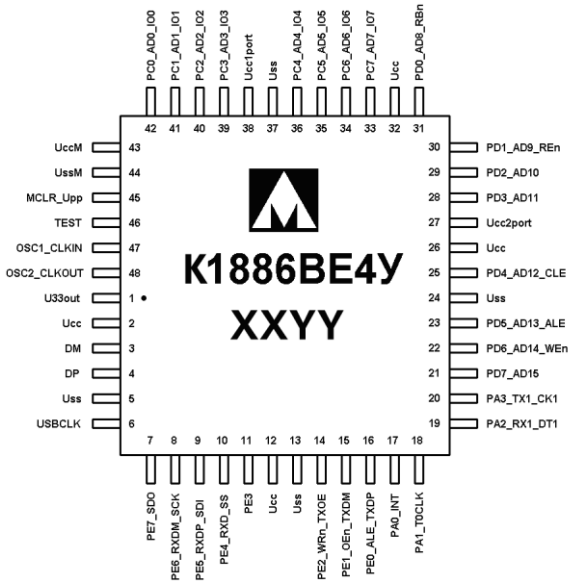
Однокристальная микро-ЭВМ с ЭСППЗУ (Flash-типа) с интерфейсом USB

1886ВЕ4У, К1886ВЕ4У, К1886ВЕ4Н4



XX- год выпуска
YY- неделя выпуска

Основные параметры микросхемы

- Тактовая частота до 33 МГц;
- Минимальная длительность цикла выполнения команды от 121 нс;
- 58 однословных инструкций (коды инструкций 16-ти разрядные);
- 8-ми разрядное АЛУ;
- Параллельные: 16-ти разрядная шина команд и 8-ми разрядная шина данных;
- Инструкция 8 х 8 битного аппаратно реализованного умножения;
- Поддержка прямого, косвенного и относительного режимов адресации
- Наличие инструкций одновременно работающих с двумя регистрами;
- Все команды (включая команду 8 х 8 битное умножение) выполняются за один цикл (от 121нс), кроме команд ветвления и чтения/записи таблиц, выполняемых за два цикла;
- Температурный диапазон:

Обозначение	Диапазон
1886ВЕ4У	минус 60 – 85 °С
К1886ВЕ4У	минус 60 – 85 °С
К1886ВЕ4УК	0 –70 °С

Тип корпуса:

- 48-и выводной металлокерамический корпус Н16.48-1В;
- микросхемы К1886ВЕ4Н4 поставляются в бескорпусном исполнении.

- 4-х векторный контроллер прерываний поддерживающий 18 источников прерываний (внешних и внутренних);
- 16-ти уровневый аппаратный стек;
- Возможность работы только с внутренней, с внутренней и внешней и только с внешней памятью программ (режимы: микроконтроллер, расширенный микроконтроллер и микропроцессор);
- Суммарный адресуемый объем памяти программ 128 КБайт (64К х 16 бит);
- Объем внутренней памяти программ 64 КБайта (32К х 16 бит);
- Внутренняя память программ перепрограммируемая FLASH типа;
- Объем внутренней памяти данных 902 байта;

- Все регистры «специального назначения» находятся в адресном пространстве памяти данных;
- До 28 универсальных линий ввода/вывода с индивидуальной настройкой направления и высокой нагрузочной способностью;
- 16-ти разрядный таймер/счетчик с 8-ми разрядным программируемым предделителем (таймер 0);
- Один универсальный синхронно-асинхронный приемопередатчик (USART), с программируемой скоростью передачи информации;
- Универсальный аппаратно реализованный контроллер и аналоговый приемопередатчик интерфейса USB 1.1, с четырьмя пользовательскими оконечными точками и со скоростью передачи до 12 Мбит/с;
- Последовательный синхронный порт с режимом последовательного периферийного интерфейса SPI;
- Универсальный контроллер внешней памяти типа NAND FLASH;
- Встроенное запоминающее устройство типа EEPROM объемом 256 байт;
- Микроконтроллер имеет систему сброса по включению и снижению напряжения питания (с таймерами отсрочки включения и запуска тактового генератора);
- Сторожевой таймер с собственным RC генератором на кристалле, для обеспечения высоконадежной защиты от сбоев;
- Защищенный режим (защита кода программы);
- Последовательный режим программирования памяти программ, возможность внутрисхемного программирования;
- Поддерживает режим энергосбережения SLEEP (с возможностью выхода с помощью внешних и внутренних прерываний и сброса);
- Четыре режима работы встроенного тактового генератора (RC генератор, использование в качестве тактовой частоты деленной на два частоты генератора USB контроллера, XT – стандартный генератор с кварцевым резонатором, EC – вход внешнего тактового сигнала);
- Встроенный регулятор напряжения на 3.3 вольта (ток до 40 мА);
- Диапазон напряжения питания ядра микроконтроллера от 4.5 до 5.5 вольт;
- Диапазон напряжения питания пользовательских выводов микроконтроллера от 3 до 5.5 вольт.

Общее описание и области применения микросхемы

Микросхема 1886ВЕ4У, К1886ВЕ4У, К1886ВЕ4Н4 (далее 1886ВЕ4У) - высокопроизводительный 8-ми разрядный RISC микроконтроллер с Гарвардской архитектурой. Он предназначен для однокристальной реализации систем передачи, обработки и хранения данных, в том числе с применением внешней энергонезависимой памяти типа NAND Flash, использующих USB интерфейс. Может использоваться для организации малопроизводительных вычислительных систем и в качестве устройства совмещения различных типов интерфейсов.

Основные области применения:

- устройства сопряжения интерфейсов;
- системы хранения и передачи данных;
- малопроизводительные вычислительные устройства;
- промышленные системы управления;
- измерительное оборудование.

Содержание

Общее описание и области применения микросхемы ... 3
Описание выводов ... 7
Структурная блок-схема микросхемы ... 10
Описание функционирования микросхемы ... 11
Встроенный тактовый генератор ... 11
Использование кварцевого или керамического резонатора ... 11
Внешний тактовый генератор ... 12
Режим RC генератора ... 13
Схема сброса микроконтроллера ... 14
Сброс по включению питания ... 15
Таймер включения питания PWRT ... 16
Таймер запуска генератора ... 16
Последовательность удержания микроконтроллера в состоянии сброса ... 17
Сброс по снижению напряжения питания ... 21
Прерывания ... 22
Регистр состояния прерываний (INTSTA) ... 23
Регистры разрешения периферийных прерываний PIE1 ... 25
Регистры запроса периферийных прерываний PIR1 ... 26
Обработка прерываний ... 27
Прерывание от вывода PA0/INT ... 27
Прерывание от вывода PA1/T0CLK ... 28
Периферийные прерывания ... 28
Сохранение регистров при прерывании ... 29
Организация памяти ... 31
Память программ ... 32
Память данных ... 34
Регистры общего назначения (GPR) ... 34
Регистры специального назначения (SFR) ... 34
Функционирование стека ... 43
Косвенная адресация ... 43
Регистры для чтения/записи таблиц ... 45
Модуль счетчика команд ... 45
Регистр выбора банка (BSR) ... 47
Считывание и запись таблиц данных ... 47
Запись таблиц во внутреннюю память ... 48
Запись таблиц во внешнюю память ... 50
Чтение таблиц ... 50
Аппаратный умножитель ... 52
Порты ввода-вывода ... 55
Регистр порта A и регистр направления данных DDRA ... 55
Регистр порта C и регистр направления данных DDRC ... 55
Регистр порта D и регистр направления данных DDRD ... 56
Регистр порта E и регистр направления данных DDRE ... 57
Блок «таймер 0» ... 58
Модуль синхронного последовательного порта SPI ... 61
Функционирование модуля SPI ... 64
Настройка выводов при работе с модулем SPI. Типовое включение ... 65
Работа модуля SPI в режиме ведущего ... 66

Работа модуля SPI в режиме ведомого ... 67
Выбор синхронного режима ведомого ... 67
Функционирование в режиме покоя (Sleep режим). Эффект сброса. ... 68
Модуль универсального синхронно-асинхронного приемопередатчика ... 69
Генератор скорости передачи данных ... 72
Асинхронный режим ... 73
Асинхронный передатчик ... 73
Асинхронный приемник ... 74
Синхронный ведущий режим ... 76
Передача данных в синхронном ведущем режиме ... 76
Прием данных в синхронном ведущем режиме ... 77
Синхронный ведомый режим ... 78
Передача данных в синхронном ведомом режиме ... 78
Прием данных в синхронном ведомом режиме ... 79
Блок внутренней EEPROM памяти данных ... 79
Регистры управления блоком внутренней EEPROM памяти данных ... 80
Работа блока по стиранию, записи и чтению данных ... 82
Включение EEPROM памяти ... 82
Стирание всей EEPROM памяти ... 83
Запись всей EEPROM памяти одним значением ... 83
Стирание строки EEPROM памяти ... 83
Запись байта в EEPROM память ... 84
Чтение байта из EEPROM памяти ... 84
Описание блока контроллера внешней NAND Flash памяти ... 85
Описание программного доступа к NAND Flash памяти ... 86
Запись команды в NAND Flash память ... 86
Запись адреса в NAND Flash память ... 88
Стирание страницы NAND Flash памяти ... 88
Запись данных в NAND Flash память ... 89
Чтение данных из NAND Flash памяти ... 89
Регистры управления контроллером NAND Flash памяти ... 90
Контроллер USB интерфейса ... 91
Описание принципов работы и функционирования контроллера USB интерфейса ... 92
Работа с контроллером USB интерфейса ... 94
Регистр функционального адреса ... 94
Регистр статуса ошибок ... 95
Регистр статуса блока ... 96
Регистр управления блока ... 97
Регистры разрешения прерываний от блока USB ... 97
Регистры статуса оконечной точки ... 101
Регистры количества слов в очереди оконечной точки ... 101
Регистры конфигурации оконечных точек ... 102
Регистр данных оконечных точек. ... 103
Регистры адреса поля дескриптора и данных дескриптора ... 103
Автоматическая инициализация USB интерфейса ... 104
Задание пользовательского описателя USB устройства ... 105
Режим передачи IN ... 112
Режим передачи OUT ... 113
Тип обмена: Interrupt Transaction ... 114
Тип обмена: Bulk Transaction ... 115
Тип обмена: Isochronous Transaction ... 115

Специальные модули микроконтроллера 116
Регистры конфигурации микроконтроллера 116
Внутрисхемное программирование микроконтроллера 118
Сторожевой таймер 118
Режим энергосбережения (SLEEP) 119
Регулятор питания на 3.3 вольта 120
Подключение напряжения питания 120
Система команд 120
Предельные и предельно-допустимые режимы работы 130
Электрические параметры микросхемы 133
Типовые зависимости 136
Габаритный чертеж микросхемы 140
Информация для заказа 142
Лист регистрации изменений 143

Описание выводов

Таблица 1

Номер вывода корпуса	Контактная площадка кристалла	Обозначение вывода	Тип вывода	Тип входного буфера	Назначение выводов
47	51	OSC1_CLKIN	вход	ST	Вход генератора в режимах RC генератора и генератора с кварцевым резонатором. Вход внешнего тактового сигнала
48	52	OSC2_CLKOUT	выход	-	Выход генератора. Подключается к кварцевому резонатору (в соответствующем режиме генератора). В режимах RC генератора и внешнего тактового сигнала - выход сигнала с частотой Fc/4 (частота циклов команд)
Порт А - параллельный двунаправленный порт ввода/вывода					**Дополнительное назначение выводов:**
17	17	PA0/INT	вход	ST	Входной порт общего назначения. PA0 может использоваться как вход внешнего прерывания. Можно настроить: фронт или спад сигнала вызовет прерывание. Только входной контакт
18	18	PA1/T0CLK	вход	ST	Входной порт общего назначения. PA1 может использоваться как вход внешнего прерывания. Можно настроить: фронт или спад сигнала вызовет прерывание. PA1 - может быть выбран как вход тактового сигнала для таймера 0. Только входной контакт
19	19	PA2/RX/DT	вход/выход(1)	ST	Двунаправленный порт общего назначения, совмещенный с линией RX/DT модуля USART. Режим вывода доступен только из USART
20	20	PA3/TX/CK	вход/выход(1)	ST	Двунаправленный порт общего назначения, совмещенный с линией TX/CK модуля USART. Режим вывода доступен только из USART
Порт С - параллельный двунаправленный порт ввода/вывода.					**Дополнительное назначение выводов:**
42	45	PC0/AD0/IO0	вход/выход	TTL	1. 8-ми разрядная шина данных IO для обращения к внешней микросхеме NAND Flash памяти. 2. Младший байт 16-ти битной системной шины в режиме микропроцессора или режиме расширенного микроконтроллера. Для реализации системной шины эти выводы могут являться выходами адреса и входами/выходами данных
41	44	PC1/AD1/IO1	вход/выход	TTL	
40	43	PC2/AD2/IO2	вход/выход	TTL	
39	42	PC3/AD3/IO3	вход/выход	TTL	
36	39	PC4/AD4/IO4	вход/выход	TTL	
35	38	PC5/AD5/IO5	вход/выход	TTL	
34	37	PC6/AD6/IO6	вход/выход	TTL	
33	36	PC7/AD7/IO7	вход/выход	TTL	

Номер вывода корпуса	Контактная площадка кристалла	Обозначение вывода	Тип вывода	Тип входного буфера	Назначение выводов
Порт D - параллельный двунаправленный порт ввода/вывода.					**Дополнительное назначение выводов:**
31	33	PD0/AD8/nRB	вход/выход	TTL	Шина управления RBn для обращения к внешней микросхеме NAND Flash памяти
30	30	PD1/AD9/nRE	вход/выход	TTL	Шина управления REn для обращения к внешней микросхеме NAND Flash памяти
29	29	PD2/AD10	вход/выход	TTL	Нет
28	28	PD3/AD11	вход/выход	TTL	Нет
25	25	PD4/AD12/CLE	вход/выход	TTL	Шина управления CLE для обращения к внешней микросхеме NAND Flash памяти
23	23	PD5/AD13/ALE	вход/выход	TTL	Шина управления ALE для обращения к внешней микросхеме NAND Flash памяти
22	22	PD6/AD14/nWE	вход/выход	TTL	Шина управления WEn для обращения к внешней микросхеме NAND Flash памяти
21	21	PD7 (AD15)	вход/выход	TTL	Нет
Порт E - параллельный двунаправленный порт ввода/вывода.					**Дополнительное назначение выводов:**
16	16	PE0/ALE/TXDP	вход/выход	TTL	1. В режиме микропроцессора или расширенного микроконтроллера PE0 становится выходом сигнала фиксации адреса внешней памяти (ALE). Адрес должен быть зафиксирован на спаде сигнала ALE. 2. Сигнал tx_dp контроллера USB в режиме тестирования 2 (USB TEST2)
15	15	PE1/OE/TXDM	вход/выход	TTL	1. В режиме микропроцессора или расширенного микроконтроллера PE1 становится сигналом управления, для чтения данных из внешней памяти (активный уровень - низкий). 2. Сигнал tx_dm контроллера USB в режиме тестирования 2 (USB TEST2)
14	14	PE2/WR/TXOE	вход/ыход	TTL	1. В режиме микропроцессора или расширенного микроконтроллера PE2 становится сигналом разрешения записи во внешнюю память (активный уровень - низкий). 2. Сигнал tx_oe контроллера USB в режиме тестирования 2 (USB TEST2)
11	11	PE3	вход/выход	TTL	Нет
10	10	PE4/RXD/SS	вход/выход	TTL	Сигнал rx_d контроллера USB в режиме тестирования 2 (USB TEST2). Сигнал Slave Select интерфейса SPI
9	9	PE5/RXDP/SDI	вход/выход	TTL	Сигнал rx_dp контроллера USB в режиме тестирования 2 (USB TEST2). Сигнал

Номер вывода корпуса	Контактная площадка кристалла	Обозначение вывода	Тип вывода	Тип входного буфера	Назначение выводов
					Serial Data Input интерфейса SPI
8	8	PE6/RXDM/SCK	вход/выход	TTL	Сигнал rx_dm контроллера USB в режиме тестирования 2 (USB TEST2). Сигнал Serial Clock интерфейса SPI
7	7	PE7/SDO	вход/выход	TTL	Сигнал Serial Data Output интерфейса SPI
6	6	USBCLK	вход	ST	Тактовая частота для контроллера USB
4	4	DP	вход/выход	-	Сигнал D+ шины USB
3	3	DM	вход/выход	-	Сигнал D- шины USB
2, 12, 26, 32	2, 12, 26, 35	U_{CC}	Напряжение питания	-	Напряжение питания ядра микроконтроллера
38	41	$U_{cc1port}$	Напряжение питания	-	Напряжение питания порта C
27	27	$U_{cc2port}$	Напряжение питания	-	Напряжение питания порта D
43	46	U_{CCM}	Напряжение питания	-	Питание внутренней Flash-памяти
5, 13, 24, 37	5, 13, 24, 40	U_{SS}	напряжение питания	-	Общий
44	47	U_{SSM}	Напряжение питания	-	Общий вывод Flash-памяти
46	49	TEST	вход	ST	Управляющий вход для перехода в тестовый режим. Для обычного режима работы должен быть подключен к USS
45	48	nMCLR/U_{PR}	вход/напряжение питания	ST	Вход внешнего «сброса» микроконтроллера с активным низким уровнем сигнала. Вход напряжения программирования
1	1	U_{33out}	напряжение питания	-	Выход внутреннего регулятора напряжения (3.3 вольта ток до 40 мА)
-	31, 32, 34, 50	-	-	-	не используются

Структурная блок-схема микросхемы

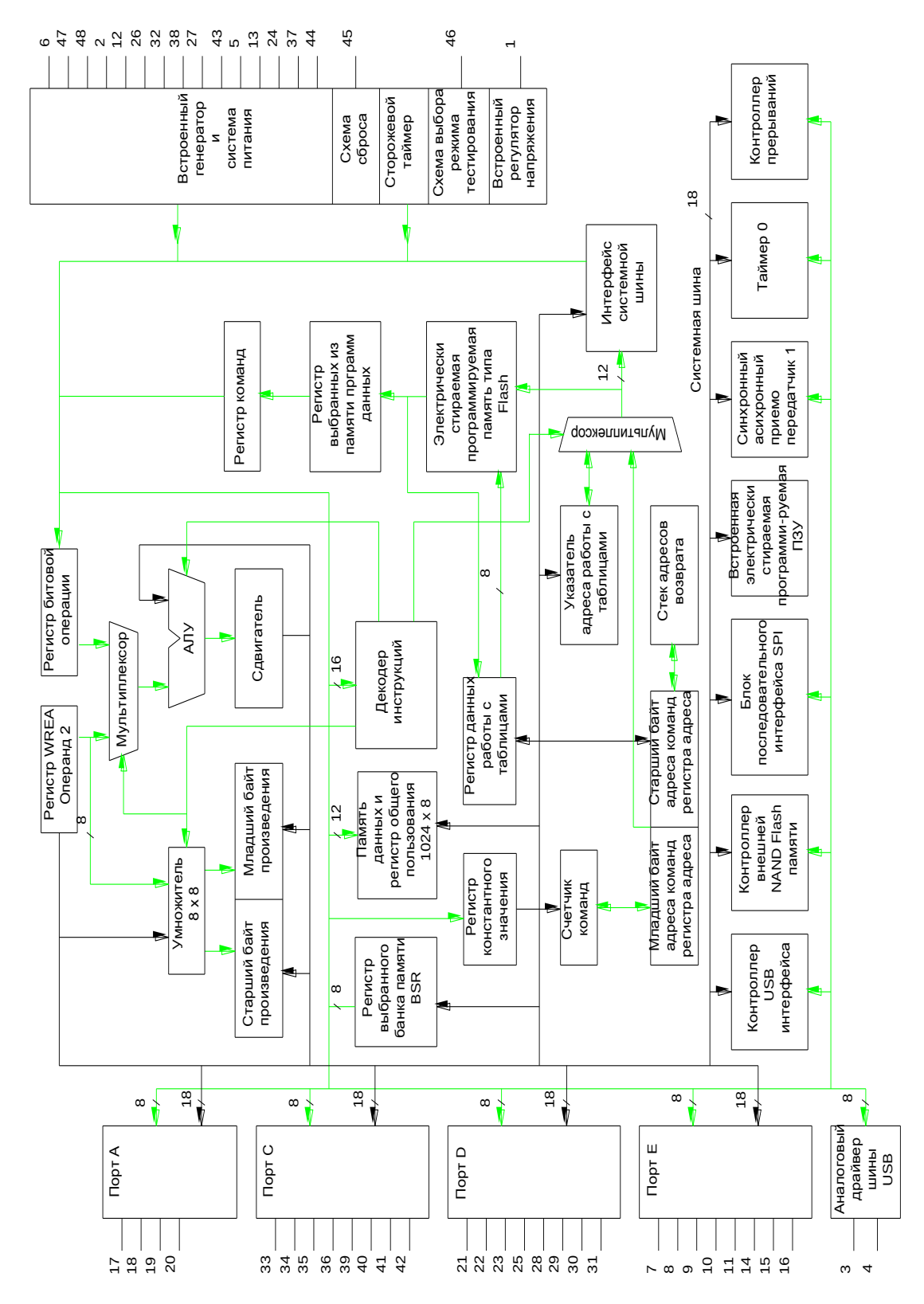


Рис. 1 Структурная блок-схема

Описание функционирования микросхемы

Встроенный тактовый генератор

Микроконтроллер использует две тактовых частоты:

- первая - для микропроцессорного ядра и периферийных модулей,
- вторая - для контроллера USB.

