соответствующий бит в регистре INT_PEND или INT_PEND1. Этот бит устанавливается даже для запрещённых (замаскированных) прерываний. Бит ожидания прерывания

очищается, когда программа переходит на подпрограмму обработки прерывания (стандартные прерывания) или на PTSCB (PTS прерывания). INT_PEND и INT_PEND1 могут быть считаны для определения ожидающих прерываний. Они также могут быть модифицированы (записаны) либо для снятия ожидания прерываний, либо для генерации прерываний программным обеспечением.

Регистр PTSSEL содержит запросы end-of-PTS прерываний. End-of-PTS прерывания показывают, что PTS требует обслуживания. Программа обработки end-of-PTS прерывания обновляет PTSCB и устанавливает соответствующий бит, разрешая обработку PTS прерываний.

7.6 Блоки управления PTS

Каждое PTS-прерывание запрашивает блок данных, называемых блоком управления PTS (PTSCB). PTSCB определяет режим PTS, число циклов PTS и адреса источника и приёмника передаваемых данных. Необходимо установить PTSCB для каждого источника прерывания, прежде чем будут разрешены прерывания PTS. Каждый PTSCB требует восемь байт данных в регистровом ОЗУ. Адрес первого (младшего) байта хранится в таблице векторов PTS в специально отведённой памяти. Блок управления должен начинаться с четного адреса. В таблице 7.6 приводятся PTSCB для каждого режима PTS. Неиспользуемые байты PTSCB могут использоваться как дополнительное ОЗУ.

Таблица 7.6 – Управление PTS

Номер байта	Одиночная передача	Блочная передача	Режим HSO	Режим HSI
8	Не используется	Не используется	Не используется	Не используется
7	Не используется	PTSBLOCK	PTSBLOCK	PTSBLOCK
6	PTSDST (HI)	PTSDST (HI)	Не используется	Не используется
5	PTSDST (LO)	PTSDST (LO)	Не используется	Не используется
4	PTSSRC (HI)	PTSSRC (HI)	PTSSRC (HI)	PTSDST (HI)
3	PTSSRC (LO)	PTSSRC (LO)	PTSSRC (LO)	PTSDST (LO)
2	PTSCON	PTSCON	PTSCON	PTSCON
1	PTSCOUNT	PTSCOUNT	PTSCOUNT	PTSCOUNT

7.6.1 Регистр PTSCOUNT

Первым байтом каждого PTSCB всегда является регистр PTSCOUNT. PTSCOUNT определяет число циклов PTS, выполняемых без участия программного обеспечения. Так как PTSCOUNT имеет 8-битное значение, максимальное количество циклов – 256. PTSCOUNT уменьшается на единицу в конце каждого цикла PTS. Когда PTSCOUNT становится равным нулю, аппаратно очищается соответствующий бит PTSSEL и устанавливается бит PTSSRV, который запрашивает end-of-PTS прерывание.

End-of-PTS прерывание

End-of-PTS прерывание является стандартным. Контроллер прерываний обслуживает его с помощью программы обработки прерывания, находящейся в памяти, расположение которой указано стандартным вектором прерывания. Например, PTS обрабатывает прерывание Transmit, если устанавливается PTSSEL.8. Вектор PTS 2050H, а соответствующий вектор end-of-PTS прерывания 2030H, – стандартный вектор прерывания Transmit. Когда передано управление программе обработки прерывания end-of-PTS, аппаратно очищается бит PTSSRV. Программа обработки прерывания должна установить бит PTSSEL, разрешающий обработку PTS прерывания.

7.6.2 Регистр PTSCON

Вторым байтом каждого PTSCB всегда является регистр PTSCON. Три бита регистра PTSCON определяют режим PTS: одиночная передача, блочная передача, HSO или HSI (таблицы 7.7, 7.8). Режим PTS определяет функцию остальных битов. PTSCON имеет одну конфигурацию для режимов одиночной и блочной передачи (рисунок 7.5) и другую для режимов сканирования АЦП, HSO и HSI (рисунок 7.6).

Таблица 7.7 – Выбор режимов PTS

Бит 7	Бит 6	Бит 5	Выбираемый режим
0	0	0	Блочная передача
0	0	1	HSO
0	1	0	Не используется
0	1	1	HSI
1	0	0	Одиночная передача
1	0	1	Не используется
1	1	0	Не используется
1	1	1	Не используется

Таблица 7.8 – Бит 3 PTSCON (режимы сканирования АЦП, HSI и HSO)

Номер бита	Мнемоника бита	Имя бита	Описание
3	UPDT	Обновление регистра	При установке этого бита соответствующий регистр будет загружен значением, которое будет в конце PTS цикла. При очистке этого бита регистр будет загружен значением, которое было в начале PTS цикла. Режим Регистр HSI PTSDST HSO PTSSRC

PTSCON
регистр управления PTS (режимы одиночной и блочной передачи)

значение после Сброса: 00H

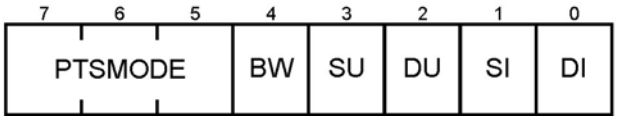


Рисунок 7.5 – Формат регистра PTSCON (режимы одиночной и блочной передачи)

PTSCON
регистр управления PTS (режимы HSO и HSI)

значение после Сброса: 00H

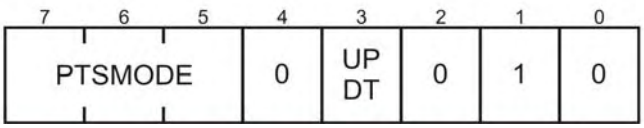


Рисунок 7.6 – Формат регистра PTSCON (режимы HSI и HSO)

7.7 Режим одиночной передачи

В режиме одиночной передачи каждый PTS цикл передает один байт или слово (установка BW бита в PTSCON) из одной ячейки памяти в другую. Этот режим обычно

используется с прерываниями по последовательному порту ввода-вывода или концу аналого-цифрового преобразования. PTSCOUNT регистр указывает количество передач (каждая передача – один PTS цикл). PTS пересылает байт или слово из ячейки, указанной регистром источника (PTSSRC), в ячейку, указанную регистром назначения (PTSDST).

PTSSRC и PTSDST могу указывать любую ячейку памяти; тем не менее, они должны указывать четный адрес, если выбрана передача слова. При установке битов автоинкремента и обновления PTS увеличивает адрес источника (если установлены биты SI и SU) и/ или адрес назначения (если установлены биты DI и DU) в конце каждого PTS цикла. Адреса увеличиваются на 1, если установлен режим передачи байта, и на 2, если установлен режим передачи слова. В режиме одиночной передачи каждая пара битов автоинкремента и обновления должна быть либо установлена, либо очищена. Программирование битов автоинкремента и обновления (0, 1) или (1, 0) – ошибка. PTSSRC и PTSDST могут быть увеличены (и обновлены) независимо друг от друга.

7.7.1 Пример режима одиночной передачи

Следующий PTSCB определяет девять PTS циклов (таблица 7.9). Каждый цикл пересылает одно слово с адреса 20H во внешнюю память. PTS передает первое слово по адресу 6000H, затем он увеличивает и обновляет адрес назначения и декрементирует PTSCOUNT регистр; он не увеличивает адрес источника. В начале второго цикла PTS пересылает второе слово с адреса 20H в адрес 6002H. Когда PTSCOUNT равен нулю, PTS заполнит область 6000H – 600FH и сформирует прерывание end-of-PTS.

Таблица 7.9 – PTSCB в режиме одиночной передачи

Наименование регистра	Содержимое
Не используется	
Не используется	
PTSDST(HI)	60H
PTSDST(LO)	00H
PTSSRC(HI)	00H
PTSSRC(LO)	20H
PTSCON	95H (DI, DU, BW = «1»)
PTSCOUNT	09H

7.8 Режим блочной передачи

В режиме блочной передачи PTS пересылает блок данных из одних ячеек памяти в другие. PTSBLOCK регистр указывает количество байтов или слов в каждом блоке (n = 1 – 32). PTS пересылает блок байтов или слов из ячейки, указанной регистром источника (PTSSRC), в ячейку, указанную регистром назначения (PTSDST).

PTSSRC и PTSDST могут указывать на любую ячейку в памяти, тем не менее, они должны указывать на четный адрес, если выбрана передача слова. При установке битов автоинкремента в регистре PTSCON PTS увеличивает адрес источника (установлен SI) и/или адрес назначения (установлен DI) в конце каждой PTS передачи. Если бит обновления также установлен, увеличенный адрес записывается в PTSSRC (установлен SU) или в PTSDST (установлен DU) после каждого PTS цикла. Установка обоих битов инкремента и обновления означает, что адрес источника и/или адрес назначения будут инкрементироваться после каждого PTS цикла. Регистры увеличиваются на единицу, если выбрана передача байта, или на два, если выбрана передача слова. Увеличение и обновление могут быть выбраны независимо (не так, как в режиме одиночной передачи).

В этом режиме важно различать PTS передачу и PTS цикл. PTS передача – это пересылка одного байта или слова от источника к месту назначения. PTS цикл состоит из передачи полного блока байтов или слов.

7.8.1 Пример режима блочной передачи

PTSCB в таблице 7.10 определяет три PTS цикла, каждый из которых передает байты из области памяти 20H – 24H в один из следующих блоков: 6000H – 6004H, 6005H – 6009H или 600AH – 600EH. Каждый PTS цикл требует появления пяти пересылок. Источник и место назначения инкрементируются после каждой передачи, но только место назначения обновляется после каждого цикла. Первый байт в каждом цикле всегда читается из ячейки 20H.

Таблица 7.10 – Режим PTSCB Block Transfer

Наименование регистра	Содержимое
Не используется	
PTSBLOCK	05H
PTSDST(HI)	60H
PTSDST(LO)	00H
PTSSRC(HI)	00H
PTSSRC(LO)	20H
PTSCON	17H (DI, SI, DU, BW = «1»)
PTSCOUNT	03H

7.9 Режим HSI

В режиме HSI PTS сбрасывает содержимое HSI FIFO в таблицу, расположенную либо во внутренней либо во внешней памяти. PTSDST регистр содержит адрес таблицы.

Любое HSI прерывание может быть использовано для запуска PTS цикла. PTSBLOCK регистр указывает, сколько HSI FIFO блоков (n = 1 – 7) будет передано в таблицу памяти в течение каждого PTS цикла. Необходимо ввести значение, соответствующее прерыванию, которое генерирует PTS цикл. Например, четвертый вход в FIFO может заставить HSI сгенерировать HSI FIFO четвертое прерывание (INT10). Если это прерывание было использовано для запуска PTS цикла, то следует записать «4» в PTSBLOCK регистр, таким образом, четыре входа в FIFO будут переписаны в таблицу.

Первым из буфера FIFO читается HSI_STATUS регистр, а затем HSI_TIME регистр. HSI_STATUS регистр содержит биты состояния результата и текущее состояние HSI выводов. HSI_TIME регистр содержит значение таймера 1 в момент HSI события (смотри пункт 12.2 для детального знакомства с HSI модулем).

Каждая PTS передача пересылает HSI_STATUS и HSI_TIME регистры в последовательные слова в памяти (таблица 7.11). При установке UPDT бита (PTSCON.3) PTSDST регистр сохраняет свое конечное значение в конце PTS цикла.

Таблица 7.11 – Таблица PTS для режима HSI

Адрес	Данные	
XXX+0AH	HSI_TIME_2	
XXX+8H	0FFH	HSI_STATUS_2
XXX+6H	HSI_TIME_1	
XXX+4H	0FFH	HSI_STATUS_1
XXX+2H	HSI_TIME_0	
XXX	0FFH	HSI_STATUS_0

7.9.1 Пример режима HSI

PTSCB определяет десять PTS циклов (таблица 7.12), каждый из которых передает семь блоков данных HSI_STATUS/HSI_TIME из HSI FIFO в таблицу, начинающуюся с ячейки памяти 100H. Адрес назначения инкрементируется после каждой передачи и обновляется с новым значением после каждого цикла.

Таблица 7.12 – PTSCB в режиме HSI

Наименование регистра	Содержимое
Не используется	
PTSBLOCK	07H
Не используется	
Не используется	
PTSDST (HI)	01H
PTSDST (LO)	00H
PTSCON	6AH (UPDT = «1»)
PTSCOUNT	0AH

7.10 Режим HSO

PTS в режиме HSO загружает HSO CAM из таблицы, расположенной либо во внутренней либо во внешней памяти. Режим HSO аналогичен режиму HSI, за исключением того, что PTS пересылает данные из таблицы в HSO CAM, а не наоборот. PTSSRC регистр содержит адрес таблицы.

Таблица 7.13 – PTS для режима HSO

Адрес	Данные	
XXX+0AH	HSI_TIME_2	
XXX+8H	0FFH	HSI_STATUS_2
XXX+6H	HSI_TIME_1	
XXX+4H	0FFH	HSI_STATUS_1
XXX+2H	HSI_TIME_0	
XXX	0FFH	HSI_STATUS_0

Длина каждого CAM регистра – 24 бита. Шестнадцать битов содержат время наступления события по таймеру 1 или таймеру 2. Оставшиеся 8 битов определяют тип события. Загрузка этой информации в таблицу памяти производится в формате, приведенном в таблице 7.13. Каждый PTS-цикл передает данные в регистры HSO_COMMAND и HSO_TIME. Данные передаются из регистров в буферный регистр HSO. Команды задерживаются в этом регистре до тех пор, пока CAM-регистр содержит информацию, в определенное время команды вводятся в CAM.

Любое HSO прерывание может быть использовано для запуска PTS цикла. PTSBLOCK регистр указывает, сколько HSO входов (n = 1 – 8) будут переданы из таблицы памяти в течение каждого PTS цикла. При установке UPDT бита (PTSCON.3) PTSSRC регистр сохраняет свое конечное значение в конце PTS цикла.

7.10.1 Пример режима HSO

PTSCB в таблице 7.14 определяет десять PTS циклов, каждый из которых передает восемь блоков данных HSO_COMMAND/HSO_TIME в HSO из таблицы, начинающейся с ячейки памяти 100H. Адрес источника инкрементируется после каждой передачи и обновляется после каждого цикла.

Таблица 7.14 – PTSCB в режиме HSO

Наименование регистра	Содержимое
Не используется	
PTSBLOCK	08H
Не используется	
Не используется	
PTSSRC(HI)	01H
PTSSRC(LO)	00H
PTSCON	2AH (UPDT = «1»)
PTSCOUNT	0AH

8 Порты ввода-вывода

Порты ввода-вывода (I/O) служат для передачи информации между главным устройством и окружающими его системами. Они способны считывать состояние системы, управлять системой, вывести статус устройства, выбрать параметры конфигурации системы, формировать управляющие сигналы, обеспечивать последовательную связь и т. д. Их использование в проекте ограничено только количеством доступных выводов I/O и воображением инженера.

8.1 Обзор функциональных возможностей

Каждый микроконвертер имеет пять 8-битных портов ввода-вывода. Каждый порт имеет внешние выводы, которые выполняют функции входа (ввод информации), выхода (вывод информации), квазидвунаправленного или двунаправленного ввода-вывода с открытым стоком или комбинацию этих функций. Почти все выводы портов I/O имеют дополнительные функции. Например, один из выводов может стать PWM выходом, тогда как другой может использоваться как аналоговый вход.

Каждый вывод порта I/O устанавливается в определенный режим по умолчанию (например, в режим выхода). Однако режим работы вывода может измениться при выборе его дополнительной функции. Например, квазидвунаправленный вывод станет выходом, если выбрать дополнительную функцию. В любом случае, вывод имеет основной набор характеристик, которые связаны с данной структурой. В таблице 8.1 показаны пять портов I/O, их дополнительные функции и функции по умолчанию. Пятая колонка таблицы показывает, какой SFR определяет, будет ли вывод работать как порт или выполнять альтернативную функцию. Некоторые выводы портов выполняют как функцию порта, так и альтернативную функцию одновременно (например, порт 0 и аналоговые входы). В этом случае пятая колонка таблицы 8.1 пустая.

Дополнительно к стандартным портам I/O, модули HSI и HSO обеспечивают функции ввода-вывода, если не нужны зависящие от времени характеристики этих выводов.

Таблица 8.1 – Функции выводов портов

Выводы портов	Тип вывода порта*	Дополнительные функции	Тип вывода порта при выбранной дополнительной функции *	Управляющий регистр SFR
1	2	3	4	5
P0.0 – P0.3	INPUT	ACH0 − ACH3	INPUT	
P0.4, P0.5, P0.6	INPUT	ACH4 – ACH6, ACHN0 – ACHN2-	INPUT	
P0.7	INPUT	ACH7, ACHN3, REFOUT	INPUT, INPUT, OUTPUT	
P1.0, P1.1	QBD	Нет		
P1.2	QBD	SS#	INPUT	
P1.3, P1.4	QBD	PWM.1, PWM.2	OUTPUT	IOC3
P1.5	QBD	MOSI	QBD	
P1.6	QBD	MISO	QBD	
P1.7	QBD	SCK	QBD	
P2.0	OUTPUT	TXD0	OUTPUT	IOC1
P2.1	INPUT	RXD0	INPUT OR OUT	SPCON0
P2.2	INPUT	EXTINT	INPUT	IOC1

Окончание таблицы 8.1

1	2	3	4	5
P2.3	INPUT	T2CLK, SCL	INPUT, INPUT OR OUT	
P2.4	INPUT	T2RST, RXD1	INPUT, INPUT OR OUT	IOC0 SPCON1
P2.5	OUTPUT	PWM.0	OUTPUT	IOC1
P2.6	QBD	T2UPDN, TXD1	INPUT, OUTPUT	IOC2, IOC1
P2.7	QBD	T2CAP, SDA	QBD, QBD	
P3, P4	ODBD	AD0 − AD15	ODBD	
_____ * Обозначения типов выводов расшифровываются в следующих разделах.				

8.1.1 Входной вывод порта

Любой вывод порта, определенный как вход (INPUT), может быть только считан. Любая операция записи игнорируется (или недопустима). Схема ввода состоит из входного фиксирующего триггера и буфера считывания (рисунок 8.1). Состояние на входе фиксируется за один такт перед разрешением чтения буфера ввода.

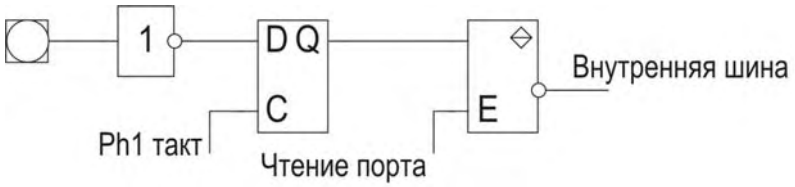


Рисунок 8.1 – Входной вывод порта

Захват происходит в течение фазы 1 (пока CLKOUT имеет низкий уровень), и значение (высокое или низкое) передается на внутреннюю шину. Если состояние на выводе изменяется во время фиксации, то новое значение можно получить, а можно и не получить, при считывании. Входные порты не имеют выходных драйверов. Ток утечки на входе при этом очень мал, так же как и емкостная нагрузка.

8.1.2 Выходной вывод порта

Выходной порт (OUTPUT) состоит из защелки порта и логики управления выходом (рисунок 8.2). Новое значение на выходе появляется во время фазы 1 (пока CLKOUT имеет низкий уровень). Это значение остается стабильным, пока другая операция записи не изменит его (или не произойдет сброс устройства).

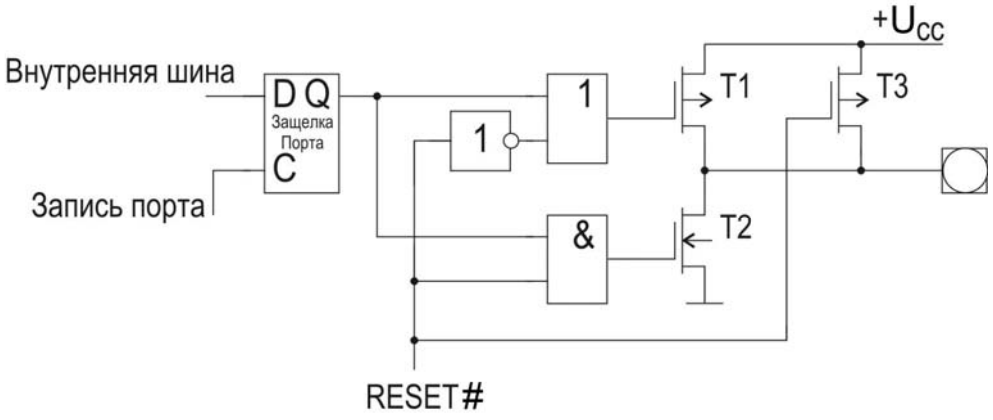


Рисунок 8.2 – Выходной вывод порта

Выходной порт не имеет схемы фиксации и буфера для чтения. При операции считывания этого порта его значение неопределенно и должно быть замаскировано.

Не существует механизма считывания значения с вывода порта, возможна только установка или очистка этого вывода. Если необходимо модифицировать значение на выходе, то это значение должно быть сохранено в памяти. Значение модифицируется в памяти и затем используется для установки или очистки выхода.

8.1.3 Квазидвунаправленный вывод порта

Выводы квазидвунаправленного порта (QBD) могут быть использованы в качестве выводов для ввода и/или вывода (схема управления направлением не нужна). Эти выводы имеют активные низкие и пассивные высокие значения. При высоком уровне на выводе поддерживается функция ввода.

На рисунке 8.3 показана общая схема вывода QBD порта. Так же как и схема входного порта, она имеет фиксирующий триггер (защелку) и буфер считывания. Как схема выходного порта, она имеет защелку данных. Структура выходного драйвера делает QBD вывод уникальным.

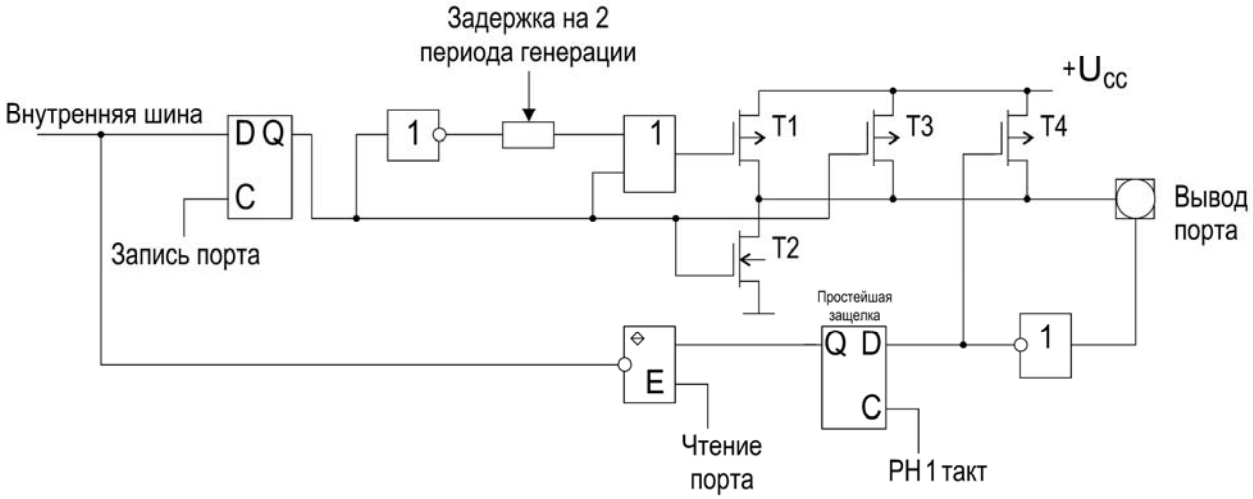


Рисунок 8.3 – Квазидвунаправленный вывод порта

Выход схемы имеет два возможных состояния: низкий уровень, обеспечивающий достаточно большой ток, и слаботочный высокий уровень, когда выход переключается с низкого уровня на высокий и переход осуществляется за более короткое время. Пока на выходе низкий уровень, переключение невозможно (т. е. невозможно обеспечить высокий уровень на выходе внешними средствами без нарушения спецификации устройства). Запись «1» в порт закрывает транзистор Т2 и открывает Т3. Чтобы переключение транзисторов происходило быстрее, на время одного такта сильноточный транзистор Т1 открывается, а затем закрывается, оставляя активным Т3. Т3 удерживает на выводе уровень логической единицы до тех пор, пока нагрузка на выводе мала.

Вывод QBD порта имеет схему считывания, которая дает возможность считывать выходное значение вывода непосредственно (в отличие от выходного порта). Однако, если выход порта подключается внешними средствами к уровню логического нуля, то чтение этого порта ошибочно покажет, что в него был записан ноль. Квазидвунаправленный вывод порта может быть входом, благодаря возможности отключения выходного транзистора Т4. Уменьшение суммарного тока, когда вывод подключен к низкому внешнему уровню, отключает транзистор Т4 при достижении значения нуля (низкий уровень, который фиксируется при напряжении около 2 В).

На рисунке 8.4 приведена переходная характеристика порта, когда транзисторы Т3 и Т4 открыты. Напряжение на выходе уменьшается от U_{CC1} до тех пор, пока ток на выходе

не достигнет порогового значения I_{TL}. В этой точке Т4 закрывается, и остается открытым только Т3.

Выходной двунаправленный порт включает в себя транзистор Т3, который открывается, когда в защелку порта записывается «1». Этот транзистор Т3 поддерживает очень маленький суммарный ток, который создает на неподключенном выводе высокий логический уровень.

Если некоторые выводы порта используются как входные, а некоторые как выходные, то программист должен быть осторожен при записи в порт. Особое внимание нужно уделить при использовании команды XOR или любой другой команды «чтение-модификация-запись». В двунаправленном порте возможно считать вместо «1» − «0», если внешнее устройство удерживает на входе уровень меньше U_{IH}.

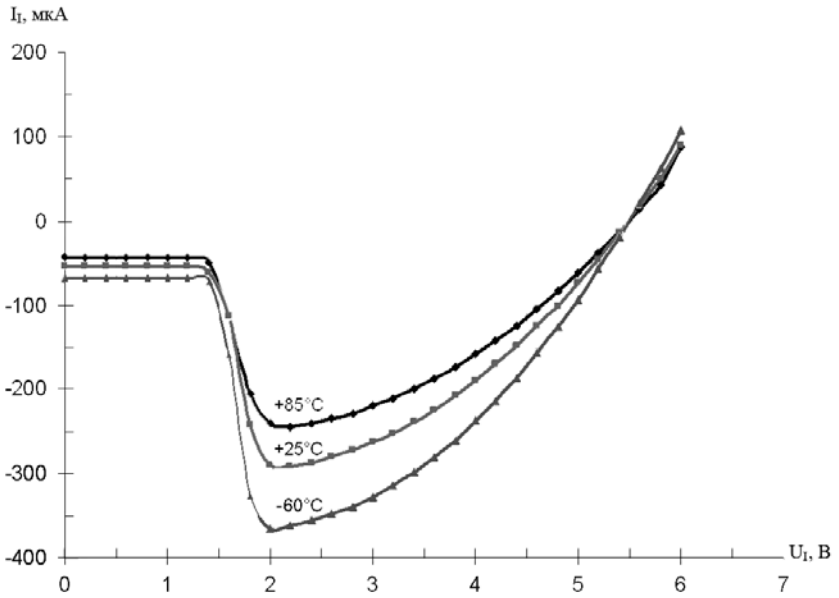


Рисунок 8.4 – Передаточная характеристика квазидвунаправленного выхода

Временные характеристики QBD порта аналогичны соответствующим характеристикам входного и выходного портов. При считывании порта его значение фиксируется во время фазы 1 (CLKOUT имеет низкий уровень) за один такт до разрешения считывания из буфера чтения. Запись в защелку порта изменяет значение на выходе во время фазы 1.

8.1.4 Двунаправленный вывод порта с открытым стоком

Работа двунаправленного порта (ODBD) с открытым стоком аналогична работе QBD порта. Отличие в том, что здесь отсутствуют транзисторы Т3 и Т4. На рисунке 8.5 приведена блок-схема вывода двунаправленного порта с открытым стоком.

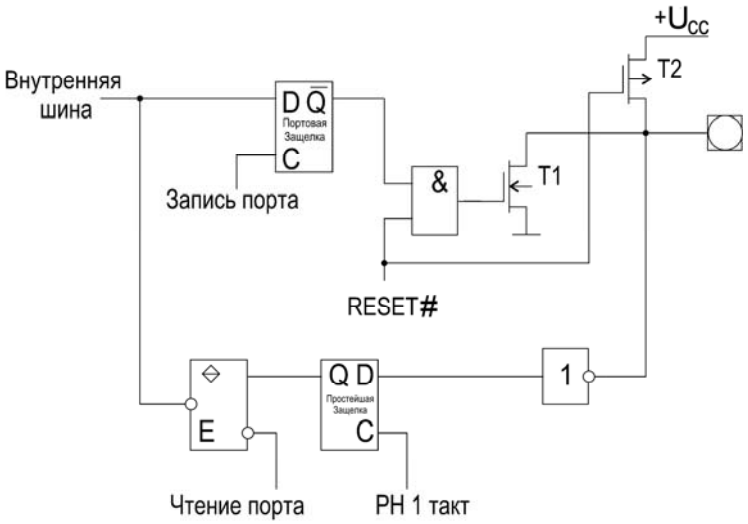


Рисунок 8.5 – Контакт двунаправленного порта с открытым стоком

Запись «0» в защелку порта открывает T1. Пока T1 открыт, его невозможно перевести в состояние высокого логического уровня внешней схемой без нарушения спецификации на устройство. Единица, записанная в защелку порта, закрывает T1 и оставляет вывод неподключенным. Внешним резистором, подсоединенным к U_{CC}, или логической схемой уровень на неподключенном выводе порта можно поднять до уровня логической единицы.

8.2 Программирование портов ввода-вывода

Каждый из пяти портов ввода-вывода связан с SFR для записи или чтения выводов порта. Некоторые порты I/O имеют дополнительный SFR, который реконфигурирует вывод порта для поддержки альтернативных функций. Не все регистры SFR, связанные с портами, доступны для чтения и записи (порт 0 – только для чтения). Кроме того, не все SFR находятся в одном и том же горизонтальном окне. (Например, управляющие регистры SFR записываются в HWindow0, а считываются обратно в HWindow15).

IOPORT0, IOPORT1 и IOPORT2 являются регистрами, используемыми для чтения порта 0, чтения/записи порта 1 и чтения/записи порта 2, соответственно. Эти регистры имеют одинаковый формат, показанный на рисунке 8.6. В таблице 8.2 обобщены операции чтения/записи и условия, которые могут привести к непредвиденному результату.

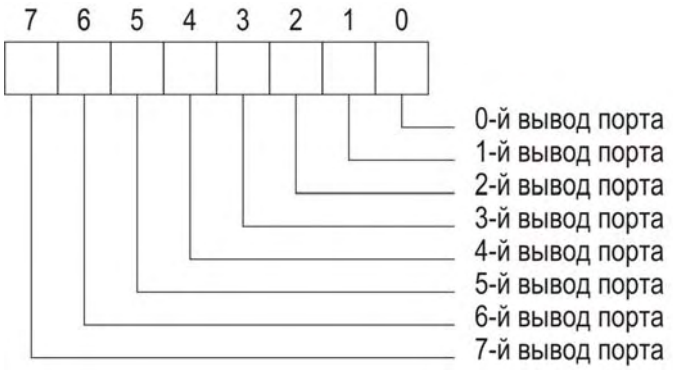


Рисунок 8.6 – Регистр SFR порта ввода-вывода

Таблица 8.2 – Операции чтения /записи порта

Порт	Доступ к порту	Операция чтения	Операция записи
0	IOPORT0	Чтение текущего значения выводов порта. Порт не может считывать, пока идет АЦ преобразование	Невозможна. Запись в IOPORT0 не действует на порт 0, но действует на скорость обращения к контроллеру
1	IOPORT1	Чтение текущего значения выводов порта. В порт была записана «»1, но обратно считывается «0», так как нуль поддерживается внешней схемой	Запись изменяет состояние защелки порта. Изменение состояния защелки не действует на вывод порта, если выбрана его дополнительная функция
2	IOPORT2	Чтение текущего значения выводов порта, исключение составляют выходные выводы. Эти выводы возвращают неопределенное значение, и оно должно быть замаскировано программно	Запись изменяет состояние защелки порта, исключение составляют входные выводы. Они не имеют защелки, поэтому записанное значение игнорируется
3, 4	IOPORT3,4 (1FFEH/1FFFFH)	Чтение текущего значения выводов порта. В порт была записана «1», обратно считывается «0», так как нуль задается извне. Порты 3 и 4 всегда доступны вместе как слово в операции чтения	Изменение состояния защелки. Любое внешнее обращение контроллера к шине игнорирует значение защелки при выполнении команды шины. Порты 3 и 4 всегда доступны вместе как слово операции записи

После сброса все порты I/O, исключая порты 3 и 4, выполняют функции портов I/O. Когда EA# имеет низкий уровень, во время переднего фронта RESET# порты 3 и 4 автоматически переключаются на поддержку системной шины адрес/данные и не могут быть использованы как порты I/O, пока они не будут реконфигурированы. Все или часть выводов портов 0, 1 и 2 обычно конфигурируются на выполнение альтернативных функций.

Использование SFR для конфигурации портов в зависимости от того, какой вывод порта I/O реконфигурируется, в этом разделе не рассматривается. Вместо этого, раздел описывает операции, которые могут выполняться портами I/O, и содержит информацию, необходимую для реконфигурации портов. Смотрите таблицу 8.1, чтобы определить, какие SFR управляют дополнительными функциями портов I/O.

8.2.1 Входной порт

При чтении порта I/O сначала фиксируется, а затем запоминается внутри устройства значение каждого вывода порта. Поэтому при каждом считывании порта фиксируется новое значение, и невозможно получать то же значение порта, считывая его каждый раз. Если значение порта необходимо многократно использовать, то применяют рабочие ячейки памяти для хранения истинного значения. Например, следующая операция считывает текущее значение порта 1 и запоминает его в ячейке памяти, обозначенной P1TEMP:

```
LDB P1TEMP, IOPORT1          ; чтение P1 и сохранение в P1TEMP
```

Определенные выводы порта 2 сконфигурированы как выходы и не содержат схем ввода. После считывания порта в любую ячейку биты SFR, содержащиеся в этой ячейке и являющиеся только выходами, должны быть замаскированы до того, как будет использовано входное значение.

Чтобы использовать вывод QBD порта как входной, необходимо убедиться в том, что транзистор, обеспечивающий мощный низкий уровень на выходе, закрыт. Запись 1 в порт закрывает этот транзистор. Например, команда, приведенная ниже, конфигурирует младшую половину порта 1 для использования в качестве входов:

LDB IOPORT1, #00001111B; запись 1 в порт для использования этих разрядов как входов.

Описание порта 0

Особенность порта 0 в том, что его выводы в одно и то же время могут использоваться как цифровые, так и аналоговые входы (для АЦ преобразования) и выход. Вывод порта 0, использующийся как цифровой вход, имеет высокий импеданс. Однако выводам порта 0, использующимся как аналоговые входы, требуется на короткое время обеспечить достаточный ток, чтобы зарядить внутреннюю эталонную емкость. Это значит, что входные характеристики вывода моментально изменятся, если на этом выводе произошло АЦ преобразование. В любом случае, если порт 0 используется как аналоговый и как цифровой I/O, необходимо подать питание на этот порт через выводы ∩VCC2 и ∩0V2. Вывод 7 порта может применяться для буферизированной выдачи высокостабильного опорного напряжения.

АЦП чувствителен к различного рода помехам. Настоятельно рекомендуем не считывать состояния порта, пока осуществляется АЦ преобразование.

8.2.2 Порт вывода

Запись в порт I/O устанавливает или очищает связанную с ним защелку порта. Выходное значение защелки порта затем используется выходным драйвером. Операция записи относится только к выходному или к двунаправленному (с открытым стоком или квазидвунаправленному) выводу. Запись в порт, работающий только на ввод, не выполняет никаких действий (т. е. защелка не устанавливается и не очищается).

Существует множество способов записи в порт I/O. Порт может быть записан непосредственно с использованием следующих команд:

```
LDB IOPORT1, #10000001B        ; установка 0 и 7 битов, очистка 1-6
LDB intrega, #81H              ; запись значения
STB intrega, IOPORT1           ; вывод значения в порт
```

Обычно непосредственно в порт записывают только для инициализации, или когда все выводы порта необходимо изменить одновременно. Чаще только один вывод необходимо установить, очистить или модифицировать. Чтобы выполнить однобитовые команды, используются команды «чтение-модификация-запись» с текущим значением порта, и выглядит это следующим образом:

```
LDB AL, IOPORT1          ; чтение текущего значения порта
ORB AL, #10000001B       ; установка 0, 7 битов без действующих 1-6 битов
LDB IOPORT1, AL          ; запись нового значения порта
```

Эта операция может быть выполнена одной командой:

ORB IOPORT1, #10000001B ; чтение IOPORT1, его модификация и обратная запись в ioport1

Следующая операция может быть использована, чтобы инвертировать значение вывода порта:

XORB IOPORT1, #1000000B ; дополнение P1.7

Во всех приведенных выше операциях значение порта считывается, модифицируется и записывается обратно. Однако есть случаи, в которых приведенные выше операции не дают желаемого результата. Например, порт 2 имеет два вывода только на выход. Выходные выводы не имеют считывающего буфера, и поэтому текущее значение порта нельзя считать. Ясно, что операция «чтение-модификация-запись» ничего не изменит.

Другой пример – порт 1, имеющий как входы, так и выходы. Все выводы порта 1 квазидвунаправленные, любой вывод, использованный как входной, требуют записи единицы. Это становится настоящей проблемой, когда значение порта считывается,

модифицируется и записывается обратно. Если на любой вход подается нулевое значение с внешнего устройства, то выполнение операции «чтение-модификация-запись» считает и запишет 0 обратно на входной вывод. Запись 0 на квазидвунаправленный вывод открывает драйвер с мощным низким уровнем запуска, и произойдет «короткое замыкание», если единица введется от внешнего устройства.

Решение этой проблемы – отображение значения порта в ячейке памяти. Операция (установка, очистка, переключение) выполняется со значением порта, сохраненным в памяти, которое затем записывается непосредственно в порт.

```
XORB P1IMAGE, #10000000B        ; инверсия отображения P1.7
LDB IOPORT1, P1IMAGE            ; запись нового значения в порт 1
```

Есть ещё один путь решения проблемы – использование операции OR с единицами для тех двунаправленных выводов, которые являются входами. Важно помнить, что одновременное использование входов и выходов одного и того же порта требует особого внимания при модификации выходного значения.

8.2.3 Доступ к портам 3 и 4

Доступ к этим портам легко осуществляется с помощью внутреннего регистра порта 3 и порта 4 (IOPORT34) с адресом 1FFEH. Порты 3 и 4 доступны как слово и как отдельные байты. Чтобы использовать порты 3 и 4 как порты I/O, устройство должно быть установлено на работу с внутренним ПЗУ.

LD и ST команды требуют использования внутренних вспомогательных регистров, это необходимо для переноса информации порта в нижний регистровый файл перед использованием данных. Если данные уже во внутренней памяти, то в команде LD нет необходимости. Пример записи значения слова в порты 3 и 4:

```
LD      intreg, portdata        ; регистр← данные
ST      intreg, 1FFEH           ; регистр → порт 3 и 4
```

Ввод из портов 3 и 4 требует, чтобы сначала записью в регистры портов была установлена конфигурация портов на ввод. Это переводит порты в высокоимпедансное состояние. Следует заметить, что порты при сбросе находятся в режиме входов. Если в порт были записаны нули, то на выводы, которые должны быть входами, необходимо записать единицы. Чтение портов 3 и 4, если в них предварительно были записаны нули, идет в следующем порядке:

```
LD      intrega, #0FFFFH        ; установка порта изменяет режим схемы
ST      intrega, 1FFEH          ; регистр → порты 3 и 4
                                ; ST и LD не
                                ; нужны, если предварительно были
                                ; записаны единицы
LD      intregb, 1FFEH          ; регистр ← порты 3 и 4
```

Форматы команд LD и ST аналогичны, только изменяется направление передачи и приема.

Описание портов 3 и 4

Выводы портов 3 и 4 имеют две функции I/O. Они выполняют функции либо двунаправленного порта с открытым стоком, либо двунаправленной системной шины адрес/данные, либо их комбинации. То, какие функции выполняют порты 3 и 4, зависит от уровня на выводе EA# во время последнего сброса и от необходимости использования внешней памяти программ и данных.

Вывод EA# не только определяет функцию портов 3 и 4, но также определяет, доступны ли порты. Если EA# имеет низкий уровень во время системного сброса, то порты 3 и 4 используются только для поддержки системной шины адрес/данные. Чтение или запись в адрес портов 3 и 4 (1FFEH/1FFFH) интерпретируется как доступ к внешней шине, а не к портам. Если необходимо, порты 3 и 4 можно «перестроить» в соответствии со схемой на рисунке 8.7. Эта схема использует стандартные защелки и драйверы, чтобы

сымитировать функцию I/O с открытым стоком. Доступ к шине для чтения/записи по адресу 1FFEH должен быть декодирован, чтобы разрешить функционирование защелок.

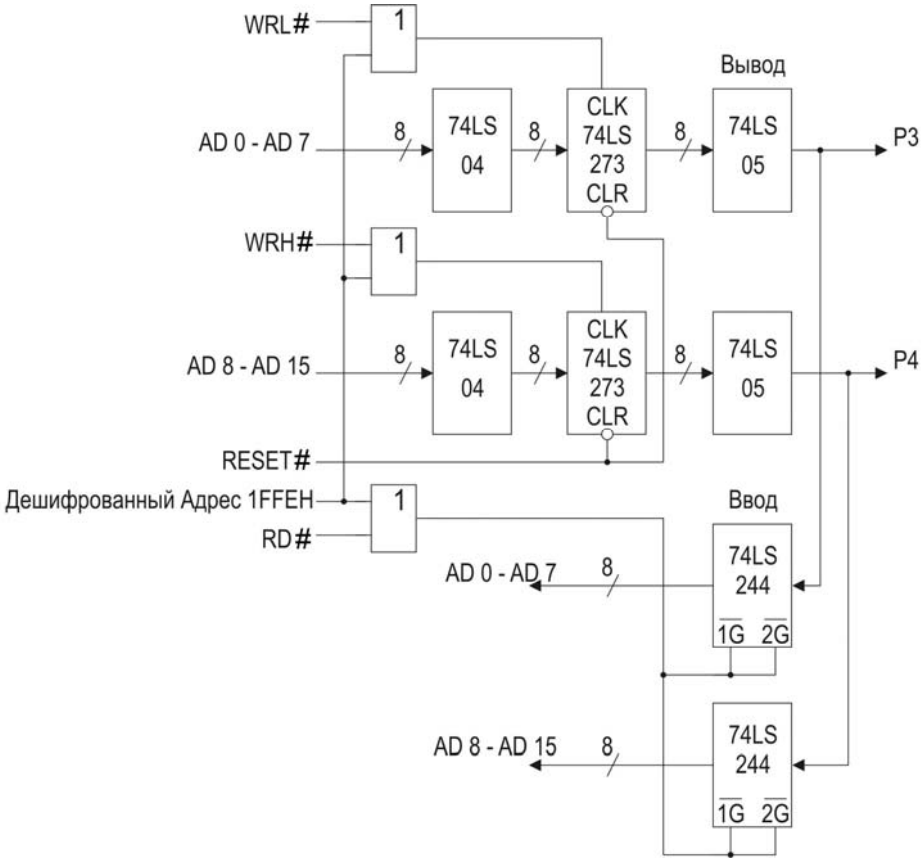


Рисунок 8.7 – Конфигурация портов 3 и 4

Когда EA# имеет высокий уровень во время сброса, порты 3 и 4 имеют две функции: I/O и системную шину адресов/данных. Обычно порты функционируют как I/O с открытым стоком. Ноль, записанный в защелку порта с адресом 1FFEH/1FFFH, открывает транзистор с мощным низким выходным уровнем, тогда как «1», записанная в защелку порта, закрывает выходной транзистор и переводит вывод в «плавающее» состояние. Запись «1» на выводы портов разрешает использовать их как входы. Чтение или запись в порт интерпретируется как внутренний доступ, если не выполняется цикл внешней шины. Порты 3 и 4 остаются портами I/O с открытым стоком все время, пока выполняется цикл внешней шины.

Любой доступ к адресу, не интерпретируемому как внутренний адрес, вызывает выполнение цикла доступа к внешней шине. Чтобы выполнить цикл шины, порты 3 и 4 временно переключаются на функцию поддержки системной шины адрес/данные. Это значит, что выводы портов имеют мощные высокие и низкие уровни во время цикла шины записи адреса и данных и имеют «плавающие» состояния во время цикла шины считывания данных. После завершения цикла выводы портов переключаются обратно на функцию портов I/O. Состояние вывода порта после переключения зависит от последнего значения, записанного в защелку порта.

В проектах, где используется системная шина адрес/данные, значение, запрограммированное в защелке порта, имеет несколько назначений. Значение защелки порта сохраняется на шине во время простоя (когда порты 3 и 4 не являются системной шиной). Запись нуля в порты 3 и 4 означает, что на шине во время простоя сохраняется низкий уровень, тогда как запись единицы означает, что шина находится в “плавающем” состоянии во время простоя. «Плавающая» шина может привести к увеличению потребления тока, но его легко ограничить подключением выводов шины через резисторы

к напряжению питания. Использование шины с низким уровнем снимает необходимость в резисторах, но необходимо быть уверенным, что никакое другое устройство не сможет подключиться к шине в то же самое время (возможно короткое замыкание).

Когда порты 3 и 4 используются исключительно как системная шина адреса/данные, имеющая нагрузочные резисторы, то на шине устанавливается значение 0FFFFH, если команда считывается из несуществующей ячейки памяти. Выборка операнда 0FFH приводит к сбросу устройства. Следует предположить, что выборка команды из несуществующей памяти говорит об аппаратной или программной ошибке, поэтому сброс устройства полезен и предотвращает дальнейшую работу устройства.

8.3 Аппаратное подключение к квазидвунаправленным портам

При использовании QBD портов как входных с возможностью переключения могут понадобиться резисторы, подключенные последовательно, если производится запись в порты после того, как они инициализированы. Например, запись «0» в порт, который подключен к #VCC1 через внешнюю переключательную схему, приведет к короткому замыканию (большой ток потребления). Суммарный ток короткого замыкания для каждого вывода может превысить 20 мА. Этот ток может выйти за допустимые пределы для вывода или порта.

Эту потенциальную проблему можно решить и программно, и аппаратно. При программировании никогда не записывайте «0» на вывод, используемый как вход. Аппаратно подключайте резистор номиналом 1 кОм, соединённый последовательно с каждым выводом, для ограничения тока до допустимого значения. Проблема не так существенна при подключении входов к электронным устройствам (TTL и CMOS) вместо переключательных схем, так как электронные устройства обычно не вызывают чрезмерного суммарного тока.

Запись в QBD порт, с присоединенными к его выводам электронными устройствами, требует особого внимания. Рассмотрим попытку переключения P1.1 в состояние выхода:

```
XORB IOPORT1, #00000010B        ; переключение P1.1
```

Проблема может возникнуть при выполнении команды. Даже если P1.1 в высоком уровне от МК, он может удерживаться в низком уровне. Это обычно случается, если вывод порта подключен к базе n-p-n транзистора, который управляет внешними устройствами. Переключение транзистора происходит при напряжении база-эмиттер чуть выше нуля, обычно около 0,7 В. МК примет это значение за «0», если в порт была записана «1». Поэтому команда XORB запишет «1» в SFR вывода порта и вывод не переключится.

Задачу можно решить аппаратно, используя внешний драйвер. Использование резисторов между выводом порта и базой транзистора часто помогает решить эту проблему, увеличивая напряжение на выводе порта. Программным решением является сохранение байта, который отображает данные, выводимые в порт, в ОЗУ. Всякий раз, когда программе необходимо изменить данные, выводимые в порт, можно модифицировать этот байт и скопировать его в порт.

Если переключатель использует длинную линию для подсоединения к выводу QBD порта, рекомендуется подсоединять его через резистор к #VCC1, чтобы уменьшить возможность возникновения помех и время задержки в линии. На очень длинных линиях с низкоскоростным обменом для уменьшения помех в дополнение к резистору полезен и конденсатор.

9 Управление синхронизацией периферийных устройств

Микроконвертер имеет функцию управления синхронизацией внутренних периферийных устройств. Если пользователь не использует какое-либо периферийное устройство, то его можно отключить, тем самым уменьшив общее потребление. Включение/выключение происходит записью соответствующих битов в регистры управления CLKCL и CLKCH, котрые находятся в области расширенного SFR по адресу 1FF0H и 1FF1H соответственно. Также в регистре CLKCL находится бит управления режимом SLOW. В SLOW режиме увеличивается длительность машинного цикла в два раза по сравнению с нормальным режимом работы ИС. Структура схемы синхронизации представлена на рисунке 9.1.

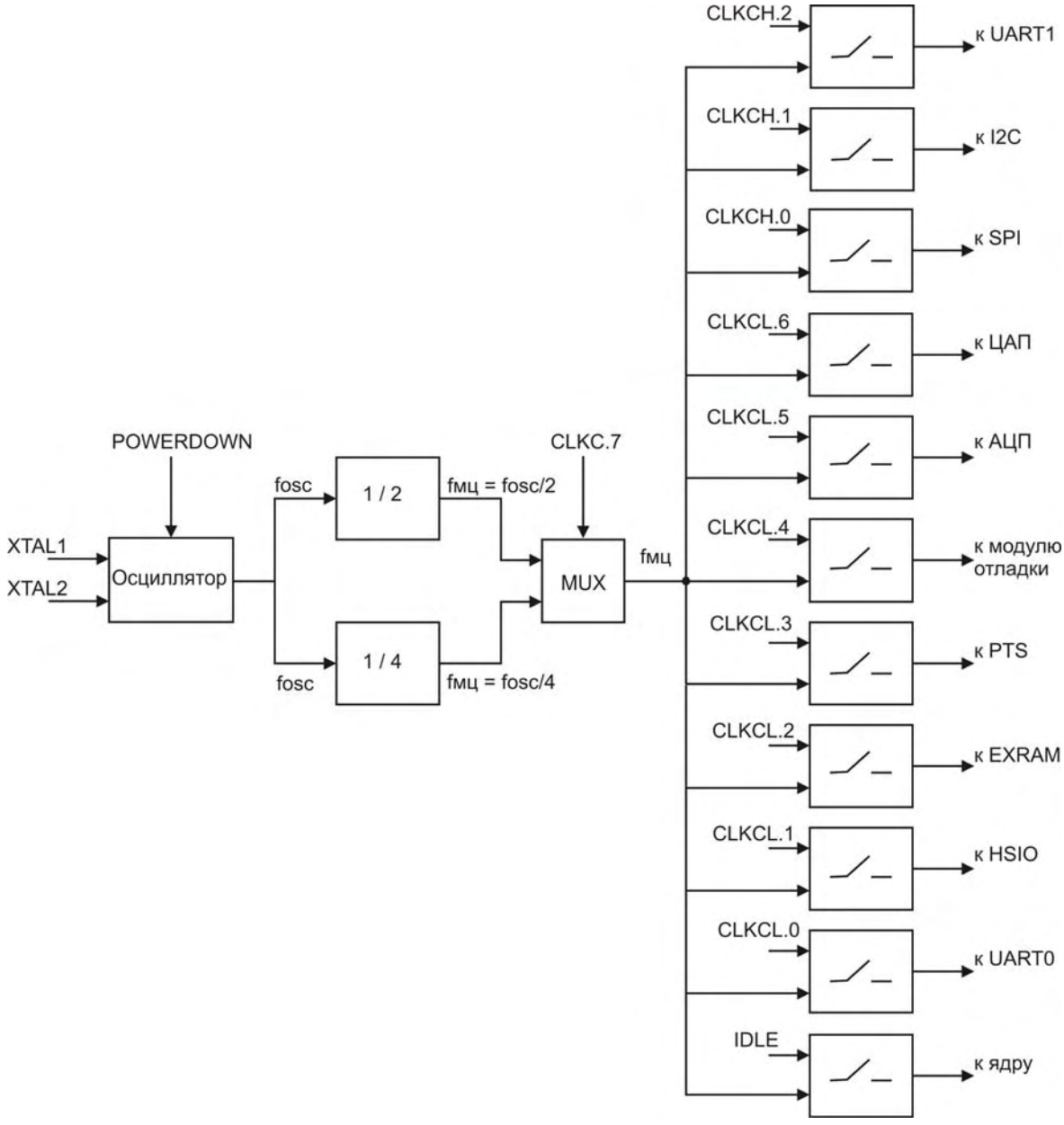


Рисунок 9.1 – Структура схемы синхронизации

В таблице 9.1 представлено время включения/выключения периферийных устройств, а также значения управляющих битов регистра CLKC. При выключении устройств запись

в их регистры невозможна, а при чтении будет выдаваться последнее значение, принятое регистрами перед отключением.

Таблица 9.1 – Параметры включения/выключения периферийных устройств

Управляющий бит регистра CLKC	Значение по умолчанию*	Время включения/выключения
1	2	3
CLKCL.0	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCL.1	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCL.2	1	Включение – один машинный цикл, выключение – один машинный цикл. При выключении адреса EXRAM назначаются во внешнюю память
CLKCL.3	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCL.4	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCL.5	1	Включение – 1 800 машинных циклов с момента первой команды на преобразование, которая может подаваться сразу же после установки бита CLKC.5, выключение – один машинный цикл
CLKCL.6	0	Включение – 350 машинных циклов (при 33 МГц), выключение – один машинный цикл
CLKCL.7	0	Включение SLOW режима один машинный цикл, выключение – один машинный цикл
CLKCH.0	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCH.1	1	Включение – один машинный цикл, выключение – один машинный цикл
CLKCH.2	1	Включение – один машинный цикл, выключение – один машинный цикл

_____

* «1» – включено, «0» – выключено.

При отключении тактовых сигналов с блоков АЦП и ЦАП, эти блоки автоматически переводятся в режим SLEEP, что позволяет существенно уменьшить энергопотребление схемы. Однако следует помнить, что выход из этих режимов – длительный процесс и определяется временем срабатывания аналоговых частей соответствующих устройств.

Не рекомендуется выключение периферийных устройств во время их работы, так как это может привести к непредсказуемым последствиям. Выводы схемы, связанные с отключаемыми устройствами, останутся в состоянии, которое было на момент отключения.

Модуль отладки может генерировать событие, при котором происходит перезапись управляющих регистров CLKCL и CLKCH. Этот режим предназначен для остановки необходимых периферийных устройств с целью сохранения в связанных с устройством регистрах результатов.

В целях уменьшения энергопотребления после сброса цифро-аналоговый преобразователь находится в режиме пониженного потребления. Для включения необходимо установить бит CLKCL.6. При выключении ЦАП возможно как чтение, так и запись связанного с ним управляющего регистра.

10 Модуль отладки (OCDS)

Модуль отладки предназначен для упрощения процесса отладки программного обеспечения пользователей или для контроля хода выполнения программы. Также может использоваться для защиты от сбоя программы, в случае, если она находится в EEPROM и не использует обращение к внешней памяти, или известно максимальное значение счетчика команд, которое может быть достигнуто при корректной работе системы. Блок формирует несколько типов откликов по нескольким типам событий. События могут быть следующие:

- доступ 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.

Каждое из данных событий может формировать следующие отклики:

- прерывание DEBUG (2060H);
- перевод в режим IDLE;
- сброс микроконвертера;
- перезагрузка регистров управления тактовыми сигналами периферии.

В таблице 10.1 представлены управляющие регистры модуля отладки.

Таблица 10.1 – Управляющие регистры модуля отладки

Обозначение	Название	Описание
1	2	3
DEBPC	Регистр задания значения основного счетчика команд 1	Хранит значение основного счетчика команд, при превышении которого может формироваться отклик модулем отладки
DEBPCEQ	Регистр задания значения основного счетчика команд 2	Хранит значение основного счетчика команд, при котором может формироваться отклик модулем отладки
DEBDAT	Регистр задания значения данных	Хранит значение данных (полного слова), при появлении которого на шине данных может формироваться отклик модулем отладки
DEBDATL	Регистр задания младшего байта данных	Хранит значение данных (младшего байта), при появлении которого на шине данных может формироваться отклик модулем отладки
DEBDATH	Регистр задания старшего байта данных	Хранит значение данных (старшего байта), при появлении которого на шине данных может формироваться отклик модулем отладки
DEBADDR	Регистр задания значения адреса операнда	Хранит значение полного адреса операнда (16 разрядов) при появлении которого на шине адресов может формироваться отклик модулем отладки
DEBCTRL	Регистр управления событиями	Выбирает тип отлика, который будет формироваться при наступлении событий

Окончание таблицы10.1

1	2	3
CURRPC	Регистр текущего значения PC	Содержит текущее значение основного счетчика команд
HAPPPC	Регистр захвата PC	Содержит адрес основного счетчика команд в момент возникновения события
CLKCLDEB	Регистр перезагрузки CLKCL	Содержит значение, которым может быть перезагружен регистр управления тактовыми сигналами периферийных устройств CLKCL
CLKCHDEB	Регистр перезагрузки CLKCH	Содержит значение, которым может быть перезагружен регистр управления тактовыми сигналами периферийных устройств CLKCH

На рисунке 10.1 представлена структурная схема возникновения событий в модуле отладки.

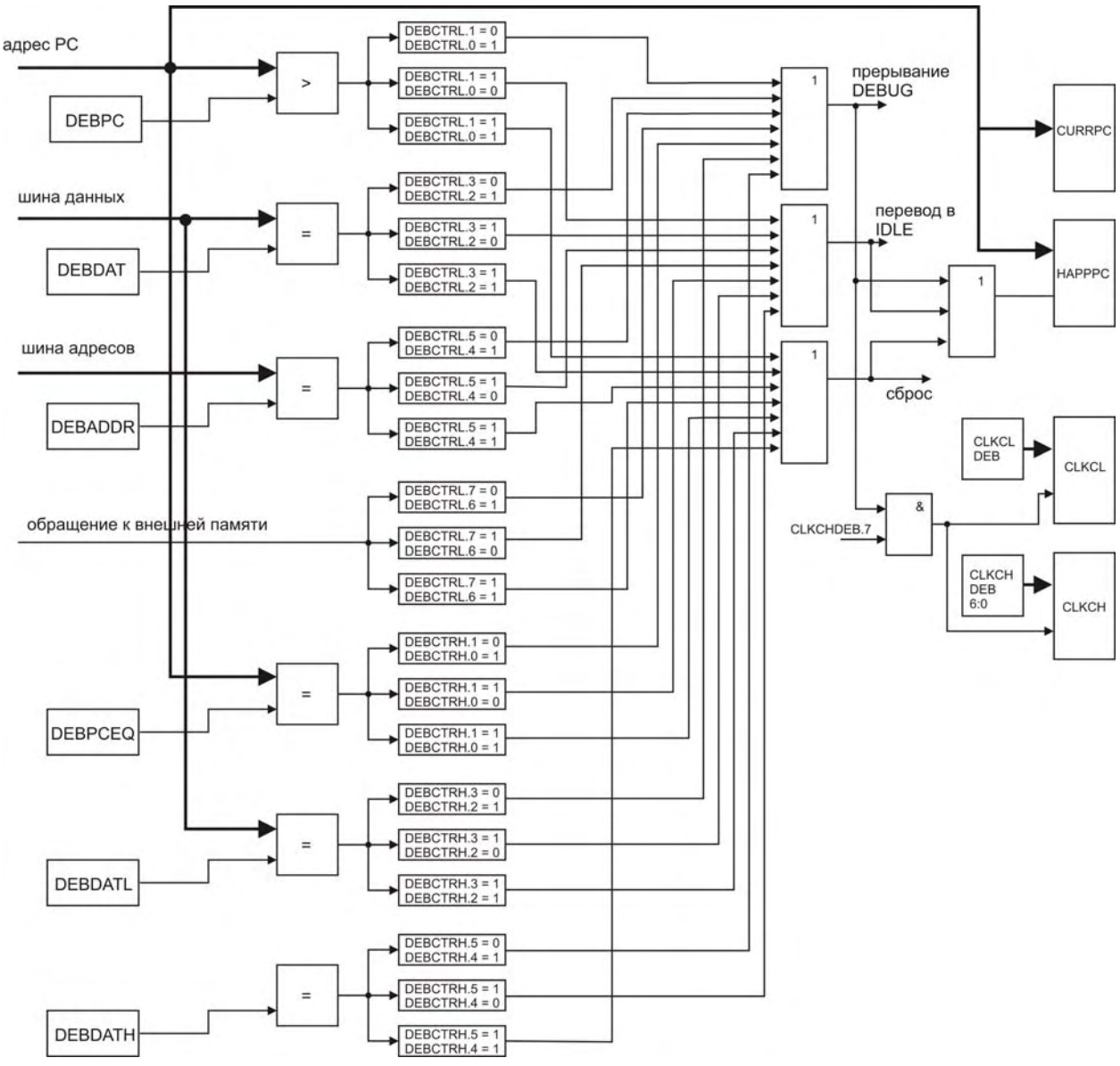


Рисунок 10.1 – Структурная схема работы модуля отладки

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.

При возникновении событий текущее значение счетчика команд записывается в регистр HAPPPC. Регистр HAPPPC не изменяет свое значение во время аппаратного сброса, поэтому может фиксировать адрес, при котором наступило событие «сброс».

Одновременно с возникновением прерывания от модуля отладки, может осуществляться перезапись регистров управления тактовыми сигналами периферийных устройств. Это позволяет отключить необходимый блок. При этом управляющие и статусные регистры не будут изменяться, а сохранят значение, которое принимали на момент отключения. В последующем возможно только чтение содержимого регистров. Существует возможность последующего включения периферийного устройства записью необходимого значения в регистры CLKCL и CLKCH. Бит 7 регистра CLKCHDEB разрешает/запрещает перезапись управляющих регистров тактовых сигналов периферийных устройств при возникновении прерывания от модуля отладки.

Стоит отметить, что адрес операнда в регистре DEBADDR нужно указывать полностью, а данные (в регисте DEBDAT) указываются пословно. Значение счетчика команд во время 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).

11 Последовательные порты ввода-вывода UART

Последовательные порты ввода-вывода UART микроконвертера 1874BE96T являются синхронно/асинхронными портами, которые включают в себя универсальный асинхронный приемник и передатчик UART. Порты имеют синхронный режим (Mode 0) и три асинхронных режима (Mode 1, Mode 2, Mode 3) как для приема, так и для передачи.
В этом разделе дан обзор каждого режима и описывается, как программировать последовательные порты.

Чтобы помочь пониманию каждого режима, следующие абзацы кратко описывают регистры, связанные с последовательными портами ввода-вывода.

SP_CON0 и SP_CON1 (SP_CONx) регистры выбирают режим обмена, разрешают и запрещают прием, проверку четности и передачу девятого бита данных (смотри подраздел 11.3).

SP_STAT0 и SP_STAT1 (SP_STATx) регистры содержат флаги “прерывание по приему” (RIx) и «прерывание при передаче» (TIx) и три других статусных бита (смотри подраздел 11.4). Следует отметить, что чтение регистров SP_STATx очищает все флаги, исключая TXEx. По этой причине рекомендуется записывать содержимое SP_STATx в рабочий регистр и выполнять команды, тестирующие биты, такие как JBC и JBS над рабочим регистром. Иначе, если будет выполнено более одной команды, тестирующей биты, возможен возврат неверного значения.

Последовательные порты принимают данные в SBUF(RX0) и SBUF(RX1) (SBUF(RXx)) регистры, а передают данные через SBUF(TX0) и SBUF(TX1) (SBUF(TXx)) регистры.

Доступ к регистрам приема и передачи для чтения и записи осуществляется в режиме горизонтальных окон и косвенной адресацией (подраздел 6.6).

Приемник и передатчик буферированы, чтобы поддерживать процесс передачи и позволить принять второй байт до того, как первый байт был считан.

Прерывания последовательных портов разрешаются и запрещаются через регистры масок (INT_MASK или INT_MASK1); ожидаемые прерывания индицируются через регистр ожидания прерывания (INT_PEND или INT_PEND1) (пункт 11.3.3).

Независимый генератор скорости управляет скоростью обмена последовательного порта. Сигналы синхронизации – либо XTAL1, либо T2CLK. BAUD_RATE0 и BAUD_RATE1 (BAUD_RATEx) регистры выбирают скорость обмена и источник синхронизации (пункт 11.3.4). Следует обращаться к приложению Б за информацией о сигналах, обсуждаемых в этом разделе.

11.1 Режим 0 (MODE 0)

Наиболее частым использованием режима 0, синхронного режима является расширение функций ввода-вывода МК через сдвиговые регистры. Схема сдвигового регистра для режима 0 приведена на рисунке 11.1. В этом режиме на вывод TXDx выводится последовательность из восьми тактовых импульсов, пока вывод RXDx принимает или передает данные. Передаются 8-битные данные, младший значащий бит является первым во времени. На рисунке 11.2 приведена диаграмма распределения этих сигналов во времени. Следует заметить, что только режим 0 использует RXDx как выход с открытым стоком. В режиме 0 RXDx должен быть запрещен для начала передачи и разрешен для начала приема (см. пункт 11.3.2). Когда RXDx запрещен, записанное в SBUF (TXx) начинает передаваться. Когда RXDx разрешен, начинается прием либо по возрастающему фронту на RXDx выводе, либо при обнулении флага RIx.

Запрещение RXDx останавливает процесс приема и запрещает дальнейшие приемы. Чтобы избежать частичного или нежелательного завершения приема, следует запрещать RXDx только перед очисткой RIx флага. Это можно осуществить в обработке прерывания использованием программных флагов или в коде основной программы использованием регистра ожидания прерывания для определения окончания приема.

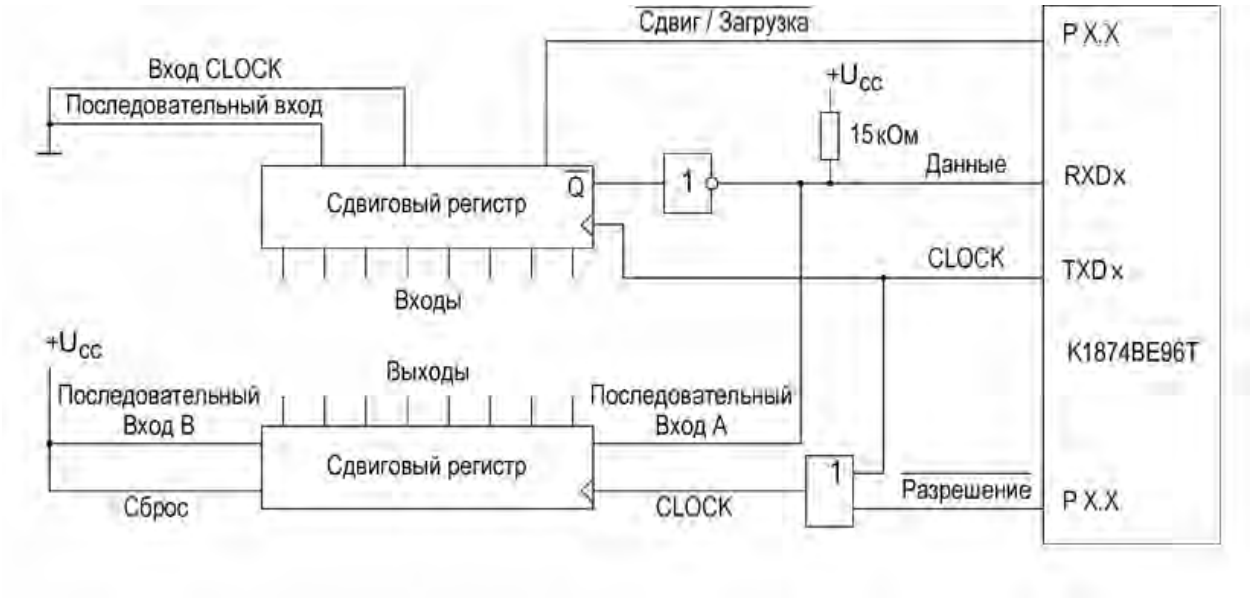


Рисунок 11.1 – Схема сдвигового регистра для режима 0

Во время приема RIx флаг устанавливается за время одного бита, после того как принят последний бит. Сигнал прерывания по приему генерируется непосредственно перед тем, как устанавливается флаг RIx. Во время передачи TIx флаг устанавливается сразу после конца передачи последнего (восьмого) бита данных. Сигнал прерывания при передаче генерируется, если TIx флаг установлен.

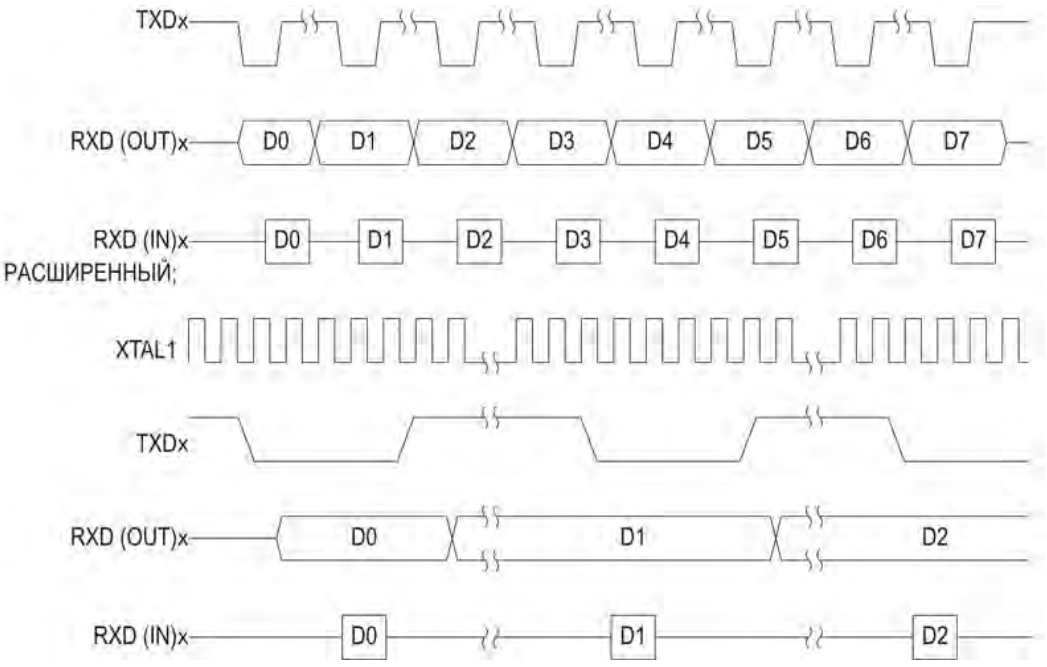


Рисунок 11.2 – Временные диаграммы для режима 0

11.2 Асинхронные режимы

Режимы 1, 2, 3 являются полнодуплексными режимами последовательной передачи/приема, это означает, что прием и передача в них могут происходить одновременно. Режим 1 является стандартным 8-битным асинхронным режимом, который используется в обычных последовательных коммуникациях. Режимы 2 и 3 – 9-битные асинхронные режимы, обычно используемые для межпроцессорных коммуникаций

(пункт 11.2.5). В режиме 2 последовательный порт генерирует прерывание, только если девятый бит равен единице. В режиме 3 последовательный порт генерирует прерывание всегда по окончанию передачи или приема данных.

Если последовательный порт сконфигурирован в режимы 1, 2 или 3, то запись в SBUF (TXx) вызывает начало передачи данных. Новые данные, размещенные в SBUF (TXx), не передаются до тех пор, пока не будет послан стоп-бит предыдущих данных. Спадающий фронт сигнала на входе RXDx вынуждает последовательный порт начать прием данных, если RXDx разрешен. Запрещение RXDx останавливает процесс приема и запрещает дальнейшие приемы (пункт 11.3.2).

11.2.1 Режим 1

Режим 1 является стандартным режимом асинхронной связи. Блок данных, используемых в этом режиме (рисунок 11.3), содержит 10 бит: стартовый бит «0», восемь битов данных (младший значащий (LSB) первый), стоповый бит «1». Если четность разрешена, то бит дополнения до четности (even parity) посылается взамен восьмого бита данных, а контроль четности проверяется при приеме.

Передача и прием управляются отдельными сигналами синхронизации сдвига. Синхросигнал сдвига передачи запускается, если инициализирован генератор скорости передачи. Сигнал сдвига для приема возобновляется, если зафиксирован перепад из 1 в 0 (стартовый бит). Таким образом, синхронизации передачи и приема может и не быть, хотя оба процесса идут на одной частоте.

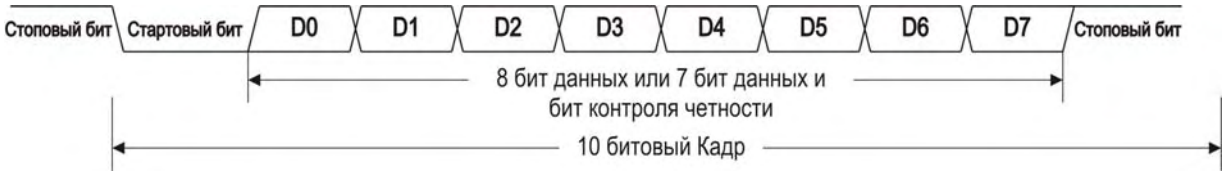


Рисунок 11.3 – Кадр последовательного порта в режиме 1

Флаги прерывания при передаче (TIx) и приеме (RIx) устанавливаются для индикации окончания операций. Во время приема флаг RIx устанавливается как раз перед окончанием стопового бита. Сигнал «прерывание по приему» генерируется, если RIx флаг установлен. Во время передачи TIx флаг устанавливается в начале стопового бита. Сигнал «прерывание для передачи» генерируется, если TIx флаг установлен. Следующий байт не может быть послан, пока не будет послан стоповый бит.

Примечание – Необходимо внимание, если соединяются более чем два устройства по последовательному каналу в полудуплексе (когда передача и прием осуществляются по одному проводу). Принимающий процессор перед началом передачи должен задержать обмен на время одного бита после установки RIx флага. Иначе передача может исказить стоповый бит, вызывая проблемы в другом устройстве, подсоединенном как приёмник.

11.2.2 Режим 2

Режим 2 – асинхронный режим, распознающий девятый бит. Этот режим обычно используется, как и режим 3, для мультипроцессорных связей. На рисунке 11.4 показан формат посылки этого режима. Он содержит стартовый бит «0», девять бит данных (LSB первый) и стоповый бит «1». При передаче в качестве девятого передаваемого бита устанавливается бит TB8x в регистре SP_CONx перед записью регистра SBUF (TXx). Бит TB8x очищается аппаратно после каждой передачи, поэтому он должен быть установлен (если необходимо) перед каждой записью в SBUF (TXx). Во время приема флаг RIx установлен, а прерывание генерируется, только если бит TB8x установлен. Это обеспечивает простой способ селектирования приема на линии данных (пункт 11.2.5). Четность в этом режиме не может быть разрешена.

Рисунок 11.4 – Кадр последовательного порта в режимах 2 и 3

11.2.3 Режим 3

Режим 3 – асинхронный 9-битный режим. Формат посылки в этом режиме идентичен режиму 2. Режим 3 отличается от режима 2 тем, что во время передачи может быть разрешена четность, и в этом случае девятый бит становится битом дополнения до четности.

Когда четность запрещена, биты 0 – 7 данных записываются в буфер передачи последовательного порта, а девятый бит данных – в бит 4 (TB8x) в регистре SP_CONx. В режиме 3 прием всегда вызывает прерывание, независимо от состояния девятого бита. Если четность запрещена, девятый бит может быть считан в бит 7 (RB8x) регистра SP_STATx. Если четность разрешена, то RB8x становится флагом ошибки по четности при приеме (RPEx).

11.2.4 Временные соотношения в режимах 2 и 3

Работа в режимах 2 и 3 сходна с работой в режиме 1. Единственное отличие – это то, что данные содержат девятый бит, а при передаче и приеме пакет содержит 11 бит. Во время приема RIx флаг устанавливается сразу после конца стопового бита. Сигнал прерывания генерируется, когда флаг RIx установлен. Во время передачи TIx флаг устанавливается в начале стоп-бита. Сигнал прерывания для передачи генерируется, когда флаг TIx установлен. Девятый бит может быть использован для передачи четности или мультипроцессорных связей.

11.2.5 Мультипроцессорные связи

Режимы 2 и 3 обеспечивают мультипроцессорные связи. В режиме 2 последовательный порт генерирует прерывание по приему (RIx), только если установлен девятый бит. В режиме 3 последовательный порт генерирует прерывание по приему (RIx) независимо от значения девятого бита. Девятый бит всегда устанавливают при передаче адреса и очищают при посылке данных. Один из вариантов использования этих режимов для мультипроцессорных связей – установить главный процессор в режим 3, а ведомый – в режим 2. Когда главный процессор хочет передать блок данных в один из нескольких ведомых, он посылает адресный блок, который идентифицирует нужный ведомый. Адресный блок будет прерывать все ведомые, так как девятый бит установлен. Каждый ведомый может проверить байт адреса и понять, его ли это адрес. Адресованный ведомый переключается в режим 3 для получения блока данных, а неадресованные ведомые остаются в режиме 2 и не прерываются.

11.3 Программирование последовательных портов

В таблице 11.1 показаны регистры, которые воздействуют на работу и функции последовательных портов.

Таблица 11.1 – Управление последовательными портами и регистры состояния

Обозначение регистров	Название регистров	Описание
1	2	3
BAUD_RATE0	Регистр скорости обмена UART0	Данным регистром выбирается скорость обмена последовательного порта и исходная синхронизация. Регистр должен заполняться двумя байтами, первым – младший значащий байт. Старший значащий байт определяет источник синхронизации. Младшие 15 бит содержат BAUD_VALUE0, беззнаковое целое число, определяющее скорость обмена
BAUD_RATE1	Регистр скорости обмена UART1	Данным регистром выбирается скорость обмена последовательного порта и исходная синхронизация. Регистр должен заполняться двумя байтами, первым – младший значащий байт. Старший значащий байт определяет источник синхронизации. Младшие 15 бит содержат BAUD_VALUE1, беззнаковое целое число, определяющее скорость обмена
IOC1	Регистр 1 управления вводом-выводом	Установка бита 5 в этом регистре разрешает функцию TXD порта P2.0 и P2.6
SBUF (RX0)	Буфер приёма последовательного порта UART0	Регистр содержит данные, полученные из последовательного порта 0
SBUF (RX1)	Буфер приёма последовательного порта UART1	Регистр содержит данные, полученные из последовательного порта 1
SBUF (TX0)	Буфер передачи последовательного порта UART0	Регистр содержит данные, готовые для передачи. В режимах 1, 2 и 3 запись в SBUF (TX0) начинает передачу. В режиме 0 запись в SBUF (TX0) начинает передачу только если прием запрещён (SP_CON0.3 = 0)
SBUF (TX1)	Буфер передачи последовательного порта UART1	Регистр содержит данные, готовые для передачи. В режимах 1, 2 и 3 запись в SBUF (TX1) начинает передачу. В режиме 0 запись в SBUF (TX1) начинает передачу только если прием запрещён (SP_CON1.3=0)
SP_CON0	Регистр управления последовательным портом UART0	Регистр выбирает режим связи, разрешает и запрещает прием, а так же проверку дополнения до четности и передачу девятого бита. Бит TB80 очищается после каждой передачи
SP_CON1	Регистр управления последовательным портом UART1	Регистр выбирает режим связи, разрешает и запрещает прием, а так же проверку дополнения до четности и передачу девятого бита. Бит TB8 очищается после каждой передачи
SP_STAT0	Регистр состояния последовательного порта UART0	Регистр содержит биты состояния последовательного порта 0. В нем есть биты состояния для ошибки переполнения при приеме (OE0), опустошения буфера передачи (TXE0), ошибки кадра (FE0), прерываний по передаче (TI0), прерываний по приему (RI0), а также ошибки четности при приеме (RPE0) или приёма 8 бита (RB80). Чтение из SP_STAT0 обнуляет биты OE0, FE0, TI0, RI0 и RPE0/RB80. Запись байта в SBUF(TX0) очищает бит TXE0

Окончание таблицы11.1

1	2	3
SP_STAT1	Регистр состояния последовательного порта UART1	Регистр содержит биты состояния последовательного порта 1. В нем есть биты состояния для ошибки переполнения при приеме (OE1), опустошения буфера передачи (TXE1), ошибки кадра (FE1), прерываний по передаче (TI1), прерываний по приему (RI1), а также ошибки четности при приеме (RPE1) или приёма 8 бита (RB81). Чтение из SP_STAT1 обнуляет биты OE1, FE1, TI1, RI1 и RPE1/RB810. Запись байта в SBUF(TX1) очищает бит TXE1
Примечание – Всегда следует записывать нули в зарезервированные биты этих регистров.		

11.3.1 Выбор режима связи и разрешение проверки чётности

Биты 0 и 1 в регистре SP_CONx задают режим последовательного порта (рисунок 11.5 и таблица 11.2). Выбор нового режима сбрасывает последовательный порт и прерывает процесс передачи (приема) в канале. Установка бита 2 разрешает проверку четности (parity). Установка бита 3 разрешает функцию RXD на выводе P2.1 или P2.4.
Бит 4 является девятым битом данных, который передается в режимах 2 и 3, или, если разрешена проверки четности, это дополнение до четности. Бит 4 (TB8x) очищается после каждой передачи.

Таблица 11.2 – Выбор режима

Бит 1	Бит 2	Выбранный режим
0	0	Режим 0 (Mode 0)
0	1	Режим 1 (Mode 1)
1	0	Режим 2 (Mode 2)
1	1	Режим 3 (Mode 3)

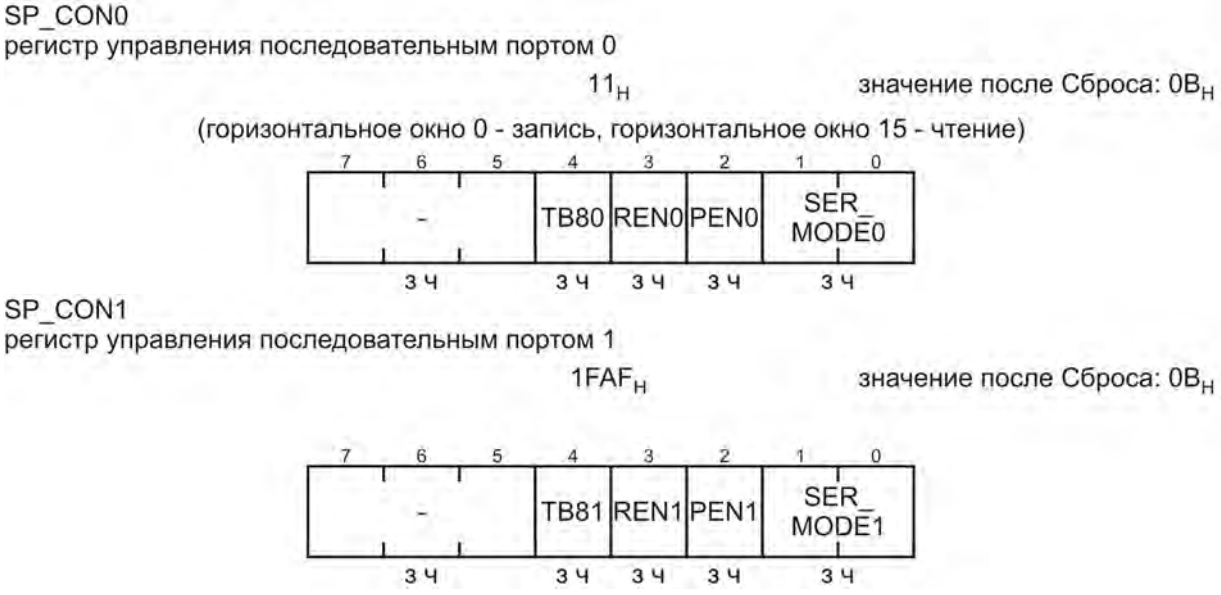


Рисунок 11.5 – Формат регистров SP_CONx

11.3.2 Конфигурирование TXDx и RXDx

В таблице 11.3 показаны выводы, связанные с последовательными портами. Чтобы разрешить TXDx, следует установить бит 5 регистра IOC1; чтобы запретить TXDx, следует обнулить этот бит. Разрешение RXDx определяется установкой бита 3 регистра SP_CONx; для запрещения RXDx следует обнулить этот бит (пункт 11.3.4 – для разрешения T2CLK). При разработке аппаратуры следует учитывать разницу в нагрузочной способности выводов TXD0 и TXD1. Так как TXD1 делит вывод с квазидвунаправленным портом, то нагрузочная способность ограничивается его возможностями.

Таблица 11.3 – Сигналы последовательных портов

Имя функции	Дополнительная функция	Выбор	Тип вывода	Описание
RXD0	P2.1/PALE#	SP_CON0.3 = «1»	I/O	Прием последовательных данных. В режимах 1, 2 и 3 RXD0 – вход для приема данных. В режиме 0 это как вход, так и выход с открытым стоком для данных
RXD1	P2.4/AINC#/ T2RST	SP_CON1.3 = «1»	I/O	Прием последовательных данных. В режимах 1, 2 и 3 RXD0 – вход для приема данных. В режиме 1 это как вход (но не с открытым стоком), так и выход для данных
T2CLK	P2.3	BAUD_RATEx MSB = «0»	I	Синхровход таймера 2 и один из двух возможных источников синхросигналов для входов генераторов скорости обмена
TXD0	P2.0	IOC1.5 = «1»	O	Передача последовательных данных. В режимах 1, 2 и 3 TXD0 – выход данных, передаваемых последовательным портом. В режиме 0 – это выход синхросигнала
TXD1	P2.6/T2UPDN/ CPVER	IOC1.5 = «1»	O	Передача последовательных данных. В режимах 1, 2 и 3 TXD1 – выход данных, передаваемых последовательным портом. В режиме 0 – это выход синхросигнала

11.3.3 Разрешение работы по прерываниям для последовательного порта

Последовательный порт может быть сконфигурирован на генерацию либо единственного прерывания Serial Ports, либо двух: прерывания для передачи (TI) и прерывания по приему (RI). Если прерывание Serial Ports замаскировано, а прерывание для передачи и приема разрешены, генерируются раздельные прерывания по RIx и TIx флагам.

Для разрешения отдельных прерываний установите биты TI_MASK и RI_MASK в регистре INT_MASK1 (таблица 11.4). Для разрешения прерываний последовательных портов (Serial Ports) следует установить бит SER_MASK в регистре INT_MASK (см. раздел 7 для получения дополнительной информации о прерываниях).

Таблица 11.4 – Регистры прерываний последовательных портов

Обозначение регистра	Имя регистра	Адрес	Описание
INT_MASK	Регистр маски прерывания	08H	Установка бита 6 этого регистра разрешает прерывание Serial Ports последовательного порта (INT06, 200CH). Очистка бита 6 запрещает (маскирует) прерывание
INT_MASK1	Регистр маски прерывания 1	13H	Установка бита 0 этого регистра разрешает прерывание для передачи (INT08, 2030H), очистка бита 0 запрещает (маскирует) прерывание. Установка бита 1 этого регистра разрешает прерывание по приему (INT09, 2032H), очистка бита 1 запрещает (маскирует) прерывание
INT_PEND	Регистр ожидания прерывания	09H	Установка бита 6 этого регистра индицирует ожидание прерывания Serial Ports (INT06) и обнуляется при переходе по вектору прерывания 200CH
INT_PEND1	Регистр ожидания прерывания 1	12H	Установка бита 0 индицирует ожидание передачи (INT09) и обнуляется по вектору прерывания 2030CH. Установка бита 1 индицирует ожидание прерывания по приему (INT09) и обнуляется по вектору прерывания 2032H

11.3.4 Программирование скорости обмена и источника синхронизации

Регистры BAUD_RATEx выбирают вход синхронизации для генератора и определяют скорость обмена для всех режимов ввода-вывода. Эти двухбайтные регистры должны загружаться побайтно. Первым всегда загружается младший значащий байт.

Старший значащий бит выбирает один из двух возможных источников синхронизации для генератора скорости. Чтобы выбрать входной сигнал от XTAL1 (f_{CI}) как источник синхронизации, установите бит 15; для выбора внешнего сигнала на вывод T2CLK очистите бит 15. Максимальная частота на входе T2CLK – $f_{CI}/4$.

Младшие 15 бит представляют переменную BAUD_VALUE как беззнаковое целое, которое соответствует скорости обмена. Максимальное значение BAUD_VALUE 7FFFH (32767 десятичное). Минимальное значение в режимах 1, 2 и 3 – это 0000H, если XTAL1 является генератором синхронизации, иначе оно – 0001H. Минимальное значение в режиме 0 всегда 0001H.

Для задания скорости обмена через определения BAUD_VALUE необходимо использовать следующие соотношения:

Синхронный режим 0:

$$BAUD\_VALUE = (f_{CI} / BAUD\_RATE \times 2) - 1 \text{ или } T2CLK / BAUD\_RATE$$

Асинхронные режимы 1, 2 и 3:

$$BAUD\_VALUE = (f_{CI} / BAUD\_RATE \times 16) - 1 \text{ или } T2CLK / BAUD\_RATE \times 8$$

В таблице 11.5 приведены стандартные значения BAUD_RATE (скоростей обмена), когда используется 16 МГц синхронизация по входу XTAL1. Из-за округления формула для вычисления BAUD_VALUE неточна, и полученное значение скорости обмена слегка

отличается от требуемого. В таблице также приведены проценты ошибок, получаемые при использовании эталонного значения BAUD_RATE. В большинстве случаев последовательная связь будет работать при разнице до 5 % в скоростях приема и передачи.

Таблица 11.5 – Значение BAUD_RATE, если используется XTAL1 в 16 МГц

Скорость обмена	BAUD_RATE*		% ошибки	
	Режим 0	Режимы 1, 2, 3	Режим 0	Режимы 1, 2, 3
9600	8340H	8067H	0,04	0,16
4800	8682H	80CFH	0,02	0,16
2400	8D04H	81A0H	0,01	0,08
1200	9A0AH	8340H	0	0,04
300	0E82BH	8D04H	0	0,01

_____

* Бит 15 всегда установлен, если XTAL1 выбран как источник синхронизации генератора скорости обмена.

11.4 Состояние последовательных портов

Состояние последовательных портов отражается в регистрах SP_STATx (рисунок 11.6). Следует заметить, что чтение регистров SP_STATx очищает все биты кроме TXEx. По этим причинам рекомендуется скопировать содержимое регистров SP_STATx в рабочий регистр, а затем выполнять команды, тестирующие биты (такие как JBC и JBS), с рабочим регистром. Иначе флаги будут очищены при выполнении команд, тестирующих биты.

Приемник МК проверяет наличие стопового бита. Если стопового бита нет в требуемое время, устанавливается ошибка кадра (FEx). Если стоповый бит обнаружен, данные загружаются из сдвигового регистра приемника в SBUF(RXx) и устанавливается флаг прерывания по приему (RIx). Если это произошло прежде, чем считан предыдущий байт из SBUF(RXx), устанавливается бит ошибки переполнения (OEx). SBUF(RXx) всегда содержит последний принятый байт и никогда – комбинацию из двух байт.

Флаг прерывания по приёму (RIx) показывает, получен ли принимаемый байт данных. Флаг прерывания для передачи (TIx) показывает, кончилась ли передача байта данных. Таким образом, эти флаги запускают прерывания и устанавливают соответствующие биты в регистре ожидания прерывания (смотри пункт 11.3.3).

Заметим, что в то время, как событие приема/передачи устанавливает бит RIx/TIx в SP_STATx и соответствующие биты ожидания прерывания (INT_PEND, INT_PEND1), программная запись RIx/TIx в SP_STATx не влияет на биты ожидания прерывания и не вызывает прерывания. Аналогично, чтение SP_STATx очищает биты RIx и TIx, но не очищает соответствующий бит в регистре ожидания прерывания.

SP_STAT0
регистр состояния последовательного порта 0

11_H значение после Сброса: $0B_H$

(горизонтальное окно 0 - чтение, горизонтальное окно 15 - запись)

7	6	5	4	3	2	1	0
RPE0/ RB80	RI0	TI0	FE0	TXE0	OE0	-	
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	

SP_STAT1
регистр состояния последовательного порта 1

$1FAD_H$ значение после Сброса: $0B_H$

7	6	5	4	3	2	1	0
RPE1/ RB81	RI1	TI1	FE1	TXE1	OE1	-	
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	

Рисунок 11.6 – Формат регистров SP_STATx

Бит “Передатчик пуст» (TXEx) устанавливается, если SBUF(TXx) и его буфер пусты и готовы принять два байта. TXEx очищается, как только байт записан в SBUF(TXx). Один байт может быть записан, если только TIx установлен. По определению если TXEx уже установлен, то передача окончена и TIx также установлен.

Флаг «Ошибка четности при передаче» (RPEx) или «Флаг приема бита 8» (RB8x) определяются в зависимости от разрешения или запрещения проверки четности соответственно. Когда проверка четности разрешена, то RPEx устанавливается, если обнаружена ошибка четности. Если проверка четности не разрешена, в RB8x помещается девятый бит данных, посылаемый в режимах 2 и 3.

12 Интерфейс 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 SMBus (Version 1.1 и 2.0), ACCESS.Bus, I2C (Version 2.1);
- поддержка стандартного, скоростного (F/S) и высокоскоростного (Hs) режимов;
- программирование действий Master или Slave;
- возможность подключения к шине нескольких ведущих устройств (режим Multi-Master);
- один определяемый программно 7/10-битный адрес для Slave;
- возможность одновременного обращения ко всем устройствам шины, так называемый «общий вызов» (global call).

Особые возможности SMBus:

- отслеживание времени ожидания линии SCL;
- наличие функции PEC (Packet Error Checking – отслеживание ошибок в пакетах данных) с использованием стандартного полинома SMBus;
- возможность обращения одного или нескольких Slave к Master с получением отклика от последнего;
- возможность последовательного опроса;
- функционирование под управлением прерываний.

12.1 Протокол шины

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

Протокол поддерживает режим Multi-Master – одно или несколько устройств, подключенных к шине, могут контролировать шину. Каждому устройству, подключенному к шине, присваивается собственный адрес, оно может выступать как передатчик или приемник, хотя некоторые периферийные устройства являются только приемниками.

Операции с данными

Устройство, которое начинает передачу данных, становится Master. Master генерирует тактовый сигнал SCL, а также инициирует и завершает передачу данных по шине.

Один бит данных передается по SDA в течение каждого импульса на SCL (рисунок 12.1). Данные стабильны, пока уровень сигнала на линии SCL высокий, могут меняться, пока уровень сигнала на линии SCL низкий.

Каждая передача данных состоит из начального состояния «Старт», ряда передач байтов данных и состояния «Стоп» при окончании передачи данных. Первым передается старший разряд (Most Significant Bit – MSB). После каждого байта (8 бит) должен следовать сигнал подтверждения (ACK).

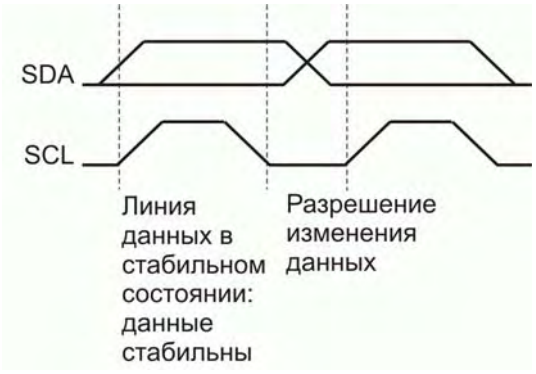


Рисунок 12.1 – Передача бита данных

Slave может увеличивать паузу между тактовыми импульсами, удерживая на линии SCL сигнал низкого уровня, пока происходит обработка принятых данных или подготовка данных для следующей передачи. Этот процесс может происходить после передачи любого бита.

«Старт» и «Стоп»

Master формирует состояния «Старт» и «Стоп» (рисунки 12.2 и 12.3). После того, как было сформировано состояние «Старт», шина считается занятой и другие ведущие не должны пытаться управлять ей. Такой статус шина сохраняет до тех пор, пока не будет сформировано состояние «Стоп».

Состояние «Старт» (S) – отрицательный перепад на SDA, когда сигнал на SCL высокого уровня.

Состояние «Стоп» (P) – положительный перепад на SDA, когда сигнал на SCL высокого уровня.

Состояние «Повторного старта» (Sr) – может быть сформировано в середине передачи, чтобы разрешить другому Slave обратиться к Master или чтобы изменить направление передачи данных без потери контроля над шиной.

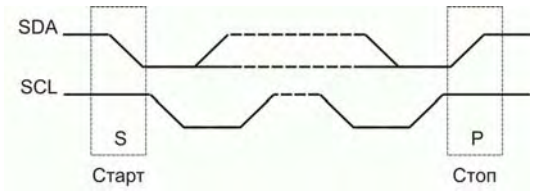


Рисунок 12.2 – Состояния «Старт» и «Стоп»

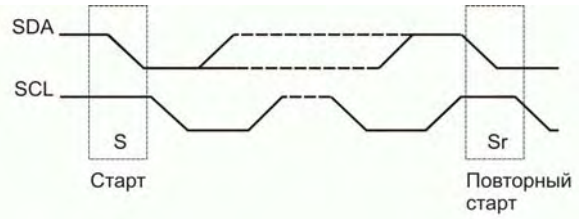


Рисунок 12.3 – Состояние «Повторный старт»

Цикл подтверждения

Цикл подтверждения (рисунки 12.4 и 12.5) включает в себя два сигнала:

1. Тактовый импульс подтверждения, который передается Master с каждым байтом данных.

2. Сигнал подтверждения (ACK), посылаемый приемным устройством.

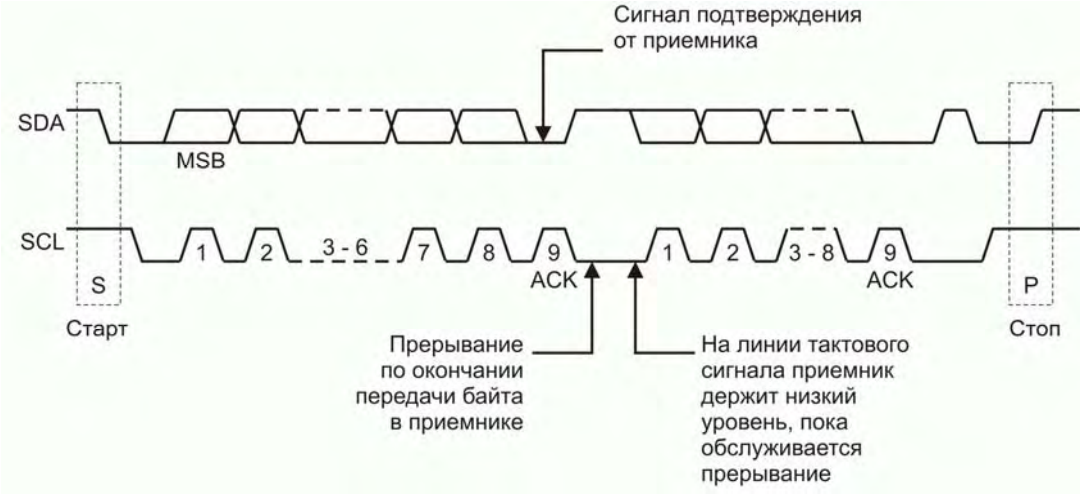


Рисунок 12.4 – Операции с данными I2C

Во время девятого тактового импульса при передаче данных Master генерирует импульс подтверждения. Передатчик освобождает линию SDA (удерживает ее в высоком уровне), чтобы позволить приемнику отправить сигнал подтверждения. В течение импульса подтверждения приемник должен сформировать сигнал подтверждения, указывая тем самым на корректный прием последнего байта данных и готовность к приему следующего байта данных.

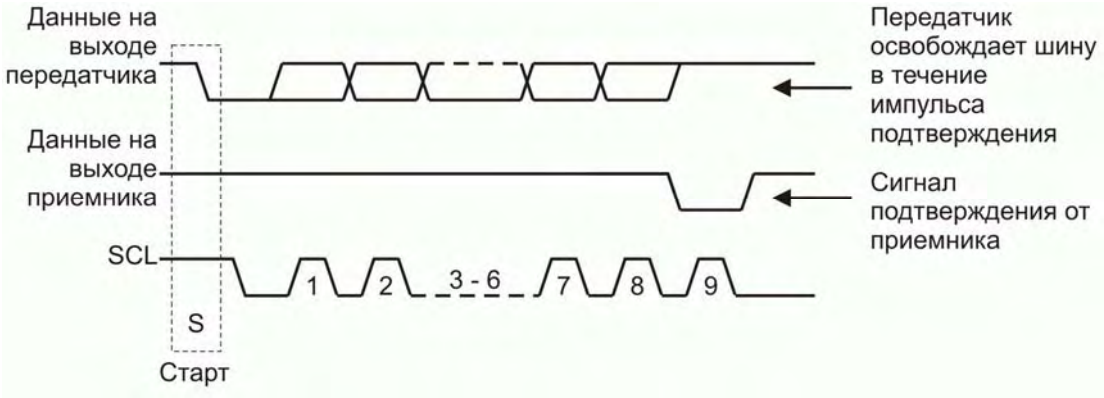


Рисунок 12.5 – Цикл подтверждения I2C

Правило «Подтверждение после каждого байта»

Master генерирует импульс подтверждения после каждого переданного байта, приемник должен формировать сигнал подтверждения после каждого принятого байта.

Существует два исключения из правила «Подтверждение после каждого байта»:

1 Когда Master является приемником, то при завершении передачи данных во время девятого тактового импульса он формирует инверсный сигнал подтверждения (NACK).

2 Если возникли проблемы при приеме или приемник переполнен, он формирует инверсный сигнал подтверждения (NACK), показывая, что не может принять дополнительные данные.

Формат передачи данных с 7-битной адресацией

На рисунке 12.6 показана завершенная передача данных. Каждому устройству, подключенному к шине, присваивается уникальный 7-битный адрес. Первые семь бит, передаваемые после состояния «Старт» – адрес Slave, восьмой бит R/W# указывает направление передачи данных (передача Slave-устройством – чтение, передача Slave-устройству – запись).

Рисунок 12.6 – Завершенная передача данных

Каждое устройство, подключенное к шине, сравнивает принятый адрес со своим собственным адресом. Если адрес устройства совпадает с принятым им адресом, то устройством формируется сигнал подтверждения. В зависимости от значения бита R/W# (1_B – чтение, 0_B – запись), адресуемое устройство становится Slave-передатчиком или Slave-приемником.

Протокол I2C предоставляет возможность одновременного обращения к устройствам, подключенным к шине при помощи адреса общего вызова. В первом передаваемом байте определяется специальный адрес общего вызова (00_H), во втором – значение общего вызова. Slave, готовые к приему данных, формируют сигнал подтверждения, остальные игнорируют общий вызов.

Протокол I2C также допускает работу в режиме Alert-отклика (Slave Transmit Alert Response Mode). Любой Slave (или несколько Slave) может послать сигнал запроса SMBAlert, с целью передать данные к Master. Master, получив сигнал SMBAlert, в ответ выдает на шину специальный адрес отклика 0001100_B с битом R/W#, равным 1_B (чтение).

Устройство (устройства), пославшее сигнал SMBAlert, получив адрес отклика, выдает на шину свой 7-битный адрес (последний бит передаваемого байта может быть равным как 0_B, так и 1_B). Если несколько Slave выдают свои адреса на шину, то передачу осуществляет устройство, выигравшее арбитраж.

Как только в ответ на появление адреса отклика одним из устройств будет выдан на шину корректный адрес, сигнал SMBAlert должен быть сброшен программно (поддержка режима Alert-отклика является уникальной особенностью устройств SMBus).

Арбитраж

Процесс распределения приоритетов (арбитраж) на линии SDA происходит в то время, когда сигнал на линии SCL высокого уровня. Арбитраж возникает в случае, когда два Master одновременно сформировали состояние «Старт», и продолжается до тех пор, пока одно устройство удерживает высокий уровень на SDA, в то время как другое удерживает низкий уровень на SDA (уровень сигнала на SCL – высокий). Master, удерживающий высокий уровень на SDA, теряет приоритет. Если Master теряет приоритет во время передачи адреса Slave (первый байт, следующий после формирования состояния «Старт»), то он переключается в режим «Slave приемник» и начинает сравнивать передаваемые адреса со своим адресом. Приоритет также может быть потерян в режиме «Master приемник» в течение цикла подтверждения или в режиме «Slave передатчик» во время получения сигнала отклика.

В случае потери приоритета устанавливаются соответствующие биты в статусном регистре SMBST (SMBST.MODE) и генерируется прерывание.

Синхронизация тактового сигнала

В случае, когда два и более Master пытаются одновременно начать передачу, необходима синхронизация их тактовых сигналов. Задача синхронизации решается просто благодаря тому, что все устройства подключаются к линии SCL по схеме «монтажное И». В результате длительность тактовых импульсов на линии SCL определяется тактовым сигналом с наименьшей длительностью импульсов, а пауза между тактовыми импульсами – тактовым сигналом с наибольшей паузой между импульсами.

Формат передачи данных с 10-битной адресацией

10-битная адресация (рисунок 12.7) позволяет использовать на шине I2C 1024 адреса Slave. После формирования состояния «Старт» передается зарезервированный адрес $11110XX_B$. 10-битная адресация полностью совместима с 7-битной адресацией. Таким образом, Slave с величиной адреса 7 бит и 10 бит могут быть подключены к одной и той же шине.

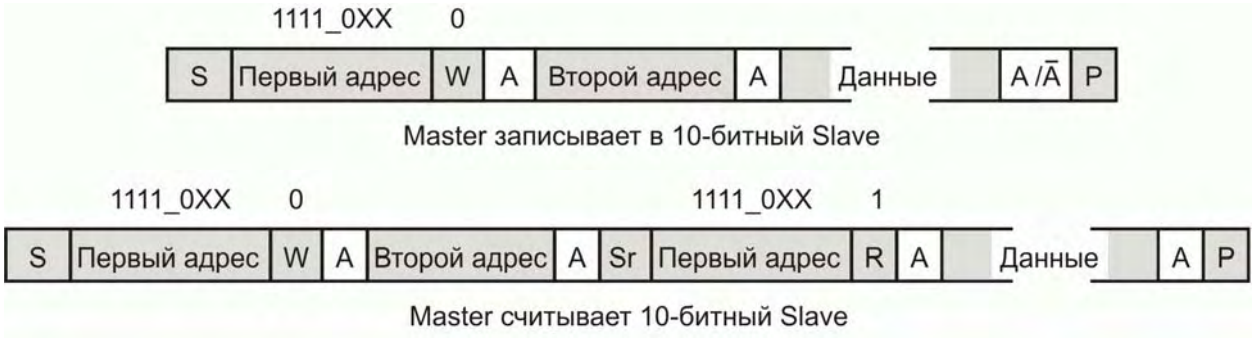


Рисунок 12.7 – 10-битная адресация Slave

Высокоскоростной режим (Hs)

По сравнению с режимом «Скоростной / Стандартный» (F/S) в высокоскоростном режиме повышается скорость передачи данных по шине. В режиме «Скоростной /Стандартный» скорость передачи ограничена 400 кГц и 100 кГц, соответственно, а в высокоскоростном режиме – 3,4 МГц, таким образом, пропускная способность увеличивается почти в 10 раз.

Шина переходит в высокоскоростной режим из режима «Быстрый / Стандартный» с помощью следующих действий:

Формирование состояния «Старт» (S).

8-битный адрес Master ($00001YYY_B$), где YYY – уникальный адрес каждого мастера, подключенного к шине.

Инверсный сигнал подтверждения (A#), передаваемый для Slave.

Так как адрес для каждого Master является уникальным, то процесс распределения приоритетов происходит только в момент передачи адреса Master. После передачи адреса один Master получает приоритет в работе с шиной. Таким образом, арбитраж и синхронизация тактового сигнала не выполняются в высокоскоростном режиме.

После выполнения вышеуказанных действий устройства, поддерживающие высокоскоростной режим, подключаются к шине. Далее Master формирует состояние «Повторного старта» (Sr), затем передается адрес Slave и бит направления передачи (R/W#).

Формат операций с данными в высокоскоростном режиме Hs аналогичен режиму F/S, таким образом, режим Hs полностью совместим с устройствами, работающими в режиме F/S.

Завершение режима Hs (рисунок 12.8) производится формированием состояния «Стоп» (P), после чего все устройства переключаются в режим F/S.

Рисунок 12.8 – Высокоскоростной режим

12.2 Функциональное описание

На рисунке 12.9 показано строение модуля I2C.

Входные и выходные каскады линий SDA и SCL

Выводы SDA и SCL используются для подключения модуля к одноименным линиям шины I2C. Входные каскады, подключенные к выводам SDA и SCL, содержат фильтры, подавляющие помехи. В режиме F/S эти фильтры подавляют все входные сигналы с длительностью меньше, чем период тактового сигнала. Выходные каскады содержат устройства с предустановкой, выполненные по схеме с открытым стоком. Работа входных и выходных каскадов разрешается при разрешении работы модуля I2C (установлен бит SMBCTL2.ENABLE).

Регистр адреса и компаратор

В регистр адреса SMBADDR может быть записан 7-битный адрес, на который I2C отвечает, когда является Slave-устройством на шине. Установка старшего бита регистра SMBADDR (SAEN) разрешает распознавание адреса Slave.

Компаратор адреса сравнивает принятый 7-битный адрес с адресом, записанным в регистр SMBADDR. При разрешении общего вызова (установлен бит SMBCTL1.GCMEN) первый принятый байт сравнивается с величиной 00_{H}.

В режиме Alert-отклика – если установлен бит SMBCTL1.SMBARE – первый принятый байт сравнивается с величиной 0001100_{B}.

При 10-битной адресации (одновременно установлены биты SMBADDR.SAEN и SMBCTL3.S10EN) старшие 5 бит первого принятого адреса сравниваются с величиной 11110B, младшие 2 бита принятого адреса сравниваются с величиной, записанной в SMBCTL3S10.2, SMBCTL3S10.1. Старший бит второго принятого адреса сравнивается с SMBCTL3.SA10.0, младшие 7 бит – с SMBADDR.ADDR.