В соответствии со спецификацией интерфейса USB 1.1 тактовая частота для контроллера USB должна быть 48 МГц при работе в режиме Full Speed, и 6 МГц при работе в режиме Low Speed. Данная частота должна подаваться от внешнего генератора на вывод USBCLK.

Тактовая частота для микропроцессорного ядра и остальной периферии может варьироваться в пределах до 33 МГц. И может как подаваться от внешнего генератора так и формироваться внутри микроконтроллера. Генератор для формирования тактовых сигналов содержится на кристалле микроконтроллера. Четыре периода тактовых сигналов генератора составляют цикл выполнения команды.

Генератор микроконтроллера может работать в четырех режимах. Режимы выбираются программированием двух битов конфигурации FOSC1 и FOSC0 при программировании микроконтроллера. Возможен выбор следующих режимов:

- LF - режим работы микроконтроллера на тактовой частоте равной S частоты контроллера USB;
- XT - генератор с внешним кварцевым или керамическим резонатором (частота до 33 МГц);
- EC - режим подачи внешнего тактового сигнала (конфигурация генератора по умолчанию);
- RC - RC генератор с частотой до 4 МГц (подключается внешняя частотозадающая RC цепочка).

При выполнении команды SLEEP тактовый генератор выключается уменьшая потребляемый ток. Состояние внутреннего тактового сигнала соответствует такту Q1.

При поступлении сигнала «сброс» от вывода MCLR при нормальной работе микроконтроллера тактовый генератор не выключается.

Использование кварцевого или керамического резонатора

В режиме тактового генератора XT, кварцевый или керамический резонатор подсоединяется к выводам OSC1_CLKIN и OSC2_CLKOUT. Генератор требует использования кварцевых резонаторов с параллельным резонансом. Использование резонаторов с последовательным резонансом может привести к получению тактовой частоты не соответствующей параметрам резонатора. Для частот превышающих 24МГц, кварцевый резонатор обычно работает на частоте гармоники. В этом случае требуется резонансный контур, чтобы уменьшить усиление на частоте основной гармоники. На Рис. 1 показаны примеры схем подключения резонаторов.

При включении напряжения питания, тактовый генератор начнет генерацию сигнала. Время необходимое для запуска генератора зависит от большого количества факторов. В их число входят: частота резонатора, емкость используемых конденсаторов (C1 и C2), скорость нарастания напряжения питания, рабочая температура, сопротивление резистора, если он подключен, режим

тактового генератора (который выбирает коэффициент усиления внутреннего инвертора). Напряжение полного размаха выхода тактового генератора может быть достаточно малым (менее 50% от U_{CC}) пока временная диаграмма тактового сигнала центрируется к $U_{CC}/2$.

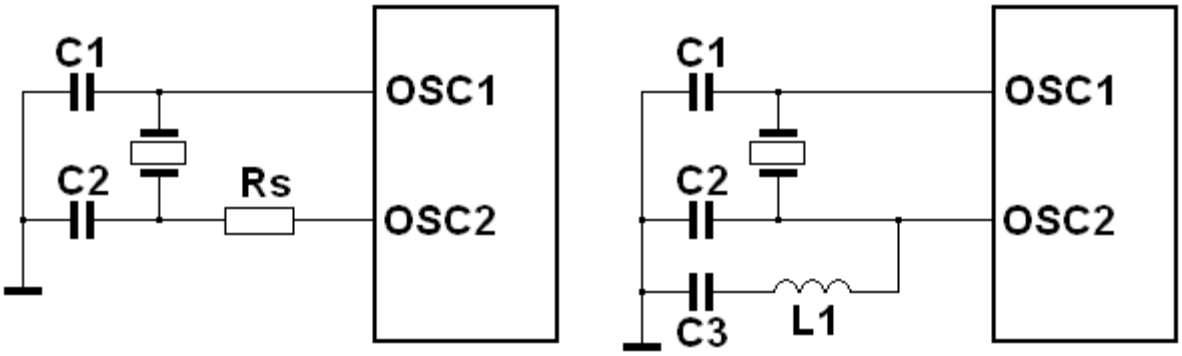


Рис. 1 Подключение резонатора.
Схема справа - для резонатора, работающего на гармониках.

Примечания:

- Резистор R_S может потребоваться для некоторых типов резонаторов.
- Параллельный резонансный контур L_1C_2 отфильтровывает основную частоту: $(2pf)^2 = 1/(L_1*C_2)$.
- C_3 (0.1 мкф) препятствует протеканию постоянного тока на землю.
- Для резонаторов необходимы внешние конденсаторы C_1 и C_2 (смотрите справочные параметры). При расчете емкости конденсатора необходимо учитывать емкость печатной платы. Большая емкость увеличивает стабильность генератора, но увеличивается время запуска и ток генератора.

Внешний тактовый генератор

В режиме внешнего генератора (EC), на ввод OSC1 может быть подан внешний тактовый сигнал с КМОП уровнями. В этом режиме, вход OSC1_CLKIN имеет высокое входное сопротивление, а вывод OSC2_CLKOUT является выходом CLKOUT (Fc/4). В качестве генераторов могут быть использованы готовые модули генераторов, обеспечивающие широкий набор тактовых частот и стабильные параметры.

Для тактирования микроконтроллера можно использовать внешний генератор тактирования контроллера USB (режим LF). Вывод OSC1 в этом случае может быть не подсоединен. На вход USBCLK должна подаваться тактовая частота для работы USB в заданном режиме. Функционирование микропроцессорного ядра будет осуществляться на частоте USBCLK деленной в два раза. Если контроллер USB не используется, то вход USBCLK должен подсоединяться к логическому нулю.

В режиме внешнего генератора (EC), при работе контроллера USB в режиме Low Speed, тактовая частота 6 МГц может одновременно подаваться на входы USBCLK и OSC1_CLKIN.

Режим RC генератора

В приложениях, не требующих высоко стабильной тактовой частоты, может быть использован режим RC генератора, который позволяет уменьшить стоимость устройства. Частота RC генератора зависит от напряжения питания, значения подключенных внешних сопротивления и емкости, и от рабочей температуры. Дополнительно частота будет варьироваться в некоторых пределах из-за технологического разброса параметров кристалла. Также на частоту будут влиять емкости между выводами корпуса и дорожками печатной платы, особенно при малых значениях емкости внешнего конденсатора. Необходимо учитывать и технологический разброс параметров внешних компонентов R и C. На
Рис. **2** показана схема подключения RC цепочки к микроконтроллеру. Если сопротивление резистора меньше 2.2 кОм частота тактового генератора может быть нестабильной или генерация может прекратиться. Для очень большого сопротивления резистора (более 1 МОм) генератор тактового сигнала становится чувствителен к внешним помехам, влажности и утечки тока. Поэтому, рекомендуется выбирать величину сопротивления резистора от 3 до 100 кОм. Генератор может работать без внешнего конденсатора, но рекомендуется использовать конденсатор с емкостью более 20пФ для стабильной работы генератора. Без внешнего конденсатора, или если конденсатор имеет очень малую емкость, частота генератора может варьироваться из-за изменений во внешних емкостях, таких как емкости проводников печатной платы или выводов компонентов.

В режиме RC генератора на выводе OSC2_CLKOUT формируется тактовый сигнал с частотой Fc/4.

Генератор в режиме RC начинает формировать тактовый сигнал сразу после достижения напряжением питания порогового уровня. Время запуска RC генератора зависит от ряда факторов: сопротивления резистора, емкости конденсатора, скорости нарастания напряжения питания, температуры и т.д.

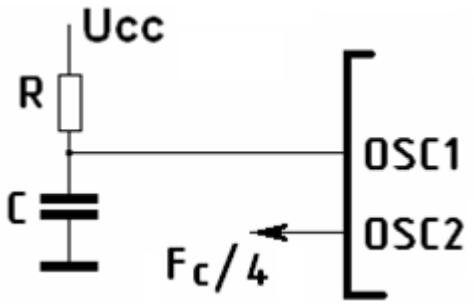


Рис. 2 Схема включения в режиме RC генератора

Выход тактового сигнала разделяется на 4 непересекающихся квадратурных тактовых сигнала, именуемых Q1, Q2, Q3, Q4. Программный счетчик (PC) увеличивается в такте Q1, а выборка команды из программной памяти и сохранение ее в регистре команд происходит по синхросигналу Q4. Команда декодируется и выполняется в течение циклов Q1-Q4. Тактовые сигналы и выполнение команд показано на
Рис. **3**.

Цикл выполнения команды состоит из четырех Q циклов (Q1, Q2, Q3, Q4). Выборка и выполнение команд происходят конвейерным способом, т.е. выборка одной команды использует тот же цикл, что и декодирование и выполнение другой команды. Благодаря конвейерной обработке команд, каждая инструкция выполняется за один цикл. Если команда изменяет счетчик команд (команды

ветвления), то для выполнения команды требуется два цикла, так как необходимо удалить выбранную команду из конвейера (см. **Рис. 1**4). Во время удаления выбирается новая команда, и затем она выполняется.

Цикл выборки команды начинается с приращения счетчика команд в такте Q1. В цикле выполнения команды, код загруженной команды помещается в регистр команд IR на такте Q1. Декодирование и выполнение команды происходит в тактах Q2, Q3, Q4. Операнд из памяти данных читается в такте Q2, а результат выполнения команды записывается в такте Q4.

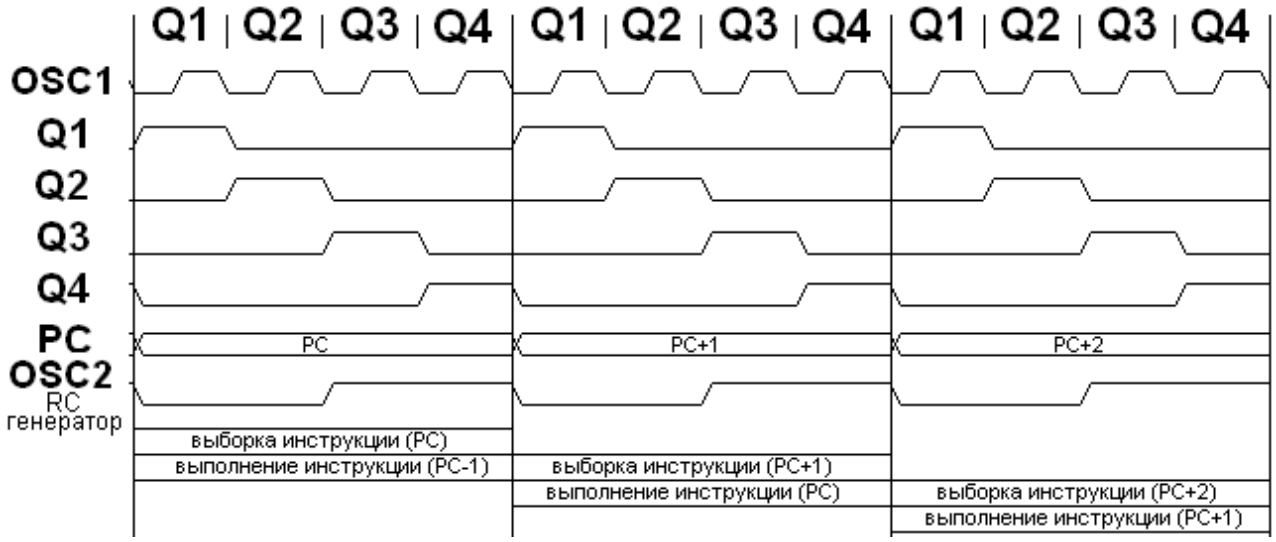


Рис. 3 Синхронизация выполнения команды

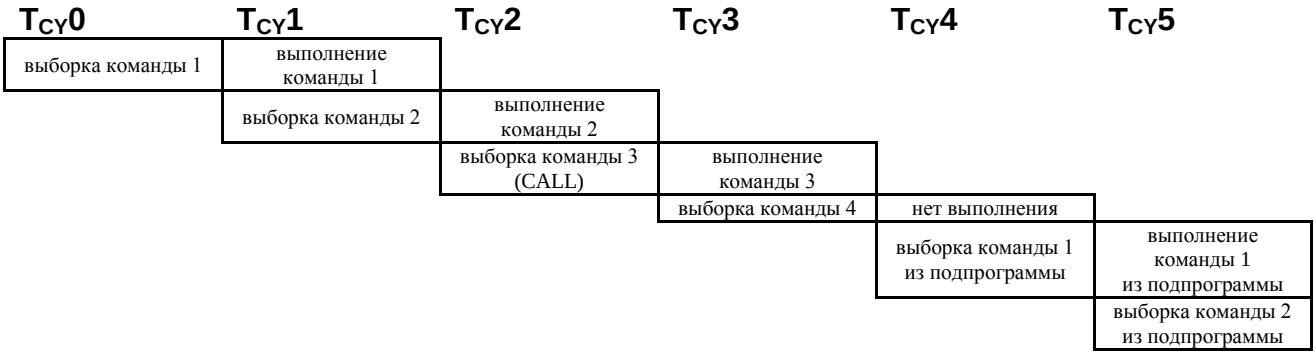


Рис. 4 Конвейерное выполнение команд

Схема сброса микроконтроллера

Микроконтроллеры различают следующие виды сброса:

- сброс по включению питания;
- сброс по снижению напряжения питания;
- сброс по внешнему сигналу MCLR;
- сброс по переполнению сторожевого таймера.

Некоторые регистры не изменяются после сброса: после сброса по включению питания они содержат неизвестное значение, а после любого другого сброса их состояние остается неизменным. Большинство других регистров переводятся в определенное состояние по сбросу. Биты TO и PD принимают

определенные значения при различных видах сброса, как показано в Таблица 3. Эти биты в соединении с битами POR и BOR, используются в программном обеспечении для определения вида сброса. В Таблица 5 представлено описание всех видов сброса для всех регистров.

При поступлении сигнала «сброс» регистры направления передачи сигналов (DDR) устанавливаются в «1», переводя выводы портов в состояние высокоимпедансных входов. Состояние сброса некоторых периферийных модулей может перевести выводы портов в другие состояния, например, такие как аналоговые входы или системная шина.

В состоянии «сброс» выход внутреннего тактового сигнала соответствует такту Q1. Если микроконтроллер находится в режиме «расширенного микроконтроллера» или «микропроцессора», то во время «сброса» на выводе PE0/ALE будет присутствовать низкий логический уровень выходного сигнала, а на PE1/OE и PE2/WR высокий уровень.

Упрощенная блок-схема схемы сброса на чипе (см. Рис. **5**):

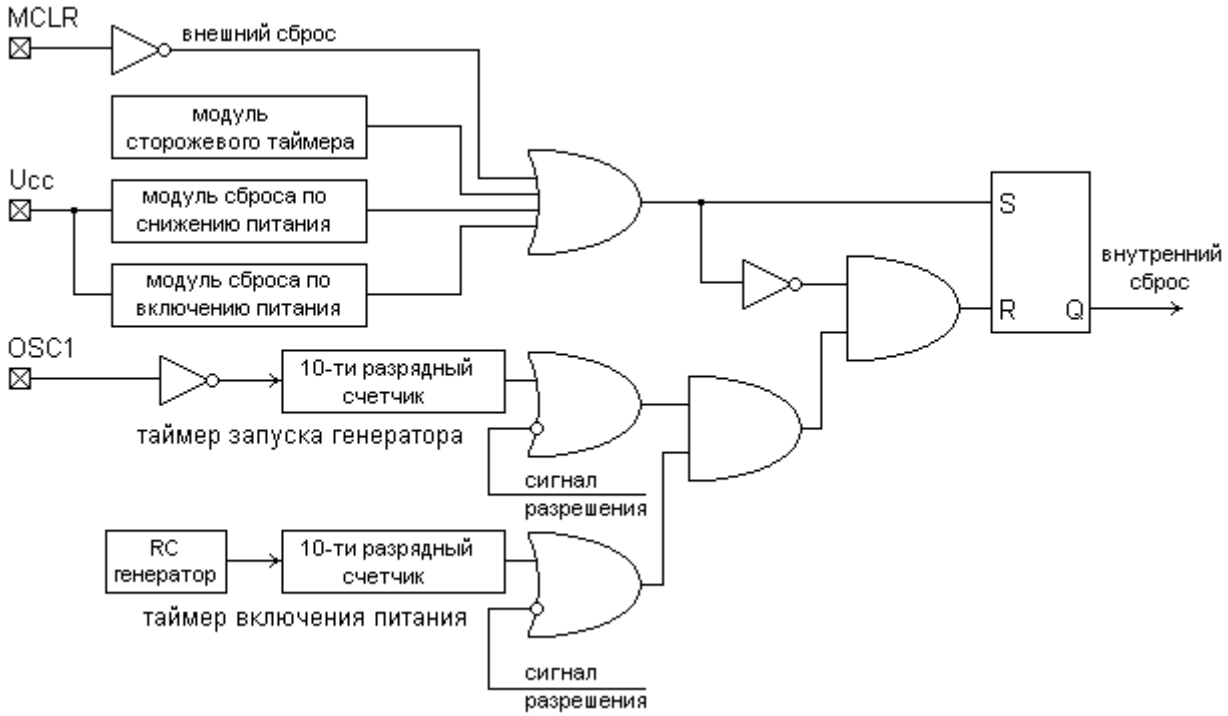


Рис. 5 Упрощенная блок-схема встроенной схемы сброса

Сброс по включению питания

Схема сброса по включению питания удерживает микроконтроллер в состоянии «сброс» до тех пор, пока напряжение питания не достигнет определенного уровня (примерно 1.4 – 2.3 В). Благодаря этой схеме, в ряде приложений можно обойтись без внешней RC цепочки, подключаемой к выводу MCLR. В этом случае вывод MCLR подключается через резистор или напрямую к напряжению питания. Внешняя схема «сброса» (см. Рис. **6**) потребуется только в случае низкой скорости нарастания напряжения питания.