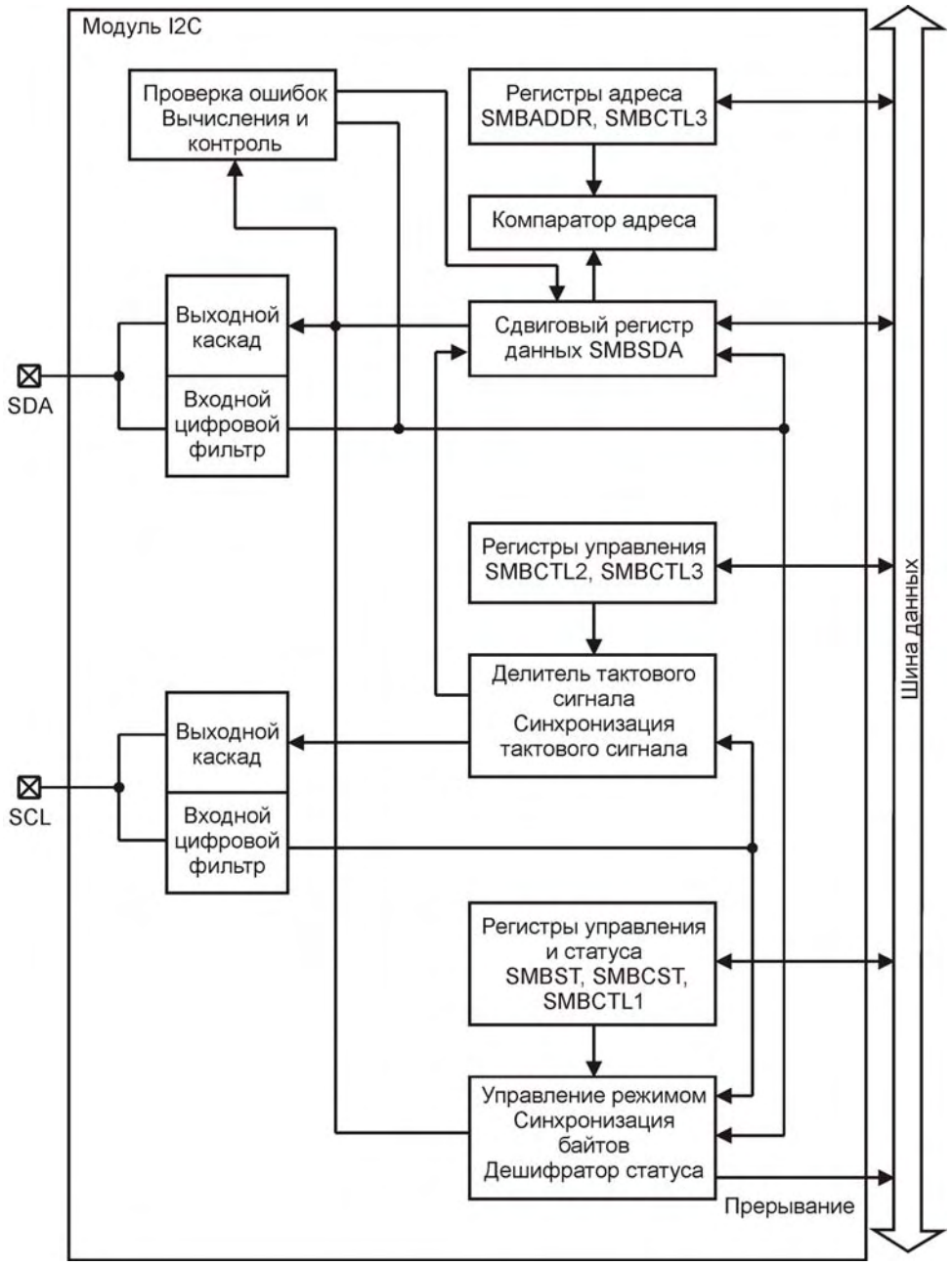


Рисунок 12.9 – Строение модуля I2C

Сдвиговый регистр данных

Сдвиговый регистр данных (SMBSDA) содержит байт данных, который может быть как принятым байтом данных, так и байтом данных, который необходимо передать. Данные одновременно принимаются и передаются, следовательно SMBSDA всегда содержит последний байт данных, которые передавались по шине. Поэтому даже если произошла потеря приоритета, то корректные данные будут сохраняться в SMBSDA до момента окончания операции с данными. Данные передаются по заднему фронту SCL (первым передается MSB), принимаются – по переднему фронту SCL.

Процесс распределения приоритетов проходит во время передачи адреса, формирования состояний «Старт» и «Стоп». Если принятые данные отличаются от переданных (например, передана «1», а принят «0»), то приоритет будет потерян.

Генерация тактового сигнала. Синхронизация

Последовательный тактовый сигнал (выходной сигнал I2C в режиме Master) формируется генератором из системного тактового сигнала.

В режиме F/S используется 7-битный предделитель. Значение старших 6 бит определяется полем SMBCTL2.SCLFRQ, младший бит всегда равен нулю (деление производится только на четное число). Минимальный коэффициент деления равен восьми, максимальный – 128.

Блок синхронизации тактового сигнала производит синхронизацию генератора тактового сигнала и выходного сигнала SCL с тактовым сигналом других устройств, подключенных к шине.

В режиме Hs коэффициент деления определяется величиной, записанной в поле SMBCTL3.HSCDIV. Предделитель уменьшен до 4 бит, значение младшего бита равно нулю.

Управление режимом и статус

Флаг прерывания находится в регистре SMBST. Также этот регистр содержит биты, показывающие режим работы I2C (Master или Slave, приемник или передатчик).

Регистр SMBCST является регистром управления и статуса, он показывает статус шины, возврат после ошибки.

Регистр SMBCTL1 управляет формированием состояний «Старт», «Повторный старт», «Стоп», а также сигнала подтверждения приема данных.

В регистрах SMBCTL2 и SMBCLT3 содержатся коэффициенты деления тактового сигнала, которые используются при работе I2C в режиме Master. Также регистр SMBCLT3 управляет 10-битной адресацией.

Функционирование

I2C поддерживает следующие режимы работы:

- «Неадресованный Slave»;
- «Master передатчик»;
- «Master приемник»;
- «Slave приемник»;
- «Slave передатчик».

Инициализация

Для начала работы шины I2C требуется произвести инициализацию. Для этого необходимо выполнение нижеследующих действий.

Разрешить работу I2C установкой бита SMBCTL2.ENABLE.

При работе в режиме Master для выбора периода тактового сигнала SCL необходимо записать нужный коэффициент деления в поле SMBCTL2.SCLFRQ.

В режиме Hs необходимо записать коэффициент деления в поле SMBCTL3.HSDIV.

Для работы в режиме Slave необходимо обнулить бит SMBCTL1.MOD.

При работе в режиме Slave значения в полях SMBCTL2.SCLFRQ и SMBCTL3.HSDIV не учитываются.

Для работы в режиме Slave необходимо записать в регистр SMBADDR значение адреса Slave и разрешить распознавание адреса установкой бита SMBADDR.SAEN.

Для разрешения расширенной 10-битной адресации записать значение старших разрядов адреса в поле SMBCTL3.S10AD и установить бит SMBCTL3.S10EN.

Установить бит SMBCTL1.GCMEN для разрешения распознавания адреса общего вызова (дополнительная функция).

Установить бит SMBCTL1.SMBARE для распознавания адреса отклика (дополнительная функция).

Если установлены требования к времени ожидания шины – записать соответствующую величину в биты SMBCST.TOCDIV и в регистр SMBTOPR для проверки времени активной работы линии SCL (максимум 25 мс). Запись в биты

SMBCST.TOCDIV величины, отличной от нуля, автоматически разрешает проверку времени ожидания шины.

Для разрешения прерывания I2C установить бит SMBCTL1.INTEN.

Режим «Неадресованный Slave»

Режим «Неадресованный Slave» (общий режим) является режимом работы по умолчанию (SMBST.MODE = IDLE (00_H)). После разрешения работы I2C шина начинает работать в режиме «Неадресованный Slave». I2C непрерывно отслеживает состояние шины и при формировании состояния «Старт» или «Повторный старт» переходит в режим «Slave приемник». Из этого режима есть возможность перейти в режим «Master передатчик» после успешного формирования состояния «Старт».

I2C возвращается в режим «Неадресованный Slave» при выполнении следующих условий:

- состояние «Старт» не было успешно сформировано, так как другое устройство удерживает линию SCL в низком уровне;
- произошла потеря приоритета при передаче пакета данных в режиме «Master передатчик» или при передаче бита R/W# в режиме «Master приемник»;
- произошла потеря приоритета во время отклика;
- отправлен инверсный сигнал подтверждения в режиме «Slave приемник» (было запрещено распознавание адреса или адрес Slave не совпал);
- получен инверсный сигнал подтверждения в режиме «Slave передатчик»;
- сформировано состояние «Стоп»;
- ошибочное состояние на шине;
- сброс I2C;
- запрещение работы I2C.

Режим «Master передатчик»

Переход в режим «Master передатчик» производится после формирования сигнала «Старт». Первый байт, следующий после формирования сигнала «Старт», передается Master-устройством и содержит адрес Slave и бит направления передачи.

В зависимости от значения бита направления передачи R/W# I2C может как остаться в режиме «Master передатчик», так и перейти в режим «Master приемник», если R/W# = 1_B. Вместо последующей передачи адреса Slave, I2C может передать адрес Master ($00001XXX_B$) для перехода в высокоскоростной режим Hs.

I2C может перейти в режим Master с помощью установки бита SMBCTL1.START. I2C проверяет, свободна ли шина (бит SMBCST.BB = 0_B), и формирует состояние «Старт». Если шина занята (SMBCST.BB = 1_B), то состояние «Старт» будет сформировано в тот момент, когда шина будет свободна и сигнал на линии SCL будет в высоком уровне.

При успешном формировании состояния «Старт», бит SMBCTL1.START станет равным нулю, будет установлен флаг SMBST.INT и линия SCL будет удерживаться в низком уровне, пока флаг не будет сброшен (SMBST.MODE = 01_H – код статуса шины STDONE). Прерывание будет сгенерировано в случае, если установлен бит SMBCTL1.INTEN.

Передача адреса и данных

Если бит SMBST.INT установлен, необходимо программно записать адрес Slave и направление передачи в сдвиговый регистр данных (SMBSDA).

После записи в регистр SMBSDA необходимо сбросить флаг прерывания SMBST.INT – очистить бит SMBCTL1.CLRST.

Линия SCL будет освобождена после сброса флага SMBST.INT и времени установки данных. Затем содержимое регистра SMBSDA будет передано на шину.

После передачи всех битов данных и получения сигнала подтверждения (после девятого тактового импульса), бит подтверждения анализируется системой, биты SMBST.MODE показывают код статуса шины I2C в зависимости от принятого адреса.

В течение передачи данных на линиях SDA и SCL отслеживаются потенциальные конфликты с другими устройствами, подключенными к шине. Если зафиксирован конфликт, передача данных прекращается. Код статуса шины определяется как SRAAPA (SMBST.MODE = 11_H), если потерян приоритет в режиме «Slave приемник», или как IDLARL (SMBST.MODE = 03_H), если потерян приоритет в режиме «Неадресованный Slave».

Если бит направления передачи R/W# = 1_B, I2C переходит в режим «Master приемник» (если не произошло ошибок).

Если выбрано требуемое направление передачи и передача адреса Slave успешно завершена (код статуса шины не MTADNA (SMBST.MODE = 05_H) и не BERROR (SMBST.MODE = $1F_H$)), то устанавливается флаг SMBST.INT, показывающий необходимость записи первого байта данных в регистр SMBSDA. Пока установлен флаг SMBST.INT, SCL удерживается в низком уровне, прерывание будет сгенерировано, если установлен бит SMBCTL1.INTEN.

В случае, когда после передачи данных Slave приемник отправляет инверсный сигнал подтверждения, на SCL будет удерживаться низкий уровень, прерывание будет сгенерировано, если установлен бит SMBCTL1.INTEN (код статуса шины MRDANA (SMBST.MODE = $0B_H$)).

Отслеживание ошибок в пакетах данных

При работе I2C в режиме «Master передатчик» после установки бита SMBCST.PECNEXT произойдет передача содержимого регистра Packet Error Checking в регистр SMBSDA и начало передачи байта Cyclic Redundancy Check (CRC) для Slave-устройства. Это произойдет после передачи последнего байта данных и до того, как будут сформированы состояния «Старт» и «Повторный старт» (таблица 12.1, рисунок 12.10).

Таблица 12.1 – Режим F/S «Master передатчик», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Следующее действие
			R/W	SMBSDA	clrst	ask	stop	start	
1	2	3	4	5	6	7	8	9	10
01_H	STDONE	Сформировано состояние «Старт»	W	Адрес Master	1	0	0	0	Передача кода Master, переход в режим Hs
				SADR/RW = 0_B					Передача SADR/RW – ACK или NACK от Slave
02_H	RSDONE	Сформировано состояние «Повторный Старт»	W	SADR/RW = 0_B	1	0	0	0	Передача SADR/RW – ACK или NACK от Slave
				SADR/RW = 1_B					Передача SADR/RW – ACK или NACK от Slave, I2C переходит в режим «Master приемник»
03_H	IDLARL	Потеря приоритета, переход в режим «Неадресованный Slave»	–		1	0	0	0	Режим «Неадресованный Slave»

Окончание таблицы 12.1

1	2	3	4	5	6	7	8	9	10
04_H	MTADPA	Отправлен адрес Slave, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
05_H	MTADNA	Отправлен адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	Состояние «Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
06_H	MTDAPA	Отправлен байт данных, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Старт»
07_H	MTDANA	Отправлен байт данных, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»

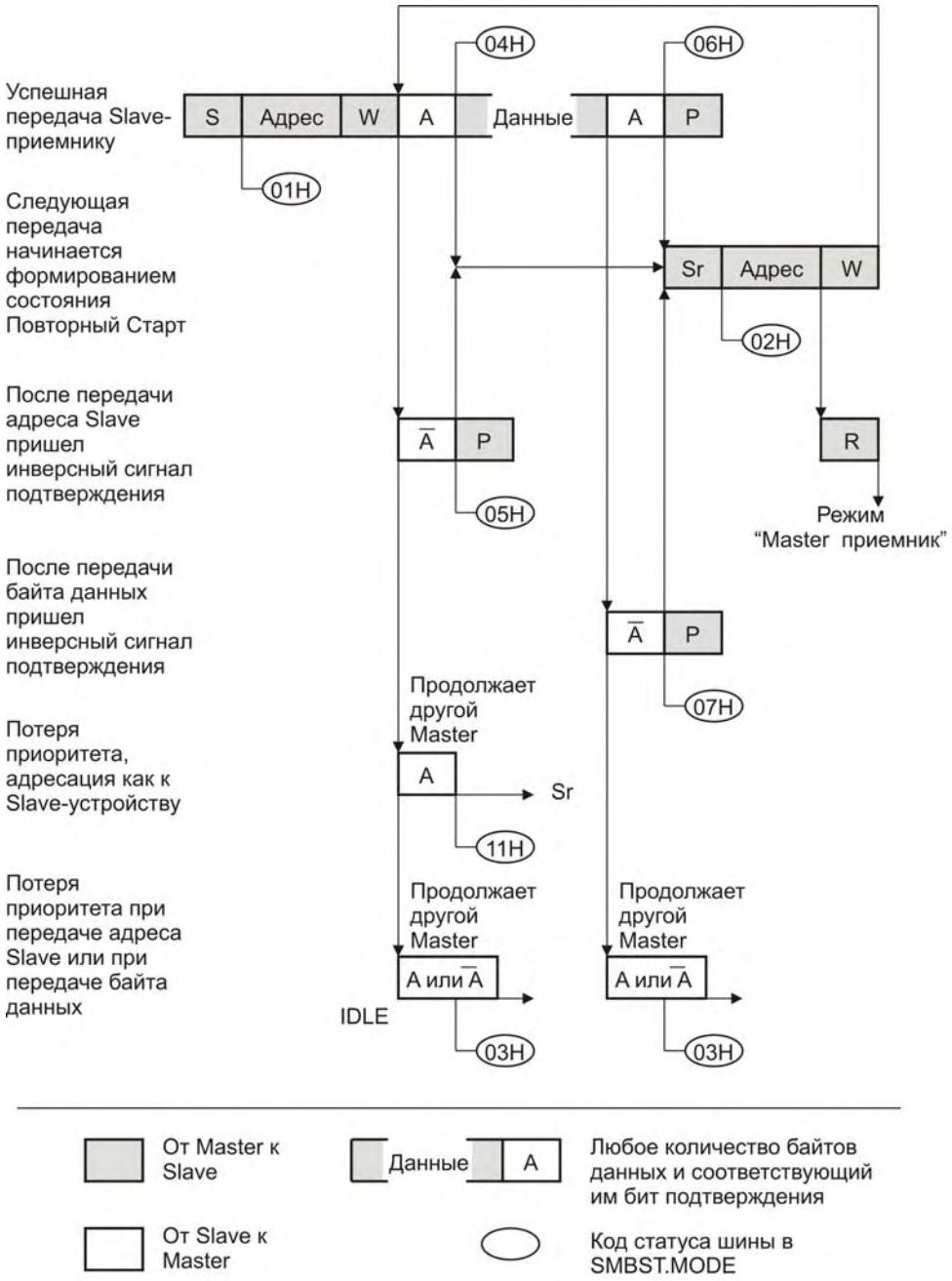


Рисунок 12.10 – Режим F/S «Master передатчик», действия и статус

Повторный старт

I2C при работе в режиме «Master передатчик» осуществляет полный контроль над шиной и может адресовать другого Slave, изменить направление передачи без потери контроля над шиной.

Для формирования состояния «Повторный старт» необходимо:

- установить бит SMBCTL1.START;
- в режиме «Master приемник» считать последний элемент данных из регистра SMBSDA;
- сбросить ожидаемый флаг прерывания установкой бита SMBCTL1.CLRST;
- впоследствии линия SCL будет освобождена, будет сгенерировано состояние «Повторный старт» и прерывание I2C (код статуса шины RSDONE).

Выход из режима «Master передатчик»

Выход из режима «Master передатчик» осуществляется формированием сигнала «Стоп». Чтобы сформировать сигнал «Стоп» необходимо:

- установить бит SMBCTL1.STOP;
- в режиме «Master приемник» считать последний элемент данных из регистра SMBSDA;
- сбросить ожидаемый флаг прерывания установкой бита SMBCTL1.CLRST.

Выполнение данных условий вызывает немедленное формирование I2C состояния «Стоп» и обнуление бита SMBCTL1.STOP. Состояние «Стоп» может быть сформировано только в том случае, когда I2C является активным Master-устройством шины (SMBST.MODE = 01_H – $0B_H$).

Высокоскоростной режим (Hs) «Master передатчик»

Вход в высокоскоростной режим «Master передатчик» осуществляется формированием состояния «Старт» с последующей передачей адреса Master ($00001XXX_B$) вместо адреса Slave. После того, как адрес Master будет предан, установится флаг SMBST.INT, сгенерируется прерывание (если был установлен бит SMBCTL1.INTEN). После успешной передачи адреса Master код статуса шины станет НМТОК. В этот момент I2C входит в высокоскоростной режим «Master передатчик» (рисунок 12.11, таблица 12.2).

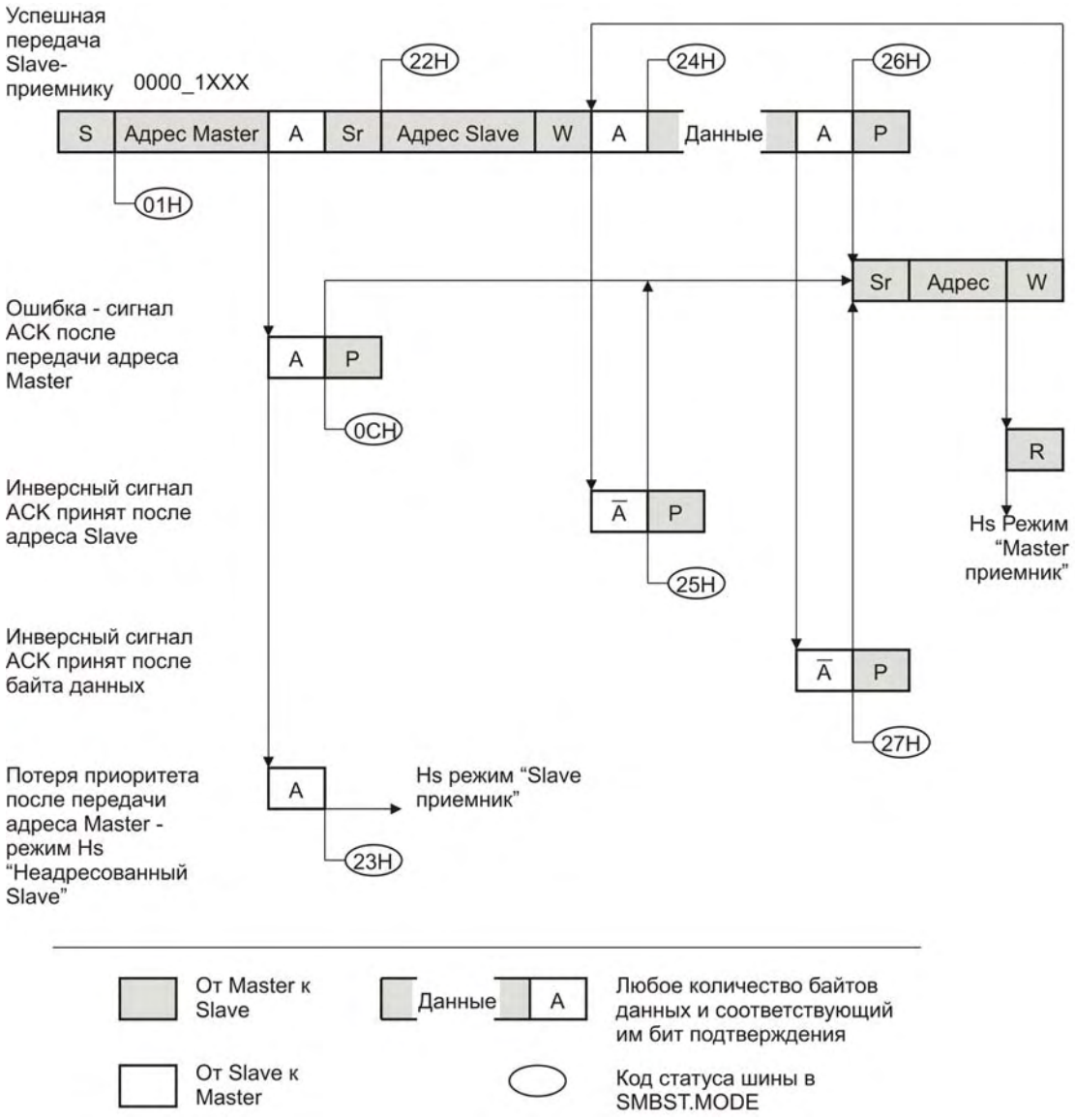


Рисунок 12.11 – Режим Hs «Master передатчик», действия и статус

Таблица 12.2 – Режим Hs «Master передатчик», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
$0C_H$	MTMCER	Передан код Master, найдена ошибка (сигнал ACK)	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
21_H	HMTMCOK	Адрес Master успешно передан, переход в режим Hs	–		1	0	0	1	«Повторный старт»
22_H	HRSDONE	Сформирован «Повторный старт»	W	SADR/RW = 0	1	0	0	0	Передача SADR/RW – сигнал ACK или NACK от Slave
				SADR/RW = 0					Передача SADR/RW – сигнал ACK или NACK от Slave, I2C перейдет в режим Hs «Master приемник»
23_H	HIDLARL	Потеря приоритета, переход в режим Hs «Неадресованный Slave»	–		1	0	0	0	Режим «Неадресованный Slave»
24_H	HMTADPA	Отправлен адрес Slave, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
25_H	HMTADNA	Отправлен адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
26_H	HMTDAPA	Отправлен байт данных, сигнал ACK	W	Данные	1	0	0	0	Передача байта данных
			–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»
27_H	HMTDANA	Отправлен байт данных, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» и последующий «Повторный старт»

Необходимо сформировать состояние «Повторный старт» установкой бита SMBCTL1.START и сбросить флаг прерывания установкой бита SMBCTL1.CLRST.

После формирования сигнала «Повторный старт» установится флаг SMBST.INT, код статуса в SMBST.MODE станет HRSDONE. Затем последует передача адреса и данных (см. «Передача адреса и данных»).

Режим «Master приемник»

Вход в режим «Master приемник» осуществляется после успешной передачи адреса Slave с битом направления передачи, равным единице (R/W# = 1_B). В режиме «Master приемник» I2C считывает с Slave-устройства, подключенного к шине, и, тем не менее, продолжает контролировать линию SDA. I2C продолжает генерировать импульсы SCL и подтверждать каждый принятый байт данных (рисунок 12.12, таблица 12.3).

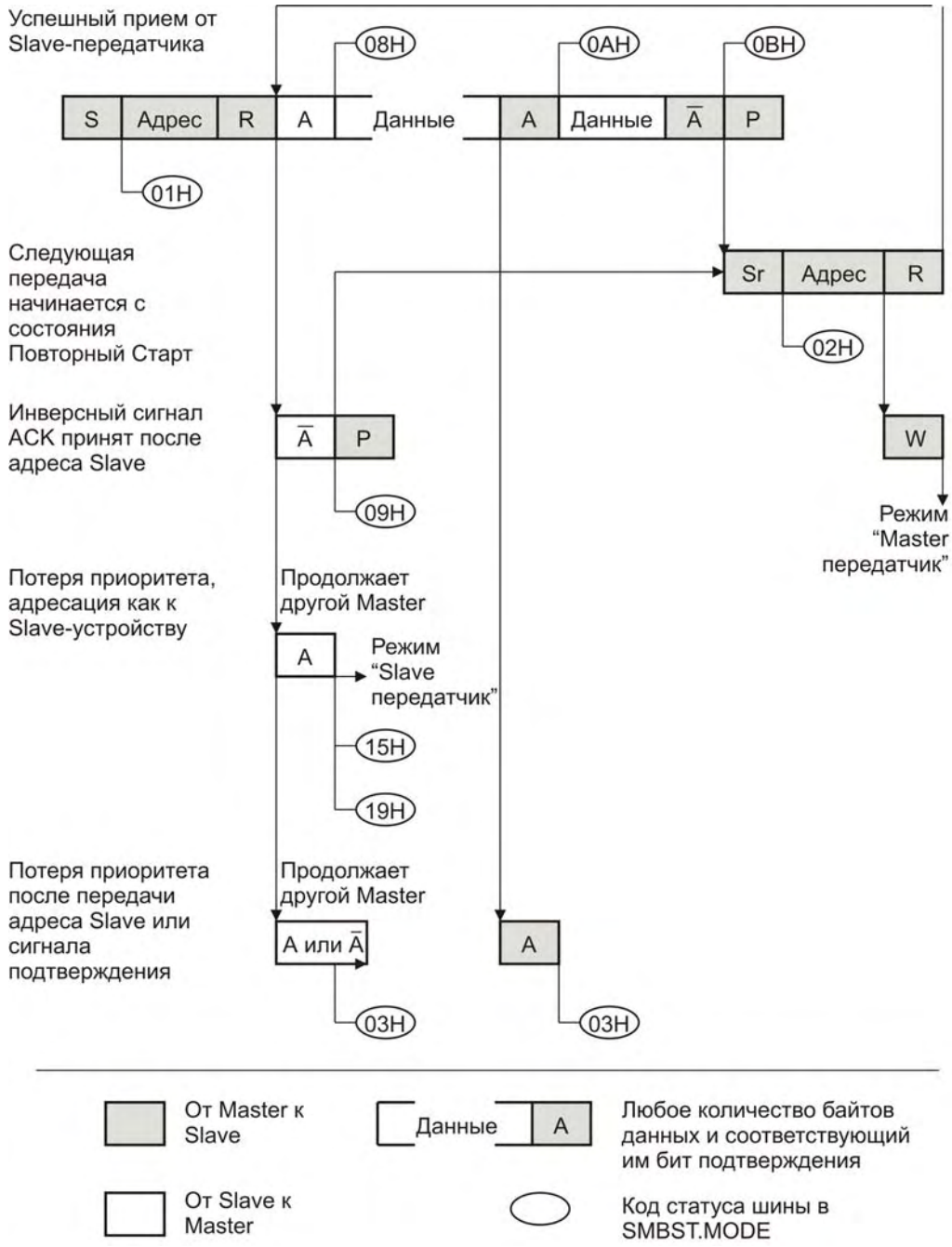


Рисунок 12.12 – Режим F/S «Master приемник»

Таблица 12.3 – Режим F/S «Master приемник»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
08_H	MRADPA	Передан адрес Slave, сигнал ACK	–		1	0	0	0	Принят байт данных, сформирован сигнал ACK
					1	1	0	0	Принят байт данных, сформирован сигнал NACK
09_H	MRADNA	Передан адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
$0A_H$	MRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$0B_H$	MRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»

После каждого принятого байта устанавливается бит SMBST.INT и данные программно считываются из регистра SMBSDA. Линия SCL удерживается в низком уровне, пока установлен флаг SMBST.INT. Установка бита SMBCTL1.CLRST очищает бит SMBST.INT, разрешая прием следующего байта данных.

Протокол шины устанавливает, что состояния «Повторный старт» или «Стоп» не должны формироваться, пока работа ведется в режиме «Master приемник» и I2C уже не является единственным устройством, контролирующем SDA. Для восстановления полного контроля над шиной в соответствии с протоколом шины Master-приемнику необходимо отправить инверсный сигнал подтверждения после приема последнего байта данных.

Для этого необходимо очистить бит SMBCTL1.ACK до сброса флага SMBST.INT во время приема предпоследнего байта. В то же самое время должен быть установлен бит SMBCST.PECNEXT, если есть запрос на PEC. Когда флаг SMBST.INT будет сброшен, линия SCL станет свободной и будет принят последний байт данных, в течение девятого тактового импульса SCL будет отправлен инверсный сигнал ACK. Затем I2C вернется в режим «Master передатчик» и сможет сформировать на шине состояния «Повторный старт» или «Стоп».

Проверка ошибок пакета данных

При разрешении проверки ошибок пакета данных (PEC), последний байт, принятый от передающего Slave-устройства, будет байт PEC. Если результат вычислений CRC отличен от нуля, то будет установлен бит SMBCST.PECFAULT, указывающий на ошибку.

Высокоскоростной режим «Master приемник»

Вход в высокоскоростной режим «Master приемник» осуществляется, если после передачи адреса Master следует адрес Slave с битом направления передачи, равным единице (R/W# = 1_B), а затем формируется состояние «Повторный старт». После входа I2C

в высокоскоростной режим «Master приемник» устанавливается флаг SMBST.INT, коды статуса шины показаны в таблице 12.4 на рисунке 12.13.

Таблица 12.4 – Высокоскоростной режим «Master приемник», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
28_H	HMRADPA	Передан адрес Slave, сигнал ACK	–		1	0	0	0	Принят байт данных, сформирован сигнал ACK
					1	1	0	0	Принят байт данных, сформирован сигнал NACK
29_H	HMRADNA	Передан адрес Slave, сигнал NACK	–		1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»
$2A_H$	HMRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$2B_H$	HMRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	1	«Повторный старт»
					1	0	1	0	«Стоп»
					1	0	1	1	«Стоп» с последующим «Повторным стартом»

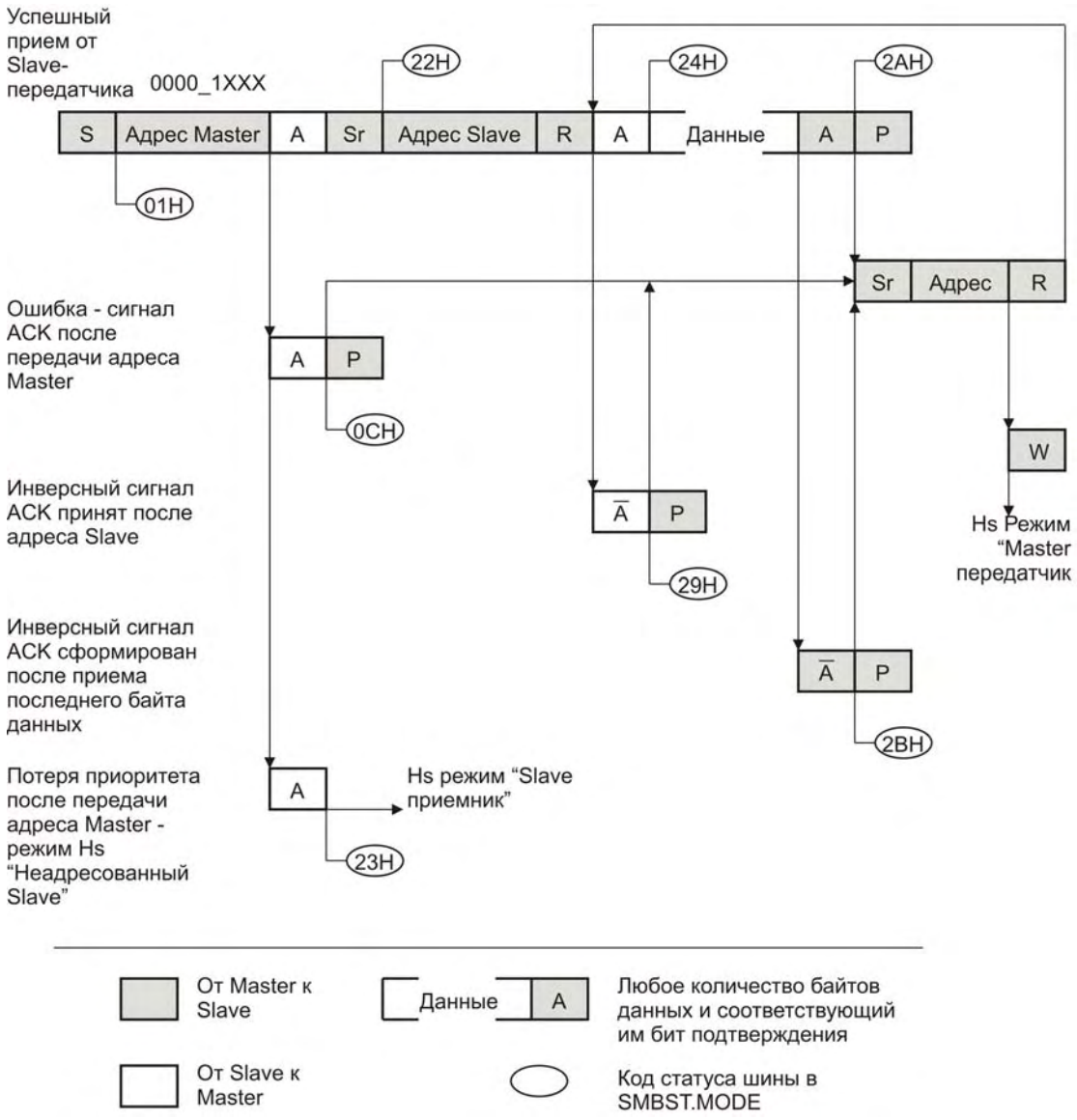


Рисунок 12.13 – Режим Hs «Master приемник»

Режим «Slave приемник»

В режиме «Slave приемник» данные принимаются от Master-передатчика. Сигнал подтверждения формируется после приема каждого байта.

Вход в режим «Slave приемник»

После разрешения на работу I2C продолжает контролировать шину. После обнаружения состояния «Старт», I2C входит в режим «Slave приемник» и начинает прием 7-битного адреса и бита направления передачи (R/W#) от Master. Переход в режим «Slave приемник» также может произойти из режима «Master передатчик» в случае потери приоритета при передаче адреса или данных.

После приема полного адреса I2C сравнивает принятый адрес с:

- полем SMBADDR.ADDR, если установлен бит SMBADDR.SAEN;
- адресом общего вызова (00_H), если установлен бит SMBCTL1.GCMEN;
- адресом отклика (0001100_B), если установлен бит SMBCTL1.SMBARE.

Сигнал подтверждения формируется в течение девятого тактового импульса на SCL, если принятый адрес совпал с адресом, записанным в регистр, адресом общего вызова или адресом отклика. После того, как произошло совпадение, устанавливается флаг SMBST.INT, изменяется значение SMBST.MODE. Если установлен бит SMBCTL1.INTEN,

то генерируется прерывание. В регистре SMBSDA содержится принятый байт, включающий адрес Slave и бит направления передачи.

В зависимости от принятого бита направления передачи I2C может перейти в режим «Slave передатчик» (R/W# = 1_B) или остаться в режиме «Slave приемник» (R/W# = 0_B).

После каждого принятого байта данных устанавливается флаг SMBST.INT, показывая необходимость считать данные из регистра SMBSDA, линия SCL удерживается в низком уровне. Необходимо программно считать принятый байт данных, а затем установить бит SMBCTL1.CLRST, чтобы сбросить флаг SMBST.INT и освободить линию SCL.

Режим «Slave приемник» – 10-битная адресация

Для использования внешней (10-битной) адресации необходимо установить бит SMBCTL2.S10EN и бит SMBADDR.SAEN. Старшие биты адреса необходимо записать в поле SMBCTL3.S10AD. После формирования сигнала «Старт» и приема первого байта, I2C сравнивает содержимое сдвигового регистра данных с величиной $11110YY_B$, где YY – являются двумя старшими битами адреса Slave, хранимыми в двух старших битах поля SMBCTL3.S10AD. Бит направления передачи должен быть равен нулю (R/W# = 0_B), показывая Slave-устройству то, что будет принята вторая часть 10-битного адреса.

Если две величины равны, то I2C формирует сигнал подтверждения и продолжает принимать младшие биты адреса. После приема первого байта не генерируется прерывание.

Master-передатчик затем передает второй байт 10-битного адреса Slave. Старший бит принятого адреса сравнивается с младшим битом поля SMBCTL3.S10AD, а младшие 7 бит принятого адреса сравниваются с величиной, определяемой полем SMBADDR.SADDR. Если величины равны, то формируется сигнал подтверждения, при этом установится флаг SMBST.INT, код статуса шины станет соответственно SRADPA или SRAPAA.

Проверка ошибок пакета данных

При разрешенной проверке ошибок пакета данных (PEC) последний байт, принятый от Master, будет байтом PEC. Если результат вычислений CRC не равен нулю, будет установлен бит SMBCST.PECFAULT, указывая на произошедшую ошибку, будет передан инверсный сигнал подтверждения. Необходимо точно знать количество байтов, передаваемых Master-устройством шины, и при чтении предпоследнего байта из регистра SMBSDA необходимо установить бит SMBCST.PECNEXT, а затем установить SMBCTL1.CLRST.

Выход из режима «Slave приемник»

Если Slave приемник больше не может принимать данные от Master, необходимо установить бит SMBCTL1.ACK до того, как будет установлен бит SMBCTL1.CLRST, сбрасывающий флаг SMBST.INT. Это послужит причиной тому, что после принятия последнего байта данных от Master, будет сформирован инверсный сигнал подтверждения. После принятия последнего байта данных установится флаг SMBST.INT. Затем необходимо программно считать последний принятый байт из регистра SMBSDA и записать единицу в бит SMBCTL1.CLRST, чтобы сбросить флаг прерывания SMBST.INT и освободить шину SCL (таблица 12.5, рисунок 12.14).

Таблица 12.5 – Режим F/S «Slave приемник», действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
10_H	SRADPA	Принят адрес Slave, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
11_H	SRAAPA	Принят адрес Slave после потери приоритета, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
12_H	SRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
13_H	SRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	0	Режим «Неадресованный Slave»
					1	0	0	1	Режим «Неадресованый Slave», формирование сигнала «Старт» после того, как шина станет свободна
$1C_H$	SSTOP	Режим Slave, обнаружено состояние «Стоп»	–		1	0	0	0	Режим «Неадресованный Slave»
					1	0	0	1	Режим «Неадресованый Slave», формирование сигнала «Старт» после того, как шина станет свободна
$1D_H$	SGADPA	Принят адрес общего вызова, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
$1E_H$	SGAAPA	Принят адрес общего вызова после потери приоритета, сигнал ACK			1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK

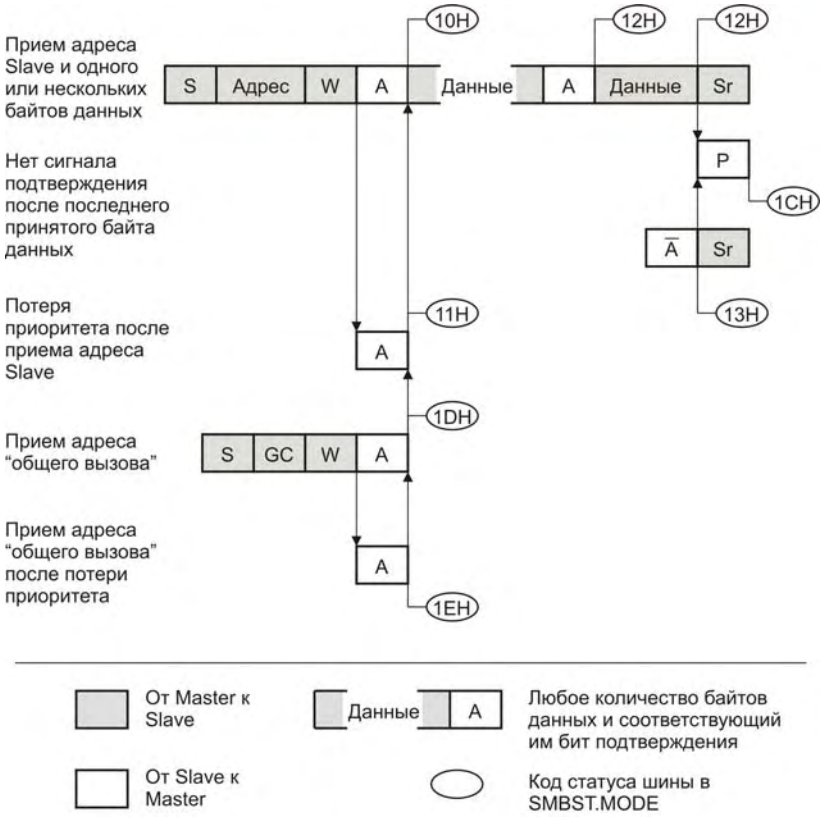


Рисунок 12.14 – Режим F/S «Slave приемник», действия и статус

Высокоскоростной режим «Slave приемник»

Вход в режим Hs «Slave приемник» осуществляется после приема адреса Master ($00001XXX_B$). После адреса Master следует состояние «Повторный старт» и адрес Slave, включающий бит направления, равный нулю (R/W# = 0_B). После приема адреса Slave, I2C производит сравнение адресов (рисунок 12.15). Более подробные действия указаны в таблице 12.6.

Таблица 12.6 – Режим Hs «Slave приемник»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
30_H	HSRADPA	Принят адрес Slave, сигнал ACK	–		1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
32_H	HSRDAPA	Принят байт данных, сигнал ACK	R	Данные	1	0	0	0	Прием байта данных, формирование сигнала ACK
					1	1	0	0	Прием байта данных, формирование сигнала NACK
33_H	HSRDANA	Принят байт данных, сигнал NACK	R	Данные	1	0	0	0	«Неадресованный Slave»
					1	0	0	1	«Неадресованный Slave», «Старт» после освобождения шины

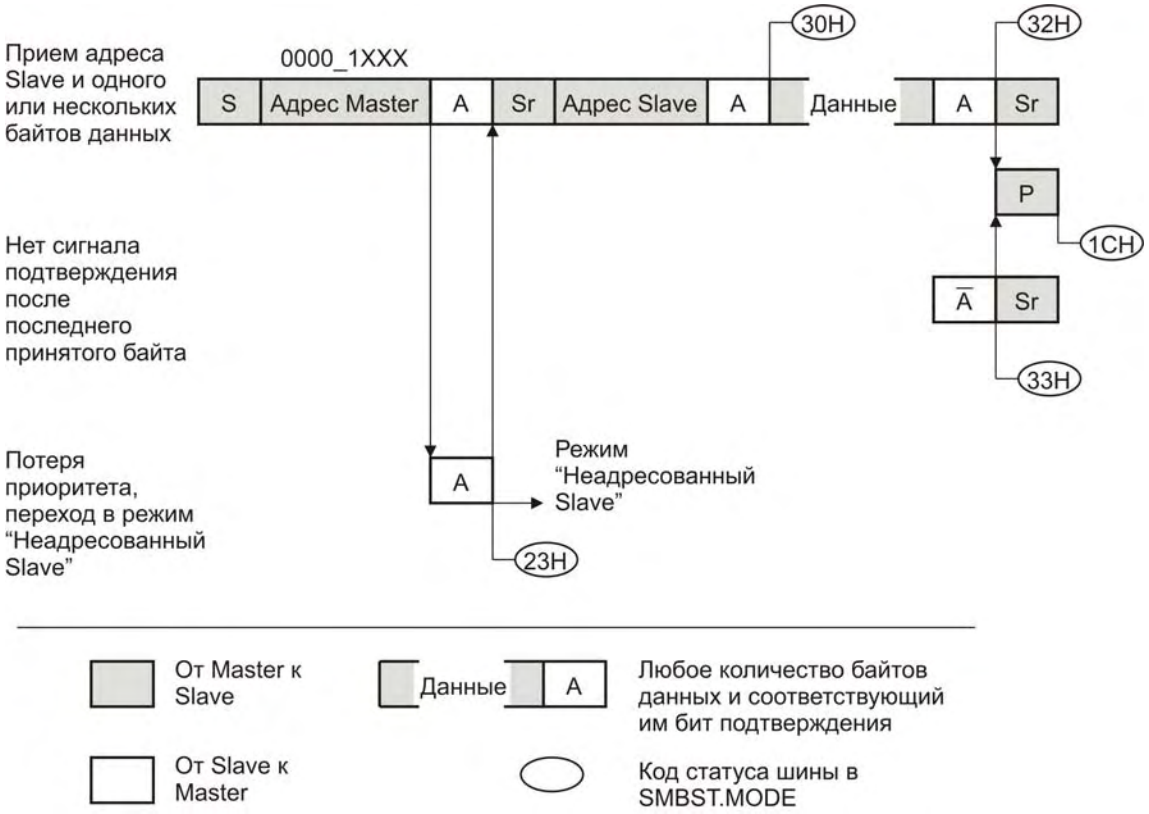


Рисунок 12.15 – Режим Hs «Slave приемник»

Режим «Slave передатчик»

В режиме «Slave передатчик» любое количество байтов данных может быть передано от I2C Master-приемнику. I2C проверяет бит подтверждения в течение девятого тактового импульса при передаче данных.

Вход в режим «Slave передатчик»

Вход в режим «Slave передатчик» осуществляется из режима «Slave приемник» после адресации I2C Master-передатчиком. Бит направления передачи, следующий после адреса Slave, должен быть равным единице (R/W# = 1_B). После этого установится флаг SMBST.INT, показывая необходимость записи данных в регистр SMBSDA. Линия SCL будет удерживаться в низком уровне, пока флаг SMBST.INT не будет сброшен. Когда первый байт передаваемых данных будет записан в регистр SMBSDA, необходимо программно установить бит SMBCTL1.CLRST, который сбросит флаг SMBST.INT. Затем линия SCL станет свободной, и будет передан байт данных.

Передача данных в этом режиме сходна с передачей в режиме «Master передатчик». Флаг SMBST.INT устанавливается после каждой успешно завершенной передачи данных, SMBST.MODE содержит соответствующий код статуса шины. Каждый следующий байт данных необходимо записывать в регистр SMBSDA, если SMBST.MODE не содержит код статуса шины STDANA, показывающий, что Master не может принять дополнительных данных.

Выход из режима «Slave передатчик» может быть осуществлен только с помощью Master-приемника. В этом случае Master-приемник формирует инверсный сигнал подтверждения после последнего принятого байта. После обнаружения инверсного сигнала подтверждения, I2C переходит в режим «Неадресованный Slave» (SMBST.MODE = IDLE) и ждет формирования на шине сигнала «Старт» (таблица 12.7).

Таблица 12.7 – Режим F/S «Slave передатчик»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
14_H	STADTA	Принят адрес Slave, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
15_H	STADPA	Принят адрес Slave после потери приоритета, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
16_H	STDAPA	Передача байта данных, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
17_H	STDANA	Передача байта данных, сигнал NACK	–		1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна
18_H	SATADP	Принят адрес отклика, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
19_H	SATAAP	Принят адрес отклика после потери приоритета, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
$1A_H$	SATDAP		W	Данные	1	X	0	0	Передача байта данных
$1B_H$	SATDAN				1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна

Режим «Slave передатчик» – 10-битная адресация

Для использования в режиме «Slave передатчик» 10-битной адресации должен быть записан адрес Slave и разрешена 10-битная адресация (см. « Режим «Slave приемник» – 10-битная адресация»).

Сначала I2C входит в режим «Slave приемник», чтобы принять полный 10-битный адрес Slave. Флаг SMBST.INT не будет установлен, линия SCL не будет удерживаться в низком уровне, статус кода шины в SMBST.MODE не изменится.

После формирования состояния «Повторный старт» последует второй байт адреса Slave, а затем – повторная передача старших битов адреса с битом направления передачи, равным единице (R/W# = 1_B). Если принятый адрес совпадет с адресом, записанным в регистрах, то будет установлен флаг SMBST.INT и I2C перейдет в режим «Slave передатчик», код статуса шины в поле SMBST.MODE будет STADPA или STAAPA (рисунок 12.16).

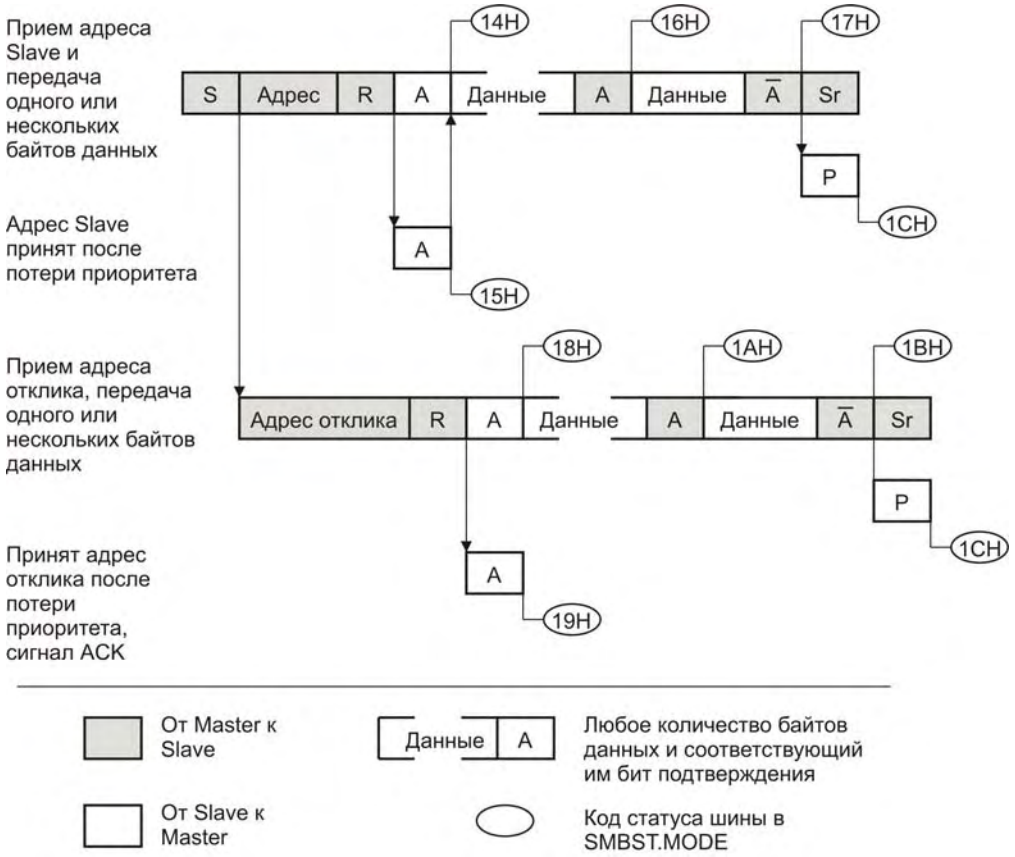


Рисунок 12.16 – Режим F/S «Slave передатчик»

Проверка ошибок пакета данных

При разрешении проверки ошибок пакета данных (PEC) последний байт, переданный Master-устройством, будет байтом PEC. Необходимо установить бит SMBCST.PECNEXT для записи байта PEC в регистр SMBSDA.

Режим Alert-отклика

Любой Slave (или несколько Slave) может послать сигнал запроса SMBAlert#, с целью передать данные к Master. Master, получив сигнал SMBAlert#, в ответ, инициализирует цикл отклика – выдает на шину специальный адрес отклика 0001100_B с битом R /W#, равным 1_B (чтение).

Устройство (устройства), пославшее сигнал SMBAlert# и получившее адрес отклика, выдает на шину свой 7-битный адрес (последний бит передаваемого байта может быть равным как 0_B, так и 1_B). Если несколько Slave выдают свои адреса на шину, то передачу осуществляет устройство, выигравшее арбитраж.

Master, который не послал адрес отклика (не инициализировал цикл отклика) в ответ на сигнал SMBAlert#, может сделать это через некоторое время.

Переход в режим Alert-отклика осуществляется после появления на шине SMBus адреса отклика (0001100_B), если установлен бит SMBCTL1.SMBARE.

Этот режим является единственным случаем, когда при функционировании I2C в качестве Slave используется арбитраж (при передаче адреса).

Высокоскоростной режим «Slave передатчик»

После передачи адреса Master ($00001XXX_B$) осуществляется вход в высокоскоростной режим. Затем следует формирование состояния «Повторный старт», передача адреса Slave с битом направления передачи, равным единице (R/W# = 1_B). После этого I2C переходит в режим «Slave передатчик». Работа в Hs режиме «Slave передатчик» почти идентична работе в режиме F/S, отличие заключается только в более высокой скорости передачи данных и в кодах статуса шины (таблица 12.8, рисунок 12.17).

Таблица 12.8 – Режим Hs «Slave передатчик»

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
34_H	HSTADPA	Принят адрес Slave, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
36_H	HSTDAPA	Передача байта данных, сигнал ACK	W	Данные	1	X	0	0	Передача байта данных
37_H	HSTDANA	Передача байта данных, сигнал NACK	–		1	X	0	0	«Неадресованный Slave»
					1	X	0	1	«Неадресованный Slave», «Старт» после того, как шина станет свободна

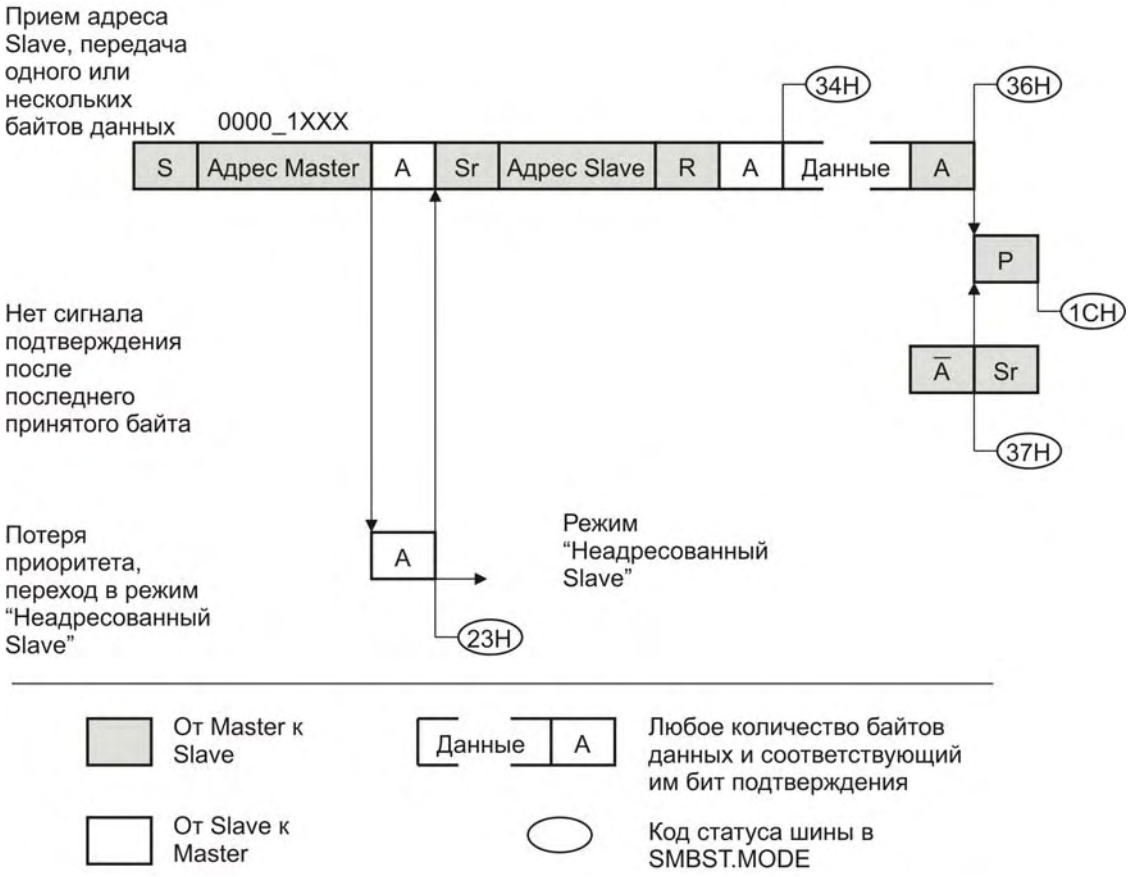


Рисунок 12.17 – Режим Hs «Slave передатчик»

Арбитраж и обнаружение ошибок на шине

Арбитраж

Приоритет в режиме «Master передатчик» может быть потерян в случаях, когда два Master одновременно формируют состояние «Старт» и начинают передачу данных. I2C по-разному реагирует на потерю приоритета в случаях, когда она произошла при передаче байта адреса или при передаче байта данных.

В случае потери приоритета при передаче байта адреса, I2C переходит в режим «Slave приемник» и начинает отслеживать передачу своего адреса Slave на шине. Если

адреса совпали, I2C остается в режиме Slave. Если адреса не совпали, I2C переходит в режим «Неадресованный Slave».

Как только произошла потеря приоритета при передаче байта данных, I2C переходит в режим «Неадресованный Slave».

Обнаружение ошибок на шине

Ошибки на шине случаются, если во время передачи адреса, данных, сигнала подтверждения обнаруживаются состояния «Старт» или «Стоп» (таблица 12.9). При нахождении ошибки на шине, I2C действует следующим образом:

Код статуса шины в SMBST.MODE становится BERROR ($1F_H$).

Генерируется прерывание, если был установлен бит SMBCTL1.INTEN.

I2C переходит из данного режима в режим «Неадресованный Slave».

Линии SDA и SCL становятся свободны.

Таблица 12.9 – Ошибки на шине, действия и статус

Код	Название	Описание	Данные		Управление: SMBCTL1				Действие
			R/W	SMBSDA	clrst	ask	stop	start	
00_H	IDLE	Информация о режиме отсутствует	–		–				Ожидание завершения текущей передачи данных
$1F_H$	BERROR	Обнаружены некорректные состояния «Старт» или «Стоп»	–		1	0	0	0	Режим «Неадресованный Slave»

Исправление ошибок на шине

В некоторых случаях I2C и другое устройство, подключенное к шине, могут обнаружить ошибку и вызвать простой шины. В этом случае может быть не завершено формирование состояния «Старт» и I2C перестанет функционировать.

Для возврата из этого состояния необходимо выполнить следующие действия:

- запретить и снова разрешить работу I2C (стачала обнулить бит SMBCTL2.ENABLE, а затем записать туда единицу);

- во время простоя проверить, не подключен ли другой активный Master к шине (бит SMBCST.BB остается равным нулю).

В этот момент некоторые Slave могут не распознать ошибку на шине. Для исправления ошибки I2C становится Master-устройством шины, формируя состояние «Старт» и передавая адрес. Затем I2C формирует состояние «Стоп», чтобы синхронизировать все Slave-устройства.

Конфигурация частоты тактового сигнала SCL

Когда I2C работает в режиме Master-устройства шины, то можно программно задавать частоту тактового сигнала на линии SCL.

Режим F/S

Величина, содержащаяся в поле SMBCTL2.SCLFRQ, определяет период тактового сигнала SCL относительно системного тактового сигнала. Также учитывается, что длительность тактовых импульсов на линии SCL определяется тактовым сигналом Master-устройства с наименьшей длительностью импульсов, а длительность паузы между тактовыми импульсами определяется тактовым сигналом Master-устройства с наибольшей паузой между импульсами.

Режим Hs

Величина, содержащаяся в поле SMBCTL3.HSDIV, определяет коэффициент деления тактового сигнала SCL относительно системного тактового сигнала. В

соответствии с протоколом шины, синхронизация тактового сигнала не применяется при работе в режиме Master, то есть другое Master-устройство не может уменьшить длительность тактового импульса. Величина в SMBCTL3.HSDIV определяет длительность тактового импульса на SCL, длительность паузы между тактовыми импульсами равна удвоенной длительности импульсов.

Время ожидания на линии SCL

Спецификация SMBus определяет наименьшее время ожидания на линии как TTIMEOUT = 25 мс. Если пауза между двумя тактовыми импульсами превысила TTIMEOUT, то устройство должно прервать текущую передачу. Master должен сформировать состояние «Старт» в процессе передачи или после ее окончания. Slave должен освободить шину. Устройства, обнаружившие данное состояние, должны восстановить свои соединения и ожидать формирования состояния «Старт» в пределах 10 мс.

Функциональное описание счетчика времени ожидания SCL

I2C содержит счетчик времени ожидания для обнаружения и уведомления о простое на шине (рисунок 12.18).

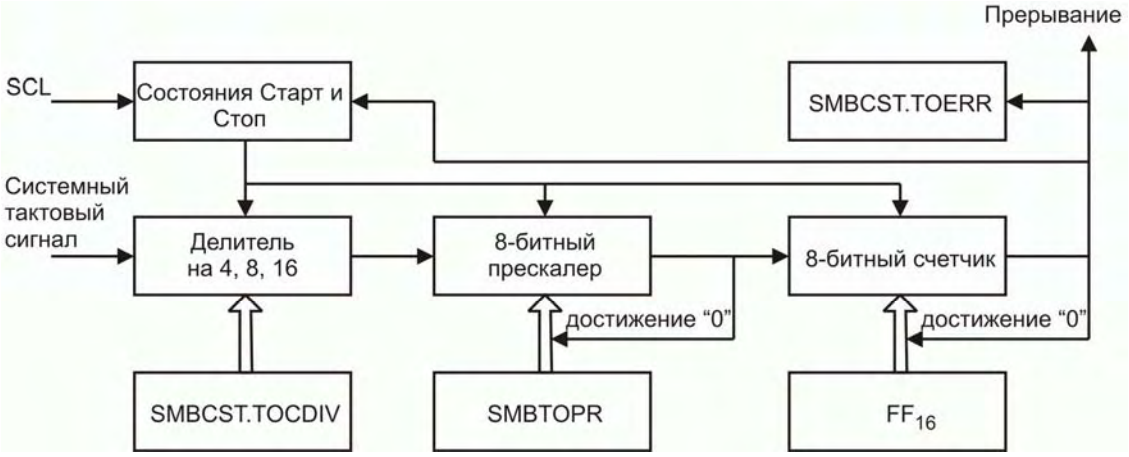


Рисунок 12.18 – Обнаружение времени ожидания

Счетчик времени ожидания включает в себя делитель, 8-битный программируемый предделитель (прескалер) и 8-битный основной счетчик. Все элементы счетчика времени ожидания начинают работу по отрицательному фронту на SCL (если работа счетчика разрешена). Положительный фронт на SCL сбрасывает значения делителя, предделителя и основного счетчика. Предделитель считает вниз, начиная с величины, записанной в регистр SMBTOPR. После достижения нуля счетчиком предделителя, он перезагружает значение из SMBTOPR. Основной счетчик считает вниз от величины FF_H. Каждое достижения нуля счетчиком предделителя декрементирует значение основного счетчика. Переход основного счетчика от нуля к FF_H вызывает остановку основного счетчика, предделителя и делителя, при этом устанавливается флаг SMBCST.TOERR. Дополнительно устанавливается флаг SMBST.INT. Прерывание генерируется, если был установлен бит SMBCTL1.INTEN.

Период времени ожидания определяется следующим выражением:

$$T_{ожид} = T_{сист} \times Div \times (SMBTOPR + 1) \times 256,$$

где $T_{сист}$ – период системного тактового сигнала;

Div – величина, определяемая SMBCST.TOCDIV (4, 8 или 16).

Подключение к порту и конфигурация

Подключение к порту и конфигурация показаны на рисунках 12.19, 12.20.

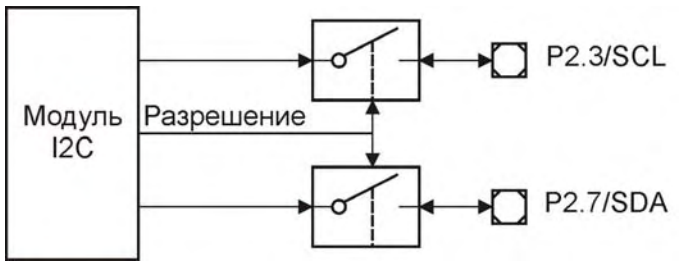


Рисунок 12.19 – Подключение модуля I2C к порту

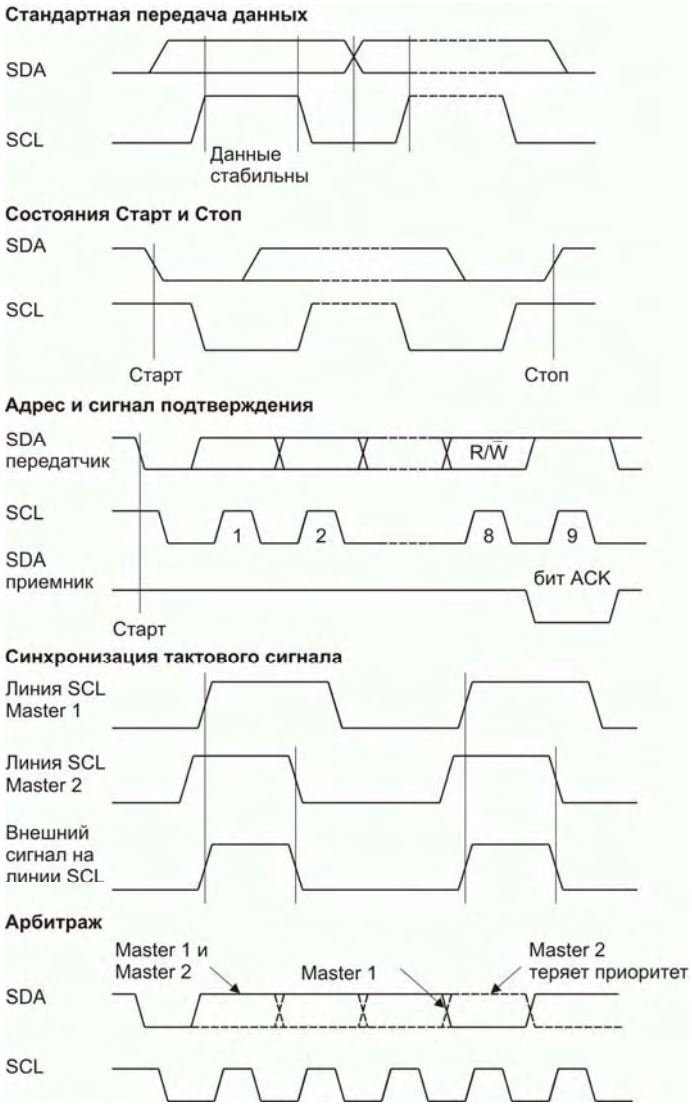


Рисунок 12.20 – Характеристики I2C

13 Синхронный последовательный интерфейс SPI

SPI (Serial Peripheral Interface) – интерфейс для последовательного обмена данными между микросхемами. Фактически это шина для подключения внешних устройств.

Основные области применения SPI интерфейса:

– датчики температуры, давления; цифровые потенциометры;

– АЦП и ЦАП;

– сенсорные экраны;

– звуковые кодеки;

– ЖК экраны;

– элементы программируемой логики (FPGA) и перепрограммируемой памяти (EEPROM) и др.

13.1 Общие определения

Четырехпроводной последовательный синхронный интерфейс SPI организуется по принципу «Master-устройство – Slave-устройство», 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.

В качестве Master-устройства (или просто Master) на шине SPI обычно выступает микроконтроллер, но им также может быть программируемая логика, DSP-контроллер или специализированная ИС. Подключенные к Master-устройству внешние устройства образуют Slave-устройства (или просто Slave) шины. В их роли выступают различного рода микросхемы, в т.ч. запоминающие устройства (EEPROM, Flash-память, SRAM), часы реального времени (RTC), АЦП/ЦАП, цифровые потенциометры, специализированные контроллеры и др.

Главным составным блоком интерфейса SPI является обычный сдвиговый регистр, сигналы синхронизации и ввода-вывода битового потока которого и образуют интерфейсные сигналы (рисунок 13.1). Таким образом, протокол SPI в общем случае является протоколом обмена данными между двумя сдвиговыми регистрами, каждый из которых одновременно выполняет и функцию приемника, и функцию передатчика. Непременным условием передачи данных по шине SPI является генерация сигнала синхронизации шины. Этот сигнал имеет право генерировать только Master шины и от этого сигнала полностью зависит работа Slave.

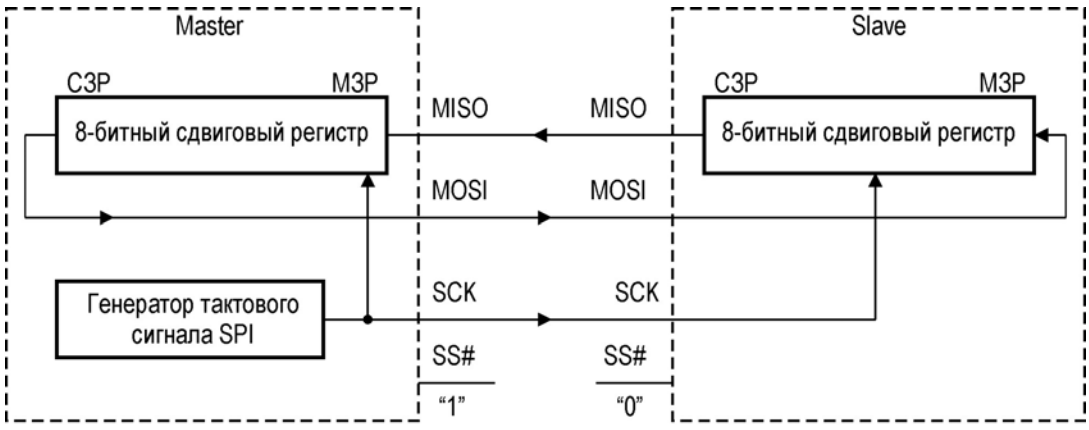


Рисунок 13.1 – Соединение Master-устройства и Slave-устройства при использовании интерфейса SPI

13.2 Электрическое подключение

Существует три типа подключения к шине SPI, в каждом из которых участвуют четыре сигнала (их основное и альтернативные обозначения сведены в таблицу 13.1).

Самое простое подключение, в котором участвуют только две микросхемы, показано на рисунке 13.1. В соединении участвуют четыре сигнала (их основное и альтернативные обозначения сведены в таблицу 13.1).

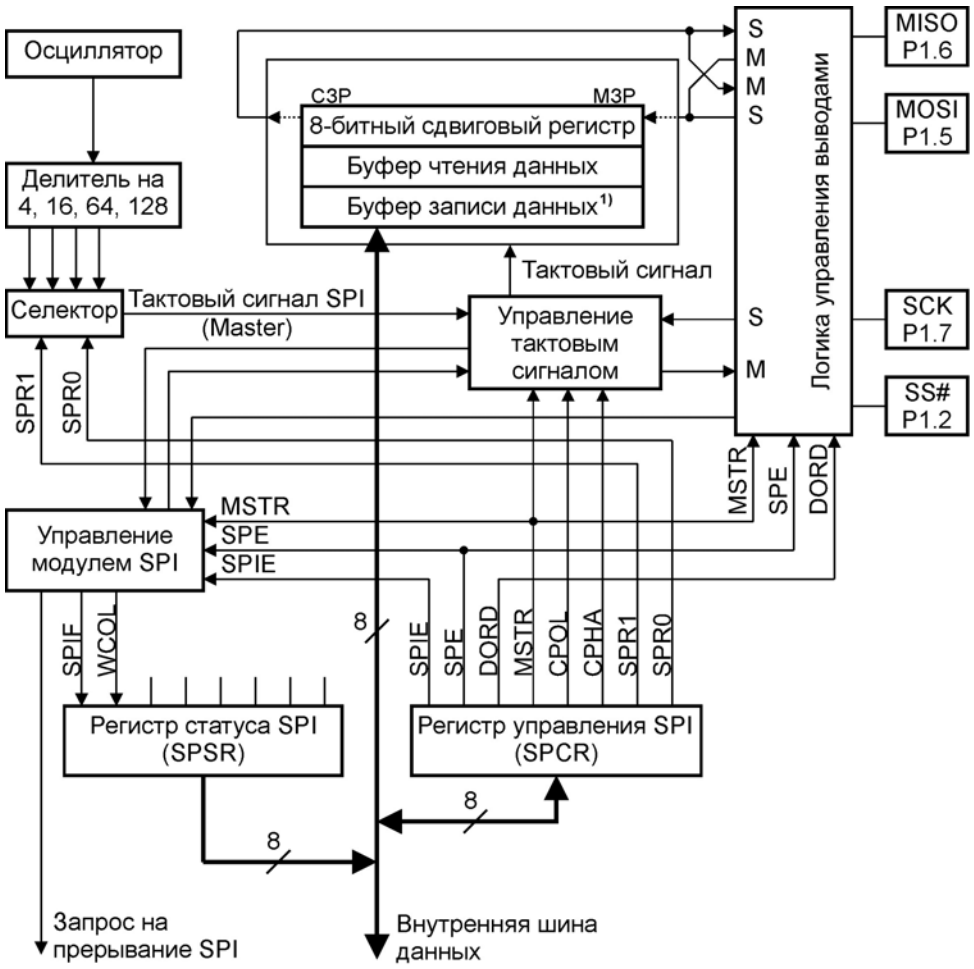
Master шины передает данные по линии MOSI синхронно с сгенерированным им же сигналом SCK, а Salve захватывает переданные биты данных по заданным фронтам принятого сигнала синхронизации. Одновременно с этим Slave отправляет свою посылку данных. Представленную схему можно упростить исключением линии MISO, если используемая в качестве Slave ИС не предусматривает ответную передачу данных или в ней нет потребности. Одностороннюю передачу данных можно встретить у таких микросхем как ЦАП, цифровые потенциометры, программируемые усилители и драйверы. Такой вариант подключения ИС в качестве Slave требует три линии связи.

Чтобы Master принимал и передавал данные, помимо наличия сигнала синхронизации, необходимо также, чтобы линия SS# (выбор Slave-устройства) была переведена в низкое состояние. В противном случае, Slave-устройство будет неактивно. Когда используется только одна внешняя ИС, может возникнуть желание исключения линии SS# за счет жесткой установки низкого уровня на входе выбора подчиненной микросхемы. Такое решение крайне нежелательно и может привести к сбоям или вообще невозможности передачи данных, т.к. вход выбора микросхемы служит для перевода ИС в её исходное состояние.

Таблица 13.1 – Электрические сигналы шины SPI

Master-устройство шины			Slave-устройство шины		
Основное обозначение	Альтернативное обозначение	Описание	Основное обозначение	Альтернативное обозначение	Описание
MOSI	DO, SDO, DOUT	Выход последовательной передачи данных	MOSI	DI, SDI, DIN	Вход последовательного приема данных
MISO	DI, SDI, DIN	Вход последовательного приема данных	MISO	DO, SDO, DOUT	Выход последовательной передачи данных
SCK	DCLOCK, CLK, SCLK	Выход синхронизации передачи данных	SCK	DCLOCK, CLK, SCLK	Вход синхронизации приема данных
SS#	CS	Выход выбора Slave-устройства (выбор микросхемы)	SS#	CS	Вход выбора Slave-устройства (выбор микросхемы)

13.3 Структура и функционирование блока SPI



1) - Буфер записи данных используется только в расширенном режиме работы SPI
СЗР - Старший значащий разряд
МЗР - Младший значащий разряд

Рисунок 13.2 – Структурная схема блока SPI

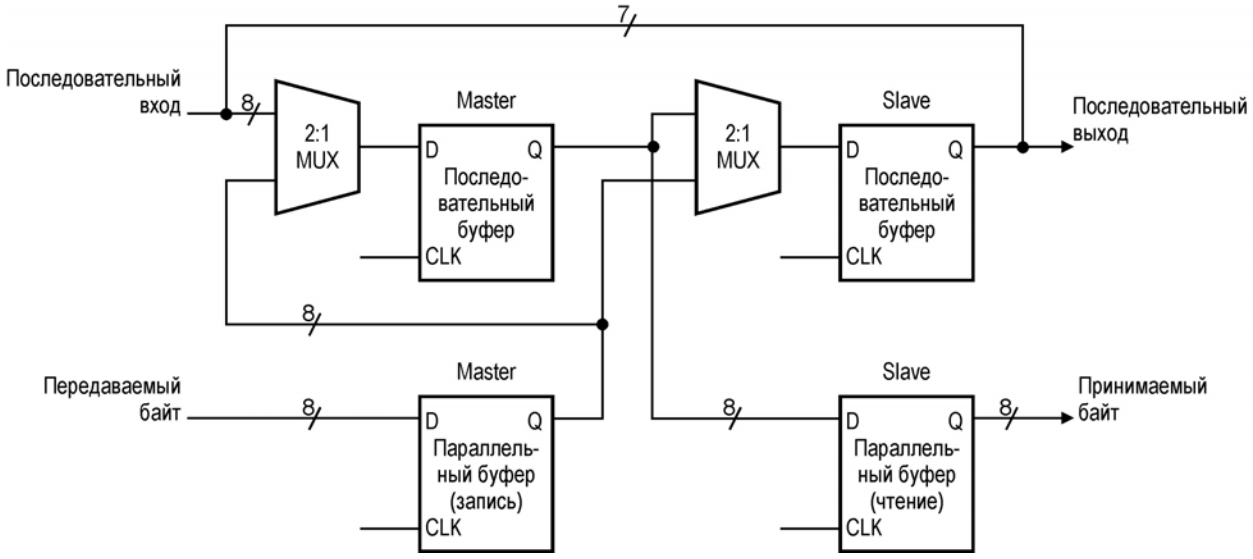


Рисунок 13.3 – Схема сдвигового регистра SPI

Два сдвиговых регистра Master-устройства и Slave-устройства можно рассматривать как один разнесенный 16-разрядный циклический сдвиговый регистр (рисунок 13.1). При сдвиге данных из Master в Salve одновременно происходит сдвиг данных из Slave в Master, т.е. в течение одного цикла сдвига происходит обмен данными между ведущим и ведомым микроконтроллерами.

Существует два режима работы SPI: стандартный (без использования буфера записи данных) и расширенный (с использованием буфера записи данных).

Стандартный режим

В стандартном режиме запись в регистр данных SPDR запускает работу генератора тактового сигнала Master-устройства, записанные данные начинают сдвигаться и передаваться с вывода MOSI и приниматься на выводе MOSI Slave-устройства. Передача может быть начата после начальной задержки, пока генератор тактового сигнала ожидает следующий полный битовый интервал при определенной скорости передачи. После сдвига одного байта генератор тактового сигнала останавливается и выставляется флаг окончания передачи SPIF в регистре SPSR. Принятый байт перемещается в буфер чтения регистра SPDR. Если одновременно установлены бит SPIE блока SPI и бит ES (разрешение прерывания последовательного порта), то генерируется прерывание.

Чтение данных из буфера после приема и запись данных в буфер для передачи осуществляются через обращение к регистру SPDR.

В стандартном режиме работы после записи данных в регистр SPDR, начинает работать сдвиговый регистр и осуществляться передача. В течение передачи регистр SPDR закрыт для обращений, поэтому попытка записи в это время данных будет проигнорирована, а в регистре SPSR установиться флаг WCOL. Передача будет продолжаться без изменений. По окончании передачи будет выставлен флаг SPIF в регистре SPSR.

Расширенный режим

Расширенный режим работы схож со стандартным режимом работы, за исключением того, что используется двойная буферизация данных. Первый записанный в SPDR байт попадает сразу в сдвиговый регистр и начинается передача. Следующий байт, записываемый в SPDR в течение передачи, попадает в буфер записи данных (рисунок 13.2), и одновременно с этим выставляется флаг WCOL, сигнализирующий о том, что дополнительный буфер заполнен.

Следует помнить, что буфер записи данных доступен для записи в течение всей передачи, поэтому при повторных обращениях будет перезаписан. Состояние флага WCOL при этом не будет изменяться – он будет оставаться установленным до окончания передачи.

По окончании передачи, байт, находящийся в этот момент в буфере записи данных, загружается в сдвиговый регистр и передача продолжается без остановки или перезапуска генератора тактового сигнала. Одновременно с этим аппаратно сбрасывается флаг WCOL.

Флаг разрешения загрузки LDEN в регистре SPSR может быть использован для определения начала передачи. LDEN установлен во время первых четырех битовых интервалов передачи.

Таким образом, записывая своевременно в регистр SPDR данные, можно осуществить передачу любого числа байт с минимальными временами задержками между посылками.

13.4 Регистры блока

Регистр управления блока SPI представлен на рисунке 13.4, а функциональные назначения его полей сведены в таблицу 13.2.

SPCR
регистр управления SPI

1FA0H значение после Сброса: 04H

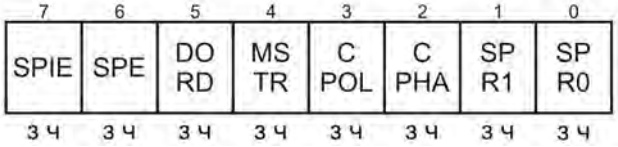


Рисунок 13.4 – Формат регистра SPCR

Таблица 13.2 – Функциональные назначения полей регистра SPCR

Поле	Биты	Тип	Описание
SPIE	7	Чтение Запись	Разрешение прерывания SPI. Данный бит вместе с битом SER_MASK в регистре INT_MASK разрешает прерывание SPI: при SPIE = 1 и SER_MASK = 1 – прерывание SPI разрешено. 0 Запрещение прерывания SPI 1 Разрешение прерывания SPI
SPE	6	Чтение Запись	Разрешение работы SPI. 0 Запрещение канала SPI 1 Разрешение канала SPI
DORD	5	Чтение Запись	Направление передачи данных. 0 Выбор MSB в качестве первого бита для передачи 1 Выбор LSB в качестве первого бита для передачи
MSTR	4	Чтение Запись	Выбор режима Master/Slave. 0 Выбор режима Slave SPI 1 Выбор режима Master SPI
CPOL	3	Чтение Запись	Полярность тактового сигнала. 0 На выводе SCK Master-устройства сигнал низкого уровня при отсутствии импульсов. 1 На выводе SCK сигнал высокого уровня при отсутствии импульсов
CPHA	2	Чтение Запись	Фаза тактового сигнала. Бит CPHA вместе с битом CPOL управляет соотношением фазы тактового сигнала и данных на шине (таблица 13.4)
SPR1, SPR0	1,0	Чтение Запись	Выбор частоты тактового сигнала SPI. Данные два бита управляют сигналом тактирования SCK Master-устройства. В случае работы устройства в режиме Slave биты SPR1 и SPR0 не действуют. Между SCK и частотой осциллятора f существуют следующие соотношения: SPR1 SPR0 SCK 0 0 f/4 (f/2 в режиме x2) 0 1 f/16 (f/8 в режиме x2) 1 0 f/64 (f/32 в режиме x2) 1 1 f/128 (f/64 в режиме x2)

П р и м е ч а н и я

1 Все параметры сигнала тактирования, а также направление передачи данных, должны быть выставлены до разрешения работы блока SPI. Только после этого можно устанавливать бит SPE.

2 Master-устройство должно быть включено раньше, чем Slave-устройство.

3 При необходимости выключения устройств, Master-устройство следует выключать первым.

Регистр состояния блока SPI представлен на рисунке 13.5, а функциональные назначения его полей сведены в таблицу 13.3.

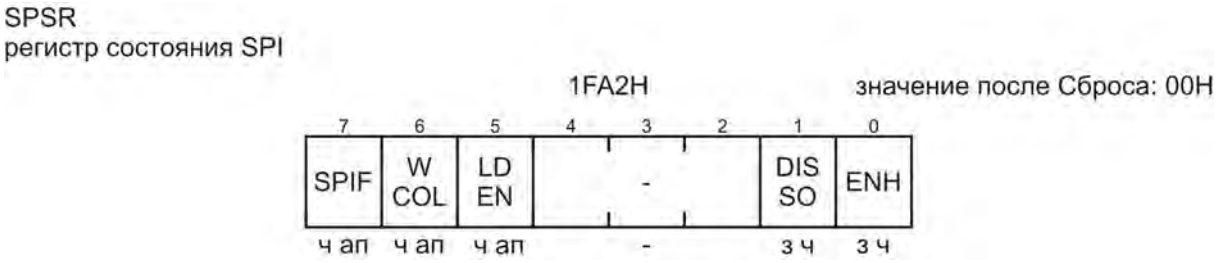


Рисунок 13.5 – Формат регистра SPSR

Таблица 13.3 – Функциональные назначения полей регистра SPSR

Поле	Биты	Тип	Описание
1	2	3	4
SPIF	7	Чтение Аппаратное влияние	Флаг прерывания SPI. При завершении операции передачи/приема устанавливается бит SPIF, и, в случае если бит SPIE = 1 и бит SER_MASK = 1, генерируется прерывание. После выставления флага SPIF можно снова записать данные в регистр SPDR или считать полученные данные из регистра SPDR. Состояние флага SPIF определяется чтением регистра SPSR, после чего, если есть необходимость сброса установленного флага, достаточно обратиться (запись/чтение) к регистру SPDR
WCOL	6	Чтение Аппаратное влияние	1. Стандартный режим (ENH = 0). Флаг конфликта записи. Флаг WCOL устанавливается при попытке записи в регистр SPDR во время передачи данных. Флаг WCOL сбрасывается одновременно с флагом SPIF. 2. Расширенный режим (ENH = 1). Флаг WCOL в расширенном режиме информирует о заполнении буфера записи данных. Если WCOL = 1, то запись в регистр SPDR приведет к перезаписи данных, уже хранящихся в буфере записи данных. В этом режиме WCOL сбрасывается только при перезагрузке данных из буфера записи данных в сдвиговый регистр
LDEN	5	Чтение Аппаратное влияние	Разрешение загрузки в буфер записи данных в расширенном режиме. При ENH = 1, LDEN = 1 и WCOL = 0, запись в буфер записи данных является надежной. Во время передачи байта выполняются условия: 1. При передачи первых четырех битов LDEN = 1 2. При передачи последних четырех битов LDEN = 0

Продолжение таблицы 13.3

1	2	3	4
DISSO	1	Чтение Запись	Бит запрещения работы передающего вывода Slave-устройства. При установке данного бита вывод MISO переходит в третье состояние, в случае, если устройство не выбрано сигналом SS#.
ENH	0	Чтение Запись	Бит выбора расширенного режима SPI. 0 SPI работает в стандартном режиме, то есть без двойной буферизации записи. 1 SPI работает в расширенном режиме с двойной буферизацией записи. Для буфера Tx используется такой же адрес, как и для регистра SPDR

Регистр данных SPDR представляет собой регистр с возможностью чтения/записи и предназначен для пересылки данных между регистровым файлом и сдвиговым регистром блока SPI. Запись в регистр SPDR инициирует передачу данных, считывание регистра приводит к чтению сдвигового регистра приема. Седьмой бит регистра является старшим значащим разрядом (MSB), а нулевой – младшим значащим разрядом (LSB). Формат регистра представлен на рисунке 13.6.

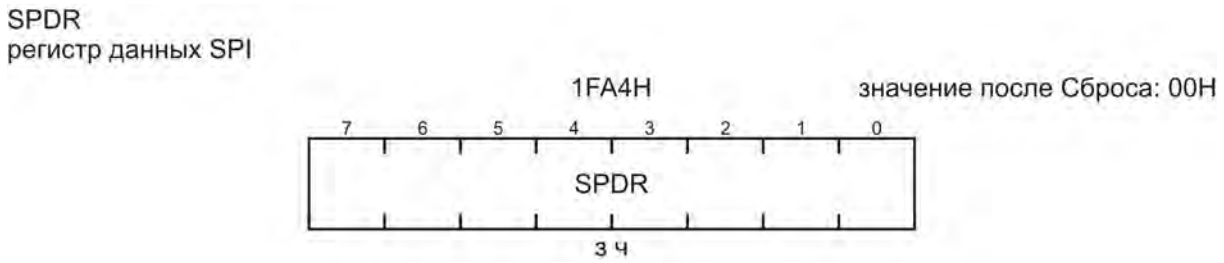


Рисунок 13.6 – Формат регистра SPDR

13.5 Протокол передачи

Функционирование входа SS#

Блок SPI в режиме Master (MSTR = 1)

В режиме Master вывод SS# не связан с блоком SPI и может использоваться как обычный вывод порта, например как выход сигнала выбора Slave-устройства.

Блок SPI в режиме Slave (MSTR = 0)

В режиме Slave вывод SS# постоянно работает на вход. Если на вывод SS# подан низкий уровень, то блок SPI активируется и вывод MISO (если это определено пользователем) становится выходом. Если вывод SS# удерживается на высоком уровне, то все выводы являются входами, блок SPI пассивен и не участвует в процессах на шине.

Сигнал тактирования SCK

Установка данных при передаче и выборка при приеме всегда выполняются по противоположным фронтам сигнала тактирования. Это необходимо для гарантирования выборки данных после надежного их установления. В качестве первого фронта в цикле передачи может выступать нарастающий или ниспадающий фронт сигнала тактирования.

Формой и частотой сигнала SCK управляют три бита регистра SPCR: бит CPHA (фаза тактового сигнала), бит CPOL (полярность тактового сигнала) и биты SPR0 и SPR1 (частота тактового сигнала, равная скорости передачи). Master является единственным устройством на шине SPI, который может управлять скоростью передачи данных и устанавливать параметры передачи.

При работе блока SPI в режиме Master с помощью битов SPR0 и SPR1 можно выбрать одну частоту (из четырех возможных) тактового сигнала. В режиме Slave блок

SPI будет работать с частотой тактового сигнала, поступающего на вход SCK. Также существует четыре возможные комбинации фазы и полярности тактового сигнала относительно последовательных данных. Биты CPHA и CPOL определяют, какой формат используется при передаче данных.

Примечание – Для корректной работы блока SPI биты CPHA, CPOL, SPR0 и SPR1 должны быть установлены до того, как будет разрешена работа блока SPI, при этом сначала должна быть разрешена работа Master-устройства, а затем Slave-устройства (Slave-устройств).

CPOL определяет активное состояние сигнала тактирования последовательной связи SPI и, следовательно, является базовым уровнем или уровнем «холостого хода», когда передачи информации на шине нет.

CPHA определяет фазу тактового сигнала относительно бита данных на выходах Master и Slave.

Комбинацией сигналов CPOL и CPHA можно задать один из четырех режимов работы блока SPI:

Режим 1 Тактовый сигнал имеет базовый уровень «0» и задержан на полпериода относительно передачи бита данных на линии MOSI. Выборка данных происходит в момент нарастающего фронта сигнала тактирования, установка данных происходит при спадающем фронте сигнала тактирования.

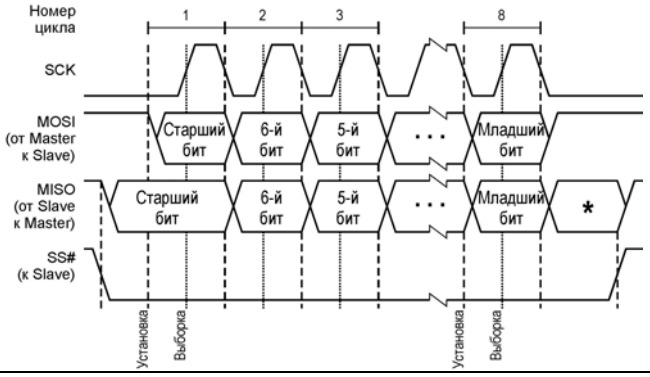
Режим 2 Тактовый сигнал имеет базовый уровень «1» и задержан на полпериода относительно передачи бита данных на линии MOSI. Выборка данных происходит в момент спадающего фронта сигнала тактирования, установка данных происходит при нарастающем фронте сигнала тактирования.

Режим 3 Тактовый сигнал синхронен битам данных на линиях MOSI и MISO. Выборка данных происходит в момент спадающего фронта сигнала тактирования, установка данных происходит при нарастающем фронте сигнала тактирования.

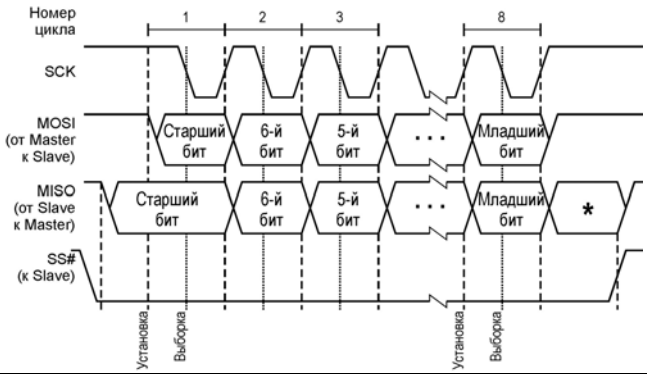
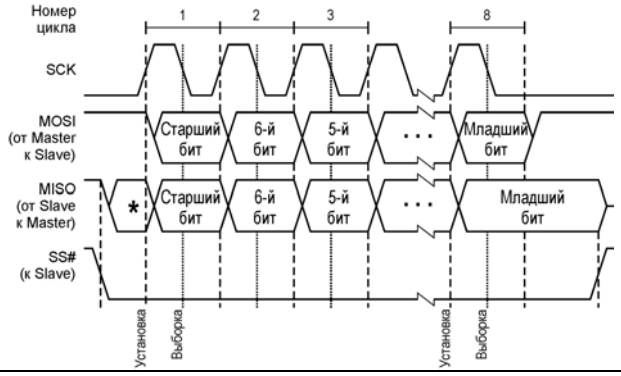
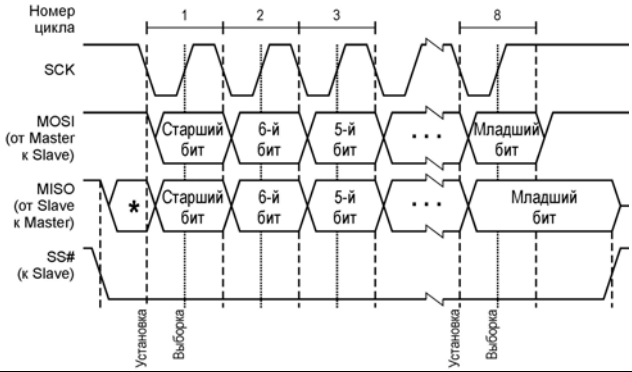
Режим 4 Тактовый сигнал синхронен битам данных на линиях MOSI и MISO. Выборка данных происходит в момент нарастающего фронта сигнала тактирования, установка данных происходит при спадающем фронте сигнала тактирования

Временные диаграммы работы блока SPI в каждом из четырех режимов сведены в таблицу 13.4.

Таблица 13.4 – Временные диаграммы работы порта

Номер режима	Значение бита	Временная диаграмма передачи одного байта данных
1	2	3
1	CPOL = 0 CPHA = 0	

Окончание таблицы 13.4

1	2	3
2	CPOL = 1 CPHA = 0	
3	CPOL = 0 CPHA = 1	
4	CPOL = 1 CPHA = 1	

* Не определено, но обычно это младший бит переданного байта.

Направление передачи данных

В большинстве SPI-форматов первым на шину передается старший разряд (MSB). Но также передача данных может начинаться передачей первым младшего разряда (LSB). За направление передачи данных отвечает бит DORD регистра SPCR.

Примечание – Master и Slave, работающие в различных режимах, являются несовместимыми, поэтому, перед выбором микросхем важно уточнить, какие режимы поддерживаются каждой из них.

Временные диаграммы

Как уже было сказано выше, следует помнить, что для корректной работы блока SPI биты CPHA, CPOL, SPR0 и SPR1 должны быть установлены до того, как будет разрешена работа блок SPI.

Master-устройство всегда включается раньше, чем Slave-устройство (Slave-устройства, если несколько).

Диаграммы синхронизации временных интервалов во время передачи данных между Master и Slave приведены на рисунках 13.7 – 13.10. Параметры передачи данных и обозначения, принятые на рисунках 13.7 – 13.10, сведены в таблицы 13.5 и 13.6.

Таблица 13.5 – Параметры в режиме Master

Обозначение	Параметр	Минимальное время, нс	Максимальное время, нс
t_{CLCL}	Период осциллятора	25,0	
t_{SCK}	Период сигнала тактирования	$4t_{CLCL}$	
t_{SHSL}	Длительность высокого логического уровня	$t_{SCK}/2$ (уточн.)	
t_{SLSH}	Длительность низкого логического уровня	$t_{SCK}/2$ (уточн.)	
t_{SR}	Время формирования переднего фронта сигнала тактирования		5 (уточн.)
t_{SF}	Время формирования заднего фронта сигнала тактирования		5 (уточн.)
t_{SIS}	Время стабилизации данных до момента выборки	5 (уточн.)	
t_{SIH}	Время удержания данных после выборки	5 (уточн.)	
t_{SOH}	Время удержания данных после начала формирования фронта тактового сигнала, следующего после фронта выборки, до начала установки новых данных		5 (уточн.)
t_{SOV}	Время формирования новых данных		10 (уточн.)

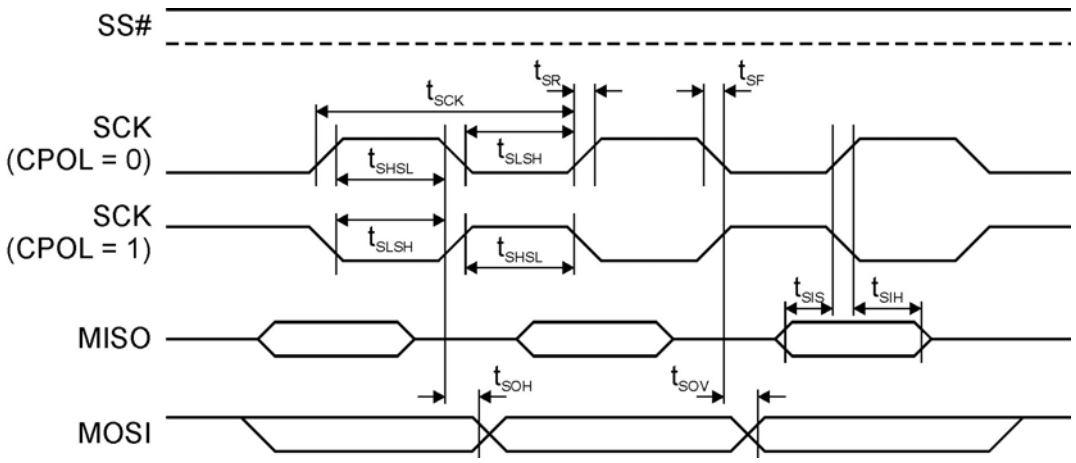


Рисунок 13.7 – Временные диаграммы для Master-устройства (CPHA = 0)

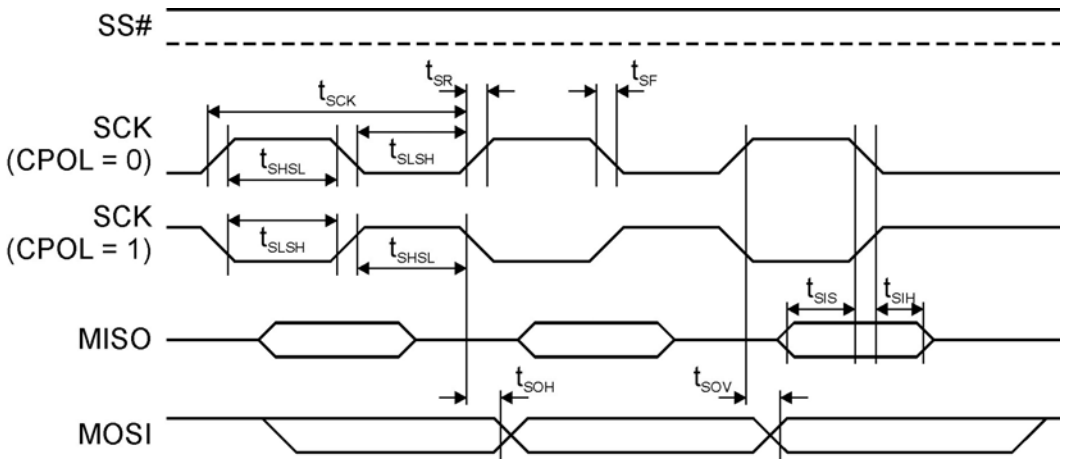


Рисунок 13.8 – Временные диаграммы для Master-устройства (CPHA = 1)

Таблица 13.6 – Временные параметры в режиме Slave

Обозначение	Параметр	Минимальное время, нс	Максимальное время, нс
t_{CLCL}	Период осциллятора	25,0	
t_{SCK}	Период сигнала тактирования	$4t_{CLCL}$	
t_{SHSL}	Длительность высокого логического уровня	$1{,}5t_{SCK}$ – 5(уточн.)	
t_{SLSH}	Длительность низкого логического уровня	$1{,}5t_{SCK}$ – 5(уточн.)	
t_{SR}	Время формирования переднего фронта сигнала тактирования		5(уточн.)
t_{SF}	Время формирования заднего фронта сигнала тактирования		5(уточн.)
t_{SIS}	Время стабилизации данных до момента выборки	5(уточн.)	
t_{SIH}	Время удержания данных после выборки	5(уточн.)	
t_{SOH}	Время удержания данных после начала формирования фронта тактового сигнала, следующего после фронта выборки, до начала установки новых данных		5(уточн.)
t_{SOV}	Время формирования новых данных		5(уточн.)
t_{SOE}	Время формирования данных Slave-устройства от момента выбора этого устройства		5(уточн.)
t_{SOX}	Время удержания данных после снятия сигнала выбора Slave-устройства		5(уточн.)
t_{SSE}	Время стабилизации данных Slave-стройства после выбора этого устройства и до начала формирования первого фронта сигнала тактирования	$4t_{CLCL}$ + 5(уточн.)	
t_{SSD}	Время от последнего фронта сигнала тактирования до момента начала снятия сигнала выбора Slave-устройства	0(уточн.)	

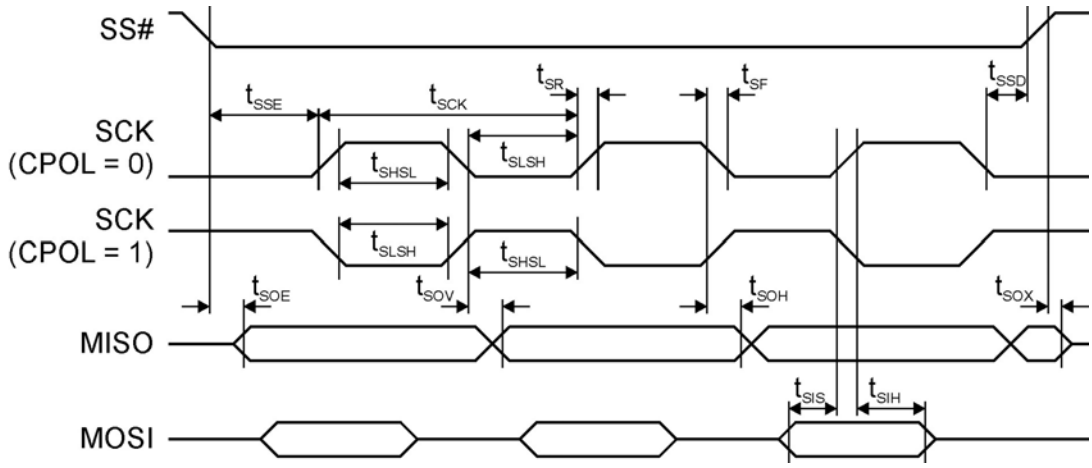


Рисунок 13.7 – Временные диаграммы для Slave-устройства (CPHA = 0)

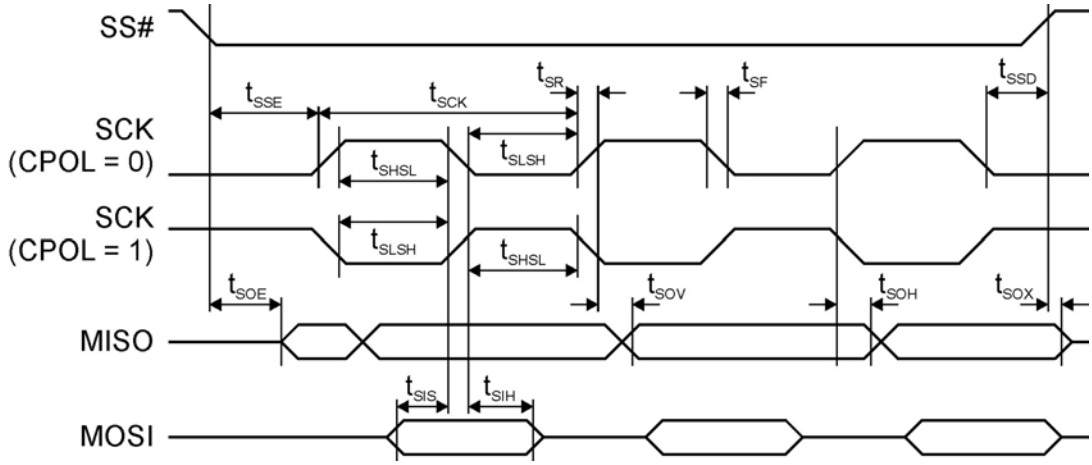


Рисунок 13.7 – Временные диаграммы для Slave-устройства (CPHA = 1)

SPI-совместимые и производные протоколы

- MICROWIRE. Протокол MICROWIRE компании National Semiconductor полностью идентичен протоколу SPI в режиме 0 (CPOL = 0, CPHA = 0).

- 3-проводной интерфейс компании Maxim. Отличие этого интерфейса состоит в том, что вместо полнодуплексной передачи по двум однонаправленным линиям здесь выполняется полудуплексная передача по одной двунаправленной линии DQ.

- QSPI. Более высокоуровневый протокол, чем SPI, позволяющий автоматизировать передачу данных без участия ЦПУ.

Интерфейс блока SPI

В зависимости от выбранного режима работы блока SPI, направления передач сигналов могут быть противоположными.

В случае работы блока в режиме Master, все его выводы (за исключением вывода MISO, который соединен с выводом порта P1.6 и является входом) являются выходами для передачи управляющих сигналов и данных к ведомым устройствам. Вывод порта P1.2 при этом может использоваться как выход сигнала выбора Slave-устройства.

В случае работы блока в режиме Slave, все его выводы (за исключением вывода MISO, который соединен с выводом порта P1.6 и является выходом) являются входами для приема управляющих сигналов и данных от ведущего устройства.

Примечание – Следует помнить, что для корректной работы порта SPI необходимо, чтобы значения битов IOPORT1.7, IOPORT1.6 и IOPORT1.5 были равны единицам, так как выходы порта SPI объеденины функциями логического «И» с соответствующими битами регистра IOPORT1.

Схема интерфейса блока SPI представлена на рисунке 13.8.

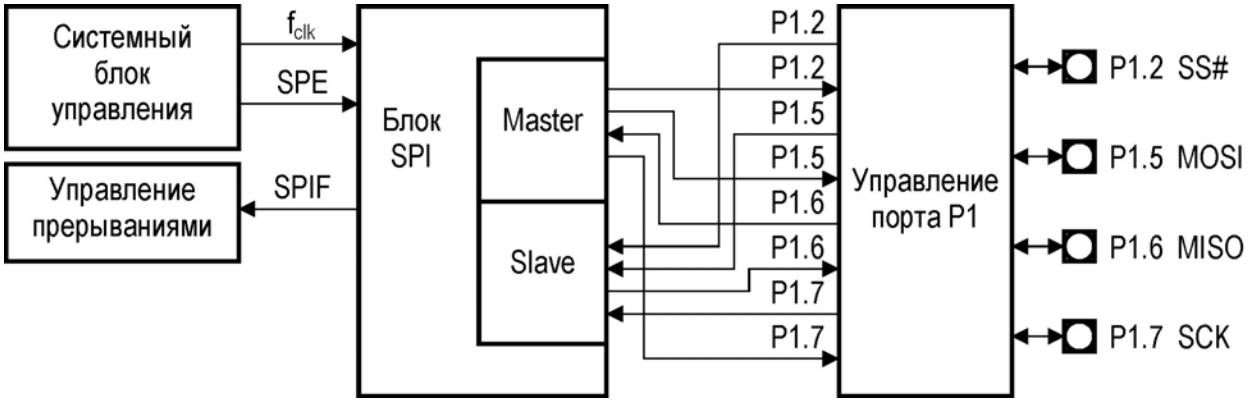


Рисунок 13.8 – Схема интерфейса блока SPI

14 Генератор псевдослучайных последовательностей PRNG

Проблема шифрования больших сообщений и потоков данных появилась сравнительно недавно – с появлением средств мультимедиа и сетей с высокой пропускной способностью, обеспечивающих передачу мультимедийных данных.

В современных ИС и информационных системах начинают применяться технологии, которые требуют передачи больших объемов данных. Это немыслимо без использования современных технологий шифрования.

Наиболее распространенным является потоковое шифрование данных. Система защиты не ждет, когда закончится передаваемое сообщение, а сразу же осуществляет его шифрование и передачу (рисунок 14.1).

Наиболее очевидным является побитовое сложение входящей последовательности (сообщения) с некоторым бесконечным или периодическим ключом, получаемым например, от генератора ПСП или генератора псевдослучайной последовательности.

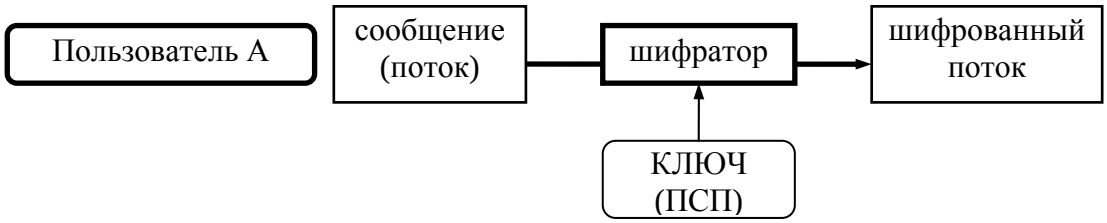


Рисунок 14.1 – Принцип потокового шифрования данных

Такой генератор псевдослучайной битовой последовательности (PseudoRandom Number Generator, или PRNG) реализован в микроконверторе 1874BE96T на основе сдвиговых регистров с линейной обратной связью. Последовательности сдвиговых регистров используются как в криптографии, так и в теории кодирования. Их теория прекрасно проработана.