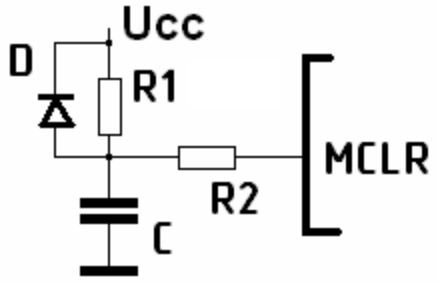


Рис. 6 Внешняя схема «сброса» по включению питания

Примечание:

Диод D предназначен для быстрой разрядки конденсатора при снижении напряжения питания. Сопротивление резистора R1 рекомендуется выбирать не более 40 кОм (чтобы падение напряжения на резисторе, из-за токов утечки вывода MCLR_Upp, не превышало 0.2 В). Резистор R2 предназначен для ограничения тока через вывод MCLR_Upp, рекомендуемая величина 100 Ом – 1 кОм.

Таймер включения питания PWRT

Таймер включения питания обеспечивает задержку включения (номинальное значение 96 мс) по сигналу схемы сброса включения питания. Это происходит после фронта внутреннего сигнала «сброса», или после фронта сигнала MCLR. Таймер включения питания работает на внутреннем RC генераторе. В течение этого времени микроконтроллер удерживается в состоянии сброса. В большинстве случаев эта задержка позволяет напряжению питания достигнуть номинального значения. Время задержки варьируется от микроконтроллера к микроконтроллеру и зависит от величины напряжения питания и температуры. Смотрите таблицу параметров.

Таймер запуска генератора

Таймер запуска генератора обеспечивает дополнительную задержку в 1024 такта генератора после окончания задержки от таймера включения питания или выхода микроконтроллера из режима SLEEP в режимах XT или LF. Таймер включения питания и таймер запуска генератора работают последовательно. Таймер запуска генератора считает каждый импульс генератора на входе OSC1. Счетчик начинает инкрементироваться после того, как амплитуда сигнала генератора достигнет порога входного буфера. Задержка гарантирует стабилизацию частоты генератора с кварцевым резонатором прежде, чем устройство выйдет из режима сброса. Длительность задержки зависит от частоты резонатора.

На рисунке ниже показана работа схемы таймера запуска генератора (распределение времени при запуске генератора). На этом рисунке показан низкочастотный генератор, время запуска которого превышает задержку таймера по включению питания. На
Рис. **7**: T_{OSC1} - время, требуемое кварцевому генератору для запуска.

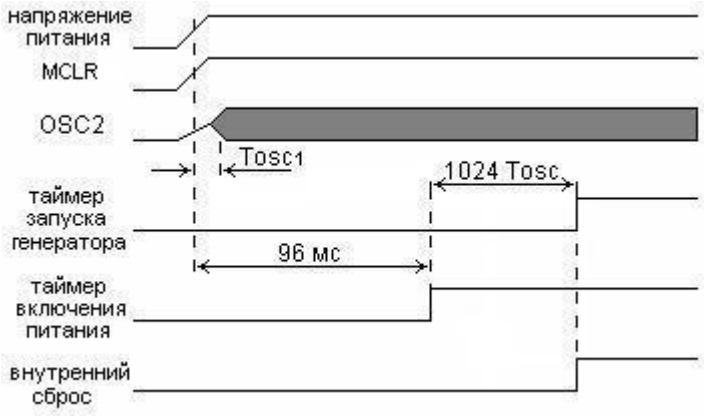


Рис. 7 Время запуска генератора

Последовательность удержания микроконтроллера в состоянии сброса

При включении питания выполняется следующая последовательность удержания микроконтроллера в состоянии сброса: внутренний сигнал «сброса» по включению питания увеличивается, пока не достигнет порогового уровня. Если сигнал MCLR находится в высоком уровне, то сначала начинает работать таймер включения питания, а после того, как он отсчитает положеное время, включается таймер запуска генератора, если MCLR в низком уровне то таймеры запускаются после фронта этого сигнала. Обычно задержка от таймера включения питания больше, за исключением низкочастотных кварцевых резонаторов. Общее время задержки также изменяется в зависимости от конфигурации генератора. Ниже показано время задержки, в зависимости от конфигурации генератора (см. Таблица 2). На
Рис. 8 отображены последовательности задержек.

Если напряжение питания устройства не соответствует спецификации электрических характеристик после окончания задержки от таймеров, то на выводе MCLR_Upp должен присутствовать низкий логический уровень, пока напряжение питания не достигнет номинального значения. Для большинства схем достаточно использования внешней RC цепочки.

Если сигнал сброса от MCLR подается во время нормальной работы микроконтроллера, то после его окончания таймеры включения питания и запуска генератора не работают, но запускается таймер задержки на обновление конфигурационных бит (типовое значение 1 мс).

Ниже (см. Таблица 2, Таблица 3) показаны состояния после сброса для некоторых битов и специальных регистров, в то время как (см. Таблица 45) показано состояние при инициализации для всех регистров.

Таблица 2

Время задержки при различных видах сброса

Конфигурация генератора	Включение или снижение напряжения питания	Выход из режима SLEEP	Сброс от MCLR
XT, LF	сумма 96 мс и 1024*T_{OSC}	1024*T_{OSC}	1 мс
EC, RC	сумма 96 мс и 1024*T_{OSC}	-	1 мс

Таблица 3

Биты статуса и их значение после «сброса»

POR	BOR (если разрешен сброс по снижению питания, иначе значение не известно)	TO	PD	Тип «сброса»
0	0	1	1	Сброс по включению питания
1	1	1	0	Выход из режима SLEEP по прерыванию (см. примечание)
1	1	0	1	Сброс от WDT при нормальном режиме работы (см. примечание)
1	1	0	0	Выход из режима SLEEP от WDT (см. примечание)
1	1	1	1	Сброс от MCLR (см. примечание)
1	0	1	1	Сброс по снижению напряжения питания
x	x	1	1	Выполнение команды CLRWDT

Примечание:

Для отмеченных видов «сброса», состояния битов статуса будут соответствовать приведенным в таблице, для случая если биты предварительно установлены в единицу.

Таблица 4

Условия сброса программного счетчика и регистра CPUSTA

Тип «сброса»		PCH:PCL	CPUSTA(3)	Задержка включения
Сброс по включению питания		0000h	--11 1100	Сумма 96 мс и $1024*T_{OSC}$
Сброс по снижению напряжения питания		0000h	--11 1110	Сумма 96 мс и $1024*T_{OSC}$
Сброс от MCLR в режиме нормальной работы		0000h	--11 1111(4)	1 мс
Сброс от MCLR в режиме SLEEP		0000h	--11 1111(4)	Большее из 1 мс или (только для режимов XT и LF) $1024*T_{OSC}$
Сброс от WDT в режиме нормальной работы		0000h	--11 0111(4)	Нет
Сброс от WDT во время режима SLEEP		0000h	--11 0011(4)	Для режимов XT и LF: $1024*T_{OSC}$
Выход из режима SLEEP по прерыванию	GLINTD установлен	PC + 1	--11 1011(4)	Для режимов XT и LF: $1024*T_{OSC}$
	GLINTD сброшен	PC + 1(1)	--10 1011(4)	Для режимов XT и LF: $1024*T_{OSC}$

Обозначения:

u = не изменяется, x = не известно, - = не реализовано, читается как «0»

Примечание:

1. При «пробуждении», выполняется эта команда. Далее команда выбирается в соответствии с вектором прерывания, а затем выполняется.
2. Программный счетчик = 0, то есть устройство переходит к вектору сброса и устанавливает регистры в состояние сброса по WDT.
3. Значение бита BOR известно только если разрешен сброс по снижению питания.
4. Состояние статусных битов соответствует приведенным в таблице для случая их предварительной установки в единицу.

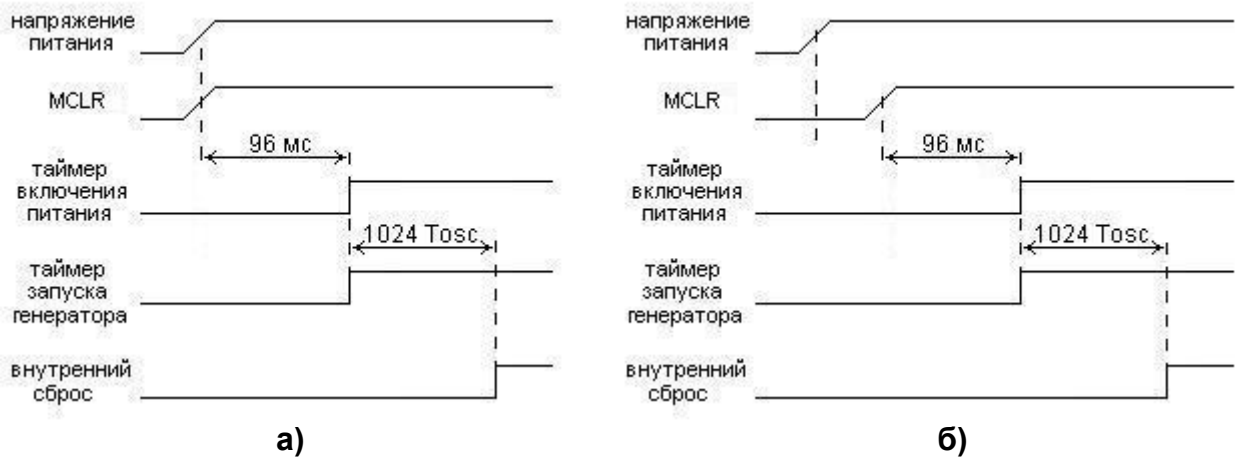


Рис. 8

Последовательность удержания в режиме сброса по включению питания:
а) MCLR подсоединен к напряжению питания
б) MCLR не подключен к напряжению питания

Таблица 5

Значения регистров при «сбросе»

Регистр	Адрес	Сброс по включению или снижению напряжения питания	Сброс от MCLR или от сторожевого таймера	Выход из режима SLEEP по прерыванию
Вне банка				
INDF0	00h	N/A	N/A	N/A
FSR0	01h	0000 0000	uuuu uuuu	uuuu uuuu
PCL	02h	0000h	0000h	PC+1(2)
PCLATH	03h	0000 0000	0000 0000	uuuu uuuu
ALUSTA	04h	1111 0000	1111 0000	1111 uuuu
T0STA	05h	0000 000-	0000 000-	0000 000-
CPUSTA(3)	06h	--11 11q0	--11 qquu	--uu qquu
INTSTA	07h	0000 0000	0000 0000	uuuu uuuu(1)
INDF1	08h	N/A	N/A	N/A
FSR1	09h	0000 0000	uuuu uuuu	uuuu uuuu
WREG	0Ah	0000 0000	0000 0000	uuuu uuuu
TMR0L	0Bh	0000 0000	0000 0000	uuuu uuuu
TMR0H	0Ch	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	0Dh	0000 0000	0000 0000	uuuu uuuu
TBLPTRH	0Eh	0000 0000	0000 0000	uuuu uuuu

BSR	0Fh	0000 0000	0000 0000	uuuu uuuu
Банк 0				
EP1_CFG1	10h	0000 0000	0000 0000	uuuu uuuu
EP1_CFG2	11h	0000 0000	0000 0000	uuuu uuuu
EP2_CFG1	12h	0000 0000	0000 0000	uuuu uuuu
EP2_CFG2	13h	0000 0000	0000 0000	uuuu uuuu
EP3_CFG1	14h	0000 0000	0000 0000	uuuu uuuu
EP3_CFG2	15h	0000 0000	0000 0000	uuuu uuuu
EP4_CFG1	16h	0000 0000	0000 0000	uuuu uuuu
EP4_CFG2	17h	0000 0000	0000 0000	uuuu uuuu
Банк 1				
DESC_ADR	10h	0000 0000	0000 0000	uuuu uuuu
DESC_DATA	11h	0000 0000	0000 0000	uuuu uuuu
PORTA	12h	---- xxxx	---- uuuu	uuuu qqqq
DDRD(5)	13h	1111 1111	1111 1111	uuuu uuuu
PORTD(4,5)	14h	xxxx xxxx	uuuu uuuu	uuuu uuuu
DDRC(5)	15h	1111 1111	1111 1111	uuuu uuuu
PORTC(4,5)	16h	xxxx xxxx	uuuu uuuu	uuuu uuuu
-	17h	---- ----	---- ----	---- ----
Банк 2				
EP1_REG	10h	0000 0000	0000 0000	uuuu uuuu
EP2_REG	11h	0000 0000	0000 0000	uuuu uuuu
EP3_REG	12h	0000 0000	0000 0000	uuuu uuuu
EP4_REG	13h	0000 0000	0000 0000	uuuu uuuu
PIR1	14h	--00 0010	--00 0010	--uu uuuu(1)
PIE1	15h	--00 0000	--00 0000	--uu uuuu
-	16h	---- ----	---- ----	---- ----
-	17h	---- ----	---- ----	---- ----
Банк 3				
EP1_STAT	10h	--00 --00	--00 --00	--uu --uu
EP2_STAT	11h	--00 --00	--00 --00	--uu --uu
EP3_STAT	12h	--00 --00	--00 --00	--uu --uu
EP4_STAT	13h	--00 --00	--00 --00	--uu --uu
EP1_CNT	14h	-000 0000	-000 0000	-uuu uuuu
EP2_CNT	15h	-000 0000	-000 0000	-uuu uuuu
EP3_CNT	16h	-000 0000	-000 0000	-uuu uuuu
EP4_CNT	17h	-000 0000	-000 0000	-uuu uuuu
Банк 4				
USB_ADR	10h	-000 0000	-000 0000	-uuu uuuu
USB_ERROR	11h	0000 0000	0000 0000	uuuu uuuu
USB_STAT	12h	-000 0000	-000 0000	-uuu uuuu
USB_CTRL	13h	--00 0000	--00 0000	--uu uuuu
USB_IE1	14h	0000 0000	0000 0000	uuuu uuuu
USB_IE2	15h	0000 ----	0000 ----	uuuu ----
USB_IE3	16h	0000 0000	0000 0000	uuuu uuuu
USB_IE4	17h	---- 0-00	---- 0-00	---- u-uu
Банк 5				
-	10h	---- ----	---- ----	---- ----
DDRE(5)	11h	1111 1111	1111 1111	uuuu uuuu
PORTE(4,5)	12h	xxxx xxxx	uuuu uuuu	uuuu uuuu

RCSTA	13h	0000 -000	0000 -000	uuuu -uuu
RCREG	14h	0000 0000	0000 0000	uuuu uuuu
TXSTA	15h	0000 --10	0000 --10	uuuu --uu
TXREG	16h	0000 0000	0000 0000	uuuu uuuu
SPBRG	17h	0000 0000	0000 0000	uuuu uuuu
Банк 6				
NAND_MODE	10h	0000 0000	0000 0000	uuuu uuuu
NAND_DATA	11h	0000 0000	0000 0000	uuuu uuuu
-	12h	---- ----	---- ----	---- ----
WRDIF	13h	0000 0101	0000 0101	uuuu uuuu
EERPOM_CONT	14h	q000 0000	q000 0000	uuuu uuuu
EERPOM_MODE	15h	0000 0000	0000 0000	uuuu uuuu
EERPOM_DATA	16h	0000 0000	0000 0000	uuuu uuuu
EERPOM_ADDR	17h	0000 0000	0000 0000	uuuu uuuu
Банк 7				
-	10h	---- ----	---- ----	---- ----
SSPCON1	11h	0000 0000	0000 0000	uuuu uuuu
-	12h	---- ----	---- ----	---- ----
SSPSTAT	13h	0000 0000	0000 0000	uuuu uuuu
SSPBUF	14h	0000 0000	0000 0000	uuuu uuuu
-	15h	---- ----	---- ----	---- ----
-	16h	---- ----	---- ----	---- ----
-	17h	---- ----	---- ----	---- ----
Вне банка				
PRODL	18h	0000 0000	0000 0000	uuuu uuuu
PRODH	19h	0000 0000	0000 0000	uuuu uuuu

Обозначение:
u = не изменяется, x = неизвестно, - = не реализовано, читается как «0»,
q = значение зависит от условия.
Примечание:
1. Один бит или более в INTSTA, PIR1, будет изменен (чтобы произошел выход).
2. Когда выход из режима SLEEP происходит по прерыванию и бит GLINTD сброшен, PC загружается вектором прерывания.

См. Таблица 4 для значений по сбросу в особых условиях.

3. Это значение, которое будет в триггере-защелке порта вывода.
4. Когда запрограммирован режим микропроцессора или расширенного микроконтроллера, работа этого порта не зависит от этих регистров.

Сброс по снижению напряжения питания

Микроконтроллеры имеют на кристалле схему сброса по снижению напряжения питания. Эта схема переводит микроконтроллер в режим сброса, когда напряжение питания опускается ниже установленного уровня, что гарантирует прекращение выполнения программ при выходе напряжения питания за установленные нормы. Прежде чем использовать схему сброса по снижению напряжения питания, проверьте электрические характеристики, чтобы удостовериться в том, что она отвечает вашим требованиям. Включение или выключение схемы сброса производится битом BODEN в слове конфигурации.

Работа схемы сброса: если напряжение питания опускается ниже U_{bor} (типовое значение 4.0 В), произойдет сброс по снижению напряжения питания. Микроконтроллер находится в состоянии сброса, пока напряжение питания не поднимется выше U_{bor}. Затем начинает работать таймер включения питания, далее таймер запуска генератора (для режимов LF и XT). Это удерживает микроконтроллер в состоянии сброса более 96 мс и 1024*T_C. Если напряжение питания опускается ниже Ubor во время работы таймера включения питания, то микроконтроллер возвращается в состояние сброса и таймеры инициализируются заново. После подъема питания, таймеры начнут отсчёт временной задержки. На Рис. **9** показаны типовые ситуации сброса.

В некоторых приложениях параметры внутренней схемы сброса по снижению питания не удовлетворяют требованиям. В этом случае должна быть применена внешняя схема сброса по снижению напряжения питания.

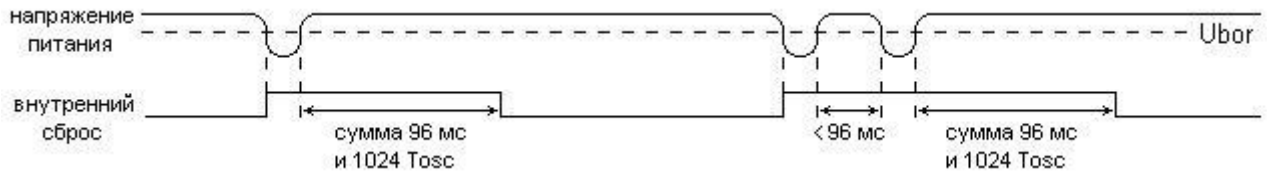


Рис. 9 Сброс по снижению напряжения питания

Прерывания

Микроконтроллеры имеют 29 источников прерывания:

- Внешнее прерывание от вывода PA0/INT,
- Переполнение таймера TMR0,
- Буфер передатчика USART пуст,
- Буфер приемника USART заполнен,
- Прерывания от модуля USB (21 источник прерываний),
- Прерывания от модуля SPI,
- Прерывания от модуля EERPOM,
- Прерывания от модуля NAND Flash,
- Изменение сигнала на выводе PA1/T0CLK (только в случае тактирования таймера TMRO от этого вывода).

Прерываниями управляют четыре регистра: CPUSTA, INTSTA, PIE1, PIR1.

Регистр CPUSTA содержит бит глобального запрещения прерываний GLINTD (Global Interrupt Disable). Если этот бит установлен, все прерывания запрещены. Этот бит является частью функциональных возможностей ядра микроконтроллера и описывается в разделе «Обработка прерываний».

Когда происходит прерывание, бит GLINTD автоматически устанавливается для запрета дальнейших прерываний, адрес возврата записывается в стек, а в PC загружается адрес вектора прерываний. Существует 4 вектора прерываний. Каждый адрес вектора прерываний предназначен для определенного источника прерываний (кроме периферийных прерываний, у которых один и тот же адрес). Эти источники следующие:

- Внешнее прерывание от вывода PA0/INT;
- Переполнение таймера TMR0;
- Изменение сигнала на выводе PA1/T0CLK;
- Любое периферийное прерывание.

Переход по адресу вектора прерывания **не сбрасывает флаг запроса прерывания**. Флаг запроса прерывания должен быть сброшен в программе обработки прерывания перед разрешением прерываний, чтобы предотвратить повторный переход на обработку прерываний. В программе обработки периферийных прерываний источник прерывания можно идентифицировать проверкой флагов запроса прерываний.

Когда выполняется условие прерывания, индивидуальные флаги запросов прерываний устанавливаются независимо от состояния бита GLINTD и соответствующих битов маски.

При внешнем прерывании происходит задержка прерывания. Для команд, выполняющихся за два машинных цикла, задержка длиннее на один машинный цикл.

Возврат из программы обработки прерываний производится по команде RETFIE. При выполнении команды происходит восстановление программного счетчика (PC) из стека и сбрасывается бит GLINTD (чтобы разрешить прерывания).

Регистр состояния прерываний (INTSTA)

Регистр INTSTA содержит флаги запроса прерываний и биты разрешений для не периферийных прерываний. Бит PEIF (флаг запроса периферийных прерываний) только читается, и объединяет по «ИЛИ» все не замаскированные флаги запросов периферийных прерываний в регистре PIR1.

Биты флагов запросов прерываний устанавливаются по заданным условиям, даже если соответствующий бит разрешения прерывания сброшен (прерывание запрещено), или бит GLINTD установлен (все прерывания запрещены).

Следует с осторожностью сбрасывать любой разрешающий бит регистра INTSTA, когда прерывания разрешены (бит GLINTD сброшен). Если какие-либо флаги запроса прерывания (T0IF, INTF, T0CKIF, или PEIF) устанавливаются в том же машинном цикле, в котором соответствующие биты разрешения прерывания сбрасываются, устройство переходит по адресу сброса (0x00). Прежде, чем запретить какое-либо прерывание, сбросом разрешающего бита регистра INTSTA, необходимо предварительно установить бит GLINTD, т.е. запретить все прерывания.

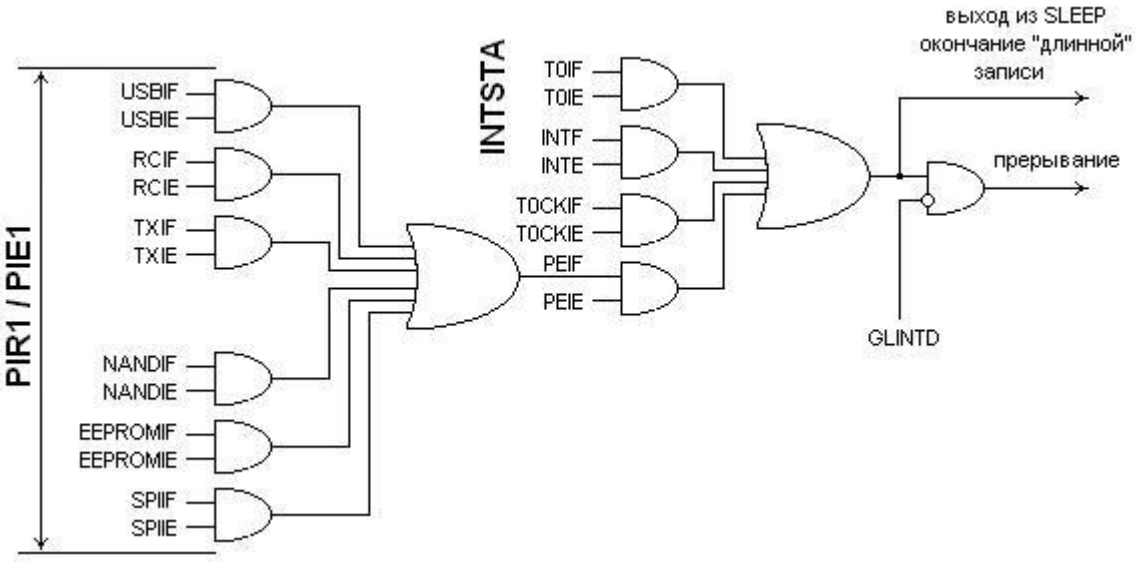


Рис. 10 Структурная схема логики прерываний

Таблица 6

Регистр INTSTA (адрес: 07h, не зависит от номера банка)

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PEIF	**T0CKIF**	**T0IF**	**INTF**	**PEIE**	**T0CKIE**	**T0IE**	**INTE**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**PEIF:** Флаг запроса периферийного прерывания. Этот бит является объединением по «ИЛИ» всех флагов запросов периферийных прерываний логически умноженных по «И» на соответствующие биты разрешения прерываний. Логика прерываний направляет выполнение программы по адресу 20h), когда происходит задержка периферийного прерывания. 1 = есть не обработанное периферийное прерывание; 0 = нет не обработанного периферийного прерывания
бит 6	**T0CKIF:** Флаг внешнего запроса прерывания от вывода PA1/T0CLK. Сбрасывается программно программой обработки прерывания. Адрес вектора 18h. 1 = на выводе PA1/T0CLK обнаружен заданный фронт сигнала; 0 = на выводе PA1/T0CLK не обнаружен заданный фронт сигнала
бит 5	**T0IF:** Флаг запроса прерывания по переполнению таймера 0. Сбрасывается программно программой обработки прерывания. Адрес вектора 10h. 1 = таймер TMR0 переполнен; 0 = таймер TMR0 не переполнен
бит 4	**INTF:** Флаг внешнего запроса прерывания от вывода PA0 /INT. Сбрасывается программно программой обработки прерывания. Адрес вектора 08h. 1 = на выводе PA0/INT обнаружен заданный фронт сигнала; 0 = на выводе PA0/INT не обнаружен заданный фронт сигнала
бит 3	**PEIE:** Бит разрешение периферийных прерываний. Этот бит действует, как бит глобального разрешения периферийных прерываний, чьи соответствующие биты разрешения прерываний также установлены. 1 = периферийные прерывания разрешены; 0 = периферийные прерывания запрещены
бит 2	**T0CKIE:** Бит разрешения внешнего прерывания от вывода PA1/T0CLK. 1 = прерывание разрешено; 0 = прерывание запрещено
бит 1	**T0IE:** Бит разрешения прерывания по переполнению таймера 0. 1 = прерывание разрешено; 0 = прерывание запрещено
бит 0	**INTE:** Бит разрешения внешнего прерывания на выводе PA0/INT. 1 = прерывание разрешено; 0 = прерывание запрещено

Обозначения:

R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры разрешения периферийных прерываний PIE1

Эти регистры содержат индивидуальные биты разрешения периферийных прерываний.

Таблица 7

Регистр PIE1 (адрес: 15h, банк 2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	**-**	**SPIIE**	**EEPROM IE**	**NANDIE**	**USBIE**	**TXIE**	**RCIE**
бит 7	6	5	4	3	2	1	бит 0

бит 7	Зарезервировано.
бит 6	Зарезервировано.
бит 5	**SPIIE:** Прерывание от блока SPI. 1 = разрешено; 0 = запрещено
бит 4	**EEPROMIE:** Прерывание от EERPOM контроллера. 1 = разрешено; 0 = запрещено
бит 3	**NANDIE:** Прерывание от NAND Flash контроллера. 1 = разрешено; 0 = запрещено
бит 2	**USBIE:** Прерывание от USB контроллера. 1 = разрешено; 0 = запрещено. Примечание: Кроме данного бита в регистрах контроллера есть флаги разрешения прерываний для различных событий в контроллере
бит 1	**TXIE:** Бит разрешения прерывания от передатчика USART (буфер передатчика пуст). 1 = прерывание разрешено; 0 = прерывание запрещено
бит 0	**RCIE:** Бит разрешения прерывания от приемника USART (в буфере приемника есть данные). 1 = прерывание разрешено; 0 = прерывание запрещено

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры запроса периферийных прерываний PIR1

Этот регистр содержат индивидуальные флаги запросов периферийных прерываний.

Примечание:

Флаги запроса прерываний устанавливаются при возникновении условий прерываний, вне зависимости от состояния флага общего запрета прерываний GLINTD и соответствующих флагов разрешения периферийных прерываний. Перед разрешением прерывания, пользователь должен сбросить флаги запросов прерываний, чтобы программа не перешла незамедлительно к подпрограмме обработки периферийных прерываний.

Таблица 8

Регистр PIR1 (адрес: 14h, банк 2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R-0
-	**-**	**SPIIF**	**EEPROM IF**	**NANDIF**	**USBIF**	**TXIF**	**RCIF**
бит 7	6	5	4	3	2	1	бит 0

бит 7	Зарезервировано
бит 6	Зарезервировано
бит 5	**SPIIF:** Флаг прерывания от блока SPI. 1 = Блок SPI закончил передачу/прием; 0 = Блок SPI свободен или ведет передачу/прием
бит 4	**EEPROMIF:** Флаг прерывания от контроллера EEPROM. 1 = закончена работа; 0 = блок свободен или ведет работу
бит 3	**NANDIF:** Флаг прерывания от контроллера NAND Flash. 1 = закончена работа; 0 = блок свободен или ведет работу
бит 2	**USBIF:** Флаг прерывания от контроллера USB. 1 = есть событие в контроллере USB; 0 = нет событий
бит 1	**TXIF:** Флаг запроса прерывания от передатчика USART. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = буфер передатчика USART пуст; 0 = буфер передатчика USART заполнен
бит 0	**RCIF:** Флаг запроса прерывания от приемника USART. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = в буфере приемника USART есть данные; 0 = буфер приемника USART пуст

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Обработка прерываний

Бит глобального запрещения прерываний GLINTD (CPUSTA<4>) разрешает все немаскированные прерывания (если сброшен), или запрещает все прерывания (если установлен). Индивидуальные прерывания запрещены через соответствующий бит разрешения в регистре INTSTA. Для запрета периферийных прерываний необходимо, чтобы был сброшен бит глобального разрешения периферийных прерываний PEIE или сброшен бит разрешения соответствующего периферийного прерывания. Запрещение периферийных прерываний через бит глобального разрешения периферийных прерываний, запрещает все периферийные прерывания. Бит GLINTD устанавливается при сбросе микроконтроллера (прерывания запрещены).

Команда RETFIE сбрасывает бит GLINTD для разрешения прерываний и загружает счетчик команд значением из вершины стека. Когда происходит прерывание, бит GLINTD автоматически устанавливается для запрета дальнейших прерываний, адрес возврата записывается в стек и счетчик команд загружается адресом вектора прерывания. Существует 4 вектора прерывания, что способствует сокращению задержки при обработке прерываний. Флаги запросов прерываний должны быть сброшены в программе перед разрешением прерываний, чтобы избежать повторного вызова.

Вектор периферийных прерываний имеет множество источников прерываний. В программе обслуживания периферийного прерывания, источник прерывания можно определить проверкой флагов запроса прерывания.

Микроконтроллер имеет 4 вектора прерываний. Адреса векторов и их приоритеты показаны в Таблица 69. Если происходит запрос двух прерываний одновременно, прерывание с большим приоритетом будет обслуживаться в первую очередь. Это означает, что адрес вектора именно этого прерывания будет загружен в счетчик команд (PC).

Таблица 9

Приоритеты и адреса векторов прерываний

Адрес	Вектор	Приоритет
0008h	Внешнее прерывание на выводе PA0/INT (INTF)	1 (самый высокий)
0010h	Прерывание по переполнению TMR0 (T0IF)	2
0018h	Внешнее прерывание по PA1/T0CLK (T0CKIF)	3
0020h	Периферийные прерывания (PEIF)	4 (самый низкий)

Примечание:

- Индивидуальные флаги запроса прерывания устанавливаются независимо от состояния соответствующего маскирующего бита или бита GLINTD.
- Прежде, чем запретить какое-либо прерывание, сбросом разрешающего бита регистра INTSTA, бит GLINTD должен быть установлен (общий запрет прерываний).

Прерывание от вывода PA0/INT

Внешнее прерывание от вывода PA0/INT происходит либо по переднему фронту сигнала, если бит INTEDG (T0STA<7>) установлен, либо по заднему фронту, если бит INTEDG сброшен. Когда активный фронт сигнала появляется на входе PA0/INT, бит INTF (INTSTA<4>) устанавливается. Это прерывание может быть

запрещено сбросом бита INTE (INTSTA<0>). Прерывание на выводе PA0/INT может выводить микроконтроллер из режима SLEEP. Смотрите раздел описания режима SLEEP.

Прерывание от вывода PA1/T0CLK

Внешнее прерывание от вывода PA1/T0CLK происходит либо по переднему фронту сигнала, если бит T0SE (T0STA<6>) установлен, либо по заднему фронту, если бит T0SE сброшен. Когда активный фронт сигнала появляется на выводе PA1/T0CLK, бит T0CKIF (INTSTA<6>) устанавливается. Формирование флага запроса прерывания T0CKIF происходит только в случае тактирования таймера TMR0 от этого вывода, т.е. это прерывание от внешнего тактового сигнала таймера. Если TMR0 тактируется от внутренней тактовой частоты, то прерывание не формируется. Прерывание может быть запрещено сбросом бита T0CKIE (INTSTA<2>). Прерывание от вывода PA1/T0CLK может выводить микроконтроллер из режима SLEEP. Смотрите раздел описания режима SLEEP.

Периферийные прерывания

Установленный флаг запроса периферийных прерываний PEIF показывает, что произошло, по крайней мере, одно периферийное прерывание. Бит PEIF только для чтения и является объединением по «ИЛИ» всех флагов запросов периферийных прерываний, логически умноженных по «И» на соответствующие биты разрешения прерываний в регистрах PIE. Некоторые периферийные прерывания могут выводить микроконтроллер из режима SLEEP. Смотрите раздел описания режима SLEEP.

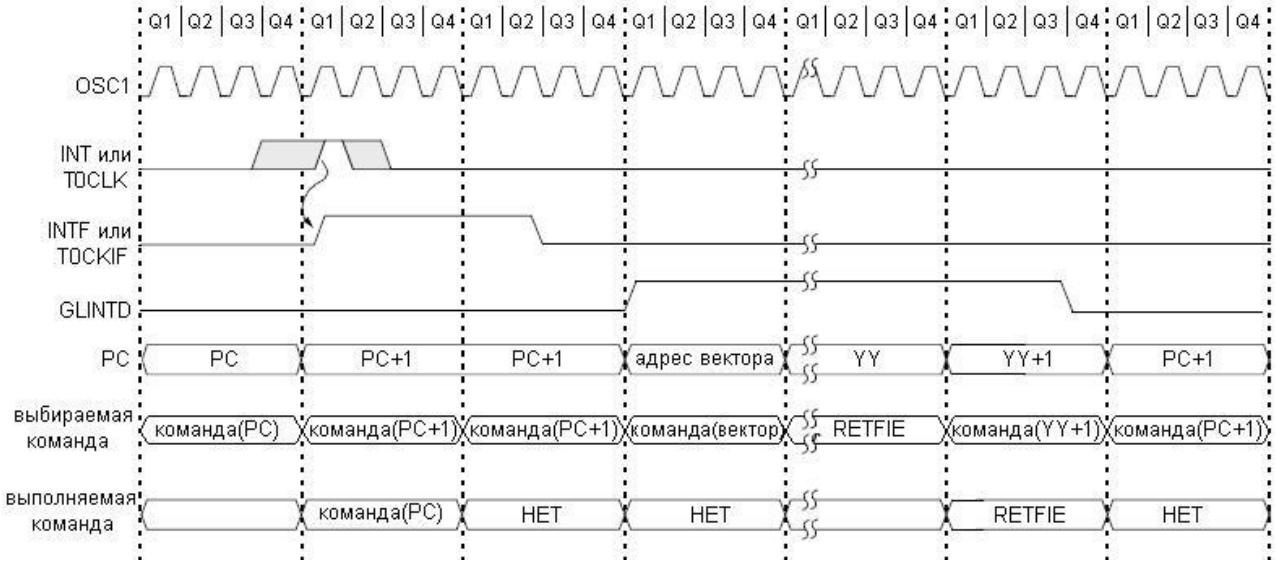


Рис. 11 Временная диаграмма обработки прерываний от выводов PA0/INT или PA1/T0CLK

Сохранение регистров при прерывании

Во время прерывания, в стеке сохраняется только значение PC для возврата. Сохранение других регистров необходимо реализовать программно.
В Пример 1 показано сохранение и восстановление информации в подпрограмме обработки прерывания. Этот пример приведен для простой схемы прерываний, где не может произойти вложенности прерываний. Содержимое регистров сохраняется в области GPR вне банков.

В Пример 2 показано сохранение и восстановление информации для случая когда требуется вложение прерываний. В приведенном примере может выполняться до максимум 6 уровней вложения прерываний. BSR хранится в области GPR вне банков, в то время как другие регистры будут храниться в определенном банке. Таким образом, эта программа может осуществить 6 записей блоков со значениями сохраняемых регистров. Для работы программа требует выделенного регистра косвенной адресации FSR0.

Сегменты кода PUSH и POP (запись в стек и чтение стека) могут быть либо в каждой подпрограмме обработки прерывания, либо могут быть вызываемыми подпрограммами. В зависимости от конкретного приложения, также может потребоваться сохранение и других регистров.