14.1 Принципы построения схем генерации псевдослучайных последовательностей

Сдвиговый регистр с обратной связью состоит из двух частей: сдвигового регистра и функции обратной связи. Сдвиговый регистр представляет собой последовательность битов. Количество битов определяется длиной сдвигового регистра. Если длина равна n битам, то регистр называется n-битовым сдвиговым регистром. Всякий раз, когда нужно извлечь бит, все биты сдвигового регистра сдвигаются вправо на одну позицию. Новый крайний левый бит является функцией всех остальных битов регистра. На выходе сдвигового регистра оказывается один, обычно младший значащий, бит. Периодом сдвигового регистра называется длина получаемой последовательности до начала ее повторения.

Простейшим видом сдвигового регистра с обратной связью является линейный сдвиговый регистр с обратной связью (linear feedback shift register, или LFSR). Обратная связь представляет собой просто функцию «исключающее ИЛИ» некоторых битов регистра, перечень этих битов называется отводной последовательностью. Иногда такой регистр называется конфигурацией Фиббоначи.

Для того чтобы конкретный LFSR имел максимальный период, многочлен, образованный из отводной последовательности и константы 1, должен быть примитивным по модулю 2. Степень многочлена является длиной сдвигового регистра.

Сами по себе LFSR являются хорошими генераторами псевдослучайных последовательностей, но они обладают некоторыми нежелательными неслучайными свойствами. Последовательные биты линейны, что делает их бесполезными для

шифрования. Для LFSR длины n внутреннее состояние представляет собой предыдущие n выходных битов генератора. Даже если схема обратной связи хранится в секрете, она может быть определена по 2n выходным битам генератора с помощью высокоэффективного алгоритма Berlekamp-Massey.

Кроме того, большие случайные числа, генерируемые с использованием идущих подряд битов этой последовательности, сильно коррелированны и для некоторых типов приложений вовсе не являются случайными. Несмотря на это, LFSR часто используются для создания алгоритмов шифрования.

Основной подход при проектировании генератора потока ключей на базе LFSR прост. Сначала берется один или несколько LFSR, обычно с различными длинами и различными многочленами обратной связи. Если длины взаимно просты, а все многочлены обратной связи примитивны, то у образованного генератора будет максимальная длина. Ключ является начальным состоянием регистров LFSR. Бит выхода представляет собой функцию, желательно нелинейную, некоторых битов регистров LFSR. Эта функция называется комбинирующей функцией, а генератор в целом – комбинационным генератором.

Можно ввести ряд усложнений. В некоторых генераторах для различных LFSR используется различная тактовая частота, иногда частота одного генератора зависит от выхода другого. Управление тактовой частотой может быть с прямой связью, когда выход одного LFSR управляет тактовой частотой другого LFSR, или с обратной связью, когда выход одного LFSR управляет его собственной тактовой частотой.

Хотя все эти генераторы чувствительны, по крайней мере теоретически, к вскрытиям вложением и вероятной корреляцией, многие из них безопасны до сих пор.

В микроконверторе генератор ПСП реализован по типу – двусторонний генератор «стоп – пошел» (рисунок 14.2). В этом генераторе используется два LFSR с одинаковой длиной n. Выходом генератора является «исключающее ИЛИ» выходов каждого LFSR. Если выход LFSR-1 в момент времени t-1 равен нулю, а в момент времени t-2 – единице, то LFSR-2 не тактируется в момент времени t. Наоборот, если выход LFSR-2 в момент времени t-1 равен «0», а в момент времени t-2 – «1», и если LFSR-2 тактируется в момент времени t, то LFSR – 1 не тактируется в момент времени t.

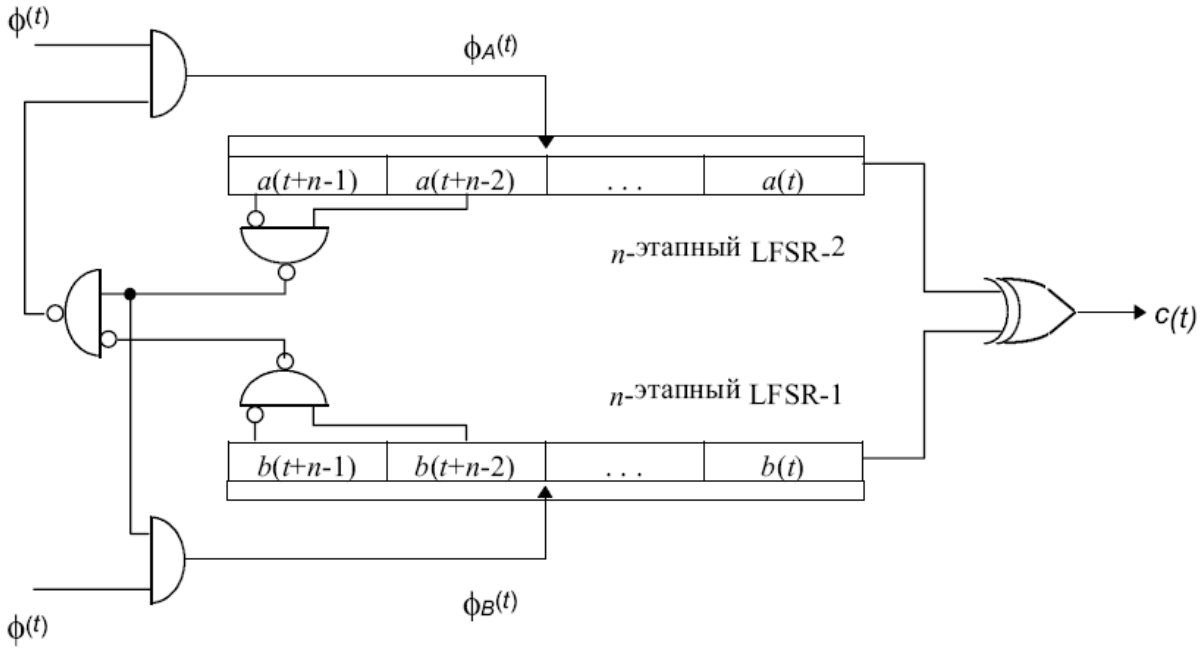


Рисунок 14.2 – Двусторонний генератор «стоп – пошел»

14.2 Управление генератором псевдослучайных последовательностей

Для ИС 1874BE96T в качестве линейных сдвиговых регистров с обратной связью выбраны 16-битные регистры с отводной последовательностью, обеспечивающей максимальный период такого генератора.

Управление генератором ПСП в микроконверторе осуществляется с помощью четырех SFR:

- PRNGREG1 (рисунок 14.3);
- PRNGREG2 (рисунок 14.4);
- PRNGCON (рисунок 14.5);
- PRNGRESULT (рисунок 14.6).

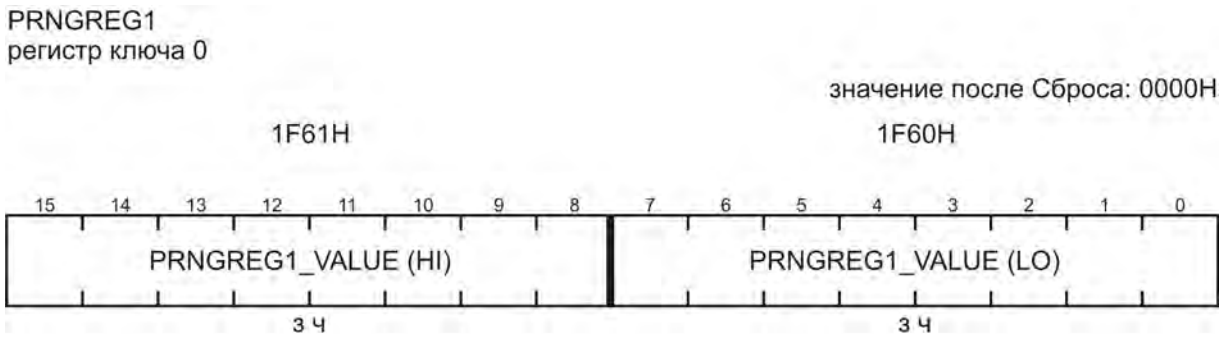


Рисунок 14.3 – Регистр управления PRNGREG1

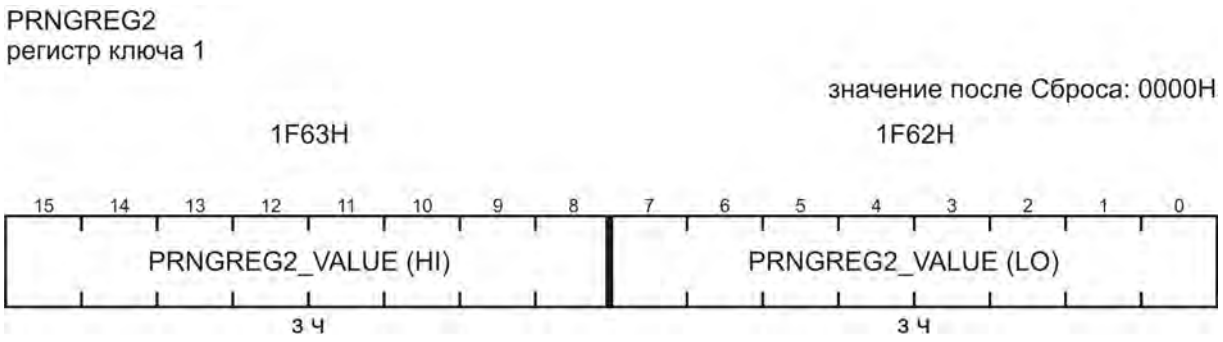


Рисунок 14.4 – Регистр управления PRNGREG2

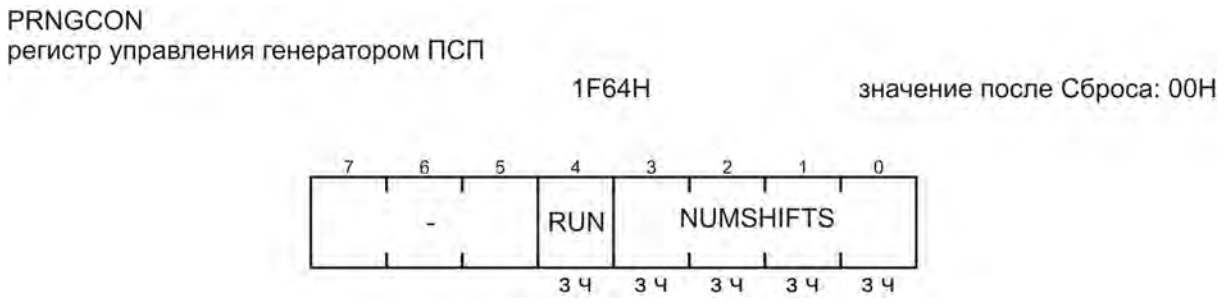


Рисунок 14.5 – Регистр управления PRNGCON

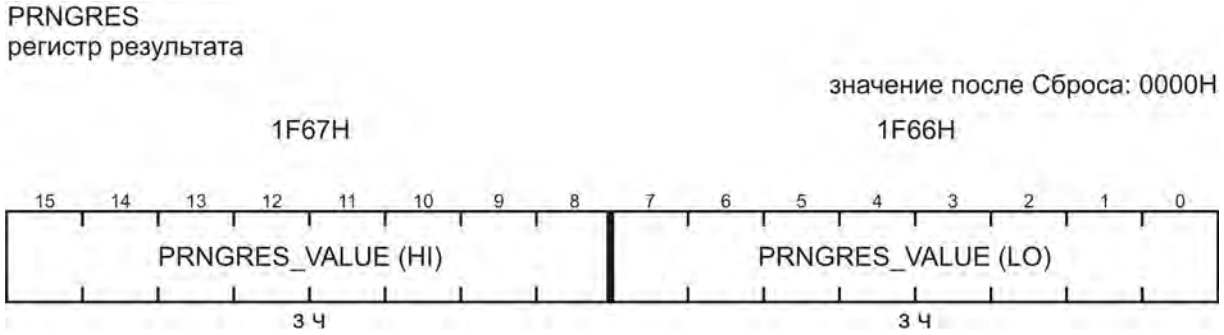


Рисунок 14.6 – Регистр результата PRNGRES

Регистры PRNGREG1 и PRNGREG2 перед началом работы генератора необходимо проинициализировать значениями (отличными от нуля), которые и будут являться ключом. По одному и тому же ключу генератор будет выдавать одинаковую последовательность битов. Регистр PRNGRES является регистром, с которого можно снимать выходную псевдослучайную последовательность битов, в виде слов, байтов или битов. Как видно на рисунке 14.5, в регистре биты 7–5 зарезервированы (не используются), бит 4 представляет собой бит, который надо устанавливать каждый раз, чтобы получить необходимое число сдвигов, а по окончании операции он сбрасывается в «0», биты 3–0 – определяют количество сдвигов, которое надо произвести. При этом чтобы определить количество сдвигов, надо к числу, которое находится в битах 3–0, прибавить «1», т. е. «0» – это один сдвиг, «1» – это два сдвига, и так далее, вплоть до числа 15, которому будет соответствовать 16 сдвигов.

Для повышения криптостойкости рекомендуется совместно со встроенным генератором комбинировать собственные программные реализации LFSR и подобных генераторов. Это приведет к усложнению схемы данного генератора и к увеличению его периода, что позволит использовать его в более «серьезных» приложениях. Несмотря на то, что программные реализации LFSR медленны, существуют определенные приемы, позволяющие увеличить производительность. Например, рекомендуется писать подобные генераторы на ассемблере, а не на «Си». Также, чтобы вычислить новый левый бит для LFSR за одну операцию, возможно использование битовых масок, которые позволят работать со словами размерности регистров, а не делать это за несколько операций для каждого бита в отдельности.

Если в микроконверторе генератор ПСП использовать не для шифрования, а например, при измерениях, то встроенный генератор без дополнительных методов повышения криптостойкости вполне сможет удовлетворить запросы.

15 Устройство высокоскоростного ввода-вывода

МК 1874BE96T содержит четыре устройства, которые вместе формируют гибкое, базируемое на таймере-счетчике устройство скоростного ввода-вывода (HSIO). Устройства, входящие в HSIO: таймер 1, таймер 2, модуль высокоскоростного ввода и модуль высокоскоростного вывода (рисунок 15.1). HSIO может измерять ширину импульсов, вырабатывать импульсную последовательность и вызывать периодические прерывания с незначительным использованием ЦПУ. В этом раздел описывается каждый модуль внутри HSIO устройства.

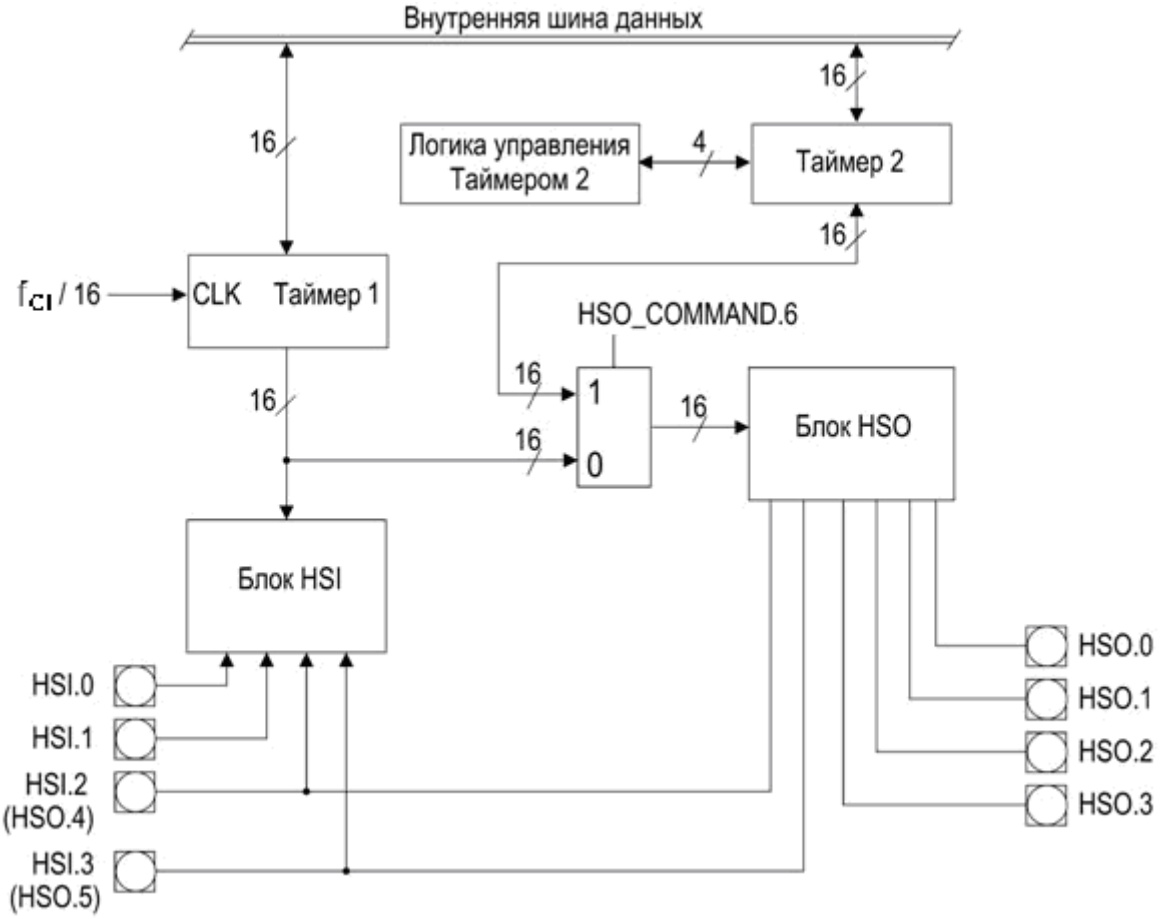


Рисунок 15.1 – Блок-схема модуля HSIO (HSI + HSO)

15.1 Таймеры

МК имеет два 16-битных таймера: таймер 1 и таймер 2. HSI-модуль всегда использует таймер 1 как базовый. HSO-модуль использует в качестве базового таймер 1 или таймер 2.

15.1.1 Функции таймера 1

Таймер 1 является стандартным 16-битным независимым таймером, который увеличивается на каждый восьмой такт процессора. Таймер 1 всегда является базовым для HSI-модуля. Он также может быть выбран как базовый для HSO-модуля. Два байта регистра TIMER1 содержат его значение. Вы можете инициализировать таймер 1, записав отличное от нуля значение в регистр TIMER1 через HWindow15. Если Вы изменили значение TIMER1 после инициализации HSI-модуля, Вы можете исказить временное соотношение между HSI-событиями. Изменение значения TIMER1 после инициализации HSO-модуля, если таймер 1 является базовым, может вызвать потерю HSO команд.

15.1.2 Функции таймера 2

Таймер 2 – реверсивный программируемый 16-битный счетчик, который может являться базовым для HSO-модуля или использоваться как высокоскоростной счетчик. Он может также фиксировать внешние события. Таймер 2 может тактироваться внешним или внутренним источником. При внешнем источнике в качестве источников тактов можно выбрать выводы T2CLK или HSI.1. Максимальная скорость счета для таймера 2 – каждый такт ЦПУ (быстрый режим) или каждый восьмой такт (нормальный режим).

Два байта регистра TIMER2 содержат его значение. Можно инициализировать таймер 2, записав отличное от нуля значение в регистр TIMER2 через HWindow0. Когда используется таймер 2 как базовый для HSO, изменение его значения после инициализации модуля HSO может вызвать пропуск HSO команд. Дополнительная информация – в пункте 15.1.6.

15.1.3 Программирование модуля таймера 2

В таблице 15.1 показаны регистры, влияющие на работу и функции модуля таймера 2.

Таблица 15.1 – Регистры управления и статуса таймера 2

Мнемоника	Имя	Описание
TIMER 2	Timer 2	Содержит значения таймера 2
T2CAPTURE	Timer 2 Capture	Нарастающий фронт на P2.7 вызывает захват значения таймера 2 в этом регистре и производит прерывание Timer 2 Capture (INT11)
INT_MASK INT_MASK1	Interrupt Mask InterruptMask1	Разрешение или запрещение прерываний таймера 2
IOC0	Input/Output Control Register 0	Выбирает внешний источник синхронизации и сброса для таймера 2
IOC1	Input/Output Control Register 1	Выбирает источники прерывания для прерывания Timer Overflow (INT07)
IOC2	Input/Output Control Register 2	Разрешает или запрещает быстрый режим и работу счетчика на увеличение или уменьшение, выбирает границу переполнения для прерывания Timer Overflow (INT12)
IOC3	Input/Output Control Register 3	Выбирает внутренний или внешний источник синхронизации для таймера 2

В таблице 15.2 показаны отдельные биты и выводы порта, которые управляют режимами таймера 2. В графе 2 показан результат установки бита или вывода порта. В графе 3 показан результат очистки бита или вывода порта.

Таблица 15.2 – Выводы и биты для управления таймером 2

Биты регистра, выводы порта	Бит равен «1»	Бит равен «0»
1	2	3
IOC 0.1	Сброс таймера 2 при каждой записи	Не действует
IOC 0.3	Разрешение внешнего сброса	Запрещение внешнего сброса
IOC 0.5	HSI.0 является источником внешнего сброса	T2RST (P2.4) является внешним источником сброса
IOC 0.7	HSI.1 является источником внешних	T2CLK (P2.3) является источником

Окончание таблицы 15.2

1	2	3
IOC 1.3	Разрешение таймера 2 как источника прерывания Timer Overflow (INT00)	Запрещение таймера 2 как источника прерывания Timer Overflow (INT00)
IOC 2.0	Разрешение режима быстрого увеличения	Запрещение режима быстрого увеличения
IOC 2.1	Разрешение счета на уменьшение	Счет только на увеличение
IOC 2.5	Прерывание на границе 7FFFH/8000H	Прерывание на границе 0FFFFH/0000H
T2UP-DN (P2.6)	Счет на уменьшение, если IOC2.1=1	Счет на увеличение, если IOC2.1 = 1
T2CAP (P2.7)	Захват данных из таймера 2 в T2CAPTURE имеет место, когда происходит положительный перепад и логический уровень сохраняется стабильным в течение одного машинного цикла	

Выбор источника тактового сигнала

Таймер 2 считает оба перепада – положительный и отрицательный. Он может тактироваться или внутренне или внешне в зависимости от состояния бита T2_ENA в регистре IOC3 (IOC3.0). Установка IOC3.0 выбирает внутренний источник синхронизации; сброс этого бита выбирает внешний источник (смотри рисунок 15.2).

Когда выбран внешний источник синхронизации, бит T2CLK_SRC в регистре IOC0 (IOC0.7) управляет выбором одного из двух внешних источников. Установка IOC0.7 выбирает HSI.1 вывод как источник внешней синхронизации, очистка выбирает вывод T2CLK (P2.3) как внешний источник синхронизации.

Когда таймер 2 синхронизируется внешне, то бит FAST_T2_ENA в регистре IOC2 (IOC2.0) управляет максимальной входной частотой на внешнем выводе. Когда установлен FAST_T2_ENA, максимальная входная скорость ввода – один раз в такт (быстрый режим). Когда FAST_T2_ENA очищен, максимальная скорость ввода – один раз в восемь тактов (нормальный режим).

Когда таймер 2 тактируется внутренне, бит FAST_T2_ENA определяет значение входной тактовой частоты. Когда устанавливается FAST_T2_ENA, тактовая частота равна f_{CI} /4, и таймер 2 увеличивается каждый такт (быстрый режим). Когда FAST_T2_ENA очищен, тактовая частота равна f_{CI} /32, и таймер 2 увеличивается каждый восьмой такт.

Если таймер 2 является базовым для HSO-модуля, используется нормальный режим. HSO-модуль требует восемь тактов для выборки всех CAM вводов. HSO событие может быть пропущено, если таймер 2 использует быстрый режим (пункт 15.1.6).

Установка направления счёта

Бит T2UP_ENA в регистре IOC2 (IOC2.1) управляет таймером 2. Таймер считает либо только на увеличение, либо на увеличение или уменьшение в зависимости от значения вывода T2UP_DN (см. рисунок 15.2). Если IOC2.1 очищен, таймер 2 всегда считает на увеличение. Если IOC2.1 установлен и на T2UP_DN низкий уровень, таймер 2 считает на увеличение. Если IOC2.1 установлен и на T2UP_DN высокий уровень, таймер 2 считает на уменьшение.

Сигнал T2UP_DN должен быть стабилен до изменения сигнала T2CLK или во время его изменения, чтобы быть уверенным, что таймер изменит направление счета в следующем такте.

Когда таймер 2 является базовым для HSO-модуля, не следует изменять направление счета, иначе события могут произойти в неправильном порядке (пункт 15.1.6).

Выбор сброса таймера 2

Таймер 2 может быть сброшен аппаратно, программно или модулем HSO (рисунок 15.2). Установка бита T2RST_ENA (IOC0.3) разрешает источник внешнего сброса. Установка бита T2RST_SRC (IOC0.5) выбирает вывод HSI.0, а очистка этого бита выбирает T2RST (P2.4) как внешний сигнал сброса таймера 2. Нарастающий фронт выбранного сигнала сбрасывает таймер 2.

Программа может сбросить этот таймер установкой SW_T2RST бита (IOC0.1). HSO может сбросить этот таймер выполнением команды RESET Timer2 (CMD_TAG = 0EH). Этот вариант сброса обычно применяется, когда HSO-модуль используется как широтно-импульсный модулятор (PWM) (пункт 15.3.3).

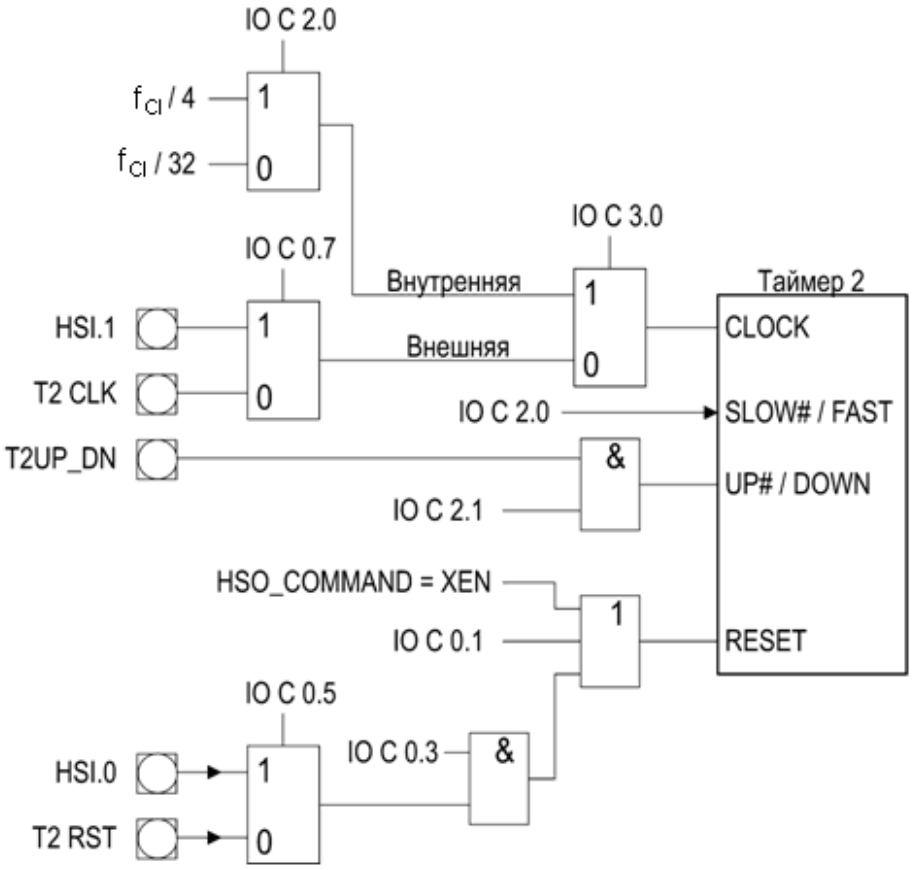


Рисунок 15.2 – Логика управления таймером 2

15.1.4 Использование внешних входов таймера 2

Значения на выводах T2UP_DN, T2CLK, T2RST и T2CAP фиксируются, когда CLKOUT имеет низкий уровень. Эти входы должны быть стабильными, по крайней мере в течение одного такта, либо должны удерживать свое значение не менее 45 нс до нарастающего фронта CLKOUT, чтобы гарантировать правильное распознавание сигналов.

Синхронизация

T2RST всегда синхронизируется от внутреннего счетчика по модулю 8. Общее время до сброса таймера зависит от того, когда активизируется T2RST. Сброс таймера может осуществиться с первого по девятый такт после установки T2RST. Это действует как для нормального, так и для быстрого режима счета.

Во время режима нормального счета T2CLK и T2CAP также синхронизируются от внутреннего счетчика по модулю 8. Во время быстрого режима эти сигналы подаются прямо на таймер 2. В случае, когда оба сигнала устанавливаются одновременно,

внутренняя схема таймера 2 гарантирует, что захват осуществится перед увеличением счетчика.

Одновременная установка T2RST, T2CLK и T2CAP

Во время одновременной установки T2RST, T2CLK и T2CAP независимо от режима счета (режим нормального счета или режим быстрого счета) и состояния внутреннего счетчика по модулю 8, выполняется следующая последовательность действий:

Последовательность событий	Содержимое TIMER2	Содержимое T2CAPTURE
1 Захват содержимого TIMER2	5A56H	5A56H
2 Сброс TIMER2	0000H	5A56H
3 Инкремент TIMER2	0001H	5A56H

15.1.5 Прерывания таймера

С таймером 1 и таймером 2 связаны три вектора прерывания:

- прерывание по переполнению таймера (Timer Overflow);
- прерывание по переполнению таймера 2 (Timer2 Overflow);
- прерывания по фиксации таймера 2 (Timer2 Capture).

Регистр IOS1 содержит флаги, которые показывают, какое событие вызвало прерывание. Чтение регистра IOS1 очищает биты $0-5$. Поэтому рекомендуется скопировать содержимое регистра IOS1 в рабочий регистр и затем выполнить команды проверки битов, такие как JBC или JBS, с рабочим регистром.

Прерывание по переполнению таймера

Таймер 1 и таймер 2 могут вызвать прерывание по переполнению таймера Timer Overflow (INT00). Установка INT_MASK.0 разрешает это прерывание. Установка IOC1.2 (таймер 1) или IOC1.3 (таймер 2) выбирает источник прерывания. Когда происходит переполнение, флаг состояния устанавливается в IOS1 регистре. Переполнение таймера 1 устанавливает IOS1.5, а переполнение таймера 2 устанавливает IOS1.4.

Прерывание по переполнению таймера 2

Таймер 2 может вырабатывать прерывание по переполнению таймера 2 (Timer 2 Overflow – INT12) взамен стандартному прерыванию по переполнению таймера (Timer Overflow). Это прерывание разрешается установкой INT_MASK1.4. Переполнение Timer2 устанавливает IOS1.4.

Таймер 2 может вырабатывать прерывание по переполнению таймера 2 при переходе границы 0FFFFH/0000H или 7FFFH/8000H. Переполнение может осуществляться в любом направлении. IOC2.5 выбирает границу переполнения. Когда установлен IOC2.5, таймер 2 вырабатывает прерывание на границе 7FFFH/8000H. В противном случае прерывание происходит на границе 0FFFFH/0000H.

Прерывание по захвату значения таймера 2 (Timer2 Capture)

Положительный перепад на выводе T2CAP вызывает загрузку значения таймера 2 в регистр T2CAPTURE. Это событие вырабатывает прерывание Timer2 Capture (INT11), если INT_MASK1.3 установлен и T2CAP удерживается более чем два такта.

15.1.6 Рекомендации при работе с таймером

Когда таймеры используются как базовые в HSI или HSO модулях, соблюдение этих рекомендаций поможет избежать возможных проблем. В этом пункте приведен перечень возможных проблем и рекомендаций.

Следует быть осторожным при записи в регистры таймеров TIMER1 и TIMER2.

Необходимо конфигурировать таймер 2 для работы в нормальном режиме (не в режиме быстрого счёта).

Следует конфигурировать таймер 2 для счета в одном направлении.

Необходимо быть осторожным при сбросе таймера 2.

Когда таймер 2 сконфигурирован на сброс от внешнего вывода, программные события могут не осуществиться при значении таймера 2, равном нулю.

Изменение значения таймера 1 после инициализации модуля HSI может исказить временное соотношение между HSI событиями. Также изменение опорного значения таймера (1 или 2) после инициализации модуля HSO может вызвать потерю или неправильный порядок запрограммированных HSO событий.

HSO требует один такт для выполнения одной команды в CAM. Но возможна ситуация, когда CAM содержит несколько команд для одного времени таймера. В этом случае HSO требуется число тактов, соответствующее количеству команд в CAM для одного времени таймера, или даже восьми тактов для полного выполнения всех команд в CAM, если они все должны быть выполнены в одно время таймера 2 при использовании его как базового для HSO. Поэтому таймер 2 должен работать в режиме нормального счёта (а не в режиме быстрого счёта). Очистка бита FAST_T2_ENA (IOC2.0) устанавливает режим нормальной работы.

В случае если CAM содержит несколько команд для одного значения времени таймера 2, когда таймер 2 используется как базовый для HSO и работает в режиме быстрого счета, будет выполнена лишь одна команда CAM. Остальные команды не будут выполнены. Кроме того, если не установлен бит разрешения фиксации в CAM команд LOCK_ENA (бит 6) в регистре IOC2 или бит LOCK_ENA установлен, но не был установлен бит фиксации в CAM команды CAM_LOCK в регистре HSO_COMMAND время программирования команды, команда будет стерта из CAM. Все остальные команды, фиксация которых в CAM разрешена, останутся в CAM и будут выполнены в следующем цикле таймера 2.

Использование режима быстрого счета рекомендуется в случае необходимости быстрого оперативного выполнения команд HSO при условии отсутствия в CAM файле команд, имеющих одинаковую временную метку.

Не следует сбрасывать таймер 2 до достижения им максимального значения времени, запрограммированного в CAM. CAM удерживает ожидаемое событие до тех пор, пока не осуществится временное соответствие. Если значение таймера 2 не достигается, событие будет оставаться ожидаемым до тех пор, пока устройство не сбросится или CAM не очистится.

Когда таймер 2 сконфигурирован на сброс внешним выводом (установлен IOC0.3), не следует программировать события, осуществляющиеся при значении таймера 2, равном нулю. Событие может не осуществиться, если HSI.0 или T2RST (P2.3) сбрасывают таймер 2. Очистка таймера 2 внешними выводами асинхронна, и таймер 2 может увеличиться на единицу до того, как HSO сможет сравнить и вызвать событие из CAM. Программирование событий при значении таймера 2, равном единице, гарантирует наличие достаточного количества времени для опознавания модулем HSO входа в CAM.

15.2 Модуль высокоскоростного ввода HSI

HSI модуль работает от четырех внешних выводов (HSI.0–HSI.3). Когда предопределенные события осуществляются на одном или более выводах, HSI модуль записывает текущее значение счетчика таймера 1 вместе с битом состояния для каждого входа. HSI модуль сохраняет данные максимум для семи событий в FIFO очереди размером (7×20) бит и для одного дополнительного события в фиксирующем регистре. Данные переносятся из FIFO в регистр захвата и могут быть считаны только после загрузки их в этот регистр.

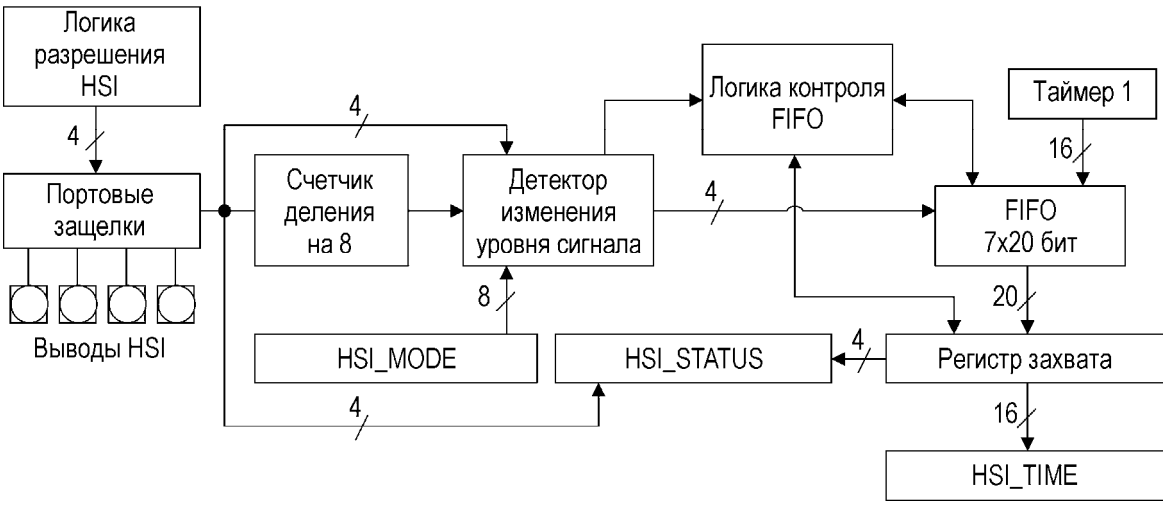


Рисунок 15.3 – Блок-схема модуля HSI

HSI (см. блок-схему модуля HSI на рисунке 15.3) может быть запрограммирован на захват каждого положительного перепада и отрицательного перепада, любого перепада или на захват последовательности из восьми положительных перепадов. Эта гибкость позволяет использовать HSI модуль для измерения различных входных параметров (длительность импульсов, период, скважность, сравнение фаз и т. д.). Запись последовательности из восьми положительных перепадов позволяет считать и измерять короткие импульсы.

15.2.1 Временные соотношения для событий в HSI

События загружаются в HSI FIFO с максимальной скоростью – одно событие в один такт. Если вывод сконфигурирован в режиме восьми переключений, максимальная скорость для положительных фронтов – восемь раз в восемь тактов, т. е. одно переключение в один такт.

Первое событие в пустом FIFO будет передаваться в регистр через один такт. Если второе событие осуществится после перемещения первого в регистр захвата, с ним обращаются как с первым событием в пустом FIFO. Дополнительные события не записываются, если и FIFO, и регистр захвата заполнены.

Если первые два события в пустом FIFO (исключая регистр захвата) осуществляются во время одной внутренней фазы, оба они записываются с одинаковым временным значением.

Логика обнаружения изменения фиксирует значение на HSI выводах, когда CLKOUT имеет низкий уровень. Чтобы гарантировать, что событие HSI запишется с правильным значением таймера 1, HSI вход должен быть стабильным, по крайней мере один машинный такт, или должен устанавливаться не менее, чем за 45 нс перед нарастающим фронтом CLKOUT. Если эти условия не выполнились, значение HSI входа может быть зафиксировано в следующем CLKOUT. Следовательно, если необходимо синхронизироваться с HSI модулем, то следует фиксировать информацию по нарастающему фронту CLKOUT.

Внутренний счетчик-делитель на восемь в HSI модуле постоянно считает перепады. Когда программа на входе HSI обнаруживает каждый восьмой положительный перепад, счетчик не сбрасывается (он сохраняет накопленное значение и продолжает считать). Если накопленное значение равно шести, потребуется только два дополнительных положительных перепада для фиксации HSI события. По этой причине лучше пропустить первое HSI событие, когда используется режим восьми перепадов.

15.2.2 Чтение информации о событии

Информация о событии может быть считана только после загрузки ее в регистр захвата. Данные из FIFO перемещаются в регистр захвата в течение одного такта. Когда событие перемещается из FIFO в регистр захвата, устанавливается бит HSI_RDY в регистре IOS1 (IOS1.7). Если прерывание разрешено и событие выбрано в качестве источника прерывания, генерируется прерывание HSI Data Available (INT02) (подробнее в пункте 15.2.3).

Для чтения фиксирующего регистра первым необходимо считать регистр HSI_STATUS. Это заставляет регистр захвата загрузить биты состояния события для HSI.0 – HSI.3 в четные биты регистра HSI_STATUS. Установленные биты состояния показывают, что событие произошло на соответствующем HSI выводе. Нечетные биты всегда содержат текущее состояние выводов HSI.0 – HSI.3.

Следующим читается регистр HSI_TIME. Регистр захвата загружает значение счетчика TIMER1 в регистр HSI_TIME. Чтение регистра HSI_TIME заставляет FIFO загрузить следующее событие в регистр захвата. Если вы читаете регистр HSI_TIME до первого чтения регистра HSI_STATUS, то биты состояния события будут перезаписаны.

15.2.3 Программирование HSI модуля

В таблице 15.3 показаны регистры, которые влияют на работу и на функции модуля HSI.

Таблица 15.3 – Регистры состояния и управления HSI

Мнемоника регистра	Имя регистра	Описание
HSI_MODE	Режим HSI	Описывает тип перепада, который вызывает событие на каждом выводе HSI
HSI_STATUS	Состояние HSI	Содержит биты обнаружения события и входные текущие уровни для каждого HSI вывода
HSI_TIME	Время HSI	Содержит 16-битное значение таймера 1 после того, как событие произошло
INT_MASK INT_MASK1	Маска прерывания Маска прерывания 1	Разрешает и запрещает прерывания от HSI
IOC0	Регистр 0 управления входом/выходом	Разрешает и запрещает каждый вывод HSI. Управляет альтернативными функциями для HSI.0 и HSI.1
IOC1	Регистр 1 управления входом/выходом	Управляет источником для HSI Data Available Interrput (INT02) прерывания
IOS1	Регистр 1 состояния входа/выхода	Показывает состояние HSI FIFO

Описание HSI события

Регистр HSI_MODE указывает для каждого вывода HSI какой из четырех типов событий вызовет захват в HSIFIFO: каждый положительный перепад, каждый отрицательный перепад, любой перепад (и положительный, и отрицательный), последовательность из восьми положительных перепадов (рисунок 15.4).

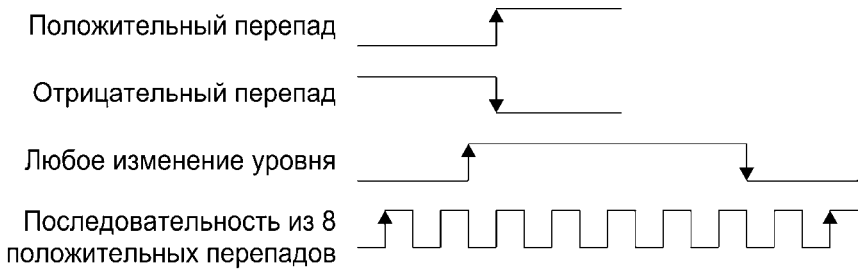


Рисунок 15.4 – Возможные события для HSI

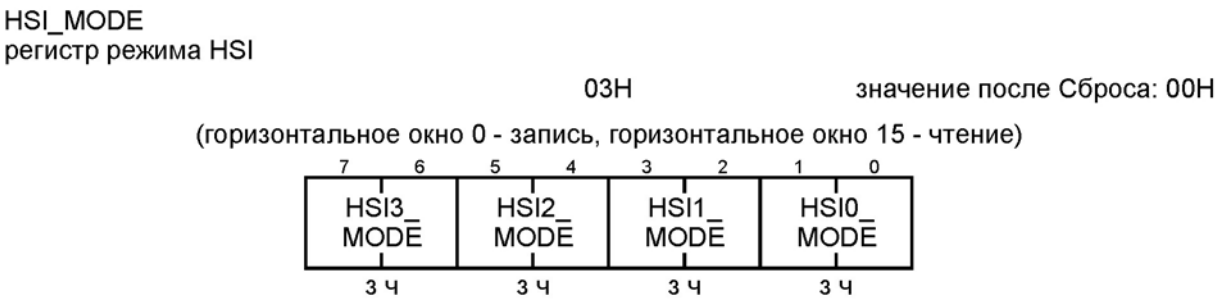


Рисунок 15.5 – Формат регистра HSI_MODE

Каждое двухбитное поле в регистре HSI_MODE программируется для того, чтобы определить режим перепада на соответствующем входе HSI (рисунок 15.5 и таблица 15.4).

Таблица 15.4 – Расшифровка режимов перепада

Биты	Описание
00	Восемь положительных перепадов вызывают захват в HSI FIFO
01	Каждый положительный перепад вызывает захват в HSI FIFO
10	Каждый отрицательный перепад вызывает захват в HSI FIFO
11	Каждый перепад (и положительный, и отрицательный) вызывает захват в HSI FIFO

Разрешение прерываний от HSI

HSI может вызывать прерывание, когда происходит одно из следующих событий:

Четвертое значение загружается в FIFO.

Шестое значение загружается в FIFO.

Значение из FIFO перемещается в регистр захвата.

Кроме того, вывод HSI.0 может быть сконфигурирован для работы как независимое внешнее прерывание. Необходимо запретить его как HSI вход, чтобы он функционировал как внешнее прерывание.

Регистры INT_MASK и INT_MASK1 содержат биты, которые разрешают и запрещают каждое прерывание. В таблице 15.5 кратко описываются условия, вызывающие каждое HSI прерывание, порядок прерываний и приоритеты.

Таблица 15.5 – Прерывания HSI

Источники прерываний	Когда вырабатывается	Имя прерывания	Приоритет *
HSI FIFO Full **	INT_MASK1.6 = 1 и шестое значение загружено в FIFO. Содержимое регистра хранения не изменяется	HSI FIFO Full (INT14)	14
	IOC1.7 = 1, INT_MASK.2 = 1, INT_MASK1.6 = 0 и шестое значение загружено в FIFO. Содержимое регистра хранения не изменяется	HSI Data Available (INT02)	2
HSI FIFO 4	INT_MASK1.2 = 1 и четвёртое значение загружено в FIFO. Содержимое регистра хранения не изменяется	HSI FIFO 4 (INT10)	10
HSI.0 External	INT_MASK1.4 = 1 и сигнал HSI.0 переключается из низкого уровня в высокий и сохраняет его в течение как минимум одного периода. Это прерывание может быть сконфигурировано, чтобы действовать как независимое внешнее прерывание, для этого HSI.0 не должно быть разрешено как вход HSI	HSI.0 Pin (INT04)	4
HSI Data Available	IOC1.7 = 0, INT_MASK.2 = 1 и данные из FIFO переносятся в регистр хранения. Это прерывание показывает, что, по крайней мере, одно HSI событие произошло и готово для обработки	HSI Data Available (INT02)	2

_____

* Приоритет 15 является высшим, нулевой – низшим.

** Шестое значение в FIFO может вызывать одно из прерываний – или HSI Data Available (INT02) или HSI FIFO Full (INT14), но нельзя его сконфигурировать для обоих прерываний одновременно.

Разрешение и запрещение выводов HSI

Можно индивидуально разрешить или запретить функции HSI для каждого вывода путем установки или очистки соответствующего бита в IOC0 регистре (рисунок 15.6).

Установка соответствующего бита в регистре IOC0 связывает определитель перепада на HSI со счетчиком-делителем на восемь. Очистка бита отсоединяет вывод от HSI логики, но не от регистра HSI_STATUS (рисунок 15.6). Запрещение действия входов HSI не запрещает и не препятствует использование каждого вывода для реализации дополнительных функций.

Назначение дополнительных функций для HSI выводов

Можно запрограммировать HSI выводы для обеспечения альтернативных функций. В таблице 15.6 показаны дополнительные функции для каждого вывода HSI. Также описывается, как выбрать дополнительную функцию.

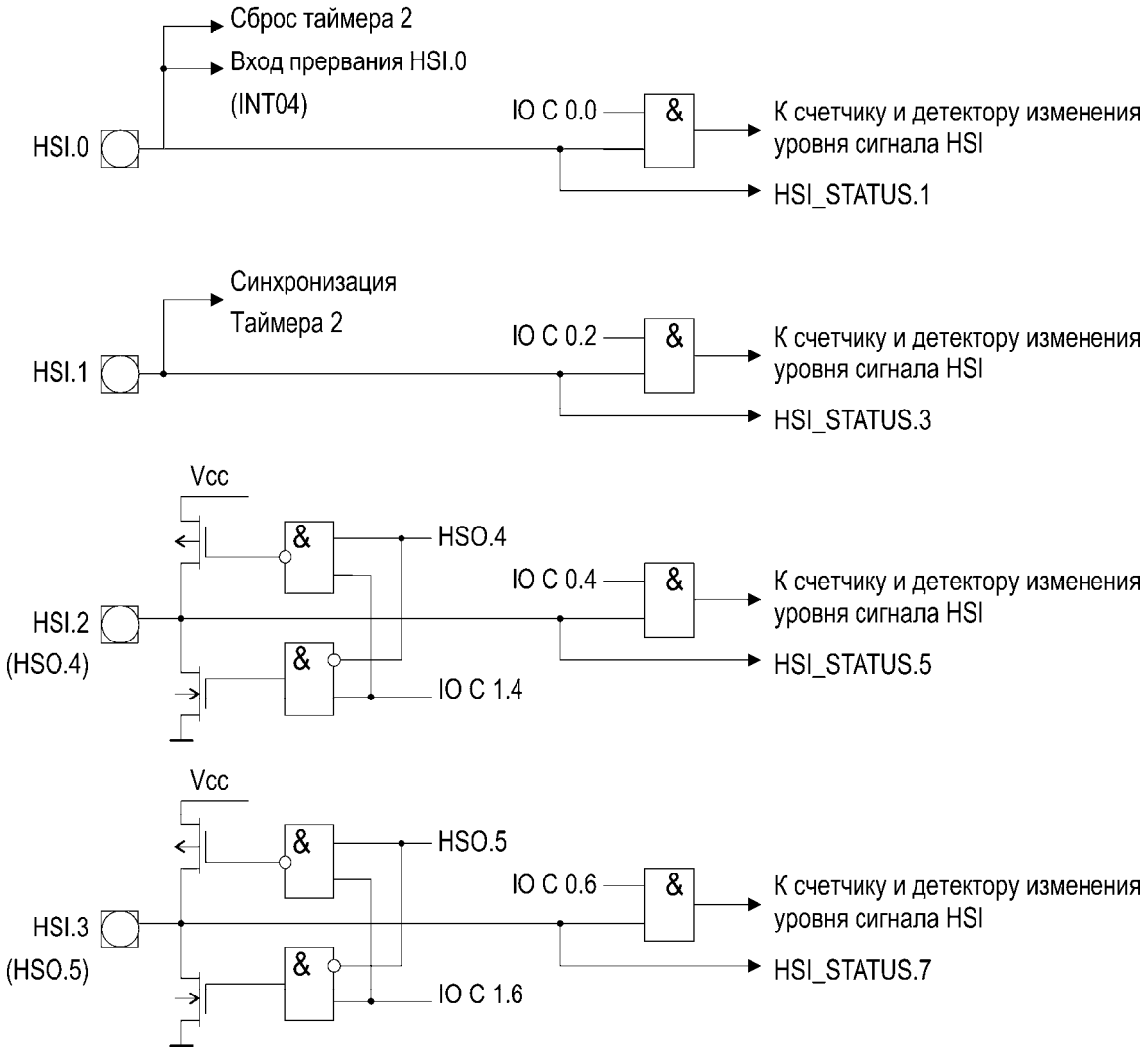


Рисунок 15.6 – Возможные входы HSI

Таблица 15.6 – Дополнительные функции для HSI выводов

Функциональное имя HSI	Дополнительные функции	Выбирается
HSI.0	Прерывание HSI.0 Pin (INT04)	Установка INT_MASK1.4 разрешает прерывание HSI.0 Pin (INT04). Рекомендуется удерживать HSI.0 в течение более одного такта, чтобы гарантировать захват сигнала
	Источник сброса таймера 2	Установка IOC0.5 выбирает HSI.0 как источник сброса таймера 2. Когда разрешено, нарастающий фронт на HSI.0 сбрасывает таймер 2
HSI.1	Источник тактов таймера 2	Установка IOC0.7 и IOC3.0 выбирает вывод HSI.1 как источник тактового сигнала таймера 2
HSI.2	HSO.4	Установка IOC1.4 разрешает действие выхода HSO.4. Примечание – HSI и HSO действия могут быть активны в одно и то же время
HSI.3	HSO.5	Установка IOC1.6 разрешает действие выхода HSO.5. Примечательно то, что HSI и HSO действия могут быть активны в одно и то же время

15.3 Модуль высокоскоростного вывода HSO

Модуль высокоскоростного вывода HSO вызывает события в определенное время с использованием таймера 1 или таймера 2 как базовые. Эти программируемые события включают: запуск АЦП, перезапуск таймера 2, генерацию до четырех программных временных задержек и установку или очистку одного или более HSO выходов (HSO.0 – HSO.5). HSO модуль сохраняет до восьми ожидаемых событий и соответствующее им время в блоке памяти CONTENT_ADDRESSABLE MEMORY (CAM). На рисунке 15.7 показана блок-схема модуля HSO.

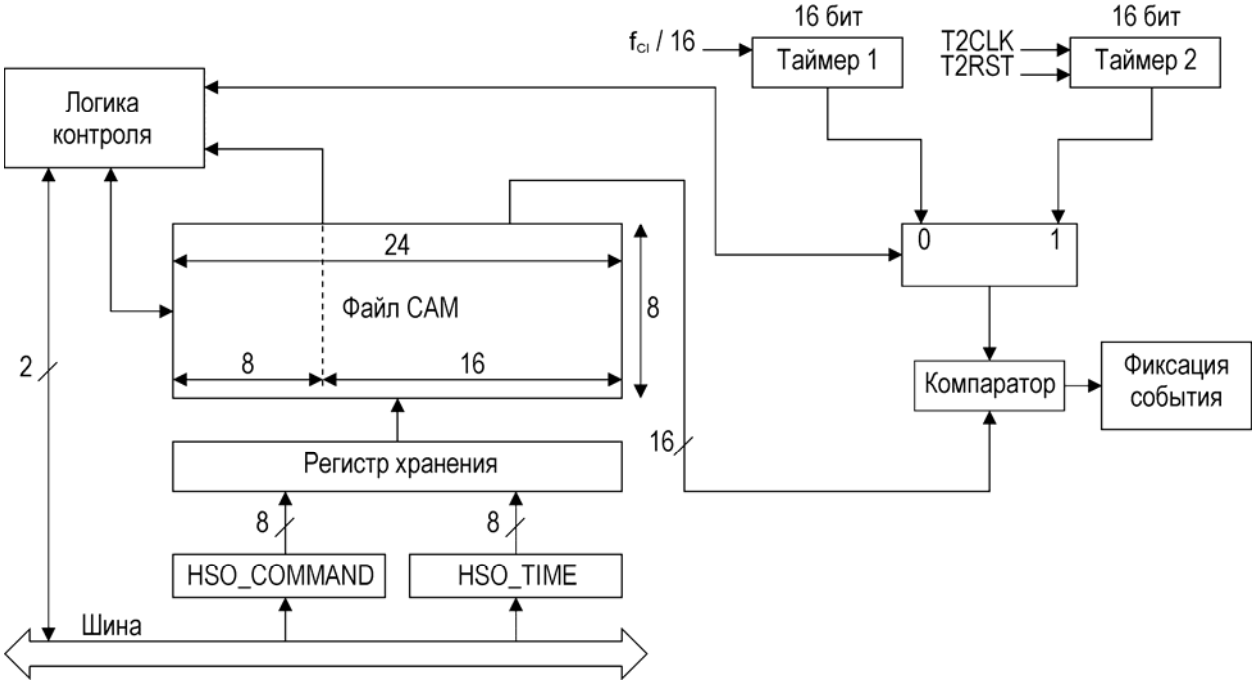


Рисунок 15.7 – Блок-схема модуля HSO

15.3.1 Функционирование HSO

CAM – основной компонент модуля HSO. Эта область памяти сохраняет до восьми команд. Каждая запись в CAM файле содержит 24 бита. 16 бит загружаются из HSO_Time регистра, чтобы определить время действия, и 8 бит загружаются из HSO_COMMAND регистра, чтобы определить тип действия: требуется ли прерывание и какой из таймеров, первый или второй, является базовым.

Примечание – Когда таймер 2 используется в модуле HSO как базовый, рекомендуется программировать его на работу в нормальном режиме (а не в режиме fast increment) и счет только в одном направлении. Подробнее – в пункте 15.1.6.

HSO сравнивает все восемь записей в CAM файле со значением таймера до фиксации события. Сравнение всех записей в CAM файле происходит одновременно, поэтому требуется только один такт, чтобы сравнить все записи для полной проверки CAM файла. HSO фиксирует событие, когда значение базового таймера соответствует запрограммированному значению времени.

HSO может вызвать 15 различных событий, которые классифицируются как внешние или внутренние. Внешние – события, которые проявляются на одном или более выходах HSO; внутренние – события, которые устанавливают программные таймеры, сбрасывают таймер 2 или запускают АЦП. Все эти события устанавливают флаги, которые могут вызывать определенные прерывания. Внешние события вырабатывают прерывание High Speed Output (INT03); внутренние события вырабатывают прерывание Software Timer (INT05).

Четыре программных таймера имеют возможность вырабатывать прерывания в заранее определенное время; в общем случае они используются для вызова программ обслуживания прерывания, которые должны повторяться с определенными интервалами. В определенное время HSO устанавливает флаг HSO в IOS1 регистре; если прерывания разрешены, это также вызовет прерывание Software Timer. Если больше чем один программный таймер срабатывает в пределах одного временного фрагмента, устанавливается несколько битов состояний. Подпрограмма обслуживания прерывания может проверить регистр IOS1, чтобы определить, какой программный таймер (или таймеры) вызвали прерывание.

15.3.2 Программирование HSO модуля

В таблице 15.7 приведены регистры, которые влияют на работу и функции модуля HSO.

Таблица 15.7 – Регистры управления и состояния модуля HSO

Мнемоника регистра	Имя регистра	Описание
HSO_COMMAND	Регистр команд HSO	Определяет, что за событие или события произойдут в HSO в определенное время
HSO_TIME	Время HSO	Определяет время, в которое HSO команда будет выполнена
INT_MASK	Маска прерывания	Разрешает или запрещает HSO прерывания
IOC1	Регистр 1 управления входом/выходом	Разрешает или запрещает HSO.4 и HSO.5 как выходы
IOC2	Регистр 2 управления входом/выходом	Разрешает и запрещает команды просмотра HSO области CAM, может также очистить HSO CAM
IOS0	Регистр 0 состояния входа/выхода	Показывает текущее состояние HSO выводов, фиксирующего регистра и CAM файла; запись в биты может устанавливать или очищать соответствующие HSO выводы
IOS1	Регистр 1 состояния входа/выхода	Содержит флаги, которые показывают, какие внутренние события вызваны прерыванием
IOS2	Регистр 2 состояния входа/выхода	Содержит флаги, которые показывают, какие внешние HSO события имели место

Программирование HSO событий

Содержимое регистров HSO_TIME и HSO_COMMAND вместе описывают каждое HSO событие. HSO_TIME регистр определяет время выполнения команды HSO. Биты 0 – 3 регистра HSO_COMMAND (рисунок 15.8) содержат команду, которая определяет, какое событие или события HSO будут вызваны в определенное время (рисунок 15.9). В таблице 15.8 описаны коды HSO команд. HSO_COMMAND регистр также определяет, будет ли событие вызывать прерывание, выполняет ли команда установку или очистку соответствующего вывода (выводов), выбирает базовый таймер для HSO-команд и управляет сохранением команд в CAM или удалением после выполнения (таблица 15.9).

Таблица 15.8 – Кодировка CMD_TAG

Код (hex)	Мнемоника команд	Описание
00	HSO0	Включить High-Speed Output 0
01	HSO1	Включить High-Speed Output 1
02	HSO2	Включить High-Speed Output 2
03	HSO3	Включить High-Speed Output 3
04	HSO4	Включить High-Speed Output 4
05	HSO5	Включить High-Speed Output 5
06	HSO01*	Включить High-Speed Outputs 0 и 1
07	HSO23*	Включить High-Speed Outputs 2 и 3
08	SWT0	Программирование программного таймера 0
09	SWT1	Программирование программного таймера 1
0A	SWT2	Программирование программного таймера 2
0B	SWT3	Программирование программного таймера 3
0C	HSOALL*	Включение High-Speed Outputs 0, 1, 2, 3, 4, 5
0D		Резерв; не используется
0E	T2RST	Сброс Timer2
0F	A-D	Запуск АЦП

_____

* В этой конфигурации два или более выводов устанавливаются или очищаются одновременно.

HSO_COMMAND
регистр управления HSO

06H значение после Сброса: XXXXH

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

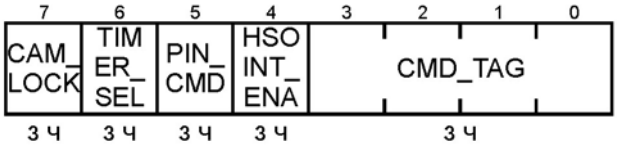


Рисунок 15.8 – Формат регистра HSO_COMMAND

Таблица 15.9 – Описание битов регистра HSO_COMMAND

Мнемоника бита	Бит	Описание
1	2	3
CMD_TAG	0 – 3	Команды HSO. Определяют какие события или событие будут (таблица 15.8)
HSOINT_ENA	4	Разрешение/ запрещение HSO прерывания. Определяет, вырабатывает ли HSO событие прерывание: «1» – вырабатывает прерывание; «0» – не вырабатывает прерывание. Когда этот бит установлен, запрограммированное на вывод HSO событие вырабатывает High-Speed Output прерывание (INT03, 2006H), а внутреннее событие вырабатывает Software Timer прерывание (INT05, 200AH). Когда прерывание произошло, биты состояния HSO события в IOS1 и IOS2 регистрах должны быть проанализированы, чтобы определить, какое событие вызвало прерывание

Окончание таблицы 15.9

1	2	3
PIN_CMD	5	Установка/очистка выбранного вывода HSO. Устанавливается или сбрасывается определенный вывод или выводы: «1» – устанавливает вывод(ы); «0» – сбрасывает вывод(ы)
TIMER_SEL	6	Выбор Timer1 / Timer2. Выбирает опорный таймер для HSO команд: «1» – выбирается таймер 2; «0» – выбирается таймер 1
CAM_LOCK	7	Блокировка команды в CAM. Когда IOS2.6 установлен (фиксация команды разрешена), этот бит управляет: должна ли HSO-команда зафиксироваться в CAM или очиститься после выполнения: «1» – фиксация команды в CAM; «0» – стереть команду из CAM после выполнения. Запись «1» в IOS2.7 очищает всё содержимое (фиксированное и нет) из CAM, как это делает сброс кристалла

Примечание – Перед программированием HSO_COMMAND и HSO_TIME регистров необходимо прочитать или IOS0.7, или IOS0.6, чтобы убедиться, что регистр захвата HSO пустой. Если IOS0.7 очищен, то регистр захвата пуст. Если IOS0.6 очищен, то регистр захвата пуст и, по крайней мере, один CAM регистр пуст. Запись в HSO_TIME, когда регистр захвата не пустой, вызывает потерю предыдущего значения фиксирующего регистра.

Для того, чтобы ввести команду в CAM-файл, первым записывается HSO_COMMAND регистр, чтобы описать событие, и затем записывается HSO_TIME регистр, чтобы определить относительное время, в которое событие произойдет. Запись в HSO_TIME регистры загружают HSO регистр захвата. Команда сохраняется в фиксирующем регистре до тех пор, пока невозможна запись в CAM файл, при освобождении команда переписывается в CAM-файл на следующий такт.

Примечание – Команда в фиксирующем регистре не будет выполняться, даже если ее время подошло. Команды для их выполнения должны находиться в CAM-файле.

Значение в HSO_TIME может быть или определенным значением таймера, или значением смещения относительно текущего значения таймера. Необходимо использовать стандартные команды загрузки, чтобы запрограммировать определенное время, иначе используется трех-операндная команда ADD, чтобы определить значение, которое является смещением относительно текущего значения таймера.

```
LDB     HSO_COMMAND, #what_to_do            ; Loads command
ADD     HSO_TIME, TIMER1, #when_to_do_it    ; ADD_3op
```

Для обеспечения правильной синхронизации минимальное время, которое может быть загружено в таймер, должно быть равно current_time_value+2. Меньшее значение может быть причиной того, что событие осуществится на 65 536 тактов позже, чем требуется. Это ограничение касается обоих таймеров: таймера 1 и таймера 2.

Следует внимательно записывать в регистры HSO_COMMAND и HSO_TIME. Прерывание может осуществиться после записи команд в HSO_COMMAND и перед записью значения времени HSO_TIME. Если программа обработки прерывания также записывает в эти регистры, команда основной программы в CAM-файле перекрывается, поэтому команда основной программы никогда не выполнится. Запрещение прерываний перед записью в HSO-модуль решает эту проблему (смотри раздел 7).

Разрешение прерываний от HSO

Установка бита HSOINT_ENA (HSO_COMMAND.4) позволяет HSO событию генерировать прерывание. HSO модуль может генерировать два различных прерывания: прерывание High-Speed Output (INT03) и прерывание Software Timer (INT05).

Внешние события вырабатывают прерывание High-Speed Output, если оно разрешено (INT_MASK.3 установлен). Биты состояния IOS2.0 – IOS2.5 связаны с прерыванием High-Speed Output. Биты состояния IOS1.0 – IOS1.3 используются с прерыванием Software Timer для команд программного таймера, а IOS2.6 – IOS2.7 для сброса таймера 2 и старта АЦ преобразования, соответственно.

Когда прерывание произошло, необходимо считать статусные биты в IOS1 и IOS2 регистрах, чтобы определить, какое событие вызвало прерывание. Чтение IOS1 регистра очищает биты 0 – 5; чтение IOS2 регистра очищает все биты. По этой причине рекомендуется копировать содержимое регистров в теневые регистры и затем выполнять команды тестирования битов, такие как JBC или JBS в теневых регистрах.

Сохранение записей в CAM

LOCK_ENA бит в IOC2 регистре (IOC2.6) разрешает или запрещает фиксацию записи с командой. Когда этот бит установлен, CAM_LOCK бит (HSO_COMMAND.7) управляет либо сохранением команд в CAM, либо их удалением после выполнения. Когда CAM_LOCK установлен, команда остается в CAM после выполнения. Это дает возможность просто генерировать периодические события или сигналы. Обнулене CAM_LOCK вызывает удаление HSO команды из CAM после выполнения.

Фиксация команд обеспечивает возможность программирования периодических или повторяющихся событий. Одним из наиболее полезных применений является использование сохранения команд для генерации повторяющихся импульсов с задаваемой длительностью (PWM) с минимумом программных затрат (пункт 15.3.3). Сохранение команды АЦ преобразования вызывает многократное выполнение АЦ преобразования. Когда таймер 2 используется как базовый в HSO, зафиксированная команда T2RST может вырабатывать периодические события; события со значением HSO_TIME меньшим, чем значение сброса таймера 2, повторяются с каждым новым сбросом таймера 2. Сохранение команд программного таймера может организовать повторяющийся вызов программных задач; программы обслуживания прерывания могут выполнять задачи без использования других команд программного таймера HSO.

Удаление команды из CAM

Три события могут удалить команду из CAM:если значение базового таймера соответствует запрограммированному времени и не запрограммировано сохранение события, сброс устройства HSO, установлен CAM_CLR бит (IOC2.7).

Чтобы удалить сохраненные события, необходимо установить CAM_CLR бит (IOC2.7) или сбросить устройство. Любое из этих действий очистит весь CAM-файл.

Отмена события

Можно отменить внешнее событие, записав противоположное событие с тем же временем в CAM. Например, если команда устанавливает HSO.1, когда TIMER1 = 1234, следует записать команду, которая очищает HSO.1; если TIMER1 = 1234, тогда HSO.1 не изменится. Однако обе команды остаются в CAM до тех пор, пока или TIMER1 не будет равен 1234 (если сохранение незапрограммировано), или устройство не будет сброшено, или не будет установлен CAM_CLR бит (IOC2.7).

Можно отменить внутреннее событие только установкой CAM_CLR бита (IOC2.7) или сбросом устройства.

Разрешение выводов HSO.4 и HSO.5

HSO.4 и HSO.5 выходы мультиплексируются с HSI.2 и HSI.3 входами, соответственно. Эти выводы могут быть разрешены для той и другой функций (рисунок 15.6). Установка HSO4_ENA бита (IOC1.4) разрешает HSO.4 как выход; установка HSO5_ENA бита (IOC1.6) разрешает HSO.5.

Использование выводов HSO.0 – HSO.5 как выходов

Выходы HSO могут дать шесть дополнительных выходных контактов, если не используются функции HSO. Чтобы использовать выходы HSO как стандартные выходы, необходимо разрешить их использование как выходов, и затем установить или очистить их записью в IOS0 регистр через HWindow 15.

Использование модуля HSO как широтно-импульсного модулятора (PWM)

И HSO модуль, и PWM модуль могут генерировать последовательность прямоугольных импульсов, в которой изменяются период и скважность импульсов. Используя соответствующие внешние компоненты, можно реализовать высокоточный восьмиразрядный цифро-аналоговый преобразователь с использованием либо HSO, либо PWM выходов (раздел 18).

HSO модуль может генерировать либо одиночную PWM последовательность, либо повторяющуюся PWM последовательность. Одиночная последовательность требует программирования периода и времени сброса (clear time) сигнала и двух HSO команд: установить активное состояние вывода и сбросить его. Для управления может быть использован таймер 1 или таймер 2.

Генерация повторяющейся последовательности производится аналогичным способом, для управления используется таймер 2. Для генерации повторяющейся последовательности требуется запрограммировать период для таймера 2, время установки (set time) (PWMx_ST) и время сброса (PWMx_CT) для каждого выходного контакта и нескольких HSO команд: одну для сброса таймера 2, несколько для установки выводов и несколько для их сброса.

В приведенном ниже примере показан способ генерации нескольких PWM сигналов (ШИМ последовательностей) с использованием HSO.0, HSO.1 и HSO.2 как PWM выходов и таймера 2 как базового. На рисунке 15.9 показаны три результирующие PWM последовательности.

```
SET_0:    HSO_COMMAND,          #11100000B   ; set HSO.0 when
LDB       HSO_TIME, #PWM0_ST R0 R0           ; timer2 = pwm0_st
LD        HSO_COMMAND,          #11100001B   ; wait 4 state times
SKIP      HSO_TIME, #PWM1_ST R0 R0           ; wait 4 state times
SKIP      HSO_COMMAND,          #11100010B
SET_1:    HSO_TIME, #PWM2_ST    ZERO_REG     ; set HSO.1 when
LDB       ZERO_REG                           ; timer2 = pwm1_st
LD        HSO_COMMAND,          #11000000B   ; wait 4 state times
SKIP      HSO_TIME, #PWM0_ST    ZERO_REG     ; wait 4 state times
SKIP      ZERO_REG
SET_2:    HSO_COMMAND,          #11000001B   ; set HSO.2 when
LDB       HSO_TIME, #PWM1_ST    ZERO_REG     ; timer2 = pwm2_st
LD        ZERO_REG                           ; wait 4 state times
SKIP                                         ; wait 4 state times
SKIP
CLEAR_0:                                     ; clear HSO.0 when
LDB                                          ; timer2 = pwm0_ct
LD                                           ; wait 4 state times
SKIP                                         ; wait 4 state times
SKIP
CLEAR_1:                                     ; clear HSO.1 when
LDB                                          ; timer2 = pwm1_ct
LD                                           ; wait 4 state times
SKIP                                         ; wait 4 state times
SKIP
```

15.3.3 Синхронизация выхода HSO

HSO выводы синхронизируются базовым таймером. Все внешние выводы HSO изменяются при достижении таймером заданного значения сразу после того, как таймер увеличился на единицу. При внутренней синхронизации таймер изменяет значение каждый восьмой такт во время фазы 1. При внешнем HSO вывод будет изменяться сразу после спада импульса CLKOUT и будет стабилен при его нарастающем фронте. Информация от HSO может быть зафиксирована при нарастающем фронте CLKOUT. Внутреннее HSO событие происходит, когда инкрементируется базовый таймер.

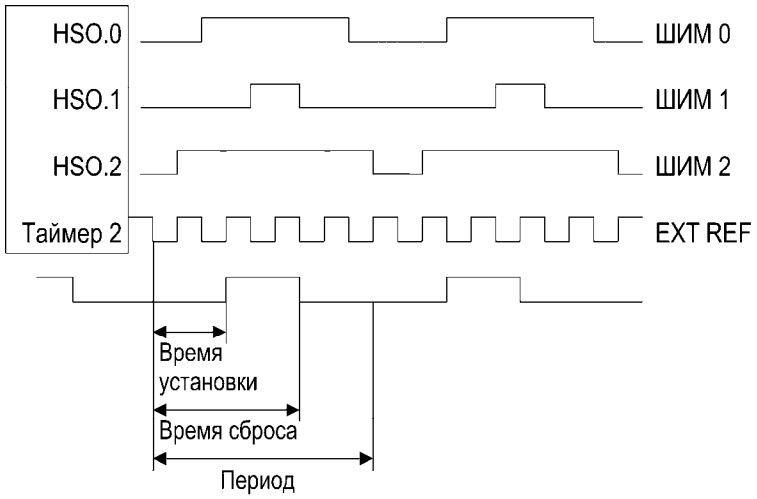


Рисунок 15.9 – Пример PWM последовательностей

16 Аналого-цифровой преобразователь и блок цифровых компараторов

Аналого-цифровой преобразователь, реализованный в контроллере, построен на основе сигма-дельта (Σ−Δ) преобразователя, реализующего алгоритм сверхвыборки, когда норма осуществления выборки во много раз превышает требуемую частоту (блок-схема АЦП на рисунке 16.1). Такая реализация позволяет значительно уменьшить шумы и гарантирует стабильно высокие динамические параметры преобразователя. Схемы цифровых компараторов осуществляют контроль результатов преобразования и генерируют прерывания при их выходе за пределы заданных значений (рисунок 16.2).

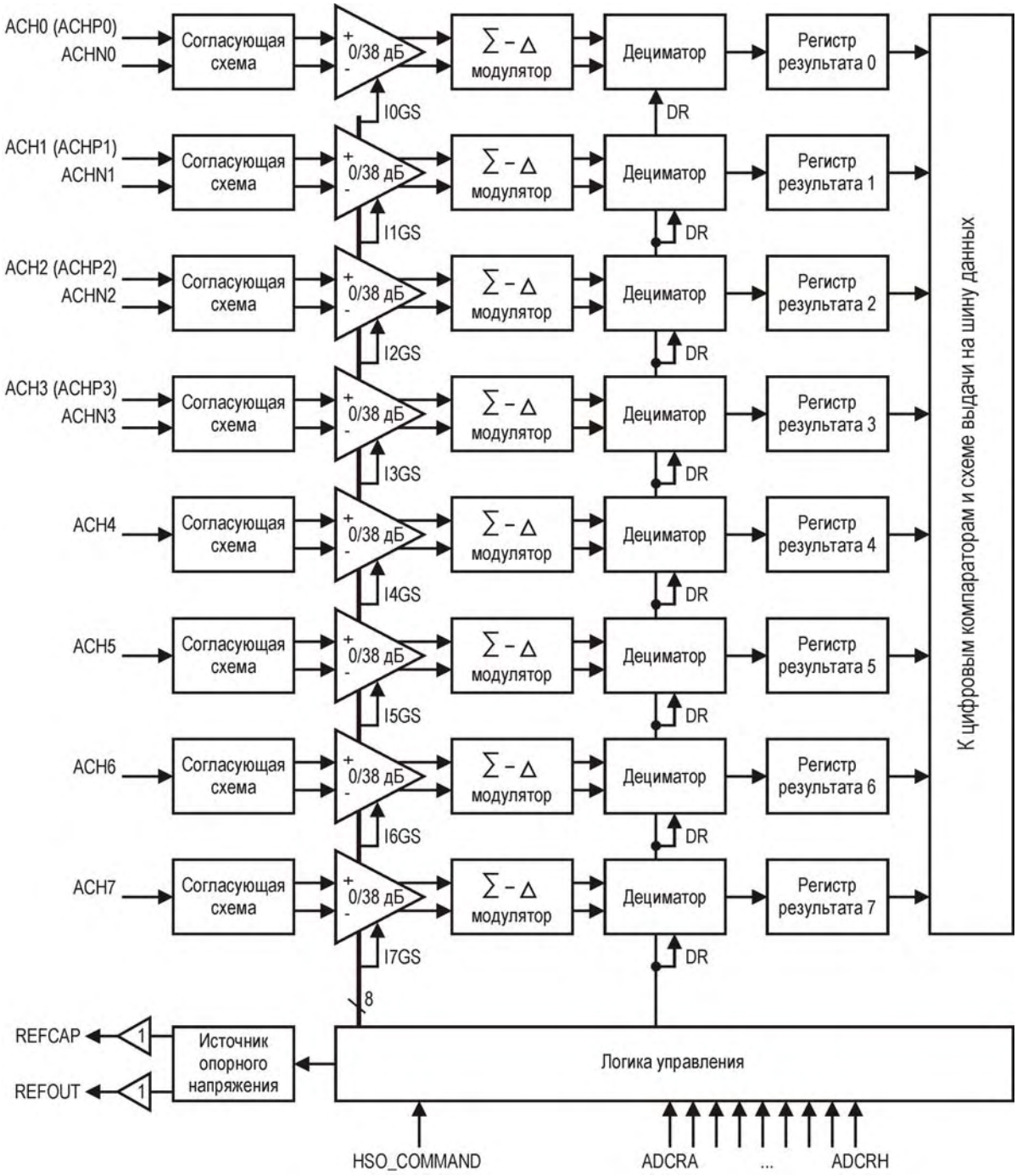


Рисунок 16.1 – Блок-схема аналого-цифрового преобразователя

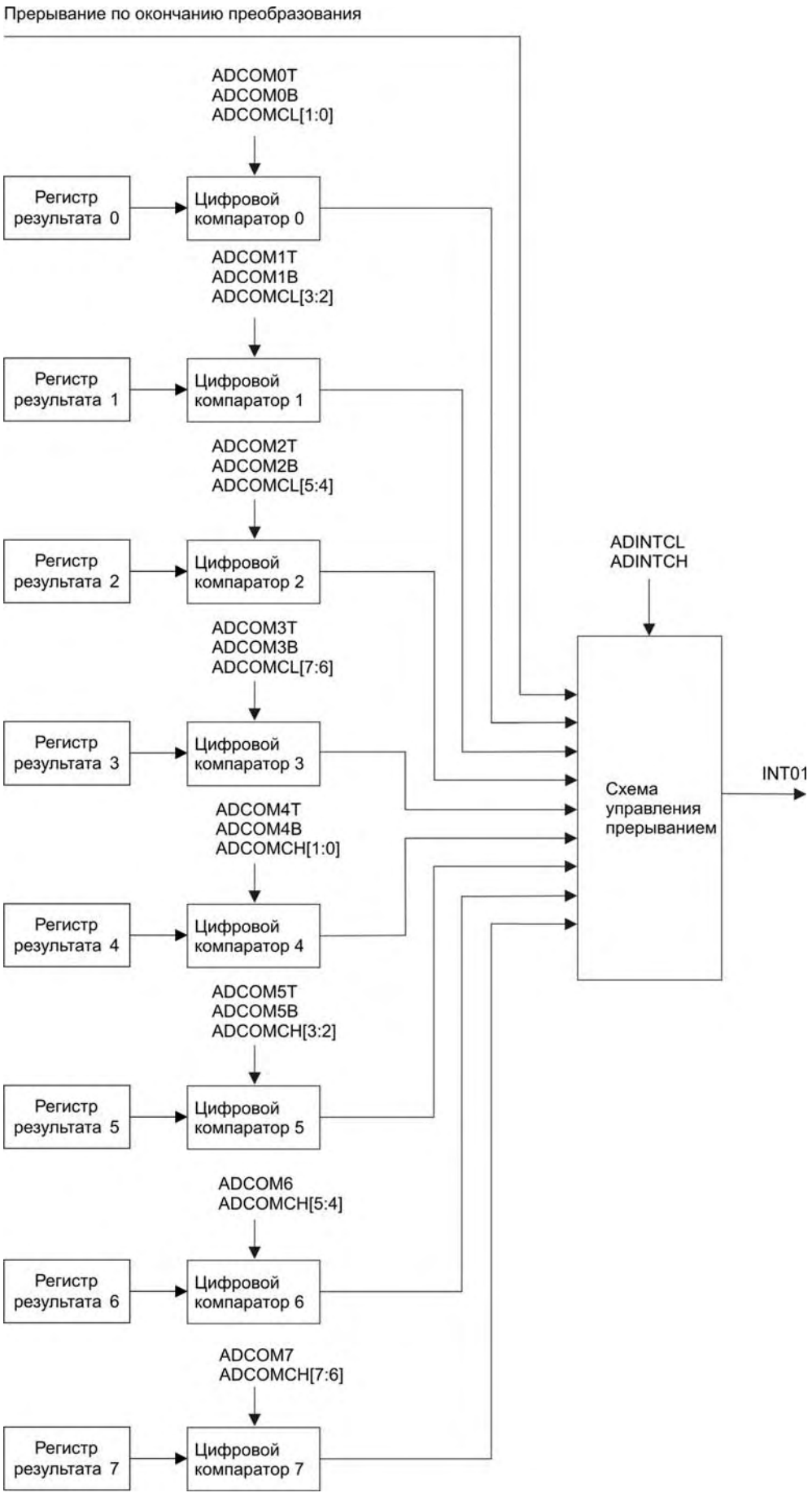


Рисунок 16.2 – Блок-схема цифровых компараторов

16.1 Обзор функций АЦП

Модуль АЦП (блок-схема на рисунке 16.1) преобразует входное напряжение сигнала на каждом из восьми каналов в 16-битное цифровое значение. Основными частями модуля АЦП являются:

- восемь идентичных схем преобразования, каждая из которых включает согласующую схему, усилитель входного сигнала, сигма-дельта модулятор, дециматор и 16-разрядный регистр результата преобразования;

- восемь аналоговых входов АЦП: четыре первых входа по умолчанию подключены к позитивным дифференциальным (дифф.) входам U_{INP} каналов 0 – 3, четыре вывода ACHNx в режиме дифференциального входа через нормально замкнутые цепи коммутатора K_1 (коммутаторы K_0, K_2, K_3 не показаны) подключены к инверсным входам U_{INN} каналов 0 – 3 АЦП (при «0» в разрядах 4 – 7 регистра ADCRG). При дифференциальном включении максимальное количество одновременно работающих каналов АЦП равно четырем (восемь выводов – по два входа на канал). Инверсия входов U_{INN} в дифференциальных усилителях каналов АЦП установлена занесением «1» в разряды регистра ADCRH. Упрощенная блок-схема переключения входов в каналах АЦП в режим одиночного (однопроводного) входа приведена на рисунке 16.3.

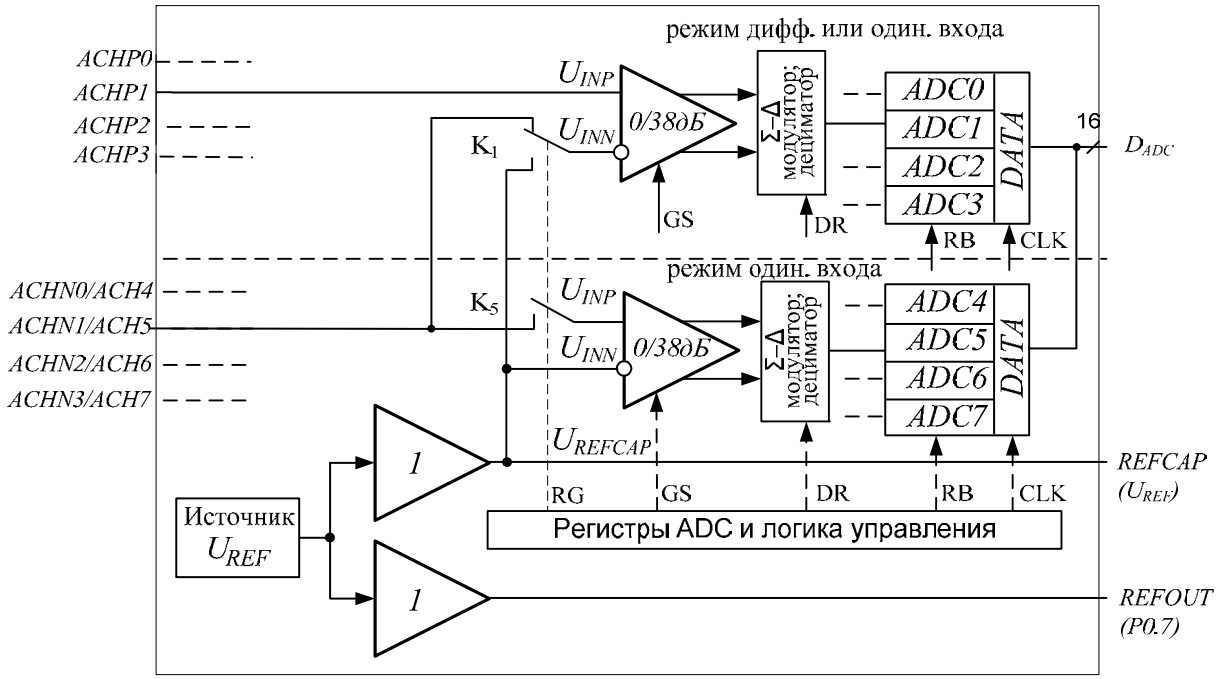


Рисунок 16.3 – Упрощенная блок-схема переключения режимов в каналах АЦП

Для переключения каналов АЦП в режим одиночного входа в соответствующие разряды регистра ADCRG заносится «1», при этом коммутаторы $K_1 \ldots K_5$ переключаются в нижнее положение (цепи коммутаторов шести каналов не показаны). При этом в каналах 0 – 3 АЦП входы U_{INN} дифференциального усилителя переключаются на шину опорного напряжения U_{REF}, а на входы U_{INP} дифференциальных усилителей каналов 4 – 7 подключаются выводы ACH4 - ACH7. Каналы 4 – 7 АЦП используются только в режиме одиночного входа;

- регистры ADCRA, ADCRB, ADCRC, ADCRD, ADCRE, ADCRF, ADCRH управляют временем преобразования, коэффициентом усиления входных сигналов, включением преобразований, запуском преобразования выбранных каналов и другими функциями АЦП. Механизм управления и детальное описание возможностей АЦП и принципов программирования выше указанных режимов приведены в подразделе 16.2;

- внутренний источник опорного напряжения АЦП микросхемы выполнен с использованием напряжения запрещенной зоны полупроводника. Опорное напряжение U_{REF} через буфер подается к каналам АЦП и на вывод REFCAP, к которому подключается конденсатор емкостью не менее 0,1 мкФ с минимальной длиной проводников;

- на вывод REFCAP для повышения стабильности опорного источника может подаваться внешний источник напряжения U_{REF} в диапазоне от 1,125 до 1,6 В с низким внутренним сопротивлением и выходным током не менее 2 мА. Точность поддержания входного опорного напряжения, заданного в выше указанном диапазоне, в процессе преобразования АЦП не менее 0,5 U_{LSB};

- при включении входов АЦП со смещением по переменному току по схемам на рисунках 16.14 и 16.16 предусмотрен буферизированный вывод опорного напряжения REFOUT установкой бита RU (CRC: 6) регистра управления. При внешнем U_{REF}, для схем согласования входных уровней АЦП потребителям, следует использовать этот же источник, так как выходной буфер REFOUT не имеет связи с выводом REFCAP. Блок-схема организации U_{REF} в каналах АЦП показана на рисунке 16.3;

- запуск преобразования АЦП осуществляется с помощью команды HSO при тактировании сигналом CLKC периферийных блоков микроконтроллера. Преобразуемый сигнал поступает на согласующую схему канала, обеспечивающую ограничение уровня, а также выбор полярности сигнала. С ограничителя сигнал поступает на усилитель с программируемым дискретным коэффициентом усиления в диапазоне от 0 до 38 дБ, далее – на сигма-дельта модулятор и дециматор. Усилитель с программируемым коэффициентом усиления каждого канала представляет собой схему на переключаемых конденсаторах и является частью сигма-дельта модулятора;

- цифровой фильтр выполняет две важные функции. Во-первых, это перемещение шума квантования за пределы требуемой полосы частот, который сформирован аналоговым модулятором, и, во-вторых, это прореживание потока битов высокой частоты до более низкой частоты 15-битных слов;

- сглаживающий прореживающий фильтр – это цифровой фильтр с характеристикой в форме sinc3, который понижает частоту дискретизации на значение, заданное в регистре управления. Делитель прореживания позволяет пользователю гибко подстроить значение выборки АЦП к требуемой программе контроллера;

- АЦП выполняет несколько функций, таких как: временная дискретизация, квантование по уровню, кодирование, аналого-цифровые преобразования. Передаточная характеристика преобразования напряжения ΔU_{IN} между входами канала АЦП в режиме дифференциального и одиночного входа приведена на рисунке 16.4 сплошной линией. В дифференциальном режиме передаточную характеристику АЦП также можно показать двумя прямыми относительно опорного напряжения U_{REF} по входам U_{INP}" и U_{INN}" (см. рисунок 16.4 пунктирные линии).

16.1.1 Соотношение режимных параметров в каналах АЦП

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

В соответствии с рисунком 16.4 начальная и конечная точки преобразования входного напряжения АЦП U_{FS-} и U_{FS+} определяются экспериментально с суммарной погрешностью не более одного U_{LSB} (в идеальном случае) из соотношений (16.1) и (16.2), или могут быть рассчитаны из уравнения передаточной характеристики АЦП (16.7), при известном коэффициенте преобразования, смотри пример 1.

$$U_{FS-} = U_{FS-,1} - 0{,}5 \cdot U_{LSB}, \qquad (16.1)$$

где $U_{FS-,1}$ – напряжение первого межкодового перехода к коду 8001h;

U_{LSB} – напряжение единицы младшего разряда. В данном случае шаг квантования определяется выражением (16.4).

Входное напряжение в конечной точке АЦП при переходе к коду 7FFFh равно

$$U_{FS+} = U_{FS(nom)+} - 0{,}5 \cdot U_{LSB}. \qquad (16.2)$$

Область значений входного напряжения АЦП, ограниченная точками (U_{FS-}–U_{FS+}), называется практическим диапазоном преобразования АЦП и обозначается U_{FSR}.

$$U_{FSR} = (U_{FS+} - U_{FS-}) - U_{LSB} = 2^N \cdot U_{LSB} - U_{LSB} = U_{LSB}(2^N - 1), \qquad (16.3)$$

где N – число разрядов.

Шаг квантования (единица младшего разряда) определяется отношением

$$U_{LSB} = \frac{U_{FSR}}{2^N - 1} = \frac{U_{FSR(nom)}}{2^N}, \qquad (16.4)$$

где $U_{FSR(nom)}$ – номинальный диапазон преобразования АЦП, определяемый выражением

$$U_{FSR(nom)} = U_{FSR} + U_{LSB} = 2^N \cdot U_{LSB}. \qquad (16.5)$$

Единица старшего знакового разряда определяется соотношением

$$U_{MSB} = \frac{U_{FSR\ (nom)}}{2}. \qquad (16.6)$$

Передаточная характеристика преобразования напряжения ΔU_{IN} канала АЦП в режиме одиночного или дифференциального входа (показана сплошной линией $U_{FS-} - U_{FR+}$ на рисунке 16.4) описывается уравнением

$$D_{ADC} = (U_{INP} - U_{INN}) \cdot K_G, \qquad (16.7)$$

где U_{INP} – входное напряжение на позитивном входе АЦП;
U_{INN} – входное напряжение на негативном входе АЦП;
D_{ADC} – выходной цифровой код АЦП;
K_G – коэффициент преобразования АЦП.

.

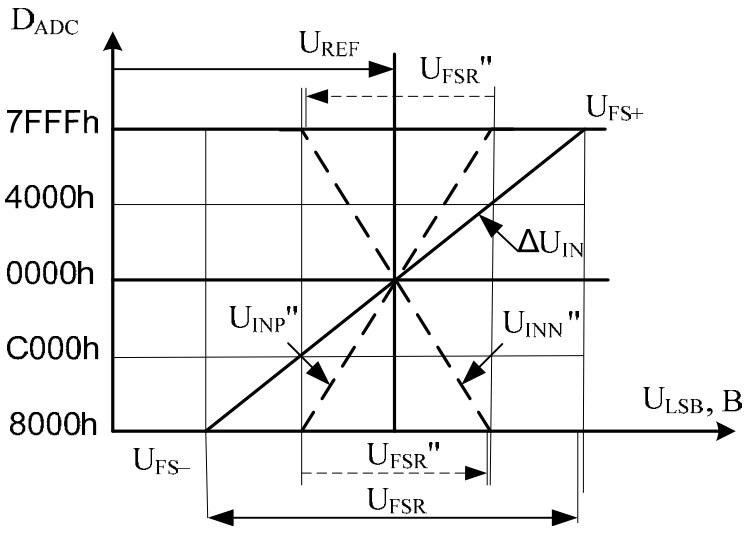


Рисунок 16.4 – Передаточная характеристика каналов АЦП

В соответствии с (16.7) и с учетом (16.5) коэффициент преобразования канала АЦП равен

$$K_G = \frac{K_{PGA} \cdot D_{ADC}}{U_{FSR\ (nom)}} = \frac{K_{PGA}}{U_{LSB}}, \qquad (16.8)$$

где K_{PGA} – коэффициент усиления АЦП, при PGA = 0 дБ равен единице, при этом коэффициент преобразования АЦП равен

$$K_G = \frac{1}{U_{LSB}}. \qquad (16.9)$$

Для передаточной характеристики по входам относительно U_{REF} (рисунок 16.4, пунктирные линии) диапазон преобразования и шаг квантования делятся пополам. При этом коэффициент преобразования по каждому из входов относительно U_{REF}, с учетом (16.4) и (16.9), равен

$$K_G'' = \pm \frac{1}{0{,}5U_{LSB}}. \qquad (16.10)$$

Выше приведенные соотношения режимных параметров позволяют определить и рассчитать шаг квантования, коэффициент преобразования и допустимые входные напряжения АЦП. Ниже приведены примеры 1 – 5 расчета диапазона входных напряжений и выходного кода АЦП с комментарием по применению.

П р и м е р 1 – Из уравнения (16.7) рассчитаем допустимый диапазон напряжений преобразования между входами дифференциального усилителя 16-разрядного АЦП в режиме дифференциального или одиночного входа ($U_{INN} = U_{REF}$) при PGA = 0 дБ. Используя типовую зависимость, рисунок 16.4а, при номинальном U_{REF} =1,25 В и температуре 25 °С, определяем U_{FSR} = 1,5 В. U_{FSR} также можно определить, используя выше приведенные соотношения (16.1) – (16.5). При U_{FSR} = 1,5 В (из (16.9) числовой коэффициент преобразования K_G равен 43690,8. По определению, цифровой код АЦП 16-разрядного АЦП равен $D_{ADC} = 2^{16} = 65536$. Относительно опорного напряжения код АЦП одной полярности составит $D_{ADC} = \pm 2^{15} = \pm 32\,768$. Максимальная разница входного напряжения $\Delta U_{IN} = (U_{FS+} - U_{FS-}) = D_{ADC} / K_G = \pm 32\,768 / 43690{,}8 = \pm 0{,}75$ В (напряжение на входах относительно U_{REF}). Напряжение между входами АЦП ΔU_{IN} должно быть не более диапазона преобразования U_{FSR} = 1,5 В, см. рисунки 16.4, 16.4б.

В то же время в дифференциальном режиме входные напряжения ($U_{INP}'' - U_{INN}''$) относительно опорного напряжения при K_G'' = ±87381 (16.10) из уравнения (16.7), при коде $D_{ADC} = 2^{15} = 32\,768$ не более: $\Delta U_{IN}'' = (U_{INP}'' - U_{INN}'') = D_{ADC}/K_G'' = 32\,768 \cdot (\pm 87381) =$ $= \pm 0{,}375$ В (**(**U_{FSR}" не более 0,75 В).

П р и м е р 2 – Из уравнения (16.7) при известном коэффициенте преобразования K_G (см. (16.9) можно определить выходной код АЦП. Для входного напряжения ΔU_{IN} = 0,375 В = ¼U_{FSR}: выходной код $D_{ADC} = \Delta U_{IN} \cdot K_G = 0{,}375 \cdot 43690{,}8 = 16\,384 = 4000h$ (см. рисунок 16.4).

П р и м е р 3 – Расчет входных напряжений преобразования при PGA = 38 дБ (дополнение к примеру 1). При PGA = 38 дБ коэффициент усиления АЦП равен $K_{PGA/38дБ} = 80$. Коэффициет преобразования АЦП в соответствии с (16.8) так же увеличится в 80 раз и составит: $K_{G/38дБ} = 43690{,}8 \cdot 80 = 3495264$. Диапазон преобразования на входах АЦП соответственно уменьшится в 80 раз и составит 18,75 мВ при U_{LSB} = 0,5722 мкВ.

При расчетах также возможно использование уравнения АЦП (16.7) с полученным выше числовым коэффициентом преобразования $K_{G/38дБ}$. При выходном цифровой коде $D_{ADC} = \pm 2^{15} = \pm 32\,768$. Границы диапазона преобразуемых напряжений на входах АЦП в данном примере равны: $\Delta U_{IN} = (U_{FS+} - U_{FS-}) = D_{ADC}/K_{G/38дБ} = \pm 32\,768/3495264 = \pm 9{,}375$ мВ.

Из примера следует, что напряжение на входах АЦП, при PGA = 38 дБ, не более ±9,375 мВ относительно U_{REF} и между входами АЦП не более $U_{FSR/38дБ}$ = 18,75 мВ.

В то же время в дифференциальном режиме входные напряжения (U_{INP}" – U_{INN}") относительно опорного напряжения при K_G"$_{/38дБ}$ = ±3495264 · 2= ±6990528 из уравнения (16.7) не более: ΔU_{IN}" = (U_{INP}" – U_{INN}") = D_{ADC}/K_G"$_{/38дБ}$ = 32 768 / (±6990528) = ±4,6875 мВ и не более U_{FSR}"$_{/38дБ}$ = 9,375 мВ. Диапазон преобразования сигма-дельта модулятора при этом равен 1,5 В.

При работе с PGA = 38 дБ в 80 раз повышаются требования к стабильности опорного напряжения АЦП и уровню электрических помех на входах АЦП.

Пример 4 − Расчет входных напряжений и кодов преобразования АЦП в диапазоне рабочих температур ведется аналогично выше изложенному с учетом типовой зависимости, приведенной на рисунке 16.4а. Из рисунка 16.4а видно, что при заданном U_{REF} диапазон преобразования АЦП ограничивается пониженной температурой среды и именно эти значения U_{FSR} следует использовать для расчета шага квантования и коэффициента преобразования при выбранном U_{REF}. При номинальном опорном напряжении U_{REF} = 1,25 В, PGA = 0 дБ и U_{CC2} = 3,3 В диапазон преобразования U_{FSR} при пониженной температуре среды минус 60 °С − не менее 1,44 В (см. таблицу 3.3).

При этом верхнее практическое значение диапазона преобразования U_{FSR} определяется потребителем, исходя из выбранного U_{REF} и диапазона рабочих температур с учетом типовой зависимости на рисунке 16.4а.

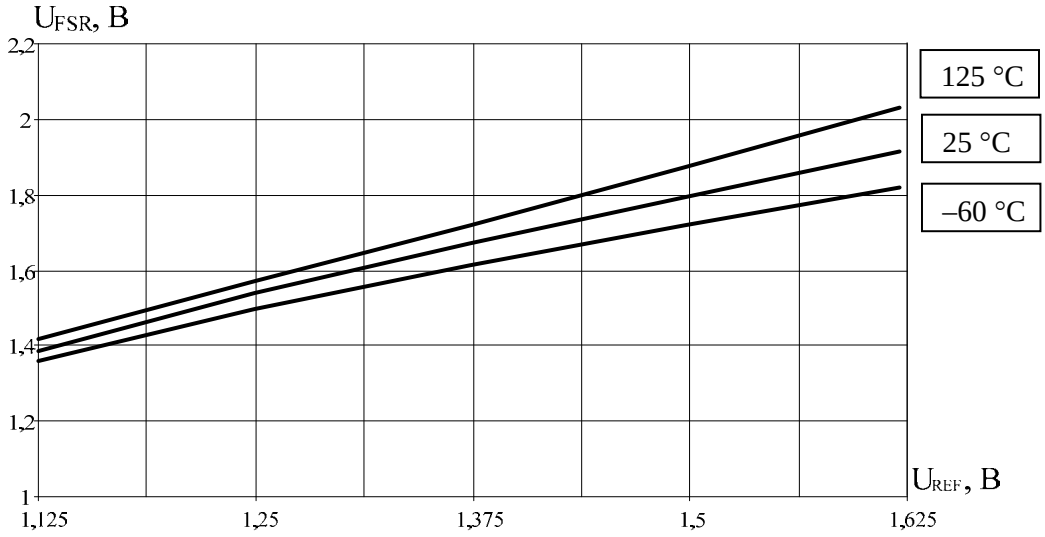


Рисунок 16.4а − Типовая зависимость диапазона преобразования каналов АЦП от опорного напряжения в диапазоне рабочих температур при PGA = 0 дБ, U_{CC2} = 3,3 В

Пример 5 − Пример временных диаграммм при оцифровке входного синусоидального напряжения ΔU_{IN} с частотой f_{IN} = 1кГц между входами АЦП и входных напряжений U_{INP}" и U_{INN}" относительно опорного напряжения при PGA= 0 дБ приведен на рисунке 16.4б.

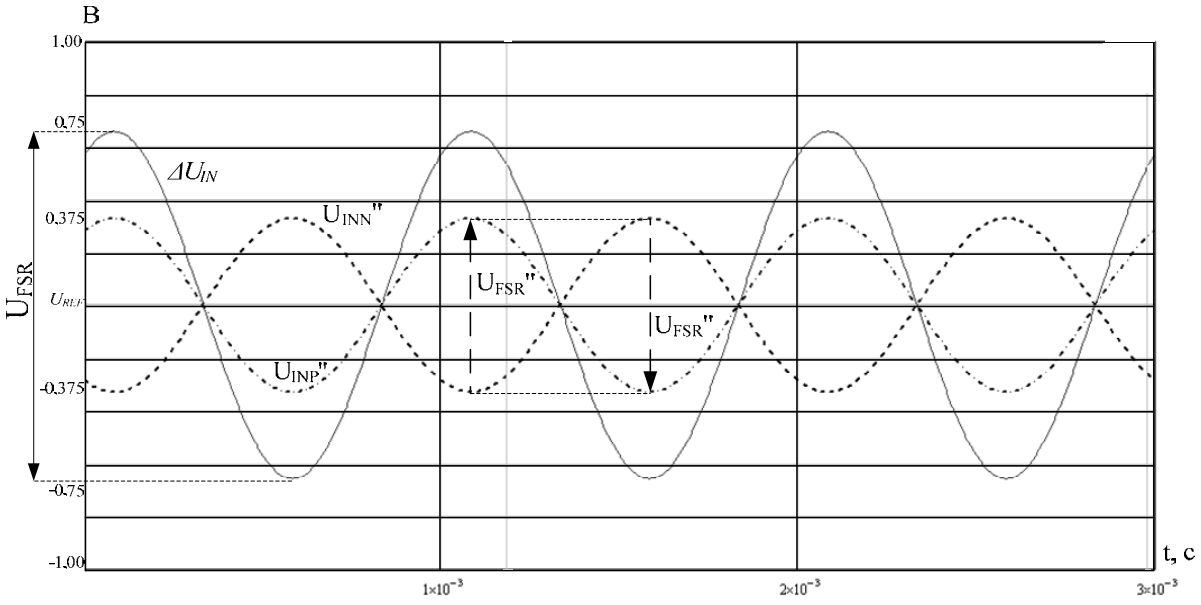


Рисунок 16.4б − Временные диаграммы входных напряжений АЦП при f_{IN} = 1 кГц

Комментарий к примерам

При PGA = 0 дБ диапазон входных напряжений преобразования между входами дифференциального усилителя АЦП в режиме дифференциального и одиночного входа равен диапазону преобразования U_{FSR} сигма-дельта модулятора АЦП (см. рисунок 16.4а). Входные напряжения при непрерывном преобразовании АЦП должны иметь 10 % запас от границ номинального диапазона преобразования. Дифференциальное включение входов АЦП позволяет избавиться от синфазной помехи.

В дифференциальном режиме возможна асимметричная подача входных напряжений относительно опорного напряжения, при этом результат преобразования (выходной код) определяется разницей входных напряжений в пределах диапазона преобразования АЦП.

Входы каналов АЦП допускают подачу входных напряжений от 0 В до U_{CC2} без гарантий результатов АЦП вне диапазона преобразования, что может использоваться при поиске границ диапазона преобразования U_{FS-}, U_{FS+} или при других оригинальных методах частного применения каналов АЦП с учетом передаточных характеристик, приведенных на рисунке 16.4.

16.2 Регистры управления и программирование АЦП

Существуют следующие способы управления АЦП:

- расширенное управление с помощью регистров управления ADCRA – ADCRH;
- запуск преобразования с помощью команды модуля HSO.

С помощью расширенного управления можно осуществить одновременный запуск любого количества выбранных каналов с требуемыми параметрами коэффициента усиления по входу для каждого из каналов. В АЦП реализована возможность запуска преобразования выбранного канала с помощью команды модуля HSO.

Ниже приведено детальное описание возможностей АЦП и принципов программирования для каждого из этих режимов.

16.2.1 Время запуска преобразования

Существует несколько способов запуска преобразования. Но независимо от способа запуска для включения каналов АЦП в работу требуется определенное время. Во время первого запуска преобразования, после включения контроллера, требуется время, равное 100 мкс, необходимое для включения источника опорного напряжения. Если запущен

старт преобразования какого-либо канала, то преобразование начнется автоматически, сразу после запуска источника опорного напряжения. Все остальные запуски не требуют дополнительного времени ожидания и начинаются непосредственно после установки соответствующих битов. Источник опорного напряжения остается включенным во время работы контроллера даже в том случае, когда никакое преобразование не выполняется. Исключение составляют режим сохранения мощности (POWERDOWN) и отключение АЦП через регистр CLKC. Переход контроллера в эти режимы вызывает выключение источника опорного напряжения. После выхода контроллера из режима сохранения мощности для запуска источника опорного напряжения так же требуется 100 мкс.

16.2.2 Запуск преобразования командой HSO

В АЦП предусмотрена возможность запуска преобразования всех каналов с помощью команды блока высокоскоростного вывода сигналов HSO. Запуск преобразования возможен лишь в том случае, если бит 1 регистра ADCRB установлен. Модуль HSO включает преобразование при выполнении команды CMD_TAG = 0FH. Процесс преобразования запускается, когда таймер 1 или таймер 2 (в зависимости от того, какой из них выбран в качестве базового) достигнет запрограммированного значения. После этого бит 1 в регистре ADCRB будет установлен, что вызовет немедленное начало преобразования всех каналов.

16.2.3 Механизм расширенного управления АЦП

Механизм расширенного управления АЦП обеспечивает гибкое управление каждым каналом, скоростью выборки, запуск преобразования любых выбранных каналов, а также перевод любого канала в режим непрерывного преобразования.

Индивидуальное управление каналами

Регистры ADCRC, ADCRD, ADCRE и ADCRF предназначены для управления коэффициентом усиления и запуском отдельных каналов модуля АЦП (рисунки 16.5 – 16.8).

ADCRC
регистр управления каналами 0 и 1 АЦП

$1FC2_H$ значение после Сброса: 00_H

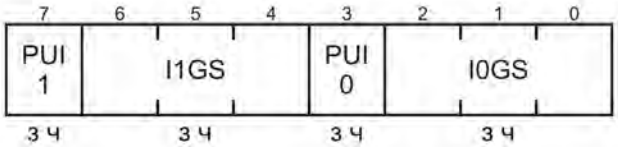


Рисунок 16.5 – Формат регистра ADCRC

Битовое поле I0GS определяет коэффициент усиления канала 0, битовое поле I1GS задает коэффициент усиления для канала 1. Бит PUI0 запускает преобразование канала 0, а бит PUI1 – преобразование канала 1 сразу после установки.

ADCRD
регистр управления каналами 2 и 3 АЦП

$1FC3_H$ значение после Сброса: 00_H

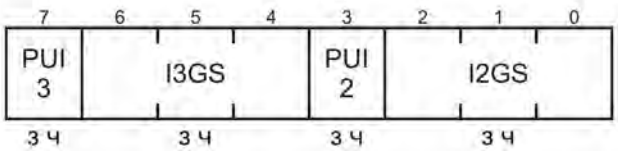


Рисунок 16.6 – Формат регистра ADCRD

Битовое поле I2GS определяет коэффициент усиления канала 2, битовое поле I3GS задает коэффициент усиления для канала 3. Бит PUI2 запускает преобразование канала 2, а бит PUI3 – преобразование канала 3 сразу после установки.

ADCRE
регистр управления каналами 4 и 5 АЦП

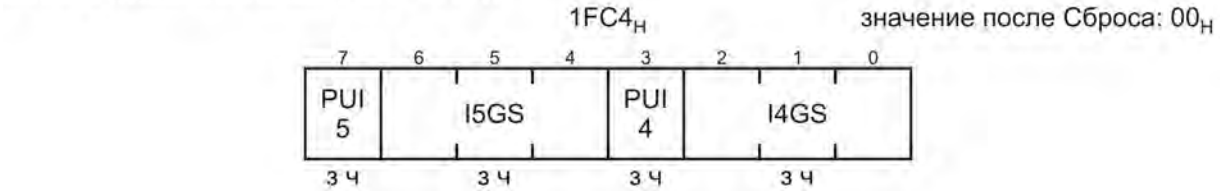


Рисунок 16.7 – Формат регистра ADCRE

Битовое поле I4GS определяет коэффициент усиления канала 4, битовое поле I5GS задает коэффициент усиления для канала 5. Бит PUI4 запускает преобразование канала 4, а бит PUI5 – преобразование канала 5 сразу после установки.

ADCRF
регистр управления каналами 6 и 7 АЦП

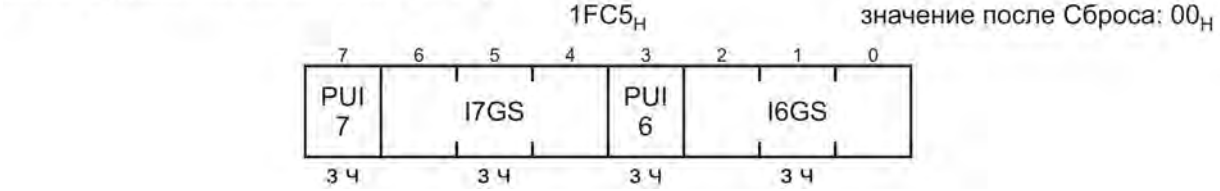


Рисунок 16.8 – Формат регистра ADCRF

Битовое поле I6GS определяет коэффициент усиления канала 6, битовое поле I7GS задает коэффициент усиления для канала 7. Бит PUI6 запускает преобразование канала 6, а бит PUI7 – преобразование канала 7 сразу после установки.