Пример 1

Сохранение регистров при прерывании (простой вариант)

```
; Адреса, которые используются для хранения значений регистров находятся во
; внутренней памяти данных. Диапазон адресов 1Ah – 1Fh (вне банков).
; При помощи команды MOVFP может быть сохранено и восстановлено до 6 ячеек.
; Эта команда не влияет на биты состояний и не нарушает значение регистра WREG.

UNBANK1     EQU   0x01A ; адрес для сохранения первой ячейки
UNBANK2     EQU   0x01B ; адрес для сохранения второй ячейки
UNBANK3     EQU   0x01C ; адрес для сохранения третьей ячейки
UNBANK4     EQU   0x01D ; адрес для сохранения четвертой ячейки
UNBANK5     EQU   0x01E ; адрес для сохранения пятой ячейки
                        ; (метка не используется в программе)
UNBANK6     EQU   0x01F ; адрес для сохранения шестой ячейки
                        ; (метка не используется в программе)

                                          ; Адрес вектора прерывания
PUSH  MOVFP     ALUSTA, UNBANK1  ; Запись в стек значения ALUSTA
      MOVFP     BSR, UNBANK2             ; Запись в стек значения BSR
      MOVFP     WREG, UNBANK3            ; Запись в стек  значения WREG
      MOVFP     PCLATH, UNBANK4  ; Запись в стек значения PCLATH
      :
                                          ;Код программы обработки прерывания
      :
POP   MOVFP     UNBANK4, PCLATH  ; Восстановление значения PCLATH
      MOVFP     UNBANK3, WREG            ; Восстановление значения WREG
      MOVFP     UNBANK2, BSR             ; Восстановление значения BSR
      MOVFP     UNBANK1, ALUSTA  ; Восстановление значения ALUSTA

      RETFIE            ; Возврат из прерывания (разрешает прерывания)
```

Пример 2

Сохранение регистров при прерывании
(вариант с поддержкой вложенных прерываний)

```
; Адреса, которые используются для хранения значений регистров находятся во
; внутренней памяти данных. Для сохранения значений регистра BSR используется
; область данных с адресами 1Ah - 1Fh (вне банков). Таким образом может быть
; сохранено до 6 блоков значений регистров. Программа использует регистр FSR0
; (и биты его управления FS1 и FS0 в регистре ALUSTA).

Nobank_FSR      EQU   0x40
Bank_FSR        EQU   0x41
ALU_Temp        EQU   0x42
WREG_TEMP       EQU   0x43
BSR_S1          EQU   0x01A ; адрес первой ячейки для сохранения BSR
BSR_S2          EQU   0x01B ; 0x1Ah-0x1Fh - адреса ячеек для сохранения BSR
BSR_S3          EQU   0x01C
BSR_S4          EQU   0x01D
BSR_S5          EQU   0x01E
BSR_S6          EQU   0x01F

; Инициализация
        CALL   CLEAR_RAM ; Очистка ОЗУ данных
;
INIT_POINTERS                   ; подготовка параметров для процедур POP и PUSH
        CLRF    BSR, F                  ; установка банков в 0
        CLRF    ALUSTA, F               ; переключение FSR0 в режим
        BSF     ALUSTA, FS1 ; автоинкрементирования
        CLRF    WREG, F                 ; сброс WREG
        MOVLW   BSR_S1                  ; загрузка FSR0 значением первого адреса для
        MOVWF   FSR0                    ; сохранения BSR
        MOVWF   Nobank_FSR
        MOVLW   0x20
        MOVWF   Bank_FSR
        :
;       Код Вашей программы
        :

                                        ; Адрес вектора прерывания
PUSH    BSF     ALUSTA, FS0 ; FSR0 - режим автоинкрементирования
        BCF     ALUSTA, FS1
        MOVFP   BSR, INDF0              ; сохранение регистра BSR
        CLRF    BSR, F                  ; установка банка 0 для периферийных регистров и
                                ; ОЗУ данных
        MOVPF   ALUSTA, ALU_Temp
        MOVPF   FSR0, Nobank_FSR        ; сохранение значения FSR, используемого
                                        ; для сохранения BSR
        MOVPF   WREG, WREG_TEMP
        MOVFP   Bank_FSR, FSR0          ; восстановление значения FSR, используемого
                                        ; для сохранения других значений
        MOVFP   ALU_Temp, INDF0         ; запись в стек значения ALUSTA
        MOVFP   WREG_TEMP, INDF0                ; запись в стек значения WREG
        MOVFP   PCLATH, INDF0           ; запись в стек значения PCLATH
        MOVPF   FSR0, Bank_FSR          ; сохранение значения FSR, используемого
                                        ; для сохранения других значений
        MOVFP   Nobank_FSR, FSR0        ; восстановление значения FSR, используемого
                                        ; для сохранения BSR
        :
```

```
;       Код программы обработки прерывания
        :
POP     CLRF        ALUSTA, F     ; FSR0 - режим автодекрементирования
        MOVFP       Bank_FSR, FSR0      ; восстановление значения FSR, используемого
                                        ; для сохранения других значений
        DECF        FSR0, F
        MOVFP       INDF0, PCLATH       ; чтение значения PCLATH
        MOVFP       INDF0, WREG         ; чтение значение WREG
        BSF         ALUSTA, FS1 ; FSR0 - режим без изменения значения регистра
        MOVPF       INDF0, ALU_Temp     ; чтение  значения ALUSTA
        MOVPF       FSR0, Bank_FSR      ; сохранение значения FSR, используемого
                                        ; для сохранения других значений
        DECF        Nobank_FSR, F
        MOVFP       Nobank_FSR, FSR0    ; восстановление значения FSR, используемого
                                        ; для сохранения BSR
        MOVFP       ALU_Temp, ALUSTA
        MOVFP       INDF0, BSR
;
        RETFIE      ; возврат из прерывания (разрешает прерывания)
```

Организация памяти

В микроконтроллере есть два блока памяти: память программ и память данных. Каждый блок имеет свою собственную шину, так что доступ к каждому блоку возможен во время одного и того же цикла генератора.

Память данных делится на RAM общего назначения и регистры специальных функций (SFRs). Функционирование SFR-регистров, которые управляют ядром микроконтроллера, описываются в этой главе. SFR-регистры, используемые для управления модулями периферии, описываются в разделах, посвященным этим конкретным модулям периферии.

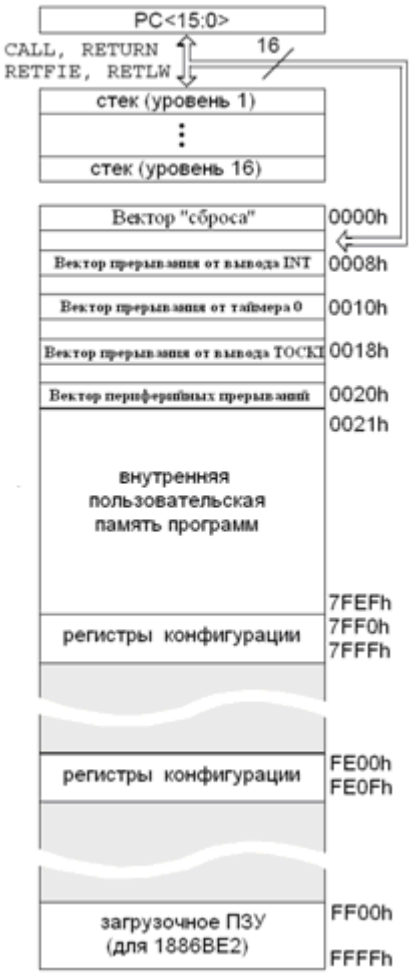


Рис. 12 Карта памяти программ и стека

Память программ

Микроконтроллеры имеют 16-ти битный счетчик команд, способный адресовать область памяти программ размером 64К х 16 бит. Вектор «сброса» имеет адрес 0000h, вектора прерывания находятся по адресам 0008h, 0010h, 0018h и 0020h (см.
Рис. **12**).

Объем внутренней памяти программ составляет 32К х 16 бит. Память перепрограммируемая FLASH типа.

Микроконтроллер может функционировать в одной из 4 возможных конфигураций памяти программ. Конфигурация выбирается битами конфигурации. Возможны следующие режимы: **микропроцессор, микроконтроллер, расширенный микроконтроллер и защищенный микроконтроллер**.

Режимы «микроконтроллера» и «защищенного микроконтроллера» обеспечивают доступ только к внутренней памяти программ. Любая попытка доступа по адресам вне памяти программ приводит к чтению неизвестного значения. Режим защищенного микроконтроллера имеет функцию защиты кода.

В режиме «расширенного микроконтроллера» имеется доступ как ко внутренней, так и ко внешней памяти программ. Выполнение автоматически переключается между внутренней и внешней памятью. 16-ти битный адрес позволяет адресовать диапазон памяти программ в 64К слов.

В режиме «микропроцессор» возможен доступ только к внешней памяти программ. Встроенная (внутренняя) память программ игнорируется. 16-ти битный адрес позволяет адресовать диапазон памяти программ в 64К слов. Микропроцессорный режим исходно установлен в незапрограммированных приборах.

Микроконтроллеры могут функционировать, когда память программ находится вне кристалла. Это режимы «микропроцессора» и «расширенного микроконтроллера». Микропроцессорный режим исходно установлен в незапрограммированных приборах.

Независимо от режима процессора память данных всегда находится на кристалле.

Когда выбран режим «микропроцессора» или «расширенного микроконтроллера», PORTC, PORTD и PORTE работают как системная шина. PORTC и PORTD – мультиплексированная шина адреса/данных, а PORTE<2:0> – сигналы управления. Форма сигналов адреса и данных показана на
Рис. **14**. Временные характеристики приведены в электрической спецификации. В режиме «расширенного микроконтроллера» когда прибор работает с внутренней памятью программ, внешние управляющие сигналы продолжают быть активными. Таким образом, они индицируют действия, которые происходят во внутренней памяти. Данные из внешней памяти игнорируются.

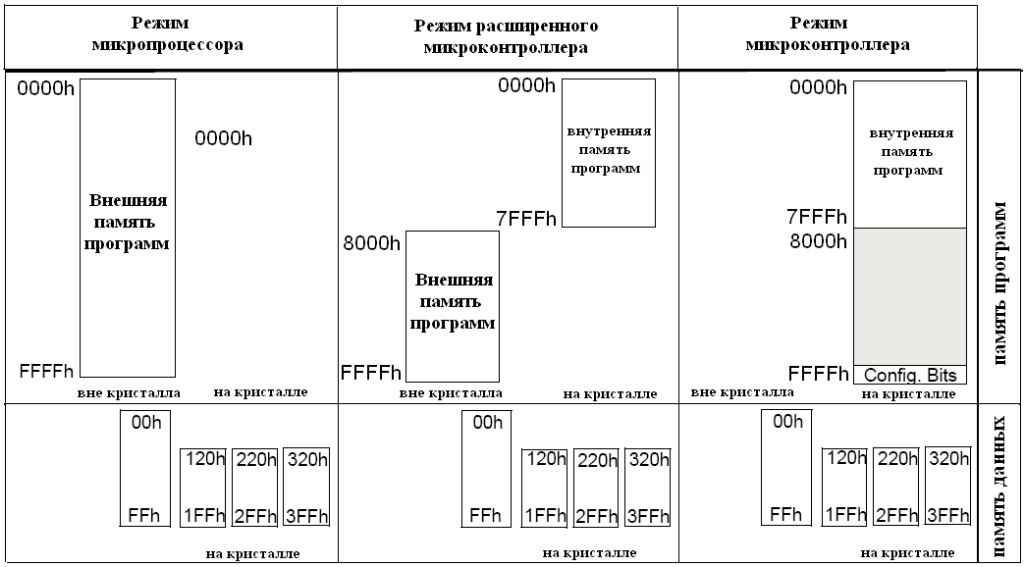


Рис. 13 Распределение памяти в различных режимах

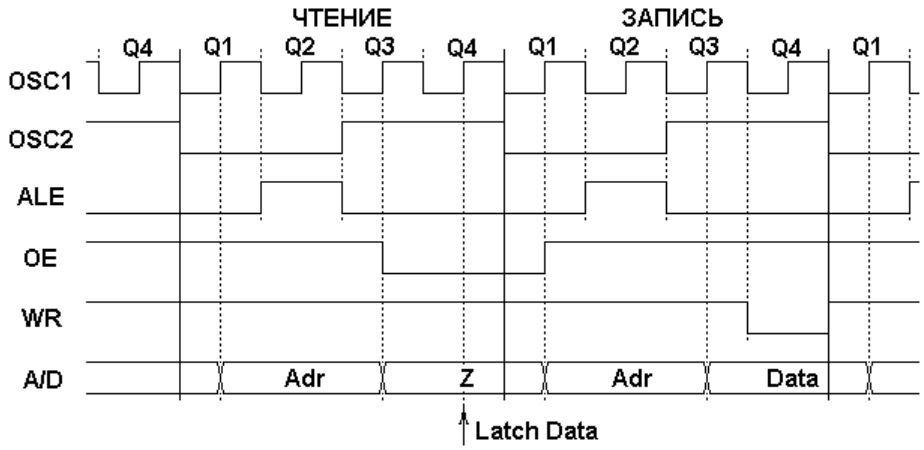


Рис. 14 Диаграммы сигналов доступа к внешней памяти программ

Память данных

Память данных разделена на 2 области. Первая – это область регистров общего назначения (GPR), и вторая – это область регистров специальных функций (SFR). SFR-регистры контролируют и обеспечивают режимы функционирования прибора.

Память данных разделена на банки, так организованы обе области. Область GPR сгруппирована в банки для того, чтобы получить объем памяти более 232 байта. Организация банками требует использования управляющих битов для выбора банка. Эти управляющие биты находятся в регистре выбора банков BSR. Если произведен доступ к области вне банков, значение регистра BSR игнорируется.

Команды MOVPF и MOVFP обеспечивают перенос значений данных из области периферии («P») в любое место в области регистров («F») и наоборот. «P»-диапазон определен адресами от 00h до 1Fh, диапазон «F» – от 00h до 0FFh. Диапазон «P» имеет 6 регистров, которые могут быть использованы как регистры общего назначения. Это может быть удобно для некоторых применений, где переменные необходимо скопировать в другие ячейки в ОЗУ общего назначения, (такие как запоминание информации о статусе во время прерывания).

Ко всей памяти данных можно обращаться используя либо прямой доступ, либо косвенный (используя регистры указателя адреса FSR0 и FSR1) (см. раздел 6-4). Косвенная адресация использует соответствующие управляющие биты BSR-регистра для доступа в области памяти данных, организованные в банки. BSR-регистр подробно рассматривается в разделе «Регистр выбора банка (BSR)».

Регистры общего назначения (GPR)

Микроконтроллеры имеют область регистров общего назначения (ОЗУ) объемом 902 байта. Эти регистры 8-ми битные. ОЗУ разбито на банки. Для облегчения переключение между этими банками существует команда «MOVLR bank». Регистр GPR не изменяется при всех типах сбросов.

Регистры специального назначения (SFR)

Регистры специального назначения (SFR) используются процессором и периферийными устройствами для управления работой прибора (см. Таблица 10). Эти регистры представляют собой статическое ОЗУ.

Регистры SFR могут быть разделены на 2 группы, те, которые связаны с функцией «ядра», и те, которые связаны с функциями периферии. Те регистры, которые связаны с «ядром», описываются ниже, а те, которые связаны с функциями периферии, описываются в соответствующем разделе для каждого модуля периферии. Регистры периферии организованы в банки, регистры «ядра» представляют собой область, не организованную в банки. Для облегчения переключения между периферийными банками используется команда «MOVLB bank».

Таблица 10

Схема адресов регистров

Адрес	**Не зависит от номера адресуемого банка.**							
00h	INDF0							
01h	FSR0							
02h	PCL							
03h	PCLATH							
04h	ALUSTA							
05h	T0STA							
06h	CPUSTA							
07h	INTSTA							
08h	INDF1							
09h	FSR1							
0Ah	WREG							
0Bh	TMR0L							
0Ch	TMR0H							
0Dh	TBLPTRL							
0Eh	TBLPTRH							
0Fh	BSR							
	Банк 0	**Банк 1**	**Банк 2**	**Банк 3**	**Банк 4**	**Банк 5**	**Банк 6**	**Банк 7**
10h	EP1_CFG1	DESC_ADR	EP1_REG	EP1_STAT	USB_ADR	-	NAND_MODE	-
11h	EP1_CFG2	DESC_DATA	EP2_REG	EP2_STAT	USB_ERROR	DDRE	NAND_DATA	SSPCON1
12h	EP2_CFG1	PORTA	EP3_REG	EP3_STAT	USB_STAT	PORTE	-	-
13h	EP2_CFG2	DDRD	EP4_REG	EP4_STAT	USB_CTRL	RCSTA	WRDIF	SSPSTAT
14h	EP3_CFG1	PORTD	PIR1	EP1_CNT	USB_IE1	RCREG	EEPROM_ CONT	SSPBUF
15h	EP3_CFG2	DDRC	PIE1	EP2_CNT	USB_IE2	TXSTA	EEPROM_ MODE	-
16h	EP4_CFG1	PORTC	-	EP3_CNT	USB_IE3	TXREG	EEPROM_ DATA	-
17h	EP4_CFG2	-	-	EP4_CNT	USB_IE4	SPBRG	EEPROM_ ADDR	-
	Не зависит от номера адресуемого банка.							
18h	PRODL							
19h	PRODH							
1Ah-1Fh	**Регистры общего назначения.**							
	Банк 0		**Банк 1**		**Банк 2**		**Банк 3**	
20h FFh	Регистры общего назначения.		Регистры общего назначения.		Регистры общего назначения.		Регистры общего назначения.	

Примечание:

- Регистры SFR в области адресов 10h-17h разбиты на банки. Младший полубайт регистра BSR определяет выбранный номер банка. Все не организованные в банки регистры игнорируют значения битов регистра BSR.
- Область памяти GPR с адресами 20h-FFh, 120h-1FFh, 220h-2FFh и 320h-3FFh разбита на банки. Старший полубайт регистра BSR определяет выбранный номер банка. Другие регистры памяти игнорируют значения битов регистра BSR.
- Чтение любого не существующего регистра дает значение равное нулю.

Таблица 11

Регистры специального назначения

Адрес	Наз-вание	бит 7	бит 6	бит 5	бит 4	бит 3	бит 2	бит 1	бит 0	POR, BOR	MCLR, WDT
Не зависит от номера адресуемого банка.											
00h	INDF0	использует содержимое FSR0 для адресации памяти данных (физически не реализован)								---- ----	---- ----
01h	FSR0	указатель 0 адреса, для косвенной адресации памяти данных								0000 0000	uuuu uuuu
02h	PCL	младшие 8 бит счетчика команд								0000 0000	0000 0000
03h	PCLATH	регистр-защелка для старших 8 бит счетчика команд								0000 0000	0000 0000
04h	ALUSTA	FS3	FS2	FS1	FS0	OV	Z	DC	C	1111 0000	1111 0000
05h	T0STA	INTEDG	T0SE	T0CS	T0PS3	T0PS2	T0PS1	T0PS0	-	0000 000-	0000 000-
06h	CPUSTA	-	-	STKAV	GLINTD	TO	PD	POR	BOR	--11 11q0	--11 qq1u
07h	INTSTA	PEIF	T0CKIF	T0IF	INTF	PEIE	T0CKIE	T0IE	INTE	0000 0000	0000 0000
08h	INDF1	использует содержимое FSR1 для адресации памяти данных (физически не реализован)								---- ----	---- ----
09h	FSR1	указатель 1 адреса, для косвенной адресации памяти данных								0000 0000	uuuu uuuu
0Ah	WREG	рабочий регистр								0000 0000	0000 0000
0Bh	TMR0L	младший байт регистра таймера 0								0000 0000	0000 0000
0Ch	TMR0H	старший байт регистра таймера 0								0000 0000	0000 0000
0Dh	TBLPTRL	младший байт табличного указателя памяти программ								0000 0000	0000 0000
0Eh	TBLPTRH	старший байт табличного указателя памяти программ								0000 0000	0000 0000
0Fh	BSR	регистр выбора банка памяти данных								0000 0000	0000 0000
Банк 0											
10h	EP1_ CFG1	-	Максимальный размер пакета первой оконечной точки							-000 0000	-000 0000
11h	EP1_ CFG2	-	BLOCK _EP1	TYPE1_ EP1	TYPE0_ EP1	MODE2 _EP1	MODE1 _EP1	MODE0 _EP1	-	- 000 00 0-	- 000 00 0-
12h	EP2_ CFG1	-	Максимальный размер пакета второй оконечной точки							-000 0000	-000 0000
13h	EP2_ CFG2	-	BLOCK _EP2	TYPE1_ EP2	TYPE0_ EP2	MODE2 _EP2	MODE1 _EP2	MODE0 _EP2	-	- 000 00 0-	- 000 00 0-
14h	EP3_ CFG1	-	Максимальный размер пакета третьей оконечной точки							-000 0000	-000 0000
15h	EP3_ CFG2	-	BLOCK _EP3	TYPE1_ EP3	TYPE0_ EP3	MODE2 _EP3	MODE1 _EP3	MODE0 _EP3	-	- 000 00 0-	- 000 00 0-
16h	EP4_ CFG1	-	Максимальный размер пакета четвертой оконечной точки							-000 0000	-000 0000
17h	EP4_ CFG2	-	BLOCK _EP4	TYPE1_ EP4	TYPE0_ EP4	MODE2 _EP4	MODE1 _EP4	MODE0 _EP4	-	- 000 00 0-	- 000 00 0-
Банк 1											
10h	DESC_ ADR	регистр адреса поля дескриптора								-000 0000	-000 0000
11h	DESC_ DATA	регистр значения поля дескриптора								0000 0000	0000 0000
12h	PORTA	-	-	-	-	PORTA [3:0]				---- xxxx	---- uuuu