После запуска преобразования бит старта преобразования этого канала остается установленным в течение всего времени и соответствует непрерывному режиму преобразования. Если необходимо остановить преобразование, то необходимо сбросить биты PUI7 – PUI0.

Включение режимов дифференциальных входов осуществляется битами регистра ADCRB.

В таблице 16.1 приведены комбинации битов управления усилением канала IxGS и соответствующие им значения усиления.

Таблица 16.1 – Установки программируемого входного усиления

Поле IxGS регистра ADCRx			Усиление (PGA), дБ
Бит 2 (бит 7)	Бит 1 (бит 6)	Бит 0 (бит 5)	
0	0	0	0
0	0	1	6
0	1	0	12
0	1	1	18
1	0	0	20
1	0	1	26
1	1	0	32
1	1	1	38

Одновременный запуск преобразования всех каналов

Функциональные возможности АЦП позволяют осуществить одновременный запуск преобразования всех каналов. Одновременный запуск преобразования всех каналов разрешается установкой бита GPU (бит 0) регистра управления ADCRB.

Прерывание по завершению преобразования

Запуск и окончание преобразования выбранных каналов, независимо от их количества, осуществляется одновременно. Сразу после завершения преобразования выставляется запрос прерывания по окончанию преобразования, и на следующий такт ЦПУ в регистры результата заносятся результаты преобразования активных каналов. В случае работы блока АЦП в режиме цифровых компараторов возможно отключения прерывания по окончанию преобразования сбросом бита INTEN управляющего регистра ADCRA.

16.2.4 Регистры управления основными и дополнительными функциями АЦП

Описанные выше регистры управляют непосредственно преобразованием каждого канала. Кроме регистров управления преобразованием в АЦП предусмотрены дополнительные регистры управления основными и дополнительными функциями (рисунок 16.9).

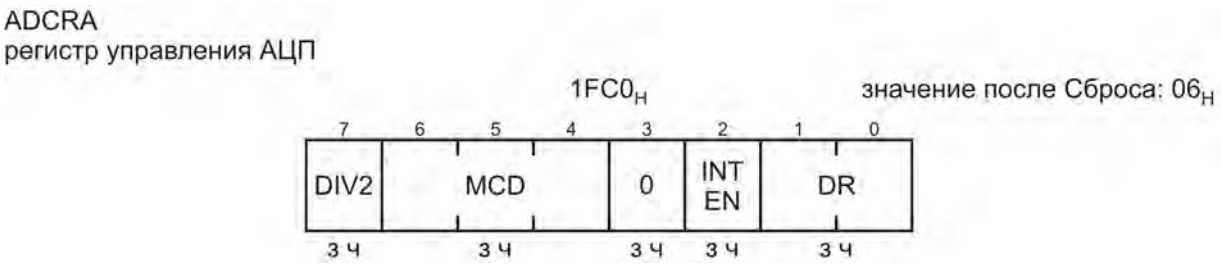


Рисунок 16.9 – Формат регистра ADCRA

Бит INTEN регистра ADCRA управляет генерацией прерывания по окончанию преобразований. Для запрещения прерывания по окончанию следует сбросить данный бит. Бит 3 регистра ADCRA предназначен для тестирования. Необходимо следить, чтобы этот бит был сброшен, иначе АЦП может неправильно функционировать. Бит DIV2 предназначен для дополнительного деления частоты на входе АЦП на два (всего АЦП, а не только аналоговой части). Рекомендуется выбирать скорость работы АЦП битами DR и MCD, а бит DIV2 использовать в случае, если требуемую частоту не удается получить с помощью других способов.

В схеме присутствует программируемый делитель тактовой частоты, который позволяет пользователю уменьшить частоту тактового сигнала АЦП в один, два, три, четыре или пять раз для формирования внутреннего опорного тактового сигнала (DMCLK) используемого для вычисления значений выборки. Коэффициент деления задается битовым полем MCD регистра ADCRA. В таблице 16.2 показаны коэффициенты деления в зависимости от установленных битов.

Таблица 16.2 – Значения внутренней опорной частоты (сигнал DMCLK) в зависимости от значений битов управления MCD регистра управления ADCRA

Поле MCD			Опорная частота (сигнал DMCLK)
Бит 6	Бит 5	Бит 4	
0	0	0	fмц
0	0	1	fмц /2
0	1	0	fмц /3
0	1	1	fмц /4
1	0	0	fмц /5
1	0	1	fмц
1	1	0	fмц
1	1	1	fмц

Битовое поле DR регистра ADCRA определяет скорость выборки. Делитель прореживания позволяет пользователю гибко подстроить значение выборки АЦП к требуемой программе. Максимальное значение скорости выборки соответствует DMCLK/256, возможные значения: DMCLK/512, DMCLK/1024 и DMCLK/2048. По умолчанию значение выборки соответствует скорости DMCLK/512. В таблице 16.3 показаны коэффициенты деления в зависимости от значения битов поля DR регистра ADCRA.

Таблица 16.3 – Значения прореживания в зависимости от состояния битов поля DR регистра ADCRA

Поле DR		Частота дискретизации
Бит 1	Бит 0	
0	0	DMCLK/2048
0	1	DMCLK/1024
1	0	DMCLK/512
1	1	DMCLK/256

Входное согласующее устройство на входе каждого канала позволяет инвертировать входной сигнал. Разрешение инверсии входного сигнала осуществляется установкой бита INV регистра управления ADCRB. Биты регистра ADCRH (рисунок 16.10) определяют каналы, на входах которых требуется произвести инверсию. Для активации инвертирования сигнала требуется установить соответствующий бит в этом регистре. Все сброшенные биты запрещают инвертирование входного сигнала соответствующего канала. Таким образом, регистр управления позволяет пользователю осуществлять выборочное инвертирование входных сигналов.

Рисунок 16.10 – Формат регистра ADCRH

Регистр ADCRB, изображенный на рисунке 16.11, осуществляет глобальное управление сбросом АЦП, одновременный запуск преобразования всех каналов,

включение буферизированного выхода REFOUT, выключение АЦП и включение дифференциального режима работы каналов (только для каналов 0 – 3).

ADCRB
регистр управления АЦП

$1FC1_H$ значение после Сброса: 32_H

7	6	5	4	3	2	1	0
0	RU	PUR EF	SEEN	R MOD	INV	PD OFF	GPU
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 16.11 – Формат регистра ADCRB

Установка бита GPU приводит к запуску преобразования всех каналов.

Бит PDOFF вместе с битом PUREF управляют включением/выключением опорного напряжения АЦП. Если бит PDOFF установлен, то опорное напряжение будет выключаться при переводе микроконвертера в режим POWERDOWN или запрещению подачи на модуль АЦП тактового сингала (регистр CLKCL). Если бит PDOFF сброшен, то управление ведется с помощью бита PUREF (1 – опорное напряжение включено, 0 – выключено). Необходимо помнить, что включение опорного напряжения происходит автоматически при запуске преобразования любого из каналов. Для выключения опорного напряжения необходимо завершить все преобразования.

С помощью установки бита RMOD можно сбросить аналоговые модуляторы, чьи биты выбора каналов CHRE7:0 регистра ADCRG установлены в единицы (рисунок 16.12). Для корректной работы данный бит должен быть сброшен.

ADCRG
регистр управления сбросом аналоговых модуляторов и включением дифференциального режима каналов АЦП

$1FC6_H$ значение после Сброса: FF_H

7	6	5	4	3	2	1	0
CHRE 7	CHRE 6	CHRE 5	CHRE 4	CHRE 3	CHRE 2	CHRE 1	CHRE 0
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 16.12 – Формат регистра ADCRG

После осуществления сброса аналоговых модуляторов потребуется выполнение не менее трех циклов преобразования для каждого из каналов для получения правильного результата.

Бит SEEN предназначен для включения режимов прямых (однополярных) и дифференциальных каналов. Установка единицы в данный бит включает режим одиночного входа тех каналов, чьи биты выбора каналов CHRE7:0 установлены в единицу. Установка битов выбора каналов в нули задает режим дифференциального включения входов (только для каналов 0 – 3). Не рекомендуется включать режим дифференциального включения входов для каналов 4 – 7, так как это приведет к неправильному функционированию АЦП.

Бит RU предназначен для включения буферизированного выхода опорного напряжения АЦП. Опорное напряжение подается через ключ на вывод 7 порта 0. Данное напряжение может применяться как для внешних схем, так и для калибровки канала АЦП в микроконвертере.

16.2.5 Результат преобразования

В состав АЦП входят восемь 16-разрядных регистров результата преобразования AD_RESULT0 – AD_RESULT7, по одному для каждого канала. Каждый из регистров результата занимает два байта в адресном пространстве контроллера и состоит из двух регистров: младшего байта регистра результата ADRESULTx(LO) и старшего байта регистра результата AD_RESULTx(HI). После окончания преобразования выбранных каналов, независимо от способа запуска преобразования, результат преобразования для каждого задействованного канала одновременно считывается с внутренней шины АЦП и помещается в соответствующий регистр результата AD_RESULTx, стирая старое значение. Содержимое регистров результата незадействованных каналов не изменяется.

ВНИМАНИЕ. После первого запуска преобразования канала, произведенного после включения питания, сброса аналоговой части АЦП, вывода контроллера или АЦП из режима сохранения мощности, требуется проведение трех циклов преобразования для получения корректного результата преобразования. После завершения каждого из этих циклов результат преобразования будет передаваться в регистр результата канала, но лишь после третьего цикла преобразования результат преобразования будет правильным. Все последующие преобразования канала, запущенные после этих трех циклов, выполняются за один цикл с выдачей корректного результата. Это условие необходимо выполнять для каждого канала АЦП.

16.3 Интерфейс с АЦ преобразователем

АЦП имеет независимые выводы для питания аналоговой и цифровой частей. Для питания аналоговой части схемы используются выводы ∩VCC3, и ∩0V3. Цифровая часть модуля подключена к шинам питания микроконтроллера. Информацию о подключении выводов питания аналоговой части АЦП и требования к источнику питания можно найти в соответствующем разделе руководства.

Выходное напряжение внутреннего источника выдается на вывод REFOUT. Вывод REFCAP используется для подключения внешнего буферного конденсатора. Температурная нестабильность напряжения REFCAP – 50 ppm/ºC. Минимальное сопротивление нагрузки, подключенной к REFOUT, составляет 1 кОм, максимальная емкость – 100 пФ.

Входное сопротивление аналого-цифровых преобразователей по входам ACHx, ACHPx, ACHNx составляет 25 кОм при частоте DMCLK = 16 МГц. Входное сопротивление обратно пропорционально частоте DMCLK ($4 \cdot 10^{11}$/ DMCLK).

Постоянное смещение сигнала аналогового входа при включении в режиме одиночного входа выполняется с использованием внутреннего источника опорного напряжения. Пример такого подключения представлен на рисунке 16.13.

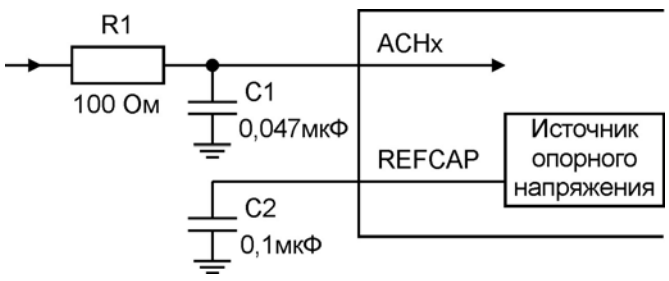


Рисунок 16.13 – Пример схемы подключения канала со смещением по постоянному току (режим одиночного входа)

В том случае, если смещение входного сигнала не равно уровню внутреннего источника опорного напряжения, должно использоваться подключение по переменному

току. Смещение входа в этом случае может быть реализовано согласно схеме, приведенной на рисунке 16.14.

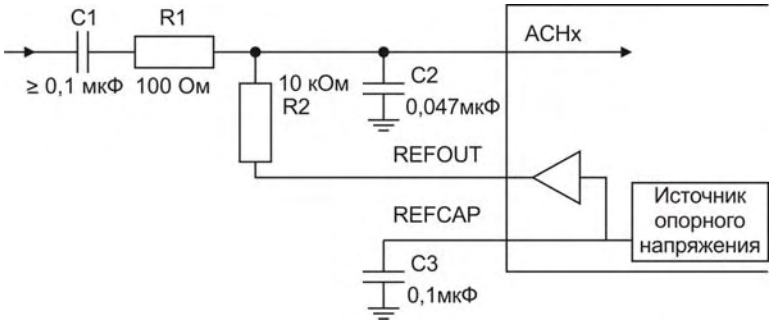


Рисунок 16.14 – Пример схемы подключения канала со смещением по переменному току (режим одиночного входа)

Пример включения АЦП в режиме дифференциальных входов представлен на рисунках 16.15 и 16.16.

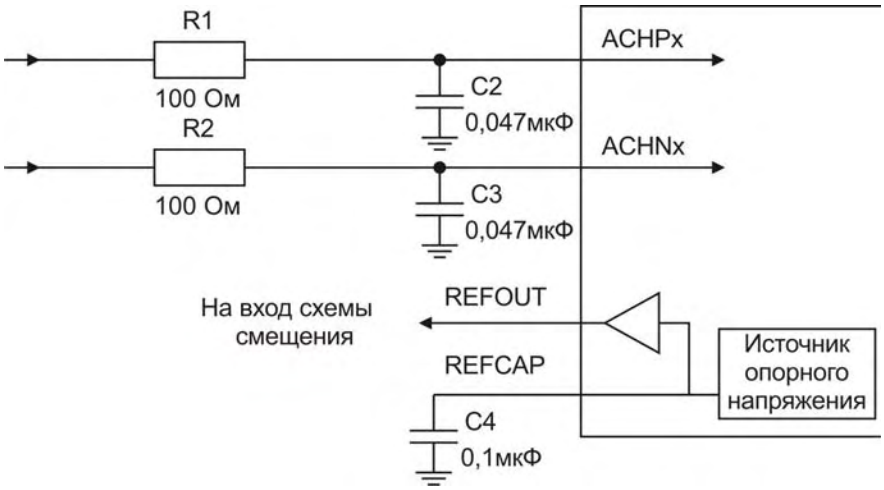


Рисунок 16.15– Пример схемы подключения канала со смещением по постоянному току (режим дифференциальных входов)

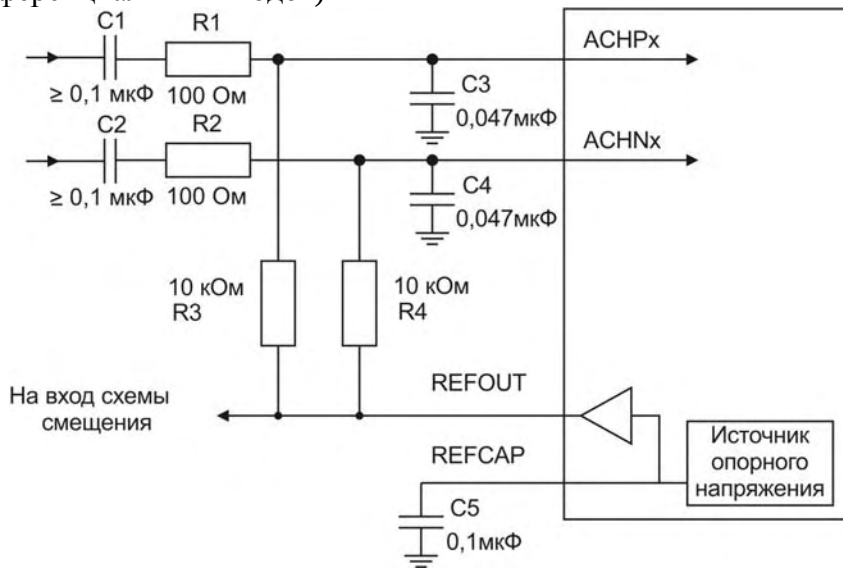


Рисунок 16.16– Пример схемы подключения канала со смещением по переменному току (режим дифференциальных входов)

Порт 0 может быть использован для операций с аналоговыми и цифровыми сигналами в одно время. Однако при чтении порта на аналоговые цепи может быть наведен небольшой шум, который способен повлиять на точность результата преобразования, протекающего в текущее время. По этой причине не рекомендуется использовать АЦП во время чтения порта 0.

16.4 Блок цифровых компараторов

Блок цифровых компараторов (рисунок 16.2) позволяет управлять генерированием прерываний АЦП. Так как основной режим работы АЦП – режим непрерывного преобразования каналов, то имеет смысл отключение прерывания по окончанию преобразования и включение прерываний по выходу контролируемого параметра за заданный диапазон. Основные функциональные возможности блока компараторов:

- генерация прерывания в случае, если результат преобразования меньше, либо равен заданному значению (рисунок 16.17);
- генерация прерывания в случае, если результат преобразования больше, либо равен заданному значению (рисунок 16.18);
- генерация прерывания в случае, если результат преобразования попадает в диапазон, заданный двумя регистрами границ – верхней и нижней (режим доступен только для каналов 0 – 5) (рисунок 16.19);
- генерация прерывания в случае, если результат преобразования выходит из диапазона, заданный двумя регистрами границ – верхней и нижней (режим доступен только для каналов 0 – 5) (рисунок 16.20).

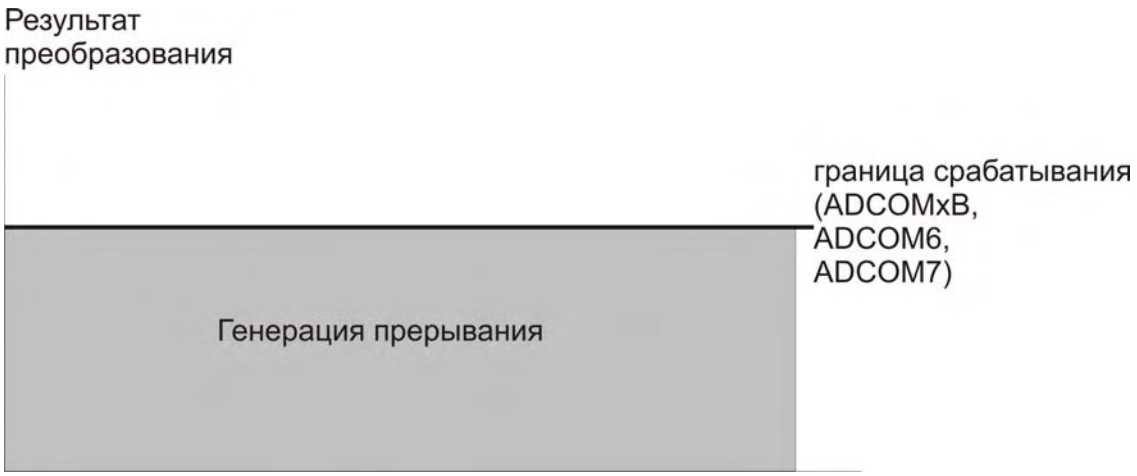


Рисунок 16.17 – Срабатывание компараторов по нижней границе

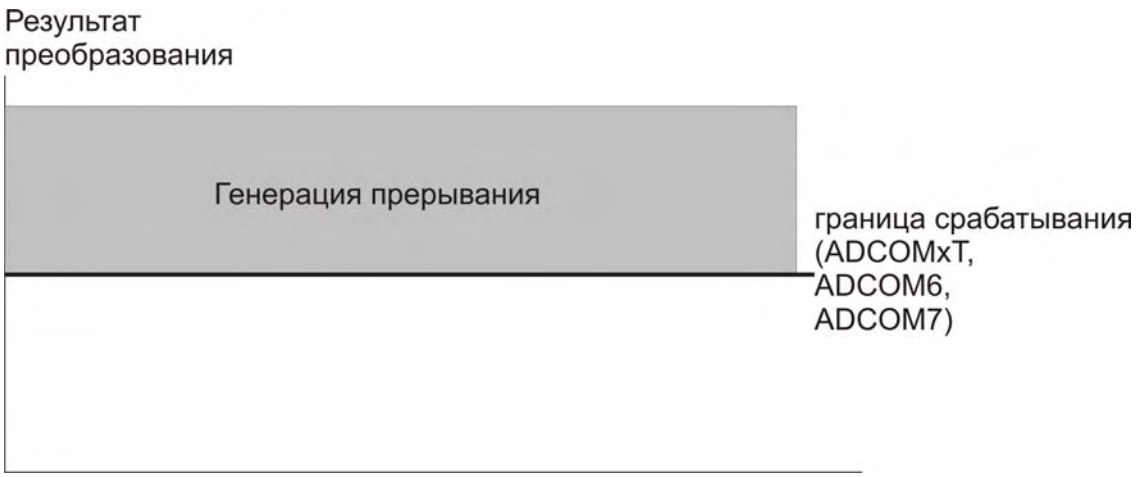


Рисунок 16.18 – Срабатывание компараторов по верхней границе

Рисунок 16.19 – Срабатывание компараторов по попаданию в диапазон

Рисунок 16.20 – Срабатывание компараторов по выходу из диапазона

В случае если АЦП не используется, компараторы могут контролировать, например, результаты арифметических расчетов. Так как регистры результатов аналого-цифровых преобразований доступны для записи, то они могут фигурировать в командах умножения, сложения, пересылки и т. д. То есть если организовать программу так, чтобы результаты вычислений хранились в регистрах результата АЦП, то возможен их аппаратный контроль. Так как каждый цифровой компаратор имеет свой приоритет, то возможна организация сложных условий сравнения (например, прерывание срабатывает, когда условие цифрового компаратора 0 не выполнилось, а компаратора 1 – выполнилось).

Блок состоит из восьми компараторов, которые каждый машинный цикл сравнивают значение, записанное в регистре результата преобразования соответствующего канала АЦП, со значением, указанным в задающих регистрах. Компараторы каналов 0 – 5 имеют по два 16-разрядных регистра, задающих как верхнее (ADCOMxT), так и нижнее (ADCOMxB) граничные значения. Компараторы каналов 6 – 7 имеют по одному 16-разрядному регистру (ADCOM6, ADCOM7), поэтому не подходят для контроля диапазона. Управляющие регистры ADCOMCL и ADCOMCH определяют, какие типы контроля использовать для каждого цифрового компаратора. Регистры ADINTCL и ADINTCH содержат флаги срабатывания компараторов и биты разрешения генерации прерывания INT01 каждым из компараторов. Флаги срабатывания компараторов устанавливаются вне зависимости, разрешена ли генерация прерывания INT0 этим каналом или нет. Формат регистров верхней и нижней границ представлен на рисунках 16.21 и 16.22.

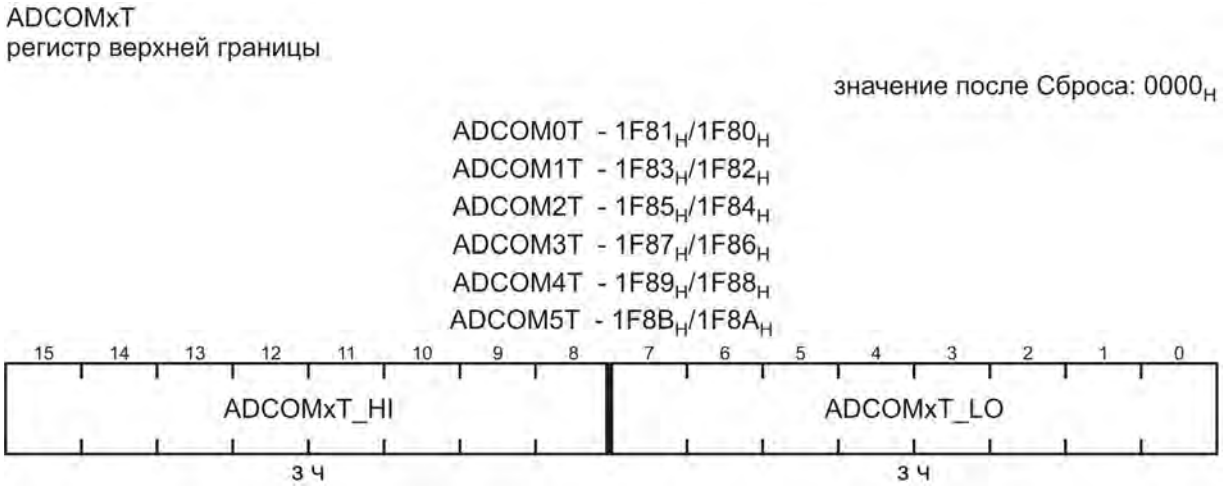


Рисунок 16.21 – Формат регистров ADCOMxT

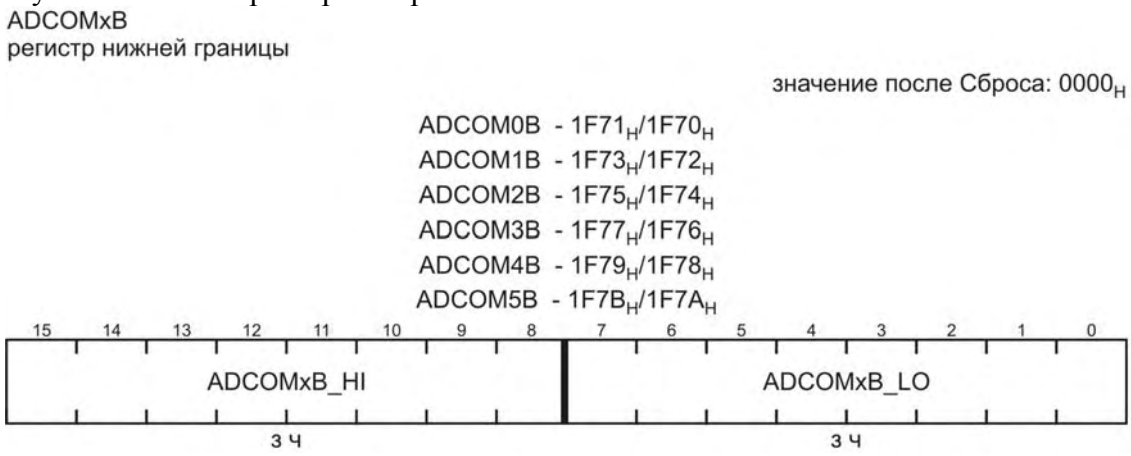


Рисунок 16.22 – Формат регистров ADCOMxB

Формат регистров границ для 6 и 7 каналов представлен на рисунке 16.23.

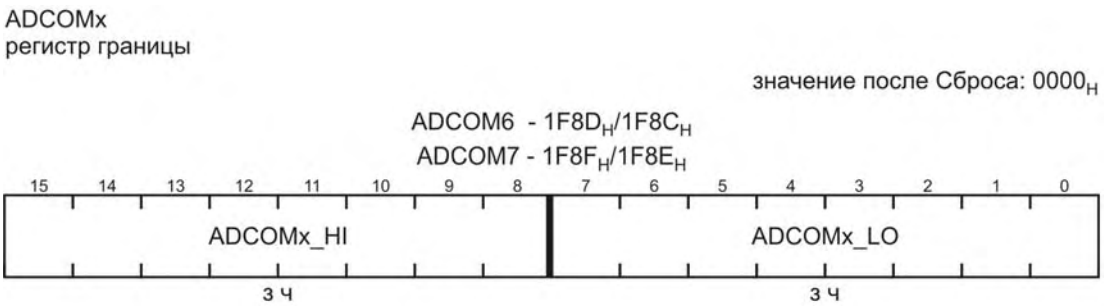


Рисунок 16.23 – Формат регистров ADCOM6 и ADCOM7

Регистры управления ADCOMCL и ADCOMCH предназначены для задания типа срабатывания компараторов. Формат регистров представлен на рисунках 16.24 и 16.25. Значения управляющих битов представлены в таблице 16.4.

ADCOMCL
регистр управления цифровыми компараторами 1

$1F7E_H$ значение после Сброса: 00_H

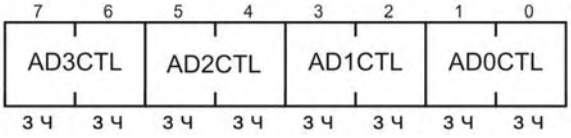


Рисунок 16.24 – Формат регистра ADCOMCL

ADCOMCH
регистр управления цифровыми компараторами 2

$1F7F_H$ значение после Сброса: 00_H

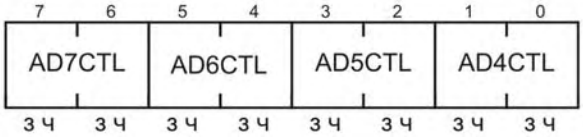


Рисунок 16.25 – Формат регистра ADCOMCH

Таблица 16.4 – Значение битов регистров управления цифровыми компараторами ADCOML и ADCOMCH

Мнемоника	Описание
ADXCTL	Биты управления типом отслеживаемых событий цифровым компаратором для АЦП: 00 – срабатывание компаратора по нижней границе (рисунок 16.17); 01 – срабатывание компаратора по верхней границе (рисунок 16.18); 10 – срабатывание компаратора по выходу из диапазона (рисунок 16.20, только для АЦП 0 – 5); 11 – срабатвание компаратора по попаданию в диапазон (рисунок 16.19. только для АЦП 0 – 5)

Каждый цифровой компаратор имеет свой приоритет. Самый большой приоритет имеют компараторы 0 и 4. Распределение приоритетов компараторов представлено в таблице 16.5.

Таблица 16.5 – Распределение приоритетов цифровых компараторов

Номер компаратора	Приоритет	Группа приоритетов
0	4	1
1	3	1
2	2	1
3	1	1
4	4	2
5	3	2
6	2	2
7	1	2

Для срабатывания прерывания цифрового компаратора необходимо, чтобы оно было разрешено (установлен соответствующий бит в регистре ADINTCL или ADINTCH), выполнилось условие выбранного компаратора и не выполнились условия компататоров с более бысоким приоритетом из группы. Так, для срабатывания компаратора 5 канала

необходимо, чтобы был установлен бит AD5INTEN, выполнилось условие компаратора 5 и не выполнилось условие компаратора 4.

В случае если произошло срабатывание цифрового компаратора, выставляется соответствующий бит в регистре ADINTCL или ADINTCH. В случае если разрешено прерывание при срабатывании какого-либо компаратора, блок генерирует прерывание INT01. Форматы управляющих регистров ADINTCL и ADINTCH представлены на рисунках 16.26 и 16.27 соответственно.

ADINTCL
регистр управления прерываниями цифровых компараторов 1

$1F7C_H$ значение после Сброса: 02_H

7	6	5	4	3	2	1	0
AD3 FLAG	AD3 INTEN	AD2 FLAG	AD2 INTEN	AD1 FLAG	AD1 INTEN	AD0 FLAG	AD0 INTEN
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 16.26 – Формат регистра ADINTCL

ADINTCH
регистр управления прерываниями цифровых компараторов 2

$1F7D_H$ значение после Сброса: 02_H

7	6	5	4	3	2	1	0
AD7 FLAG	AD7 INTEN	AD6 FLAG	AD6 INTEN	AD5 FLAG	AD5 INTEN	AD4 FLAG	AD4 INTEN
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок 16.27 – Формат регистра ADINTCH

Биты ADXINTEN разрешают генерацию прерывания соответствующим компаратором в блоке цифровых компараторов. Для разрешения надо записать единицу в нужный бит. Флаги ADXFLAG показывают, условия какого компаратора сработали. Так как по умолчанию после сброса все компараторы настроены на фиксацию момента выхода за пределы или равенства результата преобразования нижней границы (рисунок 16.17), а значение результатов всех преобразований и значения нижних границ в начальный момент равно 0000h, то биты AD0FLAG и AD4FLAG автоматически после сброса устанавливаются в состояние логической единицы.

17 Цифро-аналоговый преобразователь

Микроконвертер 1874ВЕ96Т имеет встроенный 14-разрядный цифро-аналоговый преобразователь, позволяющий получать на выходе ИС ток, пропорциональный записанному в управляющий регистр значению.

Схемы применения цифро-аналоговых преобразователей относятся не только к области преобразования код – аналог. Пользуясь их свойствами можно определять произведения двух или более сигналов, строить делители функций, аналоговые звенья, управляемые от микроконвертеров, такие как аттенюаторы, интеграторы. Важной областью применения ЦАП являются также генераторы сигналов, в том числе сигналов произвольной формы.

ЦАП могут использоваться в следующих областях:

- обработка чисел, имеющих знак;
- перемножители и делители функций;
- аттенюаторы и интеграторы на ЦАП;
- системы прямого цифрового синтеза сигналов:
    - бытовая техника;
    - обработка звука и видео;
    - системы цифровой обработки сигналов;
    - цифровое телевидение;
    - радиолокация;
    - управление процессами в промышленности;
    - медицинское оборудование;
    - телеметрия;
    - измерительные приборы и диагностическое оборудование;
- цифровая обработка сигналов.

Характеристики блока ЦАП:

- 14-бит разрешение (16 384 значений выходного тока);
- дифференциальные токовые выходы – максимум тока от 2 до 20 мА;
- потребляемая мощность блока 135 мВт;
- режим пониженного потребления 15 мВт;
- внутренний источник опорного напряжения 1,2 В.

17.1 Функциональный обзор ЦАП

Структурная схема блока представлена на рисунке 17.1.

ЦАП содержит следующие функциональные блоки:

- источник опорного напряжения 1,2 В (ИОН);
- управляющий усилитель (УУ);
- матрицу источников тока;
- переключатели тока.

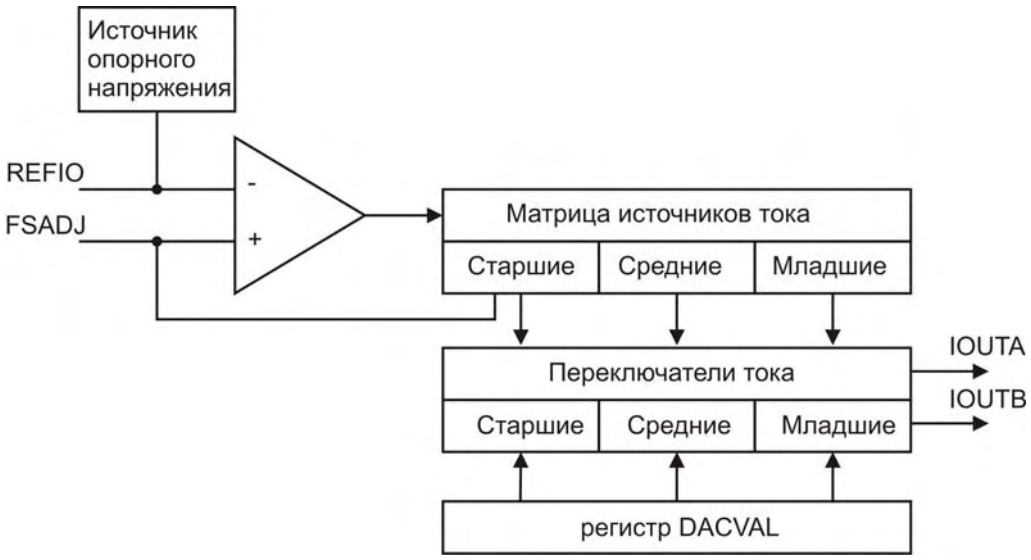


Рисунок 17.1 – Структурная схема блока ЦАП

Микросхема имеет независимые выводы для питания аналоговых блоков цифро-аналогового преобразователя. Для питания аналоговых блоков используются выводы ∩VCC3 и ∩0V3.

Блок работает от внутреннего источника опорного напряжения.

Микроконвертер имеет два токовых выхода IOUTA и IOUTB, которые могут включаться в схеме как на отдельные нагрузки, так и на дифференциальную нагрузку. Дифференциальное напряжение U_{DIFF}, формируемое на нагрузках R_{LOAD}, образуется между U_{OUTA} и U_{OUTB} и может быть преобразовано в однополярное напряжение через трансформатор или дифференциальный усилитель.

Выходное сопротивление токовых выходов можно представить как эквивалент параллельного соединения PMOS ключей с типовым сопротивлением 50 кОм и емкостью 5 пФ.

Токи выходов IOUTA и IOUTB поддерживают свое значение на этих выходах в диапазоне напряжений от минус 1,0 до 1,25 В.

Диапазон выходных напряжений на выходах стабилизированного тока IOUTA и IOUTB в положительной области незначительно зависит от тока полной шкалы I_{OUTFS}. Он ухудшается незначительно от его номинального 1,25 В для I_{OUTFS} = 20 мА до 1,0 В для I_{OUTFS} = 2 мА. Для оптимальной линейности токов I_{OUTA} и/или I_{OUTB} необходимо использовать на выходе микросхемы преобразователь ток–напряжение, что позволяет сохранять неизменное выходное сопротивление. Включение микросхемы с пониженным перепадом напряжения на выходах IOUTA и IOUTB в дифференциальном или в несимметричном включении снижает зависимость сигнала от выходного сопротивления, таким образом, улучшаются характеристики сигнала.

Значительное улучшение характеристик искажений и шумов реализуется дифференциальным включением нагрузки. Оптимальные искажения достигаются, когда максимальный размах сигнала на выходах IOUTA и IOUTB не превышает 0,5 В.

Шумовые характеристики и характеристики искажений слабо зависят от цифрового и аналогового питания, так же как и от тока полной шкалы I_{OUTFS}. При аналоговом напряжении питания 3.0 В обеспечивается максимальный уровень тока источника тока и дифференциальных ключей и обеспечивается улучшение коэффициента нелинейных искажений. Несмотря на то, что максимальный выходной ток можно установить в пределах от 2 до 20 мА, при его значении 20 мА обеспечиваются наилучшие характеристики шума и характеристики искажений. На характеристики шума влияет

напряжение цифрового питания (U_{CC1}), выходная частота сигнала и тактовая частота. В заключение приведем оптимальные условия для цифро-аналогового преобразования:

- дифференциальное включение токовых выходов;
- размах положительного напряжения на IOUTA и IOUTB ограничен до 0,5 В;
- выходной ток полной шкалы I_{OUTFS} равен 20 мА.

17.2 Источник опорного напряжения

ЦАП имеет внутренний источник опорного напряжения (ИОН). Выходное напряжение внутреннего источника выдается на выход REFIO с нагрузочной способностью не более 100 нА. В этом случае к выходу REFIO обязательно должен быть подключен внешний керамический конденсатор емкостью не менее 0,1 мкФ для обеспечения стабильности опорного напряжения. Допустимо использование внешнего источника опорного напряжения с низким выходным сопротивлением для исключения влияния внутреннего источника.

17.3 Управляющий усилитель

Управляющий усилитель позволяет регулировать выходной ток полной шкалы (I_{OUTFS}) в диапазоне от 2 до 20 мА. Для установки тока полной шкалы используется внешний регулировочный резистор (R_{SET}), подключаемый между выводами FSADJ и ∩GND. При величине сопротивления резистора 2 кОм обеспечивается максимальный ток полной шкалы 20 мА, при сопротивлении 20 кОм – минимальный ток 2 мА. АЧХ усилителя корректируется внутренней емкостью 150 пФ.

Ток полной шкалы I_{OUTFS} является функцией опорного напряжения и сопротивления внешнего резистора:

$$I_{OUTFS} = 32 \times I_{REF},$$

где $I_{REF} = U_{REFIO}/R_{SET}$.

17.4 Токовые выходы

Цифро-аналоговый преобразователь имеет комплементарные токовые выходы IOUTA и IOUTB. Сумма токов на выходах ($I_{OUTA} + I_{OUTB}$) равна току полной шкалы I_{OUTFS}. Токи вытекают из выходов во внешнюю нагрузку, подключаемую к общему выводу ∩GND.

Ток выхода IOUTA практически равен I_{OUTFS}, когда все биты входных данных установлены в состояние «1» (т. е. код ЦАП = 16383), в то время как на выходе IOUTB нет тока. Ток выходов является функцией входного кода и тока полной шкалы:

$$I_{OUTA} = (DACVALUE/16384) \times I_{OUTFS},$$

$$I_{OUTB} = ((16383 - DACVALUE)/16384) \times I_{OUTFS},$$

где код DACVALUE = 0, 1, 2 … 16383 – десятичное представление кода.

Токовые выходы обычно подключаются напрямую к нагрузочным сопротивлениям или нагружаются дифференциально на трансформатор. Если требуется соединение по постоянному току, выходы IOUTA и IOUTB должны быть напрямую подключены к правильно подобранным нагрузочным сопротивлениям R_{LOADA}, R_{LOADB}, подключаемым вторым выводом к выводу аналоговой земли ∩GND. В качестве нагрузки может использоваться кабель с сопротивлением 50 или 75 Ом. Напряжения несимметричного выхода IOUTA и IOUTB соответственно рассчитываются:

$$U_{OUTA} = I_{OUTA} \times R_{LOADA},$$

$$U_{OUTB} = I_{OUTB} \times R_{LOADB}.$$

Напряжения U_{OUTA} и U_{OUTB} во всем диапазоне выходных токов не должны превышать предельно допустимых величин минус 1,0 В и 1,25 В.

Дифференциальное выходное напряжение U_{DIFF} определяется разностью токов I_{OUTA} и I_{OUTB}, соответственно:

$$U_{DIFF} = (I_{OUTA} - I_{OUTB}) \times R_{LOAD},$$

где $R_{LOAD} = R_{LOADA} = R_{LOADB}$).

Заменив значения I_{OUTA}, I_{OUTB} и U_{DIFF}, получаем следующее выражение:

$$U_{DIFF} = \{(2\times DACVALUE - 16383)/16384\} \times (32R_{LOAD}/ R_{SET}) \times U_{REFIO}.$$

Последние два равенства показывают преимущества применения микросхемы в дифференциальном включении выходов. Во-первых, это помогает ликвидировать синфазные помехи, возникающие из-за шума I_{OUTA} и I_{OUTB}, искажений и синфазных отклонений токов. Во-вторых, обеспечивается удвоенное выходное напряжение U_{DIFF} по сравнению с напряжениями несимметричного выхода (U_{OUTA} и U_{OUTB}), таким образом, обеспечивается двойная мощность сигнала в нагрузке.

Улучшение температурного дрейфа обеспечивается для несимметричных (U_{OUTA} и U_{OUTB}) и дифференциальных (U_{DIFF}) выходов посредством подбора сопротивления R_{LOAD}, R_{SET}.

17.5 Установка выходных токов

Установку выходных токов обеспечивают переключатели тока. Необходимый набор токов выдается матрицей источников тока. При этом выходные токи матрицы разделены на три группы: старшие, средние и младшие.

Старшие токи (32 тока) равны опорному току I_{REF}. Один старший ток выдается на выход FSADJ, остальные (31 ток) поступают на переключатели тока. Подключение старших токов к выходам IOUTA, IOUTB определяется состоянием битов DACVALUE13 – DACVALUE9 управляющего регистра ЦАП.

Средние токи (15 токов) равны I_{REF} /16. Подключение средних токов к выходам IOUTA, IOUTB определяется состоянием битов DACVALUE8 – DACVALUE5.

Младшие токи (5 токов) равны соответственно I_{REF} /32, I_{REF} /64, I_{REF} /128, I_{REF} /256 и I_{REF} /512. Подключение младших токов к выходам IOUTA, IOUTB определяется состоянием битов DACVALUE4 – DACVALUE0. Формат управляющего регистра ЦАП представлен на рисунке 17.2. Для корректной работы блока необходимо записывать значение DACVAL целым словом, а не побайтно.

DACVAL
регистр управления ЦАП

$11_H/10_H$ значение после Сброса: $C000_H$

(горизонтальное окно 1 - запись, чтение)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLEEPC	SLEEP	DB13	DB12	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

Рисунок 17.2 – Формат регистра DACVAL

17.6 Включение ЦАП

В целях уменьшения общего потребления энергии системой возможен перевод блока в режим пониженного энергопотребления. Существует два способа управления этим переводом. Если установлен бит SLEEPC регистра DACVAL, то перевод в режим пониженного потребления производится автоматически. Это делается обнулением соответствующего бита регистра CLKCL (бит 6) или переводом всего МК в режим POWERDOWN. Если бит SLEEPC сброшен, то управление ведется битом SLEEP
(«1» – ЦАП выключен, «0» – ЦАП включен). По умолчанию блок цифро-аналогового преобразователя выключен. Для включения необходимо установить бит CLKCL.6.

18 Широтно-импульсный модулятор PWM

МК имеет три модуля PWM (рисунок 18.1). Каждый выход обеспечивает выдачу импульса переменной длительности с фиксированной частотой. Эти выходы могут использоваться для управления двигателями, оптимально работающими с PWM импульсами без фильтрации, или же эти импульсы могут быть отфильтрованы для получения гладких аналоговых сигналов.

В этом разделе приведен функциональный обзор модулей PWM, описывается их программирование и дан пример схемы преобразования выходов PWM в аналоговый сигнал.

Смотрите приложение Б для получения информации о выходах PWM.

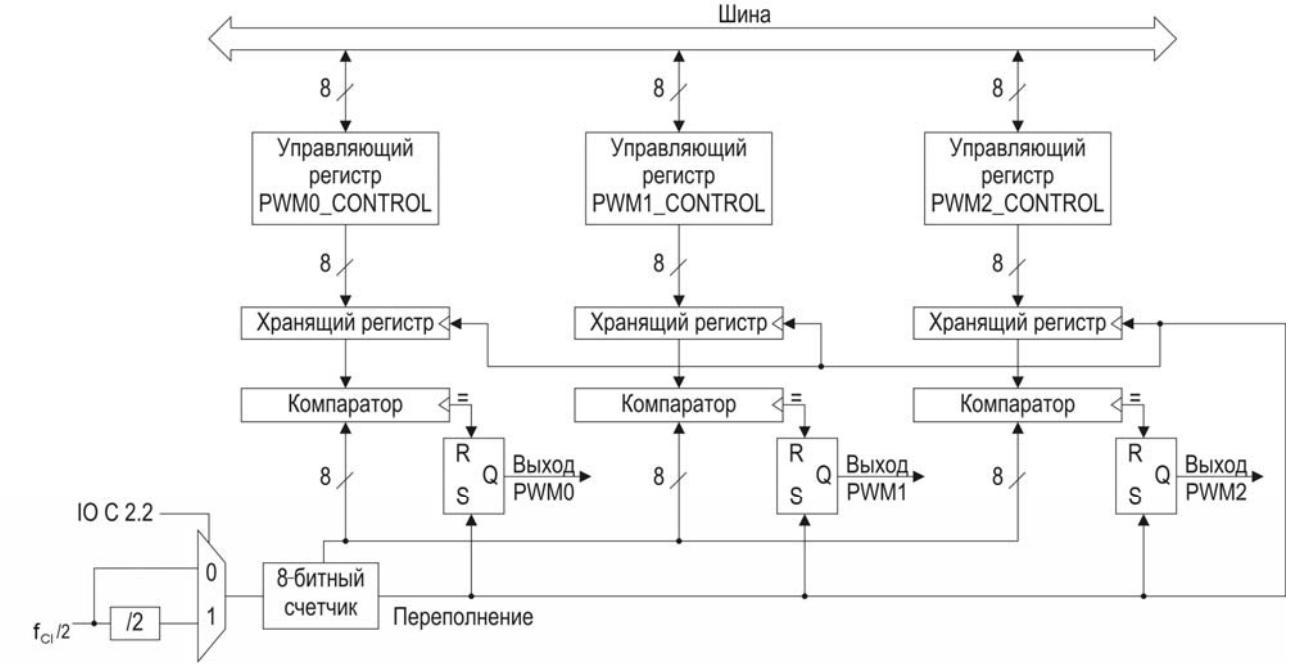


Рисунок 18.1 – Блок-схема модуля PWM

18.1 Функциональный обзор PWM

Модули PWM имеют следующие основные компоненты:

- 8-битный счетчик;
- предделитель на 2;
- три компаратора;
- три управляющих регистра (PWMx_CONTROL, где x равен 0, 1 или 2);
- три хранящих регистра (Holding Registers);
- три RS-триггера.

SLOW_PWM бит (IOC2.2) управляет периодом выходных импульсов, разрешая или запрещая работу предделителя на два. Разрешение работы предделителя заставляет 8-битный счетчик увеличиваться каждые два такта; запрещение вызывает увеличение счетчика каждый такт.

Каждый управляющий регистр содержит 8-битное значение, загружаемое в регистр хранения, когда 8-битный счетчик переполняется.

Компараторы сравнивают содержимое регистров хранения со значением счетчика. Когда значение счетчика равно нулю, выход PWMx находится в высоком уровне. Он остается в этом состоянии, пока значение счетчика не будет равно значению, находящемуся в регистре хранения, после сравнения выход переходит в низкий уровень. Загрузка в регистры PWMx_CONTROL значения 00H обеспечивает низкий уровень

на выходе. При переполнении счетчика – на выходе вновь высокий уровень. На рисунке 18.2 показана типичная выходная диаграмма на выходе PWM.

Рисунок 18.2 – Форма выходного сигнала PWM

18.2 Программирование рабочего цикла PWM

Регистр PWMx_CONTROL, вместе с битом SLOW_PWM (IOC2.2), определяет длительность высокого уровня импульса на выходе PWMx, управляя рабочим цилом. Значение, записанное в регистр PWMx_CONTROL, может обеспечивать длительность от 0 до 255 тактов (то есть от 0 до 99,6 % рабочего цикла). Установка IOC2.2 разрешает предварительный делитель PWM; полная длительность импульса становится 512 тактов, а значение регистра PWMx_CONTROL умножается на четыре. Очистка IOC2.2 запрещает работу предделителя; полная длительность импульса – 256 тактов, а значение регистра PWMx_CONTROL умножается на два. В таблице 18.1 приведены выходные частоты PWM.

Таблица 18.1 – Выходная частота PWM

IOC2.2	Частота на выводе XTAL1 (f_{CI})			
	8,0 МГц	10,0 МГц	16,0 МГц	20,0 МГц
0	15,6 кГц	19,6 кГц	31,25 кГц	39,1 кГц
1	7,8 кГц	9,8 кГц	15,63 кГц	19,5 кГц

Для вычисления периода PWM и времени (в мкс), в течение которого выход PWM остается в высоком уровне, следует использовать следующие формулы. Заметим, что значение PWMx_CONTROL – 8-битное значение и f_{CI} – частота вывода XTAL1 в МГц.

Предделитель запрещён (IOC2.2 = «0»):

$$\text{Период PWM} = \frac{512}{f_{CI}},$$

$$\text{Высокий PWM.x} = \frac{\text{PWMx\_CONTROL} \times 2}{f_{CI}}.$$

Предделитель разрешён (IOC2.2 = «1»):

$$\text{Период PWM} = \frac{1024}{f_{CI}},$$

$$\text{Высокий PWM.x} = \frac{\text{PWMx\_CONTROL} \times 4}{f_{CI}}.$$

Например, если f_{CI} равна 16 МГц, то период на выходе PWM – 32 мкс. Если PWM0_CONTROL равен 8AH (десятичное 138) и IOC2.2 очищен, PWM0 сохраняет высокий уровень 17,25 мкс (и низкий – 14,8 мкс) из общих 32 мкс; в результате

скважность – приблизительно 54 %. Когда IOC2.2 установлен, то же самое значение в регистре PWM0_CONTROL будет формировать период в 64 мкс, и PWM0 будет сохранять высокий уровень 34,5 мкс (и низкий – 29,5 мкс), скважность при этом также около 54 %.

18.3 Разрешение выходов PWM

Каждый выход PWM мультиплексирован с выводом порта. В таблице 18.2 приведены альтернативные функции порта вместе с установками регистров, которые выбирают выходы PWM взамен функции порта.

Выбор выходных функций PWM1 или PWM2 делает невозможным выполнение функций квазидвунаправленного порта 1 и разрешает выход с повышенной нагрузочной способностью.

Таблица 18.2 – Альтернативные функции выхода PWM

Выход PWM	Альтернативная функция порта	Выход PWM разрешён, если:
PWM0	P2.5	IOC1.0 = «1»
PWM1	P1.3	IOC3.2 = «1»»
PWM2	P1.4	IOC3.3 = «1»

19 Специальные режимы работы

МК 1874BE96T поддерживает три специальных рабочих режима: IDLE (холостой ход), POWERDOWN (низкое энергопотребление), SLOW (уменьшение внутренней частоты). Режимы IDLE и POWERDOWN снижают потребление энергии. Схема поддерживает также режим электрической изоляции ONCE. Данный раздел описывает каждый режим, вход и выход из режима. Смотрите команды в приложении А, информацию о сигналах – в приложении Б.

19.1 Режим IDLE

В режиме IDLE тактирование ЦПУ заморожено 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, модуль отладки 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 в IDLE режиме. Устройство должно выходить из IDLE режима каждые 64К такта, чтобы очистить регистр WATCHDOG (0AH), иначе таймер WDT сбросит устройство.

19.1.1 Вход в IDLE режим

Чтобы войти в IDLE режим, следует выполнить команду IDLPD #1.

19.1.2 Выход из режима IDLE

Прерывания или аппаратный сброс выведут устройство из IDLE режима. Из-за того, что вся периферия остается активной в этом режиме, любой разрешенный источник прерывания может выработать прерывание. Когда произойдет прерывание, ЦПУ выходит из режима холостого хода и начинает выполнять соответствующую подпрограмму обслуживания прерывания. Только прерывание, назначенное на обработку стандартным методом, может вывести контроллер из режима IDLE. Если обработка осуществляется PTS, то ИС может выполнять передачи (в том числе во внешнюю память), находясь в режиме холостого хода. При этом во время передачи выводы управления внешней шиной принимают свои активные состояния.

Если контроллер был выведен из 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 POWERDOWN переводит МК в состояние с очень низким потреблением энергии. Если значение U_{CC1} сохраняется, то регистры специальных функций (SFRs) и регистры ОЗУ сохраняют данные. Все сигналы внутреннего тактирования замораживаются в состоянии логического нуля, генератор отключается, блоки АЦП и ЦАП переводятся в SLEEP режим (по умолчанию). Энергопотребление падает. Значение I_{CC1} снижается до тока утечки устройства.

Во время режима POWERDOWN, внутренний генератор и схема тактирования запрещены. Выводы управления шиной (ALE, RD#, WR#, INST и BHE#) переведены в неактивное состояние. Все выходы сохраняют значение, находящееся 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».

19.2.1 Запрещение режима POWERDOWN

Режим POWERDOWN запрещается, когда бит PD (CCR.0) очищен. В CCR загружается байт конфигурации кристалла (CCB ячейка 2018H) при сбросе устройства.

19.2.2 Переход в режим POWERDOWN

После перехода в режим POWERDOWN завершаются следующие задачи:

- завершается обмен через последовательный порт. Когда устройство выходит из режима POWERDOWN, последовательный порт активизируется и возобновит обмен; если была потеря данных, то возможна передача или прием неправильных данных;
- завершаются все аналоговые преобразования. Если переход в режим POWERDOWN произошел во время преобразования, то результат может быть неправильным;
- если таймер WDT был разрешен, то необходимо очистить регистр WATCHDOG перед выполнением команды перехода в режим POWERDOWN. Это гарантирует, что устройство сможет выйти из режима POWERDOWN «чисто», иначе WDT может сбросить МК до стабилизации тактового генератора. (WDT не может сбросить устройство во время режима POWERDOWN, потому что генератор остановлен).

После завершения этих задач необходимо выполнить команду IDLPD #2 для перехода в режим POWERDOWN. Команда IDLPD #2 выключает внутреннее тактирование и блокирует внутренний генератор.

Примечание – Сигнал EXTINT должен удерживаться на низком уровне, пока устройство находится в режиме POWERDOWN.

19.2.3 Выход из режима POWERDOWN

Выход из режима POWERDOWN обеспечивается одним из трех способов: удержание низкого уровня сигнала на выводе VPR, удержание низкого уровня сигнала EXTINT и сброс всего микроконвертера.

Подключение VPR к низкому уровню

Один из способов выхода из режима POWERDOWN – подать низкий уровень на VPR как минимум на 50 нс. Это разрешит внутреннее фазированное тактирование и разблокирует внутренний генератор. При этом не произойдет сброса МК, а продолжится выполнение программы с команды, следующей за командой перехода в режим POWERDOWN.

Установка RESET#

Другой путь для выхода из режима POWERDOWN – установить сигнал RESET#. RESET# должен удерживаться в низком уровне 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. При этом способе выхода произойдет сброс всего контроллера и начнется выполнение программы с начального адреса.

Установка EXTINT

Последний способ для выхода из режима POWERDOWN – установка сигнала EXTINT не менее чем на 50 нс. EXTINT – обычно вход, фиксирующий прерывание. Однако в режиме POWERDOWN он используется как вход, чувствительный к уровню. Бит EXTINT_SRC (IOC1.1) определяет источник EXTINT. Установка IOC1.1 выбирает P0.7 как источник; очистка IOC1.1 выбирает P2.2 как источник. При этом способе выхода не происходит сброса МК, а продолжается выполнение программы с команды, следующей за командой перехода в режим POWERDOWN.

19.3 Режим SLOW

Микроконтроллер имеет дополнительный режим уменьшения энергопотребления, при котором внутренняя частота уменьшается в два раза и длительность машинного цикла становится равной черырем тактам сигнала XTAL. Это требует пересчета временных соотношений для периферийных устройств, длительности всех сигналов увеличатся ровно в два раза. Перевод ИС в данный режим производится установкой бита CLKCL.7 управляющего регистра CLKCL. Вернуться в нормальный режим можно, сбросив этот бит. 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 IDLE и POWERDOWN режимы, 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 ONCE является режимом тестирования, который электрически изолирует ИС от других устройств на печатной плате.

Все выводы переводятся в режим высокого сопротивления (кроме квазидвунаправленных выводов, которые переводятся в режим слабой единицы), а схема – в режим POWERDOWN. RESET# должен оставаться в состоянии высокого уровня во время режима ONCE.

Вход в режим происходит, если P2.0 удерживается в состоянии низкого уровня во время нарастающего фронта сигнала RESET#.

Выход из режима осуществляется подачей сигнала RESET#. Вывод P2.0 при этом может быть неподключенным или иметь высокий уровень.

Для уменьшения потребления в режиме ONCE одновременно с электрической изоляцией происходит прекращение внутреннего тактирования и выключение генератора (включается режим POWERDOWN). Но при этом 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 высоком уровне, а P2.2 – в низком.

20 Интерфейс внешней памяти

Микроконтроллер 1874ВЕ96Т может взаимодействовать с различными устройствами памяти. Он поддерживает или фиксированную шину разрядностью 8 бит, или динамическую шину разрядностью 8/16 бит. Существует внутренний контроль готовности для медленных внешних устройств памяти READY и несколько режимов управления шиной. Эта особенность обеспечивает большую гибкость, когда идет взаимодействие с устройствами внешней памяти. В данном разделе описываются сигналы внешней памяти и регистры, которые управляют интерфейсом с внешней памятью, а также дана информация о различных режимах и особенностях внешней шины.

20.1 Сигналы интерфейса внешней памяти

В таблице 20.1 описаны сигналы интерфейса внешней памяти. Многие из этих сигналов имеют альтернативные функции. В графе «Выбор» показаны биты регистра и значения, которые выбирают функцию из первой графы.

Таблица 20.1 – Сигналы интерфейса внешней памяти

Имя сигнала	Альтерна-тивная функция	Выбор	Тип	Описание
1	2	3	4	5
AD0 – AD7 AD8 – D15	P3.0 – P3.7 P4.0 – P4.7	Шина доступа к внешней памяти	I/O	Системная шина: адрес/данные. Эти выводы обеспечивают мультиплексированную шину адрес/данные. В течение адресной фазы шинного цикла адресные биты 0 – 15 выводятся на шину и могут быть зафиксированы с помощью ALE или ADV#. 8- или 16-битные данные передаются в течение определенной фазы. Когда доступ к шине прекращается, эти выводы возвращаются к первоначальным функциям порта ввода-вывода
ADV#	ALE	CCR.3 = 0	O	Адрес верен. Выходной сигнал становится активным только в течение доступа к внешней памяти (активный уровень низкий). ADV# показывает, что адрес зафиксирован на системной шине адрес/данные. Этот сигнал сохраняет низкий уровень до тех пор, пока шинный цикл продолжается, и возвращается в высокий уровень, как только шинный цикл завершен. Внешняя защелка может использовать этот сигнал для выделения адреса на шине адрес/данные. Дешифратор может использовать этот сигнал для выработки сигнала «выбор кристалла», используемого внешней памятью
BHE#	WRH#	CCR.2 = 1	O	Разрешение записи старшего байта. Это выходной сигнал с активным низким уровнем, устанавливающийся только при записи слова или старшего байта во внешнюю память. Чтобы выбрать записываемый байт памяти, старший байт используется вместе с A0 = «1»

Продолжение таблицы 20.1

1	2	3	4	5
ALE	ADV#	CCR.3 = 1	O	Разрешение фиксации адреса. Выходной сигнал становится активным только в течение доступа к внешней памяти. ALE показывает, что значение адресной информации этого сигнала действительно находится на системной шине адрес/данные. ALE отличается от ADV# тем, что не остается активным в течение всего шинного цикла. Внешняя защелка может использовать этот сигнал для выделения адреса с шины адрес/данные
BW	–	CCR.1 = 1	I	Разрядность внешней шины. Если CCR.1 = «1», этот сигнал управляет разрядностью шины во время внешнего доступа. Когда BW высокий, выбирается 16-битная шина, когда низкий – 8-битная. Если CCR.1 = «0», разрядность шины всегда 8 бит, при этом данный сигнал игнорируется
EA#	–	–	I	Доступ к внешней памяти. Этот сигнал с активным низким уровнем разрешает доступ к памяти вне кристалла. Если уровень высокий, это выбирает внутрикристальная EEPROM
INST	–	–	O	Выборка команды. Этот сигнал действителен только в цикле чтения внешней памяти. Если уровень сигнала высокий, INST показывает, что выбирается команда. Если сигнал низкого уровня – считываются данные. INST может использоваться в устройствах, которые требуют разделения памяти для байтов данных и команд. Примечание – ССВ-байт и вектор прерывания рассматриваются в качестве данных
RD#	–	–	O	Внешнее чтение. Это выходной сигнал с активным низким уровнем, показывающий чтение внешней памяти
READY	–	–	I	Готовность. Это входной сигнал с активным высоким уровнем, показывающий, что внешняя память готова передавать или принимать данные
WRL#	WR#	CCR.2 = 1	O	Внешняя запись. Это выходной сигнал с активным низким уровнем, показывающий, что идет запись во внешнюю память
WRH#	BHE#	CCR.2 = 0	O	Запись старшего байта. При 16-битной разрядности шины WRH# устанавливается при записи старшего байта и слова. При 8-битной разрядности (CCR.1 = «0») – при записи старшего и младшего байтов, а также слова

Окончание таблицы 20.1

1	2	3	4	5
WR#	WRL#	CCR.2 = 0	O	Запись младшего байта. При 16-битной разрядности WRL# устанавливается при записи младшего байта или слова. При 8-битной разрядности (CCR.1 = «1») – при записи старшего и младшего байтов, а также слова

20.2 Регистр конфигурации кристалла

Регистр конфигурации кристалла (CCR) управляет разрядностью шины, формированием сигнала строб записи и сигнала «адрес действителен» с количеством циклов ожидания, которые могут быть вставлены, пока вывод READY удерживается в низком уровне. Это первый байт, считываемый из памяти, после сброса устройства. На рисунке 20.1 показан формат CCR.

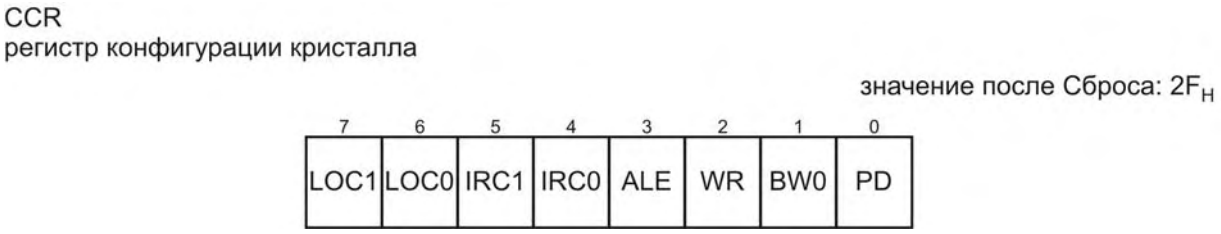


Рисунок 20.1 – Формат регистра CCR

Когда устройство сбрасывается, контроллер шины выбирает Chip Confuration Byte (CCB) из ячейки 2018H и загружает его в CCR. Во время выборки CCB адрес выставляется так же, как и при нормальных операциях. CCB выбирается из внутренней EEPROM, если сигнал EA# высокий, и из внешней памяти, если сигнал низкий. CCR загружается только в течение «последовательности» сброса. Однажды загруженный, CCR не может быть изменен до следующего сброса устройства. Обычно CCB программируется однократно, когда пользовательская программа откомпилирована и не переопределяется во время обычных операций.

Отсутствие загрузки в CCR позволяет сигналам READY и BW управлять временем и разрядностью шины во время выборки CCB. Если CCB хранится в медленной внешней памяти, необходимо установить сигнал READY в низкий уровень. Это приводит к тому, что контроллер шины автоматически вставляет до трех циклов ожидания при выборе CCB (CCR.4 и CCR.5 определяют количество циклов ожидания). Если CCB расположен в 16-битной внешней памяти, управляющий сигнал BW высокий, если в 8-битной внешней памяти – низкий. Легче управлять, если вывод BW в низком уровне (8-битная шина), пока происходит выборка CCB (даже когда в устройстве используется 16-битная внешняя шина). После загрузки CCB в CCR разрядность шины конфигурируется как фиксированная 8-битная или как динамически управляемая с помощью сигнала BW, как определено в CCR.1.

20.3 Разрядность шины и конфигурация памяти

Внешняя шина МК может работать как 8-битная или как 16-битная мультиплексированная шина адрес/данные (рисунок 20.2). Значение CCR.1 определяет разрядность шины как фиксированную 8-битную или как динамическую 16-/8-битную шину, управляемую сигналом BW. Если CCR.1 очищен, контроллер шины видит ее в 8-битном режиме. Когда выполняется код с 8-битной шины, происходит некоторое понижение производительности. Слово считывается и записывается во внешнюю память

с использованием дополнительного шинного цикла и предварительная выборка в очередь не используется полностью.

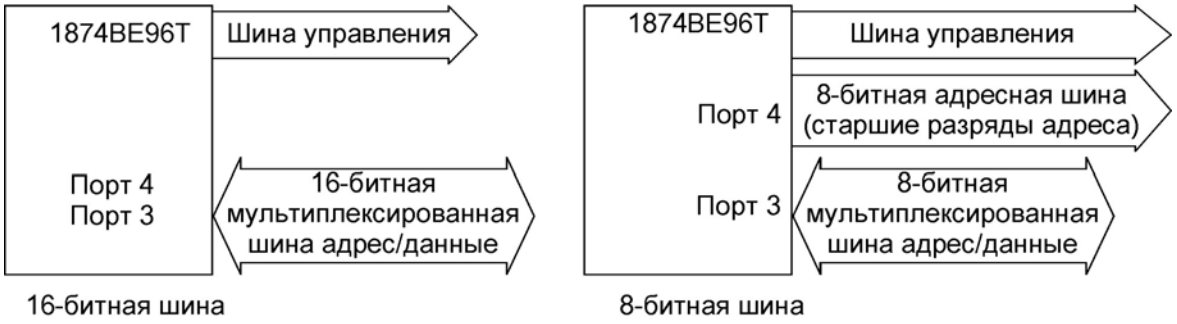


Рисунок 20.2 – Возможные варианты шины адрес/данные

Если установлен CCR.1, то сигнал BW управляет разрядностью шины. Когда BW высокий, шина имеет разрядность 16 бит, и разрядность 8 бит, когда BW имеет низкий уровень. Сигнал BW может использоваться в различных применениях. Например, система может хранить в 16-битных устройствах памяти код программы, а в 8-битных устройствах памяти – данные. Сигнал BW может быть связан с входом выборки кристалла у 8-битных устройств памяти. Когда BW имеет низкий уровень, разрешается 8-битный режим шины, и выбирается 8-битное устройство памяти. Когда BW высокий, то разрешается 16-битный режим шины и не выбирается 8-битное устройство памяти. Переключение сигнала BW нужно производить во время активного уровня сигналов RD# или WR#.

20.3.1 Шина разрядностью 16 бит

Когда контроллер сконфигурирован для работы в режиме 16-битной шины (т. е. CCR.1 установлен, и сигнал BW имеет высокий уровень), линии AD0 – AD15 формируют 16-битную мультиплексированную шину адрес/данные, шина адрес/данные разделяет выводы портов 3 и 4 (таблица 20.1). На рисунке 20.3 показаны упрощенные временные диаграммы для внешних циклов записи и чтения. Следует учитывать характеристики деталей, которые приводятся в спецификациях устройств внешней памяти.

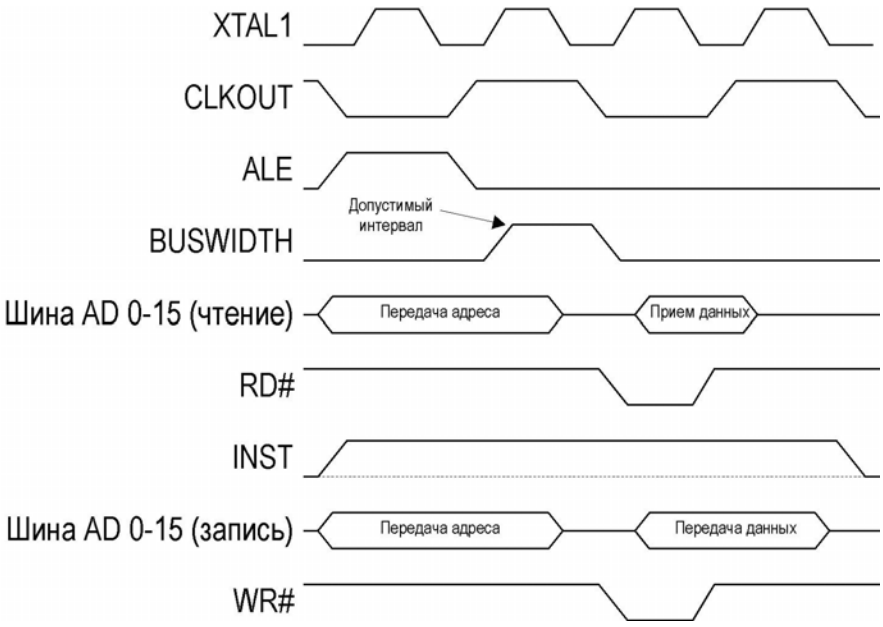


Рисунок 20.3 – Упрощенные временные диаграммы обмена по 16-битной внешней шине

Address Latch Enable (ALE) выделяет адрес на шине, стробируя регистр-защелку (такую как КР1533ИР22). Адрес («AddressOut») будет верным на шине после перехода сигнала ALE из состояния высокого уровня в низкий.

16-битный цикл чтения

Контроллер шины переключает шину на ввод и затем вырабатывает сигнал RD# (переводит в низкий уровень), таким образом, он может принимать данные. Внешняя память должна поместить данные на шину после спадающего фронта сигнала RD#. Установка сигнала INST показывает, что идет выборка команды.

16-битный цикл записи

Контроллер шины переводит сигнал WR# в низкий уровень, затем помещает данные на шину. Спадающий фронт сигнала показывает, что данные на шине стабильны. В это время внешняя система должна зафиксировать данные.

20.3.2 Шина разрядностью 8 бит

Когда МК сконфигурирован для работы в режиме 8-битной шины, линии AD0-AD7 формируют мультиплексированную шину, младший адрес и данные. Линии AD8-AD15 не мультиплексируются. Старшие адреса защелкиваются и сохраняют свое значение в течение всего шинного цикла. На рисунке 20.4 показана упрощенная временная диаграмма для циклов внешнего чтения и записи.

Сигнал ALE используется для выделения младшего адреса, стробируя регистр-защелку (КР1533ИР22).

Цикл чтения на 8-битной шине

После сброса сигнала ALE контроллер шины переключает шину на ввод и переводит сигнал RD# в низкий уровень. Затем внешняя память должна поместить данные на шину. Эти данные должны быть стабильными после нарастающего фронта сигнала RD#. Чтобы считать слово данных, контроллер шины осуществляет два последовательных чтения, читая первым младший байт, а затем старший байт.

Цикл записи на 8-битной шине

После сброса сигнала ALE контроллер шины выдает данные на AD0 – AD7 и затем устанавливает сигнал WR# в низкий уровень. Внешняя память должна зафиксировать данные до момента перехода сигнала WR# в высокий уровень. Эти данные должны удерживаться на шине после перехода сигнала WR# в высокий уровень. Чтобы записать слово данных, контроллер шины осуществляет два последовательных цикла записи, записывая первым младший байт, а затем старший байт.

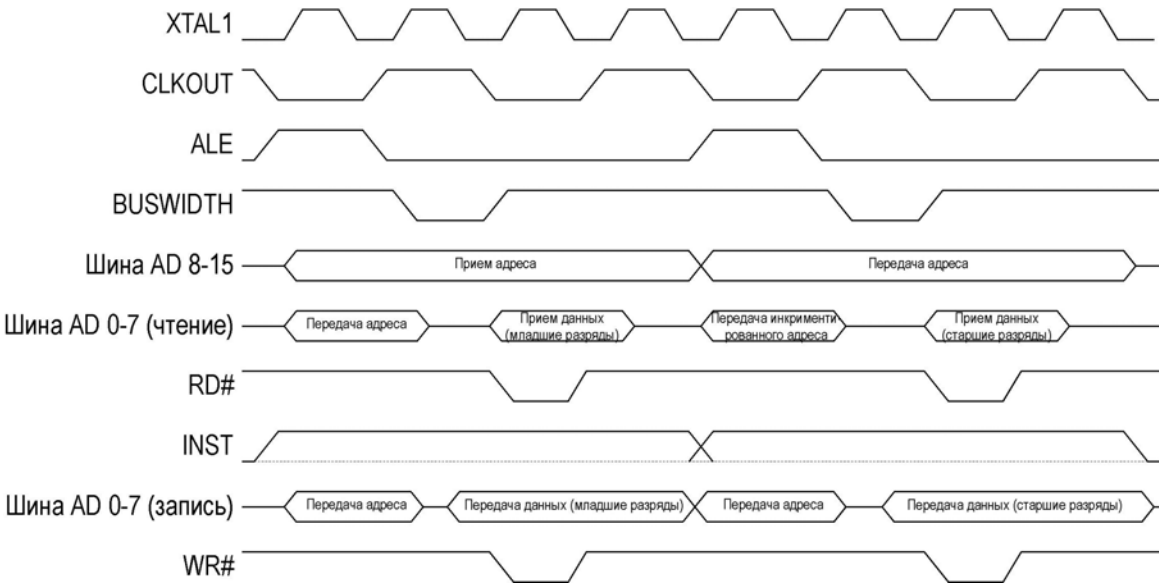


Рисунок 20.4 – Упрощенные временные диаграммы обмена по 8-битной внешней шине

20.4 Управление сигналом READY

Внутренняя схема управления готовностью позволяет для низкоскоростных устройств внешней памяти увеличивать длину шинного цикла записи и чтения у МК. Если устройство внешней памяти не готово для доступа, то необходимо установить сигнал READY в низкий уровень и удерживать его так до тех пор, пока устройство памяти не будет готово завершить операцию. До тех пор, пока сигнал READY низкий, контроллер шины вносит состояния ожидания в шинный цикл. CCR биты 4 и 5 (IRC0 и IRC1) определяют максимальное количество состояний ожиданий, которые могут быть внесены (таблица 20.2). Когда установлены оба бита, контроллер шины вносит состояния ожидания до того времени, пока внешняя память удерживает сигнал READY низким. Однако контроллер шины находится в состоянии ожидания до того момента, пока устройство внешней памяти удерживает сигнал READY, или количество состояний ожидания ограничено числом циклов (1, 2 или 3), определяемым CCR.4 и CCR.5.

Таблица 20.2 – Управление периодами ожидания

CCR.5	CCR.4	Максимальное число тактов ожидания
0	0	Ограничено одним
0	1	Ограничено двумя
1	0	Ограничено тремя
1	1	Не определено (управляется сигналом READY)

При использовании сигнала READY для управления состояниями ожидания необходимо добавить внешнее аппаратное обеспечение для счета состояний ожидания и задержки сигнала READY в течение определенного времени (смотрите в спецификации характеристики для сигнала READY). Следует учитывать, что неисправные устройства памяти могут занимать шину адрес/данные неопределённо долго.

Примечание – Управление готовностью действует только для внешней памяти. Нет возможности добавить состояния ожидания, когда идет доступ к внутренней ОЗУ или EEPROM области.

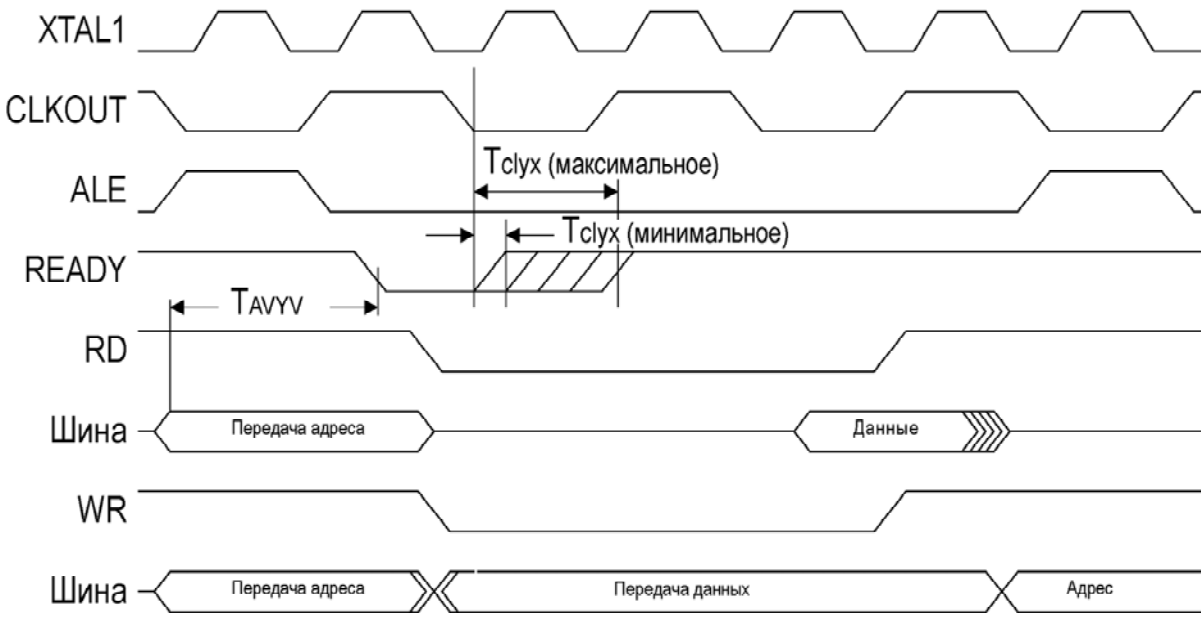


Рисунок 20.5 – Временные диаграммы для сигнала READY

20.4.1 Временные требования для сигнала READY

При использовании сигнала READY для ввода состояний ожиданий в цикл шины время установки и удержания должны быть определенными (рисунок 20.5). Так как используется дешифрированное значение адреса для выработки сигнала READY, время установки определяется от момента установки адреса на шине. Значение T_{AVYV} определяет количество времени, имеющееся для декодирования и формирования READY от момента, когда адрес уже установлен. Сигнал READY должен удерживать свое значение до тех пор, пока не кончится время Tclyx. Обычно это минимум – 0 нс с момента перехода сигнала CLKOUT в низкий уровень. Нельзя превышать максимальное время Tclyx, иначе появляются дополнительные (нежелательные) циклы ожидания.

Когда для ограничения циклов ожидания (до 1, 2 или 3 максимум) программируется CCR, можно связать прямо сигнал READY с разрешением кристалла в устройстве внешней памяти. Это простейший путь для добавления 1 – 3 циклов ожидания при выборе устройства.

Примечание – Ранние HMOS устройства использовали время установки сигнала READY, которое определялось спадающим фронтом сигнала ALE (Tllyv). Этот параметр не критичен для CMOS устройств, которые используют внутренний двухфазный тактовый сигнал. Он приводится только для сравнения.

20.5 Режимы управления шиной

CCR.2 и CCR.3 определяют, какие сигналы управления шиной будут генерироваться во время внешнего цикла чтения или записи. В таблице 20.3 показаны четыре режима управления шиной и установки CCR.3 и CCR.4 для каждого из них.

Таблица 20.3 – Режимы управления шиной

Режим управления шиной	Сигналы управления шиной	CCR.3	CCR.2
Adress Valid with Write Strobe	ADV#, RD#, WRL#, WRH#	0	0
Adress Valid Strobe	ADV#, RD#, WR#, BHE#	0	1
Write Strobe	ALE, RD#, WRL#, WRH#	1	0
Стандартный режим управления	ALE, RD#, WR#, BHE#	1	1

20.5.1 Стандартный режим управления шиной (Standart Bus Control)

В стандартном режиме управления шиной МК вырабатывает стандартные сигналы управления шиной: ALE, WR#, RD# и BHE# (рисунок 20.6). ALE устанавливается при установке адреса и может использоваться для фиксации адреса снаружи. Низкий BHE# выбирает банк памяти, который адресуется старшим байтом на шине данных. RD# устанавливается для чтения внешней памяти, и WR# устанавливается для записи во внешнюю память.

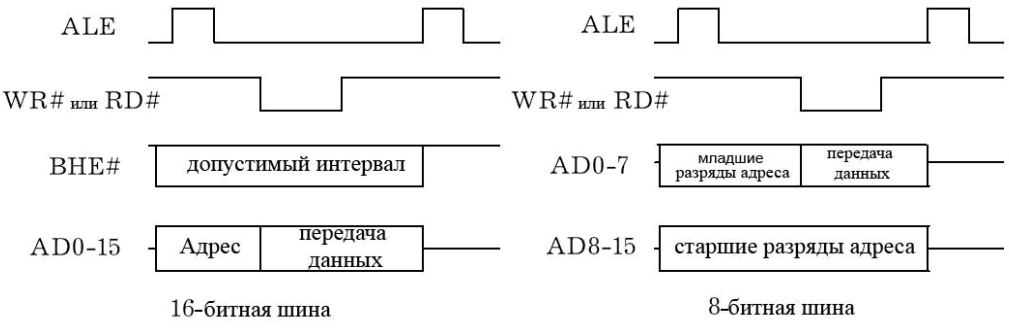


Рисунок 20.6 – Стандартное управление шиной

Когда сконфигурировано устройство для использования 16-битной шины, сигналы, разделяющие запись младшего и старшего байтов, должны быть сгенерированы для записи единственного байта. На рисунке 20.7 показана простейшая цепь, которая комбинирует BHE# и A0 для обеспечения этих сигналов (WRL# и WRH#). Схожая пара сигналов для чтения необязательна. Для чтения одного байта с 16-битной шины оба байта выставляются на шину данных, и процессор отбрасывает ненужный байт.

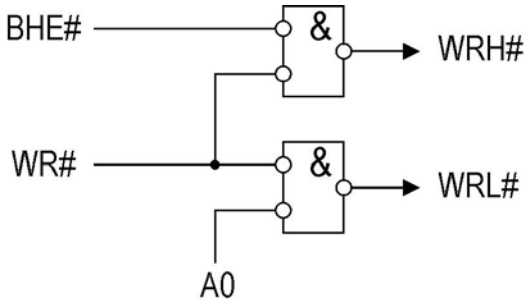


Рисунок 20.7 – Дешифрация сигналов WRL# и WRH#

На рисунке 20.8 показана 8-битная система с EPROM и RAM. EPROM расположен в нижней части памяти, а RAM находится в верхней части. Эта система конфигурируется с использованием старшего значащего бита адреса (A15) как сигнал выбора кристалла и ALE – как сигнал защелки адреса.

На рисунке 20.9 показана система, которая использует динамическую разрядность шины (установлен CCR.1). Код исполняется из двух EPROM, и данные сохраняются в RAM разрядностью в один байт. RAM расположена в старшей памяти и выбирается, если A15 имеет высокий уровень; RAM также выбирается в режиме 8-битной шины под управлением сигнала BW, который имеет низкий уровень.

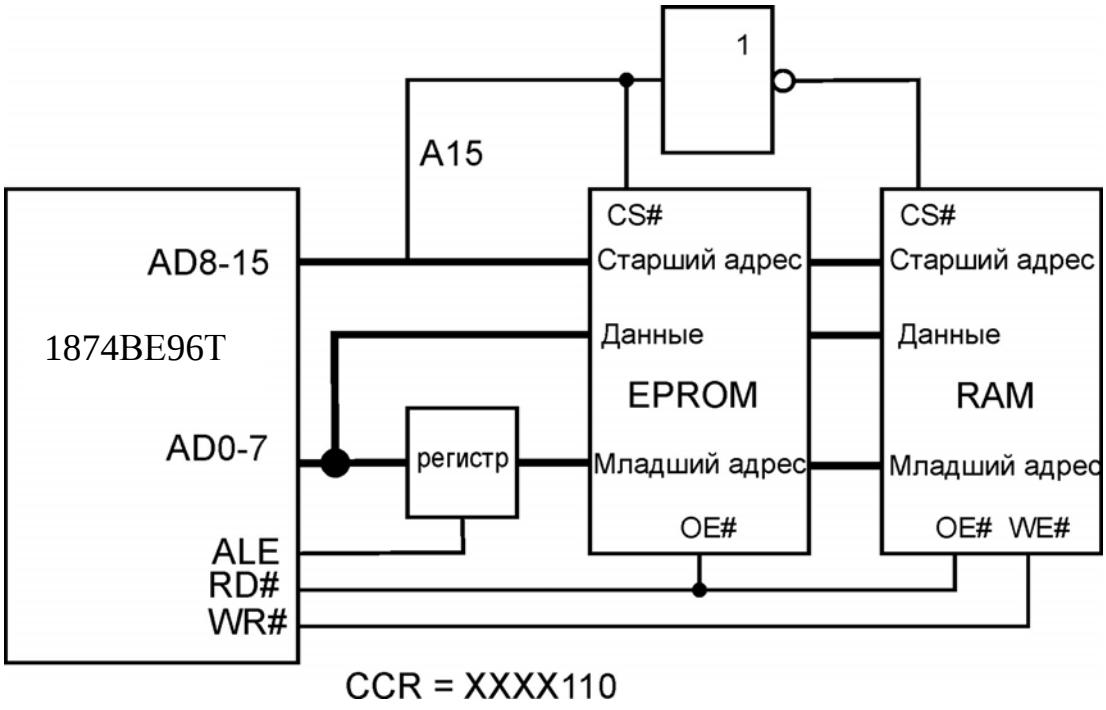


Рисунок 20.8 – Пример 8-битной системы с EPROM и RAM

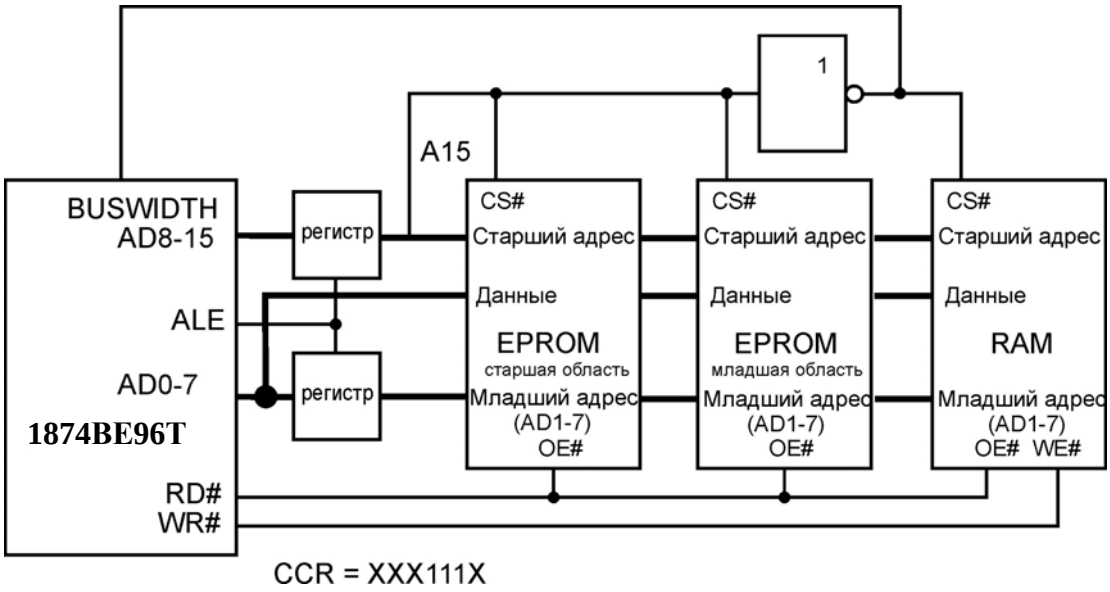


Рисунок 20.9 – Пример 16-битной системы с изменяемой разрядностью шины

20.5.2 Режим Write Strobe

Режим Write Strobe устраняет необходимость во внешней дешифрации при записи старшего и младшего байтов во внешнюю 16-битную RAM в 16-битном режиме. При выборе режима Write Strobe вырабатывается WRL# и WRH# взамен WR# и BHE#. WRL# устанавливается для всех записей младших байтов (четные адреса) и всех записей слов. WRH# устанавливается при записи старшего байта (нечетный адрес) и записи слова. В 8-битном режиме WRH# и WRL# устанавливаются для четных и нечетных адресов. Временные диаграммы режима Write Strobe показаны на рисунке 20.10.

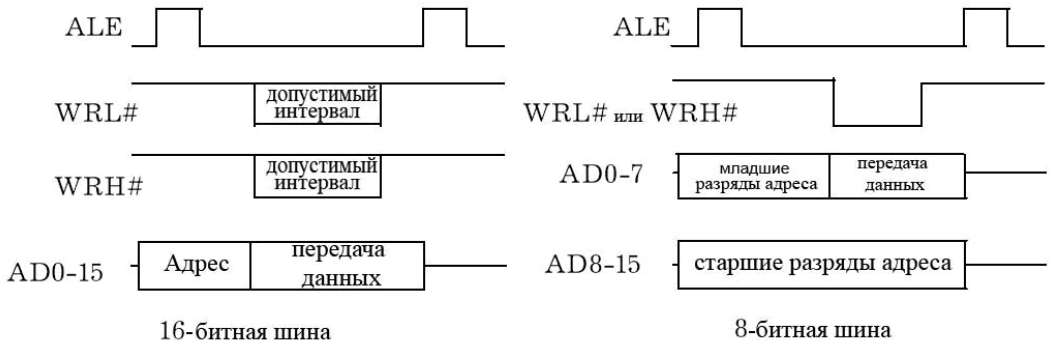


Рисунок 20.10 – Режим Write Strobe

На рисунке 20.11 показана 16-битная система, которая использует два EPROM и два RAM. Это пример конфигурации для использования режима Write Strobe. ALE фиксирует адрес, и A15 служит для выбора кристалла для EPROM и RAM. WRL# устанавливается в момент записи младшего байта и слова. WRH# устанавливается в момент записи старшего байта и слова. При этом устройство RAM не использует A0, WRL# и WRH#, определяющие выбор младшего (A0 = 0) или старшего (A0 = 1) байтов.

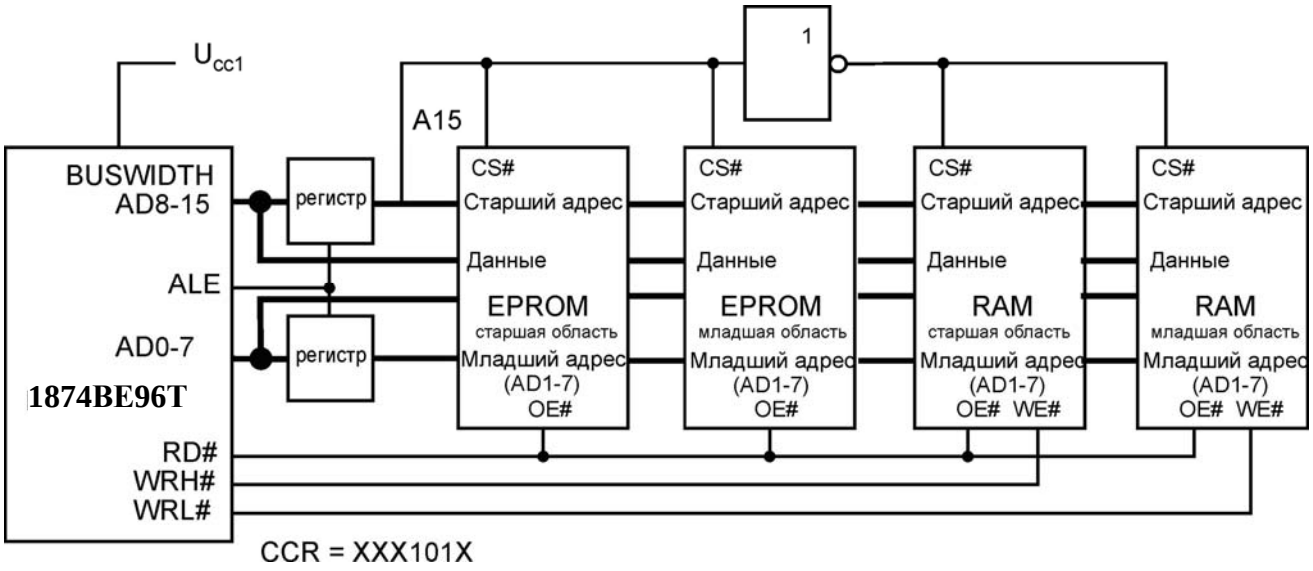


Рисунок 20.11 – Пример 16-битной системы с однобайтной записью в RAM

20.5.3 Режим Address Valid Strobe

Когда выбран Address Valid Strobe режим, МК вырабатывает сигнал Address Valid (ADV#) взамен сигнала Address Latch Enable (ALE). ADV# устанавливается после того, как установился адрес. Этот сигнал можно использовать для фиксации значения адреса и одновременно для разрешения устройства внешней памяти (рисунок 20.12).

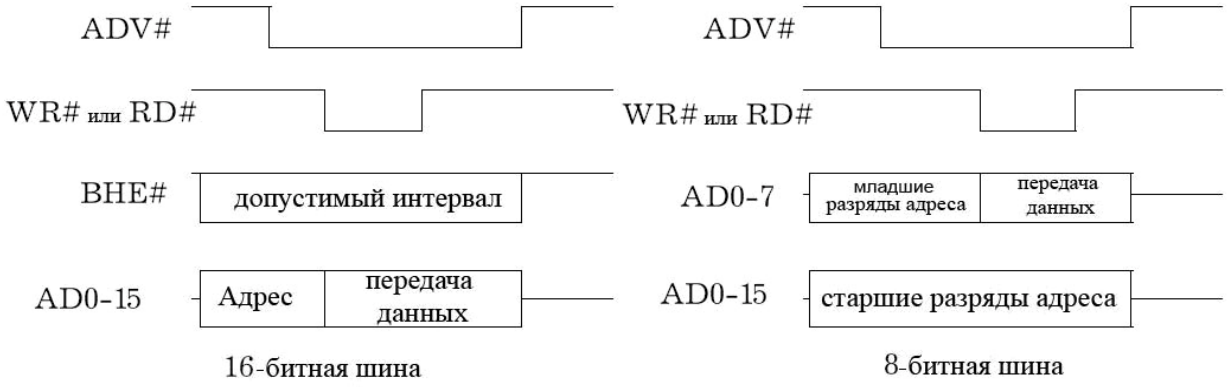


Рисунок 20.12 – Режим Address Valid Strobe

Различие между ALE и ADV# заключается в том, что ADV# устанавливается для полного цикла шины, а не для точной фиксации адреса. На рисунке 20.13 показано различие между ALE и ADV# для одного цикла чтения или записи. Примечательно, что когда циклы обмена с памятью следуют друг за другом (back-to-back), действие ADV# выглядит идентично действию ALE. Различие становится явным, только когда шина не занята. Из-за того, что ADV# имеет высокий уровень в течение этих периодов, внешняя память будет запрещена, таким образом сохраняется и экономится энергия.

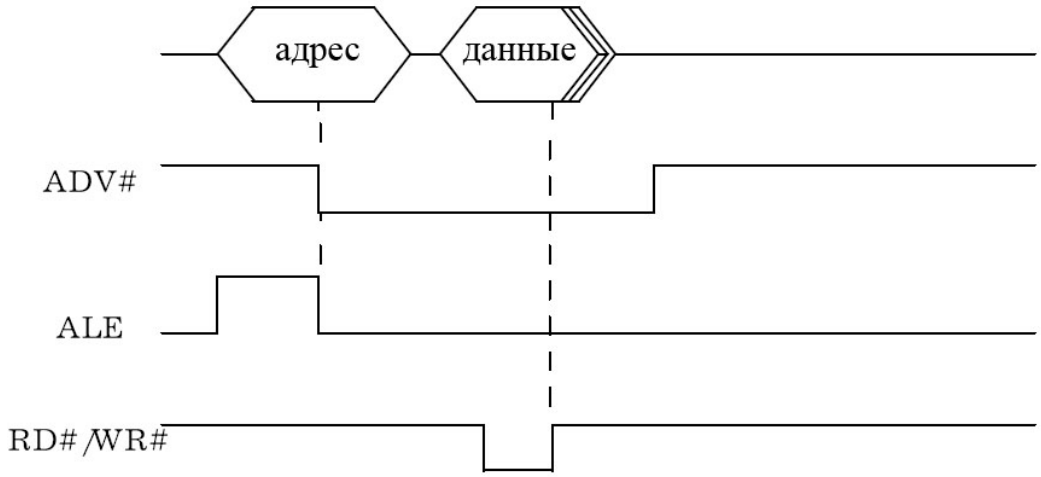


Рисунок 20.13 – Сравнение шинных циклов ALE и ADV#

На рисунках 20.14 и 20.15 показана простейшая схема, которая использует режим Address Valid Strobe. На рисунке 20.14 показана простая 8-битная система с одной EPROM для режима Address Valid Strobe. Такая конфигурация системы использует сигнал ADV# дважды: как сигнал выбора кристалла EPROM и как сигнал защелки адреса.

На рисунке 20.15 показана 16-битная система с двумя EPROM. Эта конфигурация системы использует сигнал ADV# дважды как сигнал выбора кристалла EPROM и сигнал защелки адреса.

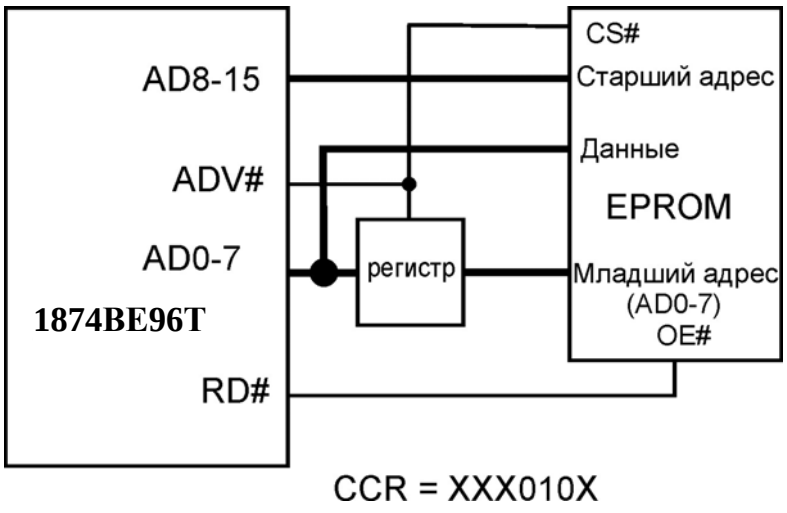


Рисунок 20.14 – 8-битная система с EPROM

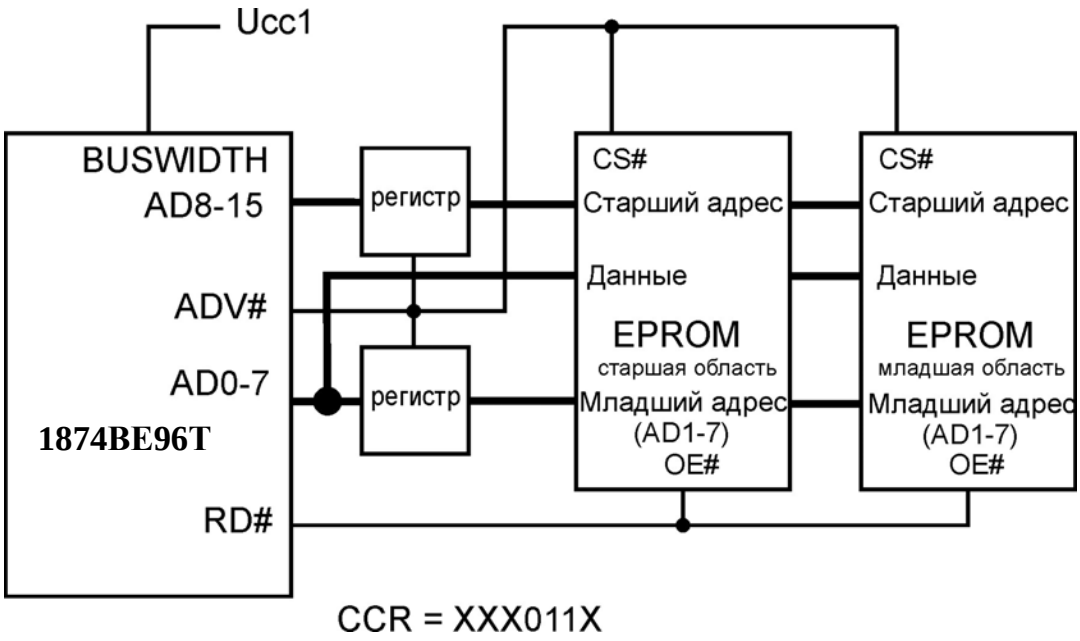


Рисунок 20.15 – 16-битная система с EPROM

20.5.4 Режим Address Valid with Write Strobe

Когда выбран режим Address Valid with Write Strobe, МК вырабатывает сигналы управления шиной ADV#, WRL#, WRH#. Этот режим применим в простой системе, использующей внешнюю 16-битную RAM. На рисунке 20.16 показаны временные соотношения. Сигнал RD#, не описанный выше, аналогичен WRL#, WRH# и WR#. Пример системы, использующей режим Address Valid with Write Strobe, приведен на рисунке 20.17.

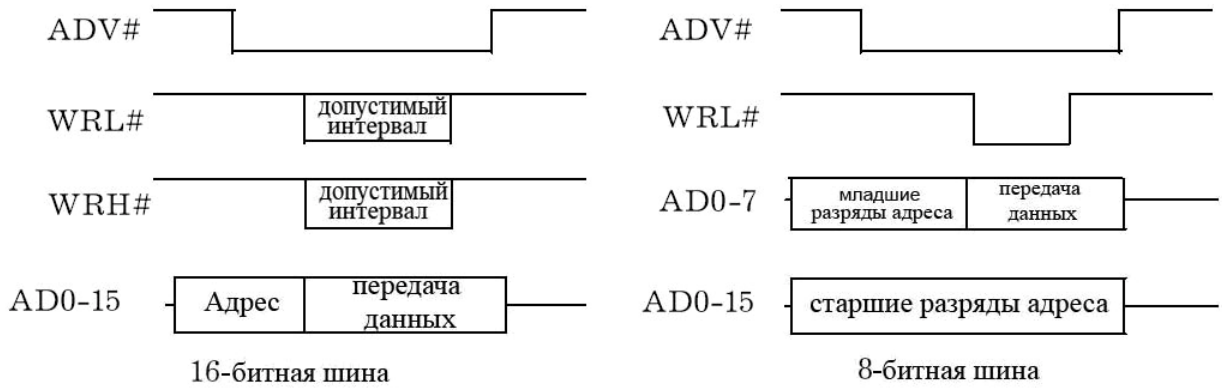


Рисунок 20.16 – Временные соотношения для режима Address Valid with Write Strobe

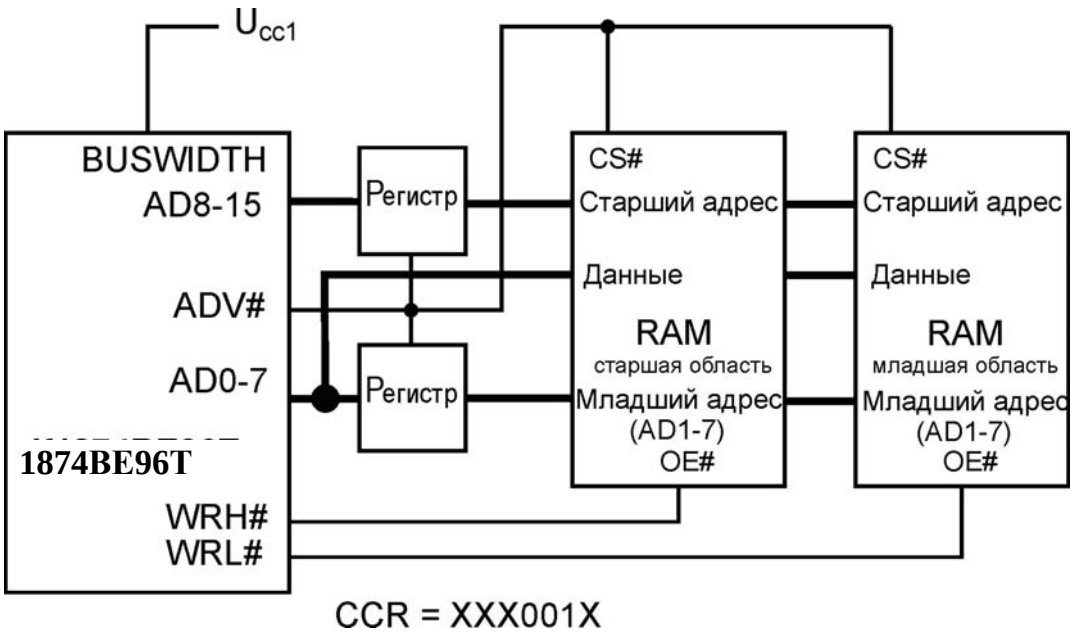


Рисунок 20.17 – 16-битная система с RAM

20.6 Временные соотношения системной шины

Большинство системных шин имеют временные соотношения, показанные на рисунке 20.18 и описанные в таблицах 20.4, 20.5.

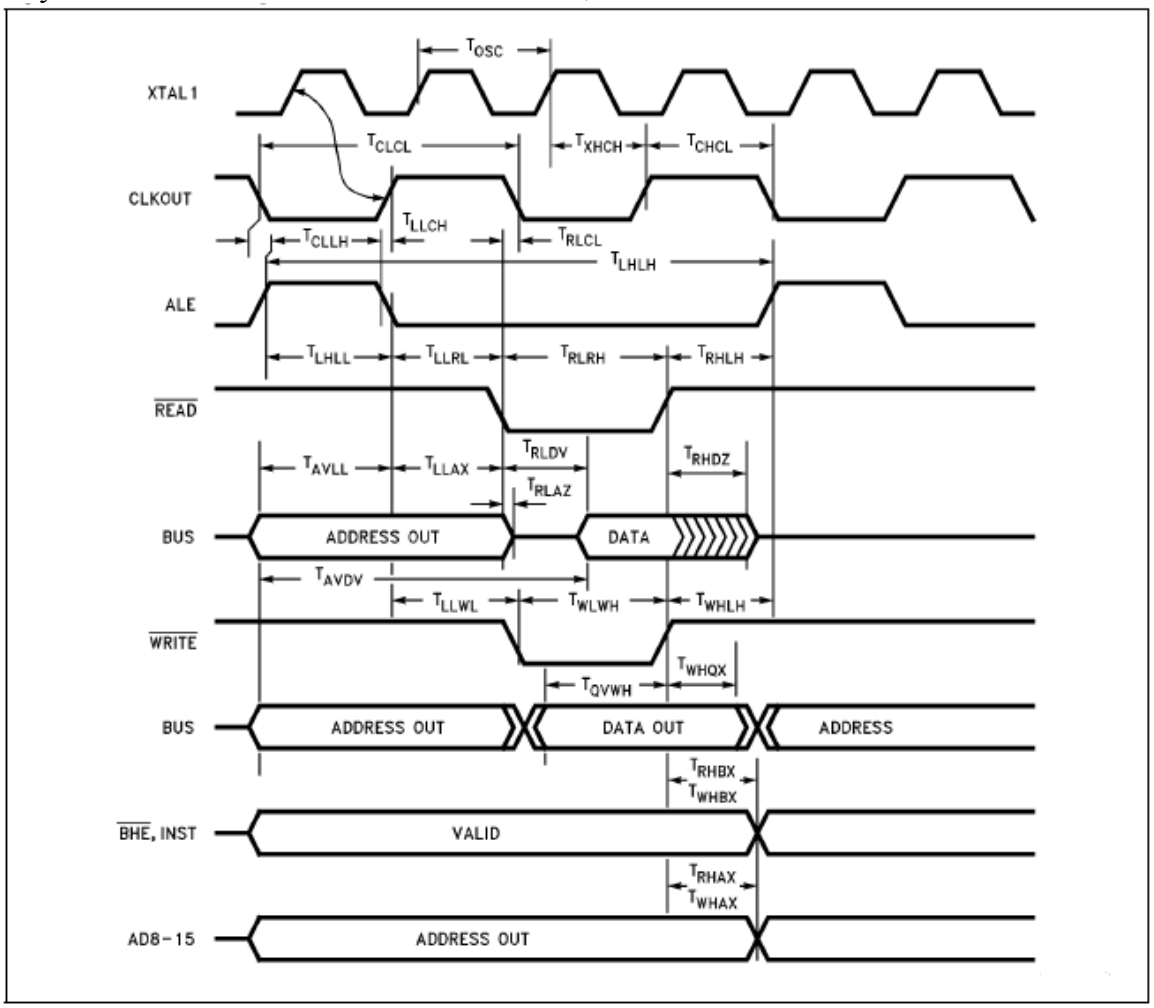


Рисунок 20.18 – Временная диаграмма системной шины

Перечень принятых сокращений

Каждый символ имеет два буквенных префикса у «Т» (для времени). Характеристика показывает сигнал и условия его появления. Символы отображают время между двумя точками сигнал/условие.