13h	DDRD(4)	регистр выбора направления данных для PORTD								1111 1111	1111 1111
14h	PORTD(3,4)	PORTD [7:0]								xxxx xxxx	uuuu uuuu
15h	DDRC(4)	регистр выбора направления данных для PORTC								1111 1111	1111 1111
16h	PORTC(3,4)	PORTC [7:0]								xxxx xxxx	uuuu uuuu
17h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
Банк 2											
10h	EP1_ REG	регистр данных FIFO первой оконечной точки								0000 0000	0000 0000
11h	EP2_ REG	регистр данных FIFO второй оконечной точки								0000 0000	0000 0000
12h	EP3_ REG	регистр данных FIFO третьей оконечной точки								0000 0000	0000 0000
13h	EP4_ REG	регистр данных FIFO четвертой оконечной точки								0000 0000	0000 0000
14h	PIR1	-	-	SPI IF	EEPRO MIF	NAND IF	USB IF	TXIF	RCIF	--00 0010	--00 0010
15h	PIE1	-	-	SPI IE	EEPRO MIE	NAND IE	USB IE	TXIE	RCIE	--00 0000	--00 0000
16h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
17h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
Банк 3											
10h	EP1_ STAT	-	-	BLK1_ DATA	ACTIVE 1	-	-	FULL1	EMPTY 1	--00 -- 00	--00 -- 00
11h	EP2_ STAT	-	-	BLK2_ DATA	ACTIVE 2	-	-	FULL2	EMPTY 2	--00 -- 00	--00 -- 00
12h	EP3_ STAT	-	-	BLK3_ DATA	ACTIVE 3	-	-	FULL3	EMPTY 3	--00 -- 00	--00 -- 00
13h	EP4_ STAT	-	-	BLK4_ DATA	ACTIVE 4	-	-	FULL4	EMPTY 4	--00 -- 00	--00 -- 00
14h	EP1_ CNT	регистр счетчика числа слов в первой оконечной точке								-000 0000	-000 0000
15h	EP2_ CNT	регистр счетчика числа слов во второй оконечной точке								-000 0000	-000 0000
16h	EP3_ CNT	регистр счетчика числа слов в третьей оконечной точке								-000 0000	-000 0000
17h	EP4_ CNT	регистр счетчика числа слов в четвертой оконечной точке								-000 0000	-000 0000
Банк 4											
10h	USB_ ADR	регистр функционального адреса USB устройства								-000 0000	-000 0000
11h	USB_ ERROR	BUS_E RR	DROP_ FRM	TO_ ERR	SEQ_E RR	NSE_E RR	PID_ ERR	CRC16 _ERR	CRC5_ ERR	0000 0000	0000 0000
12h	USB_ STAT	-	ADR_S ET	CONF_ SET	USB_R ST	EP2_ SEL	EP1_ SEL	EP0_ SEL	USB_B USY	-000 0000	-000 0000
13h	USB_ CTRL	-	-	-	-	USB_T EST2	USB_T EST1	FULL_L OW	USB_E N	--00 0000	--00 0000
14h	USB_IE1	EP4_ FULL_ IE	EP4_ EMPTY_I E	EP3_ FULL_ IE	EP3_ EMPTY_ IE	EP2_ FULL_ IE	EP2_ EMPTY_ IE	EP1_ FULL_ IE	EP1_ EMPTYI_ E	0000 0000	0000 0000
15h	USB_IE2	USB_ RST_IE	USB_ BUSY_I E	-	-	-	-	-	-	0000 -- --	0000 -- --
16h	USB_IE3	BUS_E RR_IE	DROP_ FRM_IE	TO_ER R_IE	SEQ_E RR_IE	NSE_E RR_IE	PID_ER R_IE	CRC16 _ERR_ IE	CRC5_ ERR_IE	0000 0000	0000 0000
17h	USB_IE4	-	-	-	-	USB_ ALL_IE	-	ADR_ SET_I E	CONF _SET_ IE	---- 0- 00	---- 0- 00
Банк 5											

10h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
11h	DDRE(4)	регистр выбора направления данных для PORTE								1111 1111	1111 1111
12h	PORTE(3,4)	PORTE [7:0]								xxxx xxxx	uuuu uuuu
13h	RCSTA	SPEN	RX9	SREN	CREN	-	FERR	OERR	RX9D	0000 - 000	0000 - 000
14h	RCREG	регистр принимаемых данных последовательного порта USART								0000 0000	0000 0000
15h	TXSTA	CSRC	TX9	TXEN	SYNC	-	-	TRMT	TX9D	0000 -- 10	0000 -- 10
16h	TXREG	регистр передаваемых данных последовательного порта USART								0000 0000	0000 0000
17h	SPBRG	регистр управления генератором последовательного порта USART								0000 0000	0000 0000
Банк 6											
10h	NAND_ MODE	EN_ NAND	LOW_ SPEED_ NAND	IE_BUSY_ NAND	BUSY_ NAND	MODE3 _NAND	MODE2 _NAND	MODE1 _NAND	MODE0 _NAND	0000 0000	0000 0000
11h	NAND_DA TA	регистр данных контроллера внешней NAND Flash памяти								0000 0000	0000 0000
12h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
13h	WRDIF	регистр задания временных характеристик FLASH памяти программ								0000 0101	0000 0101
14h	EEPROM_ CONT	-	EETES T	CPTES T	VEE2	VEE1	BRG2	BRG1	BRG0	1000 0000	1000 0000
15h	EEPROM_ MODE	EN_EE	-	IE_BUSY _EE	BUSY_ EE	-	MODE2 _EE	MODE1 _EE	MODE0 _EE	0000 0000	0000 0000
16h	EEPROM_ DATA	регистр данных EEPROM								0000 0000	0000 0000
17h	EEPROM_ ADDR	регистр адреса EEPROM								0000 0000	0000 0000
Банк 7											
10h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
11h	SSPCON1	WCOL	SSPOV	EN_ SSP	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
12h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
13h	SSPSTAT	SMP	CKE	-	-	-	-	-	BF	00-- ---- 0	00-- ---- 0
14h	SSPBUF	регистр принятых/передаваемых данных								0000 0000	0000 0000
15h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
16h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
17h	-	-	-	-	-	-	-	-	-	---- ----	---- ----
Не зависит от номера адресуемого банка.											
18h	PRODL	младший байт 16-ти битного результата (8x8 битное аппаратное умножение)								0000 0000	0000 0000
19h	PRODH	старший байт 16-ти битного результата (8x8 битное аппаратное умножение)								0000 0000	0000 0000

Обозначения:

x=не известно; u=не изменяется; -=не реализовано, читается «0»;q=зависит от условий.

Примечания:

1. К старшему байту счетчика команд нет прямого доступа. PCLATH - это регистр-защелка для PC<15:8>. При выполнении некоторых команд значение регистра PCLATH переписывается в PC<15:8>, или наоборот: значение PC<15:8> переписывается в PCLATH.

2. Внешний сброс от вывода MCLR не влияет на статусные биты TO и PD регистра CPUSTA.
3. Это то значение, которое будет на выходной защелке порта.
4. Когда прибор сконфигурирован для «Микропроцессорного» и «Расширенного микроконтроллерного» режимов, функционирование этого порта не связано с этими регистрами.

Регистр статуса процессора (ALUSTA)

Регистр содержит биты статуса арифметического и логического блоков и биты управления режимом для режима косвенной адресации.

Как в случае со всеми другими регистрами, в регистр ALUSTA может быть загружен результат выполнения любой команды. Если регистр ALUSTA является местом назначения для результата определенной команды, которая может изменять биты Z, DC, C и OV, то запись в эти 4 бита запрещается. Эти биты устанавливаются или сбрасываются в соответствии с результатом выполнения команды. Следовательно, результат выполнения команды с регистром ALUSTA в качестве места назначения результата может стать отличным от того, что надеялись получить. Следовательно, рекомендуется для изменения ALUSTA-регистра использовать только следующие команды: BCF, BSF, SWAPF и MOVWF, т.к. эти команды не влияют на какие-либо статусные биты. Чтобы посмотреть, как другие команды влияют на статусные биты, смотрите описание системы команд.

Арифметический и логический блок (АЛУ) может производить арифметические и логические операции над двумя операндами или с одним операндом. Все команды с одним операндом производятся либо с WREG-регистром либо с данным файловым регистром. В командах с двумя операндами один операнд – это WREG-регистр, другой – либо файловый регистр, либо 8-ми битная константа.

Таблица 12

Регистр ALUSTA (адрес: 04h, не зависит от номера банка)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
FS3	**FS2**	**FS1**	**FS0**	**OV**	**Z**	**DC**	**C**
бит 7	6	5	4	3	2	1	бит 0
бит 7,6	**FS3:FS2:** биты выбора режима FSR1 00 = автодекремент величины FSR1 после выполнения команды 01 = автоинкремент величины FSR1 после выполнения команды 1x = значение FSR1 не изменяется						
бит 5,4	**FS1:FS0:** биты выбора режима FSR0 00 = автодекремент величины FSR0 после выполнения команды 01 = автоинкремент величины FSR0 после выполнения команды 1x = значение FSR0 не изменяется						
бит 3	**OV:** бит переполнения Этот бит используется для знаковой арифметики (дополнение до 2). Он показывает переполнение, когда знаковый бит (7 бит) изменяет состояние. 1 = произошло переполнение для знаковой арифметики в арифметических операциях (т.е. результат для знаковой арифметики превысил +127, или стал меньше чем –128)						

	0 = не произошло переполнение
бит 2	**Z:** флаг нуля 1 = результат арифметической или логической операции равен 0 0 = результат арифметической или логической операции не равен 0
бит 1	**DC:** флаг десятичного переноса/заема Для команд ADDWF и ADDLW. 1 = произошел перенос из 4-го снизу бита результата 0 = не было переноса из 4-го снизу бита результата Примечание: для заема значение инверсное (для команд вычитания)
бит 0	**C:** флаг переноса/заема Для команд ADDWF и ADDLW. Отметим, что вычитание выполняется дополнением до 2 второго операнда. Для команд сдвига RRCF и RLCF этот бит загружается либо старшим, либо младшим битом регистра-источника. 1 = произошел перенос из самого значащего бита результата 0 = нет переноса из самого значащего бита результата. Примечание: для заема значение инверсное (для команд вычитания)

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистр статуса ЦПУ (CPUSTA)

Содержит статусный и управляющий биты для ЦПУ. Этот регистр содержит бит, который используется для глобального разрешения/запрещения прерываний. Если необходимо разрешить/запретить только определенное прерывание, используются регистры статуса прерываний (INTSTA) или регистры разрешения прерываний от периферии (PIE1). Регистр CPUSTA также показывает, доступность стека, и содержит флаги включения питания (PD) и переполнения сторожевого таймера (TO). Биты TO, PD и STKAV доступны только для чтения. Они устанавливаются и сбрасываются в соответствии с логикой прибора. Следовательно, результат выполнения команды с регистром CPUSTA в качестве места назначения результата выполнения может быть отличным, нежели ожидалось.

Бит POR позволяет отличить сброс при включении питания от внешнего MCLR сброса или сброса по переполнению сторожевого таймера. Бит BOR является индикатором сброса по снижению напряжения питания (только в случае если схема сброса по снижению напряжения питания включена в регистре конфигурации).

Таблица 13

Регистр CPUSTA (адрес: 06h, не зависит от номера банка)

U-0	U-0	R-1	R/W-1	R-1	R-1	R/W-0	R/W-0
-	-	STKAV	GLINTD	TO	PD	POR	BOR
бит 7	6	5	4	3	2	1	бит 0

бит 7,6	не реализованы: читаются значения равные нулю.
бит 5	**STKAV:** флаг доступа к стеку Этот флаг показывает, что величина 4-х битного указателя стека равна Fh, или произошел переход от Fh к 0h, т.е. переполнение стека. 1 = стек доступен 0 = стек полон, или произошло переполнение стека (с тех пор, как этот бит был сброшен при переполнении стека, только «сброс» (RESET) прибора может установить этот бит)
бит 4	**GLINTD:** бит глобального запрета прерываний Этот бит запрещает все прерывания. Когда прерывания разрешены, то вызвать прерывания могут только источники с установленными битами разрешения прерывания. 1 = запрещены все прерывания 0 = разрешены все немаскированные прерывания
бит 3	**TO:** флаг переполнения сторожевого таймера 1 = устанавливается после включения питания или выполнения команд CLRWDT или SLEEP 0 = после переполнения сторожевого таймера
бит 2	**PD:** флаг включения питания 1 = устанавливается после включения питания или выполнения команды CLRWDT 0 = после выполнения команды SLEEP
бит 1	**POR:** флаг сброса при включении питания 1 = не было сброса при включении питания 0 = произошел сброс при включении питания (бит должен быть установлен программно)
бит 0	**BOR:** флаг сброса по снижению напряжения питания Когда бит BODEN в регистре конфигурации установлен (разрешено): 1 = сброса при снижении питания не было 0 = произошел сброс при снижении питания (бит должен быть установлен программно) Когда бит BODEN в регистре конфигурации сброшен (запрещено): значение бита безразлично

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Регистр статуса управления TMR0 (T0STA)

Этот регистр содержит различные биты управления. Бит 7 (INTEDG) используется для выбора управляющего перепада сигнала (т.е. фронт или срез сигнала) на выводе PA0/INT, при котором будет устанавливаться флаг запроса прерывания INTF. Остальные биты конфигурируют таймер 0, его предделитель и источник тактовых сигналов.

Таблица 14

Регистр TOSTA (адрес: 05h, не зависит от номера банка)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
INTEDG	**T0SE**	**T0CS**	**T0PS3**	**T0PS2**	**T0PS1**	**T0PS0**	**-**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**INTEDG:** бит выбора управляющего перепада сигнала на выводе PA0 (INT) для прерывания. Этот бит выбирает, на фронте или спаде сигнала будет происходить прерывание. 1 = фронт сигнала на выводе PA0/INT генерирует прерывание 0 = спад сигнала на выводе PA0/INT генерирует прерывание
бит 6	**T0SE:** бит выбора управляющего перепада сигнала при внешнем тактировании таймера 0. Этот бит выбирает, на фронте или спаде сигнала таймер 0 будет инкрементироваться. Когда T0CS=0 (внешнее тактирование): 1 = фронт сигнала на выводе PA1/T0CLK инкрементирует таймер 0 и/или устанавливает T0CKIF - бит 0 = спад сигнала на выводе PA1/T0CLK инкрементирует таймер 0 и/или устанавливает T0CKIF - бит Когда T0CS=1 (внутреннее тактирование): значение бита безразлично
бит 5	**T0CS:** бит выбора источника тактирования для таймера 0. Этот бит выбирает источник синхронизации для таймера 0. 1 = внутренняя тактовая частота с генератора циклов (Тсу) 0 = внешнее тактирование с вывода PA1/T0CLK
бит 4-1	**T0PS3:T0PS0:** биты выбора предделителя для таймера 0. Эти биты позволяют выбрать величину деления предделителя: 0000 - 1:1 0001 - 1:2 0010 - 1:4 0011 - 1:8 0100 - 1:16 0101 - 1:32 0110 - 1:64 0111 - 1:128 1xxx - 1:256
бит 0	не реализовано: читается значение равное нулю

Обозначения:

R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Функционирование стека

Микроконтроллеры имеют 16x16 бит аппаратный стек (см. Рис. **12**). Стек не является частью области памяти программ или данных, указатель стека не является ни считываемым, ни записываемым. Значение счетчика команд PC помещается (PUSH) в стек, когда выполняются команды CALL и LCALL или произошло прерывание. Стек восстанавливает значение PC (POP) в случае выполнения команд RETURN, RETLW или RETFIE. Операции «PUSH» и «POP» не влияют на PCLATH (защелку).

Стек работает как круговой буфер с указателем стека, сброшенным в нулевое значение после любых типов сбросов. В стеке существует определенный бит (STKAV), позволяющий программно убедиться в том, что не произошло переполнения стека. Бит STKAV устанавливается после сброса прибора. Когда указатель стека становится равен Fh, STKAV сбрасывается. Если указатель стека проходит адреса от Fh к 0h, бит STKAV будет оставаться сброшенным до тех пор, пока не произойдет сброс прибора.

Примечание.

1. Не предусмотрен специальный статусный бит для заполненного стека. STKAV-бит может быть использован для обнаружения того, что стек заполнен, в результате чего указатель стека находится на его вершине.
2. Здесь нет командной мнемоники, называемой «PUSH» или «POP». Это действия, которые происходят при выполнении команд CALL, RETURN, RETLW и RETFIE или обращении к вектору прерывания.
3. После сброса если операция «POP» имеет место до операции «PUSH», бит STKAV будет сброшен. Это выглядит так же, как в случае когда стек полон. Если следующей выполняется операция «PUSH» (перед следующим «POP»), то бит STKAV зафиксируется сброшенным. И только сброс прибора устанавливает этот бит.

После того, как прибор 16 раз осуществил операцию «PUSH» (без операции «POP»), 17-й «push» записывает значение поверх первого. 18-й «push» записывает сверху второго «push» (и т.д.).

Косвенная адресация

Косвенная адресация – это режим адресации памяти данных, когда адрес памяти данных в команде не фиксирован. Таким образом, адрес регистра, из которого будет производиться чтение или в который будет производиться запись, может быть модифицирован программой. Это может быть удобно в случае таблиц данных, размещенных в памяти данных. На Рис. **15** показан принцип косвенной адресации. Там показана модификация значения адреса памяти данных, значением регистра.

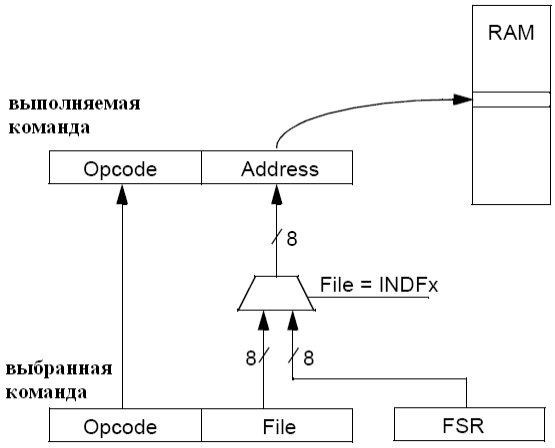


Рис. 15 Косвенная адресация

Пример 3 показывает использование косвенной адресации для очистки ОЗУ данных (от 20h до FFh) с минимальным количеством команд. Подобная концепция может быть использована для переноса определенного числа байт данных в передающий регистр USART (TXREG).

Микроконтроллер имеет две пары регистров для реализации косвенной адресации. Это: INDF0, FSR0 и INDF1, FSR1. Регистры INDF0 и INDF1 физически не реализованы. Чтение и запись в эти регистры активирует косвенную адресацию со значением адреса из соответствующего регистра FSR, который является адресом данных. FSR - это 8-ми битный регистр, позволяющий адресовать область памяти данных объемом 256 байт. Для памяти. организованной в банки, банк, к которому осуществляется доступ, определяется величиной в регистре BSR.

Если косвенно через FSR читается сам файл INDF0 или INDF1, то читаются все нули. Подобным образом, если в INDF0 (или INDF1) идет косвенная запись, операция будет эквивалентна команде NOP, и она не оказывает влияния на статусные биты.

Существуют 2 управляющих бита, связанных с каждым регистром FSR. Эти 2 бита конфигурируют FSR-регистр, чтобы:

- производить автодекремент значения (адреса) в регистре FSR после доступа к памяти с косвенной адресацией;
- производить автоинкремент значения (адреса) в регистре FSR после доступа к памяти с косвенной адресацией;
- не изменять значение (адрес) в регистре FSR после доступа к памяти с косвенной адресацией.

Эти управляющие биты находятся в регистре ALUSTA. Регистр FSR1 управляется битами FS3 и FS2, а регистр FSR0 управляется битами FS1и FS0.
Когда используются автоинкрементный или автодекрементный режимы, то изменение регистра FSR не отражается на регистре ALUSTA. Например, если при косвенной адресации регистр FSR станет равен нулю, то бит Z устанавливаться не будет. Если регистр FSR содержит величину 00h, косвенное чтение будет давать значение 00h (бит Z установлен), в то время как косвенная запись будет эквивалентна команде NOP (это не влияет на статусные биты).