Условия:

H – высокий уровень
L – низкий уровень
V – значение
X – неопределённое значение
Z – неподключенный

A – адрес
B – BHE#
C – CLKOUT
D – данные
G – BW

Сигналы:

L – ALE/ADV#
W – WR#/WRH#/WRL#
Q – выходные данные
R – RD#
X – XTALI
Y – READY

Таблица 20.4 – Описание временных соотношений

Параметр	Описание
1	2
T_{AVYV}	Время от установления Address до установления READY: максимальное время для системы памяти, устанавливающей сигнал READY после вывода адреса из МК и гарантирующей один цикл ожидания
T_{LLYH} *	Установление READY после перехода ALE в низкий уровень: максимальное время для системы памяти, устанавливающей сигнал READY после сброса ALE, гарантирует введение по крайней мере одного цикла ожидания
T_{CLYX}	Сохранение READY после сброса CLKOUT: минимальное время сохранения всегда 0 нс
T_{LLYX}	Сохранение READY после сброса ALE: минимальное время удержания сигнала READY после сброса ALE. Если максимальное значение превышается, дополнительно вводятся циклы ожидания
T_{AVGV}	Address действителен и BW активен. Максимальное время между установлением адреса и появлением BW. Если это требование превышается, МК может не ответить соответствующим шинным циклом
T_{LLGV}	Сброс ALE и BW активен: максимальное время между сбросом ALE/ADV# и появлением BW. Если это требование превышается, МК может не ответить соответствующим шинным циклом
T_{CLGX}	BW HOLD после сброса CLKOUT: минимальное время, необходимое для удержания BW, после сброса CLKOUT
T_{AVDV}	Адрес и входные данные действительны: максимальное время, которое имеют устройства памяти для выдачи данных после установления адреса
T_{RLDV}	Сброс RD# и входные данные действительны: максимальное время для выдачи данных системой памяти после установления сигнала RD#
T_{CLDV}	Сброс CLKOUT и входные данные действительны: максимальное время удержания выходных данных системы памяти после сброса CLKOUT

Продолжение таблицы 20.4

1	2
T_{RHDZ}	RD# High и “плавающие” входные данные: максимальное время перехода RD# в неактивное состояние, пока система памяти освобождает шину. Если это время не учитывается, возможен конфликт на шине
T_{RHDX}	Удержание данных после сброса RD#: время после сброса RD# в неактивное состояние, в течение которого система памяти удерживает данные на шине
F_{XTAL}	Частота на XTAL: частота входного сигнала на входе XTAL1. Скорость работы внутренней шины в устройстве равна 1/2 Fxtal
T_{OSC}	1/Fxtal: все временные соотношения относятся и к T_{OSC}
T_{XHCH}	XTAL1 высокий для CLKOUT (высокого или низкого)
T_{CLCL}	CLKOUT временной цикл: нормально $2T_{OSC}$
T_{CHCL}	Период высокого CLKOUT: необходим в системе, использующей CLKOUT как тактовый сигнал для внешних устройств
T_{CLLH}	Время от сброса CLKOUT до установки ALE/ADV#. Используется для установления других временных соотношений
T_{LLCH}	Время от спадающего фронта ALE/ADV# до нарастающего фронта CLKOUT, используется для определения других временных соотношений
T_{LHLH}	Время ALE цикла: минимальное время между импульсами ALE
T_{LHLL}	Длительность высокого уровня ALE/ADV#. Этот параметр используется при разработке внешней защелки адреса
T_{AVLL}	Установка адреса относительно заднего фронта сигнала ALE/ADV#. Время, необходимое для фиксации адреса до заднего фронта сигнала ALE/ADV#. Этот параметр используется при разработке внешней защелки адреса
T_{LLAX}	Удержание адреса после заднего фронта сигнала ALE/ADV#. Время удержания адреса после заднего фронта сигнала ALE/ADV#. Этот параметр используется при разработке внешней защелки адреса
T_{LLRL}	Задержка перехода сигнала RD# в низкий уровень относительно заднего фронта сигнала ALE/ADV#. Время между спадающим фронтом сигнала ALE/ADV# и установкой сигнала RD#. Необходимо для уверенной дешифрации устройствами памяти текущего адреса и активизации микросхем
T_{RLCL}	Переход в низкий уровень сигнала RD# относительно заднего фронта сигнала CLKOUT. Время между задним фронтом сигнала CLKOUT и установкой сигнала RD#
T_{RLRH1}	Длительность импульса сигнала RD#
T_{RHLH}	Время с момента перехода сигнала RD# в неактивное состояние до появления следующего сигнала ALE/ADV#. Используется для вычисления времени с момента снятия адреса до появления следующего установившегося адреса
T_{RLAZ}	Время между освобождением шины адреса и установлением сигнала RD#. Используется для определения момента, когда МК освобождает шину адрес/ данные
T_{LLWL}	Задержка перехода сигнала WR# в низкий уровень относительно заднего фронта сигнала ALE/ADV#. Время между спадающим фронтом сигнала ALE/ADV# и установкой сигнала WR#. Необходимо для уверенной дешифрации устройствами памяти текущего адреса и активизации микросхем
T_{CLWL}	Переход в низкий уровень сигнала WR# относительно заднего фронта сигнала CLKOUT. Время между задним фронтом сигнала CLKOUT и установкой сигнала WR#
T_{QVWH}	Время с момента фиксации данных на шине до перехода сигнала WR# в неактивное состояние. Устройства памяти должны соответствовать этому параметру

Окончание таблицы 20.4

1	2
T_{CHWH}	Время с момента перехода сигнала CLKOUT в состояние с высоким уровнем до перехода сигнала WR# в неактивное состояние
T_{WLWH}	Длительность сигнала WR#
T_{WHQX}	Время удержания данных на шине после перехода сигнала WR# в высокий уровень. Устройства памяти должны соответствовать этому параметру
T_{WHLH}	Время с момента перехода WR# в неактивное состояние до появления следующего сигнала ALE/ADV#. Также используется для вычисления времени между переходом сигнала WR# в неактивное состояние и фиксацией нового адреса на шине
T_{WHBX}	Минимальное время с момента перехода сигнала WR# в высокий уровень, в течение которого сигналы BHE#, INST, HOLD должны сохранять свои значения
T_{RHBX}	Минимальное время с момента перехода сигнала RD# в высокий уровень, в течение которого сигналы BHE#, INST, HOLD должны сохранять свои значения
T_{WHAX}	Минимальное время с момента перехода сигнала WR# в высокий уровень, в течение которого старший байт адреса (AD8 – AD15) в 8-битном режиме будет сохранять свое значение
T_{RHAX}	Минимальное время с момента перехода сигнала RD# в высокий уровень, в течение которого старший байт адреса (AD8 – AD15) в 8-битном режиме будет сохранять свое значение
_____ * Включён только для совместимости с временными соотношениями HMOS устройств.	

Таблица 20.5 – Временные соотношения системной шины

Параметр, нс	Минимальный	Максимальный	Типовой
T_{CLCL}	$2T_{OSC}$		$2T_{OSC}$
T_{CHCL}	T_{OSC}-10	T_{OSC}+15	T_{OSC}+1
T_{CLLH}	-5	+15	+4
T_{LLCH}	-20	+15	+1
T_{LHL}	T_{OSC}-10	T_{OSC}+10	T_{OSC}-5
T_{AVLL}	T_{OSC}-10	T_{OSC}+10	T_{OSC}
T_{LLAX}	T_{OSC}-35		T_{OSC}
T_{LLRL}	T_{OSC}-30		T_{OSC}
T_{RLCL}	-10	+10	+2
T_{RLRH}	T_{OSC}-5		T_{OSC}
T_{RHLH}	T_{OSC}	T_{OSC}+10	T_{OSC}+2
T_{RLAZ}		+5	+2
T_{LLWL}	T_{OSC}-10		T_{OSC}
T_{CLWL}	0	+25	4
T_{QVWH}	T_{OSC}-23		T_{OSC}
T_{CHWH}	-5	+15	5
T_{WLWH}	T_{OSC}-20		T_{OSC}
T_{WHQX}	T_{OSC}-25		T_{OSC}
T_{WHLH}	T_{OSC}-10	T_{OSC}+15	T_{OSC}
T_{WHBX}	T_{OSC}-10		T_{OSC}
T_{WHAX}	T_{OSC}-30		T_{OSC}
T_{RHBX}	T_{OSC}-10		T_{OSC}
T_{RHAX}	T_{OSC}-25		T_{OSC}

21 Программирование постоянной памяти

Микросхема 1874BE96T содержит многократно программируемое запоминающее устройство (типа EEPROM). В микросхеме под ПЗУ отводится 32 Кбайт адресного пространства с 2000H по 9FFFH

Адреса с 9000H по 9FFFH отводятся для программы программирования ПЗУ или для отладочных программ. При удержании входа VPR в низком уровне во время сброса, в основной счетчик команд загружается 9000H вместо 2080H и начинается выполнение программы-монитора, которая реализует режимы программирования, описанные в данном разделе (кроме режима RUN-TIME, поддерживаемого аппаратно). Область, выделенная под эту программу-монитор, не имеет никаких дополнительных ограничений на запись пользователем, поэтому может быть использована им как область для резервного старта или как основная область команд (т. е. можно переписать программу, которая идет по умолчанию). Но в этом случае, пользователь сам должен позаботиться о возможности программирования ИС, расположив свою программу по удобным ему адресам.

Данный раздел предназначен для помощи потребителю в программировании микроконвертеров, которое может осуществляться несколькими способами. Могут использоваться автоматический режим (AUTO), режим с использованием специального программатора (SLAVE), режим программирования во время исполнения программы (RUN-TIME) или режим программирования через последовательный порт 0 (SERIAL). Также можно выбрать режим вывода содержимого памяти (ROM-DUMP) для верификации содержимого ROM. В разделе описаны режимы программирования, их выбор и работа в них.

21.1 Общее описание

Когда выбирается режим программирования (за исключением режима RUN-TIME), устройство отзывается на него выполнением внутреннего алгоритма, который находится в области ПЗУ, начиная с адреса 9000H. В общем случае внутренний алгоритм выполняет функции программирования под руководством внешних источников информации.

Выбор потребителем режима программирования определяет, какой алгоритм выбирается из ПЗУ. Чтобы выполнить программирование правильно, устройство должно иметь следующую минимальную схему включения: XTAL1 подключен, неиспользованные входы связаны с логической единицей или нулем, как этого требует режим, питание и земля подключены. Условия работы, заданные в описании устройства, должны быть выполнены.

В данном разделе рассматриваются требования для каждого режима программирования.

21.2 Режимы программирования

Внутренняя программа МК поддерживает пять способов программирования EEPROM.

Режим автоматического программирования AUTO дает возможность самопрограммирования микроконвертера из внешнего EPROM, без использования специального программатора. Этот режим позволяет потребителю использовать прототип схемы проектируемого устройства для программирования.

Режим программирования SLAVE требует специального программатора. При использовании этого режима потребитель может программировать или верифицировать отдельные слова или наборы слов в EEPROM.

Режим RUN-TIME позволяет программировать отдельные ячейки EEPROM во время обычного выполнения программы полностью под управлением программного обеспечения. В этом случае потребителю не нужно специально устанавливать режим программирования.

Режим программирования через последовательный порт 0 SERIAL. В этом случае управляющие команды и данные поступают через порт UART0.

Режим ROM-DUMP позволяет верифицировать содержимое EEPROM путем записи массива из EEPROM во внешнюю память.

Данные режимы (кроме RUN-TIME) реализуются по умолчанию встроенным программным обеспечением НИИЭТ. Пользователь может переписать это программное обеспечение, реализовав свой алгоритм записи.

Режим RUN-TIME является аппаратным режимом и поддерживается ИС изначально. Его нельзя переписать.

В случае если по каким-либо причинам испортилось встроенное программное обеспечение, есть возможность альтернативного старта с адреса 0A000H (всегда внешний адрес) для перепрограммирования. В этом случае алгоритм перезаписи ПЗУ должен содержаться во внешней памяти.

21.2.1 Выбор режима программирования

Для выбора режима программирования используются сигналы PMODE (PMODE.0 – PMODE.3). В таблице 21.1 описаны функции PMODE, указываются значения PMODE и режимы программирования.

Следует ввести МК в режим программирования удержанием сигнала VPR в низком уровне во время нарастания сигнала RESET#.

PMODE выбирает один из пяти режимов (режим RUN-TIME не выбирается). Выводы P0.4 – P0.7 задают шестнадцатеричное значение PMODE. Четыре режима – это SLAVE, ROM-DUMP, AUTO и SERIAL. PMODE фиксируется командами (не аппаратно) в самом начале программы-монитора и должен быть стабилен некоторое время после выполнения сброса. Для переключения режимов необходимо сбросить устройство.

Таблица 21.1 – Описание функций PMODE и его значений

Значения PMODE	Режим программирования
0H – 4H	Зарезервировано
5H	SLAVE
6H	ROM-DUMP
7H – 0BH	Зарезервировано
0CH	AUTO
0DH	Зарезервировано
0EH	SERIAL

21.3 Функции выводов в режиме программирования

Чтобы обслуживать различные режимы программирования, некоторые выводы приобретают новые функции. В таблице 21.2 описаны эти новые функции.
Назначение выводов в режиме программирования – на рисунке 21.1.

Таблица 21.2 – Описание функций выводов в режиме программирования

Имя функции	Тип вывода	Режим	Описание
AINC# P2.4	Вход	SLAVE	Автоинкремент. Активный низкий уровень на входе разрешает автоинкремент. Автоинкремент позволяет считывать и записывать в ячейки EEPROM последовательно, без передачи адреса при каждом обмене
CPVER P2.6	Выход	SLAVE	Полная верификация программирования. Высокий уровень на этом выводе в режиме SLAVE показывает, что все ячейки запрограммированы верно
EA#	Вход	Все режимы програм-мирования, исключая RUN-TIME	Внешний доступ. Если на EA# подано «1» во время нарастания сигнала RESET#, а на вход VPR подан «0», устройство входит в режим программирования. EA# фиксируется только при нарастании сигнала RESET#. Изменение уровня на входе EA# после сброса не дает эффекта
PACT# P2.7	Выход	AUTO	Программирование активно. В режиме программирования AUTO низкий уровень на выходе показывает, что идет процесс программирования, а высокий уровень индицирует окончание программирования
PALE# P2.1	Вход	SLAVE	В режиме SLAVE по заднему фронту PALE считывается адрес/команда из портов 3, 4
PBUS Порты 3, 4	Вход/ Выход	Все режимы програм-мирования, исключая RUN-TIME	Шина адрес/команда/ данные. Используется как системная шина для доступа к внешней памяти в режимах ROM-DUMP и автопрограммирования. В режиме SLAVE выводы PBUS требуется подсоединять к #VCC1 через резистор
PMODE P0.4 – P0.7	Вход	Все режимы програм-мирования, исключая RUN-TIME	Выбор режима. Определяет, какой алгоритм выбран для программирования EEPROM. Сигналы PMODE фиксируются после сброса устройства и должны сохранять свое значение во время операции. В таблице 21.1 приведены значения PMODE и режимы программирования, им соответствующие
PROG# P2.2	Вход	SLAVE	Программирование. В режиме SLAVE по спаду сигнала фиксируются данные на PBUS и начинается программирование, по фронту программирование заканчивается
PVER P2.0	Выход	AUTO SLAVE	Верификация программирования. В режимах SLAVE и AUTO высокий уровень сигнала PVER – при успешном программировании ячейки, низкий – при обнаружении ошибки
VPR	Вход	Все режимы	Вход в режимы программирования. Осуществляется подачей низкого напряжения во время нарастания сигнала RESET#

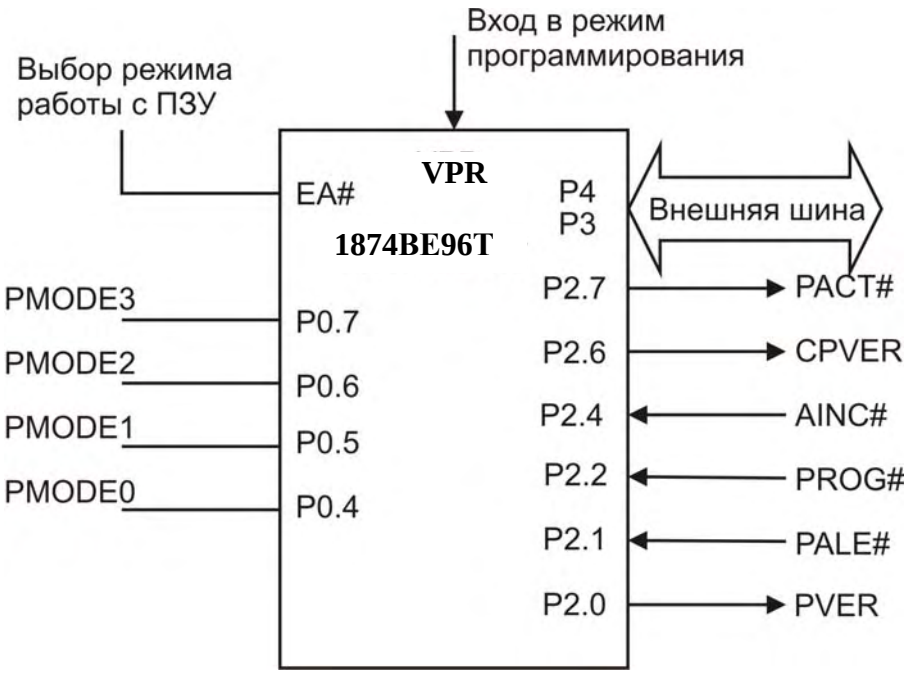


Рисунок 21.1 – Назначение выводов в режиме программирования

21.4 Распределение памяти

Содержимое памяти программ, находящейся по адресам 2080H – 9FFFH, и памяти специального назначения (2000H – 207FH) определяется пользователем. В таблице 21.3 описаны начальные и конечные адреса памяти специального назначения МК шестнадцатеричной и десятичной систем исчисления, на рисунке 21.2 – карта памяти.

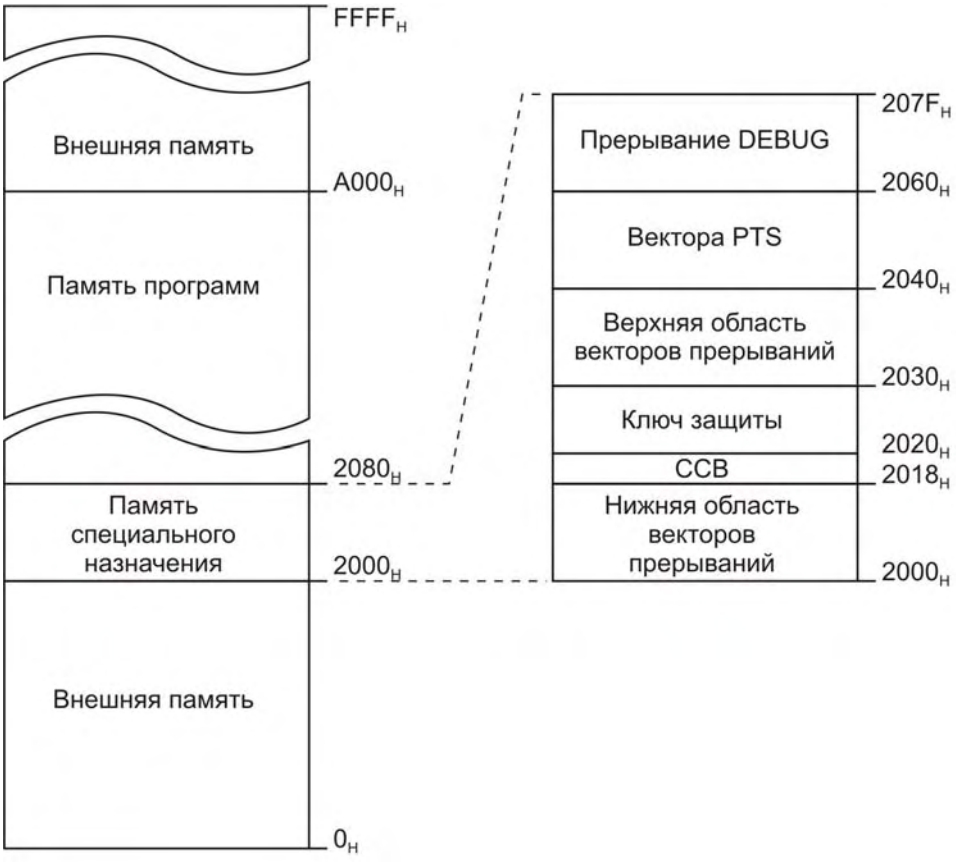


Рисунок 21.2 – Карта памяти специального назначения МК

Таблица 21.3 – Адреса памяти специального назначения МК

Обозначения	Диапазон адресов	
	Шестнадцатеричные	Десятичные
Общего назначения	2062H – 207FH	8290 – 8319
Вектор прерывания DEBUG	2060H	8288
Общего назначения	205EH – 205FH	8286-8287
Векторы PTS	2040H – 205DH	8256 – 8285
Старшие векторы прерывания	2030H – 203FH	8240 – 8255
Общего назначения	2019H – 202FH	8217 – 8239
CCB	2018H	8216
Общего назначения	2014H – 2017H	8212 – 8215
Младшие векторы прерывания	2000H – 2013H	8192 – 8211

21.5 Последовательность включения питания

1 Подать напряжение низкого уровня на вывод RESET#, пока U_{CC1} не стабилизируется. В это время VPR и EA# выводы могут быть не подключены.

2 После стабилизации генератора и U_{CC1}, оставить устройство в состоянии сброса и подать напряжение U_{OH2} на EA#. После стабилизации EA# подать напряжение U_{OL} на вывод VPR.

3 Установить значение PMODE для выбора алгоритма программирования. Значения для каждого режима приведены в таблице 21.1.

4 Подать высокий уровень на вывод RESET#.

5 Работать с выбранным алгоритмом программирования.

21.6 Защита памяти

Аппаратная и программная защита – это отдельные части в схеме защиты памяти. Аппаратная защита предохраняет от внешней записи и считывания следующим образом. В регистре состояния схемы (CCR) устанавливаются биты кода безопасности, CCR.6 (LOC0) и CCR.7 (LOC1), запрещая чтение и/или запись в EEPROM. Попытка записи вызовет цикл записи контроллера шины, однако записи не произойдет. Попытка чтения ПЗУ при выполнении программы из внешней памяти приведет к сбросу ИС.

Программная защита предотвращает неавторскую запись или верификацию EEPROM следующим образом. Механизм защиты по ключу ограничивает доступ во внутреннюю память. Если устройство введено в режим программирования, то алгоритм режима требует проверки ключа перед началом работы. Если проверка не пройдена, устройство зацикливается, пока не произойдет сброс.

Также для защиты возможно использование соответствующих функций модуля отладки.

21.6.1 Биты кода защиты в CCR

Регистр состояния схемы CCR устанавливает структуру шин и другие параметры устройства во время программирования. На рисунке 21.3 показана его структура, включая биты защиты. Процедура сброса загружает CCR из ячейки с адресом 2018H, где расположен байт конфигурации устройства CCB. Это обеспечивает защиту как втечение нормального выполнения программы, так и во время функционирования программы-монитора. Установка CCB.6 или CCB.7 воздействует на режим RUN-TIME и на любые попытки доступа к внутренней памяти от внешних устройств. При вводе ИС в режим программирования всего лишь меняется начальный адрес запуска программы (на 9000H), но CCB загружается с того же адреса, что и при нормальной работе (2018H).

CCR
регистр конфигурации кристалла

значение после Сброса: 1FH

7	6	5	4	3	2	1	0
LOC1	LOC0	IRC1	IRC0	ALE	WR	BW0	PD

Рисунок 21.3 – Формат регистра CCR

Следует установить CCR.6, чтобы сделать невозможной запись и предотвратить дальнейшее программирование области ПЗУ 2000H–9FFFH. Попытка записи вызывает цикл записи контроллера шины, однако записи не произойдет.

Следует установить CCR.7, чтобы запретить программе во внешней памяти доступ к содержимому EEPROM. Внутренняя память недоступна для считывания. Чтение данных может происходить, если основной счетчик команд (PC) находится в области 2000H – 9FFFH. При выполнении кода из внешней памяти любое чтение из области внутреннего ПЗУ приведет к сбросу ИС (рисунок 21.4).

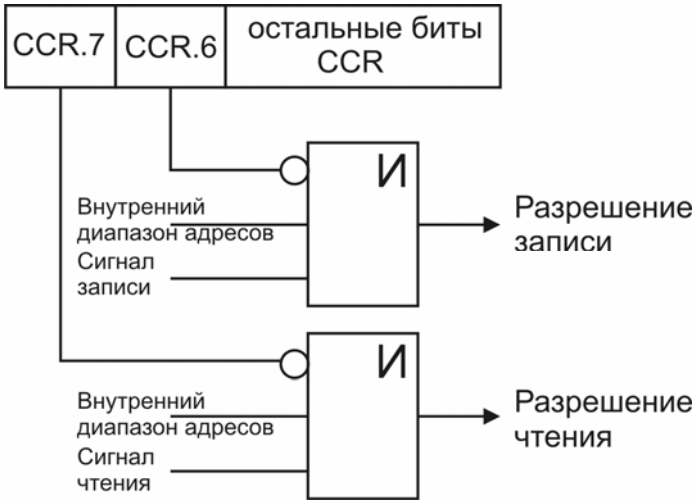


Рисунок 21.4 – Байт конфигурации кристалла

Режимы программирования требуют проверки ключа защиты до начала выполнения программирования. По умолчанию ПЗУ содержит нули, поэтому при первом программировании необходимо подавать нули в качестве ключа защиты.

21.6.2 Ключ защиты

Ключ – это 128-битная область во внутренней памяти с адреса 2020H до 202FH. В этих ячейках по умолчанию содержатся нули. Для активации ключа защиты необходимо записать пользовательские данные в эти ячейки. Ячейки ключа защиты доступны при чтении из внешней памяти и их использование не имеет смысла без установки CCR.7. Разделы о режимах AUTO, SLAVE, ROM-DUMP, SERIAL содержат информацию о проверке ключа в каждом режиме. Если проверка ключа не пройдена, устройство входит в бесконечный цикл и должно быть сброшено для выхода из него.

Программирование ключа защиты

Если во время записи ключа произошел скачок напряжения или сброс, может случайно записаться неизвестный ключ, делающий устройство недоступным для последующего программирования. Чтобы избежать этого, основной массив EEPROM программируют до программирования битов кода защиты CCB (CCB.6 и CCB.7). Но запись битов CCR.6 и CCR.7 не является необратимой, то есть ИС можно полностью стереть для последующего использования (подраздел 21.7). Если включена защита записи

ПЗУ, то подача команды на стирания любого блока памяти приведет к полной очистке памяти. Если же необходимо не допустить возможности стирания ПЗУ при защите, то необходимо обеспечивать соответствующую программную или аппаратную (при помощи блока отладки) блокировку доступа к регистру управления программированием. Так как процедура программирования состоит всегда из двух шагов – записи регистра ROMC и подачи непосредственной команды записи данных, то возможно прерывание последовательности их выполнения прерыванием DEBUG. Если были запрограммированы биты защиты, то защита включится только после перезагрузки микроконтроллера.

21.7 Режим RUN-TIME

Этот режим является аппаратно поддерживаемым и обеспечивает формирование нужных управляющих сигналов для корректного программирования EEPROM. Ячейки EEPROM можно программировать во время обычного выполнения программы. При этом ИС должна работать в режиме работы с внутренней памятью. Перед непосредственной подачей команды записи необходимо выбрать соответствующий тип операции в управляющем регистре ROMC (адрес 1FFCH). Операцией может быть: запись слова, очищение строки памяти или очищение целого блока памяти. Формат регистра ROMC представлен на рисунке 21.5

Рисунок 21.5 – Формат регистра ROMC

Большая часть битов регистра ROMC является зарезервированной для режимов тестирования ПЗУ. Установка этих битов может привести к непредсказуемым последствиям.

Бит EW – разрешает запись слова в EEPROM. Запись осуществляется подачей любой команды пересылки. После каждой записи бит сбрасывается и требуется повторная установка бита.

Бит RE – разрешает стирание строки памяти. При этом бит EW должен быть установлен. Стирание осуществляется подачей команды пересылки нулевых данных по любому адресу, находящемуся в выбранной строке. В каждом блоке 128 строк по 16 байт.

Бит BW – разрешает стирание блока памяти. При этом бит EW должен быть установлен. Стирание осуществляется подачей команды пересылки нулевых данных по любому адресу, находящемуся в выбранном блоке. В микросхеме16 блоков EEPROM, каждый емкостью 1К×16 бит. Если ИС находится в режиме защиты ПЗУ от записи, то подача команды на стирание блока приведет к полному уничтожению содержимого ПЗУ.

Бит STBY – перевод ПЗУ в режим пониженного потребления. Не следует записывать «1», если включен режим работы с внутренней памятью. При работе с внешней памятью этот бит автоматически после сброса устанавливается в «1». Время перехода из режима пониженного потребления в активный режим – не менее 5 мкс.

В таблице 21.4 представлены начальные и конечные адреса блоков, в таблице 21.5 – адреса строк на примере первого блока памяти.

Таблица 21.4 – Адреса блоков памяти EEPROM

Номер блока	Диапазон адресов	
	Шестнадцатеричные	Десятичные
1	2000H – 27FFH	8192 – 10239
2	2800H – 2FFFH	10240 – 12287
3	3000H – 37FFH	12288 – 14335
4	3800H – 3FFFH	14336 – 16383
5	4000H – 47FFH	16384 – 18431
6	4800H – 4FFFH	18432 – 20479
7	5000H – 57FFH	20480 – 22527
8	5800H – 5FFFH	22528 – 24575
9	6000H – 67FFH	24576 – 26623
10	6800H – 6FFFH	26624 – 28671
11	7000H – 77FFH	28672 – 30719
12	7800H – 7FFFH	30720 – 32767
13	8000H – 87FFH	32768 – 34815
14	8800H – 8FFFH	34816 – 36863
15	9000H – 97FFH	36864 – 38911
16	9800H – 9FFFH	38912 – 40959

Таблица 21.5 – Адреса строк в первом блоке памяти EEPROM

Номер строки	Диапазон адресов	
	Шестнадцатеричные	Десятичные
1	2000H – 200FH	8192 – 8207
2	2010H – 201FH	8208 – 8223
3	2020H – 202FH	8224 – 8239
4	2030H – 203FH	8240 – 8255
.		
.		
.		
128	27F0H – 27FFH	10224 – 10239

Ниже представлен пример программы, осуществляющей стирание первого блока памяти и записи значения 5555H в ячейку с адресом 2090H

```
LD AX,#1FFCH
LDB BX,#05H
STB BX,[AX]          ;enable erasing block

LD AX,#2000H
LD BX,#0000H
ST BX,[AX]

LD AX,#1FFCH
LDB BX,#01H
STB BX,[AX]          ;enable programming cell

LD AX,#2090H
LD BX,#5555H
ST BX,[AX]
```

Все режимы программирования используют RUN-TIME режим, только с разными источниками входных данных (последовательный порт, порты 3/4, интерфейс внешней памяти).

При записи слова ИС формирует управляющие сигналы для EEPROM нужной длительности. Для корректного осуществления программирования они не должны быть меньше установленных величин. Временные данные хранятся в регистрах:

- PPW (адрес 1FF2H/1FF3H). Хранит коэффициент для вычисления длительности программирующего импульса. Длительность должна быть не менее 4,5 мс;
- PPSETUP (адрес 1FF4H). Хранит коэффициент для вычисления времени установки данных относительно переднего фронта программирующего импульса. Время установки должно быть не менее 1 мкс;
- PPHOLD (адрес 1FF5H). Хранит коэффициент для вычисления времени удержания данных относительно заднего фронта программирующего импульса. Время удержания должно быть не менее 1 мкс.

Общая длительность как программирования одного слова, так и стирания одной строки или одного блока суммируется из времени установки, длительности программирующего импульса и времени удержания (рисунок 21.6).

Рисунок 21.6 – Программирующий импульс EEPROM

Длительность программирующего импульса рассчитывается по формуле

$$T_{PPW} = (PPW \times 16) \times Tмц,$$

где PPW – значение в десятичном представлении;

Tмц – длительность машинного цикла.

По умолчанию значение PPW соответствует 4,8 мс на частоте 33 МГц.

Время установки программирующего импульса рассчитывается по формуле

$$T_{PPSETUP} = PPSETUP \times Tмц,$$

где PPSETUP – значение в десятичном представлении;

Tмц – длительность машинного цикла.

По умолчанию значение PPW соответствует T_{PPW} = 4,8 мс на частоте 33 МГц.

Время установки программирующего импульса рассчитывается по формуле

$$T_{PPSETUP} = PPSETUP \times Tмц,$$

где PPSETUP – значение в десятичном представлении;

Tмц – длительность машинного цикла.

По умолчанию значение PPSETUP соответствует $T_{PPSETUP}$ = 1,2 мкс на частоте 33 МГц.

Время удержания программирующего импульса рассчитывается по формуле

$$T_{PPHOLD} = PPHOLD \times Tмц,$$

где PPHOLD – значение в десятичном представлении;

Тмц – длительность машинного цикла.

По умолчанию значение PPHOLD соответствует T_{PPHOLD} = 1,2 мкс на частоте 33 МГц.

После подачи команды на запись в EEPROM микроконтроллер сам формирует сигналы нужной длительности, а на время программирования переводит ядро в режим пониженного энергопотребления IDLE. В случае если планируется программирования всей памяти или стирание большого количества блоков, рекомендуется отключить периферийные устройства (особенно АЦП и ЦАП). Это позволит уменьшить потребление цифровой части схемы, обеспечив более стабильное питание ПЗУ.

21.8 Режим программирования AUTO

Этот режим является альтернативным режимом, обеспечивающим программирование с минимальными затратами. Можно программировать МК без использования программирующего устройства. Прибор программирует сам себя, выбирая данные из внешнего EPROM (ячейки 0A000H–0FFFFH и 0000H–0FFFH, смотри таблицу 21.6). CCB загружается в регистр состояния схемы (CCR). МК получает данные через внешнюю шину; поэтому CCB должен соответствовать системе памяти, установленной в режиме программирования, которая не обязательно соответствует системе памяти в разрабатываемом проекте. В начале работы алгоритма режима AUTO осуществляется проверка кода защиты. Если проверка не прошла, устройство входит в бесконечный цикл и должно быть возвращено в исходное состояние через сброс (смотри таблицу 21.7). Программирование ключа защиты описывается в пункте 21.6.2.

Таблица 21.6 – Карта памяти автоматического режима

Адреса внешнего EPROM	Адреса внутреннего EEPROM	Описание
0000H – 0FFFH	8000H – 8FFFH	Зарезервированные для данных и команд ячейки
A000H – FFFFH	2000H – 7FFFH	Зарезервированные для данных и команд ячейки
A020H – A02FH	2020H – 202FH	Ключ во время проверки

Таблица 21.7 – Функции выводов в режиме AUTO

Имя	Тип	Описание
PACT# (P2.7)	Выход	Активность программирования. Нуль указывает на продолжающееся программирование
PBUS (порты 3 и 4)	Вход / Выход	Шина адреса/управления/данных. Используется для доступа к внешней памяти
PVER (P2.0)	Выход	Проверка. Сигнал корректируется после каждого такта. Высокий уровень сигнала указывает на успешное программирование, низкий – на ошибку

Алгоритм режима автоматического программирования проверяет ключ до начала работы. Сравнивается внешняя область A020H – A02FH с внутренней 2020H – 202FH. Внешняя и внутренняя области должны совпадать, иначе устройство входит в бесконечный цикл, предотвращающий программирование.

Чтобы войти в режим AUTO, необходимо установить PMODE в 0CH и подать питание в последовательности, приведенной в подразделе 21.5. На рисунке 21.7 описана программа, содержащаяся в ПЗУ.

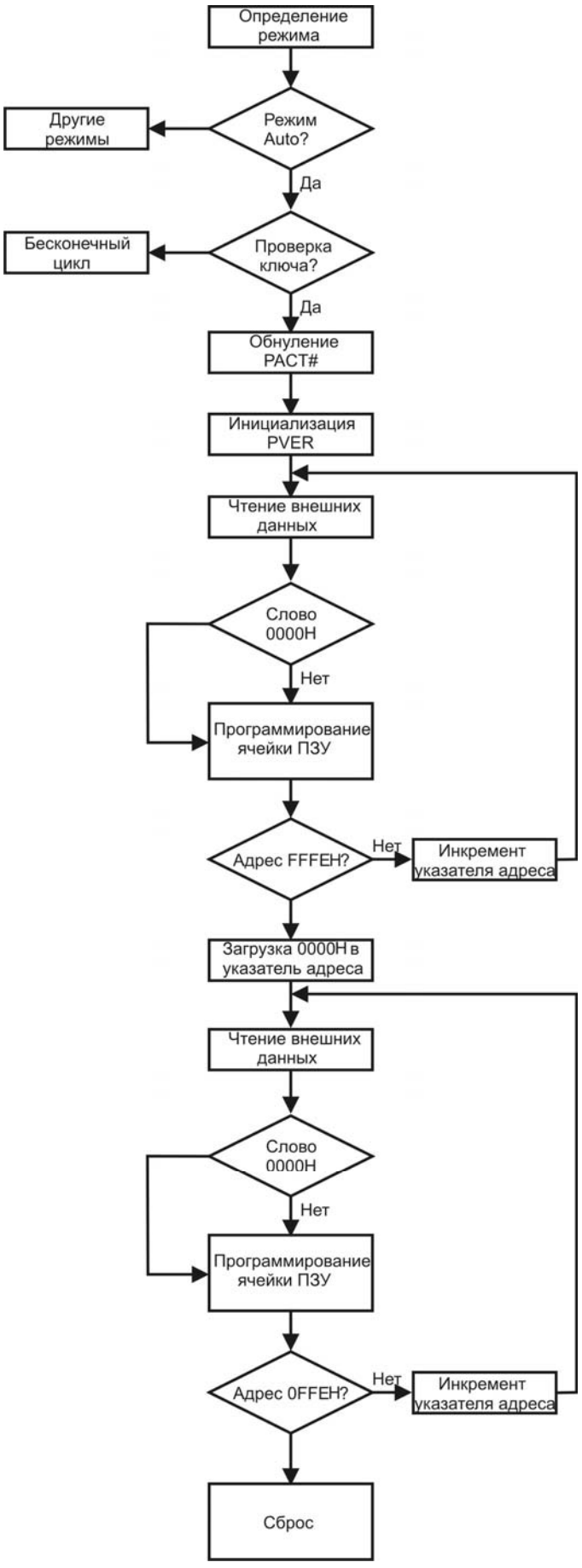


Рисунок 21.7 – Алгоритм режима AUTO

Последовательность операций в режиме AUTO.

1 Определение режима программирования (чтение PMODE).

2 Если выбран режим программирования AUTO – продолжение алгоритма. (PMODE = 0CH).

3 Сравнить ключ защиты с данными во внешней памяти по адресам 0A020H – 0A02FH.

4 Обнуление PACT# (низкий уровень P2.7).

5 Инициализация PVER (высокий уровень P2.0).

6 Чтение внешних данных, начинающихся с 0A000H.

7 Если слово не 0000H, выполняется программирование ячейки EEPROM (подробнее описано в подразделе 21.7).

8. Если адрес внешней ячейки памяти равен 0FFFEH (соответствует 24 Кбайт записанной информации), то в указатель адреса загружается 0000H и выполняются последовательно действия по чтению из внешней памяти и программированию.

9. Когда адрес достигнет 0FFEH, то происходит сброс ИС и переход в режим нормальной работы.

21.9 Режим программирования SLAVE

В течение этого режима каждая внутренняя ячейка EEPROM доступна для записи и проверки по отдельности.

Существует подрежим, в котором адрес инкрементируется автоматически, и возможно чтение или запись в ячейки ПЗУ, расположенные последовательно, без необходимости передачи адреса по шине для каждого чтения или записи. Скорость процесса увеличивается, так как устраняется необходимость формировать и вводить каждый последующий адрес.

При входе в этот режим происходит проверка ключа защиты. Если ключ неверен, запись или считывание содержимого любой ячейки запрещены.

Примечание – В данном режиме программирования порты 3 и 4 функционируют в режиме с открытым стоком, поэтому требуется подключение подтягивающих резисторов к выводам этих портов. Не допускается программирование микроконвертера в режиме SLAVE с помощью программатора для схем 1874BE76T, так как в них используется подача высокого напряжения (12,5 В) на выводы VPR и EA#. Это приведет к выходу из строя ИС 1874BE96T.

21.9.1 Режим SLAVE

МК в режиме SLAVE реагирует на команду Program Word или Dump Word. Порты (PBUS) служат для передачи адреса, данных и выборки команд. Некоторые из выводов порта 2 обеспечивают квитирование установления связи. Наименьший значащий бит порта 3 выбирает команду Program Word (P3.0 = 1) или Dump Word (P3.0 = 0). Старшие 15 битов содержат адрес программируемого или считываемого слова. 16 битов вместе представляют собой комбинацию адреса/команды. Адрес размещается с 2000H по 8FFFH. В таблице 21.8 описаны функции сигналов в режиме SLAVE.

Таблица 21.8 – Функции выводов в режиме программирования SLAVE

Имя	Тип	Описание
1	2	3
PALE# (P2.1)	Вход	Используется как разрешающий сигнал для чтения адресов и команд в устройство. PALE# обычно фиксируется на высоком уровне пользователем. PALE# разрешает чтение адресов и команд с третьего и четвертого портов
CPVER (P2.6)	Выход	Заключительная проверка программы. Когда работа закончена, высокий уровень показывает, что все ячейки записаны успешно, а низкий уровень указывает на ошибку, случившуюся во время одной из операций

Окончание таблицы 21.8

1	2	3
AINC# (P2.4)	Вход	Автоматический инкремент. Низкий активный уровень разрешает режим автоматического инкремента. Он позволяет осуществлять чтение или запись последовательных ячеек без передачи каждый раз адреса по шине. В режиме Program Word: AINC# проверяется после записи в каждую ячейку. Если AINC# установлен, адрес инкрементируется и вводится следующее слово данных. В режиме Dump Word: AINC# проверяется после каждого слова, которое было записано в порты 3 и 4. Если AINC# установлен, адрес инкрементируется и устройство готово к выводу следующего слова данных
PBUS (порты 3 и 4)	Вход/ Выход	Шина адреса/управления/данных. Используется для чтения и записи команд, адресов и данных. Порты 3 и 4 используются в режиме ввода-вывода с открытым стоком (не как системная шина). Добавив внешние нагрузочные резисторы, можно считывать данные из устройства во время процедуры Dump Word
PROG# (P2.2)	Вход	Начало программирования. PROG# используется как разрешающий сигнал для чтения данных в течение процедуры Program Word. PROG# обычно фиксируется на высоком уровне пользователем. PROG# разрешает чтение данных из портов 3/4 и программирование внутреннего ПЗУ (EEPROM). Если PROG# остается установленным, в ячейку записываются те же данные в течение двадцати пяти импульсов по 500 мкс. По этой причине данные в портах должны оставаться неизменными, пока PROG# установлен. Когда PROG# станет высоким, работа продолжается. PROG# используется как разрешение установления связи для вывода данных через порт в процедуре Dump Word. PROG# устанавливается в высокий уровень пользователем. PROG# разрешает вывод содержимого ячейки ПЗУ (EEPROM) через порт. Когда PROG# станет высоким, работа продолжается
PVER (P2.0)	Выход	Проверка правильности программирования. Сигнал корректируется после каждого программирующего импульса. Высокий уровень указывает на успешное программирование, низкий – на обнаруживаемую ошибку

21.9.2 Проверка ключа защиты в режиме SLAVE

Алгоритм программирования в режиме SLAVE в начале работы проверяет ключ защиты. Для этого необходимо первыми командами задать программирование восьми последовательных слов с 2020H по 202FH. Алгоритм сравнивает записываемые величины с содержимым ячеек 2020H – 202FH внутреннего ПЗУ. Если проверка прошла, то предоставляется доступ к устройству, иначе оно зависает и должно быть возвращено в исходное состояние через сброс. Устройство запрещает запись или вывод содержимого любой ячейки, если проверка не прошла.

21.9.3 Алгоритм режима SLAVE

Последовательность включения питания (подраздел 21.5) и таблица значений PMODE (таблица 21.1) содержат информацию, необходимую для входа в режим SLAVE. Алгоритм включает три функциональных блока: декодер адрес/команда, процедуры программирования слова (Program Word) и считывания слова (Dump Word).

На рисунке 21.8 показана программа дешифрации адреса/данных, на рисунке 21.9 – алгоритм Program Word, на рисунке 21.10 – алгоритм Dump Word.

21.9.4 Команды Program Word и Dump Word

В таблице 21.9 определены обозначения временных интервалов во временных диаграммах команд Program Word и Dump Word. На рисунках 21.11 и 21.12 приведены временные характеристики этих сигналов.

Таблица 21.9 – Обозначения временных характеристик

Обозначение	Описание
Tshll	Сброс высокого уровня до первого низкого PALE#
Tlllh	Длительность PALE# импульса
Tavll	Время установки адреса
Tllax	Время удержания адреса
Tpldv	Время от низкого PROG# до действительного слова вывода
Tphdx	Удержание данных слова вывода
Tdvpl	Время установки данных
Tpldx	Время удержания данных
Tplph	Длительность импульса PROG#
Tphll	Время от высокого PROG# до следующего низкого PALE#
Tlhpl	Время от высокого PALE# до низкого PROG#
Tphpl	Время от высокого PROG# до следующего низкого PROG#
Tphil	Время от высокого PROG# до низкого AINC#
Tilih	Длительность импульса AINC#
Tilvh	PVER удерживается после низкого AINC#
Tilpl	Время от низкого AINC# до низкого PROG#
Tphvl	Время от высокого PROG# до действительного PVER

Последовательность операций при дешифрации адреса/данных (рисунок 21.8)

1 Определение режима программирования (чтение PMODE).

2 Если выбран режим SLAVE (PMODE = 5H), продолжение алгоритма.

3 Проверка ключа безопасности.

4 Если PALE# активен, переход к следующему шагу, иначе – повторная операция.

5 Чтение данных с портов 3/4.

6 Если PVER установлен (ошибки нет), – переход к шагу 8. Если PVER не установлен – очистить CPVER, установить PVER, затем переход к шагу 7.

7 Ожидание сброса PALE#.

8 Проверка диапазона адресов портов 3 и 4.

9 Если P3.0 = 1, переход к процедуре Program Word (режим команд). Если P3.0 = 0, переход к процедуре Dump Word (режим адреса). Используется слово в портах 3 и 4 как адрес программирования или считывания.

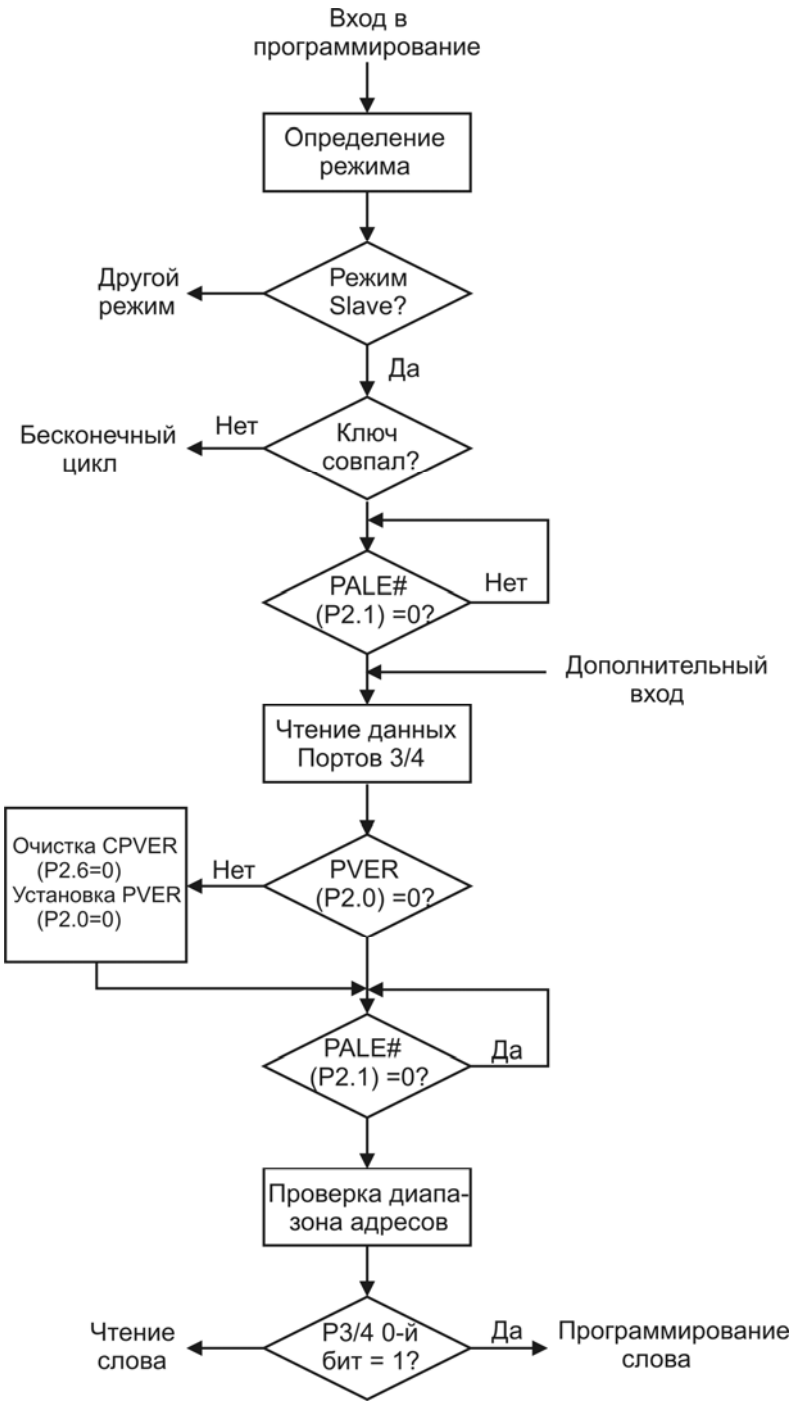


Рисунок 21.8 – Алгоритм дешифратора адреса/данных

Последовательность операций в процедуре Program Word (рисунок 21.9).

1 Ожидание установки PROG# в активный уровень.

2 При установлении PROG# – чтение данных из портов 3 и 4.

3 Вызов алгоритма программирования (подраздел 21.7).

4 Если верификация пройдена, – установка PVER, иначе – очистка PVER.

5 Удержание PROG# в активном уровне пока не завершатся программирование.

6 Если PALE# активен, повторный вход в процедуру дешифрации адреса/команд и чтение следующего адреса/команды.

7 Если PALE# сброшен, – проверка AINC#.

8 Если AINC# активен, – адрес инкрементируется на два, затем верификация.

9 Если PVER установлен (нет ошибки), – повторный вход в процедуру Program Word.

10 Если PVER сброшен (ошибка верификации), – очистка CPVER, установка PVER, затем повторный вход в процедуру Program Word.

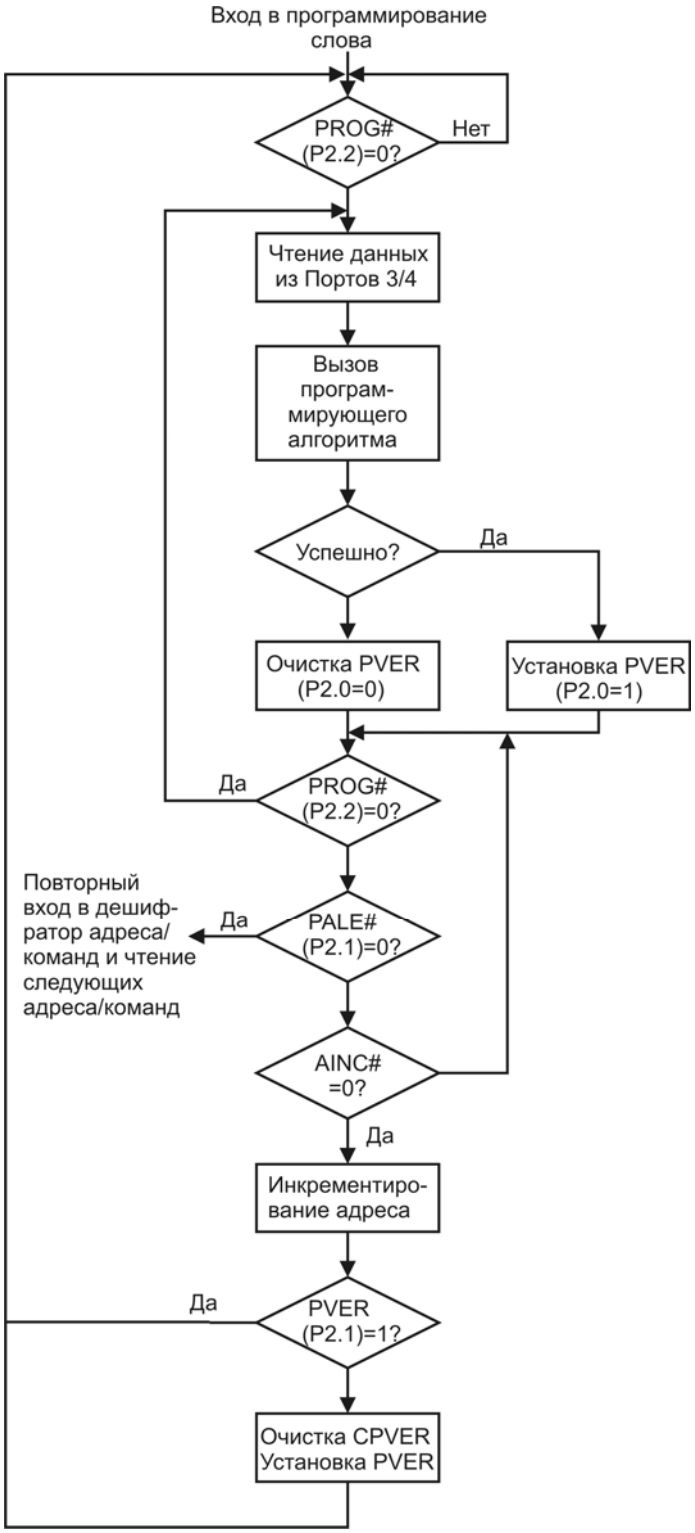


Рисунок 21.9 – Процедура Program Word

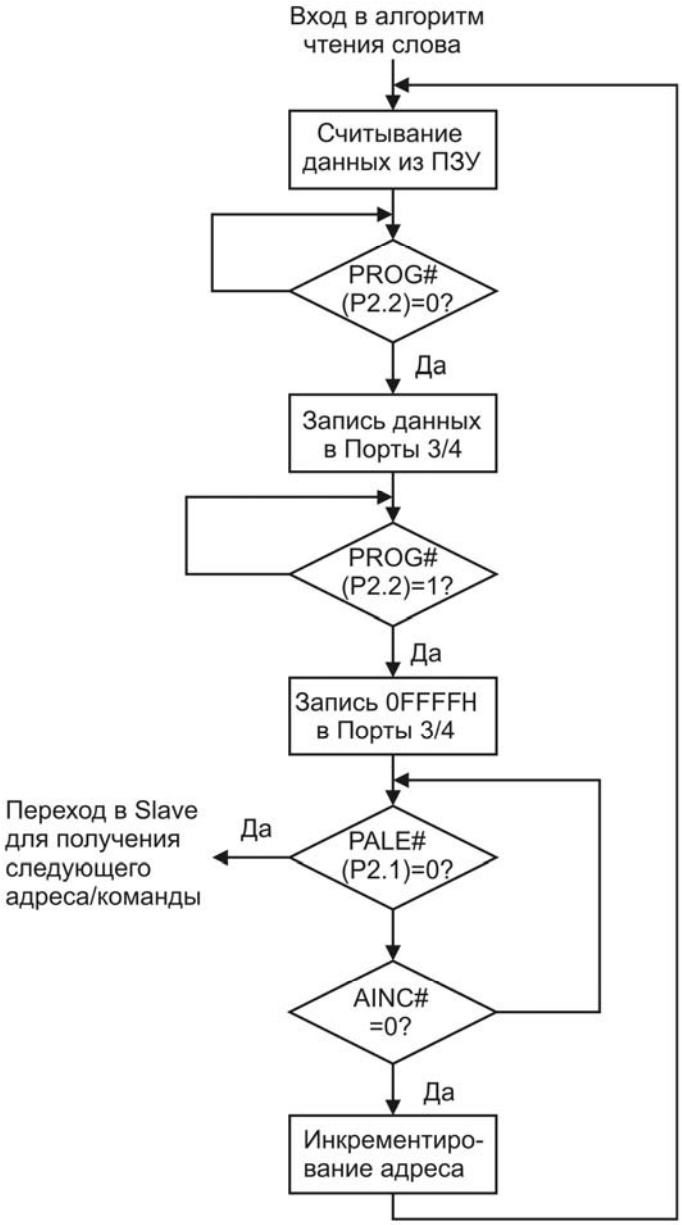


Рисунок 21.10 – Процедура Dump Word

Последовательность операций процедуры Dump Word (рисунок 21.10).

1 Чтение данных из EEPROM.

2 Ожидание установки PROG# в активный уровень.

3 При установлении PROG# – передача данных через порты 3 и 4.

4 Ожидание сброса PROG# .

5 Запись всех единиц в порты 3 и 4. (Это позволяет перевести выводы портов в третье состояние, избежав конфликта с процедурами Program Word и дешифрации адреса/команд, использующими порты 3 и 4 как входы.)

6 Если PALE# установлен, – возврат в процедуру дешифрации адреса/команд. Если PALE# сброшен – чтение AINC#.

7 Если AINC# не установлен, – возврат к шагу 6.

8 Если AINC# установлен, – адрес инкрементируется на два, затем повторный вход в эту процедуру.

На рисунке 21.11 показаны временные диаграммы команды Program Word с повторением импульсов программирования и автоинкрементом. Единица на P3.0

выбирает команду Program Word. Командой 3501H программируется ячейка слова, расположенная по адресу 3500H. Устройство получает входной сигнал PALE#, что показывает активность команды; установка PALE# фиксирует команду и адрес в портах 3 и 4 (PBUS). Установка PROG# фиксирует данные в портах 3 и 4 для начала программирования; устройство считает или выводит слово. Длительность импульса PROG# определяет длительность программирующего импульса. По нарастающему фронту сигнала PROG# идет автоматическая проверка содержимого только что запрограммированной ячейки. Установка сигнала PVER# показывает успешное программирование. AINC# служит для автоматического инкремента адреса.

На рисунке 21.12 представлена команда Dump Word. Ноль на P3.0 выбирает команду Dump Word. Посылка команды «2100H» помещает слово из адреса внутренней памяти 2100H в порты 3 и 4. PROG# управляет подключением устройства к шине. В режиме Dump Word сигнал на выводе AINC# может оставаться активным или переключаться из одного состояния в другое. Вывод PROG# автоматически инкрементирует адрес.

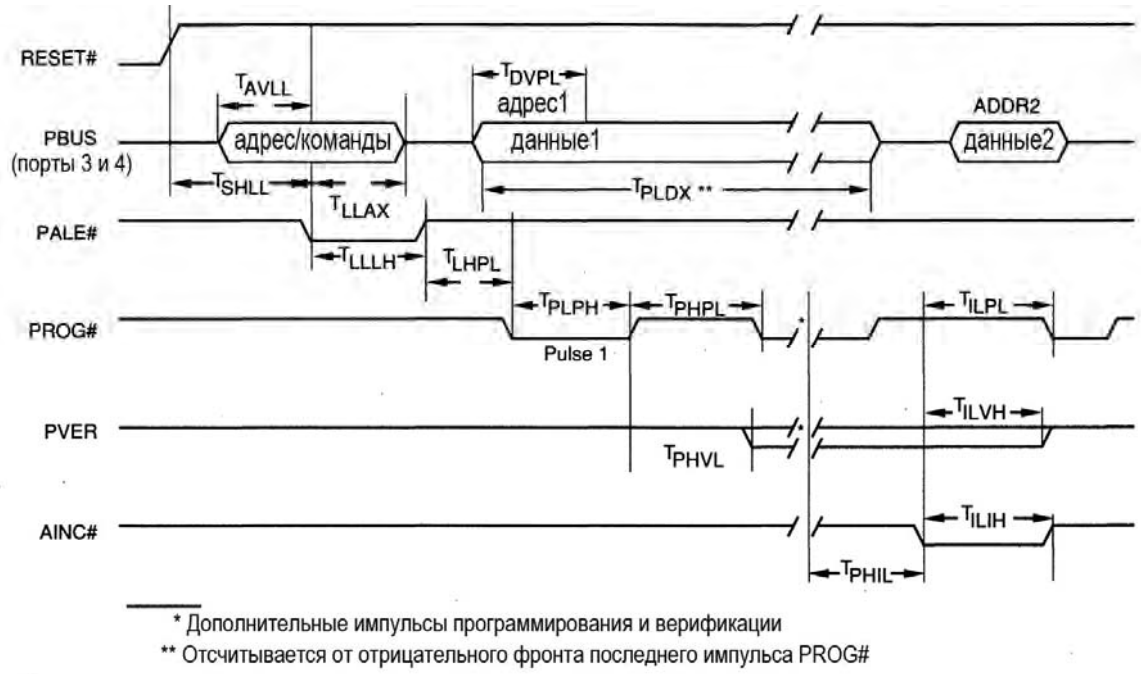


Рисунок 21.11 – Временные диаграммы в режиме Program Word

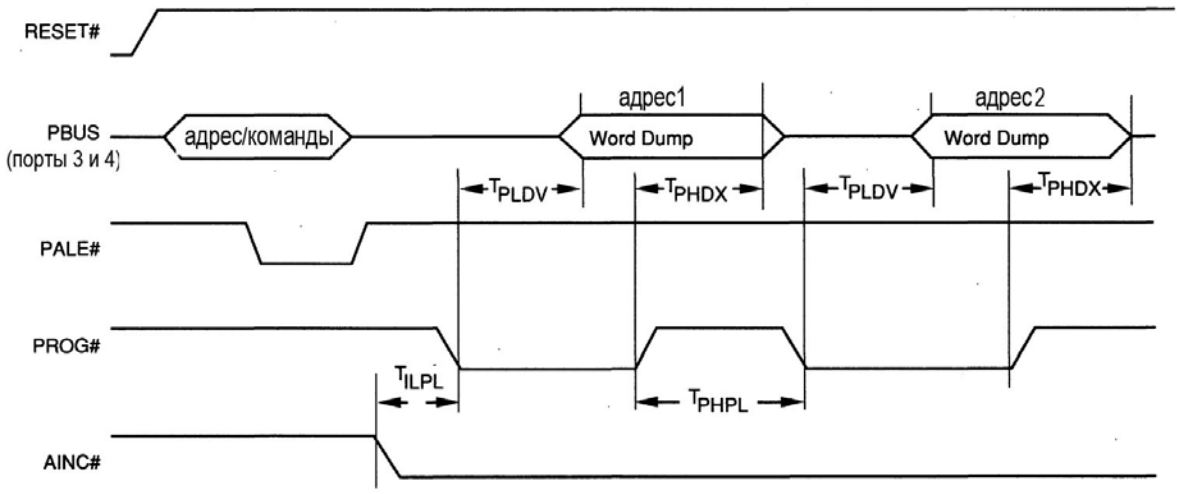


Рисунок 21.12 – Временные диаграммы режима Dump Word

21.10 Режим вывода содержимого памяти (ROM-DUMP)

Этот режим легко позволяет проверить содержимое EEPROM. Последовательность включения питания (подраздел 21.5) и таблица значений PMODE (таблица 21.1) помогут войти в режим ROM-DUMP (таблица 21.10).

После проверки ключа защиты режим записывает всю область внутренней памяти во внешнюю память. Ключ защиты во внутренней памяти должен соответствовать при проверке ключу защиты в соответствующей области внешней памяти. Если ключ не совпадает, устройство входит в бесконечный цикл и может быть выведено из него только сбросом.

Таблица 21.10 – Карта памяти режима ROM-DUMP

Устройство	Адреса внутреннего EEPROM	Адреса внешней памяти
МК	2000H – 8FFFH	2000H – 8FFFH
МК Ключ защиты	2020H – 202FH	2020H – 202FH

21.11 Режим программирования через последовательный порт SERIAL

Данные в этом режиме передаются и принимаются через выводы TXD0 (P2.0) и RXD0 (P2.1). Эти выводы следует подключить к любому смарт-терминалу, способному работать в последовательном режиме. Любой хост, использующий интерфейс RS-232C (например, ПК), должен подключаться через специальный RS-232C приемопередатчик (драйвер).

В режиме программирования последовательного порта используются специальные RISM-команды, которые позволяют считывать и записывать внутреннюю память микроконтроллера. В следующих разделах описаны RISM-команды и представлены примеры их использования.

Выводы XTAL1 и XTAL2 подключаются к кварцевому резонатору с частотой от 10 до 33 МГц. Последовательная скорость двоичной передачи по умолчанию для частоты кварца 33 МГц составляет 9 600 бод. Используя частоту отличную от 33 МГц, получится другая последовательная скорость передачи по умолчанию. Можно изменить скорость последовательной двоичной передачи по умолчанию, записав соответствующее значение в регистр скорости двоичной передачи последовательного порта (SP_BAUD0).

21.12 Ввод микроконвертера в режим программирования через последовательный порт SERIAL

Во время работы отладочной программы необходимо выбрать режим программирования, управляя сигналами PMODE (таблица 21.1). Чтобы войти в режим программирования через последовательный порт, значение PMODE должно быть 0EH.

После входа в режим SERIAL микроконтроллер проверяет бит блокировки CCB0.7. Если CCB0.7 = 1, то происходит блокировка и микроконтроллер входит в бесконечный цикл. Если CCB0.7 = 0, то программирование продолжается. Ключи защиты, определенные в 2020H – 202FH, должны быть не запрограммированы в этом режиме.

21.13 RISM-команды

После входа в режим программирования через последовательный порт микроконтроллер запускает отладочную подпрограмму TROM. Эта подпрограмма калибрует последовательный порт и его сигналы. Последовательный порт настраивается, чтобы работать в режиме 1 со скоростью двоичной передачи 9 600 бод на 33 МГц. Можно изменить эти параметры настройки по умолчанию после входа в режим SERIAL, записав значение скорости двоичной передачи в соответствующий регистр микроконтроллера.

Можно соединить микроконтроллер с любым смарт-терминалом через выводы RXD0 и TXD0. Команды, адреса и данные микроконтроллеру следует передавать через его вывод RXD0. Микроконтроллер передает данные, сдвигая их на выводе TXD0. Передавать информацию микроконтроллеру следует, используя 10-битовые фреймы

(стартовый бит, 8 бит данных, начиная с LSB, стоповый бит). Аналогично, микроконтроллер будет передавать информацию, сдвигая 10-битовые фреймы.

Выполнение передачи команд, соответствующих битам (00 – 014H). В каждый момент времени передается один байт данных или адреса, поэтому первым должен идти старший байт. Необходимо указывать полный адрес (16 бит). Так как микроконтроллер воспринимает данные в диапазоне 00 – 014H как команды, перед передачей байта данных или адреса меньше 015H необходимо отправить команду SET_DLE_FLAG, которая укажет микроконтроллеру, что следующий байт не является командой. Микроконтроллер также отвечает на каждый полученный байт тем же самым байтом, т. е. тем самым свидетельствует об успешном приеме данных или успешном выполнении команды. Исключение составляет команда TRANSMIT, после этой команды микроконтроллер отправляет байт данных, считанных из памяти.

В таблице 21.11 перечислены RISM-команды микроконтроллера. В следующих разделах объясняется, как использовать эти RISM-команды, чтобы осуществлять чтение и запись слов (байт) внутренней памяти.

Таблица 21.11– RISM-команды последовательного порта

Байты команд	Команды	Описание
00H	SET_DLE_FLAG	Команда передается, когда байт адреса или данных меньше 015H. Этот код говорит микроконтроллеру, что следующий байт, который он получит, не будет командой. Микроконтроллер воспринимает значения в диапазоне 00 – 14H как команды, если им не предшествует команда SET_DLE_FLAG
02H	TRANSMIT	Команда подается сразу после передачи команд READ_BYTE или READ_WORD. Эта команда заставляет микроконтроллер начать передачу данных с его вывода TXD. Эту команду следует отправлять один раз после команды READ_BYTE и дважды после READ_WORD
04H	READ_BYTE	Команда чтения байта
05H	READ_WORD	Команда чтения слова
07H	WRITE_BYTE	Команда записи байта
08H	WRITE_WORD	Команда записи слова
0AH	DATA_TO_ADDR	Команда подается сразу после передачи адреса
0CH	ERASE	Команда очистки ПЗУ
0FH	RESETSP	Команда сброса текущей последовательности команд

21.14 Чтение из внутренней памяти или RAM

Пять RISM-команд управляют чтением внутренней памяти:

- SET_DLE_FLAG (00H);
- DATA_TO_ADDR (0AH);
- READ_BYTE (04H);
- READ_WORD (05H);
- TRANSMIT (02H).

На рисунке 21.13 представлен алгоритм для чтения слов из внутренней памяти, а также указано, какие изменения необходимо сделать для чтения из внутренней памяти отдельных байт.

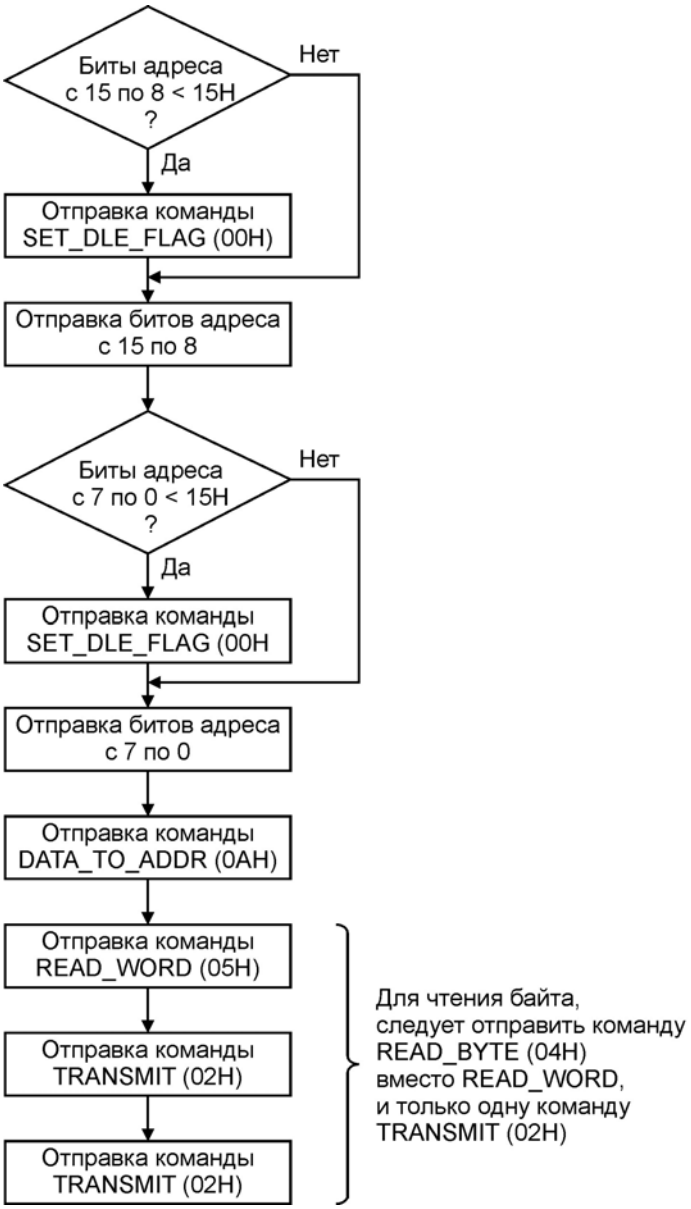


Рисунок 21.13 – Алгоритм чтения внутренней памяти микроконтроллера в режиме программирования через последовательный порт

21.15 Запись RAM

Четыре RISM-команды связаны с записью внутренней RAM:

- SET_DLE_FLAG (00H);
- DATA_TO_ADDR (0AH);
- WRITE_BYTE (07H);
- WRITE_WORD (08H).

На рисунке 21.14 показан алгоритм для записи слов во внутреннюю память.

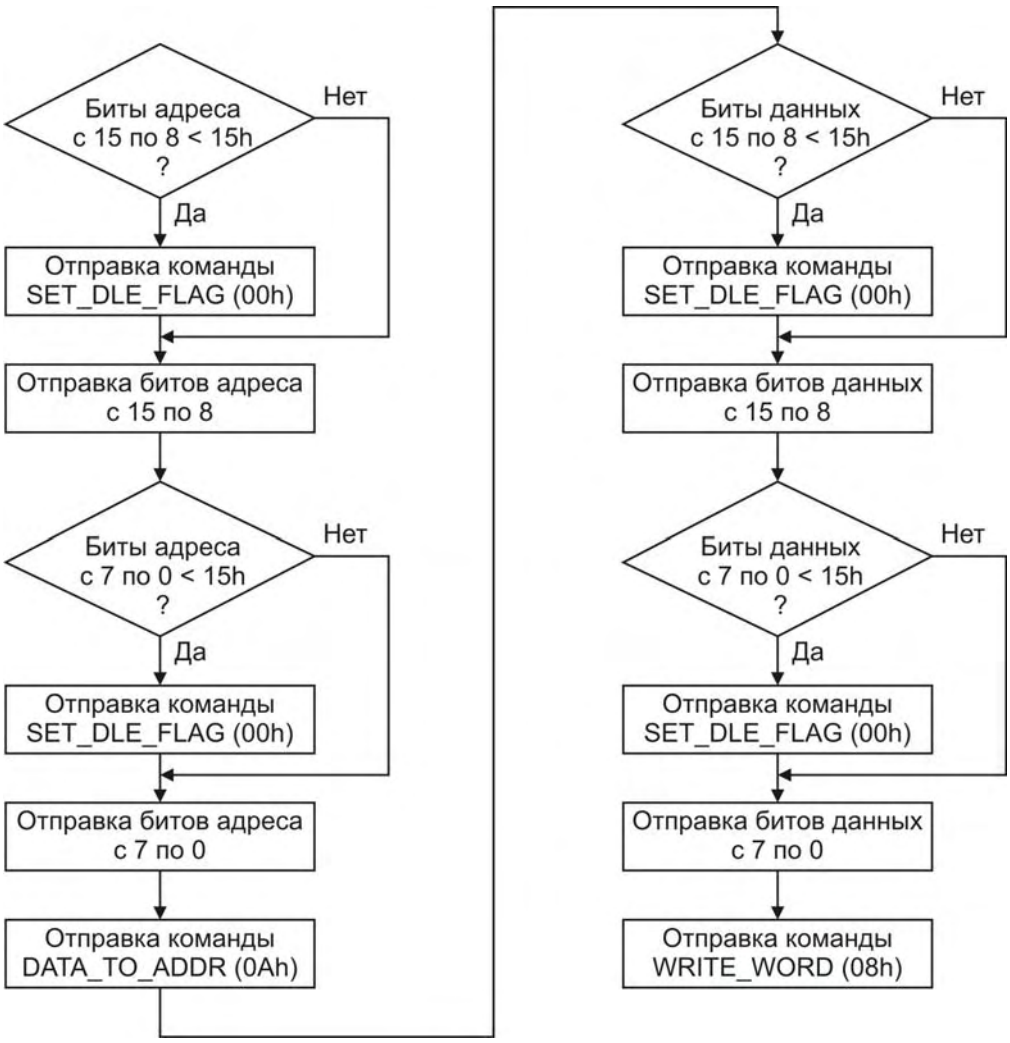


Рисунок 21.14 – Алгоритм записи во внутреннюю память микроконтроллера в режиме программирования через последовательный порт

22 Заключение

В настоящем руководстве КФДЛ.431295.042 приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения ИС 1874ВЕ96Т, которая представляет собой СБИС однокристального 16-разрядного микроконвертера с тактовой частотой до 33 МГц, ОЗУ (2024×8) бит, расширенным ОЗУ (2048×8) бит, восемью 16-разрядными АЦП с блоком цифровых компараторов, 14-разрядным ЦАП, 3-канальным ШИМ, двумя портами ввода-вывода UART, интерфейсами SPI и I2C, внутренней памятью программ (типа EEPROM) объемом 16К×16. Микроконвертер предназначен для применения в цифровых системах управления, для управления робототехническими комплексами, в системах автоматизации технологических процессов, в системах автоматизированного управления электроприводом, телекоммуникационной и другой технике.

Значения параметров, приведенные в настоящем руководстве, являются справочными.

Руководство КФДЛ.431295.042 может служить практическим пособием по применению микроконтроллеров для разработчиков систем на основе ИС 1874ВЕ96Т и программистов.

Приложение А
(обязательное)

Система команд МК 1874ВЕ96Т

В приложении дана справочная информация о системе команд МК. В нём описана каждая команда, показаны соотношения между командами и флагами PSW, показаны шестнадцатеричные коды команд, длины команд и время их выполнения.

В таблице А.1 определены переменные, используемые в таблице А.2 для замены операндов команд.

В таблице А.2 приведён список команд в алфавитном порядке и описание каждой из этих команд.

В таблицах А.3 и А.4 определены аббревиатура и символы, используемые в таблицах А.5 и А.6.

В таблице А.5 показано влияние каждой команды на флаги PSW.

В таблице А.6 показано влияние флагов PSW или соответствующих битов регистра на команды условного перехода.

В таблице А.7 приводится список шестнадцатеричных кодов команд по порядку вместе с соответствующими обозначениями команд.

В таблице А.8 дана карта кодов команд МК.

В таблице А.9 приводится список длин и кодов для каждого используемого режима адресации.

В таблице А.10 приводится время выполнения команд, измеряемых в машинных тактах.

Таблица А.1 – Переменные, используемые в операндах

Переменные	Описание
aa	Двухбитное поле внутри кода команды, определяющее основной используемый режим адресации. Это поле имеется только в кодах, в которых проводится выбор режима адресации. Поле декодируется следующим образом: 0 0 Прямая регистровая адресация 0 1 Непосредственная адресация 1 0 Косвенная адресация 1 1 Индексная адресация
baop	Однобайтный операнд, адресуются к которому любым способом
bbb	Трёхбитное поле внутри кода команды, определяющее выбор специфичного бита внутри регистра
bitno	Трёхбитное поле внутри кода команды, определяющее выбор одного из восьми битов внутри байта
breg	Однобайтный регистр во внутреннем регистровом файле. Когда не ясно, ссылается ли эта переменная на источник или приёмник, переменная помечается соответственно S или D
cadd	Адрес в программном коде.
Dbreg *	Однобайтный регистр во внутреннем регистровом файле, использующийся как операнд-приемник в команде
disp	Смещение. Расстояние между концом команды и целевой меткой в программе
Dwreg *	Регистр слова (двухбайтный) во внутреннем регистровом файле, использующийся как операнд-приемник в команде. Должен выравниваться по адресу, кратному двум
Ireg	32-битный регистр во внутреннем регистровом файле. Должен выравниваться по адресу, кратному четырем
Sbreg *	Однобайтный регистр во внутреннем регистровом файле, обслуживающий операнд-источник в команде
Swreg *	Регистр слова (двухбайтовый) во внутреннем регистровом файле, обслуживающий операнд-источник в команде. Должен выравниваться по адресу, кратному двум
waop	Двухбайтный операнд, к которому адресуются любым способом
wreg	Двухбайтный регистр во внутреннем регистровом файле. Когда не ясно, ссылается ли эта переменная на источник или приёмник, переменная помечается соответственно S или D. Должен выравниваться по адресу, кратному двум
XXX	Три старших бита смещения

_____

* Когда не ясно ссылается эта переменная на источник или приёмник, переменная помечается соответственно префиксом S или D.

Таблица А.2 – Система команд

Обозначение команды	Операция	Формат команды
1	2	3
ADD (2 операнда)	ADD WORDS (Сложение слов). Сложение слов источника и приёмника и сохранение суммы в приёмнике. (DEST) ← (DEST)+(SRC)	DEST, SRC ADD wreg, waop (011001aa)(waop)(wreg)
ADD (3 операнда)	ADD WORDS (Сложение слов). Сложение слов двух источников и сохранение суммы в приёмнике. (DEST) ← (SRC1)+(SRC2)	DEST, SRC1, SRC2 ADD Dwreg, swreg, waop (010001aa)(waop)(Swreg)(Dwreg)
ADDB (2 операнда)	ADD BYTES (Сложение байтов). Сложение байтов источника и приёмника и сохранение суммы в приёмнике (левый операнд). (DEST) ← (DEST)+(SRC)	DEST, SRC ADDB breg, baop (011101aa)(baop)(breg)
ADDB (3 операнда)	ADD BYTES (Сложение байтов). Сложение байтов двух источников и сохранение суммы в приёмнике. (DEST) ← (SRC1)+(SRC2)	DEST, SRC1, SRC2 ADDB Dbreg, Sbreg, baop (010101aa)(baop)(breg)
ADDC	ADD WORDS WITH CARRY (Сложение слов с переносом). Сложение слов источника и приёмника, сохранение суммы в приемнике и установка флага переноса (0 или «1»). (DEST) ← (DEST)+(SRC)+C	DEST, SRC ADDC wreg, waop (101001aa)(waop)(wreg)
ADDCB	ADD BYTES WITH CARRY (Сложение байтов с переносом). Сложение байтов источника и приёмника, сохранение суммы в приемнике и установка флага переноса (0 или «1») (DEST) ← (DEST)+(SRC)+C	DEST,SRC ADDCB breg, baop (101101aa)(baop)(breg)
AND (2 операнда)	LOGICAL AND WORDS (Логическое И слов). Операция И над словами источника и приёмника и сохранение результата в приёмнике. Результат – «1» в соответствующем бите, если оба операнда в этом бите имеют «1»; 0 – в остальных случаях. (DEST) ← (DEST)AND(SRC)	DEST, SRC AND wreg, waop (011000aa)(waop)(wreg)
AND (3 операнда)	LOGICAL AND WORDS (Логическое И слов). Операция И над словами двух источников и сохранение результата в приёмнике. Результат: «1» в соответствующем бите, если оба операнда в этом бите имеют «1»; 0 – в остальных случаях. (DEST) ← (SRC1)AND(SRC2)	DEST, SRC1, SRC2 AND Dwreg, Swreg, waop (010000aa)(waop)(Swreg)(Dwreg)
ANDB (2 операнда)	LOGICAL AND BYTES (Логическое И байтов). Операция И над байтами источника и приёмника и сохранение результата в приёмнике. Результат: «1» в соответствующем бите, если оба операнда в этом бите имеют «1»; «0» – в остальных случаях. (DEST) ← (DEST)AND(SRC)	DEST, SRC ANDB breg, baop (011100aa)(baop)(breg)

Продолжение таблицы А.2

1	2	3
ANDB (3 операнда)	LOGICAL AND BYTES (Логическое И байтов). Операция И над байтами двух источников и сохранение результата в приёмнике. Результат: «1» в соответствующем бите, если оба операнда в этом бите имеют «1»; «0» – в остальных случаях. (DEST) ← (SRC1)AND(SRC2)	DEST, SRC1, SRC2 ANDB Dbreg, Sbreg, baop (010100aa)(baop)(Sbreg)(Dbreg)
BMOV	BLOCK MOVE (Перемещение блоков). Перемещение блоков данных типа word из одной зоны памяти в другую. Адреса источника и приёмника вычисляются при помощи режима косвенной адресации с автоинкрементом. Длинный регистр (Ireg) содержит указатели источника и приёмника, которые хранятся в соседних регистрах слов. Регистр слова Wreg (CNTREG) определяет количество перемещений. Блоки данных могут находиться в любом месте памяти, но не должны перекрываться	DEST, SRC BMOV Ireg, wreg (11000001)(wreg)(Ireg) Примечание – CNTREG (Wreg) не декрементируется в процессе операции. Легко неумышленно создать длительную непрерываемую операцию с командой BMOV. Для создания прерываемых операций используется команда BMOVI
BMOVI	INTERRUPTABLE BLOCK MOVE (Прерываемое перемещение блоков). Перемещение блоков данных типа word из одной зоны памяти в другую. Команда идентична BMOV, исключая то, что BMOVI - прерываема. Адреса источника и приёмника вычисляются при помощи косвенного режима адресации с автоинкрементом. Длинный регистр (Ireg) адресуется к указателям источника и приёмника, которые хранятся в соседних регистрах слов. Регистр слов Wreg (CNTREG) определяет количество перемещений. Блоки данных могут находиться в любом месте памяти, но не должны перекрываться. COUNT ← (CNTREG) LOOP: SRCPTR ← (PTRS) DSTPTR ← (PTRS+2) (DSTPTR) ← (SRCPTR) (PTRS) ← SRCPTR+2 (PTSR+2) ← DSTPTR+2 COUNT ← COUNT – 1 if COUNT ≠ 0 then go to LOOP	DEST,SRC BMOVI Ireg,wreg (11001101)(wreg)(Ireg) Примечание – CNTREG (Wreg) не декрементируется, если выполнение команды не было прервано. Если BMOVI прервана, то в CNTREG сохраняется значение, бывшее в нём во время прерывания. По этой причине необходимо перезагружать CNTREG перед началом выполнения операции BMOVI
BR	BRANCH INDIRECT (Косвенное разветвление). Продолжает выполнение с адреса, определяемого операндом регистра слова PC ← (DEST)	DEST BR [wreg] (11100011)(wreg)

Продолжение таблицы А.2

1	2	3
CLR	CLEAR WORD (Очистка слова). Обнуляет значение операнда. (DEST) ← 0	DEST CLR wreg (00000001)(wreg)
CLRB	CLEAR BYTE (Очистка байта). Обнуляет значение операнда. (DEST) ← 0	DEST CLRB breg (00010001)(breg)
CLRC	CLEAR CARRY FLAG (Очистка флага переноса). Обнуляет флаг переноса. C ← 0	CLRC (11111000)
CLRVT	CLEAR OVERFLOW-TRAP FLAG (Очистка дополнительного флага переполнения). Обнуляет флаг VT VT ← 0	CLRVT (11111100)
CMP	COMPARE WORDS (Сравнение слов). Вычитает слово источника из слова приёмника. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1». (DEST) – (SRC)	DEST, SRC CMP wreg, waop (100010aa)(waop)(wreg)
CMPB	COMPARE BYTES (Сравнение байтов). Вычитает байт источника из байта приёмника. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1». (DEST) – (SRC)	DEST, SRC CMPB breg, baop (100110aa)(baop)(breg)
CMPL	COMPARE LONG (Сравнение чисел типа LONG). Сравнение величин двух операндов типа double-word (long). Операнды определяются с использованием режима прямой адресации. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1». (DEST) – (SRC)	DEST, SRC CMPL Ireg, Ireg (11000101)(src Ireg) (destIreg)
DEC	DECREMENT WORD (Декремент слова). Декрементирует величину операнда на «1». (DEST) ← (DEST) – 1	DEST DEC wreg (00000101)(wreg)
DECB	DECREMENT BYTE (Декремент байта). Декрементирует величину операнда на «1». (DEST) ← (DEST) – 1	DEST DECB breg (00010101)(breg)
DI	DISABLE INTERRUPTS (Запрещение прерываний). Запрещает прерывания. Запросы на прерывания не удовлетворяются после этой команды. Interrupt Enable (PSW.1) ← 0	DI (11111010)

Продолжение таблицы А.2

1	2	3
DIV	DIVIDE INTEGER (Деление чисел типа INTEGER). Делит содержимое приёмника - операнд типа LONG- INTEGER на содержимое источника - операнд типа INTEGER, используя знаковую арифметику. Частное сохраняет в младшем слове (т. е. в слове с меньшим адресом) приёмника, а остаток – в старшем слове. (low word DEST) ← (DEST)/(SRC) (high word DEST)← (DEST) MOD (SRC)	DEST, SRC DIV Ireg, waop (11111110)(100011aa) (waop)(Ireg)
DIVB	DIVIDE SHORT-INTEGER (Деление чисел типа SHORT INTEGER). Делит содержимое приёмника - операнд типа INTEGER на содержимое источника - операнд типа SHORT-INTEGER, используя знаковую арифметику. Частное сохраняет в младшем байте (т. е. в байте с меньшим адресом) приёмника, а остаток – в старшем байте. (low word DEST) ← (DEST)/(SRC) (high word DEST) ← (DEST)MOD(SRC)	DEST, SRC DIVB wreg, baop (11111110)(100111aa) (baop)(wreg)
DIVU	DIVIDE WORDS, UNSIGNED (Деление чисел типа WORD, незнаковых). Делит содержимое приёмника-операнда типа DOUBLE-WORD на содержимое источника-операнда слова типа WORD, используя беззнаковую арифметику. Частное сохраняет в младшем слове (т. е. в слове с меньшим адресом) приёмника, а остаток – в старшем слове. Следующие две операции проводятся одновременно. (low word DEST) ← (DEST)/(SRC) (high word DEST)← (DEST)MOD(SRC)	DEST, SRC DIVU Ireg, waop (100011aa)(waop)(Ireg)
DIVUB	DIVIDE BYTES, UNSIGNED (Деление чисел типа WORD, незнаковых). Делит содержимое приёмника - операнд типа WORD на содержимое источника - операнд типа BYTE, используя беззнаковую арифметику. Частное сохраняет в младшем байте (т. е. в байте с меньшим адресом) приёмника, а остаток – в старшем байте. Следующие две операции проводятся одновременно. (low byte DEST) ← (DEST)/(SRC) (high byte DEST)← (DEST)MOD(SRC)	DEST, SRC DIVUB wreg, baop (100111aa)(baop)(wreg)

Продолжение таблицы А.2

1	2	3

DJNZ	DECREMENT AND JUMP IF NOT ZERO
	(Декремент и переход при отсутствии нуля).
	Декрементирует величину операнда типа BYTE на «1». Если результат равен «0», управление передаётся следующей по порядку команде. Если результат не равен «0», команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.
	(COUNT) ← (COUNT) - 1
	if (COUNT)≠0 then
	PC ← PC + disp [1)]
	endif
		COUNT,ADDR
		DJNZ breg, cadd
		(11100000)(breg)(disp)

DJNZW	DECREMENT AND JUMP IF NOT ZERO WORD
	(Декремент и переход при отсутствии нуля).
	Декрементирует величину операнда типа WORD на «1». Если результат равен «0», управление передаётся следующей по порядку команде. Если результат не равен «0», команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапозоне от минус 128 до плюс 127.
	(COUNT) ← (COUNT) - 1
	if (COUNT)≠0 then
	PC ← PC + disp [1)]
	endif
		COUNT, ADDR
		DJNZW wreg, cadd
		(11100001)(wreg)(disp)

DPTS	DISABLE PERIPHERAL TRANSACTION SERVER (PTS) (Блокирование периферийного сервера обмена - PTS). Блокирует PTS. PTS Disable (PSW.2) ← 0	DPTS (11101100)
EI	ENABLE INTERRUPTS (Разрешение прерываний). Разрешает прерывания, запрашиваемые после выполнения следующего состояния. Запросы на прерывания не могут удовлетворяться немедленно после выполнения этой команды. Interrupt Enable (PSW.1) ← 1	EI (11111011)
EPTS	ENABLE PERIPHERAL TRANSACTION SERVER (PTS) (Разблокировка периферийного сервера обмена - PTS). Разблокирует PTS. PTS Enable (PSW.2) ← 1	EPTS (11101101)

Продолжение таблицы А.2

1	2	3

EXT	SIGN-EXTEND INTEGER INTO LONG-INTEGER (Знаковое расширение INTEGER в LONG-INTEGER).
	Расширяет со знаком младшее слово операнда до двойного слова,
	if (low word DEST) < 8000H then
	(high word DEST) ← 0
	else
	(high word DEST) ← 0FFFFh
	endif
		EXT Ireg
		(00000110)(Ireg)

EXTB	SIGN-EXTEND SHORT-INSGER INTO INTEGER
	(Знаковое расширение SHORT-INTEGER в INTEGER).
	Расширяет со знаком младший байт операнда до слова.
	if (low byte DEST) < 80H then
	(high byte DEST) ← 0
	else
	(high byte DEST) ←0FFH
	endif
		EXTB wreg
		(00010110)(wreg)

IDLPD	IDLE/POWERDOWN
	В зависимости от 8-битной величины операнда KEY эта команда выбирает:
	- вход в режим IDLE (KEY=1);
	- вход в режим POWERDOWN (KEY=2);
	- выполнить последовательность сброса (любое другое значение KEY, не равное «1» или «2»).
	Контроллер шины завершает цикл упреждающей выборки перед остановкой ЦПУ или сбросом.
	if KEY=1 then
	enter Idle
	else
	if KEY=2 then
	enter POWERDOWN
	else
	execute reset
		IDLPD #key
		(11110110)(key)

INC	INCREMENT WORD (Инкремент слова) Увеличение значение слова операнда на «1». (DEST) ← (DEST)+1	INC wreg (00000111)(wreg)
INCB	INCREMENT BYTE (Инкремент байта) Инкрементирует байт операнда на «1». (DEST) ← (DEST)+1	INCB breg (00010111)(breg)

Продолжение таблицы А.2

1	2	3
JBC	JUMP IF BIT IS CLEAR (Переход при обнуленном бите). Тестирует определённый бит. Если бит установлен, управление передаётся следующей по порядку команде. Если бит обнулен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if (specified bit)=0 then PC ← PC + disp [1)]	JBC breg, bitno, cadd (00110bbb)(breg)(disp)
JBS	JUMP IF BIT IS SET (Переход при установленном бите). Тестирует определённый бит. Если бит обнулен, управление передаётся следующей по порядку команде. Если бит установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if (specified bit)=1 then PC ← PC + disp [1)]	JBS breg, bitno, cadd (00111bbb)(breg)(disp)
JC	JUMP IF CARRY FLAG IS SET (Переход при установленном флаге переноса). Тестирует C - флаг переноса. Если флаг переноса обнулен, управление передаётся следующей по порядку команде. Если флаг переноса установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if C=1 then PC ← PC + disp [1)]	JC cadd (11011011)(disp)
JE	JUMP IF EQUAL (Переход при равенстве). Тестирует Z-флаг нуля. Если флаг обнулен, управление передаётся следующей по порядку команде. Если флаг нуля установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if Z=1 then PC ← PC + disp [1)]	JE cadd (11011111)(disp)

Продолжение таблицы А.2

1	2	3
JGE	JUMP IF SIGNED GREATER THAN OR EQUAL (Переход, если знаковое больше или равно). Тестирует N-флаг отрицательного результата. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг N обнулен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if N=0 then PC ← PC + disp [1)]	JGE cadd (11010110)(disp)
JGT	JUMP IF SIGNED GREATER THAN (Переход, если знаковое больше). Тестирует Z-флаг нуля и N-флаг отрицательного результата. Если один из флагов установлен, управление передаётся следующей по порядку команде. Если оба флага очищены, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if N=0 AND Z=0 then PC ← PC + disp [1)]	JGT cadd (11010010)(disp)
JH	JUMP IF HIGHER (UNSIGNED) (Переход, если больше (беззнаковое)). Тестируются флаги нуля и переноса. Если флаг переноса обнулен или флаг нуля установлен, управление передаётся следующей по порядку команде. Если флаг переноса установлен и флаг нуля очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if C=1 AND Z=0 then PC ← PC + disp [1)]	JH cadd (11011001)(disp)
JLE	JUMP IF SIGNED LESS THAN OR EQUAL (Переход, если знаковое меньше или равно). Тестирует флаги N и C. Если оба флага очищены, управление передаётся следующей по порядку команде. Если один флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if N=1 OR Z=1 then PC ← PC + disp [1)]	JLE cadd (11011010)(disp)

Продолжение таблицы А.2

1	2	3
JLT	JUMP IF SIGNED LESS THAN (Переход, если знаковое меньше). Тестирует флаг N. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if N=1 then PC ← PC + disp [1)]	JLT cadd (11011110)(disp)
JNC	JUMP IF CARRY FLAG IS CLEAR (Переход, если флаг переноса чист). Тестирует флаг C. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if C=0 then PC ← PC + disp [1)]	JNC cadd (11010011)(disp)
JNE	JUMP IF NOT EQUAL (Переход при неравенстве). Тестирует флаг нуля. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if Z=0 then PC ← PC + disp [1)]	JNE cadd (11010111)(disp)
JNH	JUMP IF NOT HIGHER (UNSIGNED) (Переход, если не больше (беззнаковое)). Тестирует флаг нуля и флаг переноса. Если флаг переноса установлен и флаг нуля очищен, управление передаётся следующей по порядку команде. Если флаг переноса установлен или флаг нуля установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if C=0 OR Z=1 then PC ← PC + disp [1)]	JNE cadd (11010001)(disp)

Продолжение таблицы А.2

1	2	3
JNST	JUMP IF STICKY BIT FLAG IS CLEAR (Переход, если флаг ST чист). Тестирует флаг ST (sticky bit). Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if ST=0 then PC ← PC + disp [1)]	JNST cadd (11010000)(disp)
JNV	JUMP IF OVERFLOW FLAG IS CLEAR (Переход, если флаг переполнения чист). Тестирует флаг V (overflow flag). Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if V=0 then PC ← PC + disp [1)]	JNV cadd (11010101)(disp)
JNVT	JUMP IF OVERFLOW-TRAP FLAG IS CLEAR (Переход, если дополнительный флаг переполнения очищен). Тестирует VT-флаг (overflow-trap flag). Если флаг установлен, эта команда очищает флаг, и управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if VT=0 then PC ← PC + disp [1)]	JNVT cadd (11010100)(disp)
JST	JUMP IF STICKY BIT FLAG IS SET (Переход, если флаг ST установлен). Тестируется флаг ST. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг ST установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if ST=1 then PC ← PC + disp [1)]	JST cadd (11011000)(disp)

Продолжение таблицы А.2

1	2	3
JV	JUMP IF OVERFLOW FLAG IS SET (Переход, если установлен флаг переполнения). Тестируется флаг V. Если флаг переполнения очищен, управление передаётся следующей по порядку команде. Если флаг переполнения установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if V=1 then PC ← PC + disp [1)]	JV cadd (11011101)(disp)
JVT	JUMP IF OVERFLOW-TRAP FLAG IS SET (Переход, если дополнительный флаг переполнения установлен). Тестируется флаг VT. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг ловушки переполнения установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if VT=1 then PC ← PC + disp [1)]	JVT cadd (11011100)(disp)
LCALL	LONG CALL (Длинный вызов). Загружает содержимое программного счётчика (адрес возврата) в стек, затем добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Операнд может иметь любой адрес во всём адресном пространстве. SP ← SP - 2 (SP) ← PC PC ← PC + disp	LCALL cadd (11101111)(disp-low) (disp-high)
LD	LOAD WORD (Загрузка слова). Загружает значение слова источника в приёмник. (DEST) ← (SRC)	DEST, SRC LD wreg, waop (101000aa)(waop) (wreg)
LDB	LOAD BYTE (Загрузка байта). Загружает значение байта источника в приёмник. (DEST) ← (SRC)	DEST, SRC LDB breg, baop (101100aa)(baop)(breg)

Продолжение таблицы А.2

1	2	3

LDBSE	LOAD BYTE SIGN-EXTENDED
	(Загрузка байта со знаковым расширением).
	Расширяет знаковое значение операнда источника типа SHORT- INTEGER и загружает его в приёмник типа INTEGER.
	(low byte DEST) ← (SRC)
	if (SRC) < 80H then
	(high byte DEST) ← 0
	else
	(high byte DEST) ← 0FFH
		DEST, SRC
		LDBSE wreg, baop
		(101111aa)(baop)
		(wreg)

LDBZE	LOAD BYTE ZERO-EXTENDED (Загрузка байта, дополненного нулём). Значение операнда источника типа BYTE дополняется нулями и загружается в приёмник типа WORD. (low byte DEST) ← (SRC) (high byte DEST) ← 0	DEST, SRC LDBZE wreg, baop (101011aa)(baop) (wreg)
LJMP	LONG JUMP (Длинный переход). Добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Операнд может иметь любой адрес во всём адресном пространстве. PC ← PC + disp	LJMP cadd (11100111) (disp-low)(disp-high)
MUL (2 операнда)	MULTIPLY INTEGERS (Умножение чисел типа INTEGER). Перемножает операнды источника и приёмника типа INTEGER, используя знаковую арифметику, и сохраняет 32-битный результат в приёмнике типа LONG- INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MUL Ireg, waop (11111110)(010111aa) (waop)(Ireg)
MUL (3 операнда)	MULTIPLY INTEGERS (Умножение чисел типа INTEGER). Перемножает операнды двух источников типа INTEGER, используя знаковую арифметику, и сохраняет 32-битный результат в приёмнике типа LONG-INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MUL Ireg, wreg, waop (11111110)(010011aa) (waop)(wreg) (Ireg)
MULB (2 операнда)	MULTIPLY SHORT-INTEGERS (Умножение чисел типа SHORT-INTEGER). Перемножает операнды источника и приёмника типа SHORT- INTEGER, используя знаковую арифметику, и сохраняет 16-битный результат в приёмнике типа INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULB wreg, baop (11111110)(011111aa) (baop)(wreg)

Продолжение таблицы А.2

1	2	3
MULB (3 операнда)	MULTIPLY SHORT-INTEGERS (Умножение чисел типа SHORT-INTEGER). Перемножает операнды двух источников типа SHORT-INTEGER, используя знаковую арифметику, и сохраняет 16-битный результат в приёмнике типа INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULB wreg, breg, baop (11111110)(010111aa) (baop)(breg) (wreg)
MULU (2 операнда)	MULTIPLY WORDS, UNSIGNED (Умножение чисел типа WORD, беззнаковое). Перемножает операнды источника и приёмника типа WORD, используя безнаковую арифметику, и сохраняет 32-битный результат в приёмнике типа DOUBLE-WORD. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULU Ireg, waop (011011aa)(waop)(Ireg)
MULU (3 операнда)	MULTIPLY WORDS, UNSIGNED (Умножение чисел типа WORD, беззнаковое). Перемножает операнды двух источников типа WORD, используя беззнаковую арифметику, и сохраняет 32-битный результат в приёмнике типа DOUBLE- WORD. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULU Ireg, wreg, waop (010011aa)(waop)(wreg) (Ireg)
MULUB (2 операнда)	MULTIPLY BYTES, UNSIGNED (Умножение чисел типа BYTE, беззнаковое). Перемножает операнды источника и приёмника типа BYTE, используя беззнаковую арифметику, и сохраняет результат типа WORD в приёмнике. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULUB wreg, baop (011111aa)(baop)(wreg)
MULUB (3 операнда)	MULTIPLY BYTES, UNSIGNED (Умножение чисел типа BYTE, незнаковое). Перемножает операнды двух источников типа BYTE, используя беззнаковую арифметику, и сохраняет результат типа WORD в приёмнике. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULUB wreg, breg, baop (010111aa)(baop)(breg) (wreg)
NEG	NEGATE INTEGER (Изменение знака числа типа INTEGER). Изменяет знак операнда типа INTEGER. (DEST) ← – (DEST)	NEG wreg (00000011)(wreg)
NEGB	NEGATE SHORT-INTEGER (Изменение знака числа типа SHORT-NTEGER). Изменяет знак операнда типа SHORT-INTEGER. (DEST) ← – (DEST)	NEGB breg (00010011)(breg)

Продолжение таблицы А.2

1	2	3
NOP	NO OPERATION (Нет операции). Ничего не делает. Управление переходит к следующей по порядку команде.	NOP (11111101)
NORML	NORMALIZE LONG-INTEGER (Нормализация числа типа LONG-INTEGER). Нормализует операнд источника типа LONG-INTEGER (левый операнд). Эта команда сдвигает операнд влево до тех пор, пока его старший значащий бит ≠ «1» или пока не будет совершен 31 сдвиг. Если в старшем значащем бите остался «0» после 31 сдвига, команда останавливает процесс и устанавливает флаг нуля. Команда сохраняет число совершённых сдвигов в приёмнике (правый операнд) (COUNT) ← 0 do while(MSB(SRC) ← 0)AND (COUNT)<31) (SRC) ← (SRC) × 2 (COUNT) ← (COUNT) + 1	SRC, DEST NORML Ireg, breg (00001111)(breg)(Ireg)
NOT	COMPLEMENT WORD (Инверсия числа типа WORD). Инвертирует значение операнда типа word (меняет каждую «1» на «0» и каждый «0» на «1»). (DEST) ← NOT(DEST)	NOT wreg (00000010)(wreg)
NOTB	COMPLEMENT BYTE (Инверсия числа типа BYTE). Инвертирует значение операнда типа byte (меняет каждую «1» на «0» и каждый «0» на «1»). (DEST) ← NOT(DEST)	NOTB breg (00010010)(breg)
OR	LOGICAL OR WORDS (Логическое ИЛИ слов). Проводит операцию ИЛИ над операндом источника типа word и операндом приёмника типа word и меняет первоначальное значение приёмника на результат. Результат равен «1» в каждом разряде, когда хотя бы один из операндов имеет «1». (DEST) ← (DEST)OR(SRC)	DEST, SRC OR breg, waop (100000aa)(waop)(breg)
ORB	LOGICAL OR BYTES (Логическое ИЛИ байтов). Проводит операцию ИЛИ над операндом источника типа byte и операндом приёмника типа byte и меняет первоначальное значение приёмника на результат. Результат равен «1» в каждом разряде, когда хотя бы один из операндов имеет «1». (DEST) ← (DEST)OR(SRC)	DEST, SRC ORB breg, baop (100100aa)(baop)(breg)
POP	POP WORD (Чтение слова из стека). Читает слово из вершины стека и помещает его в приёмник. (DEST) ← (SP) SP ← SP + 2	POP waop (110011aa)(waop)

Продолжение таблицы А.2

1	2	3
POPA	POP ALL (Читать всё из стека). Эта команда используется взамен POPF для поддержки восьми добавочных прерываний. Она читает два слова из стека и помещает первое слово в регистр INT_MASK1/WSR, а второе слово – в PSW/INT_MASK сдвоенный регистр. Эта команда инкрементирует указатель стека SP на четыре. Запросы на прерывания не могут идти сразу после этой команды. INT_MASK1/WSR ← (SP) SP ← SP + 2 PSW/INT_MASK ← (SP) SP ← SP + 2	POPA (11110101)
POPF	POP FLAGS (Чтение флагов из стека). Читает слово из вершины стека и помещает его в PSW. Запросы на прерывания не могут идти сразу после этой команды. (PSW) ← (SP) SP ← SP + 2	POPF (11110011)
PUSH	PUSH WORD (Загрузка слова в стек). Загружает операнд типа word в стек. SP ← SP - 2 (SP) ← (DEST)	PUSH waop (110010aa)(waop)
PUSHA	PUSH ALL (Загрузить всё в стек). Эта команда используется взамен PUSHF, для поддержки восьми добавочных прерываний. Она загружает два слова в стек – PSW/INT_MASK и слово, формируемое сдвоенным регистром INT_MASK1/WSR. Эта команда очищает регистры PSW, INT_MASK и INT_MASK1 и декрементирует SP (указатель стека) на четыре. Запросы на прерывания не могут идти сразу после этой команды. SP ← SP 2 (SP) ← PSW/INT_MASK PSW/INT_MASK ← 0 SP ← SP 2 (SP) ← INT_MASK1/WSR INT_MASK1 ← 0	PUSHA (11110100)
PUSHF	PUSH FLAGS (Загрузка флагов в стек). Загружает PSW в вершину стека, затем устанавливает его в нулевое состояние. Это подразумевает то, что все прерывания запрещены. Запросы на прерывания не могут идти сразу после этой команды. SP ← SP – 2 (SP) ← PSW/INT_MASK PSW/INT_MASK ← 0	PUSHF (11110010)

Продолжение таблицы А.2

RET	RETURN FROM SUBROUTINE (Возврат из подпрограммы). Считывает значение программного счётчика PC из вершины стека. PC ← (SP) SP ← SP + 2	RET (11110000)
RST	RESET SYSTEM (Сброс системы). Инициализирует PSW в 0, PC в 2080H, а SFR – в их начальные значения. Выполнение этой команды ведёт к тому, что контакт RESET# находится в низком состоянии 16 машинных тактов. SFR Reset Status Pin Reset Status PSW ← 0 PC ← 2080H	RST (11111111)
SCALL	SHORT CALL (Короткий вызов). Загружает содержимое PC (адрес возврата) в стек, затем добавляет к PC смещение между концом этой команды и меткой. Смещение должно быть в пределах от минус 1024 до плюс 1023 включительно. SP ← SP – 2 (SP) ← PC PC ← PC + disp [1)]	SCALL cadd (00101xxx)(disp-low)
SETC	SET CARRY FLAG (Установка флага переноса). Устанавливает флаг переноса. C ← 1	SETC (11111001)
SHL	SHIFT WORD LEFT (Сдвиг слова влево). Сдвигает операнд типа word приёмника влево столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31(1FH) включительно. Правые биты результата заполняются нулями. Смещение производится через флаг переноса. Temp ← (COUNT) do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp - 1	SHL wreg, #count (00001001)(count) (wreg) или SHL wreg, breg (00001001)(breg)(wreg)

Продолжение таблицы А.2

1	2	3
SHLB	SHIFT BYTE LEFT (Сдвиг байта влево). Сдвигает операнд типа byte приёмника влево столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика задается непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Правые биты результата заполняются нулями. Смещение производится через флаг переноса. Temp ← (COUNT) do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp - 1	SHLB breg, #count (00011001)(count) (breg) или SHLB breg, breg (00011001)(breg)(breg)
SHLL	SHIFT DOUBLE-WORD LEFT (Сдвиг числа типа DOUBLE- WORD влево). Сдвигает операнд типа double-word приёмника влево столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика задается непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Правые биты результата заполняются нулями. Смещение производится через флаг переноса. Temp ← (COUNT) ; do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp - 1	SHLL Ireg, #count (00001101)(count) (breg) или SHLL Ireg, breg (00001101)(breg)(Ireg)
SHR	LOGICAL RIGHT SHIFT WORD (Логический сдвиг слова вправо). Сдвигает операнд типа word приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика задается непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Левые биты результата заполняются нулями. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, устанавливается флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [2)] Temp ← Temp – 1	SHR wreg, #count (00001000)(count) (wreg) или SHR wreg, breg (00001000)(breg)(wreg)

Продолжение таблицы А.2

1	2	3
SHRA	ARITHMETIC RIGHT SHIFT WORD (Арифметический сдвиг слова вправо). Сдвигает операнд типа word приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH), включительно. Если в оригинале величина старшего бита была «0», сдвигаются нули. Если величина была «1», сдвигаются единицы. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [3)] Temp ← Temp – 1	SHRA wreg, #count (00001010)(count) (wreg) или SHRA wreg, breg (00001010)(breg)(wreg)
SHRAL	LOGICAL RIGHT SHIFT DOUBLE-WORD (Логический сдвиг двойного слова вправо). Сдвигает операнд типа double-word приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Левые биты результата заполняются нулями. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [2)] Temp ← Temp – 1	SHRAL lreg, #count (00001110)(count)(wreg) или SHRAL lreg, breg (00001110)(breg)(wreg)

Продолжение таблицы А.2

1	2	3
SHRAB	ARITHMETIC RIGHT SHIFT BYTE (Арифметический сдвиг слова вправо). Сдвигает операнд типа byte приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH), включительно. Если в оригинале величина старшего бита была «0», сдвигаются нули. Если величина была «1», сдвигаются единицы. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [3)] Temp ← Temp – 1	SHRAB breg, #count (00011010)(count)(breg) или SHRAB breg, breg (00011010)(breg)(breg)
SHRB	LOGICAL RIGHT SHIFT BYTE (Логический сдвиг байта вправо). Сдвигает операнд типа byte приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Левые биты результата заполняются нулями. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [2)] Temp ← Temp - 1	SHRB breg, #count (00011000)(count)(breg) или SHRB breg, breg (00011000)(breg)(breg)

Продолжение таблицы А.2

1	2	3
SHRL	LOGICAL RIGHT SHIFT DOUBLE-WORD (Логический сдвиг вправо числа типа DOUBLOE-WORD). Сдвигает операнд типа double-word приёмника вправо столько раз, сколько установлено специальным операндом-счётчиком. Значение счётчика может быть задано непосредственно, в пределах от 0 до 15 (0FH) включительно, или как содержимое какого-либо регистра со значением в пределах от 0 до 31 (1FH) включительно. Левые биты результата заполняются нулями. Смещение производится через флаг переноса. Эта команда очищает флаг ST в начале команды. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST. Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2 [2)] Temp ← Temp - 1	SHRL Ireg, #count (00001100)(count)(Ireg) или SHRL Ireg, breg (00001100)(breg)(Ireg)
SJMP	SHORT JUMP (Короткий переход). Добавляет к PC смещение между концом этой команды и меткой перехода. Смещение может быть в пределах от минус 1024 до плюс 1023 включительно. PC ← PC + disp [1)]	SJMP cadd (00100xxx)(disp-low)
SKIP	TWO BYTE NO-OPERATION (Двухбайтная пустая команда). Ничего не делает. Управление передаётся следующей по порядку команде. Это двухбайтная NOP, где второй байт – любая величина, которая просто игнорируется	SKIP breg (00000000)(breg)
ST	STORE WORD (Сохранение слова). Сохраняет величину операнда источника типа word (левый операнд) в приёмнике (правый операнд). (DEST) ← (SRC)	SRC, DEST ST wreg,waop (110000aa)(waop)(wreg)
STB	STORE BYTE (Сохранение байта). Сохраняет величину операнда источника типа byte (левый операнд) в приёмнике (правый операнд). (DEST) ← (SRC)	SRC, DEST STB breg, baop (110001aa)(baop)(breg)
SUB (2 операнда)	SUBTRACT WORDS (Вычитание слов). Вычитает операнд источника типа word из операнда приёмника типа word, сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (DEST) – (SRC)	DEST, SRC SUB wreg, waop (011010aa)(waop)(wreg)

Продолжение таблицы А.2

1	2	3
SUB (3 операнда)	SUBTRACT WORDS (Вычитание слов). Вычитает операнд первого источника типа word из операнда второго источника, сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (SRC1) – (SRC2)	DEST, SRC1, SRC2 SUB Dwreg, Swreg, waop (010010aa)(waop)(Swreg)(Dwreg)
SUBB (2 операнда)	SUBTRACT BYTES (Вычитание байтов). Вычитает операнд источника типа byte из операнда приёмника типа byte, сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (DEST) – (SRC)	DEST, SRC SUBB breg, baop (010110aa)(baop)(breg)
SUBB (3 операнда)	SUBTRACT BYTES (Вычитание байтов). Вычитает операнд первого источника типа byte из операнда второго источника, сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (SRC1) – (SRC2)	DEST, SRC1, SRC2 SUBB Dbreg, Sbreg, baop (010110aa)(baop)(Sbreg)(Dbreg)
SUBC	SUBTRACT WORDS WITH BORROW (Вычитание слов с заёмом). Вычитает операнд источника типа word из операнда приёмника типа word. Если флаг переноса установлен, SUBC вычитает «1» из результата. Сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (DEST) – (SRC) – (C)	DEST, SRC SUBC wreg, waop (101010aa)(waop)(wreg)
SUBCB	SUBTRACT BYTES WITH BORROW (Вычитание байтов с заёмом). Вычитает операнд источника типа byte из операнда приёмника типа byte. Если флаг переноса установлен, SUBC вычитает «1» из результата. Сохраняет результат в приёмнике и устанавливает флаг переноса при заёме. (DEST) ← (DEST) – (SRC) – (C)	DEST, SRC SUBCB breg, baop (101110aa)(baop)(breg)

Продолжение таблицы А.2

1	2	3
TIJMP	TABLE INDIRECT JUMP (Табличный косвенный переход). Выполнение продолжается по адресу, выбранному из таблицы адресов. Первый регистр типа word, TBAZE, содержит 16-битный адрес начала таблицы. Второй регистр типа word, INDEX, содержит 16-битный адрес байта, который содержит индекс в таблице. Величина INDEX должна быть в пределах от 0 до 128. Операнд #byte, INDEX_MASK – 8-битные непосредственно получаемые данные по маскированию INDEX. INDEX_MASK суммируется (AND) с INDEX для определения смещения (OFFSET). OFFSET умножается на два, затем добавляется к базовому адресу (TBASE) для определения адреса приёмника (DEST X). [INDEX] AND INDEX_MASK =OFFSET (2×OFFSET)+ TBASE=DEST X PC ← (DEST X)	TIJMP wreg, [wreg], #byte (11100010)(wreg)(#byte) (wreg)
TRAP	SOFTWARE TRAP (Программное пошаговое выполнение). Эта команда ведёт к запросу прерывания, вектор которого расположен в 2010H. Эта операция не влияет на состояние флага разрешения прерываний в PSW(1). Запросы прерываний не могут следовать сразу после этой команды. SP ← SP - 2 (SP) ← PC PC ← (2010H)	TRAP (11110111) Примечание – Этой команды нет в версии 1.2 языка ассемблера 8096. Команда TRAP предназначена для использования средствами Intel. Эти средства не поддерживают применение пользователем этой команды
XCH	EXCHANGE WORD (Обмен слов). Меняет значение операнда источника типа word со значением операнда приёмника типа word. (DEST) ← (SRC); (SRC) ← (DEST)	DEST, SRC XCH wreg, waop (00000100)(waop)(wreg) (00001011)(waop)(wreg)
XCHB	EXCHANGE BYTE (Обмен байтов). Меняет значение операнда источника типа byte со значением операнда приёмника типа byte. (DEST) ← (SRC); (SRC) ← (DEST)	DEST, SRC XCHB breg, baop (00010100)(baop)(breg) (00011011)(baop)(breg)

Окончание таблицы А.2

1	2	3
XOR	LOGICAL EXCLUSIVE-OR WORDS (Логическое «исключающее ИЛИ» слов). Логически складывает операнды типа word источника и приёмника и сохраняет результат в приёмнике. Результат имеет «1» в битовой позиции, если один из операндов (но не оба) имеет «1» в этой позиции, и нули при всех других значениях битовых позиций. (DEST) ← (DEST)XOR(SRC)	DEST, SRC XOR wreg, waop (100001aa)(waop)(wreg)
XORB	LOGICAL EXCLUSIVE-OR BYTES (Логическое «исключающее ИЛИ» байтов). Логически складывает операнды типа byte источника и приёмника и сохраняет результат в приёмнике. Результат имеет «1» в битовой позиции, если один из операндов (но не оба) имеет «1» в этой позиции, и нули при всех других значениях битовых позиций. (DEST) ← (DEST)XOR(SRC)	DEST, SRC XORB breg, baop (100101aa)(baop)(breg)

_____

[1)] Сдвиг (disp) расширяется до 16 бит со знаком.