Косвенная адресация позволяет за один цикл передавать данные по всему адресному пространству памяти данных. Это возможно с использованием команд MOVPF и MOVFP, где либо «P» либо «F» задано как INDF0 или INDF1. Если источник или приемник при косвенной адресации – это память, организованная в банки, то ячейка доступа будет определяться значением в регистре BSR.

Пример 3
Косвенная адресация

```
        MOVLW   0x20
        MOVWF   FSR0            ; FSR0 = 20h
        BCF     ALUSTA,FS1      ; Задание режима
        BSF     ALUSTA,FS0      ; автоинкрементирования FSR
        BCF     ALUSTA,C        ; C = 0
        MOVLW   END_RAM + 1
LP      CLRF    INDF0,F     ; очистка ячейки памяти (FSR-указатель адреса)
        CPFSEQ  FSR0            ; сравнение: FSR0 = END_RAM+1?
        GOTO    LP              ; Нет, очистка продолжается
        :                       ; Да, вся память очищена.
```

Регистры для чтения/записи таблиц

Регистры указателя таблиц TBLPTRL и TBLPTRH формируют 16-ти битное значение для адресации пространства памяти программ размером 64 Кслов. Указатель таблиц используется командами TABLWT и TABLRD. Эти команды позволяют осуществить передачу данных между областями данных и программ. Указатель таблиц служит в качестве 16-ти битного адреса слова внутри программной памяти. Регистр защелки таблиц – 16-ти разрядный регистр. Старший байт регистра TBLATH, младший байт TBLATL. Регистры не относятся ни к области памяти программ, ни к области памяти данных. Защелка таблиц используется для временной фиксации данных во время их передачи между памятью программ и памятью данных (см. описания команд TABLRD, TABLWT, TLRD и TLWT). Для более полного описания этих регистров и функционирования чтения/записи таблиц см. раздел 7.

Модуль счетчика команд

Счетчик команд PC - это 16-ти битный регистр. PCL - младший байт счетчика команд находится в области памяти данных. PCL можно читать и записывать точно так же как и любой другой регистр. PCH – это старший байт счетчика команд, и он не имеет прямой адресации. Т.к. PCH находится вне памяти программ и данных, то используется 8-ми битный регистр PCLATH в качестве удерживающей защелки для старшего байта счетчика команд. PCLATH находится в памяти данных. Пользователь может считывать или записывать PCH через PCLATH.

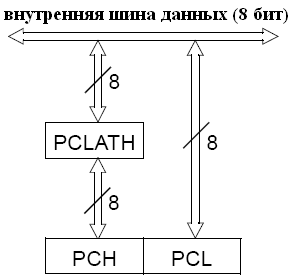


Рис. 16 Функционирование счётчика команд

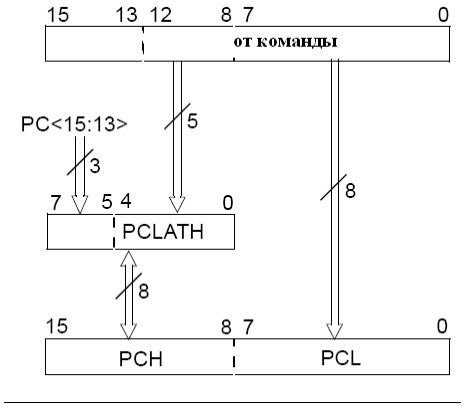


Рис. 17 Счетчик команд при выполнении инструкций CALL и GOTO

16-ти битный счетчик команд инкрементируется после выборки команды в течение цикла Q1 до тех пор пока:

- не изменится следующими командами: GOTO, CALL, LCALL, RETURN, RETLW или RETFIE;
- не изменится при переходе к вектору прерывания;
- не изменится в результате записи в регистр PCL результата выполнения команды.

Эти «переходы» эквивалентны вынужденному циклу «NOP» с переходом по адресу.

Рис. **16** и

Рис. **17** показывают функционирование счетчика команд в различных ситуациях.

Работа счетчика команд (PC) и регистра PCLATH для различных команд:

- **Команда LCALL:**

8-ми битный адрес указан в коде команды, PCLATH не изменяется.
PCLATH –> PCH; биты команды <7: 0> -> PCL

- **Любая команда чтения из PCL:**

PCL –> шина данных –> ALU или приемник; PCH –> PCLATH

- **Любая команда записи в PCL:**

8-ми битные данные –> шина данных –> PCL; PCLATH –> PCH

- **Любая команда чтения – модификации – записи PCL (например ADDWF PCL,F):**

Чтение: PCL -> шина данных -> ALU
Запись: 8-ми битный результат -> шина данных -> PCL
PCLATH –> PCH

- **Команда RETURN:**

Содержимое стека -> PC<15:0>

- **Команды CALL, GOTO:**

13-ти битный адрес указан в коде команды
биты команды <12:0> -> PC<12:0>
PC<15:13> -> PCLATH<7:5>
биты команды <12:8> -> PCLATH<4:0>

Команды чтения – модификации – записи воздействуют только на PCL. PCH загружается значением из PCLATH. Для примера ADDWF PCL,F приведет к переходу в пределах текущей страницы. Если PC=03F0h, WREG=30h и PCLATH=03h до начала действия команды, то после ее действия PC=0320h. Чтобы выполнить правильный 16-ти байтный переход, необходимо вычислить 16-ти

битный адрес приемника, записать старший байт в PCLATH и тогда записать младший в PCL.

Следующие команды, связанные со счетчиком команд, не изменяют PCLATH:

- LCALL, RETLW и RETFIE;
- переход к вектору прерывания;
- команды чтения – модификации - записи и записи для PCL.

Регистр выбора банка (BSR)

Регистр выбора банка BSR используется для переключения между банками в области памяти данных (см.
Рис. **18**). Младший полубайт используется для выбора банка периферийного регистра, для его записи используется команда «MOVLB bank». Старший полубайт используется для выбора банка памяти общего назначения (ОЗУ), для его записи используется команда «MOVLR bank». Если выбранный банк физически не реализован, то при его чтении будут считываться все нули. Любая запись в область памяти будет соответственно устанавливать или сбрасывать биты состояния АЛУ.

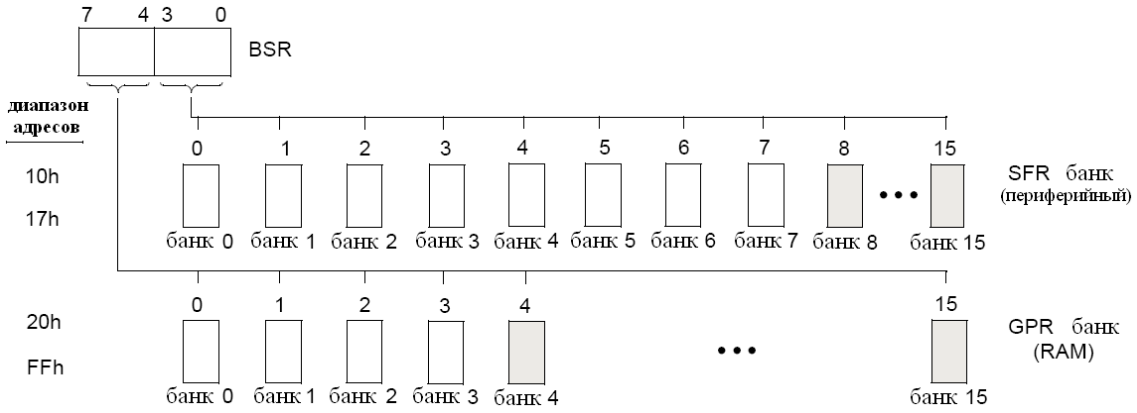


Рис. 18 Функционирование BSR

Считывание и запись таблиц данных

Микроконтроллер имеет 4 команды, которые дают возможность переносить данные из области памяти данных в область памяти программ и наоборот. Т.к. память программ 16-ти битная, а память данных 8-ми битная, то для переноса 16-ти битной величины данных в память данных или из памяти данных требуется 2 операции. Для записи данных из памяти данных в память программ используются 2 следующие команды: TLWT t,f и TABLWT t,i,f. Для записи данных из памяти программ в память данных используются 2 следующие команды: TLRD t,f и TABLRD t,i,f. Операнд команды TABLWT - «i» определяет: требуется ли автоматически инкрементировать величину 16-ти битного регистра TBLPTR (для следующей записи). В Пример 4 регистр TBLPTR автоматически не инкрементируется. Рис. 19 показывает выполнение этих 4 команд. Шаги показывают последовательность операций. Память программ может быть как внутренней, так и внешней (для режима «микропроцессор» или «расширенный микроконтроллер»).

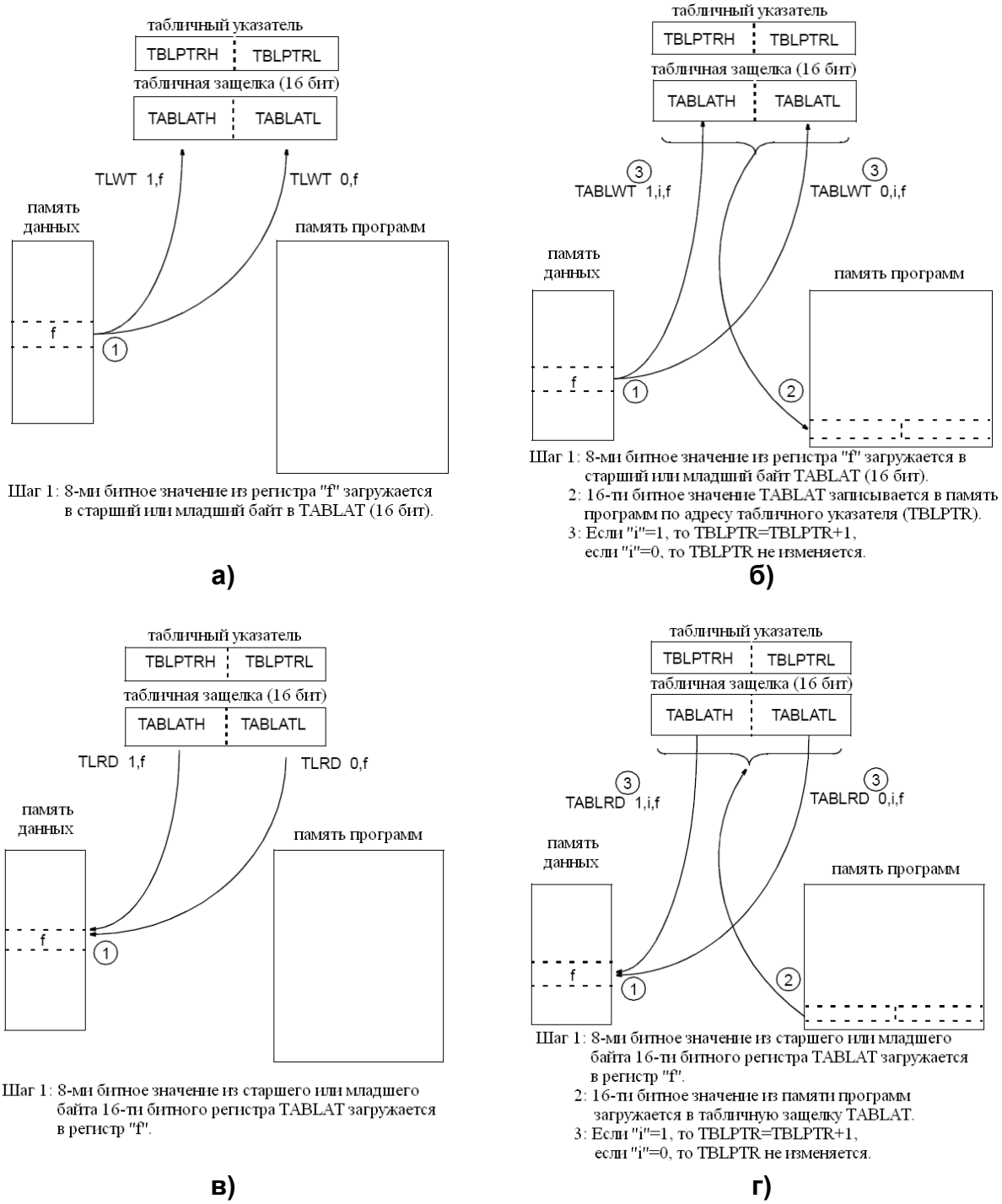


Рис. 19 Выполнение команд чтения/записи таблиц

а) Выполнение команды TLWT **б)** Выполнение команды TABLWT

в) Выполнение команды TLRD **г)** Выполнение команды TABLRD

Запись таблиц во внутреннюю память

Записи таблиц во внутреннюю память запускает операцию «длинной записи». «Длинная запись» необходима для программирования внутренней FLASH памяти. Выполнение команд останавливается во время цикла «длинной записи». «Длинная запись» будет закончена любым разрешенным прерыванием. Чтобы гарантировать, что ячейка памяти запрограммирована, требуется время программирования не менее: см. спецификацию. Для окончания «длинной записи» обычно разрешается только одно прерывание, чтобы гарантировать, что никакие другие прерывания

преждевременно не закончат «длинную запись». Последовательность событий для программирования ячейки внутренней памяти программ будет следующей:

1. Запретить все источники прерываний, за исключением прерывания для окончания записи.
2. Подать на вывод MCLR_Upp напряжение программирования.
3. Сбросить сторожевой таймер (WDT).
4. Произвести запись таблицы. Прерывание закончит длинную запись.
5. Верифицировать ячейку памяти (чтение таблицы).

При программировании необходимо выполнять требования спецификации. Нарушение спецификации (включая температуру) может привести к тому, что ячейка памяти будет запрограммирована не полностью и со временем может стереться. Если напряжение программирования (Upp) не подано, то команда записи таблицы будет выполнена за 2 цикла, и память программ не изменится.

Закончить операцию «длинной записи» могут только следующие события: источник прерывания или «сброс». Для окончания «длинной записи» по прерыванию, требуется чтобы было разрешено прерывание и был установлен флаг запроса прерывания.

Если для окончания «длинной записи» используется периферийное прерывание, то это прерывание должно быть разрешено и бит флага запроса прерывания должен быть установлен. Флаг запроса прерывания не сбрасывается при переходе по адресу вектора прерывания. Бит GLINTD определяет будет ли программа переходить к вектору прерывания после окончания «длинной записи». Если GLINTD сброшен, то программа переходит к вектору прерывания, если GLINTD установлен – не переходит**. Исключение составляет окончание «длинной записи» по прерыванию от «таймера 0».** В этом случае независимо от бита GLINTD программа продолжит выполнение, т.е. перехода к вектору прерывания не будет.

Длительность импульсов записи во флеш память задается 8-ми разрядным регистром WRDIF (коэффициент деления частоты генератора). Требуемое значение длительности 5 мкс. Коэффициент деления выбирается: K = 5 мкс/T_C, где T_C в мкс. Значение регистра после сброса равно 05h. Если предполагается производить запись во флеш память и частота генератора отличается от 1 МГц, то необходимо предварительно загрузить в этот регистр необходимое значение. Регистр доступен по чтению и записи.

Таблица 15

Воздействие прерываний на операцию «длинной записи»

источник прерываний	GLINTD	бит разрешения	флаг запроса	действие
PA0/INT, PA1/0CLK, TMR0(прим.), периферийные прерывания	0	1	1	Заканчивает длинную запись таблиц во внутреннюю память программ, переходит к вектору прерывания
	0	1	0	Нет
	1	0	x	Нет
	1	1	1	Заканчивает длинную запись таблиц, не переходит к вектору прерывания

Примечание:

В случае «таймера 0» переход к вектору прерывания не происходит.

Запись таблиц во внешнюю память

Команда записи таблиц во внешнюю память всегда выполняется за 2 цикла. Второй цикл записывает данные в ячейку внешней памяти. Последовательность событий для записи во внешнюю память та же самая, что и для записи во внутреннюю.

Пример 4
Запись таблицы

```
CLRWDT                          ; Очистка сторожевого таймера
MOVLW     HIGH (TBL_ADDR)       ; Загрузка адреса таблицы
MOVWF     TBLPTRH
MOVLW     LOW (TBL_ADDR)
MOVWF     TBLPTRL
MOVLW     HIGH (DATA)           ; Загрузка старшего байта в
TLWT      1,WREG                ; табличную защелку TABLATH
MOVLW     LOW (DATA)            ; Загрузка младшего байта в табличную защелку
TABLWT    0,0,WREG              ; TABLATL и запись в память программ (внешнюю)
```

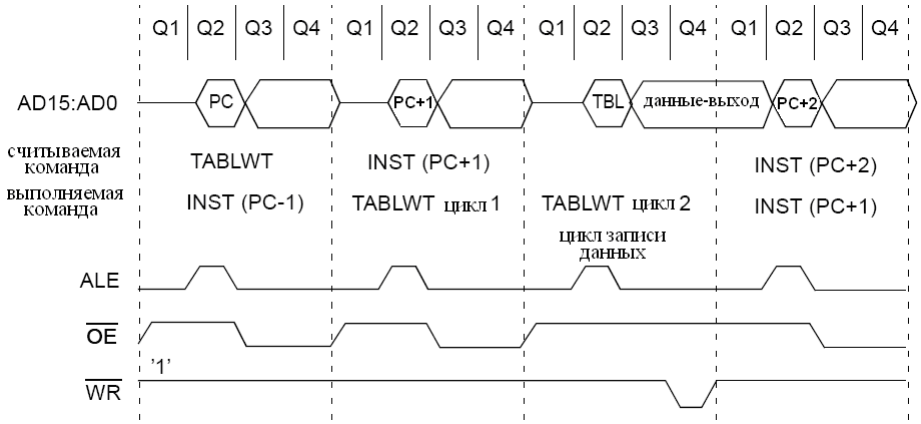


Рис. 20 Временная диаграмма выполнения команды TABLWT

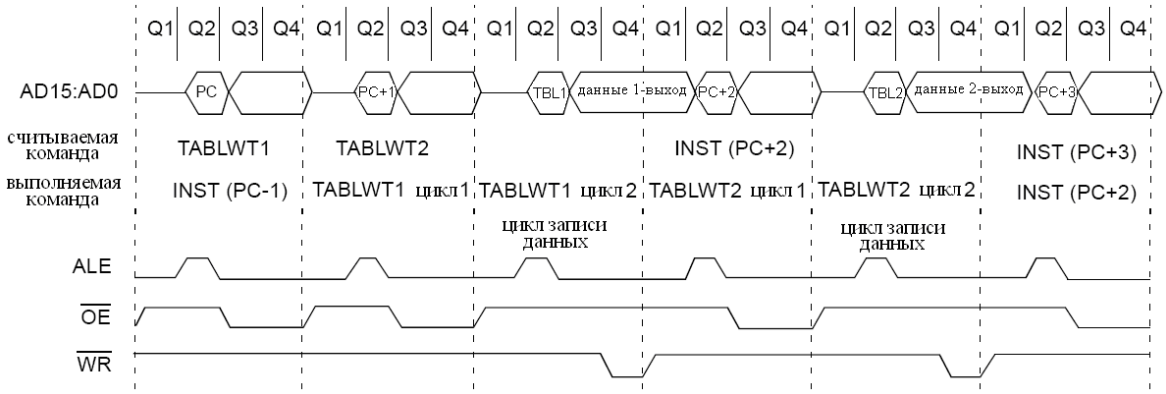


Рис. 21 Временная диаграмма последовательности команд TABLWT

Чтение таблиц

Операция чтения таблиц осуществляет чтение памяти программ. Это позволяет хранить константы в памяти программ и считывать их в память данных при необходимости. Пример 5 показывает считывание 16-ти битной величины из

памяти программ с адресом из TBLPTR. После того, как незначащий байт был считан из TABLATH, в TABLATH загружается 16-ти битная величина из памяти программ с адресом из TBLPTR, и значение TBLPTR инкрементируется. При первом чтении данные из памяти программ загружаются в защелку, а данные считываемые из защелки рассматриваются как незначащее (пустое) чтение (в «f» были загружены неизвестные данные). Режим косвенной адресации через INDF0 должен быть сконфигурирован либо с автоинкрементированием либо с автодекрементированием регистра указателя адреса FSR0.