[2)] В этой операции DEST/2 заменяет деление без знака.

[3)] В этой операции DEST/2 заменяет деление со знаком.

Таблица А.3

Обозначение	Название флага PSW
C	Флаг переноса (Carry Flag)
N	Флаг отрицательного результата (Negative Flag)
ST	Флаг дополнительного бита (Sticky Bit Flag)
V	Флаг переполнения (Overflow Flag)
VT	Дополнительный флаг переполнения (Overflow-Trap Flag)
Z	Флаг нуля (Zero Flag)

Таблица А.4

Символ	Описание
+	Команда устанавливает или сбрасывает флаг, то есть модифицирует, когда 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 неопределённом состоянии

Таблица А.5

Команда	Флаг PSW					
	Z	N	C	V	VT	ST
1	2	3	4	5	6	7
ADD, ADDB	Z	+	+	+	↑	-
ADDC, ADDCB	+	+	+	+	↑	-
AND, ANDB	↓	+	0	0	-	-
BMOV, BMOVI	+	-	-	-	-	-
BR (Косвенный)	-	-	-	-	-	-
CLR, CLRB	-	0	0	0	-	-
CLRC	1	-	0	-	-	-
CLRVT	-	-	-	-	0	-
CMP, CMPB	-	+	+	+	↑	-
CMPL	+	+	+	+	+	-
DEC, DECB	+	+	+	+	↑	-
DI	+	-	-	-	-	-
DIV, DIVB, DIVU, DIVUB	-	-	-	+	↑	-
DJNZ, DJNZW	-	-	-	-	-	-
DPTS	-	-	-	-	-	-
EI	-	-	-	-	-	-
EPTS	-	-	-	-	-	-
EXT, EXTB	-	+	0	0	-	-
IDLPD Верный параметр	+	-	-	-	-	-
IDLPD Неверный параметр	-	0	0	0	0	0
INC	0	+	+	+	↑	0
INCB	+	+	+	+	↑	-
JBC, JBS, JC, JE, JGE, JGT, JH, JLE, JLT, JNC, JNE, JNH, JNST, JNV	+	-	-	-	-	-
JNVT	-	-	-	-	0	-
JST, JV	-	-	-	-	-	-
JVT	-	-	-	-	0	-
LCALL, LD, LDB, LDBSE, LDBZE	-	-	-	-	-	-
LJMP	-	-	-	-	-	?
MUL, MULB, MULU, MULUB	-	-	-	-	-	?
NEG, NEGB	-	+	+	+	↑	-
NOP	+	-	-	-	-	-
NORML	-	?	0	-	-	-
NOT, NOTB	+	+	0	0	-	-
OR, ORB	+	+	0	0	-	-
POP	+	-	-	-	-	-
POPA, POPF	-	+	+	+	+	+
PUSH	+	-	-	-	-	-
PUSHA, PUSHF	-	0	0	0	0	0
RET	0	-	-	-	-	-
RST	-	0	0	0	0	0
SCALL	0	-	-	-	-	-
SETC	-	-	1	-	-	-
SHL, SHLB, SHLL	-	?	+	+	↑	-
SHR	+	0	+	0	-	+

Окончание таблицы А.5

1	2	3	4	5	6	7
SHRA, SHRAB, SHRAL	+	+	+	0	-	+
SHRB, SHRL	+	0	+	0	-	+
SJMP	+	-	-	-	-	-
SKIP	-	-	-	-	-	-
ST, STB	-	-	-	-	-	-
SUB, SUBB	-	+	+	+	↑	-
SUBC, SUBCB	+	+	+	+	↑	-
TIJMP	↓	-	-	-	-	-
TRAP	-	-	-	-	-	-
XCH, XCHB	-	-	-	-	-	-
XOR, XORB	-	+	0	0	-	-

Таблица А.6 – Влияние флагов PSW или тестируемых битов на команды условного перехода

Команда	Переход осуществляется, если	Продолжает, если
DJNZ	декрементированный байт ≠ 0	декрементированный байт = 0
DJNZW	декрементированное слово ≠ 0	декрементированное слово = 0
JBC	выбранный регистровый бит = 0	выбранный регистровый бит = 1
JBS	выбранный регистровый бит = 1	выбранный регистровый бит = 0
JNC	C = 0	C = 1
JNH	C = 0 OR Z = 1	C = 1 AND Z = 0
JC	C = 1	C = 0
JH	C = 1 AND Z = 0	C = 0 OR Z = 1
JGE	N = 0	N = 1
JGT	N = 0 AND Z = 0	N = 1 OR Z = 1
JLT	N = 1	N = 0
JLE	N = 1 OR Z = 1	N = 0 AND Z = 0
JNST	ST = 0	ST = 1
JST	ST = 1	ST = 0
JNV	V = 0	V = 1
JV	V = 1	V = 0
JNVT	VT = 0	VT = 1 (сбрасывает VT)
JVT	VT = 1 (сбрасывает VT)	VT = 0
JNE	Z = 0	Z = 1
JE	Z = 1	Z = 0

В таблице А.7 приводится список кодов команд по возрастанию, с соответствующими обозначениями команд.

Таблица А.7 – Коды команд МК

Шестнадцатерич-ный код	Обозначение команды	Шестнадцатерич-ный код	Обозначение команды
1	2	3	4
00	SKIP	8F	DIVU Indexed
01	CLR	90	ORB Direct
02	NOT	91	ORB Immediate
03	NEG	92	ORB Indirect
04	XCH	93	ORB Indexed
05	DEC	94	XORB Direct

Продолжение таблицы А.7

1	2	3	4
06	EXT	95	XORB Immediate
07	INC	96	XORB Indirect
08	SHR	97	XORB Indexed
09	SHL	98	CMPB Direct
0A	SHRA	99	CMPB Immediate
0B	XCH	9A	CMPB Indirect
0C	SHRL	9B	CMPB Indexed
0D	SHLL	9C	DIVUB Direct
0E	SHRAL	9D	DIVUB Immediate
0F	NORML	9E	DIVUB Indirect
10	Reserved	9F	DIVUB Indexed
11	CLRB	A0	LD Direct
12	NOTB	A1	LD Immediate
13	NEGB	A2	LD Indirect
14	XCHB	A3	LD Indexed
15	DECB	A4	ADDC Direct
16	EXTB	A5	ADDC Immediate
17	INCB	A6	ADDC Indirect
18	SHRB	A7	ADDC Indexed
19	SHLB	A8	SUBC Direct
1A	SHRAB	A9	SUBC Immediate
1B	XCHB	AA	SUBC Indirect
1C-1F	Reserved	AB	SUBC Indexed
20-27	SJMP	AC	LDBZE Direct
28-2F	SCALL	AD	LDBZE Immediate
30-37	JBC	AE	LDBZE Indirect
38-3F	JBS	AF	LDBZE Indexed
40	AND Direct (3 операнда)	B0	LDB Direct
41	AND Immediate (3 операнда)	B1	LDB Immediate
42	AND Indirect (3 операнда)	B2	LDB Indirect
43	AND Indexed (3 операнда)	B3	LDB Indexed
44	ADD Direct (3 операнда)	B4	ADDCB Direct
45	ADD Immediate (3 операнда)	B5	ADDCB Immediate
46	ADD Indirect (3 операнда)	B6	ADDCB Indirect
47	ADD Indexed (3 операнда)	B7	ADDCB Indexed
48	SUB Direct (3 операнда)	B8	SUBCB Direct
49	SUB Immediate (3 операнда)	B9	SUBCB Immediate
4A	SUB Indirect (3 операнда)	BA	SUBCB Indirect
4B	SUB Indexed (3 операнда)	BB	SUBCB Indexed
4C	MULU Direct (3 операнда)	BC	LDBSE Direct
4D	MULU Immediate (3 операнда)	BD	LDBSE Immediate
4E	MULU Indirect (3 операнда)	BE	LDBSE Indirect
4F	MULU Indexed (3 операнда)	BF	LDBSE Indexed
50	ANDB Direct (3 операнда)	C0	ST Direct
51	ANDB Immediate (3 операнда)	C1	BMOV
52	ANDB Indirect (3 операнда)	C2	ST Indirect
53	ANDB Indexed (3 операнда)	C3	ST Indexed
54	ADDB Direct (3 операнда)	C4	STB Direct

Продолжение таблицы А.7

1	2	3	4
55	ADDB Immediate (3 операнда)	C5	CMPL
56	ADDB Indirect (3 операнда)	C6	STB Indirect
57	ADDB Indexed (3 операнда)	C7	STB Indexed
58	SUBB Direct (3 операнда)	C8	PUSH Direct
59	SUBB Immediate (3 операнда)	C9	PUSH Immediate
5A	SUBB Indirect (3 операнда)	CA	PUSH Indirect
5B	SUBB Indexed (3 операнда)	CB	PUSH Indexed
5C	MULUB Direct (3 операнда)	CC	POP Direct
5D	MULUB Immediate (3 операнда)	CD	BMOVI
5E	MULUB Indirect (3 операнда)	CE	POP Indirect
5F	MULUB Indexed (3 операнда)	CF	POP Indexed
60	AND Direct (3 операнда)	D0	JNST
61	AND Immediate (3 операнда)	D1	JNH
62	AND Indirect (3 операнда)	D3	JNC
63	AND Indexed (3 операнда)	D4	JNVT
64	ADD Direct (3 операнда)	D5	JNV
65	ADD Immediate (3 операнда)	D6	JGE
66	ADD Indirect (3 операнда)	D7	JNE
67	ADD Indexed (3 операнда)	D8	JST
68	SUB Direct (3 операнда)	D9	JH
69	SUB Immediate (3 операнда)	DA	JLE
6A	SUB Indirect (3 операнда)	DB	JC
6B	SUB Indexed (3 операнда)	DC	JVT
6C	MULU Direct (3 операнда)	DD	JV
6D	MULU Immediate (3 операнда)	DE	JLT
6E	MULU Indirect (3 операнда)	DF	JE
6F	MULU Indexed (3 операнда)	E0	DJNZ
70	ANDB Direct (3 операнда)	E1	DJNZW
71	ANDB Immediate (3 операнда)	E2	TIJMP
72	ANDB Indirect (3 операнда)	E3	BR (Indirect)
73	ANDB Indexed (3 операнда)	E4	Зарезервировано
74	ADDB Direct (3 операнда)	E5	Зарезервировано
75	ADDB Immediate (3 операнда)	E6	Зарезервировано
76	ADDB Indirect (3 операнда)	E7	LJMP
77	ADDB Indexed (3 операнда)	E8	Зарезервировано
78	SUBB Direct (3 операнда)	E9	Зарезервировано
79	SUBB Immediate (3 операнда)	EA	Зарезервировано
7A	SUBB Indirect (3 операнда)	EB	Зарезервировано
7B	SUBB Indexed (3 операнда)	EC	DPTS
7C	MULUB Direct (3 операнда)	ED	EPTS
7D	MULUB Immediate (3 операнда)	EE	Зарезервировано *
7E	MULUB Indirect (3 операнда)	EF	LCALL
7F	MULUB Indexed (3 операнда)	F0	RET
80	OR Direct	F1	Зарезервировано
81	OR Immediate	F2	PUSHF
82	OR Indirect	F3	POPF
83	OR Indexed	F4	PUSHA
84	XOR Direct	F5	POPA

Окончание таблицы А.7

1	2	3	4
85	XOR Immediate	F6	IDLPD
86	XOR Indirect	F7	TRAP
87	XOR Indexed	F8	CLRC
88	CMP Direct	F9	SETC
89	CMP Immediate	FA	DI
8A	CMP Indirect	FB	EI
8B	CMP Indexed	FC	CLRVT
8C	DIVU Direct	FD	NOP
8D	DIVU Immediate	FE	DIV/DIVB/MUL/MULB**
8E	DIVU Indirect	FF	RST

_____

* Код EE зарезервирован; однако он не генерирует прерывания невыполнимого кода.

** Умножение и деление со знаком – 2-байтные команды. Для любой знаковой команды первый байт – «FE», второй – код соответствующей беззнаковой команды. Например: код MULU (3 операнда) direct – «4C», поэтому код MUL (3 операнда) direct – «FE4C».

В таблице А.8 первая цифра кода команды – по вертикали, вторая – по горизонтали. Соответствующее обозначение команды показано на пересечении двух чисел.

Таблица А.8 – Карта кодов команд МК

Код	x0	x1	x2	x3	x4	x5	x6	x7
0x	SKIP	CLR	NOT	NEG	XCH di	DEC	EXT	INC
1x	*	CLRB	NOTB	NEGB	XCHB di	DECB	EXTB	INCB
2x	SJMP							
3x	JBC							
	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	AND 3op				ADD 3op			
	di	im	in	ix	di	im	in	ix
5x	ANDB 3op				ADDB 3op			
	di	im	in	ix	di	im	in	ix
6x	AND 2op				ADD 2op			
	di	im	in	ix	di	im	in	ix
7x	ANDB 2op				ADDB 2op			
	di	im	in	ix	di	im	in	ix
8x	OR				XOR			
	di	im	in	ix	di	im	in	ix
9x	ORB				XORB			
	di	im	in	ix	di	im	in	ix
Ax	LD				ADDC			
	di	im	in	ix	di	im	in	ix
Bx	LDB				ADDCB			
	di	im	in	ix	di	im	in	ix
Cx	ST	BMOV	ST		STB	CMPL	STB	
	di	-	in	ix	di	-	in	ix
Dx	JNST	JNH	JGT	JNC	JNVT	JNV	JGE	JNE
Ex	DJNZ	DJNZW	TIJMP	BR in	*	*	*	LJMP
Fx	RET	*	PUSHF	POPF	PUSHA	POPA	IDLPD	TRAP

Окончание таблицы А.8

Код	x8	x9	xA	xB	xC	xD	xE	xF
0x	SHR	SHL	SHRA	XCH in	SHRL	SHLL	SHRAL	NORML
1x	SHRB	SHLB	SHRAB	XCHB in	*	*	*	*
2x	SCALL							
3x	JBS							
	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
4x	SUB 3op				MULU 3op **			
	di	im	in	ix	di	im	in	ix
5x	SUBB 3op				MULUB 3op **			
	di	im	in	ix	di	im	in	ix
6x	SUB 2op				MULU 2op **			
	di	im	in	ix	di	im	in	ix
7x	SUBB 2op				MULUB 2op **			
	di	im	in	ix	di	im	in	ix
8x	CMP				DIVU *			
	di	im	in	ix	di	im	in	ix
9x	CMPB				DIVUB **			
	di	im	in	ix	di	im	in	ix
Ax	SUBC				LDBZE			
	di	im	in	ix	di	im	in	ix
Bx	SUBCB				LDBSE			
	di	im	in	ix	di	im	in	ix
Cx	PUSH				POP	BMOVI	POP	
	di	im	in	ix	di	-	in	ix
Dx	JST	JH	JLE	JC	JVT	JV	JLT	JE
Ex	*	*	*	*	DPTS	EPTS	***	LCALL
Fx	CLRC	SETC	DI	EI	CLRVT	NOP	**	RST

_____

* Резервный код команды.

** Умножение и деление со знаком – 2-байтные команды. Для любой знаковой команды первый байт – «FE», второй – код соответствующей незнаковой команды.

*** Код EE зарезервирован; однако он не генерирует прерывания невыполнимого кода.

В таблице А.9 приведён список команд с их длинами и кодами для каждого режима адресации. Каждый режим адресации занимает два столбца. В первом столбце приведены длины команд, а во втором – шестнадцатеричные коды. Для индексных команд в первом столбце длины команд приведены в виде S/L, где S – длина коротко-индексной команды и L – длина длинно-индексной команды. Дефис в любом столбце показывает, что данный способ адресации для данной команды не применим.

Таблица А.9 – Длина команды и шестнадцатеричный код

Арифметическая команда (группа 1)								
Обозначение команды	прямая		непосредственная		косвенная*		индексная S/L*	
1	2		3		4		5	
ADD (2 оп)	3	64	4	65	3	66	4/5	67
ADD (3 оп)	4	44	5	45	4	46	5/6	47
ADDB (2 оп)	3	74	3	75	3	76	4/5	77

Продолжение таблицы А.9

1	2		3		4		5	
ADDB (3 оп)	4	54	4	55	4	56	5/6	57
ADDC	3	A4	4	A5	3	A6	4/5	A7
ADDCB	3	B4	3	B5	3	B6	4/5	B7
CMP	3	88	4	89	3	8A	4/5	8B
CMPB	3	98	3	99	3	9A	4/5	9B
SUB (2 оп)	3	68	4	69	3	6A	4/5	6B
SUB (3 оп)	4	48	5	49	4	4A	5/6	4B
SUBB (2 оп)	3	78	3	79	3	7A	4/5	7B
SUBB (3 оп)	4	58	4	59	4	5A	5/6	5B
SUBC	3	A8	4	A9	3	AA	4/5	AB
SUBCB	4	B8	3	B9	3	BA	4/5	BB
Арифметическая команда (группа 2)								
Обозначение	прямая		непосредственная		косвенная*		индексная S/L*	
DIV	4	FE 8C	5	FE 8D	4	FE 8E	5/6	FE 8F
DIVB	4	FE 9C	4	FE 9D	4	FE 9E	5/6	FE 9F
DIVU	3	8C	4	8D	3	8E	4/5	8F
DIVUB	3	9C	3	9D	3	9E	4/5	9F
MUL (2 оп)	4	FE 6C	5	FE 6D	4	FE 6E	5/6	FE 6F
MUL (3 оп)	5	FE 4C	6	FE 4D	5	FE 4E	6/7	FE 4F
MULB (2 оп)	4	FE 7C	4	FE 7D	4	FE 7E	5/6	FE 7F
MULB (3 оп)	5	FE 5C	5	FE 5D	5	FE 5E	6/7	FE 5F
MULU (2 оп)	3	6C	4	6D	3	6E	4/5	6F
MULU (3 оп)	4	4C	5	4D	4	4E	5/6	4F
MULUB (2 оп)	3	7C	3	7D	3	7E	4/5	7F
MULUB (3 оп)	4	5C	4	5D	4	5E	5/6	5F
Логические команды								
Обозначение	прямая		непосредственная		косвенная*		индексная S/L*	
AND (2 оп)	3	60	4	61	3	62	4/5	63
AND (3 оп)	4	40	5	41	4	42	5/6	43
ANDB (2 оп)	3	70	3	71	3	72	4/4	73
ANDB (3 оп)	4	50	4	51	4	52	5/5	53
OR	3	80	4	81	3	82	4/5	83
ORB	3	90	3	91	3	92	4/5	93
XOR	3	84	4	85	3	86	4/5	87
XORB	3	94	3	95	3	96	4/5	97

Продолжение таблицы А.9

Стековые команды								
Обозначение	прямая		непосредственная		косвенная*		индексная S/L*	
1	2		3		4		5	
POP	2	CC	-	-	2	CE	s	CF
POPA	-	-	-	-	1	F5	-	-
POPF	-	-	-	-	1	F3	-	-
PUSH	2	C8	3	C9	2	CA	S	св
PUSHA	-	-	-	-	1	F4	-	-
PUSHF	-	-	-	-	1	F2	-	-
Команды работы с данными								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
LD	3	A0	4	A1	3	A2	4/5	A3
LDB	3	B0	3	B1	3	B2	4/5	B3
LDBSE	3	BC	3	BD	3	BE	4/5	BF
LDBZE	3	AC	3	AD	3	AE	4/5	AF
ST	3	C0	-	-	3	C2	4/5	C3
STB	3	C4	-	-	3	C6	4/5	C7
XCH	3	04	-	-	-	-	4/5	0B
XCHB	3	14	-	-	-	-	4/5	1B
Команды перехода								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
BR (косв.)	-	-	-	-	2	E3	2	E3
LJMP	-	-	-	-	-	-	-/2	E7
SJMP	-	-	-	-	-	-	2/-	20-27 **
TIJMP	4	E2	4	E2	-	-	-/4	E2
Команды вызова								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
LCALL	-	-	-	-	-	-	-/3	EF
SCALL	-	-	-	-	-	-	2/-	28-2F **
RET	-	-	-	-	1	F0	-	-
TRAP	1	F7	-	-	-	-	-	-
Команды условного перехода								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
DJNZ	-	-	-	-	-	-	3/-	E0
DJNZW	-	-	-	-	-	-	3/-	E1
JBC	-	-	-	-	-	-	3/-	30-37
JBS	-	-	-	-	-	-	3/-	38-3F
JC	-	-	-	-	-	-	1/-	DB
JE	-	-	-	-	-	-	1/-	DF
JGE	-	-	-	-	-	-	1/-	D6
JGT	-	-	-	-	-	-	1/-	D2
JH	-	-	-	-	-	-	1/-	D9
JLE	-	-	-	-	-	-	1/-	DA
JLT	-	-	-	-	-	-	1/-	DE
JNC	-	-	-	-	-	-	1/-	D3
JNE	-	-	-	-	-	-	1/-	D7
JNH	-	-	-	-	-	-	1/-	D1
JNST	-	-	-	-	-	-	1/-	D0

Продолжение таблицы А.9

1	2		3		4		5	
JNV	-	-	-	-	-	-	1/-	D5
JNVT	-	-	-	-	-	-	1/-	D4
JST	-	-	-	-	-	-	1/-	D8
JV	-	-	-	-	-	-	1/-	DD
JVT	-	-	-	-	-	-	1/-	DC
Команды сдвига								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
NORML	3	0F	-	-	-	-	-	-
SHL	3	09	-	-	-	-	-	-
SHLB	3	19	-	-	-	-	-	-
SHLL	3	0D	-	-	-	-	-	-
SHR	3	08	-	-	-	-	-	-
SHRA	3	0A	-	-	-	-	-	-
SHRAB	3	1A	-	-	-	-	-	-
SHRAL	3	0E	-	-	-	-	-	-
SHRB	3	18	-	-	-	-	-	-
SHRL	3	0C	-	-	-	-	-	-
Блочные команды								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
BMOV	-	-	-	-	-	-	-/3	C1
BMOVI	-	-	-	-	-	-	-/3	CD
Специальные команды								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
CLRC	1	F8	-	-	-	-	-	-
CLRVT	1	FC	-	-	-	-	-	-
DI	1	FA	-	-	-	-	-	-
EI	1	FB	-	-	-	-	-	-
IDLPD	-	-	1	F6	-	-	-	-
NOP	1	FD	-	-	-	-	-	-
RST	1	FE	-	-	-	-	-	-
SETC	1	F9	-	-	-	-	-	-
SKIP	2	00	-	-	-	-	-	-
Команды управления PTS								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
DPTS	1	EC	-	-	-	-	-	-
EPTS	1	ED	-	-	-	-	-	-
Остальные команды								
Обозначение	прямая		непосредственная		косвенная *		индексная S/L *	
CLR	-	01	-	-	-	-	-	-
CLRB	-	11	-	-	-	-	-	-
CMPL	-	C5	-	-	-	-	-	-
DEC	-	05	-	-	-	-	-	-
DECB	-	15	-	-	-	-	-	-
EXT	-	06	-	-	-	-	-	-
EXTB	-	16	-	-	-	-	-	-
INC	-	07	-	-	-	-	-	-
INCB	-	17	-	-	-	-	-	-

Окончание таблицы А.9

1	2		3		4		5	
NEG	-	03	-	-	-	-	-	-
NEGB	-	13	-	-	-	-	-	-
NOT	-	02	-	-	-	-	-	-
NOTB	-	12	-	-	-	-	-	-

_____

* Косвенная обычная и косвенная автоинкрементная адресация имеют одинаковые коды команд, также как и коротко- и длинно-индексная адресация. Для использования косвенного обычного или коротко-индексного режима второй байт команды должен быть чётным. Для использования косвенного автоинкрементного или длинно-индексного режима второй байт должен быть нечётным.

** Для этих команд (SCALL, SJMP) три младших значащих бита кода команды соединяются с 8 битами для формирования 11-битного дополнительного кода смещения.

В таблице А.10 приведён список команд в алфавитном порядке по группам, с их временем выполнения, измеренным в State Times – машинных тактах. В таблице время выполнения для косвенного и индексного режимов адресации обозначены как R/M, где R – время выполнения с использованием SFRs, внутреннего RAM или расширенного RAM (0H – BFFH), а M – время выполнения с использованием доступа к внешней памяти или ПЗУ. Время M 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).

Таблица А.10 – Время выполнения команд МК (в машинных тактах)

Арифметическая команда (группа 1)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	длинная
1	2	3	4	5	6	7
ADD (2 ops)	4	4	5/9	6/10	6/10	6/10
ADD (3 ops)	4	4	5/9	6/10	6/10	6/10
ADDB (2 ops)	4	4	5/9	6/10	6/10	6/10
ADDB (3 ops)	4	4	5/9	5/9	6/10	6/10
ADDC	4	4	5/9	5/9	6/10	6/10
ADDCB	4	4	5/9	6/10	6/10	6/10
CMP	4	4	5/9	6/10	6/10	6/10
CMPB	4	4	5/9	6/10	6/10	6/10
SUB (2 ops)	4	4	5/9	6/10	6/10	6/10
SUB (3 ops)	4	4	5/9	6/10	6/10	6/10
SUBB (2 ops)	4	4	5/9	6/10	6/10	6/10
SUBB (3 ops)	4	4	5/9	6/10	6/10	6/10
SUBC	4	4	5/9	6/10	6/10	6/10
SUBCB	4	4	5/9	6/10	6/10	6/10

Продолжение таблицы А.10

1	2	3	4	5	6	7
Арифметические команды (группа 2)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	длинная
DIV	8	8	9/13	10/14	10/14	10/14
DIVB	6	6	7/11	8/12	8/12	8/12
DIVU	8	8	9/13	10/14	10/14	10/14
DIVUB	6	6	7/11	8/12	8/12	8/12
MUL (2 ops)	5	5	6/10	7/11	7/11	7/11
MUL (3 ops)	5	5	6/10	7/11	7/11	7/11
MULB (2 ops)	4	4	5/9	6/10	6/10	6/10
MULB (3 ops)	4	4	5/9	6/10	6/10	6/10
MULU (2 ops)	5	5	6/10	7/11	7/11	7/11
MULU (3 ops)	5	5	6/10	7/11	7/11	7/11
MULUB(2 ops)	4	4	5/9	6/10	6/10	6/10
MULUB(3 ops)	4	4	5/9	6/10	6/10	6/10
Логические команды						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	длинная
AND (2ops)	4	4	5/9	6/10	6/10	6/10
AND (3 ops)	4	4	5/9	6/10	6/10	6/10
ANDB (2 ops)	4	4	5/9	6/10	6/10	6/10
ANDB (3 ops)	4	4	5/9	6/10	6/10	6/10
OR	4	4	5/9	6/10	6/10	6/10
ORB	4	4	5/9	6/10	6/10	6/10
XOR	4	4	5/9	6/10	6/10	6/10
XORB	4	4	5/9	6/10	6/10	6/10
Стековые команды (внутренний стек)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	длинная
POP	3	-	4/8	4/8	4/8	4/8
POPA	3	-	-	-	-	-
POPF	1	-	-	-	-	-
PUSH	2	2	3/7	4/8	4/8	4/8
PUSHA	4	-	-	-	-	-
PUSHF	2	-	-	-	-	-
Стековые команды (внешний стек)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	длинная
POP	7	-	8/13	8/13	8/13	8/13
POPA	7	-	-	-	-	-
POPF	5	-	-	-	-	-
PUSH	6	6	7/12	8/13	8/13	8/13
PUSHA	8	-	-	-	-	-
PUSHF	6	-	-	-	-	-

Продолжение таблицы А.10

1	2	3	4	5	6	7
Команды работы с данными						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	Длинная
LD	2	1	3/7	4/8	4/8	4/8
LDB	2	1	3/7	4/8	4/8	4/8
LDBSE	2	1	3/7	4/8	4/8	4/8
LDBZE	2	1	3/7	4/8	4/8	4/8
ST	2	-	3/7	4/8	4/8	4/8
STB	2	-	3/7	4/8	4/8	4/8
XCH	4	-	-	-	6/10	6/10
XCHB	4	-	-	-	6/10	6/10
Команды перехода						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	Длинная
BR (Indirect)	-	-	2	2	2	2
LJMP	-	-	-	-	-	1
SJMP	-	-	-	-	1	-
TIJMP внутр./внутр. внеш./внутр. внеш./внеш.	 5 9 14	 5 9 14				
Команды вызова (внутренний стек)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	Длинная
LCALL	-	-	-	-	-	3
RET	-	-	2	-	-	-
SCALL	-	-	-	-	3	-
TRAP	1	-	-	-	-	-
Команды вызова (внешний стек)						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	Длинная
LCALL	-	-	-	-	-	7
RET	-	-	4	-	-	-
SCALL	-	-	-	-	7	-
TRAP	5	-	-	-	-	-
Специальные команды						
Обозначение команды	прямая	непосред-ственная	косвенная		индексная	
			обычная	авто-инкрементная	короткая	Длинная
CLRC	1	–	–	–	–	–
CLRVT	1	–	–	–	–	–
DI	1	–	–	–	–	–
EI	1	–	–	–	–	–

Продолжение таблицы А.10

1		2	3	4	5	6	7
IDLPD	неверный параметр	–	5	–	–	–	–
	правильный параметр	–	5	–	–	–	–
NOP		1	–	–	–	–	–
RST (включая выборку байта конфигурации)		40	–	–	–	–	–
SETC		1	–	–	–	–	–
SKIP		1	–	–	–	–	–
Команды управления PTS							
Обозначение команды		прямая	непосред-ственная	косвенная		индексная	
				обычная	авто-инкремент-ная	короткая	длинная
DPTS		1	–	–	–	–	–
EPTS		1	–	–	–	–	–
Остальные команды							
Обозначение команды		прямая	непосред-ственная	косвенная		индексная	
				обычная	авто-инкремент-ная	короткая	длинная
CLR		2	–	–	–	–	–
CLRB		2	–	–	–	–	–
CMPL		6	–	–	–	–	–
DEC		3	–	–	–	–	–
DECB		3	–	–	–	–	–
EXT		4	–	–	–	–	–
EXTB		4	–	–	–	–	–
INC		3	–	–	–	–	–
INCB		3	–	–	–	–	–
NEG		3	–	–	–	–	–
NEGB		3	–	–	–	–	–
NOT		3	–	–	–	–	–
NOTB		3	–	–	–	–	–

Окончание таблицы А.10

Команды условного перехода	
Обозначение команды	Коротко-индексная адресация
DJNZ	4
DJNZW	4
JBC	2
JBS	2
JC	1
JE	1
JGE	1
JGT	1
JH	1
JLE	1
JLT	1
JNC	1
JNE	1
JNH	1
JNST	1
JNV	1
JNVT	1
JST	1
JV	1
JVT	1
Команды сдвига	
Обозначение команды	Прямая адресация
NORML	6
SHL	4 (3 - сдвига нет)
SHLB	4 (3 - сдвига нет)
SHR	4 (3 - сдвига нет)
SHRA	4 (3 - сдвига нет)
SHRAB	4 (3 - сдвига нет)
SHRAL	6 (5 - сдвига нет)
SHRB	4 (3 - сдвига нет)
SHRL	6 (5 - сдвига нет)
Блочные команды	
Обозначение команды	Длинная индексная адресация
BMOV	Внутренняя/внутренняя – 7 на слово Внешняя/внутренняя – 11 на слово Внешняя /внешняя – 16 на слово
BMOVI	Внутренняя /внутренняя – 7 на слово Внешняя / внутренняя – 11 на слово Внешняя/ внешняя – 16 на слово

Приложение Б
(обязательное)

Описание сигналов

В приложении Б содержится информация о функциях выводов МК 1874ВЕ96Т.
В таблице Б.1 приведено описание граф, используемых в таблице Б.2.
В таблице Б.2 дано описание функций выводов.
В таблице Б.3 показано состояние выводов управления и ввода-вывода после сброса.
В таблице Б.4 приведено описание состояния вывода.

Таблица Б.1 – Описание граф таблицы Б.2

Название колонки	Описание
Имя	Описывает функции вывода, расположенные по алфавиту. Многие выводы имеют более чем одну функцию. Каждая функция вывода занесена в эту колонку; если он имеет дополнительные функции, то они описаны в колонке «Дополнительная функция»
Дополнительная функция	Описывает все другие функции, обеспечиваемые этим выводом
Выбор	Описывает установки регистра, состояние вывода или другие условия, которые заставляют вывод функционировать, как описано в колонке "Имя". Тире показывает «нет» или «не применяется»
Тип	Определяет функцию вывода, описанную в колонке «Имя», как вход (I), выход (O), двунаправленный (I/O), квазидвунаправленный (QBD), питание (PWR) или земля (GND). Примечательно, что все входы, за исключением RESET#, являются фиксирующими входами. RESET# является уровнечувствительным входом. Во время режима POWERDOWN схема POWERDOWN использует вход EXTINT как уровнечувствительный вход
Описание	Краткое описание функции вывода, приведенного в колонке «Имя»

Таблица Б.2 – Описание сигналов

Имя	Дополнительная функция	Выбор	Тип	Описание
1	2	3	4	5
ACH0	P0.0/ACHP0	ADCRG.0 = 1 и ADCRB.4 = 1	I	ACH0 –ACH7 являются аналоговыми входами АЦП в режиме однополярных входов. Эти выводы могут использоваться отдельно как аналоговые входы (ACHx) или цифровые входы (P0.x). ∩0V2 и ∩VCC2 выводы должны быть подключены для функционирования АЦП и порта 0
ACH1	P0.1/ACHP1	ADCRG.1 = 1 и ADCRB.4 = 1		
ACH2	P0.2/ACHP2	ADCRG.2 = 1 и ADCRB.4 = 1		
ACH3	P0.3/ACHP3	ADCRG.3 = 1 и ADCRB.4 = 1		
ACH4	P0.4/ACHN0/ PMODE0	ADCRG.4 = 1 и ADCRB.4 = 1		
ACH5	P0.5/ACHN1/ PMODE1	ADCRG.5 = 1 и ADCRB.4 = 1		
ACH6	P0.6/ACHN2/ PMODE2	ADCRG.6 = 1 и ADCRB.4 = 1		
ACH7	P0.7/ACHN3/ EXTINT/ REFOUT/ PMODE3	ADCRG.7 = 1 и ADCRB.4 = 1		

Продолжение таблицы Б.2

1	2	3	4	5
ACHN0 ACHN1 ACHN2 ACHN3	P0.4/ACH4/ PMODE0 P0.5/ACH5/ PMODE1 P0.6/ACH6/ PMODE2 P0.7/ACH7/ EXTINT/ REFOUT/ PMODE3	ADCRG.0 = 0 или ADCRB.4 = 0 ADCRG.1 = 0 или ADCRB.4 = 0 ADCRG.2 = 0 или ADCRB.4 = 0 ADCRG.3 = 0 или ADCRB.4 = 0	I	ACHN0–ACHN3 являются аналоговыми инверсными входами АЦП0 – АЦП3 в режиме дифференциального включения входов
ACHP0 ACHP1 ACHP2 ACHP3	P0.0/ACH0 P0.1/ACH1 P0.2/ACH2 P0.3/ACH3	ADCRG.0 = 0 или ADCRB.4 = 0 ADCRG.1 = 0 или ADCRB.4 = 0 ADCRG.2 = 0 или ADCRB.4 = 0 ADCRG.3 = 0 или ADCRB.4 = 0	I	ACHP0–ACHP3 являются аналоговыми прямыми входами АЦП0 – АЦП3 в режиме дифференциального включения входов
AD0-AD7 AD8-AD15	P3.0-P3.7 P4.0-P4.7	Шина доступа к внешним адресам	I/O	Системная шина адрес-данные. Эти выводы обеспечивают мультиплексированную шину адрес-данные. В течение адресной фазы шинного цикла адресные биты 0-15 выводятся на шину и могут быть зафиксированы с помощью ALE или ADV#. 8- или 16-битные данные передаются в течение фазы передачи данных
ADV#	ALE	CCR.3 = 0	O	Значение адреса. Выходной сигнал становится активным только во время доступа к внешней памяти. ADV# показывает, что значение данного адреса действительно находится на системной шине адрес-данные. Этот сигнал сохраняет низкий уровень до тех пор, пока шинный цикл продолжается, и возвращается в высокий уровень, как только шинный цикл завершен. Внешняя защелка может использовать этот сигнал для выделения адреса с шины адрес/данные. Он также может использоваться с шифратором для выработки сигнала выборки кристалла внешней памяти

Продолжение таблицы Б.2

1	2	3	4	5
AINC#	P2.4/T2RST/ RXD1	EA# = Vea (режим программ-мирования)	I	Автоувеличение на единицу. В режиме управляемого программирования (SLAVE Programming) этот входной сигнал с активным низким уровнем разрешает автоувеличение на единицу (auto-increment). Автоувеличение на единицу позволяет читать или записывать последовательные ячейки EPROM без передачи адреса по шине программирования для каждого цикла чтения или записи
ALE	ADV#	CCR.3 = 1	O	Разрешение функции адреса. Этот выходной сигнал становится активным только во время доступа к внешней памяти (активный уровень - высокий). ALE показывает, что значение адресной информации действительно находится на системной шине адрес/данные и сигнализирует о начале цикла шины. ALE отличается от ADV# тем, что не остается активным в течение всего шинного цикла. Внешняя защелка может использовать этот сигнал для выделения адреса с шины адрес/данные
BHE#	WRH#	CCR.2 = 1	O	Разрешение старшего байта. Это выходной сигнал с активным низким уровнем, устанавливающимся только при записи во внешнюю память слова или старшего байта. BHE# показывает, что данные передаются (записываются) через старшую половину системной шины адрес/данные. Чтобы выбрать записываемый байт памяти, BHE# используется вместе с A0: BHE# A0 Записываемый байт 0 0 оба байта 0 1 только старший байт 1 0 только младший байт
BW	–	CCR.1 = 1	I	Ширина шины. Если CCR.1 = 1, этот сигнал выбирает ширину шины во время внешнего доступа. Когда BW высокий, выбирается 16-битная разрядность шины; когда низкий – 8-битная. Если CCR.1 = 0, ширина шины всегда 8 бит, при этом BW-сигнал игнорируется

Продолжение таблицы Б.2

1	2	3	4	5
CLKOUT	–	IOC3.1	O	Тактовый выход. Выход внутреннего тактового генератора. Частота CLKOUT равна 1/2 частоты на входе (XTAL1) и скважность сигнала CLKOUT равна двум. Очистка IOC3.1 разрешает CLKOUT; установка IOC3.1 запрещает сигнал
CPVER*	P2.6/T2UP-DN/TXD1	EA# =Vea (режим программ-мирования)	O	Общая верификация программы. Этот выходной сигнал с высоким активным уровнем позволяет обнаружить любую ошибку сравнения, которая имела место в режиме программирования. CPVER остается высоким до тех пор, пока не появится ошибка сравнения, в это время он становится низким. Если существует хотя бы одна ошибка, CPVER остается в низком уровне, пока устройство не выйдет из режима программирования. Когда сигнал CPVER имеет высокий уровень, это показывает, что все ячейки запрограммированы верно во время режима программирования
EA#		–	I	Доступ к внешней памяти. Этот сигнал с активным низким уровнем разрешает доступ к памяти вне кристалла. Если уровень высокий, это выбирает внутрикристальный EEPROM. EA# фиксируется только по нарастающему фронту сигнала RESET#

Продолжение таблицы Б.2

1	2	3	4	5
EXTINT	P2.2/PROG#	IOC1.1 = 0	I	Внешнее прерывание. IOC1.1 назначает этот вход или на P2.2 или на P0.7. EXTINT должен удерживаться более чем два машинных такта, чтобы гарантировать его захват. EXTINT – динамический вход (sampled input), однако схема POWERDOWN использует его как чувствительный к уровню вход в режиме POWERDOWN. P2.2 всегда вырабатывает EXTINT1 прерывание (INT13, 203AH). Когда IOC1.1 = 0, нарастающий фронт на выводе P2.2 также вырабатывает прерывание EXTINT (INT07, 200EH). EXTINT и EXTINT1 должны удерживаться более чем два машинных такта, чтобы гарантировать, что они зафиксированы
EXTINT	P0.7/PMODE.3 /ACH7	IOC1.1 = 1	I	Когда IOC1.1 = 1, нарастающий фронт на входе P0.7 вырабатывает внешнее прерывание EXTINT(INT07, 200EH). EXTINT должен удерживаться более чем два машинных такта, чтобы гарантировать его захват
FSADJ	–	–	–	Вывод подстройки выходного тока ЦАП
HSI.0	INT04 прерывание/ источник сброса T2	IOC0.0 = 1	I	Вход для модуля высокоскоростного ввода. HSI.0 вывод может также использоваться для выработки прерывания. Нарастающий фронт на HSI.0 вырабатывает HSI.0 прерывание (INT04, 2008H). HSI.0 должен удерживаться более чем два машинных такта, чтобы гарантировать его захват
HSI.1	Источник тактового сигнала для T2	IOC0.2 = 1	I	Вход для модуля высокоскоростного ввода. Когда IOC0.7 = 0 и IOC3.0 = 0, вывод HSI.1 действует как источник тактов T2
HSI.2	HSO.4	IOC0.4 = 1	I	Вход для модуля высокоскоростного ввода. Примечание – Функции HSI и HSO могут быть активными в одно время, в этом случае вывод действует как выход, управляемый HSO

Продолжение таблицы Б.2

1	2	3	4	5
HSI.3	HSO.5	IOCO.6 = 1	I	Вход для модуля высокоскоростного ввода. Примечание – Функции HSI и HSO могут быть активными в одно время, в этом случае вывод действует как выход, управляемый HSO
HSO.0 – HSO.2	– –	– –	O	Выход для модуля высокоскоростного вывода
HSO.3		–		
HSO.4	HSI.2	IOC1.4 = 1	O	Выход для модуля высокоскоростного вывода Примечание – Функции HSI и HSO могут быть активными в одно время, в этом случае вывод действует как выход, управляемый HSO
HSO.5	HSI.3	IOC1.6 = 1	O	Выход для модуля высокоскоростного вывода. Примечание – Функции HSI и HSO могут быть активными в одно время, в этом случае вывод действует как выход, управляемый HSI
INST	–	–	O	Выборка команды. Этот сигнал действителен только в цикле чтения внешней памяти. Если уровень сигнала высокий, INST показывает, что выбирается команда, если уровень низкий – считываются данные.INST может использоваться в устройствах, которые требуют разделения байтов памяти для данных и команд
IOUTA	–	–	O	Токовый выход ЦАП
IOUTB	–	–	O	Комплементарный выход ЦАП
MISO	P1.6	SPCR.6 = 1 и IOPORT1.6 = 1	I/O	Вход данных ведущего, выход данных ведомого. При разрешенной работе интерфейса SPI вывод функционирует как вход, если установлен режим ведущего, или как выход, если установлен режим ведомого. Используется для приема данных в режиме ведущего или передачи данных в режиме ведомого
MOSI	P1.5	SPCR.6 = 1 и IOPORT1.5 = 1	I/O	Вход данных ведомого, выход данных ведущего. При разрешенной работе интерфейса SPI функционирует как вход, если установлен режим ведомого, или как выход, если установлен режим ведущего. Используется для приема данных в режиме ведомого или передачи данных в режиме ведущего

Продолжение таблицы Б.2

1	2	3	4	5
NMI	–	–	I	Немаскируемое прерывание. Положительный перепад вызывает немаскируемое прерывание через вектор, размещенный в ячейке 203EH. NMI должен удерживаться более чем один машинный такт, чтобы гарантировать его захват. NMI используется системами отладки фирмы Intel, это может привести к конфликтам с программным обеспечением пользователя. Когда NMI не используется, он должен быть подключен к низкому уровню
P0.0 P0.1 P0.2 P0.3 P0.4 P0.5 P0.6 P0.7	ACH0/ACHP0 ACH1/ACHP1 ACH2/ACHP2 ACH3/ACHP3 ACH4/ACHN0/ PMODE0 ACH5/ACHN1/ PMODE1 ACH6/ACHN2/ PMODE2 ACH7/ACHN3/ PMODE3/ REFOUT/ EXTINT	–	I	Порт 0. Этот порт является 8-битным, с высоким сопротивлением, порт только на ввод. Порт 0 только читается через ячейку 0EH в Hwindow0. P0.0 – P0.7 являются цифровыми входами. Эти входы могут индивидуально использоваться на аналоговые входы или цифровые входы (P0.x). Возможно функционирование этих выводов одновременно как аналоговых, так и цифровых входов, но это не рекомендуется из-за того, что чтение порта 0 во время АЦ преобразования может привести к непредсказуемому результату преобразования
P1.0 P1.1 P1.2 P1.3 P1.4 P1.5 P1.6 P1.7	– – SS# PWM1 PWM2 MOSI MISO SCK	– – – IOC3.2 = 0 IOC3.3 = 0 SPCR.6 = 0 SPCR.6 = 0 SPCR.6 = 0	QBD	Порт 1. Это 8-битный квазидвунаправленный порт ввода-вывода. Порт 1 читается и записывается через ячейку 0FH в Hwindow 0. Дополнительные функции SPI порта действуют как выводы стандартного ввода-вывода (не квазидвунаправленные)
P2.0 P2.1 P2.2 P2.3 P2.4 P2.5 P2.6 P2.7	TXD0/PVER RXD0/PALE# EXTINT/ PROG# T2CLK/SCL T2RST/AINC#/RXD1 PWM0 T2UPDN/CPVER/TXD1 T2CAP/PACT/SDA	IOC1.5 = 0 – – – – IOC1.0 = 0 IOC1.5 = 0 SMBCTL2.0=0	O I I I I O QBD QBD	Порт 2. Это многофункциональный 8-битный порт. Порт 2 читается и записывается через ячейки 10H в Hwindow 0

Продолжение таблицы Б.2

1	2	3	4	5
P3.0 – P3.7 P4.0 – P4.7	AD0-AD7 AD8-AD15	EA# = 1	I/O	Порты 3 и 4. Это 8-битные двунаправленные порты ввода–вывода с выходами с открытым стоком. Эти выводы образуют мультиплексированную шину адрес/данные и имеют внутреннюю низкоомную нагрузку на U_{CC1}. Порты 3 и 4 могут быть прочитаны и записаны только как слово; в ячейке 1FFEH. Во время режима программирования эти порты действуют как PBUS
PACT#*	P2.7/T2CAP/SDA	EA# = Vea (режим программи-рования)	O	Активное программирование. В режиме автоматического программирования низкий уровень сигнала PACT# показывает, что осуществляется программирование
PALE#*	P2.1/RXD0	EA# =Vea (режим программи-рования)	I	Программирующий ALE. Когда PALE# установлен, команды и данные с портов 3 и 4 читаются в устройство
PMODE0* PMODE1* PMODE2* PMODE3*	P0.4/ACH4/ACHN0 P0.5/ACH5/ACHN1 P0.6/ACH6/ACHN2 P0.7/ACH7/ACHN3/ REFOUT	EA# =Vea (режим програм-мирования)	I	Выбор режима программирования. Определяют алгоритм программирования EPROM, который будет выполняться. PMODE фиксируется после сброса устройства, когда EA# =Vea; PMODE должны быть стабильными, пока устройство работает
PROG#*	P2.2 /EXTINT	EA# =Vea (режим програм-мирования)	I	Начало программирования. Вход с активным низким уровнем имеет значение только во время подчиненного программирования (Slave Programming Mode). Когда установлен PROG#, это вызывает программирование данных с шины программирования в EPROM. Когда PROG# неактивен, программирующий импульс заканчивается
PVER*	P2.0/TXD0	EA# =Vea (режим програм-мирования)	O	Сравнение программы. В режиме программирования этот выходной сигнал с активным высоким уровнем показывает, что слово запрограммировано верно
PWM0	P2.5	IOC1. 0 = 1	O	Выход широтно-импульсного модулятора 0 (PWM). Если PWM0 специально удерживается в высоком уровне в момент нарастающего фронта RESET#, то устройство входит в режим тестирования
PWM1	P1.3	IOC3.2 = 1	O	Выход PWM1
PWM2	P1.4	IOC3.3 = 1	O	Выход PWM2

Продолжение таблицы Б.2

1	2	3	4	5
RD#	–	–	O	Внешнее чтение. Это выходной сигнал с активным низким уровнем, показывающий чтение внешней памяти
READY	–	–	I	Вход готовности. Этот сигнал используется для удлинения цикла внешней памяти, выработавшей «состояния ожидания» для согласования с медленной памятью. Когда READY высокий, ЦПУ продолжает работать в нормальном режиме. Если READY принимает низкий уровень перед спадающим фронтом сигнала CLKOUT, контроллер памяти вводит циклы ожидания, пока в момент CLKOUT не будет высокого уровня на READY, или до тех пор, пока количество циклов ожиданий не будет равно количеству, запрограммированному в CCR.4 и CCR.5. READY игнорируется для всей внутренней памяти. READY является активным во время выборки CCR
REFOUT	P0.7/ACH7/ACHN3/ EXTINT/PMODE3	ADCRB.6=1	–	Буферизированный выход опорного напряжения АЦП. Возможен одновременный вывод опорного напряжения и включения преобразования восьмого АЦП в режиме прямого хода для калибровки
REFCAP	–	–	–	Вывод подключения внешнего источника опорного напряжения АЦП/Вывод подключения фильтрующего конденсатора источника опорного напряжения АЦП
REFIO	–	–	–	Вывод подключения внешнего источника опорного напряжения ЦАП/Вывод подключения фильтрующего конденсатора источника опорного напряжения ЦАП
RESET#	–	–	I/O	Вход сброса и выход с открытым стоком из кристалла. Спадающий фронт сигнала RESET# инициирует процесс сброса. Когда RESET# устанавливается впервые, кристалл открывает транзистор с нагрузкой на #0V1, соединенный с выводом RESET, на 16 машинных тактов. Эта функция может также быть активизирована переполнением сторожевого таймера или выполнением команды RST. В режиме POWERDOWN процесс сброса вызывает переход кристалла в нормальный режим работы

Продолжение таблицы Б.2

1	2	3	4	5
RXD0	P2.1/PALE#	SP_CON0.3=1	I/O	Последовательный вход данных порта UART0. В режимах 1, 2 и 3 RXD0 используется для приема данных с последовательного порта. В режиме 0 он действует как вход или как выход
RXD1	P2.4/T2RST/AINC#	SP_CON1.3=1	I/O	Последовательный вход данных порта UART1. В режимах 1, 2 и 3 RXD1 используется для приема данных с последовательного порта. В режиме 0 он действует как вход или как выход
SCK	P1.7	SPCR.6 = 1 и IOPORT1.7=1	I/O	Вход/выход тактовой частоты порта SPI. В случае работы в режиме ведущего, вывод работает на выдачу тактового сигнала, в случае работы в режиме ведомого – на прием тактового сигнала
SCL	P2.3/T2CLK	SMBCTL2.0 =1	I/O	Вход/выход тактовой частоты порта I2C. При активизации работы интерфейса I2C функционирует как вход или выход с открытым стоком
SDA	P2.7/T2CAP/PAST#	SMBCTL2.0 =1	I/O	Вход/выход данных порта I2C. При активизации работы интерфейса I2C функционирует как вход или выход с открытым стоком
SS#	P1.2	SPCR.6 = 1	I	Вход разрешает выбор ведомого порта SPI. При активном уровне сигнала и включенном режиме SLAVE, порт SPI будет осуществлять прием/передачу данных
T2CAP	P2.7/PACT#/SDA	–	I	Нарастающий фронт на P2.7 фиксирует значение таймера 2 в регистр T2CAPTURE и вырабатывает прерывание фиксации таймера 2 (Timer Capture) (INT11, 2036H). T2CAP должен удерживаться более чем два машинных такта, чтобы гарантировать захват
T2CLK	P2.3/SCL	IOC0.7 = 0 и IOC3.0 =0 RATE.15=0	I	Вход тактирования таймера 2 и вход генератора, задающего скорость передачи для последовательного порта
T2RST	P2.4/AINC#/RXD1	IOC0.3 = 1 и IOC0.5 =0	I	Сброс таймера 2. Нарастающий фронт на T2RST сбрасывает таймер 2
T2UPDN	P2.6/CPVER	IOC2.1 = 1	I	Управление направлением счета таймера 2. Этот вход с активным высоким уровнем управляет направлением счета таймера 2. Когда T2UP-DN имеет высокий уровень, таймер 2 считает на уменьшение ; когда T2UP-DN имеет низкий уровень, таймер 2 считает на увеличение

Продолжение таблицы Б.2

1	2	3	4	5
TXD0	P2.0/PVER	IOC1.5 = 1	O	Выход последовательных данных порта UART0. В режимах 1, 2 и 3 TXD0 используется для передачи данных последовательного порта. В режиме 0 он используется как выход тактовых импульсов.
TXD1	P2.6/T2UPDN/CPVER	IOC1.5 = 1	O	Выход последовательных данных порта UART1. В режимах 1, 2 и 3 TXD1 используется для передачи данных последовательного порта. В режиме 0 он используется как выход тактовых импульсов.
VPR	–	–	I	Вход в режим программирования/ Вход «старт с альтернативного адреса», а также вход «возврат из режима POWERDOWN»
WR#	WRL#	CCR.2 = 1	O	Внешняя запись. Это выходной сигнал с активным низким уровнем, устанавливающийся при записи во внешнюю память
WRH#	BHE#	CCR.2 = 0	O	Запись старшего байта. В 16-битном шинном режиме WRH# устанавливается при записи старших байта или слова, в 8-битном шинном режиме (CCR.1=0) - при записи старшего и младшего байтов, а также слова
WRL#	WR#	CCR.2 = 0	O	Запись младшего байта. В 16-битном шинном режиме WRL# устанавливается при записи младшего байта или слова, в 8-битном шинном режиме (CCR.1 = 1) – при записи старшего и младшего байтов, а также слова
#VCC1	–	–	PWR	Напряжение питания цифровой части устройства (3,3 В)
∩VCC2	–	–	PWR	Опорное напряжение для АЦП, вывод ∩VCC2 также является напряжением питания для аналоговой части АЦП и логики, используемой для чтения порта 0. Вывод ∩VCC2 должен быть подключен для нормального функционирования АЦП и порта 0

Окончание таблицы Б.2

1	2	3	4	5
∩VCC3	–	–	PWR	Опорное напряжение для ЦАП, ∩VCC3 также является напряжением питания для аналоговой части ЦАП. ∩VCC3 должен быть подключен для нормального функционирования ЦАП
#0V1	–	–	GND	Цифровая часть микросхемы (0 В). Имеется несколько выводов #0V1, все они должны быть соединены
∩0V2	–	–	GND	Аналоговая земля для АЦП (0 В)
∩0V3	–	–	GND	Аналоговая земля для ЦАП (0 В)
XTAL1	–	–	I	Вход инвертора внутрикристального генератора и внутреннего генератора тактов. Если используется внешний генератор, XTAL1 также служит как вход тактов МК
XTAL2	–	–	O	Выход инвертора внутрикристального генератора

_____

* Функция реализуется только при использовании стандартной программы-монитора НИИЭТ, записанного в ПЗУ.

В таблице Б.3 описаны функции МК 1874ВЕ96Т (по умолчанию), состояния выходных управляющих контактов и портов ввода-вывода во время и после сброса.

Таблица Б.3 – Состояние вывода во время и после сброса

Название вывода	Мультиплек-сированные выводы портов	Состояние вывода во время сброса	Состояние вывода после сброса
1	2	3	4
ACH0-ACH7	P0.0-P0.7	Входы [1)]	Входы [1)]
PORT1	P1.0-P1.7	Слабый Pull-Up	Слабый Pull-Up
TXD0	P2.0	Сильный Pull-Up	Выход
RXD0	P2.1	Вход [2)]	Вход [2)]
EXTINT	P2.2	Вход [2)]	Вход [2)]
T2CLK	P2.3	Вход [2)]	Вход [2)]
T2RST	P2.4	Вход [2)]	Вход [2)]
PWM0	P2.5	Сильный Pull-Down	Выход
–	P2.6-P2.7	Слабый Pull-Up	Слабый Pull-Up
AD0-AD15	P3.0-P4.7	Слабый Pull-Up	Шина адрес/данные или порт ввода-вывода [3)]
HSI.0, HSI.1	–	Вход [2)]	Вход [2)]
HSI.2/ HSO.4	–	Вход [2)]	Вход [2)]

Окончание таблицы Б.3

1	2	3	4
HSI.3/ HSO.5	–	Вход [2)]	Вход [2)]
HSO.0- HSO.3	–	Слабый Pull-Down	Выход
ALE	–	Слабый Pull-Up	Выход
BHE#	–	Слабый Pull-Up	Выход
BW	–	Вход [2)]	Вход [2)]
CLKOUT	–	CLKOUT Выход	CLKOUT Выход
EA#	–	Вход [2)]	Вход [2)]
INST	–	Слабый Pull-Up	Выход
NMI	–	Слабый Pull-Down	Слабый Pull-Down
RD#	–	Слабый Pull-Up	Выход
READY	–	Вход [2)]	Вход [2)]
RESET#	–	Сильный Pull-Up	Сильный Pull-Up
WR#	–	Слабый Pull-Up	Выход

[1)] Эти выводы могут быть неподключенными. Однако рекомендуется подключить их к высокому или низкому логическому уровню.

[2)] Эти выводы должны быть подключены и не оставаться плавающими. Входное напряжение не должно превышать уровень $U_{\#VCC}$ в режиме нормальной работы.

[3)] Состояние этих выводов зависит от конфигурации устройства. Если шина адрес/данные активна, выводы действуют как шинные формирователи, иначе они действуют как порт ввода/вывода с открытым стоком и являются неподключенными (плавающими).

Таблица Б.4 – Описания состояния вывода

Состояние вывода	Значение вывода
Слабый Pull-Up	100 мкА
Сильный Pull-Up	1 мА
Слабый Pull-Down	200 мкА
Сильный Pull-Down	500 мкА
Высокий уровень	U_{OH}
Низкий уровень	U_{OL}

Примечание – В таблице приведены приблизительные максимальные значения; они необходимы для общего сведения и не гарантируются. Точные значения некоторых параметров приведены в таблице3.2.

Приложение В
(рекомендуемое)

Регистры

В данном приложении дана справочная информация о регистрах микроконвертера.

В таблице В.1 представлен список модулей и основных элементов МК и связанных с ними конфигурационных регистров и регистров статуса.

Таблица В.1 – Модули и связанные с ними регистры

A/D	HSI	HSO	PTS	PWM
ADCRA ADCRB ADCRC ADCRD ADCRE ADCRF ADCRG ADCRH ADRESULT0 – ADRESULT7 ADCOM0T – ADCOM5T ADCOM0B – ADCOM5B ADCOM6 ADCOM7 ADCOMCL ADCOMCH ADINTCL ADINTCH	HSI_TIME HSI_STATUS HSI_TIME IOC0 IOC1	HSO_COMMAND HSO_TIME IOC1 IOC2 IOS0 IOS1 IOS2	PTSBLOCK PTSCON PTSCOUNT PTSDST PTS_REG PTS_S/D PTSSEL PTSSRC PTSSRV	PWM0_CONTROL PWM1_CONTROL PWM2_CONTROL IOC2 IOC3
INTERRUPTS	I/O PORTS	UART	SPI	I2C
INT_MASK INT_MASK1 INT_PEND INT_PEND1	IOPORT0 IOPORT1 IOPORT2 IOPORT34	BAUD_RATE0 SBUF(RX0) SBUF(TX0) SP_CON0 SP_STAT0 BAUD_RATE1 SBUF(RX1) SBUF(TX1) SP_CON1 SP_STAT1	SPCR SPSR SPDR	SMBSDA SMBST SMBCST SMBCTL1 SMBADDR SMBCTL2 SMBTOPR SMBCTL3
TIMER 1	TIMER 2	Chip Config	Stack	Watchdog
TIMER1 IOC1 IOS1	TIMER2 T2CAPTURE IOC0 IOC1 IOC2 IOC3	CCR	SP	WATCHDOG

Окончание таблицы В.1

A/D	HSI	HSO	PTS	PWM
Window Select	Zero	DAC	DEBUG	EEPROM
WSR	ZERO_REG	DACVAL	DEBPC DEBPCEQ DEBDAT DEBDATL DEBDATH DEBADDR DEBCTRL CURRPC HAPPPC CLKCLDEB CLKCHDEB	ROMC PPW PPSETUP PPHOLD
Clock control	PRNG			
CLKCL CLKCH	PRNGCON PRNGREG1 PRNGREG2 PRNGRES			

В таблице В.2 представлен список горизонтальных окон (HWindow) со специальными функциональными регистрами (SFR) и функциями, имеющимися в них.

Таблица В.2 – Регистры специальных функций (SFR) горизонтальных окон (HWindows)

Шестнадцатерич-ный код	Чтение/ Запись	HWindow 0	HWindow 1	HWindow 15
1	2	3	4	5
00	Чтение Запись	ZERO_REG (LO) ZERO_REG (LO)	ZERO_REG (LO) ZERO_REG (LO)	ZERO_REG (LO) ZERO_REG (LO)
01	Чтение Запись	ZERO_REG (HI) ZERO_REG (HI)	ZERO_REG (HI) ZERO_REG (HI)	ZERO_REG (HI) ZERO_REG (HI)
02	Чтение Запись			
03	Чтение Запись	HSI_MODE	Зарезервирован	HSI_MODE
04	Чтение Запись	HSI_TIME (LO) HSO_TIME (LO)	PTSSEL (LO) PTSSEL (LO)	HSO_TIME (LO) HSI_TIME (LO)
05	Чтение Запись	HSI_TIME (HI) HSO_TIME (HI)	PTSSEL (HI) PTSSEL (HI)	HSO_TIME (HI) HSI_TIME (HI)
06	Чтение Запись	HSI_STATUS HSO_COMMAND	PTSSRV (LO) PTSSRV (LO)	HSO_COMMAND HSI_STATUS, биты 0, 2, 4, 6
07	Чтение Запись	SBUF (RX) SBUF (TX)	PTSSRV (HI) PTSSRV (HI)	SBUF (TX) SBUF (RX)
08	Чтение Запись	INT_MASK INT_MASK	INT_MASK INT_MASK	INT_MASK INT_MASK
09	Чтение Запись	INT_PEND INT_PEND	INT_PEND INT_PEND	INT_PEND INT_PEND

Окончание таблицы В.2

1	2	3	4	5
0A	Чтение Запись	TIMER1 (LO) WATCHDOG	Зарезервирован	WATCHDOG TIMER1 (LO)
0B	Чтение Запись	TIMER1 (HI) IOC2	Зарезервирован	IOC2, кроме бита 7 [1)] TIMER1 (HI)
0C	Чтение Запись	TIMER2 (LO) TIMER2 (LO)	IOC3 IOC3	T2CAPTURE (LO) T2CAPTURE (LO)
0D	Чтение Запись	TIMER2 (HI) TIMER2 (HI)	Зарезервирован	T2CAPTURE (HI) T2CAPTURE (HI)
0E	Чтение Запись	IOPORT0 BAUD_RATE	Зарезервирован	Зарезервирован
0F	Чтение Запись	IOPORT1 IOPORT1	Зарезервирован	Зарезервирован
10	Чтение Запись	IOPORT2 IOPORT2	DACVAL(LO)	Зарезервирован
11	Чтение Запись	SP_STAT SP_CON	DACVAL(HI)	INT_PEND1 INT_PEND1
12	Чтение Запись	INT_PEND1 INT_PEND1	INT_PEND1 INT_PEND1	INT_PEND1 INT_PEND1
13	Чтение Запись	INT_MASK1 INT_MASK1	INT_MASK1 INT_MASK1	INT_MASK1 INT_MASK1
14	Чтение Запись	WSR WSR	WSR WSR	WSR WSR
15	Чтение Запись	IOS0 IOC0	Зарезервирован	IOC0, кроме бита 1[1)] IOS0, кроме битов 6 и 7
16	Чтение Запись	IOS1 IOS1	PWM2_CONTROL PWM2_CONTROL	IOC1 IOS1, кроме битов 6 и 7 [2)]
17	Чтение Запись	IOS2 PWM0_CONTROL	PWM1_CONTROL PWM1_CONTROL	PWM0_CONTROL IOS2 [2)]
18	Чтение Запись	SP (LO) SP (LO)	SP (LO) SP (LO)	SP (LO) SP (LO)
19	Чтение Запись	SP (HI) SP (HI)	SP (HI) SP (HI)	SP (HI) SP (HI)

[1)] IOC2.7 (седьмой бит регистра IOC2) и IOC0.1 не снабжены защёлкой и будут читаться как «1».

[2)] Запись в регистры SP_STAT, IOS1 и IOS2 в горизонтальном окне 15 устанавливает биты статуса, но не вызывает прерывания.

В таблице В.3 представлен список SFR, расположенных в алфавитном порядке, мнемоник с названиями, адресами, состояниями после сброса.

Таблица В.3 – Имена SFRs, адреса SFRs и величины SFRs после сброса

Обозначение регистра	Название регистра	Шестнадца-теричный код	Значение в двоичном коде после сброса
1	2	3	4
ADCOM0B	Регистр нижней границы 0	1F70, 1F71	0000 0000 0000 0000
ADCOM1B	Регистр нижней границы 1	1F72, 1F73	0000 0000 0000 0000
ADCOM2B	Регистр нижней границы 2	1F74, 1F75	0000 0000 0000 0000
ADCOM3B	Регистр нижней границы 3	1F76, 1F77	0000 0000 0000 0000
ADCOM4B	Регистр нижней границы 4	1F78, 1F79	0000 0000 0000 0000
ADCOM5B	Регистр нижней границы 5	1F7A, 1F7B	0000 0000 0000 0000
ADCOM0T	Регистр верхней границы 0	1F80, 1F81	0000 0000 0000 0000
ADCOM1T	Регистр верхней границы 1	1F82, 1F83	0000 0000 0000 0000
ADCOM2T	Регистр верхней границы 2	1F84, 1F85	0000 0000 0000 0000
ADCOM3T	Регистр верхней границы 3	1F86, 1F87	0000 0000 0000 0000
ADCOM4T	Регистр верхней границы 4	1F88, 1F89	0000 0000 0000 0000
ADCOM5T	Регистр верхней границы 5	1F8A, 1F8B	0000 0000 0000 0000
ADCOM6	Регистр границы 6	1F8C, 1F8D	0000 0000 0000 0000
ADCOM7	Регистр границы 7	1F8E, 1F8F	0000 0000 0000 0000
ADCOMCL	Регистр управления цифровыми компараторами 1	1F7E	0000 0000
ADCOMCH	Регистр управления цифровыми компараторами 2	1F7F	0000 0000
ADCRA	Регистр CRA АЦП	1FC0	0000 0110
ADCRB	Регистр CRB АЦП	1FC1	0011 0010
ADCRC	Регистр CRC АЦП	1FC2	0000 0000
ADCRD	Регистр CRD АЦП	1FC3	0000 0000
ADCRE	Регистр CRE АЦП	1FC4	0000 0000
ADCRF	Регистр CRF АЦП	1FC5	0000 0000
ADCRG	Регистр CRG АЦП	1FC6	0000 0000
ADCRH	Регистр CRH АЦП	1FC7	0000 0000
ADINTCL	Регистр управления прерываниями цифровых компараторов 1	1F7C	0000 0010
ADINTCH	Регистр управления прерываниями цифровых компараторов 2	1F7D	0000 0010
ADRESULT0	Регистр результата канала 0	1FD0,1FD1	0000 0000 0000 0000
ADRESULT1	Регистр результата канала 1	1FD2,1FD3	0000 0000 0000 0000
ADRESULT2	Регистр результата канала 2	1FD4,1FD5	0000 0000 0000 0000
ADRESULT3	Регистр результата канала 3	1FD6,1FD7	0000 0000 0000 0000
ADRESULT4	Регистр результата канала 4	1FD8,1FD9	0000 0000 0000 0000
ADRESULT5	Регистр результата канала 5	1FDA,1FDB	0000 0000 0000 0000
AD_RESULT6	Регистр результата канала 6	1FDC,1FDD	0000 0000 0000 0000
AD_RESULT7	Регистр результата канала 7	1FDE,1FDF	0000 0000 0000 0000
BAUD_RATE0	Регистр скорости передачи UART0	0E	0000 0000
BAUD_RATE1	Регистр скорости передачи UART1	1FA6	0000 0000

Продолжение таблицы В.3

1	2	3	4
CLKCL	Регистр управления тактовыми сигналами устройств 1	1FF0	0011 1111
CLKCLDEB	Регистр перезагрузки CLKCL	1FF6	0000 0000
CLKCH	Регистр управления тактовыми сигналами устройств 2	1FF1	1111 1111
CLKCHDEB	Регистр перезагрузки CLKCH	1FF7	0000 0000
CURRPC	Регистр текущего значения PC	1FE8,1FE9	XXXX XXXX XXXX XXXX
DACVAL	Регистр значения тока ЦАП	10,11	1100 0000 0000 0000
DEBADDR	Регистр задания значения адреса операнда DEBUG	1FE4,1FE5	0000 0000 0000 0000
DEBCTRL	Регистр управления событиями DEBUG 1	1FE6	0000 0000
DEBCTRH	Регистр управления событиями DEBUG 2	1FE7	0000 0000
DEBDAT	Регистр задания данных DEBUG	1FE2,1FE3	0000 0000 0000 0000
DEBDATH	Регистр задания старшего байта данных DEBUG	1FEF	0000 0000
DEBDATL	Регистр задания младшего байта данных DEBUG	1FEE	0000 0000
DEBPC	Регистр задания значения основного счетчика команд 1 DEBUG	1FE0,1FE1	0000 0000 0000 0000
DEBPCEQ	Регистр задания значения основного счетчика команд 2 DEBUG	1FEC,1FED	0000 0000 0000 0000
HAPPPC	Регистр хранения PC	1FEA,1FEB	XXXX XXXX XXXX XXXX
HSI_MODE	Регистр режима высокоскоростного ввода	03	1111 1111
HSI_STATUS	Регистр статуса высокоскоростного ввода	06	X0X0 X0X0
HSI_TIME	Регистр времени высокоскоростного ввода	04,05	XXXX XXXX XXXX XXXX
HSO_COMMAND	Регистр управления высокоскоростным выводом	06	XXXX XXXX
HSO_TIME	Регистр времени высокоскоростного вывода	04,05	XXXX XXXX XXXX XXXX
INT_MASK	Регистр маски прерывания	08	0000 0000
INT_MASK1	Регистр маски прерывания 1	13	0000 0000
INT_PEND	Регистр хранения прерывания	09	0000 0000
INT_PEND1	Регистр хранения прерывания 1	12	0000 0000
IOC0	0 регистр управления вводом–выводом	15	0000 00X0
IOC1	1 регистр управления вводом-выводом	16	0010 0001
IOC2	2 регистр управления вводом–выводом	0B	X000 0000

Продолжение таблицы В.3

1	2	3	4
IOC3	3 регистр управления вводом–выводом	0C	1111 0000
IOPORT0	Регистр порта 0	0E	XXXX XXXX
IOPORT1	Регистр порта 1	0F	1111 1111
IOPORT2	Регистр порта 2	10	1100 0001
IOPORT34	Регистр портов 3, 4	1FFE,1FFF	1111 1111 1111 1111
IOS0	Регистр статуса ввода-вывода 0	15	0000 0000
IOS1	Регистр статуса ввода-вывода 1	16	0000 0000
IOS2	Регистр статуса ввода-вывода 2	17	0000 0000
PPW	Регистр длительности программирующего импульса	1FF2,1FF3	0001 0011 1000 1000
PPSETUP	Регистр управления временем установки программирующего импульса	1FF4	0001 0100
PPHOLD	Регистр управления временем удержания программирующего импульса	1FF5	0001 0100
PRNGCON	Регистр управления PRNG	1F64	0000 0000
PRNGREG1	Регистр ключа 0 PRNG	1F61,1F60	0000 0000 0000 0000
PRNGREG2	Регистр дключа 1 PRNG	1F63,1F62	0000 0000 0000 0000
PRNGRES	Регистр результата PRNG	1F67,1F66	0000 0000 0000 0000
PTSSEL	Регистр выбора PTS	04,05	0000 0000 0000 0000
PTSSRV	Регистр обслуживания PTS	06,07	0000 0000 0000 0000
PWM0_CONTROL	Регистр управления PWM0	17	0000 0000
PWM1_CONTROL	Регистр управления PWM1	16	0000 0000
PWM2_CONTROL	Регистр управления PWM2	17	0000 0000
ROMC	Регистр управления операциями EEPROM	1FFC,1FFD	0000 0000 00X0 0000
SBUF(RX0/TX0)	Буфер последовательного порта UART0	07	0000 0000
SBUF(RX1)	Буфер приема последовательного порта UART1	1FAB	0000 0000
SBUF(TX1)	Буфер передачи последовательного порта UART1	1FA9	0000 0000
SMBADDR	Регистр собственного адреса I2C	1FB8	0000 0000
SMBCST	Регистр управления и статуса I2C	1FB4	0000 0000
SMBCTL1	Регистр управления 1 I2C	1FB6	0000 0000
SMBCTL2	Регистр управления 2 I2C	1FBA	0000 0000
SMBCTL3	Регистр управления 3 I2C	1FBE	0000 0000
SMBSDA	Сдвиговый регистр данных I2C	1FB0	XXXX XXXX
SMBST	Регистр статуса I2C	1FB2	0000 0000

Окончание таблицы В.3

1	2	3	4
SMBTOPR	Регистр прескалера времени ожидания SCL	1FBC	0000 0000
SP	Указатель стека	18,19	0001 1101 0001 1000
SPCR	Регистр управления SPI	1FA0	0000 0100
SPDR	Регистр данных SPI	1FA4	0000 0000
SPSR	Регистр статуса SPI	1FA2	0000 0000
SP_CON0	Регистр управления последовательным портом UART0	11	0000 1011
SP_CON1	Регистр управления последовательным портом UART1	1FAF	0000 1011
SP_STAT0	Регистр статуса последовательного порта UART0	11	0000 1011
SP_STAT1	Регистр статуса последовательного порта UART1	1FAD	0000 1011
T2CAPTURE	Регистр захвата таймера 2	0C,0D	XXXX XXXX XXXX XXX
TIMER1	Регистр значения таймера 1	0A,0B	0000 0000 0000 0000
TIMER2	Регистр значения таймера 2	0C,0D	0000 0000 0000 0000
WATCHDOG	Регистр сторожевого таймера (WDT)	0A	XXXX XXXX
WSR	Регистр выбора окна	14	XXXX 0000
ZERO_REG	Регистр нуля	00,01	0000 0000 0000 0000

В таблице В.4 представлен список прерываний МК, расположение векторов для нормальных и PTS прерываний, приоритеты прерываний.

Таблица В.4 – Источник векторов прерывания, расположение в памяти и приоритеты у МК комплекта

Номер прерывания	Название прерывания	Источник	Вектор прерывания	Вектор для PTS	Приоритет [1)]
1	2	3	4	5	6
Специальный	Unimplemented Opcode	Неисполняемый код	2012H	–	–
Специальный	Software Trap	Команда TRAP	2010H	–	–
INT16	DEBUG	Модуль отладки	2060H	–	16
INT15	NMI [2)]	NMI	203EH	–	15
INT14	HSI FIFO Full	Очередь HSI заполнена	203CH	205CH	14
INT13	EXTINT1 [2)]	P2. 2	203AH	205AH	13
INT12	Timer 2 Overflow	Переполнение таймера 2	2038H	2058H	12
INT11	Timer 2 Capture [2)]	Захват таймера 2	2036H	2056H	11

Окончание таблицы В.4

1	2	3	4	5	6
INT10	HSI FIFO 4	Четвертый вход HSI FIFO	2034H	2054H	10
INT09	Receive UART	Флаги RI UART0 и UART1	2032H	2052H	9
INT08	Transmit UART	Флаги TI UART0 и UART1	2030H	2050H	8
INT07	EXTINT [2)]	P2.2 или P0.7	200EH	204EH	7
INT06	Serial Ports	RI и TI флаги UART0 и UART1, SPIF флаг SPI, INT флаг I2C	200CH	204CH	6
INT05	Software Timer	Программные таймеры 0 – 3. Сброс таймера 2. Начало АЦП	200AH	204AH	5
INT04	HSI.0 Pin [2)]	HSI.0	2008H	2048H	4
INT03	HSO	HSO.0 – HSO.5	2006H	2046H	3
INT02	HSI Data Available	Очередь HSI заполнена или регистр захвата HSI загружен	2004H	2044H	2
INT01	A/D Conversion	АЦ преобразование завершено или сработал цифровой компаратор	2002H	2042H	1
INT00	Timer Overflow	Переполнение таймера 1 или таймера 2	2000H	2040H	0

Примечание – Любое из маскируемых прерываний INT14 – INT00 может быть определено с помощью PTS. Любое PTS-прерывание имеет приоритет над всеми другими маскируемыми прерываниями.

[1)] Неисполняемый код и запрос на прерывание от программного обеспечения не имеют приоритета. Они сразу поступают в контроллер прерываний для обслуживания. DEBUG имеет наивысший приоритет из всех приоритизированных прерываний. Любое PTS-прерывание имеет приоритет перед всеми остальными маскируемыми прерываниями.

[2)] Эти прерывания могут считаться независимыми внешними прерываниями.

В таблице В.5 представлен список источников прерывания с их векторами прерывания, битами маски, разрешающими прерывания, и биты выбора источника.

Таблица В.5

Источник прерывания	Вектор прерывания	Номер прерывания	Бит, разрешающий прерывание [1)]	Выбор источника [2)]
1	2	3	4	5
АЦ преобразование завершено	A/D Conversion	INT01	INT_MASK.1	ADCRA.2
Сработал цифровой компаратор 0	A/D Conversion	INT01	INT_MASK.1	ADINTCL.0
Сработал цифровой компаратор 1	A/D Conversion	INT01	INT_MASK.1	ADINTCL.2
Сработал цифровой компаратор 2	A/D Conversion	INT01	INT_MASK.1	ADINTCL.4
Сработал цифровой компаратор 3	A/D Conversion	INT01	INT_MASK.1	ADINTCL.6
Сработал цифровой компаратор 4	A/D Conversion	INT01	INT_MASK.1	ADINTCH.0
Сработал цифровой компаратор 5	A/D Conversion	INT01	INT_MASK.1	ADINTCH.2
Сработал цифровой компаратор 6	A/D Conversion	INT01	INT_MASK.1	ADINTCH.4
Сработал цифровой компаратор 7	A/D Conversion	INT01	INT_MASK.1	ADINTCH.6
Начало АЦП	Software Timer	INT05	INT_MASK.5	–
Четвертый вход HSI FIFO	HSI FIFO 4	INT10	INT_MASK1.2	–
HSI FIFO Full	HSI FIFO Full	INT14	INT_MASK1.6	–
	HSI Data Available	INT02	INT_MASK.2	IOC1.7 = 1
Регистр захвата HSI загружен	HSI Data Available	INT02	INT_MASK.2	IOC1.7 = 0
Контакт HSI.0	HSI. 0	INT04	INT_MASK.4	–
HSI.0 - HSI.5	High-Speed Output	INT03	INT_MASK.3	–
NMI	NMI	INT15	–	–
Счетчик команд больше DEBPC	DEBUG	INT16	–	DEBCTRL.1=0, DEBCTRL.0= 1
На шине данных значение DEBDAT	DEBUG	INT16	–	DEBCTRL.3=0, DEBCTRL.2= 1
На шине адресов значение DEBADDR	DEBUG	INT16	–	DEBCTRL.5=0, DEBCTRL.4= 1
Счетчик команд равен DEBPCEQ	DEBUG	INT16	–	DEBCTRL.9=0, DEBCTRL.8= 1
Обращение к внешней памяти	DEBUG	INT16	–	DEBCTRL.7=0, DEBCTRL.6= 1

Окончание таблицы В.5

1	2	3	4	5
На шине данных младший байт DEBDATL	DEBUG	INT16	–	DEBCTRL.11=0, DEBCTRL.10= 1
На шине данных старший байт DEBDATH	DEBUG	INT16	–	DEBCTRL.13=0, DEBCTRL.12= 1
P0.7	EXTINT	INT07	INT_MASK.7	IOC1.1 = 1
P2.2	EXTINT	INT07	INT_MASK.7	IOC1.1 = 0
	EXTINT1	INT13	INT_MASK1.5	–
Флаг RI UART0	Receive UART	INT09	INT_MASK1.1	–
	Serial Ports	INT06	INT_MASK.6	–
Флаг RI UART1	Receive UART	INT09	INT_MASK1.1	–
	Serial Ports	INT06	INT_MASK.6	–
Программные таймеры 0 – 3	Software Timer	INT05	INT_MASK.5	–
Флаг TI UART0	Transmit UART	INT08	INT_MASK1.0	–
	Serial Ports	INT06	INT_MASK.6	–
Флаг TI UART1	Transmit UART	INT08	INT_MASK1.0	–
	Serial Ports	INT06	INT_MASK.6	–
Флаг SPIF порта SPI	Serial Ports	INT06	INT_MASK.6	–
Флаг INT порта I2C	Serial Ports	INT06	INT_MASK.6	–
Переполнение таймера 1	Timer Overflow	INT00	INT_MASK.0	IOC1.2 = 1
Захват таймера 2	Timer 2 Capture	INT11	INT_MASK1.4	–
Переполнение таймера 2	Timer Overflow	INT00	INT_MASK.0	IOC1.3 = 1
	Timer 2 Overflow	INT12	INT_MASK1.4	–
Сброс таймера 2	Software Timer	INT05	INT_MASK.5	–
Команда TRAP	Software TRAP	Специ-альный	–	–
Неисполняемый код	Unimplemented Opcode	Специ-альный	–	–

———

[1)] В этой колонке для каждого источника прерываний приводится список битов маски, которые должны быть установлены, чтобы это прерывание было возможным. Три источника прерываний могут запрашивать по два различных прерывания – 8096BH-совместимое прерывание и новое, отдельное прерывание (HSI FIFO Full, EXTINT1, Timer 2 Overflow). В любом случае для одного источника возможно только одно прерывание. Биты маски могут устанавливаться только для одного из двух возможных прерываний.

[2)] Три из 8096BH-совместимых прерываний (HSI Data Available, EXTINT и Timer Overflow) могут запрашиваться двумя источниками. В этой колонке приведены значения битов регистра IOC1 для выбора источника прерывания.

Далее приведено описание регистров микроконвертера, форматы регистров (рисунки В.1 – В.118) и функциональные назначения полей регистров в таблицах В.6 – В.133.

ADCOM0B Регистр нижней границы цифрового компаратора 0

1F71H/1F70H (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 0. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT0 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM0T.

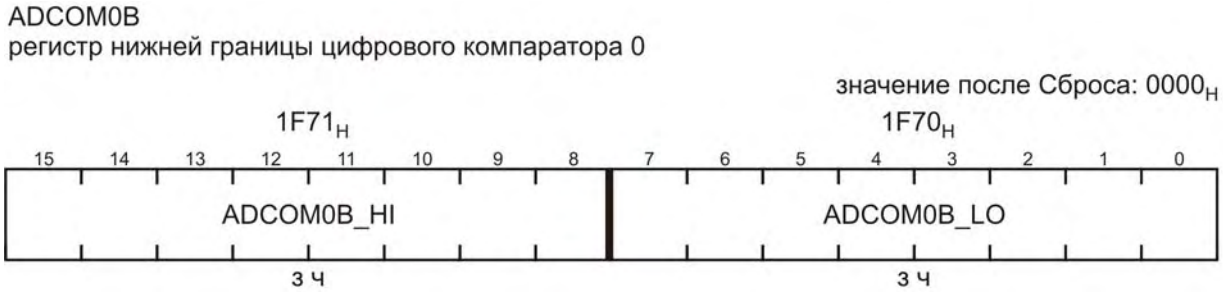


Рисунок В.1 – Формат регистра ADCOM0B

Таблица В.6 – Функциональные назначения полей регистра ADCOM0B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM0B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 0
8-15	ADCOM0B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 0

ADCOM1B Регистр нижней границы цифрового компаратора 1

1F73H/1F72H (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 1. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT1 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM1T.

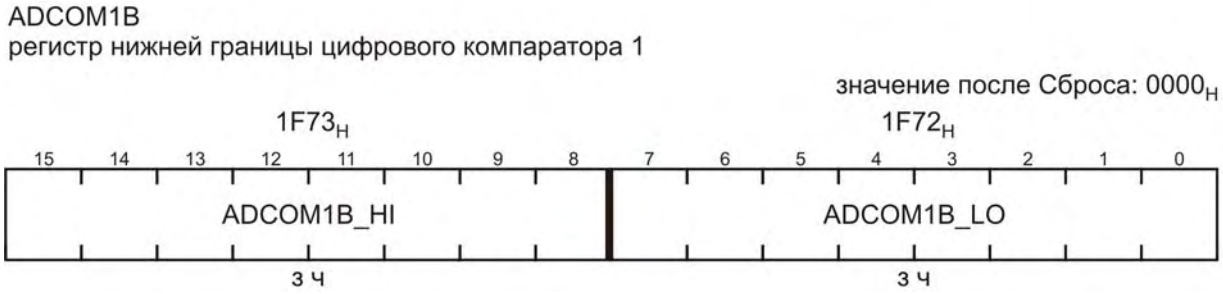


Рисунок В.2 – Формат регистра ADCOM1B

Таблица В.7 – Функциональные назначения полей регистра ADCOM1B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM1B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 1
8-15	ADCOM1B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 1

ADCOM2B Регистр нижней границы цифрового компаратора 2

1F75H/1F74H (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 2. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT2 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM2T.

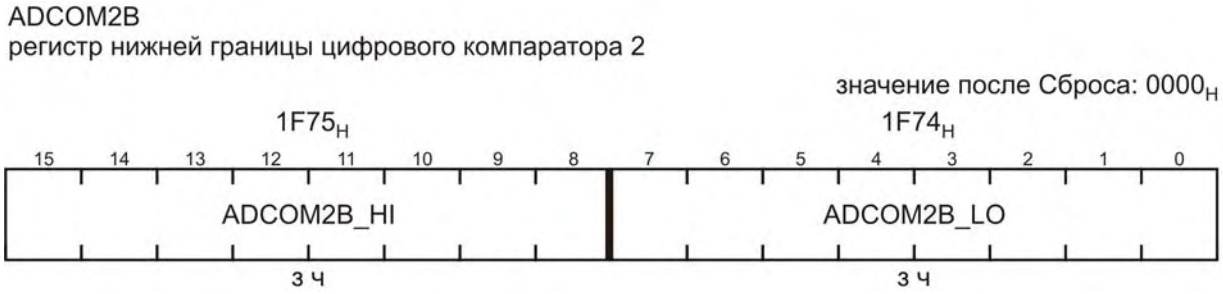


Рисунок В.3 – Формат регистра ADCOM2B

Таблица В.8 – Функциональные назначения полей регистра ADCOM2B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM2B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 2
8-15	ADCOM2B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 2

ADCOM3B Регистр нижней границы цифрового компаратора 3

1F77H/1F76H (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 3. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT3 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM3T.

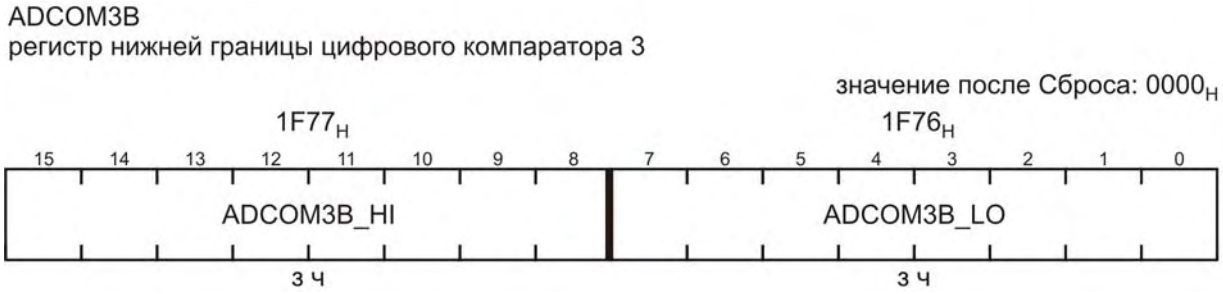


Рисунок В.4 – Формат регистра ADCOM3B

Таблица В.9 – Функциональные назначения полей регистра ADCOM3B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM3B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 3
8-15	ADCOM3B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 3

ADCOM4B Регистр нижней границы цифрового компаратора 4

1F79H/1F78H (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 4. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT4 за пределы заданных границ задается в регистре ADINTCH. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM4T.

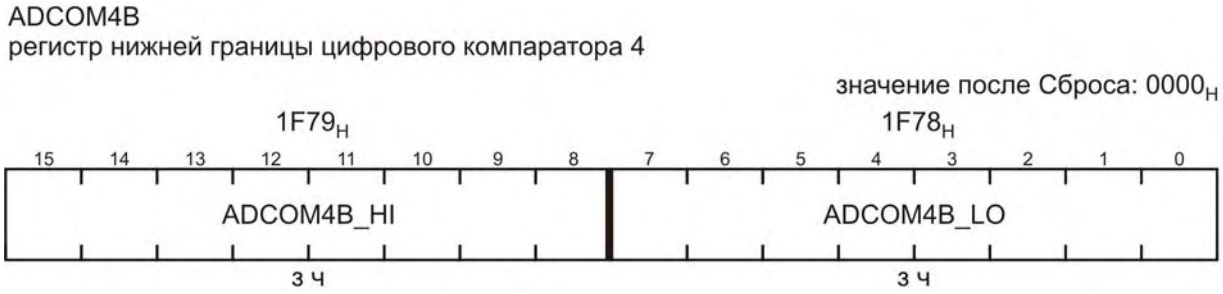


Рисунок В.5 – Формат регистра ADCOM4B

Таблица В.10 – Функциональные назначения полей регистра ADCOM4B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM4B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 4
8-15	ADCOM4B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 4

ADCOM5B Регистр нижней границы цифрового компаратора 5

1F7BH/1F7AH (чтение/запись)

Регистр содержит нижнюю границу срабатывания цифрового компаратора 5. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT5 за пределы заданных границ задается в регистре ADINTCH. Возможно задание как отдельно нижней границы, так и диапазона совместно с регистром ADCOM5T.

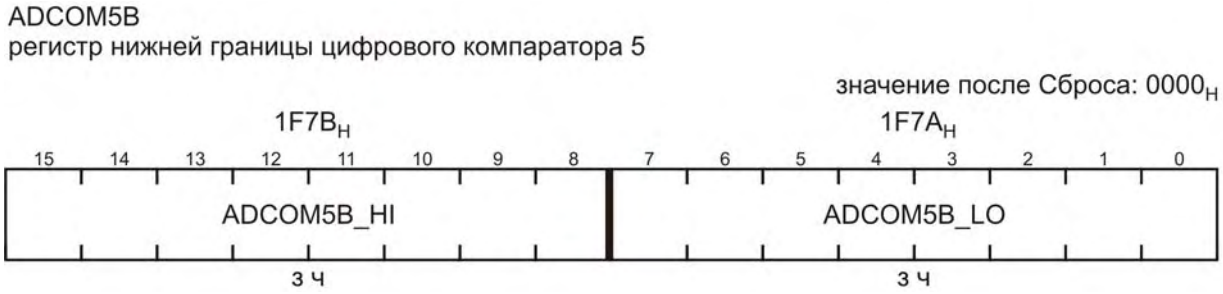


Рисунок В.6 – Формат регистра ADCOM5B

Таблица В.11 – Функциональные назначения полей регистра ADCOM5B

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM5B_LO	Младший байт нижней границы	Содержит младший байт нижней границы срабатывания цифрового компаратора 5
8-15	ADCOM5B_HI	Старший байт нижней границы	Содержит старший байт нижней границы срабатывания цифрового компаратора 5

ADCOM0T Регистр верхней границы цифрового компаратора 0

1F81H/1F80H (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 0. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT0 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM0B.

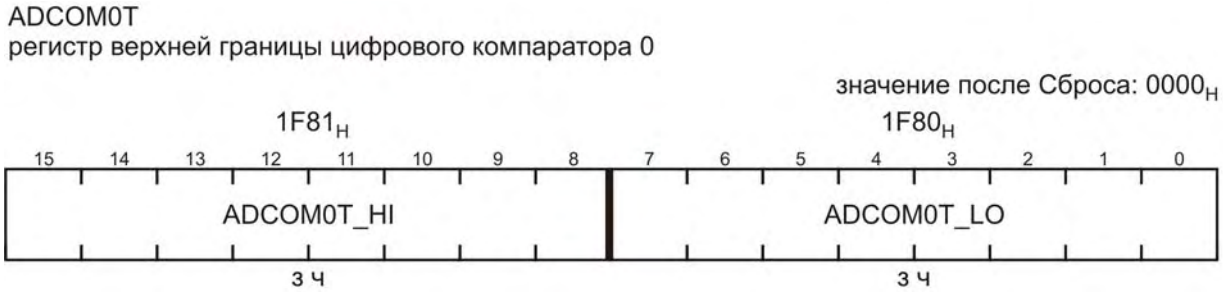


Рисунок В.7 – Формат регистра ADCOM0T

Таблица В.12 – Функциональные назначения полей регистра ADCOM0T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM0T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 0
8-15	ADCOM0T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 0

ADCOM1T Регистр верхней границы цифрового компаратора 1

1F83H/1F82H (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 1. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT1 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM1B.

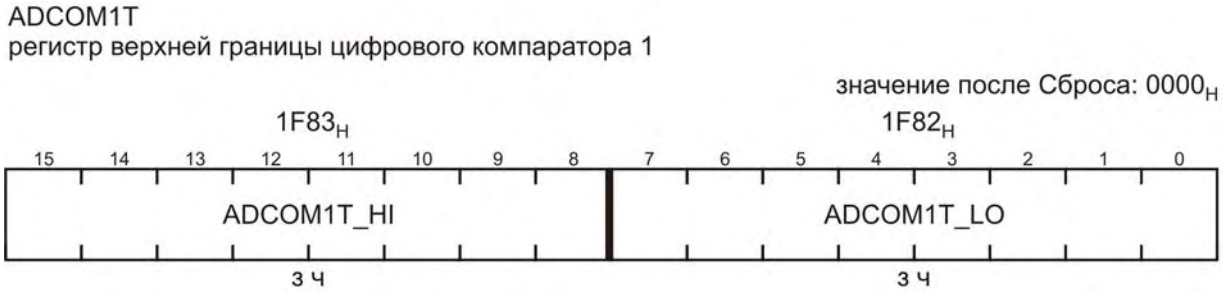


Рисунок В.8 – Формат регистра ADCOM1T

Таблица В.13 – Функциональные назначения полей регистра ADCOM1T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM1T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 1
8-15	ADCOM1T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 1

ADCOM2T Регистр верхней границы цифрового компаратора 2

1F85H/1F84H (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 2. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT2 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM2B.

Рисунок В.9 – Формат регистра ADCOM2T

Таблица В.14 – Функциональные назначения полей регистра ADCOM2T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM2T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 2
8-15	ADCOM2T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 2

ADCOM3T Регистр верхней границы цифрового компаратора 3

1F87H/1F86H (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 3. Условия срабатывания компаратора устанавливаются в регистре ADCOMCL. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT3 за пределы заданных границ задается в регистре ADINTCL. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM3B.

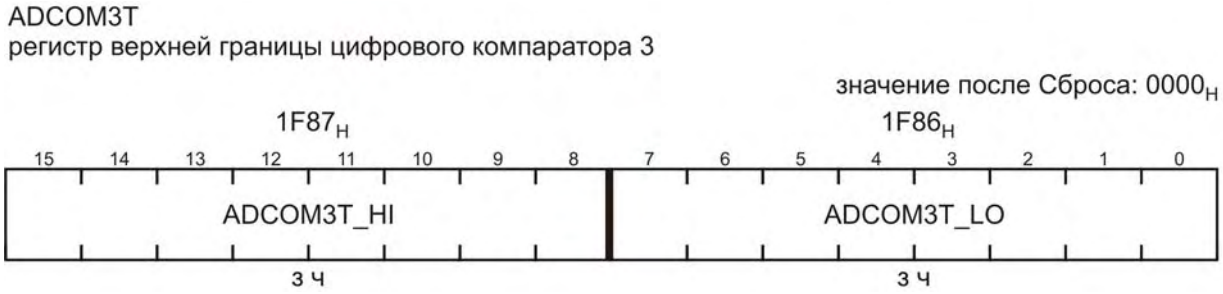


Рисунок В.10 – Формат регистра ADCOM3T

Таблица В.15 – Функциональные назначения полей регистра ADCOM3T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM3T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 3
8-15	ADCOM3T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 3

ADCOM4T Регистр верхней границы цифрового компаратора 4

1F89H/1F88H (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 4. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT4 за пределы заданных границ задается в регистре ADINTCH. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM4B.

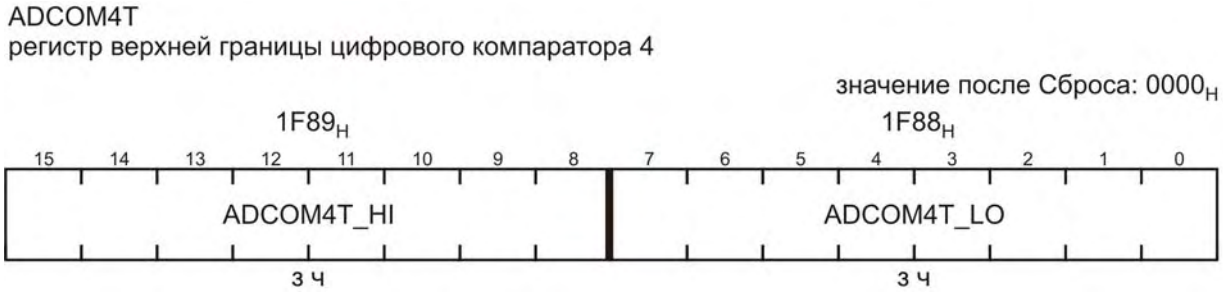


Рисунок В.11 – Формат регистра ADCOM4T

Таблица В.16 – Функциональные назначения полей регистра ADCOM4T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM4T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 4
8-15	ADCOM4T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 4

ADCOM5T Регистр верхней границы цифрового компаратора 5

1F8BH/1F8AH (чтение/запись)

Регистр содержит верхнюю границу срабатывания цифрового компаратора 5. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT5 за пределы заданных границ задается в регистре ADINTCH. Возможно задание как отдельно верхней границы, так и диапазона совместно с регистром ADCOM5B.

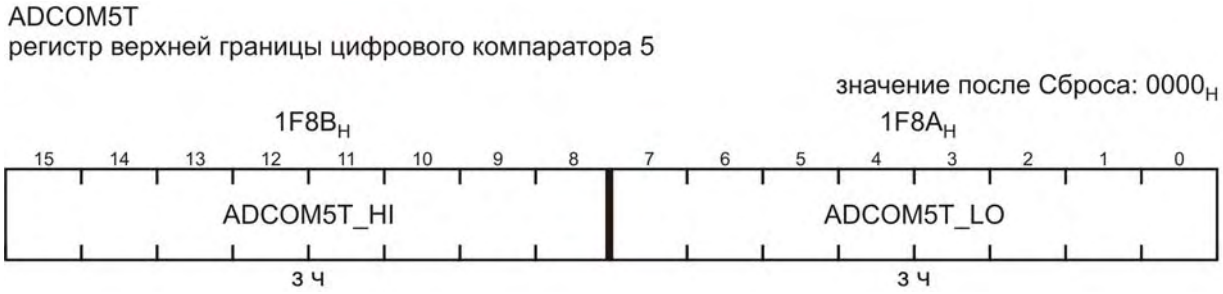


Рисунок В.12 – Формат регистра ADCOM5T

Таблица В.17 – Функциональные назначения полей регистра ADCOM5T

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM5T_LO	Младший байт верхней границы	Содержит младший байт верхней границы срабатывания цифрового компаратора 5
8-15	ADCOM5T_HI	Старший байт верхней границы	Содержит старший байт верхней границы срабатывания цифрового компаратора 5

ADCOM6 Регистр границы цифрового компаратора 6

1F8DH/1F8CH (чтение/запись)

Регистр содержит границу срабатывания цифрового компаратора 6. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT6 за пределы заданной границы задается в регистре ADINTCH.

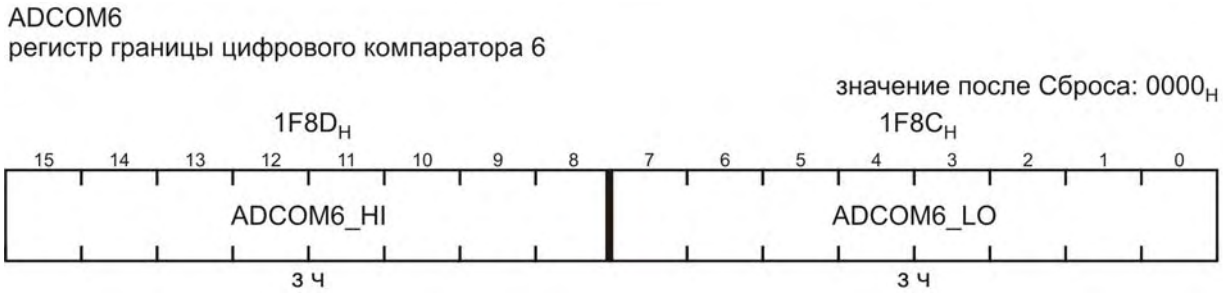


Рисунок В.13 – Формат регистра ADCOM6

Таблица В.18 – Функциональные назначения полей регистра ADCOM6

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM6_LO	Младший байт границы	Содержит младший байт границы срабатывания цифрового компаратора 6
8-15	ADCOM6_HI	Старший байт границы	Содержит старший байт границы срабатывания цифрового компаратора 6

ADCOM7 Регистр границы цифрового компаратора 7

1F8FH/1F8EH (чтение/запись)

Регистр содержит границу срабатывания цифрового компаратора 7. Условия срабатывания компаратора устанавливаются в регистре ADCOMCH. Возможность генерации прерывания A/D Conversion при выходе результата преобразования АЦП ADRESULT7 за пределы заданной границы задается в регистре ADINTCH.

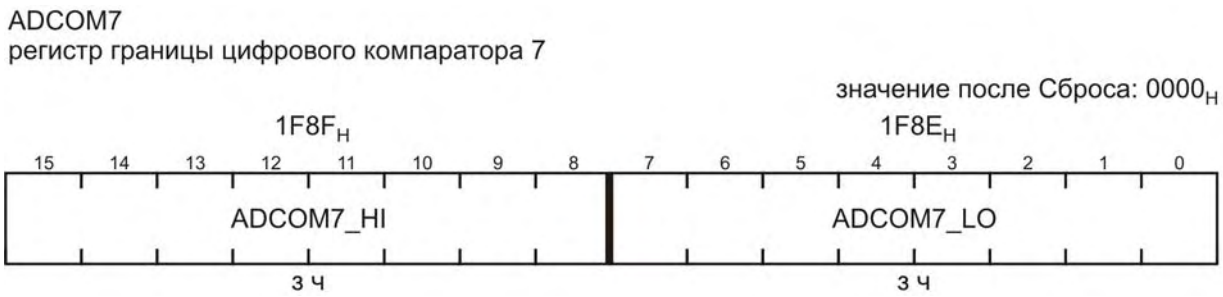


Рисунок В.14 – Формат регистра ADCOM7

Таблица В.19 – Функциональные назначения полей регистра ADCOM7

Номер бита	Мнемоника	Имя бита	Описание
0-7	ADCOM7_LO	Младший байт границы	Содержит младший байт границы срабатывания цифрового компаратора 7
8-15	ADCOM7_HI	Старший байт границы	Содержит старший байт границы срабатывания цифрового компаратора 7

ADCOMCL Регистр управления цифровыми компараторами 1

1F7EH (чтение/запись)

Регистр управляет режимами работы компараторов 0–3. Возможны четыре режима работы компараторов: срабатывание по нижней границе (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxB или ADCOM6,7); срабатывание по верхней границе (если результат преобразования больше или равен значению, заданному в регистре ADCOMxT или ADCOM6,7); срабатывание по попаданию в диапазон (только для каналов 0 – 5) (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxT, и больше или равен значению, заданному в регистре ADCOMxB); срабатывание по выходу из диапазона (только для каналов 0 – 5) (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxB, или больше или равен значению, заданному в регистре ADCOMxT). Флаги срабатывания компараторов 0–3 и биты управления генерацией прерываний находятся в регистре ADINTCL.

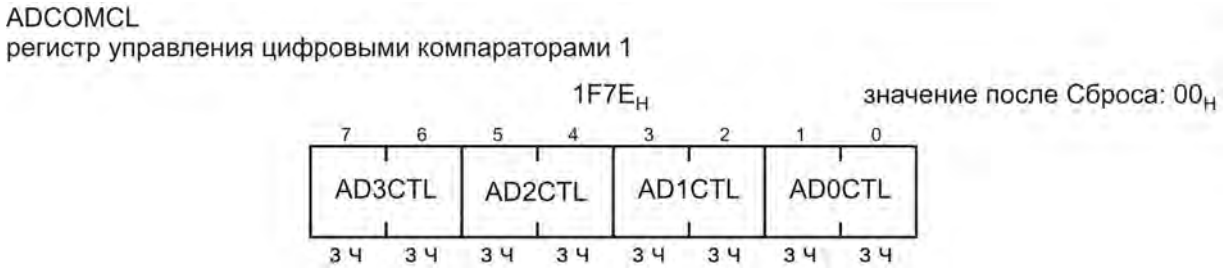


Рисунок В.15 – Формат регистра ADCOMCL

Таблица В.20 – Функциональные назначения полей регистра ADCOMCL

Номер бита	Мнемо-ника	Имя бита	Описание
0-1	AD0CTL	Биты управления цифровым компаратором 0	Определяет режим работы компаратора 0: 00 – срабатывание по выходу за нижнюю границу, 01 – срабатывание по выходу за верхнюю границу, 10 – срабатывание по выходу из диапазона, 11 – срабатывание по по попаданию в диапазон
2-3	AD1CTL	Биты управления цифровым компаратором 1	Определяет режим работы компаратора 1: 00 – срабатывание по выходу за нижнюю границу, 01 – срабатывание по выходу за верхнюю границу, 10 – срабатывание по выходу из диапазона, 11 – срабатывание по по попаданию в диапазон
4-5	AD2CTL	Биты управления цифровым компаратором 2	Определяет режим работы компаратора 2: 00 – срабатывание по выходу за нижнюю границу, 01 – срабатывание по выходу за верхнюю границу, 10 – срабатывание по выходу из диапазона, 11 – срабатывание по по попаданию в диапазон
6-7	AD3CTL	Биты управления цифровым компаратором 3	Определяет режим работы компаратора 3: 00 – срабатывание по выходу за нижнюю границу, 01 – срабатывание по выходу за верхнюю границу, 10 – срабатывание по выходу из диапазона, 11 – срабатывание по по попаданию в диапазон

ADCOMCH Регистр управления цифровыми компараторами 2

1F7FH (чтение/запись)

Регистр управляет режимами работы компараторов 4 – 7. Возможны четыре режима работы компараторов: срабатывание по нижней границе (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxB или ADCOM6,7); срабатывание по верхней границе (если результат преобразования больше или равен значению, заданному в регистре ADCOMxT или ADCOM6,7); срабатывание по попаданию в диапазон (только для каналов 0 – 5) (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxT, и больше или равен значению, заданному в регистре ADCOMxB); срабатывание по выходу из диапазона (только для каналов 0 – 5) (если результат преобразования меньше или равен значению, заданному в регистре ADCOMxB, или больше или равен значению, заданному в регистре ADCOMxT). Флаги срабатывания компараторов 4 – 7 и биты управления генерацией прерываний находятся в регистре ADINTCH.

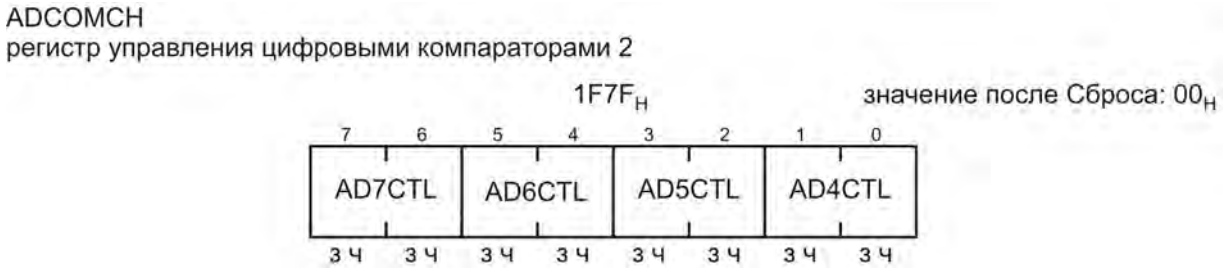


Рисунок В.16 – Формат регистра ADCOMCH

Таблица В.21 – Функциональные назначения полей регистра ADCOMCH

Номер бита	Мнемо-ника	Имя бита	Описание
0-1	AD4CTL	Биты управления цифровым компаратором 4	Определяет режим работы компаратора 4: «00» – срабатывание по выходу за нижнюю границу, «01» – срабатывание по выходу за верхнюю границу, «10» – срабатывание по выходу из диапазона, «11» – срабатывание по по попаданию в диапазон
2-3	AD5CTL	Биты управления цифровым компаратором 5	Определяет режим работы компаратора 5: 00 – срабатывание по выходу за нижнюю границу, 01 – срабатывание по выходу за верхнюю границу, 10 – срабатывание по выходу из диапазона, 11 – срабатывание по по попаданию в диапазон
4-5	AD6CTL	Биты управления цифровым компаратором 6	Определяет режим работы компаратора 6: 00 – срабатывание, если результат преобразования меньше граничного значения, 01 – срабатывание, если результат преобразования больше граничного значения
6-7	AD7CTL	Биты управления цифровым компаратором 7	Определяет режим работы компаратора 7: 00 – срабатывание, если результат преобразования меньше граничного значения, 01 – срабатывание, если результат преобразования больше граничного значения

ADCRA Регистр выборочного запуска преобразования АЦП

1FC0H (чтение/запись)

Регистр CRA управляет делителем тактовой частоты АЦП, устанавливает частоту выборки и управляет генерацией прерывания по окончанию преобразования. Запрещение генерации по окончанию преобразования применяется при работе цифровых компараторов. Задание скорости выборки с помощью битового поля DR позволяет уменьшить время преобразования. При уменьшении частоты выборки точность преобразования ухудшается. Бит DIV2 используется при дополнительном управлении частотой АЦП. Рекомендуется выбирать скорость работы АЦП битами DR и MCD, а бит DIV2 использовать в случае, если требуемую частоту не удается получить с помощью других способов.

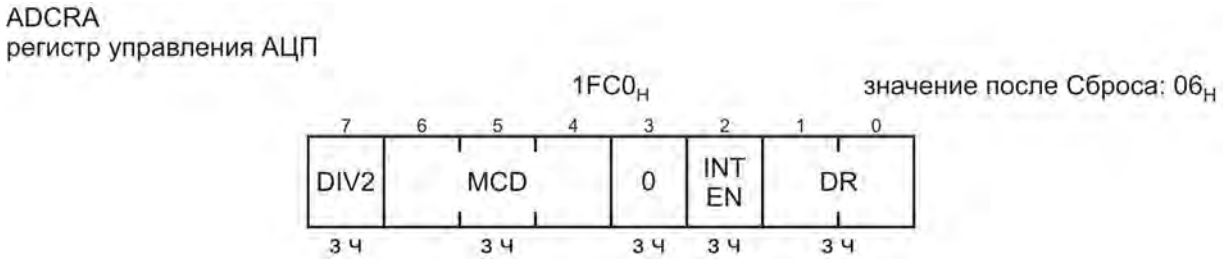


Рисунок В.17 – Формат регистра ADCRA

Таблица В.22 – Функциональные назначения полей регистра ADCRA

Номер бита	Мнемо-ника	Имя бита	Описание
0 – 1	DR	Частота дискретизации	Биты определяют частоту выборки. Биты 1 0 частота 0 0 DMCLK/2048 0 1 DMCLK/1024 1 0 DMCLK/512 1 1 DMCLK/256
2	INTEN	Разрешение прерывания по окончанию преобразования	Бит определяет, будет ли окончание преобразования формировать прерывание A/D Conversion: «0» – окончание преобразования не генерирует прерывание, «1» – окончание преобразования генерирует прерывание
3	-	-	Зарезервирован. Всегда записывать «0»
4 – 6	MCD	Делитель опорной тактовой частоты	Биты определяют значение коэффициента деления опорной тактовой частоты АЦП: Биты 6 5 4 частота 0 0 0 DMCLK/1 0 0 1 DMCLK/2 0 1 0 DMCLK/3 0 1 1 DMCLK/4 1 0 0 DMCLK/5 1 0 1 DMCLK/1 1 1 0 DMCLK/1 1 1 1 DMCLK/1
7	DIV2	Дополнительный делитель тактовой частоты	Бит управляет дополнительным делителем на 2 тактовой частоты: 0 – дополнительный делитель выключен, 1 – дополнительный делитель включен

ADCRB Регистр управления АЦП

1FC1H (чтение/запись)

Регистр управления АЦП позволяет осуществить одновременный запуск преобразования всех каналов, сброс аналоговых модуляторов выбранных каналов, включает режим инверсии каналов, разрешает выдачу опорного напряжения и управляет включением дифференциального режима входов. После установки бита сброса аналоговых модуляторов произойдет сброс модуляторов каналов, указанных в регистре ADCRG. Выбор каналов в регистре ADCRG следует осуществить до записи в регистр CRC.

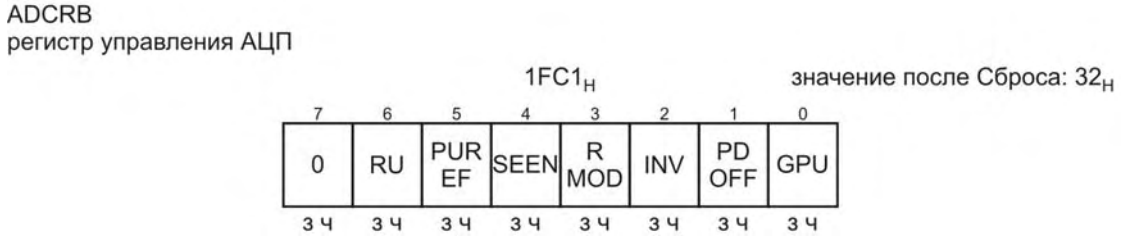


Рисунок В.18 – Формат регистра ADCRB

Таблица В.23 – Функциональные назначения полей регистра ADCRB

Номер бита	Мнемо-ника	Имя бита	Описание
0	GPU	Глобальный запуск преобразования	Бит запускает преобразование всех каналов непосредственно после установки. «1» – запуск преобразования «0» – нет глобального запуска
1	PDOFF	Бит управления выключением опорного напряжения	Бит управляет выключением опорного напряжения. «1» – опорное напряжение выключается при POWERDOWN или при запрещении тактового сигнала АЦП (CLKCL); «0» – опорное напряжение не выключается при POWERDOWN или при запрещении тактового сигнала АЦП (CLKCL)
2	INV	Инверсный режим каналов	Бит разрешения инверсии каналов. «1» – инверсия разрешена для каналов, чьи биты CHI установлены в регистре ADCRH; «0» – инверсия каналов запрещена
3	RMOD	Сброс аналоговых модуляторов	Бит разрешает сброс аналоговых модуляторов. «1» – сброс разрешен для каналов, чьи биты CHRE установлены в регистре ADCRG; «0» – сброс запрещен
4	SEEN	Включение дифференциального режима входов	Бит совместно с битами CHRE включает дифференциальный режим: «1» – режим прямого входа для каналов, чьи биты CHRE установлены в регистре ADCRG. Те каналы, чьи биты CHRE сброшены, функционируют в режиме дифференциальных входов; «0» – не использовать
5	PUREF	Включение опорного напряжения	Бит включения опорного напряжения. «1» – опорное напряжение включено; «0» – опорное напряжение выключено. Источник автоматически включается, если активирован любой из каналов.

Окончание таблицы В.23

Номер бита	Мнемоника	Имя бита	Описание
6	RU	Включение буферизированного опорного напряжения	Бит управляет подачей буферизированного опорного напряжения на вывод P0.7: «1» – опрорное напряжение подается; «0» – опорное напряжение не подается
7	-	-	Зарезервировано. Всегда записывать «0»

ADCRC Регистр управления каналами 0 и 1 АЦП

1FC2H (чтение/запись)

Регистр управления CRC осуществляет индивидуальное управление каналами 0 и 1 АЦП: позволяет программировать коэффициент усиления входов каналов 0 и 1 и независимый запуск преобразования каналов 0 и 1. Преобразование запускается непосредственно после установки бита старта преобразования.

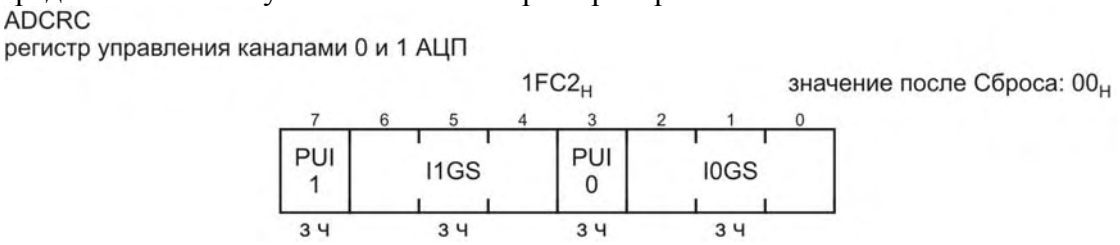


Рисунок В.19 – Формат регистра ADCRC

Таблица В.24 – Функциональные назначения полей регистра ADCRC

Номер бита	Мнемоника	Имя бита	Описание
0 – 2	I0GS	Коэффициент усиления канала 0	Биты определяют коэффициент усиления канала 0, дБ: бит 2 1 0 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
3	PUI0	Запуск преобразования канала 0	Бит запуска непрерывного преобразования канала 0: «1» – старт преобразования; «0» – нет преобразования
4 – 6	I1GS	Коэффициент усиления канала 1	Биты определяют коэффициент усиления канала 1, дБ: бит 6 5 4 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
7	PUI1	Запуск преобразования канала 1	Бит запуска непрерывного преобразования канала 1: «1» – старт преобразования; «0» – нет преобразования

ADCRD Регистр управления каналами 2 и 3 АЦП

1FC3H (чтение/запись)

Регистр управления CRD осуществляет индивидуальное управление каналами 2 и 3 АЦП: позволяет программировать коэффициент усиления входов каналов 2 и 3 и независимый запуск преобразования каналов 2 и 3. Преобразование запускается непосредственно после установки бита старта преобразования.

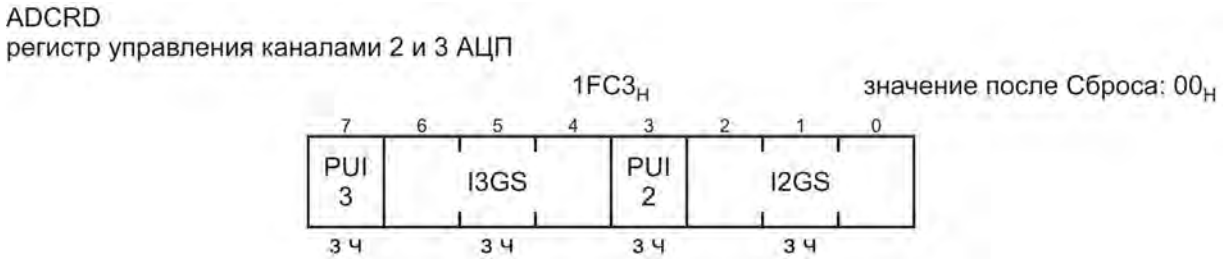


Рисунок В.20 – Формат регистра ADCRD

Таблица В.25 – Функциональные назначения полей регистра ADCRD

Номер бита	Мнемоника	Имя бита	Описание
0 – 2	I2GS	Коэффициент усиления канала 2	Биты определяют коэффициент усиления канала 2, дБ: бит 2 1 0 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
3	PUI2	Запуск преобразования канала 2	Бит запуска непрерывного преобразования канала 2: «1» – старт преобразования; «0» – нет преобразования
4 – 6	I3GS	Коэффициент усиления канала 3	Биты определяют коэффициент усиления канала 3, дБ: бит 6 5 4 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
7	PUI3	Запуск преобразования канала 3	Бит запуска непрерывного преобразования канала 3: «1» – старт преобразования; «0» – нет преобразования

ADCRE Регистр управления каналами 4 и 5 АЦП

1FC4H (чтение/запись)

Регистр управления CRE осуществляет индивидуальное управление каналами 4 и 5 АЦП: позволяет программировать коэффициент усиления входов каналов 4 и 5 и независимый запуск преобразования каналов 4 и 5. Преобразование запускается непосредственно после установки бита старта преобразования.

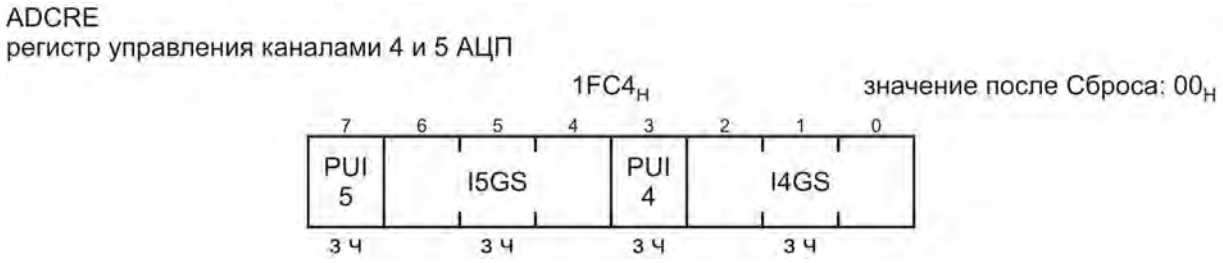


Рисунок В.21 – Формат регистра ADCRE

Таблица В.26 – Функциональные назначения полей регистра ADCRE

Номер бита	Мнемоника	Имя бита	Описание
0 – 2	I4GS	Коэффициент усиления канала 4	Биты определяют коэффициент усиления канала 4, дБ: бит 2 1 0 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
3	PUI4	Запуск преобразования канала 4	Бит запуска непрерывного преобразования канала 4: «1» – старт преобразования; «0» – нет преобразования
4 – 6	I5GS	Коэффициент усиления канала 5	Биты определяют коэффициент усиления канала 5, дБ: бит 6 5 4 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
7	PUI5	Запуск преобразования канала 5	Бит запуска непрерывного преобразования канала 5: «1» – старт преобразования; «0» – нет преобразования

ADCRF Регистр управления каналами 6 и 7 АЦП

1FC5H (чтение/запись)

Регистр управления CRF осуществляет индивидуальное управление каналами 6 и 7 АЦП: позволяет программировать коэффициент усиления входов каналов 6 и 7 и независимый запуск преобразования каналов 6 и 7. Преобразование запускается непосредственно после установки бита старта преобразования.

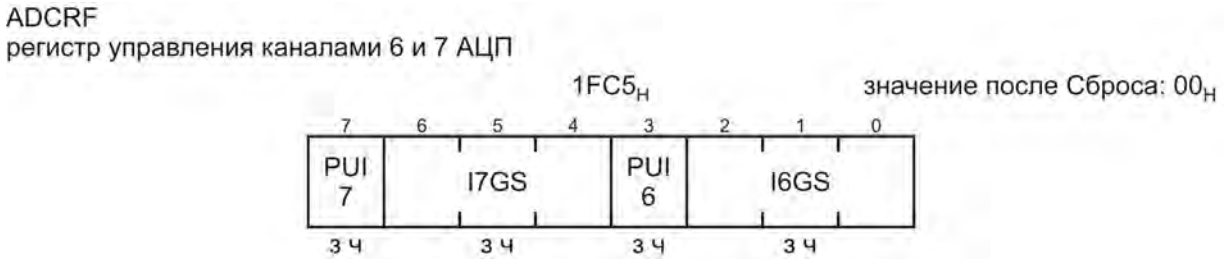


Рисунок В.22 – Формат регистра ADCRF

Таблица В.27 – Функциональные назначения полей регистра ADCRF

Номер бита	Мнемоника	Имя бита	Описание
0 – -2	I6GS	Коэффициент усиления канала 6	Биты определяют коэффициент усиления канала 6, дБ: бит 2 1 0 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
3	PUI6	Запуск преобразования канала 6	Бит запуска непрерывного преобразования канала 6: 1 = старт преобразования 0 = нет преобразования
4 – 6	I6GS	Коэффициент усиления канала 7	Биты определяют коэффициент усиления канала 7, дБ: бит 6 5 4 коэффициент 0 0 0 0 0 0 1 6 0 1 0 12 0 1 1 18 1 0 0 20 1 0 1 26 1 1 0 32 1 1 1 38
7	PUI7	Запуск преобразования канала 7	Бит запуска непрерывного преобразования канала 7: «1» – старт преобразования; «0» – нет преобразования

ADCRG Регистр управления сбросом аналоговых модуляторов и включением дифференциального режима каналов АЦП

1FC6H (чтение/запись)

Регистр управления определяет каналы, аналоговые модуляторы которых необходимо сбросить в начальное состояние или каналы, которые необходимо перевести в режим работы с дифференциальными входами. Регистр осуществляет только выбор каналов, поэтому установка соответствующих битов каналов не приводит к сбросу аналоговых модуляторов. Сброс модуляторов выбранных каналов начинается после установки бита RMOD в регистре ADCRB. Обнуление битов CHRE включает дифференциальные режимы каналов (записывать нули только в биты для каналов 0 – 3).

Рисунок В.23 – Формат регистра ADCRG

Таблица В.28 – Функциональные назначения полей регистра ADCRG

Номер бита	Мнемо-ника	Имя бита	Описание
0	CHRE0	Выбор канала 0	Определяет, будет ли выбран канала 0 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
1	CHRE1	Выбор канала 1	Определяет, будет ли выбран канала 1 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
2	CHRE2	Выбор канала 2	Определяет, будет ли выбран канала 2 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
3	CHRE3	Выбор канала 3	Определяет, будет ли выбран канала 3 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
4	CHRE4	Выбор канала 4	Определяет, будет ли выбран канала 4 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
5	CHRE5	Выбор канала 5	Определяет, будет ли выбран канала 5 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим

Окончание таблицы В.28

Номер бита	Мнемо-ника	Имя бита	Описание
6	CHRE6	Выбор канала 6	Определяет, будет ли выбран канала 6 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим
7	CHRE7	Выбор канала 7	Определяет, будет ли выбран канала 7 для сброса или выбран режим дифференциального включения: «1» – сброс разрешен/режим прямого входа; «0» – сброс запрещен/дифференциальный режим

ADCRH Регистр управления режимом инверсии каналов АЦП

1FC7H (чтение/запись)

Регистр управления определяет каналы, для которых будет осуществляться инверсия входного сигнала. Для включения режима инверсии требуется установка бита глобального разрешения инверсии INV в регистре ADCRB. Для успешного включения режима инверсии запись в регистры ADCRH и ADCRB можно осуществлять в любой последовательности, но обязательно до запуска преобразования требуемых каналов.

Рисунок В.24 – Формат регистра ADCRH

Таблица В.29 – Функциональные назначения полей регистра ADCRH

Номер бита	Мнемо-ника	Имя бита	Описание
0	CHI0	Выбор канала 0	Определяет, будет ли включен режим инверсии для канала 0: «1» – инверсия разрешена; «0» – инверсия запрещена
1	CHI1	Выбор канала 1	Определяет, будет ли включен режим инверсии для канала 1: «1» – инверсия разрешена; «0» – инверсия запрещена
2	CHI2	Выбор канала 2	Определяет, будет ли включен режим инверсии для канала 2: «1» – инверсия разрешена; «0» – инверсия запрещена
3	CHI3	Выбор канала 3	Определяет, будет ли включен режим инверсии для канала 3: «1» – инверсия разрешена; «0» – инверсия запрещена
4	CHI4	Выбор канала 4	Определяет, будет ли включен режим инверсии для канала 4: «1» – инверсия разрешена; «0» – инверсия запрещена
5	CHI5	Выбор канала 5	Определяет, будет ли включен режим инверсии для канала 5: «1» – инверсия разрешена; «0» – инверсия запрещена
6	CHI6	Выбор канала 6	Определяет, будет ли включен режим инверсии для канала 6: «1» – инверсия разрешена; «0» – инверсия запрещена
7	CHI7	Выбор канала 7	Определяет, будет ли включен режим инверсии для канала 7: «1» – инверсия разрешена; «0» – инверсия запрещена

ADINTCL Регистр управления прерываниями цифровых компараторов 1

1F7CH (чтение/запись)

Регистр управляет генерацией прерываний при срабатывании цифровых компараторов для каналов 0 – 3, а также содержит флаги срабатывания компараторов. Если флаг срабатывания компаратора установлен, и прерывание от этого компаратора разрешено, то генерируется запрос на прерывание A/D Conversion. Биты флагов устанавливаются аппаратно или программно, сбрасываются только программно.

Рисунок В.25 – Формат регистра ADINTCL

Таблица В.30 – Функциональные назначения полей регистра ADINTCL

Номер бита	Мнемо-ника	Имя бита	Описание
0	AD0INTEN	Разрешение прерывания от канала 0	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 0: «1» – запрос будет генерироваться; «0» – запрещена генерация
1	AD0FLAG	Флаг срабатывания компаратора от канала 0	Показывает, произошло ли срабатывание компаратора 0: «1» – компаратор сработал; «0» – компаратор не сработал
2	AD1INTEN	Разрешение прерывания от канала 1	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 1: «1» – запрос будет генерироваться; «0» – запрещена генерация
3	AD1FLAG	Флаг срабатывания компаратора от канала 1	Показывает, произошло ли срабатывание компаратора 1: «1» – компаратор сработал; «0» – компаратор не сработал
4	AD2INTEN	Разрешение прерывания от канала 2	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 2: «1» – запрос будет генерироваться; «0» – запрещена генерация
5	AD2FLAG	Флаг срабатывания компаратора от канала 2	Показывает, произошло ли срабатывание компаратора 2: «1» – компаратор сработал; «0» – компаратор не сработал
6	AD3INTEN	Разрешение прерывания от канала 3	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 3: «1» – запрос будет генерироваться; «0» – запрещена генерация

Окончание таблицы В.30

Номер бита	Мнемо-ника	Имя бита	Описание
7	AD3FLAG	Флаг срабатывания компаратора от канала 3	Показывает, произошло ли срабатывание компаратора 3: «1» – компаратор сработал; «0» – компаратор не сработал

ADINTCH Регистр управления прерываниями цифровых компараторов 1

1F7DH (чтение/запись)

Регистр управляет генерацией прерываний при срабатывании цифровых компараторов для каналов 4 – 7, а также содержит флаги срабатывания компараторов. Если флаг срабатывания компаратора установлен, и прерывание от этого компаратора разрешено, то генерируется запрос на прерывание A/D Conversion. Биты флагов устанавливаются аппаратно или программно, сбрасываются только программно.

Рисунок В.26 – Формат регистра ADINTCH

Таблица В.31 – Функциональные назначения полей регистра ADINTCH

Номер бита	Мнемо-ника	Имя бита	Описание
0	AD4INTEN	Разрешение прерывания от канала 0	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 4: «1» – запрос будет генерироваться; «0» – запрещена генерация
1	AD4FLAG	Флаг срабатывания компаратора от канала 0	Показывает, произошло ли срабатывание компаратора 4: «1» – компаратор сработал; «0» – компаратор не сработал
2	AD5INTEN	Разрешение прерывания от канала 1	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 5: «1» – запрос будет генерироваться; «0» – запрещена генерация
3	AD5FLAG	Флаг срабатывания компаратора от канала 1	Показывает, произошло ли срабатывание компаратора 5: «1» – компаратор сработал; «0» – компаратор не сработал
4	AD6INTEN	Разрешение прерывания от канала 2	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 6: «1» – запрос будет генерироваться; «0» – запрещена генерация
5	AD6FLAG	Флаг срабатывания компаратора от канала 2	Показывает, произошло ли срабатывание компаратора 6: «1» – компаратор сработал; «0» – компаратор не сработал
6	AD7INTEN	Разрешение прерывания от канала 3	Определяет, будет ли генерироваться запрос на прерывание при срабатывании компаратора 7: «1» – запрос будет генерироваться; «0» – запрещена генерация
7	AD7FLAG	Флаг срабатывания компаратора от канала 3	Показывает, произошло ли срабатывание компаратора 7: «1» – компаратор сработал; «0» – компаратор не сработал

AD_RESULT0 Регистр результата канала 0 АЦП

1FD1H/1FD0H (чтение/запись)

Регистр AD_RESULT0 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 0, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

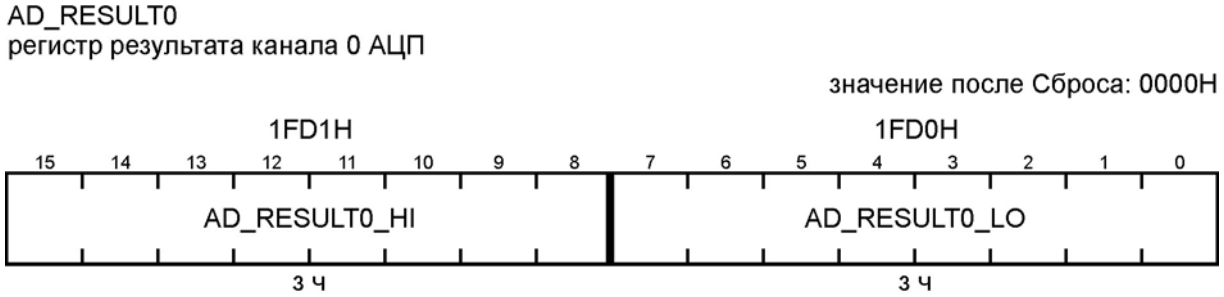


Рисунок В.27 – Формат регистра AD_RESULT0

Таблица В.32 – Функциональные назначения полей регистра AD_RESULT0

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT0_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 0
8 – 15	AD_RESULT0_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 0

AD_RESULT1 Регистр результата канала 1 АЦП

1FD3H/1FD2H (чтение/запись)

Регистр AD_RESULT1 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 1, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

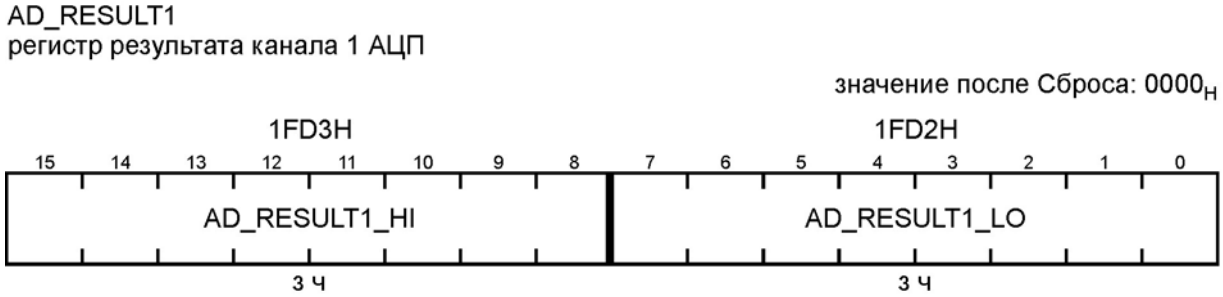


Рисунок В.28 – Формат регистра AD_RESULT1

Таблица В.33 – Функциональные назначения полей регистра AD_RESULT1

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT1_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 1
8 – 15	AD_RESULT1_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 1

AD_RESULT2 Регистр результата канала 2 АЦП

1FD5H/1FD4H (чтение/запись)

Регистр AD_RESULT2 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 2, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

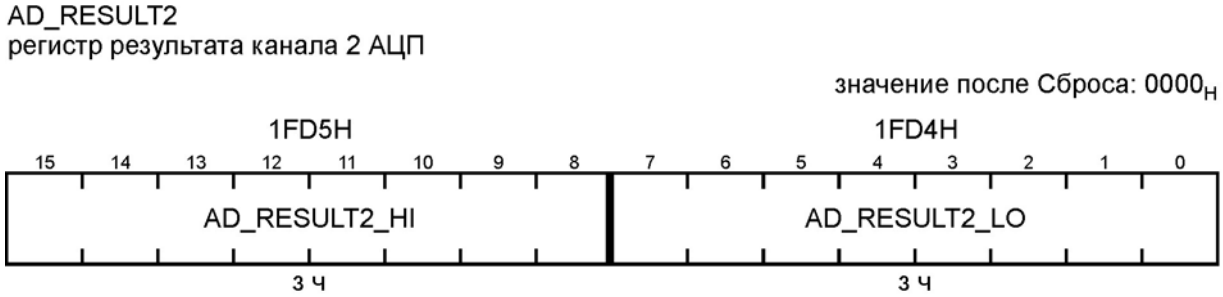


Рисунок В.29– Формат регистра AD_RESULT2

Таблица В.34 – Функциональные назначения полей регистра AD_RESULT2

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT2_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 2
8 – 15	AD_RESULT2_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 2

AD_RESULT3 Регистр результата канала 3 АЦП

1FD7H/1FD6H (чтение/запись)

Регистр AD_RESULT3 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 3, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

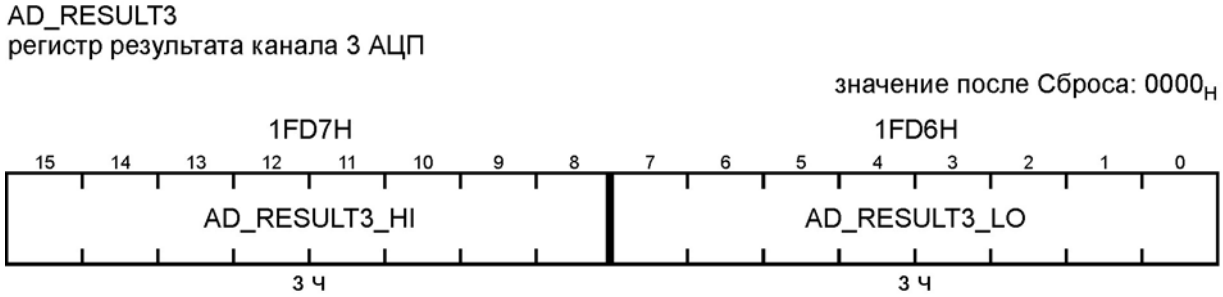


Рисунок В.30 – Формат регистра AD_RESULT3

Таблица В.35 – Функциональные назначения полей регистра AD_RESULT3

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT3_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 3
8 – 15	AD_RESULT3_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 3

AD_RESULT4 Регистр результата канала 4 АЦП

1FD9H/1FD8H (чтение/запись)

Регистр AD_RESULT4 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 4, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

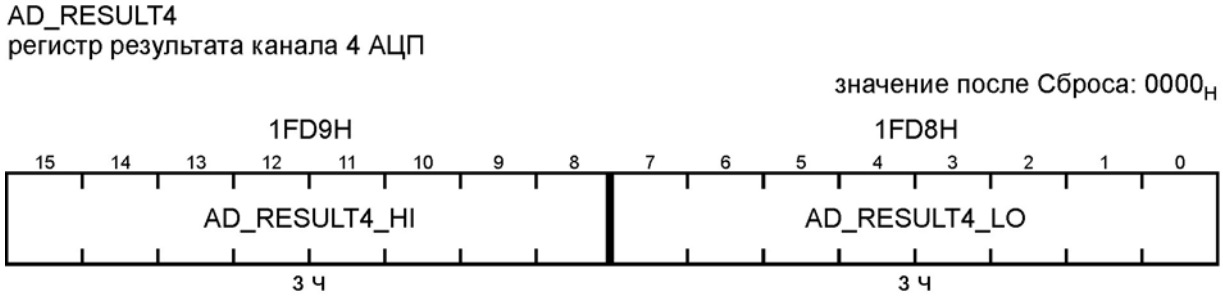


Рисунок В.31 – Формат регистра AD_RESULT4

Таблица В.36 – Функциональные назначения полей регистра AD_RESULT3

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT4_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 4
8 – 15	AD_RESULT4_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 4

AD_RESULT5 Регистр результата канала 5 АЦП

1FDBH/1FDAH (чтение/запись)

Регистр AD_RESULT5 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 5, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

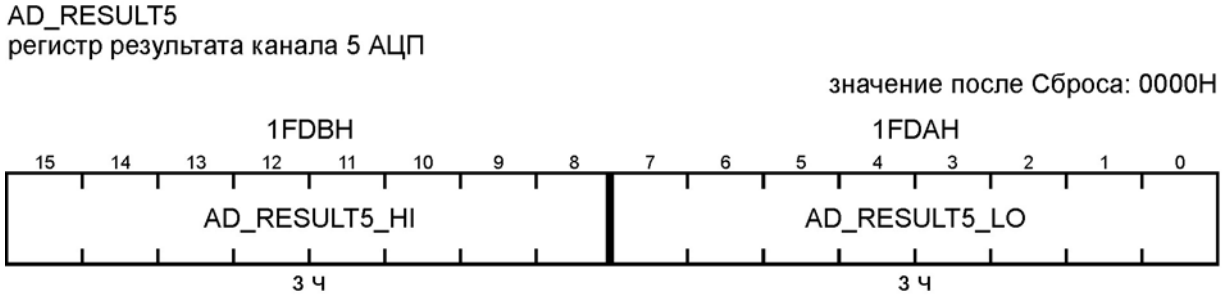


Рисунок В.32 – Формат регистра AD_RESULT5

Таблица В.37 – Функциональные назначения полей регистра AD_RESULT5

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT5_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 5
8 – 15	AD_RESULT5_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 5

AD_RESULT6 Регистр результата канала 6 АЦП

1FDDH/1FDCH (чтение/запись)

Регистр AD_RESULT6 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 6, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

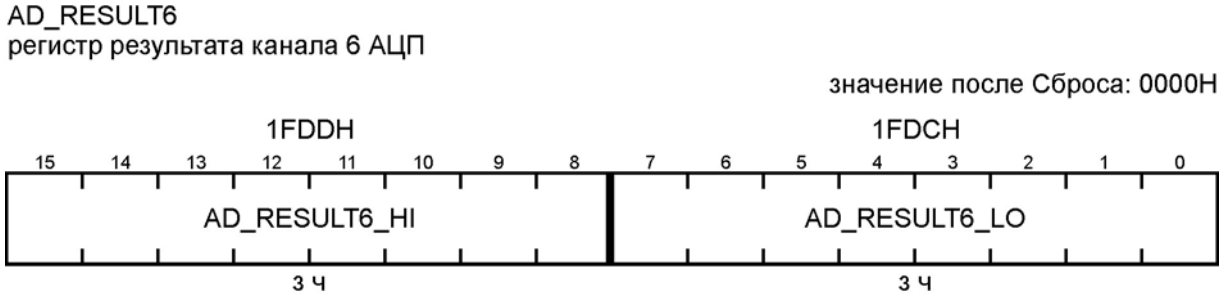


Рисунок В.33 – Формат регистра AD_RESULT6

Таблица В.38 – Функциональные назначения полей регистра AD_RESULT6

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT6_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 6
8 – 15	AD_RESULT6_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 6

AD_RESULT7 Регистр результата канала 7 АЦП

1FDFH/1FDEH (чтение/запись)

Регистр AD_RESULT7 состоит из двух байтов. Старший байт содержит восемь старших значащих разрядов результата преобразования. Младший байт содержит восемь младших значащих разрядов результата преобразования. Во время выполнения текущего преобразования регистр содержит данные предыдущего преобразования. Результат преобразования записывается в регистр после окончания преобразования канала 7, затирая предыдущие данные, следовательно, чтобы избежать потери данных, эти два байта необходимо считать перед окончанием текущего преобразования.

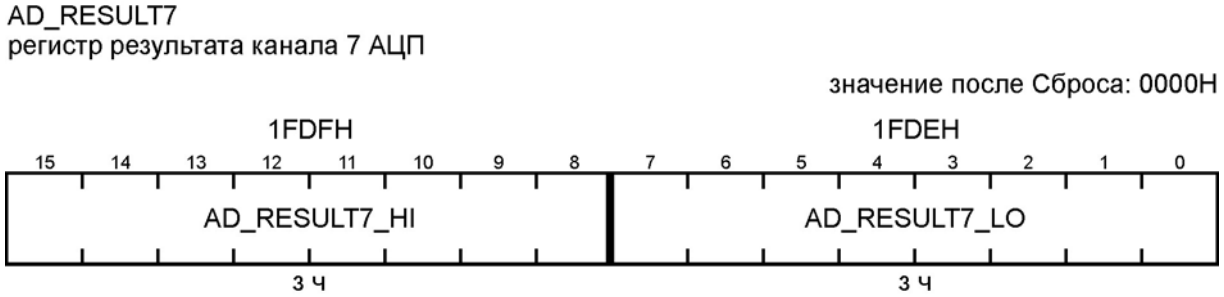


Рисунок В.34 – Формат регистра AD_RESULT7

Таблица В.39 – Функциональные назначения полей регистра AD_RESULT7

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	AD_RESULT7_LO	Результат преобразования	Восемь младших значащих битов результата преобразования канала 7
8 – 15	AD_RESULT7_HI	Результат преобразования	Восемь старших значащих битов результата преобразования канала 7

BAUD_RATE0 Регистр скорости передачи UART0

0EH

Горизонтальное окно 0 (запись)

Регистр скорости передачи выбирает скорость передачи (в бодах) последовательного порта UART0 и источник синхронизации. Он должен быть записан как два байта, сначала младший байт. Старший значащий бит выбирает источник синхронизации (clock source). Младшие 15 битов BAUD_VALUE – незнаковое целое, которое определяет скорость передачи в бодах. BAUD_ VALUE имеет максимальное значение 32 767 и может быть равен нулю только при использовании XTAL1 в асинхронных режимах 1, 2 и 3.

Рисунок В.35 – Формат регистра BAUD_RATE0

Таблица В.40 – Функциональные назначения полей регистра BAUD_RATE0

Номер бита	Мнемоника	Имя бита	Описание
0-14	BAUD_VALUE	Скорость передачи в бодах	Эти биты определяют скорость передачи в бодах. Первым загружается младший байт
15	CLOCK_SRC	Источник синхронизации последовательного порта	Этот бит определяет, от какого источника (внутреннего или внешнего) синхронизируется последовательный порт: «1» – XTAL1 (внутренний); «0» – T2CLK (внешний)

Для определения скорости передачи (в бодах) используются следующие формулы.

Синхронный режим 0:

BAUD_VALUE = (f_{CI} / (Baud Rate × 8)) – 1 или T2CLK / Baud Rate

Асинхронные режимы 1, 2 и 3:

BAUD_VALUE = (f_{CI} / (Baud Rate × 16)) – 1 или T2CLK/ (Baud Rate × 8),

где f_{CI} – частота XTAL1, МГц.

Значения скорости передачи в бодах при использовании XTAL1 на частоте 16 МГц показаны в таблице В.41.

Таблица В.41 – Скорости передачи на частоте 16 МГц

Скорость передачи (бод)	Режим 0	Режимы 1, 2, 3
9 600	8340H	8067H
4 800	8682H	80CFH
2 400	8D04H	81A0H
1 200	9A0AH	8340H
300	E82BH	8D04H

BAUD_RATE1 Регистр скорости передачи UART1

1FA6H

Регистр скорости передачи выбирает скорость передачи (в бодах) последовательного порта UART1 и источник синхронизации. Он должен быть записан как два байта, сначала младший байт. Старший значащий бит выбирает источник синхронизации (clock source). Младшие 15 битов BAUD_VALUE – незнаковое целое, которое определяет скорость передачи в бодах. BAUD_ VALUE имеет максимальное значение 32 767 и может быть равен нулю только при использовании XTAL1 в асинхронных режимах 1, 2 и 3.

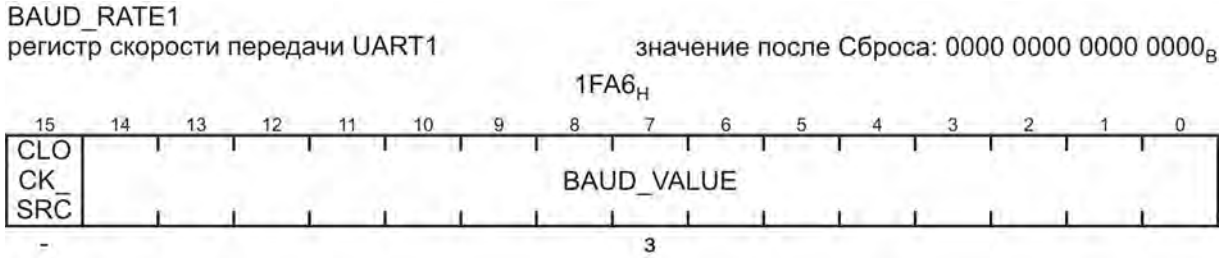


Рисунок В.36 – Формат регистра BAUD_RATE1

Таблица В.42 – Функциональные назначения полей регистра BAUD_RATE1

Номер бита	Мнемоника	Имя бита	Описание
0-14	BAUD_VALUE	Скорость передачи в бодах	Эти биты определяют скорость передачи в бодах. Первым загружается младший байт
15	CLOCK_SRC	Источник синхронизации последовательного порта	Этот бит определяет, от какого источника (внутреннего или внешнего) синхронизируется последовательный порт: «1» – XTAL1 (внутренний); «0» – T2CLK (внешний)

Для определения скорости передачи (в бодах) используются следующие формулы.

Синхронный режим 0:

BAUD_VALUE = (f_{CI} / (Baud Rate × 8)) – 1 или T2CLK / Baud Rate

Асинхронные режимы 1, 2 и 3:

BAUD_VALUE = (f_{CI} / (Baud Rate × 16)) – 1 или T2CLK/ (Baud Rate × 8),

где f_{CI} – частота XTAL1, МГц.

Значения скорости передачи в бодах при использовании XTAL1 на частоте 16 МГц показаны в таблице В.43.

Таблица В.43 – Скорости передачи на частоте 16 МГц

Скорость передачи (бод)	Режим 0	Режимы 1, 2, 3
9 600	8340H	8067H
4 800	8682H	80CFH
2 400	8D04H	81A0H
1 200	9A0AH	8340H
300	E82BH	8D04H

CCR Регистр конфигурации кристалла

Регистр конфигурации кристалла CCR управляет режимами пониженного энергопотребления, разрядностью шин, управляющими сигналами шин, режимом READY и защитой внутренней памяти.

CCR загружается из байта конфигурации кристалла (CCB), расположенного в ячейке 2018H во внутренней или внешней памяти, в зависимости от входа EA#. (Низкий уровень EA# – внешняя память; высокий уровень EA# – внутренняя память.) CCB – первый байт, выбираемый из памяти после сброса устройства. CCR загружается только однажды: при выполнении последовательности сброса; однажды загруженный CCR не может быть изменён до следующего сброса устройства.

Если контакт READY имеет низкий уровень во время выборки CCR, контроллер шин автоматически выставляет максимум три состояния ожидания в шинном цикле CCR. Это позволяет производить выборку CCR из медленной памяти. CCR.4 и CCR.5 управляют числом состояний ожидания в шинном цикле.

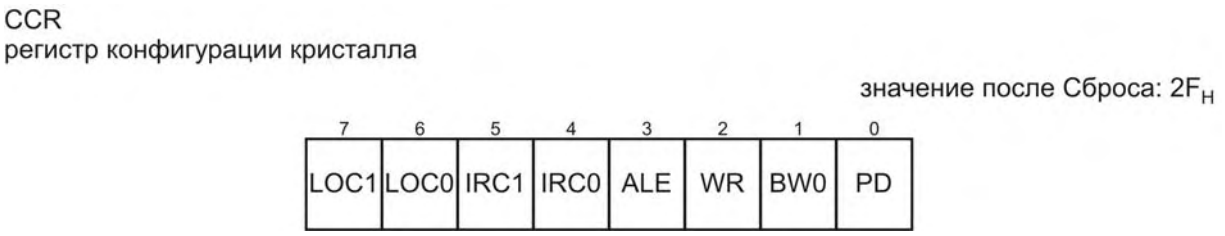


Рисунок В.37 – Формат регистра CCR

Таблица В.44 – Функциональные назначения полей регистра CCR

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	PD	Разрешение режима POWERDOWN	Определяет, ведёт ли команда IDLPD#2 к переходу устройства в режим POWERDOWN. «1» – режим возможен; «0» – режим невозможен
1	BW0	Управление разрядностью шины	Выбор: изменяемая или 8-битная разрядность шины. «1» – изменяемая разрядность шины; управляется выводом BW: BW = 1, – 16-битная шина; BW = 0, – 8-битная шина. «0» – 8-битная разрядность шины; вывод BW игнорируется
2	WR	Выбор режима Write Strobe	Выбор генерируемых стробирующих сигналов записи для 16-битных циклов: «1» – WR# и BHE# генерируются в режимах Standard Bus и Address Valid Strobe. «0» – WRL# и WRH# генерируются в режимах Write Strobe и Address Valid with Strobe Write

Окончание таблицы В.44

1	2	3	4
3	ALE	Выбор режима Address Valid Strobe	Выбор сигналов, извещающих о фиксации адреса: «1» – ALE фиксирует правильный адрес в режимах Standard Bus и Write Strobe. «0» – ADV# фиксирует адрес вместо ALE и может быть использован в качестве сигнала выбора кристалла внешней памяти
4,5	IRC0, IRC1	Внутреннее управление готовностью внешних устройств	Определяют число состояний ожидания, которые могут быть введены, пока контакт READY поддерживается в низком состоянии. Состояние ожидания поддерживается до тех пор, пока либо сигнал READY не станет высоким, либо не будет достигнуто заданное число состояний ожидания. IRC1 IRC0 Количество состояний ожидания 0 0 1 0 1 2 1 0 3 1 1 управляется только сигналом READY
6,7	LOC0, LOC1	Биты блокировки	Определяют вид программной защиты внутренней памяти. LOC1 LOC0 Вид защиты 0 0 Нет защиты 0 1 Защита только записи 1 0 Защита только чтения 1 1 Защита чтения и записи

CLKCL Регистр управления тактовыми сигналами устройств 1

1FF0H (чтение/запись)

Регистр управления периферийными устройствами управляет тактированием (включением/выключением) отдельных периферийных устройств, переводом в режим SLEEP аналоговых блоков АЦП и ЦАП, включением режима SLOW.

Рисунок В.38 – Формат регистра CLKCL

Таблица В.45 – Функциональные назначения полей регистра CLKCL

Номер бита	Мнемоника	Имя бита	Описание
0	UART0EN	Разрешение работы UART0	Подается ли тактовый сигнал на блок UART0: «1» – подается; «0» – не подается
1	HSIOEN	Разрешение работы HSIO	Подается ли тактовый сигнал на блок HSIO: «1» – подается; «0» – не подается
2	EXRAMEN	Разрешение работы EXRAM	Разрешено ли использование расширенного ОЗУ. При запрещении область EXRAM назначается во внешнюю память. «1» – разрешено; «0» – не разрешено
3	PTSEN	Разрешение работы PTS	Подается ли тактовый сигнал на блок PTS: «1» – подается; «0» – не подается
4	DEBUGEN	Разрешение работы модуля отладки	Подается ли тактовый сигнал на блок отладки: «1» – подается; «0» – не подается
5	ADCEN	Разрешение работы АЦП	Определяет, подается ли тактовый сигнал на блок АЦП. При отключении тактового сигнала аналоговая часть переводится в режим SLEEP. «1» – подается; «0» – не подается
6	DACEN	Разрешение работы ЦАП	Подается ли тактовый сигнал на блок ЦАП. При отключении тактового сигнала аналоговая часть переводится в режим SLEEP. «1» – подается; «0» – не подается
7	SLOW	Управление режимом SLOW	Включен ли режим медленного тактирования или нет: «1» – включен; «0» – не включен

CLKCLDEB Регистр перезагрузки CLKCL

1FF6H (чтение/запись)

Регистр содержит значение, которое будет загружаться в регистр управления тактовыми сигналами устройств CLKCL (при установленном бите CLKCHDEB.7) одновременно с запросом на прерывание DEBUG. Это позволяет выключить тактовый сигнал с определенных устройств для того, чтобы их регистры не изменили значение. Перезапись регистра CLKCL происходит только в момент запроса на прерывание.

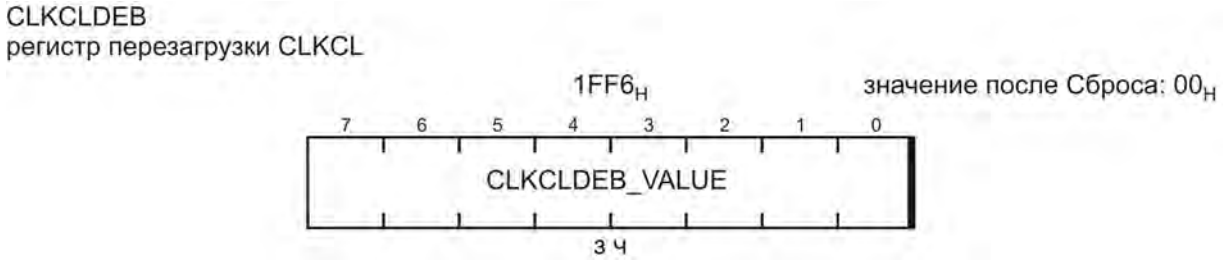


Рисунок В.39 – Формат регистра CLKCLDEB

Таблица В.46– Функциональные назначения полей регистра CLKCLDEB

Номер бита	Мнемоника	Имя бита	Описание
0-7	CLKCLDEB_VALUE	Данные для загрузки	Данные, которые будут загружены в CLKCL

CLKCH Регистр управления тактовыми сигналами устройств 2

1FF1H (чтение/запись)

Регистр управления периферийными устройствами управляет тактированием (включением/выключением) отдельных периферийных устройств.

Рисунок В.40 – Формат регистра CLKCH

Таблица В.47 – Функциональные назначения полей регистра CLKCH

Номер бита	Мнемоника	Имя бита	Описание
0	SPIEN	Разрешение работы SPI	Подается ли тактовый сигнал на блок SPI: «1» – подается; «0» – не подается
1	I2CEN	Разрешение работы I2C	Подается ли тактовый сигнал на блок I2C: «1» – подается; «0» – не подается
2	UART1EN	Разрешение работы UART1	Подается ли тактовый сигнал на блок UART1: «1» – подается; «0» – не подается
3-7	-	-	Зарезервированы

CLKCHDEB Регистр перезагрузки CLKCH

1FF7H (чтение/запись)

Регистр содержит значение, которое будет загружаться в биты 0 – 6 регистра управления тактовыми сигналами устройств CLKCH (при установленном бите CLKCHDEB.7) одновременно с запросом на прерывание DEBUG. Это позволяет выключить тактовый сигнал с определенных устройств для того, чтобы их регистры не изменили значение. Бит разрешения перезаписи не сбрасывается аппаратно, но перезапись регистра CLKCH происходит только в момент запроса на прерывание.

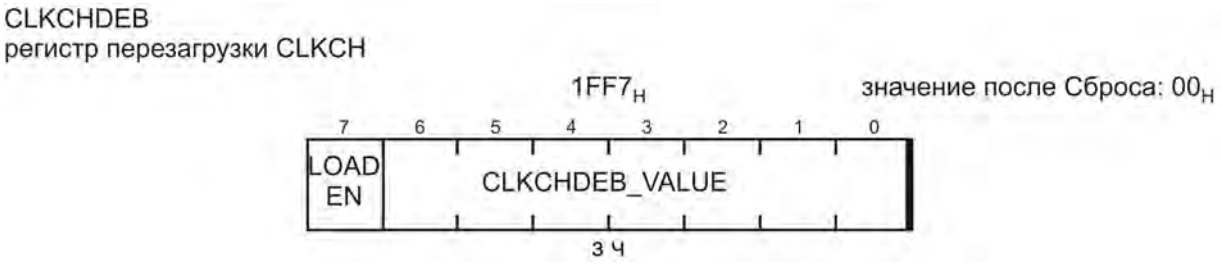


Рисунок В.41 – Формат регистра CLKCHDEB

Таблица В.48 – Функциональные назначения полей регистра CLKCLDEB

Номер бита	Мнемоника	Имя бита	Описание
0-6	CLKCHDEB_VALUE	Данные для загрузки	Данные, которые будут загружены в CLKCH
7	LOADEN	Разрешение перезаписи регистров управления тактированием	Бит управляет перезаписью регистров CLKL и CLKCH: «1» – разрешена перезапись по запросу на прерывание DEBUG; «0» – перезапись запрещена

CURRPC Регистр текущего значения основного счетчика команд

1FE9H/1FE8H (чтение)
Регистр содержит текущее значение основного счетчика команд.

Рисунок В.42 – Формат регистра CURRPC

Таблица В.49 – Функциональные назначения полей регистра CURRPC

Номер бита	Мнемоника	Имя бита	Описание
0-7	CURRPC_LO	Младший байт счетчика команд	Содержит младший байт текущего значения основного счетчика команд
8-15	CURRPC_HI	Старший байт счетчика команд	Содержит старший байт текущего значения основного счетчика команд

DACVAL Регистр управления ЦАП

11H/10H

Горизонтальное окно 1 (запись/чтение)

Регистр управления ЦАП выбирает ток, протекающий через выводы IOUTA, IOUTB. Также этот регистр отвечает за перевод блока ЦАП в режим пониженного энергопотребления.

Рисунок В.43 – Формат регистра DACVAL

Таблица В.50 – Функциональные назначения полей регистра DACVAL

Номер бита	Мнемоника	Имя бита	Описание
0-7	DB0 – DB7	Младшие разряды тока выхода ЦАП	Эти биты определяют ток, протекающий через выводы ЦАП
8-13	DB8 – DB13	Старшие разряды тока выхода ЦАП	Эти биты определяют ток, протекающий через выводы ЦАП
14	SLEEP	Бит ручного перевода в режим пониженного потребления	Совместно с битом SLEEPC управляет режмом пониженного потребления: «1» – ЦАП переводится в режим пониженного потребления; «0» – ЦАП находится в активном режиме
15	SLEEPC	Автоматическое управление режимом пониженного потребления	Бит управляет переводом ЦАП в режим пониженного потребления: «1» – перевод осуществляется автоматически при включении режима POWERDOWN или выключении тактового сигнала; «0» – управление осуществляется битом SLEEP

DEBADDR Регистр значения адреса операнда отладки

1FE5H/1FE4H (чтение/запись)

Регистр содержит значение адреса операнда, по появлению которого на шине адресов модуль отладки зафиксирует событие. Адрес операнда должен быть представлен в полном, 16-разрядном виде. Тип действия по этому событию определяется в регистре DEBCTRL.

Рисунок В.44 – Формат регистра DEBADDR

Таблица В.51 – Функциональные назначения полей регистра DEBADDR

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBADDR_LO	Младший байт адреса операнда	Содержит младший байт адреса операнда модуля отладки.
8-15	DEBADDR_HI	Старший байт адреса операнда	Содержит старший байт адреса операнда модуля отладки.

DEBCTRL Регистр управления модулем отладки 1

1FE6H (чтение/запись)

Регистр управляет событиями модуля отладки: запрещает события, выбирает тип отклика на событие. После сброса регистр выставляется в состояние, запрещающее любые действия модулю отладки.

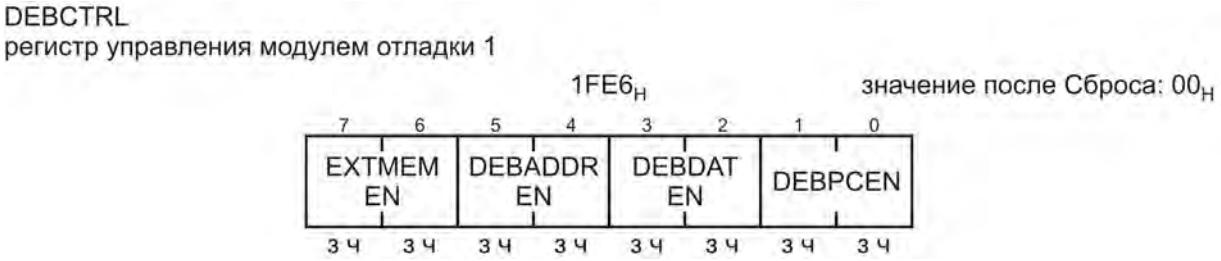


Рисунок В.45– Формат регистра DEBCTRL

Таблица В.52 – Функциональные назначения полей регистра DEBCTRL

Номер бита	Мнемоника	Имя бита	Описание
0-1	DEBPCEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено в случае, если значение счетчика команд будет больше значения, заданного в регистре DEBPC: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
2-3	DEBDATEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по появлению значения, заданного в регистре DEBDAT, на внутренней шине данных: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
4-5	DEBADDREN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по появлению значения, заданного в регистре DEBADDR, на внутренней шине адресов операндов: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
6-7	EXTMEMEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по событию-выборке команд из области внешней памяти: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС

DEBCTRH Регистр управления модулем отладки 2

1FE7H (чтение/запись)

Регистр управляет событиями модуля отладки: запрещает события, выбирает тип отклика на событие. После сброса регистр выставляется в состояние, запрещающее любые действия модулю отладки.

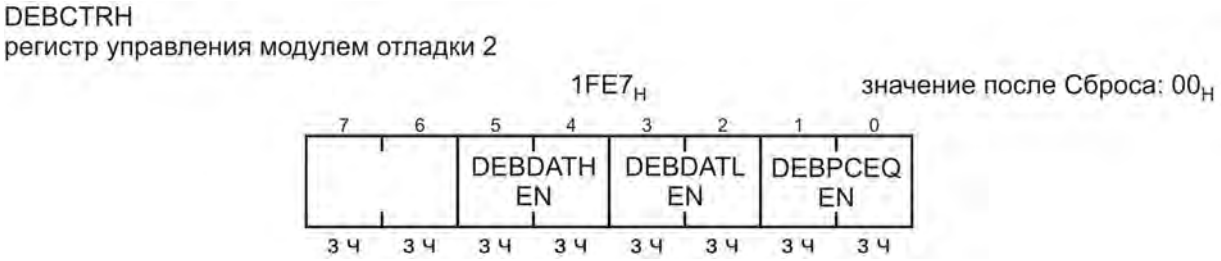


Рисунок В.46 – Формат регистра DEBCTRL

Таблица В.53 – Функциональные назначения полей регистра DEBCTRL

Номер бита	Мнемоника	Имя бита	Описание
0-1	DEBPCEQEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по совпадению текущего счетчика команд со значением, заданным в регистре DEBPCEQ: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
2-3	DEBDATLEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по появлению значения, заданного в регистре DEBDATL, на младших разрядах внутренней шины данных: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
4-5	DEBDATHEN	Управление событием модуля отладки	Определяет, какое действие будет выполнено по появлению значения, заданного в регистре DEBDATH, на старших разрядах внутренней шины данных: 00 = нет действия 01 = прерывание DEBUG (2060H) 10 = перевод в IDLE 11 = сброс ИС
6-7	–	–	Зарезервировано

DEBDAT Регистр значения данных отладки

1FE3H/1FE2H (чтение/запись)

Регистр содержит значение данных, при появлении которого на шине данных модуль отладки зафиксирует событие. Модуль отладки фиксирует только полное совпадение 16-разрядного слова. Тип действия по этому событию определяется в регистре DEBCTRL.

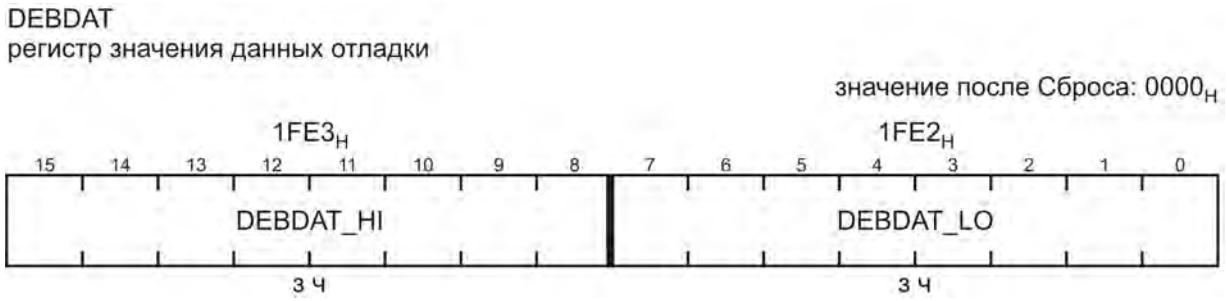


Рисунок В.47 – Формат регистра DEBDAT

Таблица В.54 – Функциональные назначения полей регистра DEBDAT

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBDAT_LO	Значение младшего байта данных для отладки	Содержит младший байт данных, при появлении которого модуль отладки зафиксирует событие
8-15	DEBDAT_HI	Значение старшего байта данных для отладки	Содержит старший байт данных, при появлении которого модуль отладки зафиксирует событие

DEBDATH Регистр значения старшего байта данных отладки

1FEFH (чтение/запись)

Регистр содержит значение байта данных, при появлении которого на старших разрядах внутренней шины данных модуль отладки зафиксирует событие. Тип действия по этому событию определяется в регистре DEBCTRH.

Рисунок В.48 – Формат регистра DEBDATH

Таблица В.55 – Функциональные назначения полей регистра DEBDATH

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBDATH	Значение старшего байта данных для отладки	Содержит байт данных, при появлении которого на старших разрядах внутренней шины данных модуль отладки зафиксирует событие

DEBDATL Регистр значения младшего байта данных отладки

1FEEH (чтение/запись)

Регистр содержит значение байта данных, при появлении которого на младших разрядах внутренней шины данных модуль отладки зафиксирует событие. Тип действия по этому событию определяется в регистре DEBCTRH.

Рисунок В.49 – Формат регистра DEBDATL

Таблица В.56 – Функциональные назначения полей регистра DEBDATL

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBDATL	Значение младшего байта данных для отладки	Содержит байт данных, при появлении которого на младших разрядах внутренней шины данных модуль отладки зафиксирует событие

DEBPC Регистр значения счетчика команд для отладки 1

1FE1H/1FE0H (чтение/запись)

Регистр содержит значение счетчика команд, при превышении которого модуль отладки зафиксирует соответствующее событие. Тип действия по этому событию определяется в регистре DEBCTRL.

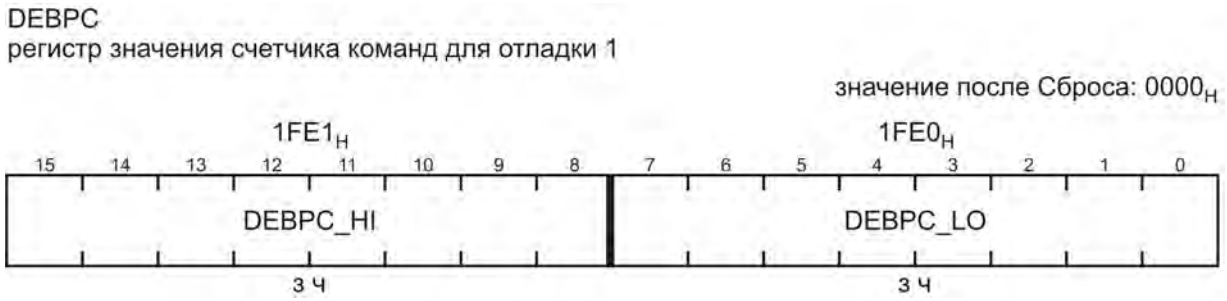


Рисунок В.50 – Формат регистра DEBPC

Таблица В.57 – Функциональные назначения полей регистра DEBPC

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBPC_LO	Значение младшего байта счетчика команд	Содержит младший байт основного счетчика команд, при превышении которого модуль отладки фиксирует событие
8-15	DEBPC_HI	Значение старшего байта счетчика команд	Содержит старший байт основного счетчика команд, при превышении которого модуль отладки фиксирует событие

DEBPCEQ Регистр значения счетчика команд для отладки 2

1FEDH/1FECH (чтение/запись)

Регистр содержит значение счетчика команд, при достижении которого модуль отладки зафиксирует соответствующее событие. Тип действия по этому событию определяется в регистре DEBCTRH.

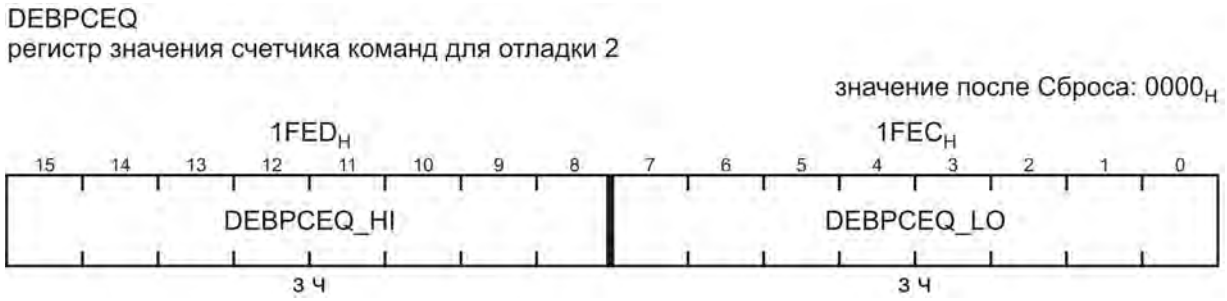


Рисунок В.51 – Формат регистра DEBPCEQ

Таблица В.58 – Функциональные назначения полей регистра DEBPCEQ

Номер бита	Мнемоника	Имя бита	Описание
0-7	DEBPCEQ_LO	Значение младшего байта счетчика команд	Содержит младший байт основного счетчика команд, при достижении которого модуль отладки фиксирует событие
8-15	DEBPCEQ_HI	Значение старшего байта счетчика команд	Содержит старший байт основного счетчика команд, при достижении которого модуль отладки фиксирует событие

HAPPPC Регистр фиксации значения PC модулем отладки

1FEBH/1FEAH (чтение/запись)

Регистр содержит значение счетчика команд на момент возникновения события в модуле отладки. Не сбрасывается во время RESET, поэтому может фиксировать адрес счетчика команд, при котором модуль отладки сформирует запрос на сброс.

Рисунок В.52 – Формат регистра HAPPPC

Таблица В.59 – Функциональные назначения полей регистра HAPPPC

Номер бита	Мнемоника	Имя бита	Описание
0-7	HAPPPC_LO	Значение младшего байта фиксации счетчика команд	Содержит младший байт значения основного счетчика команд на момент возникновения события
8-15	HAPPPC_HI	Значение Старшего байта фиксации счетчика команд	Содержит старший байт значения основного счетчика команд на момент возникновения события

HSI_MODE Регистр режима HSI

03H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Режим высокоскоростного ввода определяет для каждого канала HSI, какое из четырех типов событий будет фиксироваться в буфере HSI FIFO: каждый передний фронт, каждый задний фронт, любой фронт (передний или задний) или серия из восьми передних фронтов. Каналы должны индивидуально получить разрешающие сигналы через регистр IOC0.

HSI_MODE
регистр режима HSI

03H значение после Сброса: 00H

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

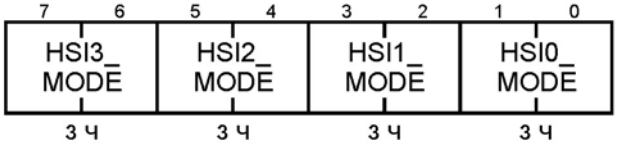


Рисунок В.53 – Формат регистра HIS_MODE

Таблица В.60 – Функциональные назначения полей регистра HIS_MODE

Номер бита	Мнемоника	Имя бита	Описание
0-1	HSI0_MODE	Режим HSI.0	Режим работы нулевого канала HSI.0
2-3	HSI1_MODE	Режим HSI.1	Режим работы первого канала HSI.1
4-5	HSI2_MODE	Режим HSI.2	Режим работы второго канала HSI.2
6-7	HSI3_MODE	Режим HSI.3	Режим работы третьего канала HSI.3

Каждое 2-битное поле определяет режим работы для соответствующего канала.

Таблица В.61 – Кодирование режимов работы для модуля HSI

Бит 1	Бит 0	Описание
0	0	В буфере HSI FIFO фиксируется каждый восьмой передний фронт
0	1	В буфере HSI FIFO фиксируется каждый передний фронт
1	0	В буфере HSI FIFO фиксируется каждый задний фронт
1	1	В буфере HSI FIFO фиксируется любой фронт (передний и задний)

HSI_STATUS Регистр состояния HSI

06H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись битов 0, 2, 4, 6)

Регистр HSI_STATUS показывает наличие событий в HSI и текущие состояния каналов. Регистр HSI_TIME содержит временную метку. Чтение регистра HSI_ TIME перезагружает данный регистр. Если HSI_TIME считывается перед HSI_STATUS, то информация о состоянии, ассоциируемая с временной меткой регистра HSI_TIME, теряется.

Если данный HSI регистр не содержит событий, то биты состояния событий регистра не определены, однако биты состояния каналов могут быть считаны в любое время. Это позволяет читать каналы HSI как входы даже тогда, когда они не используются в HSI-устройстве. Запись в HSI_STATUS в горизонтальном окне 15 устанавливает биты состояния событий, но не влияет на состояние выводов. Текущее состояние вывода не обязательно соответствует статусному биту события, так как другие события могут произойти со времени последнего записанного состояния каналов.

Регистр IOC0 управляет альтернативными функциями HSI.0 – HSI.3, а регистр HSI_MODE управляет событиями каналов, которые будут захватываться системой HSI в очередь FIFO.

HSI_STATUS
регистр состояния HSI

06H значение после Сброса: X0X0X0X0B

(горизонтальное окно 0 - чтение, горизонтальное окно 15 - запись битов 0, 2, 4, 6)

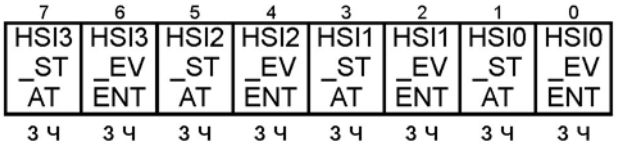


Рисунок В.54 – Формат регистра HIS_STATUS

Таблица В. 62 – Функциональные назначения полей регистра HSI_STATUS

Номер бита	Мнемоника	Имя бита	Описание
0	HSI0_EVENT	Событие канала HSI.0	«1» – событие в канале HSI.0 имеет место; «0» – событий нет
1	HSI0_STAT	Состояние канала HSI.0	Текущее состояние канала HSI.0
2	HSI1_EVENT	Событие канала HSI.1	«1» – событие в канале HSI.1 имеет место; «0» – событий нет
3	HSI1_STAT	Состояние канала HSI.1	Текущее состояние канала HSI.1
4	HSI2_EVENT	Событие канала HSI.2	«1» – событие в канале HSI.2 имеет место; «0» – событий нет
5	HSI2_STAT	Состояние канала HSI.2	Текущее состояние канала HSI.2
6	HSI3_EVENT	Событие канала HSI.3	«1» – событие в канале HSI.3 имеет место; «0» – событий нет
7	HSI3_STAT	Состояние канала HSI.3	Текущее состояние канала HSI3

HSI_TIME Регистр времени HSI

04H, 05H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Регистр HSI-TIME содержит время, в которое происходит HSI-событие. Когда событие в канале HSI имеет место, регистр HSI_TIME загружается текущим 16-битным значением таймера 1. Время хранится в буфере HSI FIFO вместе с четырьмя битами состояния событий из регистра HSI_STATUS.

Чтение регистра HSI_TIME перегружает регистр HSI_STATUS. Если считывается HSI_TIME перед HSI_STATUS, информация, связанная с временной меткой в регистре HSI_TIME, теряется. Если регистр HSI_STATUS не содержит событий, HSI_TIME не определён. Запись в HSI_TIME в горизонтальном окне 15 перезагружает регистр хранения, перезаписывая другие данные.

Рисунок В.55 – Формат регистра HIS_TIME

Таблица В.63 – Функциональные назначения полей регистра HIS_TIME

Номер бита	Мнемоника	Имя бита	Описание
0-7	HSI_TIME (LO)	Время событий HSI	Младшие восемь битов времени событий высокоскоростного ввода
8-15	HSI_TIME (HI)	Время событий HSI	Старшие восемь битов времени событий высокоскоростного ввода

HSO_COMMAND Регистр управления HSO

06H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Модуль высокоскоростного вывода (HSO) может генерировать события в определённое время с минимальными затратами CPU. Регистр управления HSO определяет, какое событие (события) будет иметь в модуле HSO время, определённое регистром HSO_TIME.

HSO_COMMAND
регистр управления HSO

06H значение после Сброса: XXXXH

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

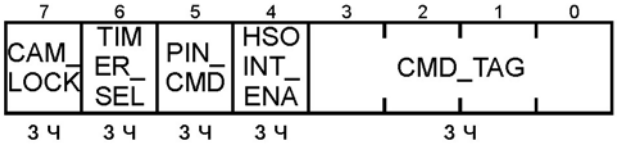


Рисунок В.56 – Формат регистра HSO_COMMAND

Таблица В.64 – Функциональные назначения полей регистра HSO_COMMAND

Номер бита	Мнемоника	Имя бита	Описание
0-3	CMD_TAG	Команды HSO	Определяет, какое событие (события) будет (будут) происходить во время, заданное в HSO_TIME
4	HSOINT_ENA	Разрешение/ Запрет HSO прерывания	Определяет, будет ли событие HSO генерировать прерывание. «1» – генерация прерывания «0» – нет прерывания Когда этот бит установлен, программируемый вывод генерирует прерывание HSO (INT03, 2006H), а внутренние события генерируют прерывания Software Timer (INT05, 200AH). После генерации прерывания биты состояния событий HSO в регистрах IOS1 и IOS2 должны быть опрошены для определения, какое событие вызвало прерывание
5	PIN_CMD	Установка/ Сброс выбранного вывода HSO	Определяет, будет ли команда сбрасывать определённый вывод или выводы. «1» – установка вывода (выводов) «0» – сброс вывода (выводов)
6	TIMER_SEL	Выбор таймера 1 / таймера 2	Выбирает таймер для команд HSO. «0» –выбор таймера 1 «1» – выбор таймера 2
7	CAM_LOCK	Захват команды в CAM	Когда IOC2.6 установлен (возможен захват команд), этот бит определяет, будет ли команда HSO оставаться в CAM или сбрасываться после выполнения. «1» – захват команды в CAM «0» – сброс команды из CAM после выполнения Запись «1» в IOC2.7 сбрасывает все команды из CAM, как и сброс кристалла

Таблица B.65 – Кодирование CMD_TAG

Бит 3	Бит 2	Бит 1	Бит 0	Обозначение команды	Определение
0	0	0	0	HSO0	Переключение на HSO.0
0	0	0	1	HSO1	Переключение на HSO.1
0	0	1	0	HSO2	Переключение на HSO.2
0	0	1	1	HSO3	Переключение на HSO.3
0	1	0	0	HSO4	Переключение на HSO.4
0	1	0	1	HSO5	Переключение на HSO.5
0	1	1	0	HSO01 *	Переключение на HSO.0 и HSO.1
0	1	1	1	HSO23 *	Переключение на HSO.2 и HSO.3
1	0	0	0	SWT0	Программирование таймера 0
1	0	0	1	SWT1	Программирование таймера 1
1	0	1	0	SWT2	Программирование таймера 2
1	0	1	1	SWT3	Программирование таймера 3
1	1	0	0	HSOALL *	Переключение на HSO.0 – HSO.5
1	1	0	1	–	Зарезервировано; не используется
1	1	1	0	T2RST	Сброс таймера 2
1	1	1	1	A_D	Начало аналого-цифрового преобразования

_____

* В этих комбинациях битов два или более выводов устанавливаются или сбрасываются одновременно.

HSO_TIME Регистр времени HSO

05/04H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Регистр времени HSO определяет время выполнения команды HSO. Команды определяются битами HSO_COMMAND.0 – HSO_COMMAND.3, а таймер выбирается битом HSO_COMMAND.6. Когда события загружаются в CAM, сначала записываются в HSO_COMMAND, затем – в HSO_TIME.

Рисунок В.57 – Формат регистра HSO_TIME

Таблица В.66 – Функциональные назначения полей регистра HSO_TIME

Номер бита	Мнемоника	Имя бита	Описание
0-7	HSO_TIME (LO)	Время выполнения команд HSO, младший байт	Это младший байт регистра времени выполнения команд HSO
8-15	HSO_TIME (HI)	Время выполнения команд HSO, старший байт	Это старший байт регистра времени выполнения команд HSO

INT_MASK Регистр масок прерываний

08H

Все горизонтальные окна (чтение/запись)

Регистр INT_MASK разрешает или запрещает (маскирует) отдельные прерывания. (PSW.2 глобально разрешает или запрещает обслуживание всех маскируемых прерываний.) Из INT_MASK можно читать и записывать как в байтовый регистр во всех горизонтальных окнах. INT_MASK – младший байт PSW (слова состояния программы); следовательно, PUSHF или PUSHA записывают содержимое этого регистра в стек, а POPF или POPA восстанавливают его.

Рисунок В.58 – Формат регистра INT_MASK

Таблица В.67 – Функциональные назначения полей регистра INT_MASK

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	TIMER_MASK	Timer 1 или Timer 2 Overflow	Установка этого бита разрешает прерывание Timer Overflow (INT00, 2000H). Таймер 1 и таймер 2 оба могут генерировать прерывание INT00. Установка IOC1.2 выбирает в качестве источника таймер 1; а установка IOC1.3 – таймер 2. Таймер 2 может генерировать или INT00 или INT12, но не оба одновременно
1	AD_MASK	A/D Conversion	Установка этого бита разрешает прерывание по завершению АЦП (INT01, 2002H)
2	HSIDAT_MASK	HSI Data Available/ FIFO Full	Установка этого бита разрешает прерывание HSI Data Available (INT02, 2004H). IOC1.7 выбирает источник прерывания.
3	HSO_MASK	HSO Output Event	Установка этого бита разрешает прерывание HSO (INT03, 2004H).
4	HSI0_MASK	HSI. 0 внешнее прерывание	Установка этого бита разрешает прерывание от внешнего контакта HSI.0 (INT04, 2008H)
5	SWT_MASK	Software Timer	Установка этого бита разрешает прерывание Software Timer (INT05, 200AH)
6	SER_MASK	Serial Porты	Установка этого бита разрешает прерывание Serial Ports (INT06, 200CH)
7	EXTINT_MASK	Прерывание от EXTINT или P0.7	Установка этого бита разрешает прерывание EXTINT (INT07, 200EH). IOC1.1 выбирает источник прерывания (P0.7, P2.2)

INT_MASK1 Регистр масок прерываний 1

13H

Все горизонтальные окна (чтение/запись)

INT_MASK1 разрешает или запрещает (маскирует) индивидуальные прерывания (PSW.2 глобально разрешает или запрещает обслуживание всех маскируемых прерываний). INT_MASK1 можно читать и записывать во всех горизонтальных окнах. PUSHA записывает содержимое этого регистра в стек, а POPA – восстанавливает его.

Рисунок В.59 – Формат регистра INT_MASK1

Таблица В.68 – Функциональные назначения полей регистра

Номер бита	Мнемоника	Имя бита	Описание
0	TI_MASK	Прерывание Transmit UART	Установка этого бита разрешает прерывание Transmit UART (INT08, 2030H).
1	RI_MASK	Прерывание Receive UART	Установка этого бита разрешает прерывание Receive UART (INT09, 2032H).
2	HSI4_MASK	Прерывание HSO FIFO 4	Установка этого бита разрешает прерывание HSI FIFO 4 (INT10, 2034H)
3	T2CAP_MASK	Прерывание Timer 2 Capture	Установка этого бита разрешает прерывание Timer 2 Capture (INT11, 2036H)
4	T2OVF_MASK	Прерывание Timer 2 Overflow	Установка этого бита разрешает прерывание Timer 2 Overflow (INT12, 2038H). IOC2.5 выбирает предел переполнения для INT12
5	EXTINT1_MASK	Прерывание от контакта EXTINT1	Установка этого бита разрешает прерывание EXTINT1 (INT13, 203AH). P2. 2 может генерировать или прерывание EXTINT (INT07) или прерывание EXTINT1 (INT13)
6	FIFO_MASK	Прерывание HSI FIFO Full	Установка этого бита разрешает прерывание HSI FIFO Full (INT14, 203CH)
7	NMI_MASK	NMI	Этот нефункционирующий бит маски существует условно. Немаскируемое прерывание (NMI) разрешено и при 0, и при 1. В этот бит всегда записывают ноль

INT_PEND Регистр ожидания прерываний

09H

Все горизонтальные окна (чтение/запись)

Когда аппаратное обеспечение обнаруживает запрос на прерывание, оно устанавливает соответствующий бит в INT_PEND или INT_PEND1. Когда процессор перешел на обработку прерывания по соответствующему вектору, аппаратное обеспечение очищает бит запроса. Регистр INT_PEND можно читать или записывать во всех горизонтальных окнах. Программное обеспечение может генерировать прерывание установкой соответствующего бита запроса прерывания.

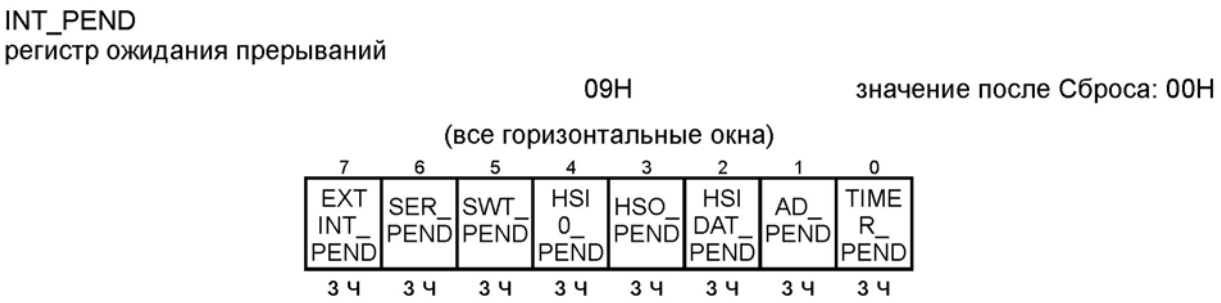


Рисунок В.60 – Формат регистра INT_PEND

Таблица В.69 – Функциональные назначения полей регистра INT_PEND

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	TIMER_PEND	Timer 1 или Timer 2 Overflow	Когда этот бит установлен, он показывает на запрос прерывания Timer Overflow (INT00). Он очищается, когда произошел переход по вектору с адресом 2000H
1	AD_PEND	A/D Conversion	Когда этот бит установлен, он показывает на запрос прерывания A/D Conversion (INT01). Он очищается, когда произошел переход по вектору с адресом 2002H
2	HSIDAT_PEND	HSI Data Available / FIFO Full	Когда этот бит установлен, он показывает на запрос прерывания HSI Data Available (INT02). Он очищается, когда произошел переход по вектору с адресом 2004H
3	HSO_PEND	HSO Output Event	Когда этот бит установлен, он показывает на запрос прерывания HSO (INT03). Он очищается, когда произошел переход по вектору с адресом 2006H
4	HSI0_PEND	HSI.0 внешнее прерывание	Когда этот бит установлен, он показывает на запрос прерывания HSI.0 (INT04). Он очищается, когда произошел переход по вектору с адресом 2008H
5	SWT_PEND	Software Timer	Когда этот бит установлен, он показывает на запрос прерывания Software Timer (INT05). Он очищается, когда произошел переход по вектору с адресом 200AH
6	SER_PEND	Serial Ports	Когда этот бит установлен, он показывает на запрос прерывания Serial Ports (INT06). Он очищается, когда произошел переход по вектору с адресом 200CH
7	EXTINT_PEND	Прерывание от EXTINT или P0.7	Когда этот бит установлен, он показывает на запрос прерывания EXTINT (INT07). Он очищается, когда произошел переход по вектору с адресом 200EH

INT_PEND1 Регистр ожидания прерываний 1

12H

Все горизонтальные окна (чтение/запись)

Когда аппаратное обеспечение обнаруживает запрос на прерывание, оно устанавливает соответствующий бит в INT_PEND или INT_PEND1. Когда процессор перешел на обработку прерывания по соответствующему вектору, аппаратное обеспечение очищает бит запроса. Регистр INT_PEND1 можно читать или записывать во всех горизонтальных окнах. Программное обеспечение может генерировать прерывание установкой соответствующего бита запроса прерывания.

INT_PEND1
регистр ожидания прерываний 1

12H значение после Сброса: 00H

(все горизонтальные окна)

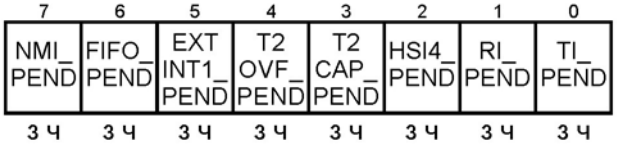


Рисунок В.61 – Формат регистра INT_PEND1

Таблица В.70 – Функциональные назначения полей регистра INT_PEND1

Номер бита	Мнемоника	Имя бита	Описание
0	TI_PEND	Прерывание Transmit UART	Когда этот бит установлен, он показывает на запрос прерывания Transmit UART(INT08). Он очищается, когда произошел переход по вектору с адресом 2030H
1	RI_PEND	Прерывание Receive UART	Когда этот бит установлен, он показывает на запрос прерывания Receive UART(INT09). Он очищается, когда произошел переход по вектору с адресом 2032H
2	HSI4_PEND	Прерывание HSO FIFO 4	Когда этот бит установлен, он показывает на запрос прерывания HSO FIFO 4 (INT10). Он очищается, когда произошел переход по вектору с адресом 2034H
3	T2CAP_PEND	Прерывание Timer 2 Capture	Когда этот бит установлен, он показывает на запрос прерывания Timer 2 Capture (INT11). Он очищается, когда произошел переход по вектору с адресом 2036H
4	T2OVF_PEND	Прерывание Timer 2 Overflow	Когда этот бит установлен, он показывает на запрос прерывания Timer 2 Overflow (INT12). Он очищается, когда произошел переход по вектору с адресом 2038H
5	EXTINT1_PEND	Прерывание от контакта EXTINT1	Когда этот бит установлен, он показывает на запрос прерывания EXTINT1 (INT13). Он очищается, когда произошел переход по вектору с адресом 203AH
6	FIFO_PEND	Прерывание HSI FIFO Full	Когда этот бит установлен, он показывает на запрос прерывания HSI FIFO Full (INT14). Он очищается, когда произошел переход по вектору с адресом 203CH
7	NMI_PEND	NMI	Когда этот бит установлен, он показывает на запрос прерывания NMI (INT15). Он очищается, когда произошел переход по вектору с адресом 203EH

IOC0 Регистр 0 управления вводом-выводом

15H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Регистр IOC0 выбирает внешний генератор тактовых импульсов и источники перезагрузки таймера 2 и разрешает или запрещает функции HSI для четырех выводов HSI. Когда IOC0 считывается из горизонтального окна 15, IOC.1 всегда читается как «1», потому что его значение не буферизуется.

IOC0
регистр 0 управления вводом/выводом

15H значение после Сброса: 0000 00X0B

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

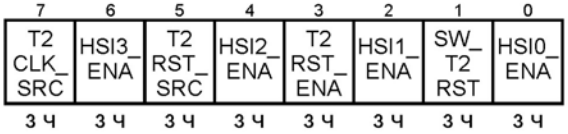


Рисунок В.62 – Формат регистра IOC0

Таблица В.71 – Функциональные назначения полей регистра IOC0

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	HSI0_ENA	Разрешение HSI.0 как входа HSI	Этот бит определяет, будут ли состояния на выводе HSI.0 фиксироваться в буфере HSI FIFO: «1» – разрешение HSI; «0» – запрещение HSI
1	SW_T2RST	Программный сброс таймера 2	Запись «1» в этот бит сбрасывает таймер 2. Этот бит всегда читается как «1» в горизонтальном окне 15. «1» – сброс таймера 2 после каждой записи; «0» – нет действия
2	HSI1_ENA	Разрешение HSI.1 как входа HSI	Этот бит определяет, будут ли состояния на выводе HSI.1 фиксироваться в буфере HSI FIFO «1» – разрешение HSI; «0» – запрещение HSI
3	T2RST_ENA	Источник внешнего сброса таймера 2	Этот бит разрешает внешний сброс таймера 2. Источник внешнего сброса – нарастающий фронт либо T2RST, либо HSI.0, как выбрано в IOC0.5. «1» – разрешение внешнего сброса; «0» – запрещение
4	HSI2_ENA	Разрешение HSI.2 как входа HSI	Этот бит определяет, будут ли состояния на выводе HSI.2 фиксироваться в буфере HSI FIFO «1» – разрешение HSI; «0» – запрещение HSI
5	T2RST_SRC	Источник сброса таймера 2	Этот бит выбирает источник внешнего сброса таймера 2. IOC0.3 должен быть установлен для разрешения внешнего сброса. «1» – нарастающий фронт на выводе HSI.0; «0» – нарастающий фронт на выводе (P2.4) T2RST
6	HSI3_ENA	Разрешение HSI.3 как входа HSI	Этот бит определяет, будут ли состояния на выводе HSI. 3 фиксироваться в буфере HSI FIFO «1» – разрешение HSI; «0» – запрещение HSI
7	T2CLK_SRC	Входной тактовый сигнал таймера 2	Этот бит выбирает внешний источник тактовых импульсов таймера 2. IOC3.0 должен быть очищен для разрешения внешнего источника тактовых импульсов. «1» – вывод HSI.1; «0» – вывод T2CLK (P2.3)

IOC1 Регистр 1 управления вводом-выводом

16H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Регистр IOC1 выбирает выходные функции P2.5 и P2.0; разрешает или запрещает вывод с HSO.4 или HSO.5; выбирает источники прерываний для EXTINT (INT07, 200EH), переполнения таймера (INT00, 2000H) и доступности HSI данных (INT02, 2004H).

IOC1
регистр 1 управления вводом/выводом

16H значение после Сброса: 21H

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

7	6	5	4	3	2	1	0
HSI_ INT	HSO 5_ ENA	TXD_ SEL	HSO 4_ 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

Рисунок В.63 – Формат регистра IOC1

Таблица В.72 – Функциональные назначения полей регистра IOC1

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	PWM_SEL	Выбор вывода P2.5/ PWM	Этот бит определяет, будет ли P2.5 функционировать как PWM выход или как стандартный вывод порта. «1» – выход PWM; «0» – стандартный вывод порта
1	EXTINT_SRC	Выбор источника внешнего прерывания INT07	Этот бит выбирает источник внешнего прерывания EXTINT (INT07, 200EH) для разрешения быть установленным «1» – P0.7; «0» – P2.2
2	T1OVF_INT	Разрешение прерывания Timer 1 Overflow	И таймер 1 и таймер 2 могут генерировать прерывание Timer Overflow (INT00, 2000H). Этот бит определяет, будет ли переполнение таймера 1 генерировать прерывание. Для разрешения прерывания IOS1.5 показывает статус прерывания. «1» – источник прерывания таймер 1; «0» – таймер 1 не является источником
3	T2OVF_INT	Разрешение прерывания Timer 2 Overflow	И таймер 1 и таймер 2 могут генерировать прерывание Timer Overflow (INT00, 2000H). Этот бит определяет, будет ли переполнение таймера 2 генерировать прерывание. Для разрешения прерывания IOS1.4 показывает статус прерывания. «1» – источник прерывания таймер 2; «0» – таймер 2 не является источником

Окончание таблицы В.72

1	2	3	4
4	HSO4_ENA	Разрешение работы выхода HSO.4	HSO.4 мультиплексируется с выводом HSI.2. Этот бит разрешает работу HSO.4. «1» – разрешает вывод; «0» – запрещает вывод
5	TXD_SEL	Выбор вывода P2.0/ TXD0 и P2.6/ TXD1	Этот бит определяет, будет ли P2.0 и P2.6 функционировать как выход последовательного порта передатчика или как стандартный вывод порта. «1» – выход последовательного порта TXD; «0» – стандартный вывод порта
6	HSO5_ENA	Разрешение работы выхода HSO.5	HSO.5 мультиплексируется с выводом HSI.3. Этот бит разрешает работу HSO.5. «1» – разрешает выход; «0» – запрещает выход
7	HSI_INT	Выбор источника прерывания HSI	Этот бит определяет источник прерывания HSI Data Available (INT02, 2004H). Должен быть установлен для разрешения INT_MASK.2. «1» – очередь HSI FIFO заполнена; «0» – регистр захвата HSI загружен

IOC2 Регистр 2 управления вводом-выводом

0BH

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Регистр IOC2 управляет тремя параметрами таймера 2, предварительным делением частоты для PWM и АЦП и источником времени для аналого-цифрового преобразования. IOC2 также разрешает или запрещает захват команды в HSO CAM и может очистить всё содержимое HSO CAM.

IOC2
регистр 2 управления вводом/выводом

$0B_H$ значение после Сброса: $X0000000_B$

(горизонтальное окно 0 - запись, горизонтальное окно 15 - чтение)

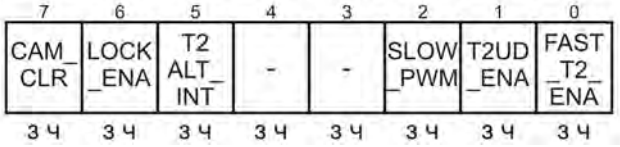


Рисунок В.64 – Формат регистра IOC2

Таблица В.73 – Функциональные назначения полей регистра IOC2

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	FAST_T2_ENA	Разрешение быстрого инкремента таймера 2	Этот бит определяет, будет ли таймер 2 работать в нормальном режиме или в режиме быстрого инкремента. «1» – режим быстрого инкремента; считает каждый такт «0» – нормальный режим; считает каждые восемь тактов Когда разрешён режим быстрого инкремента, не стоит использовать таймер 2 в качестве базового при HSO и сбрасывать таймер 2.
1	T2UD_ENA	Разрешение режима прямого/ обратного счёта для таймера 2	Этот бит определяет, будет ли таймер 2 функционировать как счётчик прямого счёта или как счётчик прямого и обратного счёта. «1» – если P2.6 = 0 – прямой счёт; если P2.6 = 1 – обратный счёт; «0» – только прямой счёт
2	SLOW_PWM	Разрешение предварительного делителя частоты синхронизации для управления выходом PWM	Этот бит управляет периодом на выходе PWM путём разрешения или запрещения предварительного делителя частоты синхронизации (деление на два) для PWM.1, PWM.2 и PWM.3. «1» – разрешение, период на выходе PWM = 512 машинных тактов. «0» – запрещение, период на выходе PWM = 256 машинных тактов

Окончание таблицы В.73

1	2	3	4
3	–	–	Зарезервировано
4	–	–	Зарезервировано
5	T2ALT_INT	Выбор границы переполнения таймера 2	Этот бит выбирает границу переполнения для прерывания по переполнению таймера 2 (INT12, 2038H). равен 1 для разрешения прерывания. «1» – граница 7FFFH/8000H. «0» – граница 0FFFFH/0000H
6	LOCK_ENA	Разрешение фиксирования команд в CAM файле	Этот бит разрешает и запрещает захват команд. Когда этот бит управляет тем, будут ли отдельные команды фиксироваться в CAM файле или стираться после выполнения. «1» – разрешение фиксации команд. «0» – запрещение фиксации команд. Запись 1 в IOC2.7 очищает содержимое CAM файла во всех случаях так же, как и при сбросе кристалла
7	CAM_CLR	Очистка всего содержимого CAM файла	Установка этого бита очищает всё содержимое (даже фиксированные команды) из HSO CAM файла. Этот бит не запоминается; он всегда читается как 1 в горизонтальном окне 15

IOC3 Регистр 3 управления вводом-выводом

0CH

Горизонтальное окно 1 (чтение/запись)

Регистр IOC.3 выбирает внутренний или внешний источник синхронизации для таймера 2 и выбирает функции входов P1.2 и P1.3.

IOC3
регистр 3 управления вводом/выводом

0CH значение после Сброса: F0H

(горизонтальное окно 1 - запись, чтение)

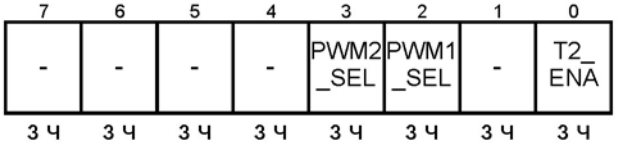


Рисунок В.65 – Формат регистра IOC3

Таблица В.74 – Функциональные назначения полей регистра IOC3

Номер бита	Мнемоника	Имя бита	Описание
0	T2_ENA	Разрешение внутреннего источника синхронизации таймера 2	Этот бит управляет, будет ли таймер 2 синхронизироваться внутренне или внешне. «1» – внутренний источник; «0» – внешний источник; если IOC0.7 = 1, то HSI.1; если IOC0.7 = 0, то T2CLK (P2.3). IOC2.0 определяет, будет ли таймер 2 считать каждый такт (быстрый инкрементный режим) или каждые восемь тактов (нормальный режим)
1	Reserved	Зарезервиро-ван	Этот бит зарезервирован, всегда записывают нули
2	PWM1_SEL	Выбор PWM1	Этот бит выбирает функцию P1.3. «1» – вывод PWM1; «0» – вывод квазидвунаправленного порта. Функции этого вывода могут переключаться без сброса устройства
3	PWM2_SEL	Выбор PWM2	Этот бит выбирает функцию P1.4. «1» – вывод PWM2; «0» – вывод квазидвунаправленного порта. Функции этого вывода могут переключаться без сброса устройства
4-7	–	–	Зарезервированы; всегда записывают нули

IOPORT0 Регистр ввода-вывода порта 0

0EH

Горизонтальное окно 0 (чтение)

Порт 0 – только входной порт. Для уменьшения помех значение на входе порта 0 фиксируется, только когда происходит чтение. Выводы могут использоваться для аналогового ввода в АЦП (ACHx) и цифрового ввода (P0.x) одновременно, но это не рекомендуется. Вывод P0.7 может также использоваться как вход внешнего прерывания.

IOPORT0
регистр ввода/вывода порта 0

0EH значение после Сброса: XXH

(горизонтальное окно 0 - чтение)

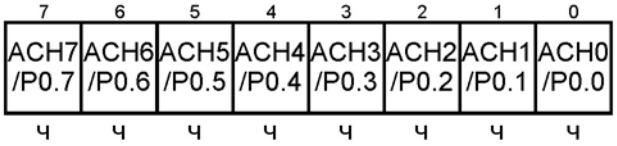


Рисунок В.66 – Формат регистра IOPORT0

Таблица В.75 – Функциональные назначения полей регистра IOPORT0

Номер бита	Мнемоника	Имя бита	Описание
0	ACH0/P0.0	ACH0/ P0.0	АЦП канал 0 / вход P0.0
1	ACH1/P0.1	ACH1/ P0.1	АЦП канал 1 / вход P0.1
2	ACH2/P0.2	ACH2/ P0.2	АЦП канал 2 / вход P0.2
3	ACH3/P0.3	ACH3/ P0.3	АЦП канал 3 / вход P0.3
4	ACH4/P0.4	ACH4/ P0.4	АЦП канал 4 / вход P0.4
5	ACH5/P0.5	ACH5/ P0.5	АЦП канал 5 / вход P0.5
6	ACH6/P0.6	ACH6/ P0.6	АЦП канал 6 / вход P0.6
7	ACH7/P0.7	ACH7/ P0.7	АЦП канал 7 / вход P0.7

IOPORT1 Регистр ввода-вывода порта 1

0FH

Горизонтальное окно 0 (чтение/запись)

Если не выбраны дополнительные функции, то все выводы порта 1 – квазидвунаправленные. Три вывода порта разделяют функции с сигналами захвата шины; два вывода порта разделяют функции совместно с выводами PWM. Когда выводы порта сконфигурированы на ввод-вывод, то они могут быть считаны или записаны. Когда выводы порта настроены на дополнительные функции, то их можно только считать. Для корректного функционирования порта SPI необходимо записать в биты P1.7, P1.6, P1.5 единицы.

Рисунок В.67 – Формат регистра IOPORT1

Таблица В.76 – Функциональные назначения полей регистра IOPORT1

Номер бита	Мнемоника	Имя бита	Описание
0	P1.0	P1.0	Контакт ввода-вывода P1.0
1	P1.1	P1.1	Контакт ввода-вывода P1.1
2	P1.2	P1.2	Контакт ввода-вывода P1.2
3	P1.3/PWM1	P1.3/ PWM1	Контакт ввода-вывода P1.3/ выход PWM1. Установка IOC3.2 разрешает использовать P1.3 в качестве выхода PWM1
4	P1.4/PWM2	P1.4/ PWM2	Контакт ввода-вывода P1.4/ выход PWM2. Установка IOC3.3 разрешает использовать P1.4 в качестве выхода PWM2
5	P1.5	P1.5	Контакт ввода-вывода P1.5
6	P1.6	P1.6	Контакт ввода-вывода P1.6
7	P1.7	P1.7	Контакт ввода-вывода P1.7

IOPORT2 Регистр ввода-вывода порта 2

10H

Горизонтальное окно 0 (чтение/запись)

Порт 2 содержит контакты только для ввода, контакты только для вывода и квазидвунаправленные контакты. Дополнительные функции должны быть разрешены в соответствующих управляющих регистрах.

IOPORT2
регистр ввода/вывода порта 2

10H значение после Сброса: C1H

(горизонтальное окно 0 - чтение, запись)

7	6	5	4	3	2	1	0
P2.7/ T2 CAP	P2.6/ T2UP -DN	P2.5/ PWM 0	P2.4/ T2 RST	P2.3/ T2 CLK	P2.2/ EXT INT	P2.1/ RXD	P2.0/ TXD
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок В.68 – Формат регистра IOPORT2

Таблица В.77 – Функциональные назначения полей регистра IOPORT2

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	P2.0/TXD	P2.0 / TXD	Выходной контакт P2.0. Установка IOC1.5 разрешает использовать контакт как TXD, который служит выходом последовательного порта в режимах 1, 2 и 3 и контактом синхронизации для сдвигового регистра в режиме 0
1	P2.1/RXD	P2.1 / RXD	Входной контакт P2.1 разрешает использовать контакт как RXD. В режимах 1, 2 и 3 RXD используется для приёма данных в последовательный порт. В режиме 0 он служит для ввода или вывода данных (во время вывода контакт работает с открытым стоком)
2	P2.2/EXTINT	P2.2 / EXTINT	Входной контакт P2.2. Очистка IOC1.1 выбирает P2.2 в качестве источника прерывания EXTINT (INT07, 200EH). P2.2 может генерировать прерывание или EXTINT (INT07) или EXTINT1 (INT13, 203AH)
3	P2.3/T2CLK	P2.3 / T2CLK	Входной контакт P2.3. Очистка IOC0.7 разрешает использовать контакт в качестве внешней входной синхронизации для таймера 2
4	P2.4/T2RST	P2.4/ T2RST	Входной контакт P2.4. Очистка IOC2.5 разрешает использовать контакт для внешнего сброса таймера 2. IOC0.3 должен быть установлен для разрешения внешнего сброса

Окончание таблицы В.77

1	2	3	4
5	P2.5/PWM0	P2.5/ PWM0	Выходной контакт P2.5. Установка IOC1.0 разрешает использовать этот контакт в качестве выхода PWM0
6	P2.6/T2UP-DN	P2.6/ T2UP-DN	Квазидвунаправленный контакт P2.6. Установка IOC2.1 разрешает использовать контакт для непосредственного управления направлением счета таймера 2. Таймер 2 считает на увеличение, когда на контакте «0», и на уменьшение, когда – «1»
7	P2.7/T2CAP	P2.7/ T2CAP	Квазидвунаправленный контакт P2.7. Нарастающий фронт на контакте P2.7 записывает значение таймера 2 в регистр T2CAPTURE и генерирует прерывание Timer 2 Capture (INT11, 2036H)

IOPORT3/4 Регистр ввода-вывода портов 3 и 4

1FFEH

Порты 3 и 4 – двунаправленные с открытым стоком. Дополнительная функция данных портов – мультиплексированная шина адрес/данные. Порты автоматически выполняют функции системной шины, когда на EA# низкий уровень, и функционируют как порт с открытым стоком, когда на EA# высокий уровень. Порты 3 и 4 могут быть считаны или записаны только как слово по адресу 1FFEH.

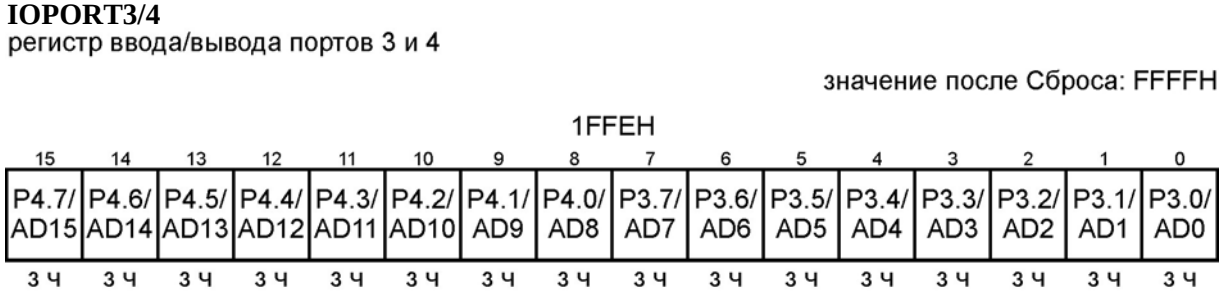


Рисунок В.69 – Формат регистра IOPORT3/4

Таблица В.78 – Функциональные назначения полей регистра IOPORT3/4

Номер бита	Мнемоника	Имя бита	Описание
0-7	P3.0 – P3.7/ AD0 – AD7	P3.0 – P3.7/ AD0 – AD7	Двунаправленный порт ввода-вывода P3.0 – P3.7, когда EA# = 1; в остальное время: AD0 – AD7
8-15	P4.0 – P4.7/ AD8 – AD15	P4.0 – P4.7/ AD8 – AD15	Двунаправленный порт ввода-вывода P4.0 – P4.7, когда EA# = 1; в остальное время: AD8 – AD15

IOS0 Регистр 0 состояния ввода-вывода

15H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Регистр IOS0 показывает текущее состояние контактов HSO. Запись в соответствующие биты (IOS0.0 – IOS0.5) в HWindow 15 может устанавливать или сбрасывать контакты HSO. IOS0.6 и IOS0.7 показывают текущее состояние HSO CAM файла и фиксирующего регистра HSO. (IOS0.6 и IOS0.7 нельзя записывать).

Рисунок В.70 – Формат регистра IOS0

Таблица В.79 – Функциональные назначения полей регистра IOS0

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	HSO.0_STAT	Состояние HSO.0	Текущее состояние контакта HSO.0
1	HSO.1_STAT	Состояние HSO.1	Текущее состояние контакта HSO.1
2	HSO.2_STAT	Состояние HSO.2	Текущее состояние контакта HSO.2
3	HSO.3_STAT	Состояние HSO.3	Текущее состояние контакта HSO.3
4	HSO.4_STAT	Состояние HSO.4	Текущее состояние контакта HSO.4. HSO.4 – вывод двунаправленного порта, мультиплексированный с контактом HSI.2. Установка IOC1.4 разрешает работу выхода HSO.4
5	HSO.5_STAT	Состояние HSO.5	Текущее состояние канала HSO.5. HSO.5 – вывод двунаправленного порта, мультиплексированный с контактом HSI.3. Установка IOC1.6 разрешает работу выхода HSO.5
6	HRCAM_STAT	Состояние HSO CAM и фиксирующего регистра	Текущее состояние фиксирующего регистра HSO и CAM. «0» – регистр захвата HSO пуст и по крайней мере одно слово CAM пусто; «1» – регистр захвата HSO загружен. Во избежание перезаписи текущего значения не следует записывать в регистр захвата, пока не сбросится этот бит или бит IOS0.7
7	HR_STAT	Состояние фиксирующего регистра HSO	Текущее состояние фиксирующего регистра HSO. «0» – регистр захвата HSO пуст; «1» – регистр захвата HSO загружен. Во избежание перезаписи текущего значения не записывайте в регистр захвата, пока не сбросится этот бит или бит IOS0. 6

IOS1 Регистр 1 состояния ввода-вывода

16H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Регистр IOS1 содержит флаги, которые показывают, какие события установили запрос на прерывание. IOS1.0 – IOS1.5 показывают состояния программных таймеров: таймера 1 и таймера 2. Чтение IOS1 сбрасывает биты IOS1.0 – IOS1.5. Запись в IOS1.0 – IOS1.5 изменяет эти биты, но не вызывает прерывания.

IOS1.6 и IOS1.7 показывают состояние очереди HSI FIFO и буферного регистра HSI. В IOS1.6 и IOS1.7 нельзя записывать.

IOS1
регистр 1 состояния ввода/вывода

16H значение после Сброса: 00H

(горизонтальное окно 0 - чтение, горизонтальное окно 15 - запись)

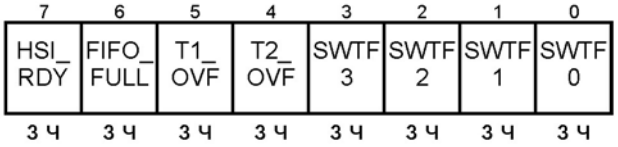


Рисунок В.71 – Формат регистра IOS1

Таблица В.80 – Функциональные назначения полей регистра IOS1

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	SWTF0	Флаг программного таймера 0	Этот бит, когда он установлен, показывает, что программный таймер 0 сработал и вызывает прерывание INT05 (200AH). INT_MASK.5 должен быть установлен для разрешения прерывания
1	SWTF1	Флаг программного таймера1	Этот бит, когда он установлен, показывает, что программный таймер 1 сработал и вызывает прерывание INT05 (200AH). INT_MASK.5 должен быть установлен для разрешения прерывания
2	SWTF2	Флаг программного таймера 2	Этот бит, когда он установлен, показывает, что программный таймер 2 сработал и вызывает прерывание INT05 (200AH). INT_MASK.5 должен быть установлен для разрешения прерывания
3	SWTF3	Флаг программного таймера 3	Этот бит, когда он установлен, показывает, что программный таймер 3 сработал и вызывает прерывание INT05 (200AH). INT_MASK.5 должен быть установлен для разрешения прерывания
4	T2_OVF	Флаг переполнения таймера 2	И таймер 1, и таймер 2 могут генерировать прерывание Timer Overflow (INT00, 2000H). Этот бит, когда установлен, показывает, что таймер 2 вызвал прерывание. INT_MASK.0 должен быть установлен для разрешения прерывания

Окончание таблицы B.80

1	2	3	4
5	T1_OVF	Флаг переполнения таймера 1	И таймер 1, и таймер 2 могут генерировать прерывание переполнения таймера (INT00, 2000H). Этот бит, когда установлен, показывает, что таймер 1 вызвал прерывание. INT_MASK.0 должен быть установлен для разрешения прерывания
6	FIFO_FULL	Флаг шестой записи в FIFO	Этот бит, когда установлен, показывает, что буфер HSI FIFO имеет шесть или более записей, без учета буферного регистра. Это событие может генерировать либо прерывание HSI Data Available (INT02, 2004H), либо прерывание HSI FIFO Full (INT14, 203CH), но не оба сразу
7	HSI_RDY	Готовность данных в буферном регистре HSI	Этот бит, когда установлен, показывает, что буферный регистр HSI загружен. Когда IOC1.7 сброшен, это событие генерирует прерывание HSI Data Available (INT02, 2004H). INT_MASK.2 должен быть установлен для разрешения прерывания

IOS2 Регистр 2 состояния ввода-вывода

17H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Регистр IOS2 содержит флаги, которые показывают, какие события HSO имели место. Запись в IOS2 устанавливает или сбрасывает биты состояния, но не вызывает прерывание. Чтение IOS2 очищает все биты.

IOS2
регистр 2 состояния ввода/вывода

17H значение после Сброса: 00H

(горизонтальное окно 0 - чтение, горизонтальное окно 15 - запись)

7	6	5	4	3	2	1	0
AD_EVENT	T2RST_EVENT	HSO5_EVENT	HSO4_EVENT	HSO3_EVENT	HSO2_EVENT	HSO1_EVENT	HSO0_EVENT
З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч	З Ч

Рисунок В.72 – Формат регистра IOS2

Таблица В.81– Функциональные назначения полей регистра IOS2

Номер бита	Мнемоника	Имя бита	Описание
0	HSO0_EVENT	Событие на выводе HSO.0	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.0
1	HSO1_EVENT	Событие на выводе HSO.1	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.1
2	HSO2_EVENT	Событие на выводе HSO.2	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.2
3	HSO3_EVENT	Событие на выводе HSO.3	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.3
4	HSO4_EVENT	Событие на выводе HSO.4	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.4
5	HSO5_EVENT	Событие на выводе HSO.5	Этот бит, когда он установлен, показывает, что команда HSO выполнена на контакте HSO.5
6	T2RST_EVENT	Событие сброс таймера 2	Этот бит, когда он установлен, показывает, что команда HSO сбросила таймер 2.
7	AD_EVENT	Событие начала А/Ц преобразования	Этот бит, когда он установлен, показывает, что команда HSO запустила АЦ преобразование

PPW Регистр задания длительности программирующего импульса EEPROM

1FF3H/1FF2H (чтение/запись)

Регистр определяет длительность программирующего импульса EEPROM. Эта длительность не должна быть менее 4,5 мс. Программирующий импульс заданной длительности используется для записи слова, очистки блока, очистки строки. Длительность импульса рассчитывается по формуле

$$T_{PPW} = (PPW \times 16) \times Тмц,$$

где PPW – значение регистра в десятичном виде;

Тмц – длительность машинного цикла.

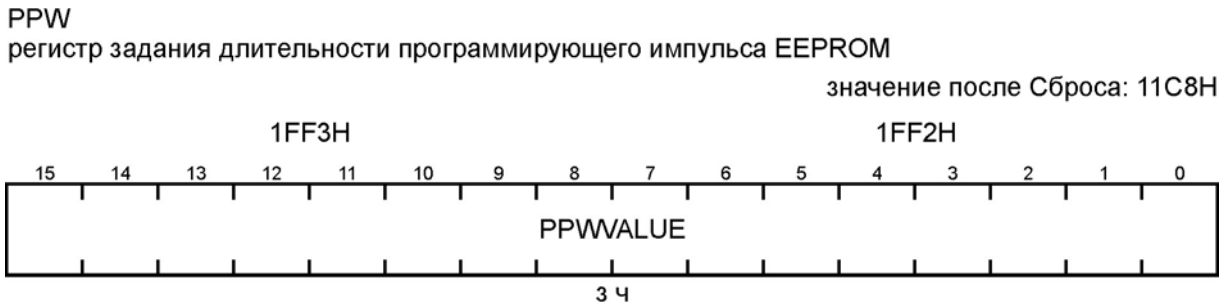


Рисунок В.73 – Формат регистра PPW

Таблица В.82 – Функциональные назначения полей регистра PPW

Номер бита	Мнемоника	Имя бита	Описание
0-15	PPWVALUE	Длительность программирующего импульса	Содержит значение длительности программирующего импульса EEPROM

PPSETUP Регистр задания времени установления программирующего импульса EEPROM

1FF4H (чтение/запись)

Регистр определяет время установления программирующего импульса EEPROM. Эта длительность не должна быть менее 1 мкс. Время установления рассчитывается по формуле

$$T_{PPSETUP} = PPSETUP \times Tмц,$$

где PPSETUP – значение регистра в десятичном виде;

Tмц – длительность машинного цикла.

Рисунок В.74 – Формат регистра PPSETUP

Таблица В.83 – Функциональные назначения полей регистра PPSETUP

Номер бита	Мнемоника	Имя бита	Описание
0-7	PPSETUP	Время установления	Содержит значение времени установления для программирующего импульса EEPROM

PPHOLD Регистр задания времени удержания программирующего импульса EEPROM

1FF5H (чтение/запись)

Регистр определяет время удержания программирующего импульса EEPROM. Эта длительность не должна быть менее 1 мкс. Время удержания рассчитывается по формуле

$$T_{PPHOLD} = PPHOLD \times Тмц,$$

где PPHOLD – значение регистра в десятичном виде;

Тмц – длительность машинного цикла.

Рисунок В.75 – Формат регистра PPHOLD

Таблица В.84 – Функциональные назначения полей регистра PPHOLD

Номер бита	Мнемоника	Имя бита	Описание
0-7	PPHOLD	Время удержания	Содержит значение времени удержания для программирующего импульса EEPROM

PRNGCON Регистр управления генератором ПСП

1F64H (чтение/запись)

Регистр управляет работой генератора ПСП: включает генерацию и определяет количество сдвигов. Для старта генератора необходимо установить бит RUN, после окончания операции бит автоматически сброситься.

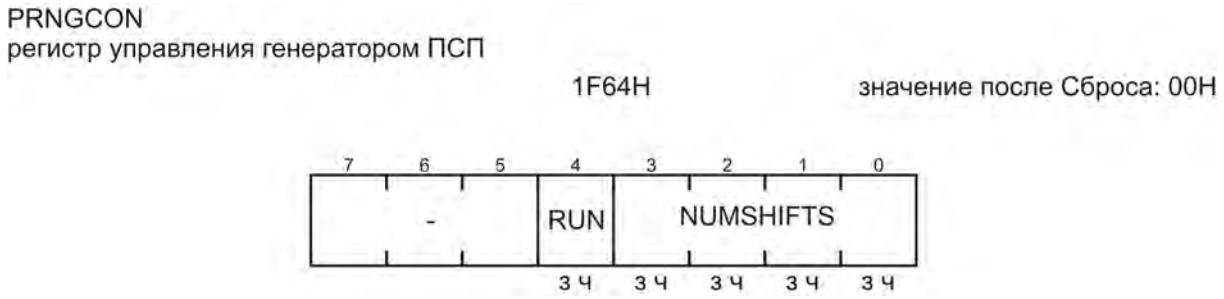


Рисунок В.76 – Формат регистра PRNGCON

Таблица В.85– Функциональные назначения полей регистра PRNGCON

Номер бита	Мнемоника	Имя бита	Описание
0-3	NUMSHIFTS	Количество сдвигов	Содержит количество сдвигов, осуществляемое при генерации ПСП: 0000 – 1 сдвиг …… 1111 – 16 сдвигов
4	RUN	Бит запуска	Запускает генерацию: «1» – запустить генерацию/ генерация продолжается «0» – генерация не идет
5-7	–	–	Зарезервировано

PRNGREG1 Регистр ключа 0

1F61H/1F60H (чтение/запись)
Содержит ключ «0», который используется для генерации ПСП.

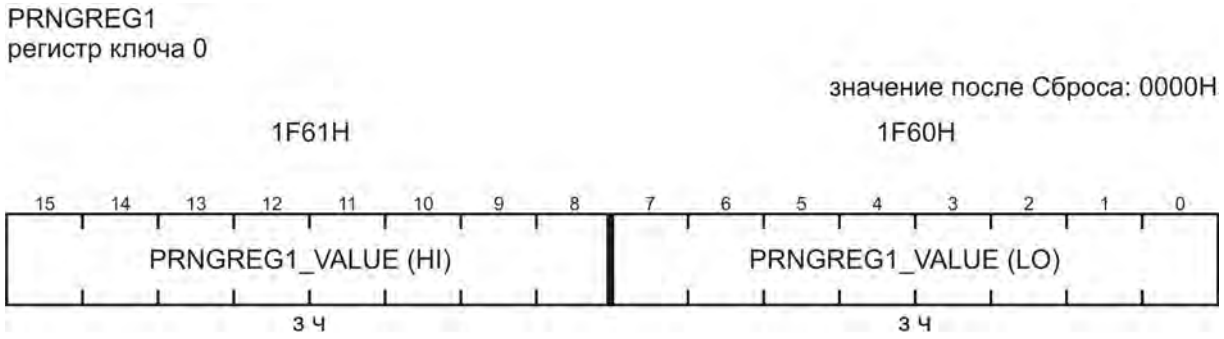


Рисунок В.77 – Формат регистра PRNREG1

Таблица В.86 – Функциональные назначения полей регистра PRNGREG1

Номер бита	Мнемоника	Имя бита	Описание
0-3	PRNGREG1_VALUE (LO)	Младший байт ключа «0»	Содержит младший байт ключа «0»
4-7	PRNGREG1_VALUE (HI)	Старший байт ключа «0»	Содержит старший байт ключа «0»

PRNGREG2 Регистр ключа 1

1F63H/1F62H (чтение/запись)
Содержит ключ «1», который используется для генерации ПСП.

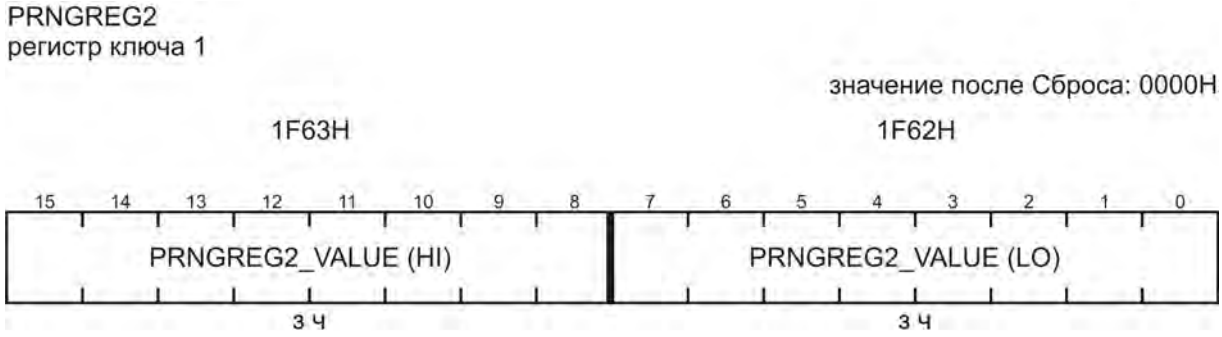


Рисунок В.78 – Формат регистра PRNREG2

Таблица В.87 – Функциональные назначения полей регистра PRNGREG2

Номер бита	Мнемоника	Имя бита	Описание
0-3	PRNGREG2_VALUE (LO)	Младший байт ключа «1»	Содержит младший байт ключа «1»
4-7	PRNGREG2_VALUE (HI)	Старший байт ключа «1»	Содержит старший байт ключа «1»

PRNGRES Регистр результата ПСП

1F67H/1F66H (чтение/запись)
Содержит результат генерации ПСП.

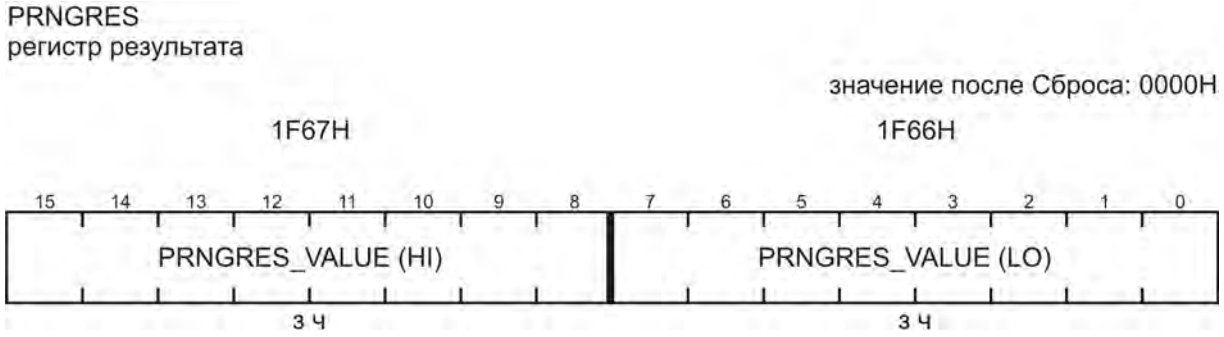


Рисунок В.79 – Формат регистра PRNRES

Таблица В.88 – Функциональные назначения полей регистра PRNGRES

Номер бита	Мнемоника	Имя бита	Описание
0-3	PRNGRES_VALUE (LO)	Младший байт результата	Содержит младший байт результата генерации ПСП
4-7	PRNGRES_VALUE (HI)	Старший байт результата	Содержит старший байт результата генерации ПСП

PSW Слово состояния программы

Нет прямого доступа к регистру.

PSW в действительности состоит из двух байтов. Старший байт – слово состояния, которое описано здесь; младший байт – регистр INT_MASK. Слово состояния содержит один бит PSW.1, который разрешает или запрещает обработку всех маскируемых прерываний, один бит PSW.2, который разрешает или запрещает программный сервер обмена PTS, и шесть двоичных флагов, которые отражают состояние программы пользователя.

Слово состояния PSW прямо не доступно. Для доступа к слову состояния следует записать его значение в стек (PUSHF), а затем считать это значение в регистр (POP test_reg). Команды PUSHF и PUSHA сохраняют PSW в системном стеке, POPF и POPA: восстанавливают его.

Команды EI и DI устанавливают и сбрасывают PSW.1; команды EPTS и DPTS устанавливают и сбрасывают PSW.2. Различные команды тестируют, устанавливают и сбрасывают двоичные флаги.

В приложении А приведена таблица, которая показывает воздействие команд на флаги PSW, и таблица, которая показывает воздействие флагов PSW на команды условного перехода.

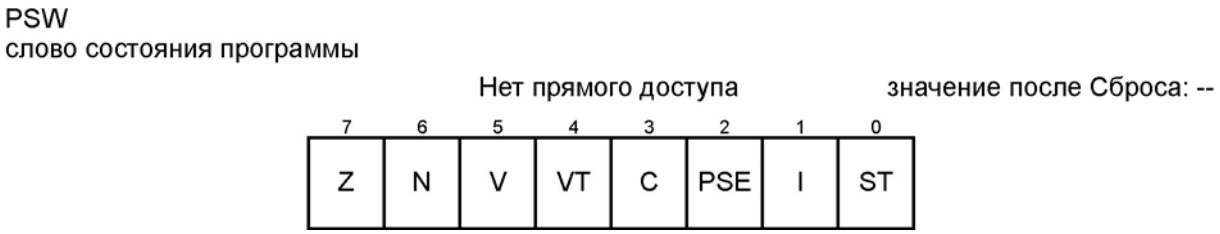


Рисунок В.80 – Формат регистра PSW

Таблица В.89 – Функциональные назначения полей регистра PSW

Номер бита	Мнемо-ника	Имя бита	Описание
1	2	3	4
0	ST	Флаг Sticky	Этот флаг устанавливается для того, чтобы показать, что во время сдвига вправо, «1» была сдвинута во флаг переноса и затем сдвинута из него. Этот бит не определён после операции умножения. Флаг ST может использоваться вместе с флагом переноса для более точного округления. (Подробности – в описании флага переноса)
1	I	Запрещение прерываний (глобальное)	Этот бит разрешает или запрещает обслуживание всех маскируемых прерываний (это все прерывания, кроме NMI, TRAP и неподдерживаемого кода). MASK1 индивидуально разрешают или запрещают прерывания. Команда EI устанавливает этот бит; DI – очищает его. «1» – разрешение обработки прерываний, «0» – запрещение обработки прерываний
2	PSE	Разрешение PTS	Этот бит глобально разрешает или запрещает периферийный сервер обмена PTS. Команда EPTS устанавливает этот бит; DPTS – сбрасывает его. «1» – разрешение PTS; «0» – запрещение PTS

Окончание таблицы В.89

1	2	3	4
3	C	Флаг переноса	Этот флаг устанавливается, чтобы показывать состояние арифметического переноса из старшего значащего бита ALU или состояние последнего бита, сдвинутого из операнда. Если вычитание содержит заём, флаг переноса устанавливается. C Значение сдвинутых битов 0 < 1/2 LSB 1 ≥ 1/2 LSB Обычно результат округляется до большего значения при установленном флаге переноса. Флаг Sticky бита позволяет улучшить точность округления. C ST Значение сдвинутых битов 0 0 = 0 0 1 > 0 и < 1/2 LSB 1 0 = 1/2 LSB 1 1 > 1/2 LSB и < 1 LSB
4	VT	Дополнитель-ный флаг переполнения	Этот флаг устанавливается, когда флаг переноса установлен, но он очищается только командами CLRVT, JVT, JNVT. Это позволяет тестировать возможное состояние переполнения в конце последовательности родственных арифметических операций, которое обычно более эффективно, чем тестирование флага переполнения после любой операции
5	V	Флаг переполнения	Этот флаг устанавливается, если после выполнения операции получен результат, выходящий из 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 в следующих случаях: Команда Результат DIVUB > 155 (0FFH) DIVU > 65535 (0FFFFH) DIVB < -127 (81H) или > +127 (7FH) DIV < -32767 (8001H) или > +32767 (7FFFH)
6	N	Флаг отрицательного результата	Этот флаг устанавливается, если после выполнения операции получен отрицательный результат. Флаг правильный, если даже случилось переполнение. При всех сдвиговых операциях и операции NORML этот флаг устанавливается равным старшему значащему биту результата, даже если счётчик сдвигов равен нулю
7	Z	Флаг нуля	Этот флаг устанавливается, если результат операции равен нулю. При операциях сложения с переносом и вычитания с заёмом 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

PTSBLOCK Регистр блока PTS

Смещение 7 в управляющем блоке PTS

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

PTSBLOCK
регистр блока PTS

значение после Сброса: XXH

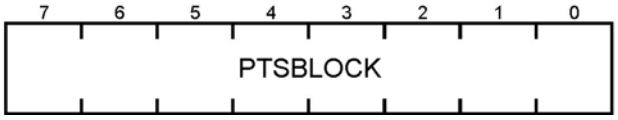


Рисунок В.81 – Формат регистра PTSBLOCK

Таблица В.90 – Функциональное назначение полей регистра PTSBLOCK

Номер бита	Мнемоника	Имя бита	Описание
0-7	PTSBLOCK	Блочный счётчик PTS	Управление числом передач, которые будут иметь место. Значение зависит от режима PTS. Режим: Значение: Передача блоков 1–32 HSI 1–7 HSO 1–8 Во всех режимах запись нуля в этот регистр ведёт к максимальному числу передач. (32 – для режима блочной передачи; 7 – для режима HSI; 8 – для режима HSO)

PTSCON Регистр управления PTS

Смещение 1 в управляющем блоке PTS

Три бита регистра PTSCON определяют режим PTS: одиночная передача, блочная передача, HSO или HSI. Режим PTS определяется пятью битами. PTSCON имеет одну конфигурацию для режимов одиночной и блочной передач и другую – для режимов HSO и HSI. Конфигурации описываются отдельно. (Бит PSW.2, управляемый командами DPTS и EPTS, разрешает или запрещает PTS).

PTSCON
регистр управления PTS (режимы одиночной и блочной передачи)

значение после Сброса: 00H

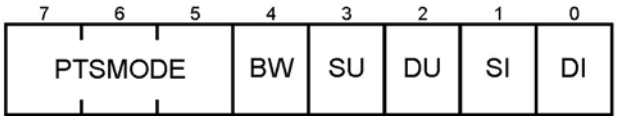


Рисунок В.82 – Формат регистра PTSCON (режимы одиночной и блочной передачи)

Таблица В.91 – Функциональные назначения полей регистра PTSCON (режимы одиночной и блочной передачи)

Номер бита	Мнемоника	Имя бита	Описание
0	DI	Автоинкремент PTSDST	Установка этого бита ведёт к инкременту регистра – приёмника обмена в конце каждого цикла обмена
1	SI	Автоинкремент PTSSRC	Установка этого бита ведёт к инкременту регистра – источника обмена в конце каждого цикла обмена
2	DU	Обновление PTSDST	Установка этого бита ведёт к сохранению регистром PTSDST его окончательного значения в конце PTS цикла. Его очистка ведёт к возвращению регистру значения, которое существовало в начале PTS цикла
3	SU	Обновление PTSSRC	Установка этого бита ведёт к сохранению регистром PTSSRC его окончательного значения в конце PTS цикла. Его очистка ведёт к возвращению регистру значения, которое существовало в начале PTS цикла
4	BW	Передача байта/ слова	Установкой этого бита выбирается байтовая передача. Очисткой выбирается передача слова
5-7	PTSMODE	Режим обмена	Эти три бита определяют режим обмена: Бит 7 6 5 0 0 0 одиночная передача 1 0 0 блочная передача

PTSCON
регистр управления PTS (режимы HSO и HSI)

значение после Сброса: 02_H

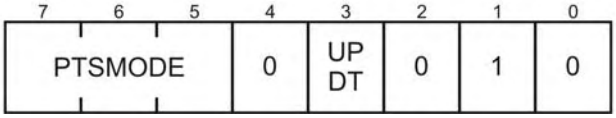


Рисунок В.83 – Формат регистра PTSCON (режимы HSO, HSI)

Таблица В.92 – Функциональные назначения полей регистра PTSCON (режимы HSO, HSI)

Номер бита	Мнемоника	Имя бита	Описание
0	–	–	Всегда 0
1	–	–	Всегда 1
2	–	–	Всегда 0
3	UPDT	Обновление регистра	Загрузка этого бита ведёт к загрузке соответствующего регистра значением, которое существует в конце обмена. Его очистка ведёт к загрузке этого регистра значением, существовавшим в начале цикла обмена. Режим Регистр HSI PTSDST HSO PTSSRC
4	–	–	Всегда 0.
5-7	PTSMODE	Режим обмена PTS	Эти биты определяют режим работы PTS: Бит 7 6 5 0 1 1 HSO 0 0 1 HSI

PTSCOUNT Счетный регистр PTS

Смещение 0 в управляющем блоке PTS.

Регистр PTSCOUNT используется во всех режимах обмена. PTSCOUNT определяет количество циклов обмена, которые будут последовательно осуществляться без вмешательства CPU. Так как PTSCOUNT – 8-битная величина, то максимальное число циклов – 256. PTSCOUNT декрементируется в конце каждого цикла обмена. Когда PTSCOUNT достигает нуля, аппаратное обеспечение сбрасывает соответствующий бит PTSSEL и устанавливает бит PTSSRV, который запрашивает прерывание end-of-PTS. Когда это прерывание вызвано, аппаратное обеспечение сбрасывает бит PTSSRV. Бит PTSSEL может устанавливаться вручную для восстановления канала обмена.

PTSCOUNT
счетный регистр PTS

значение после Сброса: XXH

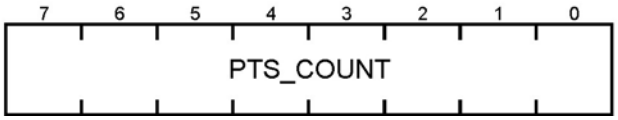


Рисунок В.84 – Формат регистра PTSCOUNT

Таблица В.93 – Функциональные назначения полей регистра PTSCOUNT

Номер бита	Мнемоника	Имя бита	Описание
0-7	PTS_COUNT	Последовательные циклы обмена	Определяют количество циклов обмена, которые будут последовательно осуществляться без вмешательства CPU. Максимальное значение равно 256

PTSDST Регистр приёмника PTS

Смещение 4 в управляющем блоке PTS.

Регистр PTSDST используется в одиночной и блочной передачах и режиме HSI. Регистр PTSDST указывает на ячейки памяти приёмника. PTSDST не обязательно инкрементируется в конце цикла обмена. В режиме одиночной передачи PTSCON.0 и PTSCON.2 определяют, будет ли PTSDST инкрементироваться. В режиме блочной передачи PTSCON.0 определяет, будет ли PTSDST инкрементироваться после любой передачи, а PTSCON.2 определяет, будет ли PTSDST сохранять своё последнее значение или восстанавливать исходное. В режиме HSI PTSCON.3 определяет, будет ли PTSDST обновляться в конце цикла обмена.

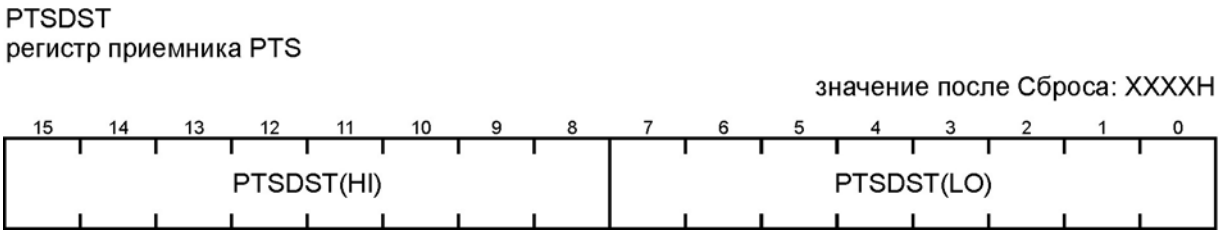


Рисунок В.85 – Формат регистра PTSDST

Таблица В.94 – Функциональные назначения полей регистра PTSDST

Номер бита	Мнемоника	Имя бита	Описание
0-7	PTSDST(LO)	Адрес приёмника PTS, младший байт	Младший байт адреса приёмника PTS
8-15	PTSDST(HI)	Адрес приёмника PTS, старший байт	Старший байт адреса приёмника PTS

PTSSEL Регистр выбора PTS (HI/LO)

05/04H

Горизонтальное окно 1 (чтение/запись)

Регистр PTSSEL состоит из двух байтов. PTSSEL выбирает – цикл PTS или обычная подпрограмма будет обрабатывать прерывание для каждого из 15 запрашиваемых прерываний. Установка бита выбирает цикл PTS; очистка бита выбирает обычную подпрограмму обработки прерываний.

Когда PTSCOUNT достигает нуля, аппаратное обеспечение сбрасывает соответствующий бит PTSSEL и устанавливает бит PTSSRV, который запрашивает прерывание end-of-PTS. Когда это прерывание будет обработано, аппаратное обеспечение сбросит бит PTSSRV. Для восстановления канала обмена бит PTSSEL должен устанавливаться вручную.

Любое прерывание может маскироваться соответствующим битом в регистрах INT_MASK и INT_MASK1. PTS разрешается или запрещается битом PSW.2.

PTSSEL
регистр выбора PTS (HI/LO)

значение после Сброса: 0000H

05H 04H

(горизонтальное окно 1 - чтение, запись)

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	RI _SEL	TI _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

Рисунок В.86 – Формат регистра PTSSEL

Таблица В.95 – Функциональные назначения полей регистра PTSSEL

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	T1OVF_SEL	Выбор прерывания Timer Overflow	Установка этого бита ведёт к тому, что прерывание Timer Overflow (INT00) будет обрабатываться циклом PTS
1	AD_SEL	Выбор прерывания A/D Conversion	Установка этого бита ведёт к тому, что прерывание A/D Conversion (INT01) будет обрабатываться циклом PTS
2	HSIDAT_SEL	Выбор прерывания HSI Data Available/ FIFO Full	Установка этого бита ведёт к тому, что прерывание HSI Data Available (INT02) будет обрабатываться циклом PTS
3	HSO_SEL	Выбор прерывания Event HSO	Установка этого бита ведёт к тому, что прерывание Event HSO (INT03) будет обрабатываться циклом PTS
4	HSI0_SEL	Выбор прерывания внешний HSI. 0	Установка этого бита ведёт к тому, что внешнее прерывание HSI.0 (INT04) будет обрабатываться циклом PTS

Окончание таблицы B.95

1	2	3	4
5	SWT_SEL	Выбор прерывания Software Timer	Установка этого бита ведёт к тому, что прерывание Software Timer (INT05) будет обрабатываться циклом PTS
6	SER_SEL	Выбор прерывания Serial Ports	Установка этого бита ведёт к тому, что прерывание Serial Ports (INT06) будет обрабатываться циклом PTS
7	EXTINT_SEL	Выбор прерывания от контакта EXTINT или P0.7	Установка этого бита ведёт к тому, что прерывание от контакта EXTINT или P0.7 (INT07) будет обрабатываться циклом PTS
8	TI_SEL	Выбор прерывания Transmit UART	Установка этого бита ведёт к тому, что прерывание Transmit (INT08) будет обрабатываться циклом PTS
9	RI_SEL	Выбор прерывания Receive UART	Установка этого бита ведёт к тому, что прерывание Receive (INT09) будет обрабатываться циклом PTS
10	HSI4_SEL	Выбор прерывания HSI FIFO 4	Установка этого бита ведёт к тому, что прерывание HSI FIFO 4 (INT10) будет обрабатываться циклом PTS
11	T2CAP_SEL	Выбор прерывания Timer 2 Capture	Установка этого бита ведёт к тому, что прерывание Timer 2 Capture (INT11) будет обрабатываться циклом PTS
12	T2OVF_SEL	Выбор прерывания Timer 2 Overflow	Установка этого бита ведёт к тому, что прерывание Timer 2 Overflow (INT12) будет обрабатываться циклом PTS
13	EXTINT1_SEL	Выбор прерывания от контакта EXTINT1	Установка этого бита ведёт к тому, что прерывание контакта EXTINT1 (INT13) будет обрабатываться циклом PTS
14	FIFO_SEL	Выбор прерывания HSI FIFO Full	Установка этого бита ведёт к тому, что прерывание HSI FIFO Full (INT14) будет обрабатываться циклом PTS
15	–	–	Зарезервировано; всегда записывают нуль

PTSSRC Регистр источника PTS

Смещение 2 в управляющем блоке PTS

Регистр PTSSRC используется в режимах одиночной передачи, блочной передачи и HSO. Регистр PTSSRC указывает на ячейки памяти источника. PTSSRC не обязательно инкрементируется в конце цикла обмена. В режиме одиночной передачи PTSCON.1 и PTSCON.3 определяют, будет ли PTSSRC инкрементироваться. В режиме блочной передачи PTSCON.1 определяет, будет ли PTSSRC инкрементироваться после каждой передачи, а PTSCON.3 определяет, будет PTSSRC сохранять своё последнее значение или восстанавливать своё первоначальное значение. В режиме HSO PTSCON.3 определяет, будет ли PTSSRC обновляться в конце цикла обмена.

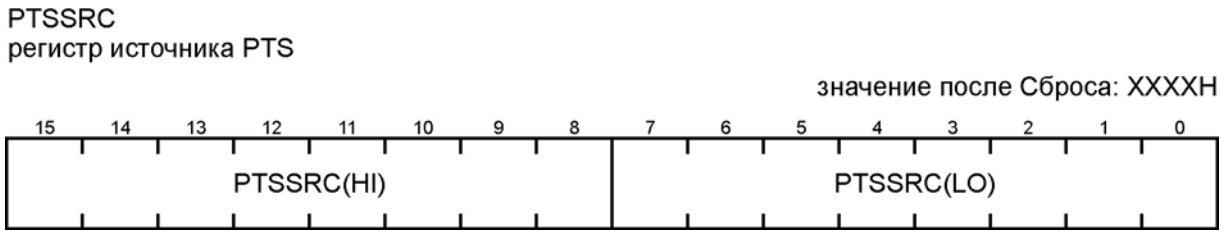


Рисунок В.87 – Формат регистра PTSSRC

Таблица В.96 – Функциональные назначения полей регистра PTSSRC

Номер бита	Мнемоника	Имя бита	Описание
0-7	PTSSRC(LO)	Младший байт адреса источника обмена	Младший байт адреса источника обмена
8-15	PTSSRC(HI)	Старший байт адреса источника обмена	Старший байт адреса источника обмена

PTSSRV Регистр обслуживания PTS

07/06H Горизонтальное окно 1 (чтение/запись)

Регистр PTSSRV состоит из двух байтов. PTSSRV используется аппаратным обеспечением для указания на то, что последний цикл обмена обслужен. Когда PTSCOUNT достигает нуля, аппаратное обеспечение сбрасывает соответствующий бит PTSSEL и устанавливает бит PTSSRV, который запрашивает прерывание end-of-PTS. Когда это прерывание вызвано, аппаратное обеспечение сбрасывает бит PTSSRV. Для восстановления канала обмена бит PTSSEL должен устанавливаться вручную.

PTS разрешается или запрещается битом PSW.2, который устанавливается командой EPTS и сбрасывается командой DPTS. Прерывания разрешаются или запрещаются (маскируются) соответствующими битами в регистрах INT_MASK или INT_MASK1. Обработка отдельных прерываний циклами обмена разрешается соответствующими битами в регистре PTSSEL.

Векторы прерывания end-of-PTS инициализируются через те же ячейки памяти, что и соответствующие векторы обычных прерываний. Например, если PTSSEL.8 установлен, прерывание передачи (TI) обрабатывается вектором PTS по адресу 2050H, с вектором end-of-PTS по адресу 2030H. Расположение векторов прерываний смотрите в таблице B.4.

PTSSRV
регистр обслуживания PTS

значение после Сброса: 0000H

07H								06H							
(горизонтальное окно 1 - чтение, запись)															
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	RI _SRV	TI _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

Рисунок B.88 – Формат регистра PTSSRV

Таблица B.97– Функциональные назначения полей регистра PTSSRV

Номер бита	Мнемоника	Имя бита	Описание
1	2	3	4
0	T1OVF_SRV	Обслуживание прерывания Timer 1 Overflow	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Timer 1 Overflow достигает нуля, что вызывает прерывание end-of-PTS с вектором 2000H
1	AD_SRV	Обслуживание прерывания A/D Conversion	Этот бит устанавливается, когда PTSCOUNT для цикла PTS A/D Conversion достигает нуля, что вызывает прерывание end-of-PTS с вектором 2002H
2	HSIDAT_SRV	Обслуживание прерывания HSI Data Available/ FIFO Full	Этот бит устанавливается, когда PTSCOUNT для цикла PTS HSI Data Available достигает нуля, что вызывает прерывание end-of-PTS с вектором 2004H
3	HSO_SRV	Обслуживание прерывания Event HSO	Этот бит устанавливается, когда PTSCOUNT для цикла PTS HSO достигает нуля, что вызывает прерывание end-of-PTS с вектором 2006H

Окончание таблицы В.97

1	2	3	4
4	HSI0_SRV	Обслуживание прерывания HSI.0	Этот бит устанавливается, когда PTSCOUNT для цикла PTS HSI.0 достигает нуля, что вызывает прерывание end-of-PTS с вектором 2008H
5	SWT_SRV	Обслуживание прерывания Software Timer	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Software Timer достигает нуля, что вызывает прерывание end-of-PTS с вектором 200AH
6	SER_SRV	Обслуживание прерывания Serial Ports	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Serial Port достигает нуля, что вызывает прерывание end-of-PTS с вектором 200CH
7	EXTINT_SRV	Обслуживание прерывания от контакта EXTINT или P0.7	Этот бит устанавливается, когда PTSCOUNT для цикла PTS EXTINT достигает нуля, что вызывает прерывание end-of-PTS с вектором 200EH
8	TI_SRV	Обслуживание прерывания Transmit UART	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Transmit достигает нуля, что вызывает прерывание end-of-PTS с вектором 2030H
9	RI_SRV	Обслуживание прерывания Receive UART	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Receive достигает нуля, что вызывает прерывание end-of-PTS с вектором 2032H
10	HSI4_SRV	Обслуживание прерывания HSI FIFO 4	Этот бит устанавливается, когда PTSCOUNT для цикла PTS HSI FIFO 4 достигает нуля, что вызывает прерывание end-of-PTS с вектором 2034H
11	T2CAP_SRV	Обслуживание прерывания Timer 2 Capture	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Timer 2 Capture достигает нуля, что вызывает прерывание end-of-PTS с вектором 2036H
12	T2OVF_SRV	Обслуживание прерывания Timer 2 Overflow	Этот бит устанавливается, когда PTSCOUNT для цикла PTS Timer 2 Overflow достигает нуля, что вызывает прерывание end-of-PTS с вектором 2038H
13	EXTPIN_SRV	Обслуживание прерывания от вывода EXTINT1	Этот бит устанавливается, когда PTSCOUNT для цикла PTS EXTINT1 достигает нуля, что вызывает прерывание end-of-PTS с вектором 203AH
14	FIFO_SRV	Обслуживание прерывания HSI FIFO Full	Этот бит устанавливается, когда PTSCOUNT для цикла PTS HSI FIFO Full достигает нуля, что вызывает прерывание end-of-PTS с вектором 203EH
15	–	–	Зарезервировано; всегда записывают нуль

PWM0_CONTROL Регистр управления PWM0

17H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

PWM0 мультиплексирован с P2.5. Для разрешения функции вывода PWM0 бит IOC1.0 должен быть установлен.

8-битный счётчик PWM инкрементируется каждый машинный такт (с запрещённым предварительным делителем PWM) или каждые два машинных такта (с разрешённым предварительным делителем PWM). Когда счётчик равен 0, выход PWM0 имеет высокий уровень. Он остаётся высоким, пока величина счётчика не достигнет величины регистра PWM0_CONTROL, в момент совпадения на выходе появляется низкий уровень. Когда счётчик переполняется, выход опять переключается на высокий уровень. Когда PWM0_CONTROL равен 0, выход всегда остаётся в низком уровне.

Регистр PWM0_CONTROL вместе с IOC2.2 определяет, как долго вывод PWM0 задерживается на высоком уровне во время импульса, эффективно управляя циклом работы. Значение, записываемое в регистр PWM0_CONTROL, может быть от 0 до 255 машинных тактов (с заполнением от 0 % до 99,6 %).

Установка IOC2.2 разрешает предварительный делитель частоты синхронизации на два для PWM; очистка IOC2.2 запрещает его. Когда предварительный делитель частоты разрешён, общая длительность импульса равна 512 машинных тактов, и значение PWM0_ CONTROL умножается на четыре. Когда он запрещён, общая длительность импульса равна 256 машинных тактов и значение PWM0_CONTROL умножается на два.

Рисунок В.89 – Формат регистра PWM0_CONTROL

Таблица В.98 – Функциональные назначения полей регистра PWM0_CONTROL

Номер бита	Мнемоника	Имя бита	Описание
0-7	STATE_TIMES	Длительность высокого уровня на выходе PWM0 в машинных тактах	Эти биты определяют количество машинных тактов, в течение которых PWM0 на выходе удерживает высокий уровень. В этот регистр записывается шестнадцатеричная величина (00H-FFH)

Для определения периода PWM и времени, в течение которого выход будет иметь высокий уровень, используются формулы, сведенные в таблицу В.99.

Таблица В.99 – Формулы для определения периода PWM и времени, в течение которого выход будет иметь высокий уровень

Определяемая величина, мкс	Формула	
	Предварительный делитель частоты синхронизации запрещён (IOC2.2 = 0)	Предварительный делитель частоты синхронизации разрешён (IOC2. 2 = 1)
Период PWM	$\frac{512}{f_{CI}}$	$\frac{1024}{f_{CI}}$
Высокий уровень PWMx	$\frac{PWMx\_CONTROL \times 2}{f_{CI}}$	$\frac{PWMx\_CONTROL \times 4}{f_{CI}}$

Примечание – В приведенных формулах f_{CI} – частота XTAL1 в МГц, а x, равный 0, 1 или 2, – номер PWM.

Например, если f_{CI} = 16 МГц, то период вывода PWM равен 32 мкс.
Если IOC2.2 = 0 и PWM_CONTROL = 8AH (138, десятичное), то выход удерживается в высоком уровне 17,25 мкс (в низком 14, 8 мкс) из всех 32 мкс, что составляет примерно 54 % рабочего цикла. Когда IOC2.2 = 1, то эти же значения приведут к периоду в 64 мкс, при этом высокий уровень будет в течение 34,5 мкс (низкий 29,5 мкс), что также составляет около 54 % рабочего цикла.

PWM1_CONTROL Регистр управления PWM1

16H

Горизонтальное окно 1 (чтение/запись)

PWM1 мультиплексирован с P1.3. Для разрешения функции вывода для PWM1 IOC3.2 должен быть установлен.

Регистр PWM1_CONTROL вместе с IOC2.2 определяет, как долго выход PWM1 будет находиться в высоком уровне. Величина, записываемая в регистр PWM1_CONTROL, может быть от 0 до 255 машинных тактов (0 – 99,6) % рабочего цикла. Для получения дополнительной информации смотрите описание регистра PWM0_CONTROL.

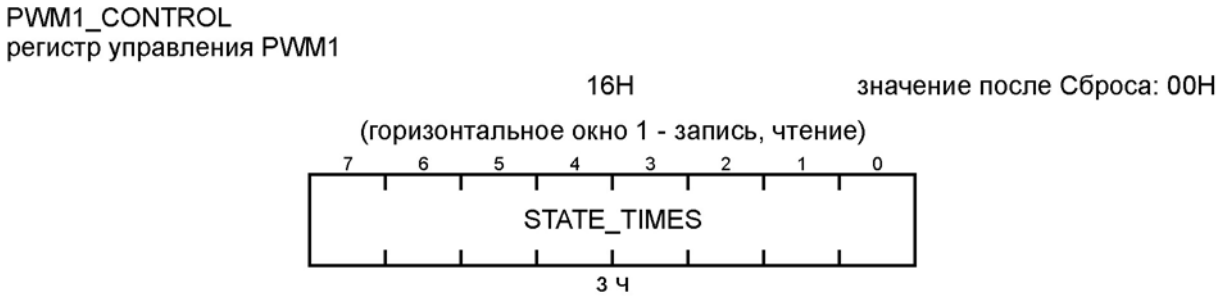


Рисунок В.90 – Формат регистра PWM1_CONTROL

Таблица В.100 – Функциональные назначения полей регистра PWM1_CONTROL

Номер бита	Мнемоника	Имя бита	Описание
0-7	STATE_TIMES	Длительность высокого уровня на выходе PWM1 в машинных тактах	Эти биты определяют количество машинных тактов, в течение которых PWM1 на выходе удерживает высокий уровень. В этот регистр записывается шестнадцатеричная величина (00H-FFH)

PWM2_CONTROL Регистр управления PWM2

17H

Горизонтальное окно 1 (чтение/запись)

PWM2 мультиплексирован с P1.4. Для разрешения функции вывода для PWM2 IOC3.3 должен быть установлен.

Регистр PWM2_CONTROL вместе с IOC2.2 определяет, как долго выход PWM2 будет находиться в высоком уровне. Величина, записываемая в регистр PWM2_CONTROL, может быть от 0 до 255 машинных тактов (от 0 до 99,6)% рабочего цикла. Дополнительная информация дана в описании регистра PWM0_CONTROL.

Рисунок В.91 – Формат регистра PWM2_CONTROL

Таблица В.101 – Функциональные назначения полей регистра PWM2_CONTROL

Номер бита	Мнемоника	Имя бита	Описание
0-7	STATE_TIMES	Длительность высокого уровня на выходе PWM2 в машинных тактах	Эти биты определяют количество машинных тактов, в течении которых PWM2 на выходе удерживает высокий уровень. В этот регистр записывается шестнадцатеричная величина (00H-FFH)

ROMC Регистр управления программированием EEPROM

1FFDH/1FFCH (чтение/запись)

Регистр управляет режимами работы с EEPROM. Установка или очистка битов может приводить к записи слова, очистке строки памяти или блока. Зарезервированные биты используются для технологической проверки ПЗУ и при установке этих битов в процессе нормального функционирования возможны сбои.

Рисунок В. 92 – Формат регистра ROMC

Таблица В.102 – Функциональные назначения полей регистра ROMC

Номер бита	Мнемоника	Имя бита	Описание
0	EW	Разрешение записи	Разрешает операции записи слова. Используется вместе с битом RE и BE для операций очистки. После каждой операции сбрасывается аппаратно
1	RE	Стирание строки	Разрешает стирание строки памяти. Для стирания необходима установка бита EW и подача нулевых данных на любую ячейку, находящуюся в строке. После каждой операции стирания строки аппаратно сбрасывается
2	BW	Стирание блока	Разрешает стирание блока памяти. Для стирания необходима установка бита EW и подача нулевых данных на любую ячейку, находящуюся в блоке. После каждой операции стирания блока аппаратно сбрасывается
5	STBY	Режим пониженного энерго-потребления	Переводит ПЗУ в режим пониженного энергопотребления. Не следует записывать «1», если включен режим работы с внутренней памятью. При работе с внешней памятью этот бит автоматически после сброса устанавливается в «1». Время перехода из режима пониженного потребления в активный режим – не менее 5 мкс
3, 4, 6 – 15		Зарезервированы	

SBUF (RX0) Буферный регистр приёма последовательного порта UART0

07H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Буферный регистр приёма последовательного порта SBUF(RX0) содержит данные, принятые из последовательного порта UART0. Приёмник последовательного порта имеет двойную буферизацию и может принимать второй байт данных до того, как первый байт считан. Данные задерживаются в сдвиговом регистре приёмника, пока не получен последний бит данных, а затем байты данных загружаются в SBUF (RX0). Если данные в сдвиговом регистре загружаются в SBUF (RX0) до того, как предыдущий байт считан, будет установлен бит ошибки переполнения (SP_STAT0.2). Данными в SBUF (RX0) всегда будет последний принятый байт, но никогда комбинация последних двух байтов.

Рисунок В.93 – Формат регистра SBUF (RX0)

Таблица В.103 – Функциональные назначения полей регистра SBUF (RX0)

Номер бита	Мнемоника	Имя бита	Описание
0-7	RX0_DATA	Принятые данные	Эти биты содержат последний байт, принятый из последовательного порта UART0

SBUF (RX1) Буферный регистр приёма последовательного порта UART1

1FABH

Буферный регистр приёма последовательного порта SBUF(RX1) содержит данные, принятые из последовательного порта UART1. Приёмник последовательного порта имеет двойную буферизацию и может принимать второй байт данных до того, как первый байт считан. Данные задерживаются в сдвиговом регистре приёмника, пока не получен последний бит данных, а затем байты данных загружаются в SBUF (RX1). Если данные в сдвиговом регистре загружаются в SBUF (RX1) до того, как предыдущий байт считан, будет установлен бит ошибки переполнения (SP_STAT1.2). Данными в SBUF (RX1) всегда будет последний принятый байт, но никогда комбинация последних двух байтов.

Рисунок В.94 – Формат регистра SBUF (RX1)

Таблица В.104 – Функциональные назначения полей регистра SBUF (RX1)

Номер бита	Мнемоника	Имя бита	Описание
0-7	RX1_DATA	Принятые данные	Эти биты содержат последний байт, принятый из последовательного порта UART1

SBUF (TX0) Передающий буферный регистр последовательного порта UART0

07H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Передающий буферный регистр последовательного порта SBUF(TX0) содержит данные, которые готовы к передаче. В режимах 1, 2 и 3 запись в SBUF (TX0) начинает передачу. В режиме 0 запись в SBUF (TX0) начинает передачу только при запрещённом приёме (SP_CON0.3=0).

Рисунок В.95 – Формат регистра SBUF (TX0)

Таблица В.105 – Функциональные назначения полей регистра SBUF (TX0)

Номер бита	Мнемоника	Имя бита	Описание
0-7	TX0_DATA	Данные для передачи	Эти биты содержат байт данных, предназначенный для передачи последовательным портом

SBUF (TX1) Передающий буферный регистр последовательного порта UART1

1FA9H

Передающий буферный регистр последовательного порта SBUF(TX1) содержит данные, которые готовы к передаче. В режимах 1, 2 и 3 запись в SBUF (TX1) начинает передачу. В режиме 0 запись в SBUF (TX1) начинает передачу только при запрещённом приёме (SP_CON1.3=0).

SBUF (TX1)
передающий буферный регистр последовательного порта UART1

$1FA9_H$ значение после Сброса: 00_H

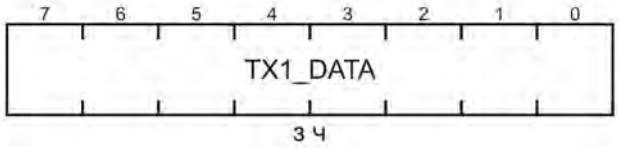


Рисунок В.96 – Формат регистра SBUF (TX1)

Таблица В.106 – Функциональные назначения полей регистра SBUF (TX1)

Номер бита	Мнемоника	Имя бита	Описание
0-7	TX1_DATA	Данные для передачи	Эти биты содержат байт данных, предназначенный для передачи последовательным портом

SP Указатель стека

19/18H

Все горизонтальные окна (чтение/запись)

Системный указатель стека может указывать на любую ячейку в 64К-байтной внутренней или внешней памяти; он должен хранить выровненное слово и должен быть инициализирован перед использованием. Указатель стека декрементируется перед командой PUSH и инкрементируется после команды POP, так что указатель стека нужно инициализировать на 2 байта выше верхней ячейки стека. Если стековые операции не производятся, ячейки 18H и 19H могут использоваться как стандартное ОЗУ (RAM).

Рисунок В.97– Формат регистра SP

Таблица В.107– Функциональные назначения полей регистра SP

Номер бита	Мнемоника	Имя бита	Описание
0-7	SP_LO	Младший байт указателя стека	Младший байт системного указателя стека
8-15	SP_HI	Старший байт указателя стека	Старший байт системного указателя стека

SMBADDR Регистр собсвенного адреса I2C

1FB8H
Содержит собственный адрес порта I2C и бит разрешения адреса Slave.

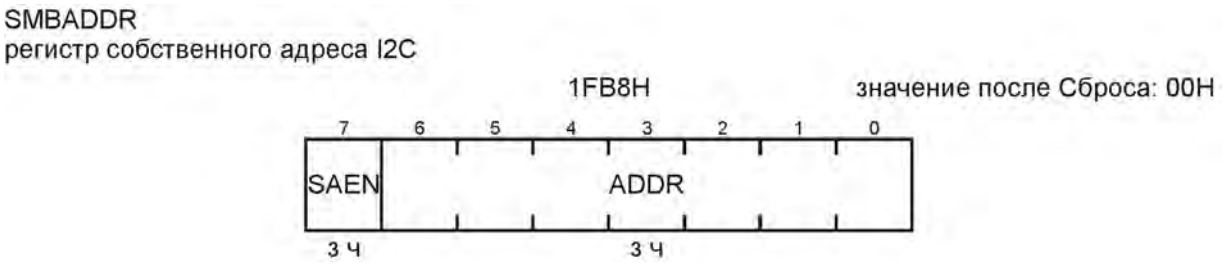


Рисунок В.98 – Формат регистра SMBADDR

Таблица В.108 – Функциональные назначения полей регистра SMBADDR

Номер бита	Мнемоника	Имя бита	Описание
6-0	ADDR	Собственный адрес	Собственный адрес. Содержит 7-битный адрес I2C. При работе I2C в режиме Slave первые 7 бит, принятые после состояния «Старт», сравниваются с этим полем. Если значение ADDR совпадает с принятым адресом, то SMBST.MODE принимает значение 001_B
7	SAEN	Разрешение адреса Slave	Разрешение адреса Slave 1 Показывает, что поле ADDR содержит значение адреса, разрешает сравнение ADDR и принятого байта адреса

SMBCST Регистр управления и статуса I2C

1FB4H

Регистр содержит биты занятия шины, коэффициента деления для предделителя, бит ошибки ожидания на шине, бит теста SDA и бит разрешения удержания SCL для исправления ошибки.

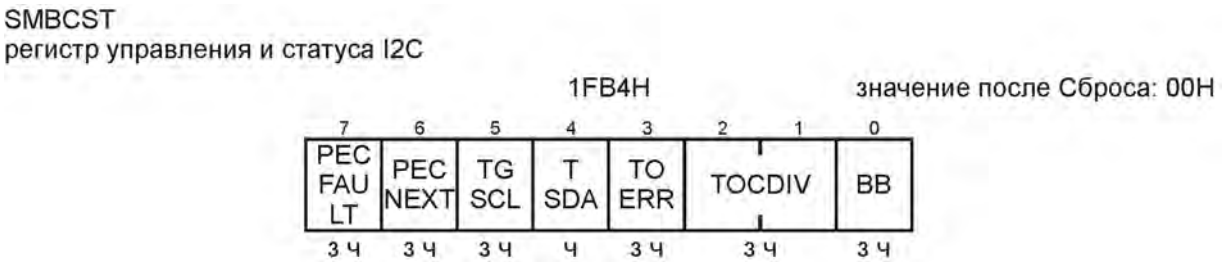


Рисунок В.99 – Формат регистра SMBCST

Таблица В.109 – Функциональные назначения полей регистра SMBCST

Номер бита	Мнемоника	Имя бита	Описание
0	BB	Бит «Шина занята»	Шина занята 0 Бит обнуляется при обнаружении состояния «Стоп» или при запрещении работы I2C 1 Устанавливается, когда шина активна (линии SDA и SCL в низком уровне) или про формировании сигнала «Старт». Показывает, что шина занята
2, 1	TOCDIV	Коэффициент деления	Эти биты определяют коэффициент деления для прескалера времени ожидания SCL. 00 Запрещение работы, нет тактового сигнала 01 Деление на 4 10 Деление на 8 11 Деление на 16
3	TOERR	Ошибка ожидания на шине	Ошибка ожидания на шине. 0 Бит очищается при записи «1» в бит SMBCTL1.CLRST 1 Зафиксировано состояние ожидания на линии SCL. Бит устанавливается при достижении нуля основным счетчиком времени ожидания
4	TSDA	Тест SDA	Тест SDA Содержит текущее значение SDA. Этот бит может быть использован при исправлении ошибки на шине. Этот бит только для чтения, попытка записи в него игнорируется
5	TGSCL	Удерживание SCL в неактивном уровне	Удерживание SCL в неактивном уровне. Разрешает удерживание SCL в неактивном уровне на период исправления ошибки. 1 Удерживает SCL в неактивном уровне на один цикл. Попытка записи в этот бит при высоком уровне сигнала на SDA игнорируется. Бит очищается при завершении удерживания SCL

SMBCTRL1 Регистр управления 1 I2C

1FB6H

SMBCTL1 является 8-битным регистром конфигурации и управления I2C. После сброса и при запрещении работы I2C (SMBCTL2.ENABLE = 0_B) регистр принимает значение 00_H.

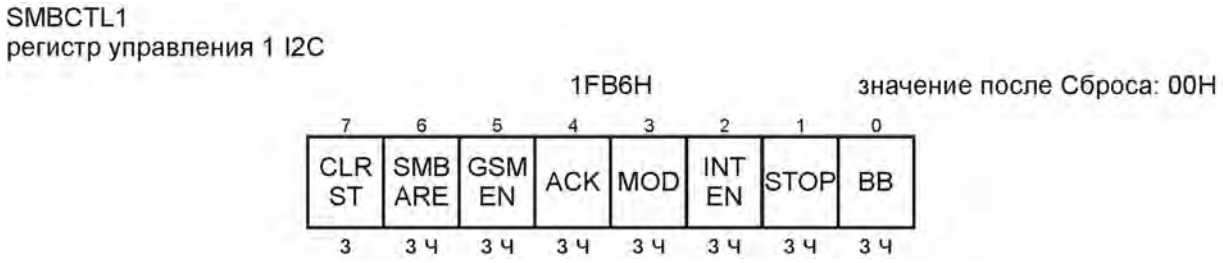


Рисунок В.100 – Формат регистра SMBCTRL1

Таблица В.110– Функциональные назначения полей регистра SMBCTRL1

Номер бита	Мнемоника	Имя бита	Описание
0	START	Бит «Старт»	«Старт» 1 Устанавливается при необходимости формирования состояния «Старт» на шине Бит очищается при завершении формирования состояния «Старт» или после обнаружения ошибки на шине (SMBST.MODE = $1F_H$)
1	STOP	Бит «Стоп»	«Стоп» 1 При работе в режиме Master – формирование состояния «Стоп», что завершит или прервет текущую передачу данных Бит очищается после завершения формирования состояния «Стоп»
2	INTEN	Бит разрешения прерывания	Разрешение прерывания 0 Прерывание запрещено 1 Прерывание разрешено
3	–	–	Зарезервировано
4	ACK	Бит подтверждения	Бит подтверждения ACK игнорируется в режиме передатчика. 1 Сообщает передающему устройству о необходимости остановки передачи данных

Окончание таблицы B.110

Номер бита	Мнемоника	Имя бита	Описание
5	GCMEN	Бит разрешения распознавания адреса общего вызова	Разрешение распознавания адреса общего вызова 0 I2C не распознает адрес общего вызова. Бит очищается при выходе I2C из режимов Halt или Idle 1 Разрешение распознавания среди принятых адресов адреса общего вызова (00_H) в режиме Slave
6	SMBARE	Бит разрешения распознавания адреса отклика	Разрешение распознавания адреса отклика 0 I2C не отвечает на адрес отклика на шине. Бит очищается при выходе I2C из режимов Halt или Idle 1 Разрешение распознавания среди принятых адресов адреса отклика (0001100_B) в режиме Slave
7	CLRST	Бит очистки статуса	Очистка статуса 1 Очищает бит статусный бит SMBST.INT. Запись «0» в этот бит не имеет эффекта, этот бит всегда читается как «0»

SMBCTRL2 Регистр управления 2 I2C

1FBAH
SMBCTL1 управляет разрешением работы порта I2C и частотой SCL.

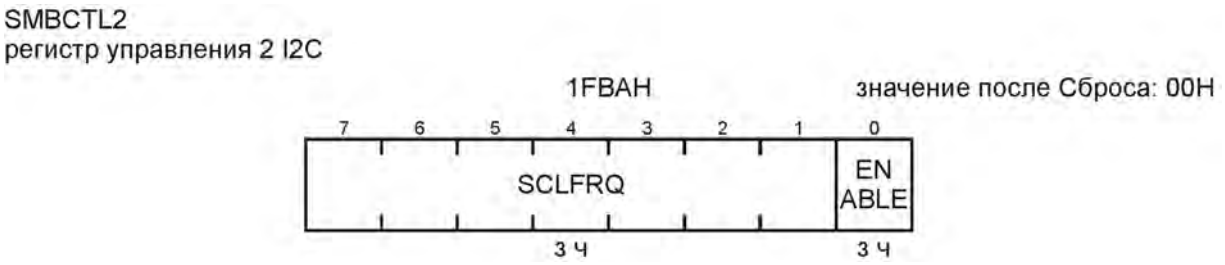


Рисунок В.101 – Формат регистра SMBCTRL2

Таблица В.111 – Функциональные назначения полей регистра SMBCTRL2

Номер бита	Мнемоника	Имя бита	Описание
0	ENABLE	Бит разрешения работы	Разрешение работы. 0 Запрещение работы I2C. При очищении бита ENABLE регистры SMBCTL1, SMBST, SMBCST обнуляются, прекращается генерация тактового сигнала 1 Разрешение работы I2C
7	SCLFRQ	Частота SCL	Частота SCL. Это поле определяет значение периода SCL при работе I2C в режиме Master. В поле SCLFRQ может быть записано значение от четырех до 127. При выборе значения меньше четырех, оно автоматически дополняется до четырех

SMBCTRL3 Регистр управления 3 I2C

1FBEH

Регистр содержит старшие разряды 10-битного адреса, бит разрешения 10-битной адресации и значение коэффициента деления для тактового сигнала в режиме Hs.

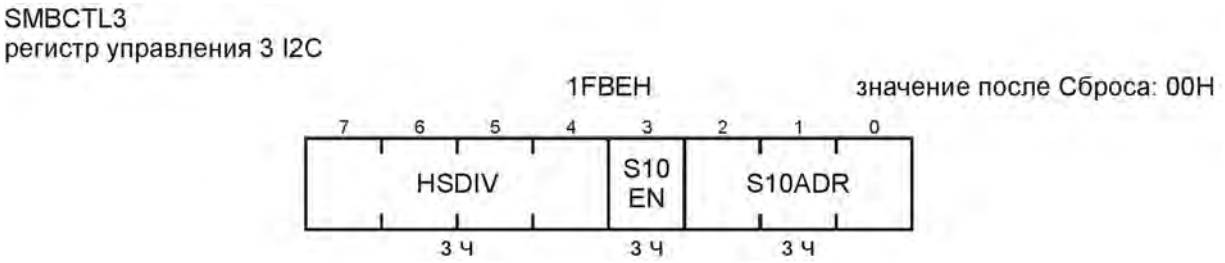


Рисунок В.102 – Формат регистра SMBCTRL3

Таблица В.112 – Функциональные назначения полей регистра SMBCTRL3

Номер бита	Мнемоника	Имя бита	Описание
0-2	S10ADR	Старшие разряды 10-битного адреса	10-битный адрес Slave. Содержит старшие биты 10-битного адреса Slave. Старшие 5 бит первого принятого адреса сравниваются с величиной 1110_B, младшие 3 бита – с величиной S10ADR.2 - S10ADR.1. Старший бит второго принятого адреса сравнивается с величиной S10ADR.0, младшие 7 бит – со значением поля SMBADDR.SADR
3	S10EN	Разрешение 10-битной адресации	Разрешение 10-битной адресации. 1 Разрешает распознавание 10-битного адреса Slave Для распознавания 10-битного адреса требуется одновременная установка битов S10EN и SMBADDR.SAEN
4-7	HSDIV	Делитель тактового сигнала SCL в режиме Hs	Делитель тактового сигнала SCL в режиме Hs Определяет значение периода SCL при работе I2C в высокоскоростном режиме. В поле HSDIV может быть записано значение от двух до 16. При записи числа, меньшего двух, к начальному значению по умолчанию добавляется два. То есть, при выборе значения делителя, равного единице, результирующим будет значение три

SMBSDA Сдвиговый регистр данных I2C

1FB0H

Регистр SMBSDA является 8-битным сдвиговым регистром, используемым для передачи и приема данных. Наиболее значимый бит (MSB) передается (принимается) первым, а наименее значимый бит (LSB) передается (принимается) последним.

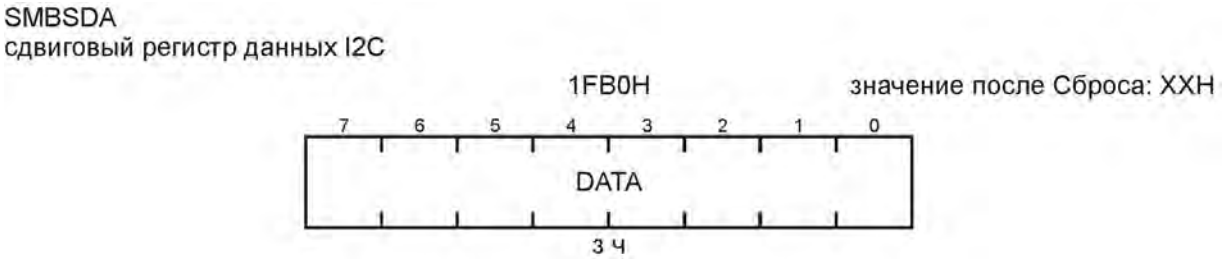


Рисунок В.103 – Формат регистра SMBSDA

Таблица В.113 – Функциональные назначения полей регистра SMBSDA

Номер бита	Мнемоника	Имя бита	Описание
0-7	DATA	Данные	Запись в SMBSDA возможна только в случае, когда установлен флаг SMBST.INT. Попытки записи регистра в других случаях будут проигнорированы. Регистр SMBSDA не очищается после сброса, а содержит случайные данные до того, как будет записан программно или до приема и сдвига данных

SMBST Регистр статуса I2C

1FB2H

Регистр SMBST является 8-битным регистром, содержащим текущее значение статуса I2C. Регистр только для чтения. После сброса и во время запрещения работы I2C, SMBST принимает значение 00_H.

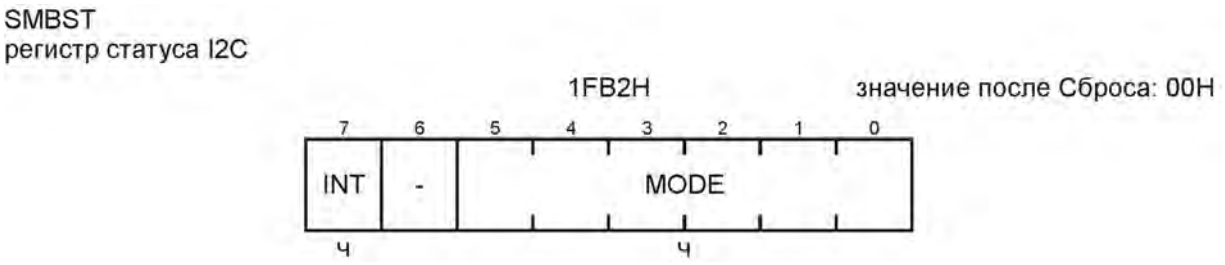


Рисунок В.104 – Формат регистра SMBST

Таблица В.114 – Функциональные назначения полей регистра SMBST

Номер бита	Мнемоника	Имя бита	Описание
0-5	MODE	Текущий статус и режим работы I2C	Текущий статус и режим работы I2C. Расшифровка кода статуса шины приведена в таблице
6	–	–	Зарезервировано
7	INT	Флаг прерывания	Флаг прерывания. 0 Устанавливается после записи «1» в бит SMBCTL1.CRST или при запрещении работы I2C при обнулении бита SMBCTL2.ENABLE 1 Устанавливается после девятого тактового импульса на SCL (SCL – в низком уровне), в любое время разрешено программное вмешательство. (Условия выставления флага прерывания – см. ниже)

Таблица В.115 – Режимы работы I2C

Режим	Код	Мнемонический код	Описание
1	2	3	4
Общий	00_H	IDLE	Idle, недоступна достоверная информация о статусе
F/S Master	01_H	STDONE	Сформировано состояние «Старт»
	02_H	RSDONE	Сформировано состояние «Повторный старт»
	03_H	IDLARL	Потеря приоритета, вход в режим «Неадресованный Slave»
F/S Master передатчик	04_H	MTADPA	Отправлен адрес Slave, сигнал ACK
	05_H	MTADNA	Отправлен адрес Slave, сигнал NACK
	06_H	MTDAPA	Отправлен байт данных, сигнал ACK
	07_H	MTDANA	Отправлен байт данных, сигнал NACK
F/S Master приемник	08_H	MRADPA	Отправлен адрес Slave, сигнал ACK
	09_H	MRADNA	Отправлен адрес Slave, сигнал NACK
	$0A_H$	MRDAPA	Принят байт данных, сигнал ACK
	$0B_H$	MRDANA	Принят байт данных, сигнал NACK
F/S Master	$0C_H$	MTMCER	Адрес Master передан с ошибкой (сигнал ACK)

Окончание таблицы В.115

1	2	3	4
-	$0D_H$ -$0F_H$	-	Не используются
F/S Slave приемник	10_H	SRADPA	Принят адрес Slave, сигнал ACK
	11_H	SRAAPA	Принят адрес Slave после потери приоритета, сигнал ACK
	12_H	SRDAPA	Принят байт данных, сигнал ACK
	13_H	SRDANA	Принят байт данных, сигнал NACK
F/S Slave передатчик	14_H	STADPA	Принят адрес Slave, сигнал ACK
	15_H	STAAPA	Принят адрес Slave после потери приоритета, сигнал ACK
	16_H	STDAPA	Отправлен байт данных, сигнал ACK
	17_H	STDANA	Отправлен байт данных, сигнал NACK
F/S Slave передатчик отклик	18_H	SATADP	Принят адрес отклика, сигнал ACK
	19_H	SATAAP	Принят адрес отклика после потери приоритета, сигнал ACK
	$1A_H$	SATDAP	Отправлены данные отклика, сигнал ACK
	$1B_H$	SATDAN	Отправлены данные отклика, сигнал NACK
F/S Slave	$1C_H$	SSTOP	В режиме Slave зафиксировано состояние «Стоп»
	$1D_H$	SGADPA	Принят адрес общего вызова, сигнал ACK
	$1E_H$	SDAAPA	Принят адрес общего вызова после потери приоритета, сигнал ACK
Общий	$1F_H$	BERROR	Ошибка на шине (некорректное формирование состояний «Старт» или «Стоп»)
Hs Master	20_H	-	Не используется
	21_H	HMTMCOK	Адрес Master передан – переход в режим Hs
	22_H	HRSDONE	Формирование состояния «Повторный старт»
	23_H	HIDLARL	Потеря приоритета, переход в режим Hs «Неадресованный Slave»
Hs Master передатчик	24_H	HMTADPA	Отправлен адрес Slave, сигнал ACK
	25_H	HMTADNA	Отправлен адрес Slave, сигнал NACK
	26_H	HMTDAPA	Отправлен байт данных, сигнал ACK
	27_H	HMTDANA	Отправлен байт данных, сигнал NACK
Hs Master приемник	28_H	HMRADPA	Отправлен адрес Slave, сигнал ACK
	29_H	HMRADNA	Отправлен адрес Slave, сигнал NACK
	$2A_H$	HMRDAPA	Принят байт данных, сигнал ACK
	$2B_H$	HMRDANA	Принят байт данных, сигнал NACK
-	$2C_H$ -$2F_H$	-	Не используются
Hs Slave приемник	30_H	HSRADPA	Принят адрес Slave, сигнал ACK
	31_H	-	Не используется
	32_H	HSRDAPA	Принят байт данных, сигнал ACK
	33_H	HSRDANA	Принят байт данных, сигнал NACK
Hs Slave приемник	34_H	HSTADPA	Принят адрес Slave, сигнал ACK
	35_H	-	Не используется
	36_H	HSTDAPA	Отправлен байт данных, сигнал ACK
	37_H	HSTDANA	Отправлен байт данных, сигнал NACK
-	38_H -$3F_H$	-	Не используется

Условия выставления флага прерывания:

- при работе в режимах «Master передатчик», «Master приемник», «Slave приемник» или «Slave передатчик»;
- после обнаружения совпадения адреса (совпадение адреса Slave, адреса отклика, адреса общего вызова); необходимо программно проверять содержимое регистра SMBSDA для определения, какое совпадение произошло;
- после успешного формирования состояний «Старт» или «Повторный старт»;
- если был получен инверсного сигнала подтверждения (NACK);
- при успешном формировании состояний «Стоп» или «Повторный старт»;
- после обнаружения ошибки на шине.

Линия SCL удерживается в низком уровне, пока установлен флаг прерывания SMBST.INT. Следующие условия устанавливают флаг прерывания, но не приводят к удерживанию SCL в низком уровне (ожидание на линии SCL):

- состояние «Стоп» в режиме Slave (SMBST.MODE = SSTOP ($1C_H$));
- потеря приоритета приводит к входу I2C в режим «Неадресованный Slave» (SMBST.MODE = IDLARL(03_H) или SMBST.MODE = HIDLARL (23_H));
- передан байт данных, инверсный сигнал подтверждения NACK (17_H).

SMBTOPR Регистр прескалера времени ожидания SCL

1FBCH
Содержит значение предделителя времени ожидания.

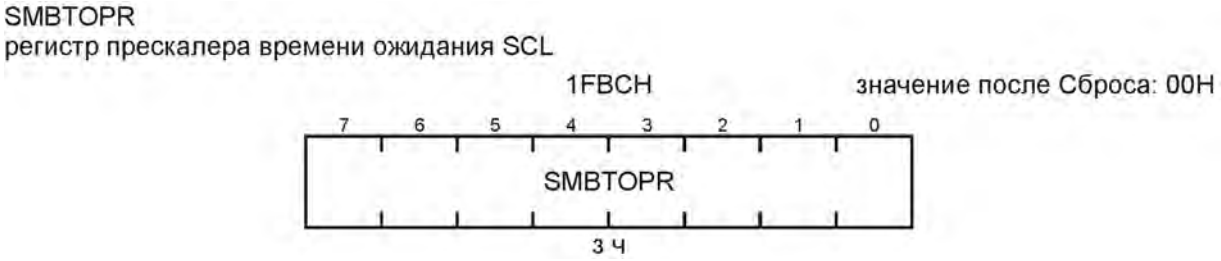


Рисунок В.105 – Формат регистра SMBTOPR

Таблица В.116 – Функциональные назначения полей регистра SMBTOPR

Номер бита	Мнемоника	Имя бита	Описание
0-7	SMBTOPR	Прескалер времени ожидания SCL	Прескалер времени ожидания SCL. Содержит значение прескалера времени ожидания. После сброса принимает значение 00_H

SPCR Регистр управления SPI

1FA0H

Управляет работой порта SPI. Выбирает частоту работы, фазу и полярность тактового сигнала, режим Master/Slave, направление передачи данных, разрешает работу порта и генерирование прерываний.

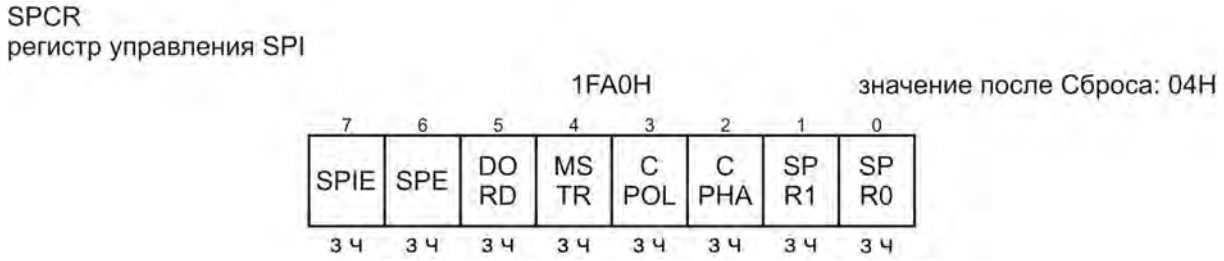


Рисунок В.106 – Формат регистра SPCR

Таблица В.117 – Функциональные назначения полей регистра SPCR

Номер бита	Мнемоника	Имя бита	Описание
0-1	SPR0, SPR1	Частота тактового сигнала	Выбор частоты тактового сигнала SPI. Данные два бита управляют сигналом тактирования SCK Master-устройства. В случае работы устройства в режиме Slave биты SPR1 и SPR0 не действуют. Между SCK и частотой осциллятора f существуют следующие соотношения: SPR1 SPR0 SCK 0 0 f/4 (f/2 в режиме x2) 0 1 f/16 (f/8 в режиме x2) 1 0 f/64 (f/32 в режиме x2) 1 1 f/128 (f/64 в режиме x2)
2	CPHA	Выбор фазы	Фаза тактового сигнала. Бит CPHA вместе с битом CPOL управляет соотношением фазы тактового сигнала и данных на шине
3	CPOL	Выбор полярности	Полярность тактового сигнала 0 На выводе SCK Master-устройства сигнал низкого уровня при отсутствии импульсов. 1 На выводе SCK сигнал высокого уровня при отсутствии импульсов
4	MSTR	Master/Slave	Выбор режима Master/Slave 0 Выбор режима Slave SPI 1 Выбор режима Master SPI
5	DORD	Направление передачи данных	Направление передачи данных. 0 Выбор MSB в качестве первого бита для передачи 1 Выбор LSB в качестве первого бита для передачи
6	SPE	Разрешение SPI	Разрешение работы SPI. 0 Запрещение канала SPI 1 Разрешение канала SPI

Окончание таблицы В.117

Номер бита	Мнемоника	Имя бита	Описание
7	SPIE	Разрешение прерывания	Разрешение прерывания SPI. Данный бит вместе с битом SER_MASK в регистре INT_MASK разрешает прерывание SPI: при SPIE = 1 и ES = 1 – прерывание SPI разрешено. 0 Запрещение прерывания SPI 1 Разрешение прерывания SPI
П р и м е ч а н и я 1 Все параметры сигнала тактирования, а также направление передачи данных, должны быть выставлены до разрешения работы блока SPI. Только после этого можно устанавливать бит SPE. 2 Master-устройство должно быть включено раньше, чем Slave-устройство. 3 При необходимости выключения устройств, Master-устройство следует выключать первым.			

SPDR Регистр данных SPI

1FA4H

При чтении содержит данные, принятые портом SPI. Запись данных в этот регистр инициирует процесс приема/передачи в соответствии со спецификацией.

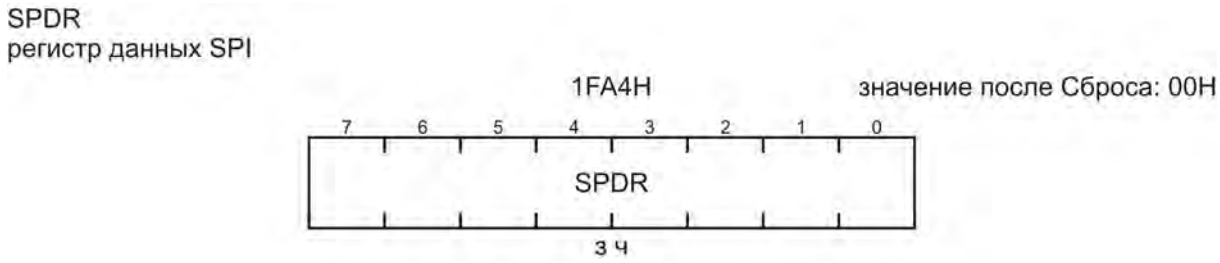


Рисунок В.107 – Формат регистра SPDR

Таблица В.118 – Функциональные назначения полей регистра SPDR

Номер бита	Мнемоника	Имя бита	Описание
0-7	SPDR	Данные приема/передачи	При чтении содержит данные, принятые портом SPI. Запись в регистр инициирует передачу данных

SPSR Регистр статуса SPI

1FA2H

Содержит статусные биты порта SPI. Управляет работой вывода MISO для Slave устройства.

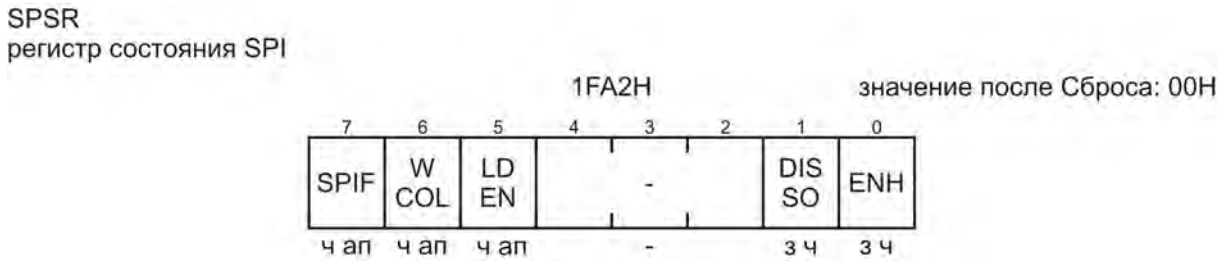


Рисунок В.108 – Формат регистра SPSR

Таблица В.119 – Функциональные назначения полей регистра SPSR

Номер бита	Мнемоника	Имя бита	Описание
0	ENH	Расширенный режим	Бит выбора расширенного режима SPI. 0 SPI работает в стандартном режиме, то есть без двойной буферизации записи. 1 SPI работает в расширенном режиме с двойной буферизацией записи. Для буфера Tx используется такой же адрес, как и для регистра SPDR
1	DISSO	Третье состояние	Бит запрещения работы передающего вывода Slave-устройства. При установке данного бита вывод MISO переходит в третье состояние, в случае, если устройство не выбрано (сигнал SS# находится в некактивном уровне).
2-4	–	–	Зарезервировано
5	LDEN	Разрешение загрузки	Разрешение загрузки в буфер записи данных в расширенном режиме. При ENH = 1, LDEN = 1 и WCOL = 0, запись в буфер записи данных является надежной. Во время передачи байта выполняются условия: 1. При передаче первых четырех битов LDEN = 1. 2. При передаче последних четырех битов LDEN = 0
6	WCOL	Буфер данных загружен	1. Стандартный режим (ENH = 0). Флаг конфликта записи. Флаг WCOL устанавливается при попытке записи в регистр SPDR во время передачи данных. Флаг WCOL сбрасывается одновременно с флагом SPIF. 2. Расширенный режим (ENH = 1). Флаг WCOL в расширенном режиме информирует о заполнении буфера записи данных. Если WCOL = 1, то запись в регистр SPDR приведет к перезаписи данных, уже хранящихся в буфере записи данных. В этом режиме WCOL сбрасывается только при перезагрузке данных из буфера записи данных в сдвиговый регистр

Окончание таблицы В.119

Номер бита	Мнемоника	Имя бита	Описание
7	SPIF	Флаг прерывания	Флаг прерывания SPI. При завершении операции передачи/приема устанавливается бит SPIF, и, в случае, если бит SPIE = 1 и бит SER_MASK = 1 генерируется прерывание. После выставления флага SPIF можно снова записать данные в регистр SPDR или считать полученные данные из регистра SPDR. Состояние флага SPIF определяется чтением регистра SPSR, после чего, если есть необходимость сброса установленного флага, достаточно обратиться (запись/чтение) к регистру SPDR

SP_CON0 Регистр управления последовательным портом UART0

11H

Горизонтальное окно 0 (запись), горизонтальное окно 15 (чтение)

Регистр управления последовательным портом SP_CON0 выбирает режим связи, разрешает или запрещает приём, а также контролирует чётность и позволяет работать с 9-битной передачей данных. TXD0 мультиплексирован с P2.0. Для разрешения функционирования TXD0 IOC1.5 должен быть установлен. TXD0 служит как канал передачи для режимов 1, 2 и 3 последовательного порта и как канал синхронизации для сдвигового регистра в режиме 0. RXD0 мультиплексирован с P2.1. Для разрешения функционирования RXD0 SP_CON0.3 должен быть установлен. RXD0 получает данные последовательного порта в режимах 1, 2 и 3 и служит для ввода или вывода данных в режиме 0 (при выводе он функционирует как выход с открытым стоком).

Рисунок В.109 – Формат регистра SP_CON0

Таблица В.120 – Функциональные назначения полей регистра SP_CON0

Номер бита	Мнемоника	Имя бита	Описание
0, 1	SER_MODE0	Выбор режима	Эти два бита выбирают режим связи. Бит 1 Бит 0 Режим 0 0 Режим 0 0 1 Режим 1 1 0 Режим 2 1 1 Режим 3
2	PEN0	Разрешение контроля чётности	В режимах 1 и 3 установка этого бита разрешает функцию чётности. В режиме 2 этот бит должен быть очищен. Когда этот бит установлен, TB8 содержит дополнение до четности для передачи. С разрешённым контролем чётности при приёме
3	REN0	Разрешение приёма	Установка этого бита разрешает функционирование RXD0 в канале P2.1/ RDX0. Когда этот бит установлен, по срезу сигнала начинается приём в режимах 1, 2 и 3. В режиме 0 этот бит должен быть сброшен для начала передачи и установлен для начала приёма. Очистка этого бита останавливает приём и препятствует дальнейшему приёму
4	TB80	Передача 9-ого бита данных	Это девятый бит данных, который передаётся в режимах 2 и 3. Этот бит сбрасывается после любой передачи, поэтому он должен быть установлен до записи в SBUF (TX0). Когда SPCON0.2 установлен, этот бит содержит дополнение до чётности
5 – 7	-	-	Зарезервированы; всегда записывают нуль

SP_CON1 Регистр управления последовательным портом UART1

1FAFH

Регистр управления последовательным портом SP_CON1 выбирает режим связи, разрешает или запрещает приём, а также контролирует чётность и позволяет работать с 9-битной передачей данных. TXD1 мультиплексирован с P2.6. Для разрешения функционирования TXD1 IOC1.5 должен быть установлен. TXD1 служит как канал передачи для режимов 1, 2 и 3 последовательного порта и как канал синхронизации для сдвигового регистра в режиме 0. RXD1 мультиплексирован с P2.4. Для разрешения функционирования RXD1 SP_CON1.3 должен быть установлен. RXD1 получает данные последовательного порта в режимах 1, 2 и 3 и служит для ввода или вывода данных в режиме 0 (при выводе он функционирует как выход с открытым стоком).

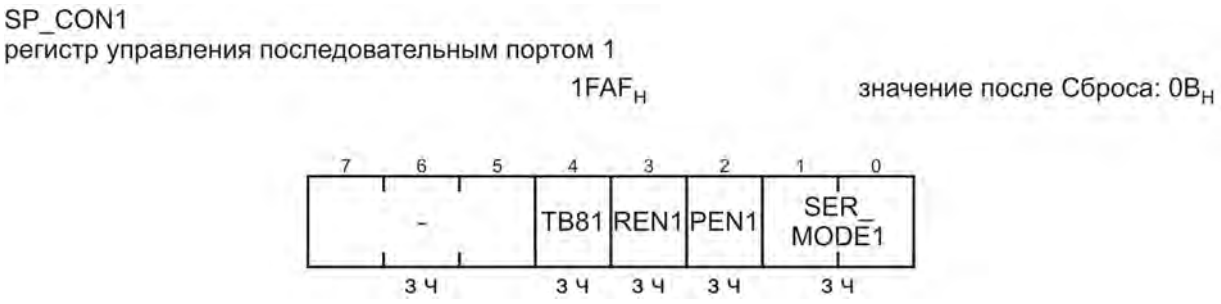


Рисунок В.110 – Формат регистра SP_CON1

Таблица В.121 – Функциональные назначения полей регистра SP_CON1

Номер бита	Мнемоника	Имя бита	Описание
0, 1	SER_MODE1	Выбор режима	Эти два бита выбирают режим связи. Бит 1 Бит 0 Режим 0 0 Режим 0 0 1 Режим 1 1 0 Режим 2 1 1 Режим 3
2	PEN1	Разрешение контроля чётности	В режимах 1 и 3 установка этого бита разрешает функцию чётности. В режиме 2 этот бит должен быть очищен. Когда этот бит установлен, TB8 содержит дополнение до четности для передачи. С разрешённым контролем чётности при приёме
3	REN1	Разрешение приёма	Установка этого бита разрешает функционирование RXD1 в канале P2.4/ RDX1. Когда этот бит установлен, по срезу сигнала начинается приём в режимах 1, 2 и 3. В режиме 0 этот бит должен быть сброшен для начала передачи и установлен для начала приёма. Очистка этого бита останавливает приём и препятствует дальнейшему приёму
4	TB81	Передача 9-ого бита данных	Это девятый бит данных, который передаётся в режимах 2 и 3. Этот бит сбрасывается после любой передачи, поэтому он должен быть установлен до записи в SBUF (TX1). Когда SPCON1.2 установлен, этот бит содержит дополнение до чётности
5 – 7	-	-	Зарезервированы; всегда записывают нуль

SP_STAT0 Регистр состояния последовательного порта UART0

11H

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Регистр состояния последовательного порта содержит биты, которые показывают состояние последовательного порта UART0.

Рисунок В.111 – Формат регистра SP_STAT0

Таблица В.122 – Функциональные назначения полей регистра SP_STAT0

Номер бита	Мнемоника	Имя бита	Описание
0, 1	–	–	Зарезервированы; всегда записывают нули
2	OE0	Ошибка переполнения	Этот бит устанавливается, если данные из сдвигового регистра приёма загружены в регистр SBUF (RX0), до того, как предыдущий байт считан. Чтение из SP_STAT0 очищает этот бит
3	TXE0	SBUF (TX) пуст	Этот бит устанавливается, если буфер передачи пуст и готов к приёму двух символов. Он очищается после записи байта в SBUF (TX0)
4	FE0	Ошибка кадра	Этот бит устанавливается, если стоповый бит не найден в течение определённого периода времени. Чтение из SP_STAT0 очищает этот бит
5	TI0	Прерывание передачи	Этот бит устанавливается в начале передачи стопового бита. Чтение из SP_STAT0 очищает этот бит
6	RI0	Прерывание при приёме	Этот бит устанавливается после выборки последнего бита данных. Чтение из SP_STAT0 очищает этот бит. Этот бит не обязательно сбрасывать для нормального приёма следующих данных последовательным портом
7	RPE0/RB80	Ошибка приёма по чётности/ принятый восьмой бит	RPE0 устанавливается при запрещённой проверке чётности (SP_CON0.2 = 0) и получении девятого бита данных, равного единице. RP80 устанавливается при разрешённой проверке чётности (SP_CON0.2 = 1) и возникшей ошибке чётности. Чтение из SP_STAT0 очищает этот бит

SP_STAT1 Регистр состояния последовательного порта UART1

1FADH

Регистр состояния последовательного порта содержит биты, которые показывают состояние последовательного порта UART1.

Рисунок В.112 – Формат регистра SP_STAT1

Таблица В.123 – Функциональные назначения полей регистра SP_STAT1

Номер бита	Мнемоника	Имя бита	Описание
0, 1	–	–	Зарезервированы; всегда записывают нули
2	OE1	Ошибка переполнения	Этот бит устанавливается, если данные из сдвигового регистра приёма загружены в регистр SBUF (RX1), до того, как предыдущий байт считан. Чтение из SP_STAT1 очищает этот бит
3	TXE1	SBUF (TX) пуст	Этот бит устанавливается, если буфер передачи пуст и готов к приёму двух символов. Он очищается после записи байта в SBUF (TX1)
4	FE1	Ошибка кадра	Этот бит устанавливается, если стоповый бит не найден в течение определённого периода времени. Чтение из SP_STAT1 очищает этот бит
5	TI1	Прерывание передачи	Этот бит устанавливается в начале передачи стопового бита. Чтение из SP_STAT1 очищает этот бит
6	RI1	Прерывание при приёме	Этот бит устанавливается после выборки последнего бита данных. Чтение из SP_STAT1 очищает этот бит. Этот бит не обязательно сбрасывать для нормального приёма следующих данных последовательным портом
7	RPE1/RB81	Ошибка приёма по чётности/ принятый восьмой бит	RPE1 устанавливается при запрещённой проверке чётности (SP_CON1.2 = 0) и получении девятого бита данных, равного единице. RP81 устанавливается при разрешённой проверке чётности (SP_CON1.2 = 1) и возникшей ошибке чётности. Чтение из SP_STAT1 очищает этот бит

T2CAPTURE Регистр фиксации значения таймера 2

0DH/0CH

Горизонтальное окно 15 (чтение/запись)

Передний фронт сигнала в P2.7 ведёт к запоминанию значения таймера 2 в регистре T2CAPTURE и генерированию прерывания Timer 2 Capture (INT11, 2036H). Для разрешения прерывания необходимо установить бит INT_MASK1.3.

Рисунок B.113 – Формат регистра T2CAPTURE

Таблица B.124 – Функциональные назначения полей регистра T2CAPTURE

Номер бита	Мнемоника	Имя бита	Описание
0-7	T2_VALUE (LO)	Младший байт таймера 2	Эти биты содержат младший байт захваченного значения таймера 2
8-15	T2_VALUE (HI)	Старший байт таймера 2	Эти биты содержат старший байт зафиксированного значения таймера 2

TIMER1 Регистр таймера 1

0BH/0AH

Горизонтальное окно 0 (чтение), горизонтальное окно 15 (запись)

Два байта регистра TIMER1 содержат значение таймера 1. В этот регистр можно записывать; при инициализации таймера 1 можно записать значение, отличное от нуля.

Таймер 1 – 16-битный, автономно работающий таймер, который инкрементируется каждый восьмой машинный такт. Установка IOC1.2 ведёт при переполнении таймера 1 к генерации прерывания Timer Overflow (INT00, 2000H), если в INT_MASK.0 также установлено разрешение прерывания.

Необходима осторожность, если во время записи в таймер 1 работают модули HSI и HSO. Время наступления HSO-события, записанного в CAM, зависит от точного соответствия с таймером 1. Аналогично, изменение таймера 1 между входными событиями в режиме HSI будет искажать временные соотношения между событиями.

Рисунок В.114 – Формат регистра TIMER1

Таблица В.125 – Функциональные назначения полей регистра

Номер бита	Мнемоника	Имя бита	Описание
0-7	TIMER1 (LO)	Значение таймера 1 младший байт	Эти биты составляют младший байт значения таймера 1
8-15	TIMER1 (HI)	Значение таймера 1 старший байт	Эти биты составляют старший байт значения таймера 1

TIMER2 Регистр таймера 2

0DH/0CH

Горизонтальное окно 0 (чтение/запись)

Два байта регистра TIMER2 содержат значение таймера 2. В этот регистр можно записывать; для инициализации таймера 2 разрешается записывать значение, отличное от нуля.

Таймер 2 – 16-битный счётчик, который может быть синхронизирован либо внутренне, либо внешне, в зависимости от состояния IOC3.0. Если выбран внешний источник синхронизации, то IOC0.7 определяет, какой из двух внешних источников будет использоваться. В зависимости от состояния IOC2.0 таймер 2 считает или каждые восемь импульсов (нормальный режим) или каждый импульс (режим быстрого инкрементирования).

IOC2.1 управляет направлением счета таймера 2. Когда IOC2.1 = 0, таймер 2 считает на увеличение; когда IOC2.1 = 1, таймер 2 считает на уменьшение или на увеличение, в зависимости от состояния P2. 6. Необходима осторожность, когда это разрешено, и ведётся работа с HSO. Если таймер 2 не считает от начала до конца все 65 536 раз, время события в CAM файле могут никогда не совпасть с текущим значением таймера.

Переполнение таймера 2 может генерировать либо прерывание Timer Overflow (INT00, 2000H), либо прерывание Timer 2 Overflow (INT12, 2038H). Установите или сбросьте IOC1.3, INT_MASK.0 и INT_MASK1.4 для установки желаемого прерывания.

Регистр может обнуляться аппаратно, программно или командой HSO, в зависимости от состояния IOC0.3, IOC0.1 и HSO_COMMAND.0 – HSO_COMMAND.3, соответственно.

Рисунок В.115 – Формат регистра TIMER2

Таблица В.126 – Функциональные назначения полей регистра TIMER2

Номер бита	Мнемоника	Имя бита	Описание
0-7	TIMER2 (LO)	Значение таймера 2 младший байт	Эти биты составляют младший байт значения таймера 2
8-15	TIMER2 (HI)	Значение таймера 2 старший байт	Эти биты составляют старший байт значения таймера 2

WATCHDOG Регистр сторожевого таймера

0AH

Горизонтальное окно 0 (сброс/ разрешение)

Горизонтальное окно 15 (чтение)

Несброшенный через каждые 65536 машинных тактов сторожевой таймер перезагружает МК. Для очистки сторожевого таймера записывают последовательно, сразу друг за другом, «1EH» и «E1H» в ячейку 0AH в горизонтальном окне 0. Очистка этого регистра первый раз разрешает работу сторожевого таймера с начальным значением 0000H, которое инкрементируется каждый машинный такт. После разрешения сторожевой таймер может быть запрещен только через сброс. Восемь старших значащих битов значения сторожевого таймера можно прочитать в горизонтальном окне 15.

Рисунок В.116 – Формат регистра WATCHDOG

Таблица В. 127 – Функциональные назначения полей регистра WATCHDOG

Номер бита	Мнемоника	Имя бита	Описание
0-7	WATCHDOG	Значение сторожевого таймера	Этот байт содержит восемь старших значащих битов текущего значения сторожевого таймера

WSR Регистр выбора окна

14H

Все горизонтальные окна (чтение/запись)

Биты регистра выбирают горизонтальные (HWindows) и вертикальные окна (VWindows). PUSHA записывает этот регистр в стек; POPA восстанавливает его.

Горизонтальные окна позволяют получить доступ к регистрам специальных функций (SFR) путём преобразования 24-байтного окна в младшие 24 байта младшего регистрового файла. Горизонтальные окна 0, 1 и 15 реализованы в МК. Все другие горизонтальные окна зарезервированы. Вертикальные окна отображают сегменты ОЗУ (RAM) в старшую часть младшего регистрового файла, в 32-, 64- или 128-байтные сегменты. МК имеет 2048 байтов внутреннего RAM (ОЗУ), которые могут быть преобразованы в шестьдесят четыре 32-байтные вертикальные окна, в тридцать два 64-байтных вертикальных окна или в шестнадцать 128-байтных вертикальных окна. Горизонтальные и вертикальные окна не могут быть активными одновременно. Для переключения окон следует записывать номер окна в биты 0 – 3 и устанавливать или очищать биты 4 – 6 как надо. Дополнительная информация о выборе окна дана в таблицах В.129 – В.132. В таблице В.129 описан выбор горизонтального окна, в таблице В.130 – выбор 128-байтных вертикальных окон, в таблице В.131 – выбор 64-байтных вертикальных окон, в таблице В.132 – выбор 32-байтных вертикальных окон.

Рисунок В.117 – Формат регистра WSR

Таблица В.128 – Функциональные назначения полей регистра WSR

Номер бита	Мнемо-ника	Имя бита	Описание
0 – 3	W0–W3	0 – 3 биты выбора 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
4	W4	Четвертый бит выбора окна	Этот бит устанавливается при выборе 128-байтных вертикальных окон. В других случаях он должен быть равен нулю. Установка этого бита определяет 128-байтные вертикальные окна
5	W5	Пятый бит выбора окна	Установка этого бита выбирает 64-байтные вертикальные окна. В других случаях он должен быть равен нулю
6	W6	Шестой бит выбора окна	Установка этого бита выбирает 32-байтные вертикальные окна. В других случаях он должен быть равен нулю
7	–	Зарезер-вирован	

Таблица В.129 – Выбор горизонтального окна

Желаемое горизонтальное окно	Биты WSR								Шестнадцатеричное значение
	7	6		4	3	2	1	0	
Hwindow 0	X	0	0	0	0	0	0	0	00H
Hwindow 1	X	0	0	0	0	0	0	1	01H
Hwindow 15	X	0	0	0	1	1	1	1	15H

Таблица В.130 – Выбор 128-байтных вертикальных окон

Желаемое вертикальное окно	Начальный адрес	Биты WSR								Шестнадцатеричное значение
		7	6	5	4	3	2	1	0	
0	0000H	X	0	0	1	0	0	0	0	10H
1	0080H	X	0	0	1	0	0	0	1	11H
2	0100H	X	0	0	1	0	0	1	0	12H
3	0180H	X	0	0	1	0	0	1	1	13H
4	0200H	X	0	0	1	0	1	0	0	14H
5	0280H	X	0	0	1	0	1	0	1	15H
6	0300H	X	0	0	1	0	1	1	0	16H
7	0380H	X	0	0	1	0	1	1	1	17H
8	0400H	X	0	0	1	1	0	0	0	18H
9	0480H	X	0	0	1	1	0	0	1	19H
10	0500H	X	0	0	1	1	0	1	0	1AH
11	0580H	X	0	0	1	1	0	1	1	1BH
12	0600H	X	0	0	1	1	1	0	0	1CH
13	0680H	X	0	0	1	1	1	0	1	1DH
14	0700H	X	0	0	1	1	1	1	0	1EH
15	0780H	X	0	0	1	1	1	1	1	1FH

Таблица В.131 – Выбор 64-байтных вертикальных окон

Желаемое вертикальное окно	Начальный адрес	Биты WSR								Шестнадцатерич-ное значение
		7	6	5	4	3	2	1	0	
1	2	3	4	5	6	7	8	9	10	11
0	0000H	X	0	1	0	0	0	0	0	20H
1	0040H	X	0	1	0	0	0	0	1	21H
2	0080H	X	0	1	0	0	0	1	0	22H
3	00C0H	X	0	1	0	0	0	1	1	23H
4	0100H	X	0	1	0	0	1	0	0	24H
5	0140H	X	0	1	0	0	1	0	1	25H
6	0180H	X	0	1	0	0	1	1	0	26H
7	01C0H	X	0	1	0	0	1	1	1	27H
8	0200H	X	0	1	0	1	0	0	0	28H
9	0240H	X	0	1	0	1	0	0	1	29H
10	0280H	X	0	1	0	1	0	1	0	2aH
11	02C0H	X	0	1	0	1	0	1	1	2bH
12	0300H	X	0	1	0	1	1	0	0	2cH
13	0340H	X	0	1	0	1	1	0	1	2dH

Окончание таблицы В.131

1	2	3	4	5	6	7	8	9	10	11
14	0380H	X	0	1	0	1	1	1	0	2eH
15	03C0H	X	0	1	0	1	1	1	1	2fH
16	0400H	X	0	1	1	0	0	0	0	30H
17	0440H	X	0	1	1	0	0	0	1	31H
18	0480H	X	0	1	1	0	0	1	0	32H
19	04C0H	X	0	1	1	0	0	1	1	33H
20	0500H	X	0	1	1	0	1	0	0	34H
21	0540H	X	0	1	1	0	1	0	1	35H
22	0580H	X	0	1	1	0	1	1	0	36H
23	05C0H	X	0	1	1	0	1	1	1	37H
24	0600H	X	0	1	1	1	0	0	0	38H
25	0640H	X	0	1	1	1	0	0	1	39H
26	0680H	X	0	1	1	1	0	1	0	3aH
27	06C0H	X	0	1	1	1	0	1	1	3bH
28	0700H	X	0	1	1	1	1	0	0	3cH
29	0740H	X	0	1	1	1	1	0	1	3dH
30	0780H	X	0	1	1	1	1	1	0	3eH
31	07C0H	X	0	1	1	1	1	1	1	3fH

Таблица В.132 – Выбор 32-байтных вертикальных окон

Желаемое вертикальное окно	Начальный адрес	Биты WSR								Шестнадцатеричное значение
		7	6	5	4	3	2	1	0	
1	2	3	4	5	6	7	8	9	10	11
0	0000H	X	1	0	0	0	0	0	0	40H
1	0020H	X	1	0	0	0	0	0	1	41H
2	0040H	X	1	0	0	0	0	1	0	42H
3	0060H	X	1	0	0	0	0	1	1	43H
4	0080H	X	1	0	0	0	1	0	0	44H
5	00A0H	X	1	0	0	0	1	0	1	45H
6	00C0H	X	1	0	0	0	1	1	0	46H
7	00E0H	X	1	0	0	0	1	1	1	47H
8	0100H	X	1	0	0	1	0	0	0	48H
9	0120H	X	1	0	0	1	0	0	1	49H
10	0140H	X	1	0	0	1	0	1	0	4AH
11	0160H	X	1	0	0	1	0	1	1	4BH
12	0180H	X	1	0	0	1	1	0	0	4CH
13	01A0H	X	1	0	0	1	1	0	1	4DH
14	01C0H	X	1	0	0	1	1	1	0	4EH
15	01E0H	X	1	0	0	1	1	1	1	4FH
16	0200H	X	1	0	1	0	0	0	0	50H
17	0220H	X	1	0	1	0	0	0	1	51H
18	0240H	X	1	0	1	0	0	1	0	52H
19	0260H	X	1	0	1	0	0	1	1	53H
20	0280H	X	1	0	1	0	1	0	0	54H
21	02A0H	X	1	0	1	0	1	0	1	55H
22	02C0H	X	1	0	1	0	1	1	0	56H
23	02E0H	X	1	0	1	0	1	1	1	57H
24	0300H	X	1	0	1	1	0	0	0	58H

Окончание таблицы В.132

1	2	3	4	5	6	7	8	9	10	11
25	0320H	X	1	0	1	1	0	0	1	59H
26	0340H	X	1	0	1	1	0	1	0	5aH
27	0360H	X	1	0	1	1	0	1	1	5bH
28	0380H	X	1	0	1	1	1	0	0	5cH
29	03A0H	X	1	0	1	1	1	0	1	5dH
30	03C0H	X	1	0	1	1	1	1	0	5eH
31	03E0H	X	1	0	1	1	1	1	1	5fH
32	0400H	X	1	1	0	0	0	0	0	60H
33	0420H	X	1	1	0	0	0	0	1	61H
34	0440H	X	1	1	0	0	0	1	0	62H
35	0460H	X	1	1	0	0	0	1	1	63H
36	0480H	X	1	1	0	0	1	0	0	64H
37	04A0H	X	1	1	0	0	1	0	1	65H
38	04C0H	X	1	1	0	0	1	1	0	66H
39	04E0H	X	1	1	0	0	1	1	1	67H
40	0500H	X	1	1	0	1	0	0	0	68H
41	0520H	X	1	1	0	1	0	0	1	69H
42	0540H	X	1	1	0	1	0	1	0	6AH
43	0560H	X	1	1	0	1	0	1	1	6BH
44	0580H	X	1	1	0	1	1	0	0	6CH
45	05A0H	X	1	1	0	1	1	0	1	6DH
46	05C0H	X	1	1	0	1	1	1	0	6EH
47	05E0H	X	1	1	0	1	1	1	1	6FH
48	0600H	X	1	1	1	0	0	0	0	70H
49	0620H	X	1	1	1	0	0	0	1	71H
50	0640H	X	1	1	1	0	0	1	0	72H
51	0660H	X	1	1	1	0	0	1	1	73H
52	0680H	X	1	1	1	0	1	0	0	74H
53	06A0H	X	1	1	1	0	1	0	1	75H
54	06C0H	X	1	1	1	0	1	1	0	76H
55	06E0H	X	1	1	1	0	1	1	1	77H
56	0700H	X	1	1	1	1	0	0	0	78H
57	0720H	X	1	1	1	1	0	0	1	79H
58	0740H	X	1	1	1	1	0	1	0	7aH
59	0760H	X	1	1	1	1	0	1	1	7bH
60	0780H	X	1	1	1	1	1	0	0	7cH
61	07A0H	X	1	1	1	1	1	0	1	7dH
62	07C0H	X	1	1	1	1	1	1	0	7eH
63	07E0H	X	1	1	1	1	1	1	1	7fH

ZERO_REG Регистр нуля

01/00H

Все горизонтальные окна (чтение/ запись)

Двухбайтный регистр нуля всегда равен нулю. Он используется как фиксированный источник нуля при сравнениях и вычислениях. ZERO_REG может также использоваться как переменная типа WORD в длинно-индексной ссылке. Комбинация выбора регистра и режима адресации делает возможной прямую адресацию к любой ячейке памяти. Команда CMPL (сравнение чисел типа LONG) вместе со средствами регистра ZERO_REG сравнивает с 32-битным нулём.

Рисунок В.118 – Формат регистра ZERO_REG

Таблица В.133 – Функциональные назначения полей регистра ZERO_REG

Номер бита	Мнемоника	Имя бита	Описание
0 – 7	ZERO (LO)	Регистр нуля, младший байт	Младший байт регистра нуля
8 – 15	ZERO (HI)	Регистр нуля, старший байт	Старший байт регистра нуля

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Подп.	Дата
	изме-ненных	заме-нен-ных	новых	анну-лиро-ванных				
-	-	-	все	-	418			08.08.12
1	1	-	-	-	-			28.08.12
2	-	7, 8, 217	-		-			20.02.13