Пример 5
Чтение таблицы

```
MOVLW     HIGH (TBL_ADDR)     ; Загрузка адреса таблицы
MOVWF     TBLPTRH
MOVLW     LOW (TBL_ADDR)
MOVWF     TBLPTRL
TABLRD    0,1,DUMMY           ; Пустое чтение из табличной защелки, чтение
                              ;памяти программ в табличную защелку,
                              инкрементирование TBLPTR
TLRD      1,INDF0             ; Чтение старшего байта из табличной защелки TABLATH
TABLRD    0,1,INDF0           ; Чтение младшего байта из табличной защелки TABLATL,
                              ; чтение памяти программ в табличную защелку и
                              ; инкрементирование TBLPTR
```

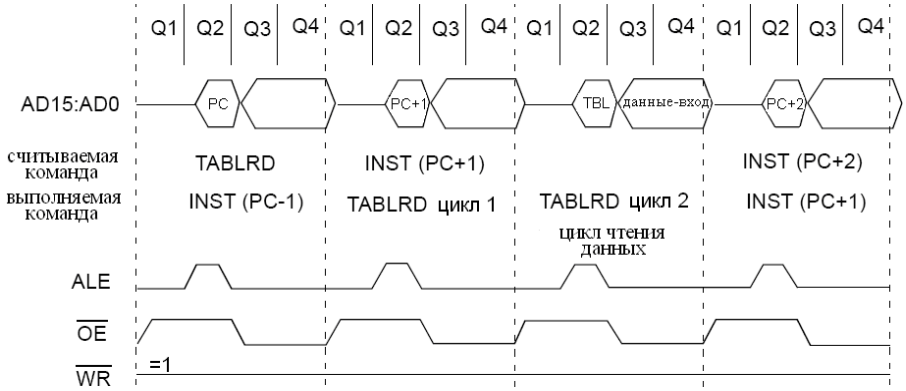


Рис. 22 Временная диаграмма выполнения команды TABLRD (внешняя память программ)

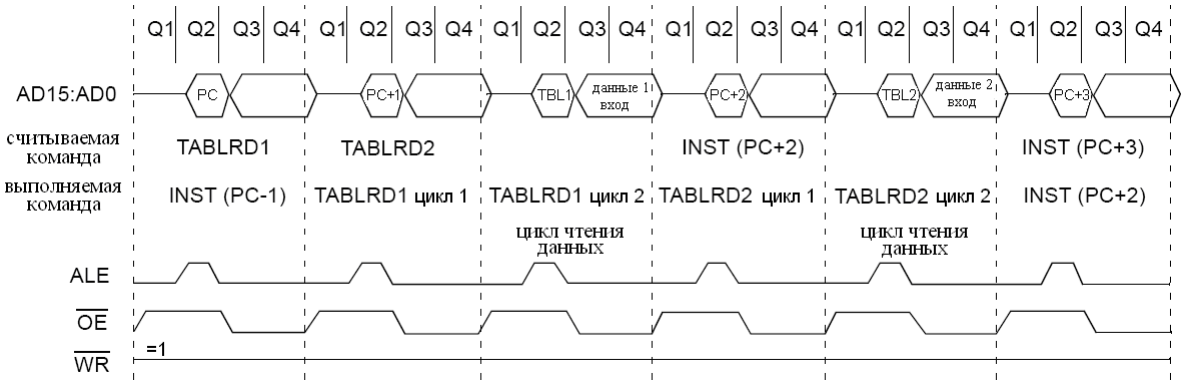


Рис. 23 Временная диаграмма последовательности команд TABLRD (внешняя память программ)

Аппаратный умножитель

Микроконтроллеры имеют 8x8 битный аппаратный умножитель, включенный в АЛУ прибора. Из-за того, что умножение реализовано аппаратно, оно выполняется за один цикл. Реализация выполнения умножения за один цикл обеспечивает более высокую вычислительную производительность и уменьшает требования к размеру кода для алгоритмов умножения. Увеличение производительности позволяет использовать прибор для применений, ранее предназначенных только для цифровых сигнальных процессоров.Беззнаковое умножение дает 16-ти битный результат. Результат хранится в 16-ти битном регистре PRODH:PRODL. Умножение не влияет ни на какие флаги в регистре ALUSTA. Регистры PRODH и PRODL доступны только для чтения.

Ниже приведено сравнение быстродействия микроконтроллеров, использующих аппаратно реализованное умножение с выполнением за один цикл, и производящих те же самые вычисления но без аппаратно реализованного умножения (см. Таблица 16).

Пример 6 показывает последовательность действий при 8x8 беззнаковом умножении. Требуется только одна команда, когда один аргумент для умножения уже загружен в регистр WREG.

Пример 7 приводит последовательность действий при 8x8 знаковом умножении. Для вычисления знаковых битов аргументов тестируется знаковый бит каждого аргумента и производится соответствующее вычитание. Результат хранится в регистрах RESH:PRODL.

Пример 9 приводит последовательность действий при 16x16 беззнаковом умножении. В примере 8 приводится используемый алгоритм. 32-х битный результат хранится в 4-х регистрах, RES3:RES0.

Пример 11 приводит последовательность действий при 16x16 знаковом умножении. В примере 10 приводится используемый алгоритм. 32-х битный результат хранится в 4-х регистрах, RES3:RES0. Для вычисления знаковых битов аргументов тестируется знаковый бит каждого аргумента и производится соответствующее вычитание.

Таблица 16

Сравнение производительности

	Метод умножения	**Объем программы (слов)**	**Кол-во циклов (максимум)**	**Время выполнения (мкс)**		
				33 МГц	**16 МГц**	**8 МГц**
8 x 8 без знака	без аппаратного умножителя	13	69	8.364	17.25	34.50
	аппаратный умножитель	1	1	0.121	0.25	0.50
8 x 8 со знаком	без аппаратного умножителя	-	-	-	-	-
	аппаратный умножитель	7	7	0.848	1.75	3.5
16 x 16 без знака	без аппаратного умножителя	21	242	29.333	60.50	121.0
	аппаратный умножитель	24	24	2.91	6.0	12.0
16 x 16 со	без аппаратного умножителя	52	254	30.788	63.50	127.0

знаком	аппаратный умножитель	36	36	4.36	9.0	18.0

Пример 6

Программа 8 x 8 битного беззнакового умножения

```
MOVFP   ARG1,WREG   ;
MULWF   ARG2        ; ARG1 * ARG2 -> PRODH:PRODL
```

Пример 7

Программа 8 x 8 битного умножения со знаком

```
MOVFP   ARG1,WREG
MULWF   ARG2        ; ARG1 * ARG2 -> PRODH:PRODL
MOVPF   PRODH,RESH  ; PRODH -> RESH
BTFSC       ARG2,SB     ; Тест бита знака
SUBWF   RESH,F      ; RESH = RESH - ARG1
MOVFP   ARG2,WREG
BTFSC       ARG1,SB     ; Тест бита знака
SUBWF   RESH,F      ; RESH = RESH - ARG2
```

Пример 8

Алгоритм 16 x 16 битного беззнакового умножения

RES3:RES0 = ARG1H:ARG1L * ARG2H:ARG2L
= (ARG1H * ARG2H * 2^{16}) + (ARG1H * ARG2L * 2^{8}) + (ARG1L * ARG2H * 2^{8})+(ARG1L * ARG2L)

Пример 9

Программа 16 x 16 битного беззнакового умножения

```
MOVFP   ARG1L,WREG
MULWF   ARG2L       ; ARG1L * ARG2L -> PRODH:PRODL
MOVPF   PRODH,RES1
MOVPF   PRODL,RES0

MOVFP   ARG1H,WREG
MULWF   ARG2H       ; ARG1H * ARG2H -> PRODH:PRODL
MOVPF   PRODH,RES3
MOVPF   PRODL,RES2

MOVFP   ARG1L,WREG
MULWF   ARG2H       ; ARG1L * ARG2H -> PRODH:PRODL
MOVFP   PRODL,WREG
ADDWF   RES1,F      ; Сложение промежуточных результатов
MOVFP   PRODH,WREG
ADDWFC  RES2,F
CLRF    WREG,F
ADDWFC  RES3,F

MOVFP   ARG1H,WREG
MULWF   ARG2L       ; ARG1H * ARG2L -> PRODH:PRODL
MOVFP   PRODL,WREG
ADDWF   RES1,F      ; Сложение промежуточных результатов
MOVFP   PRODH,WREG
ADDWFC  RES2,F
CLRF    WREG,F
ADDWFC  RES3,F
```

Пример 10

Алгоритм 16 x 16 битного умножения со знаком

RES3:RES0 = ARG1H:ARG1L * ARG2H:ARG2L
= (ARG1H * ARG2H * 2^{16})+(ARG1H * ARG2L * 2^{8})+(ARG1L * ARG2H * 2^{8})+(ARG1L * ARG2L)+
(-1 * ARG2H<7> * ARG1H:ARG1L * 2^{16}) + (-1 * ARG1H<7> * ARG2H:ARG2L * 2^{16})

Пример 11

Программа 16 x 16 битного умножения со знаком

```
        MOVFP       ARG1L,WREG
        MULWF       ARG2L           ; ARG1L * ARG2L -> PRODH:PRODL
        MOVPF       PRODH,RES1
        MOVPF       PRODL,RES0

        MOVFP       ARG1H,WREG
        MULWF       ARG2H           ; ARG1H * ARG2H -> PRODH:PRODL
        MOVPF       PRODH,RES3
        MOVPF       PRODL,RES2

        MOVFP       ARG1L,WREG
        MULWF       ARG2H           ; ARG1L * ARG2H -> PRODH:PRODL
        MOVFP       PRODL,WREG
        ADDWF       RES1,F          ; Сложение промежуточных результатов
        MOVFP       PRODH,WREG
        ADDWFC      RES2,F
        CLRF        WREG,F
        ADDWFC      RES3,F

        MOVFP       ARG1H,WREG
        MULWF       ARG2L           ; ARG1H * ARG2L -> PRODH:PRODL
        MOVFP       PRODL,WREG
        ADDWF       RES1,F          ; Сложение промежуточных результатов
        MOVFP       PRODH,WREG
        ADDWFC      RES2,F
        CLRF        WREG,F
        ADDWFC      RES3,F

        BTFSS       ARG2H,7         ; ARG2H:ARG2L отрицательно?
        GOTO        SIGN_ARG1       ; нет, проверка ARG1
        MOVFP       ARG1L,WREG
        SUBWF       RES2,F
        MOVFP       ARG1H,WREG
        SUBWFB      RES3,F

SIGN_ARG1
        BTFSS       ARG1H,7         ; ARG1H:ARG1L отрицательно?
        GOTO        CONT_CODE       ; нет, окончание
        MOVFP       ARG2L,WREG
        SUBWF       RES2,F
        MOVFP       ARG2H,WREG
        SUBWFB      RES3,F

CONT_CODE
```

Порты ввода-вывода

Микроконтроллеры имеют четыре порта ввода-вывода. Порты «C», «D» и «E» имеют регистр направления данных DDR, который используется для конфигурации выводов порта на вход или на выход. Некоторые выводы портов могут иметь дополнительное назначение.

Когда выводы портов сконфигурированы как выводы периферийного устройства, значение, содержащееся в регистре DDR неизвестно. После окончания работы с периферийным модулем пользователю желательно заново установить значение регистра DDR. Для некоторых других периферийных устройств (которые требуют входных выводов) требуется выставление битов направления передачи данных в регистре DDR.

Сигнал «сброса» переводит выводы в режим входа с высоким входным сопротивлением. Но некоторые периферийные модули могут внести изменения, такие например как перевод в режим аналогового входа или системной шины.

Регистр порта A и регистр направления данных DDRA

«Порт A» 4-х разрядный. Этот порт не имеет регистра направления данных (DDR). По сигналу «сброс», выводы «порта A» принудительно конфигурируются, как «вход» с высоким входным сопротивлением. Направление данных на выводах PA2/RX/DT и PA3/TX/CK, контролируется периферийным модулем. По сигналу «сброс», периферийный модуль неактивен, при этом выводы переведены в состояние «вход». При чтении «порта A» считывается состояние с выводов.

Вывод PA0/INT может работать как обычный вход или как вход внешнего прерывания. Вывод PA1/T0CLK может работать как обычный вход или как вход тактового сигнала для «таймера 0». Выводы PA2/RX/DT и PA3/TX/CK мультиплексированы с периферийным модулем USART. Настройка PA2 и PA3 как выходов, проводится автоматически периферийным модулем.

Таблица 17

Название	Бит	Тип входного буфера	Функция
PA0/INT	0	Триггер Шмитта	Вход порта или вход внешнего прерывания
PA1/T0CLK	1		Вход порта или вход тактового сигнала «таймера 0».
PA2/RX/DT	2		Вход порта или вход приемника асинхронного USART, или вход/выход данных синхронного USART
PA3/TX/CK	3		Вход порта или выход асинхронного передатчика USART, или вход/выход синхросигнала синхронного USART

Регистр порта C и регистр направления данных DDRC

«Порт C» - это 8-ми разрядный двунаправленный порт ввода/вывода. Направление данных управляется регистром направления DDRC. Значения «1» в регистре DDRC конфигурирует соответствующие выводы порта как вход. Значения

«0» в регистре DDRC конфигурирует соответствующие выводы порта как выход. При чтении регистра «порта C» (PORTC) считывается состояние с выводов, а при записи в регистр, значение записывается в регистр защелку. В режимах «микропроцессор» и «расширенный микроконтроллер» «порт C» является младшим байтом шины адреса/данных (AD7-AD0) системной шины. При работе контроллера внешней NAND Flash памяти, данный порт используется для организации шины данных к микросхеме NAND Flash. Данный порт имеет отдельный вывод задания напряжения питания $U_{cc1port}$. На который может подаваться напряжение от 3.0 до 5.5 В (но не более питания ядра U_{CC}). Таким образом порт может быть использован для работы с микросхемами 3.3 В логики.

Примечание:
«Порт C» и «порт D» имеют единое напряжение питания, поэтому выводы $U_{CC1port}$ и $U_{CC2port}$ должны быть объединены на печатной плате.

Таблица 18

Название	Бит	Тип входного буфера	Функция
PC0/AD0/IO0	0	ТТЛ	Вход/выход порта, или вывод системной шины адреса/данных, или вывод шины данных NAND Flash памяти
PC1/AD1/IO1	1		
PC2/AD2/IO2	2		
PC3/AD3/IO3	3		
PC4/AD4/IO4	4		
PC5/AD5/IO5	5		
PC6/AD6/IO6	6		
PC7/AD7/IO7	7		

Регистр порта D и регистр направления данных DDRD

«Порт D» - это 8-ми разрядный двунаправленный порт ввода/вывода. Направление данных управляется регистром направления DDRD. Значения «1» в регистре DDRD конфигурирует соответствующие выводы порта как вход. Значения «0» в регистре DDRD конфигурирует соответствующие выводы порта как выход. При чтении регистра «порта D» (PORTD) считывается состояние с выводов, а при записи в регистр, значение записывается в регистр защелку. В режимах «микропроцессор» и «расширенный микроконтроллер» «порт D» является старшим байтом шины адреса/данных (AD15-AD8) системной шины. При работе контроллера внешней NAND Flash памяти, данный порт используется для организации сигналов управления к микросхеме NAND Flash. Данный порт имеет отдельный вывод задания напряжения питания $U_{cc2port}$. На который может подаваться напряжение от 3.0 до 5.5 В (но не более питания ядра Ucc). Таким образом порт может быть использован для работы с микросхемами 3.3 В логики.

Примечание:
«Порт C» и «порт D» имеют единое напряжение питания, поэтому выводы $U_{CC1port}$ и $U_{CC2port}$ должны быть объединены на печатной плате.

Таблица 19

Название	Бит	Тип входного буфера	Функция
PD0/AD8/RBn	0	ТТЛ	Вход/выход порта, или вывод системной шины адреса/данных, или сигнал RBn для NAND Flash памяти
PD1/AD9/REn	1		Вход/выход порта, или вывод системной шины адреса/данных, или сигнал REn для NAND Flash памяти
PD2/AD10	2		Вход/выход порта, или вывод системной шины адреса/данных
PD3/AD11	3		Вход/выход порта, или вывод системной шины адреса/данных
PD4/AD12/CLE	4		Вход/выход порта, или вывод системной шины адреса/данных, или сигнал CLE для NAND Flash памяти
PD5/AD13/ALE	5		Вход/выход порта, или вывод системной шины адреса/данных, или сигнал ALE для NAND Flash памяти
PD6/AD14/WEn	6		Вход/выход порта, или вывод системной шины адреса/данных, или сигнал WEn для NAND Flash памяти
PD7/D15	7		Вход/выход порта, или вывод системной шины адреса/данных

Регистр порта E и регистр направления данных DDRE

«Порт E» - это 8 разрядный двунаправленный порт ввода/вывода. Направление данных определяется регистром направления DDRE. Значение «1» в регистре DDRE конфигурирует соответствующий вывод порта как вход. Значение «0» в регистре DDRE конфигурирует соответствующий вывод порта как выход. При чтении регистра «порта E» (PORTE) считывается состояние с выводов, а при записи в регистр, значение записывается в регистр защелку. В режимах «микропроцессор» и «расширенный микроконтроллер» «порт E» выводит управляющие сигналы для системной шины: разрешение защелки адреса (ALE), разрешение выхода (OE) и запись (WR). Активный уровень для сигналов OE и WR - низкий. В режиме тестирования USB_TEST2 этот порт используется для реализации USB интерфейса в обход аналогового приемопередатчика. Часть выводов «порт E» используются для интерфейса SPI.

Примечание:

В режимах «микропроцессор» и «расширенный микроконтроллер» выводы «порта E», не являющиеся управляющими сигналами системной шины, переключаются в режим входа.

Таблица 20

Название	Бит	Тип входного буфера	Функция
PE0/ALE/TXDP	0	ТТЛ	Вход/выход порта, или выход сигнала фиксации адреса внешней памяти, или цифровой вывод TXDP
PE1/OE/TXDM	1		Вход/выход порта, или выход сигнала чтения данных из внешней памяти, или цифровой вывод TXDM
PE2/WR/TXOE	2		Вход/выход порта, или выход сигнала разрешения записи во внешнюю память, или цифровой вывод TXOE
PE3	3		Вход/выход порта
PE4/RXD/SS	4		Вход/выход порта, или цифровой вывод RXD, или сигнал Slave Select SPI
PE5/RXDP/SDI	5		Вход/выход порта, или цифровой вывод RXDP, или сигнал Serial Data Input SPI
PE6/RXDM/SCK	6		Вход/выход порта, или цифровой вывод RXDM, или сигнал Serial Clock SPI
PE7/SDO	7		Вход/выход порта, или сигнал Serial Data Output SPI

Блок «таймер 0»

Блок «таймер 0» состоит из 16-разрядного таймер/счетчика. Старший байт представлен регистром TMR0H, а младший байт – регистром TMR0L. Оба регистра доступны по чтению и записи. Программно-управляемый 8-разрядный делитель частоты позволяет создать на основе блока 24-разрядный счетчик. Источник тактовых импульсов задается битом T0CS регистра T0STA. Счетчик может изменять свое состояние или от внутренних тактовых импульсов, или от внешних, подаваемых на вход PA1/T0CLK. Управление «таймером 0» осуществляется с помощью регистра T0STA.

Таблица 21

Регистр T0STA (адрес: 05h, не зависит от номера банка)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
INTEDG	**T0SE**	**T0CS**	**T0PS3**	**T0PS2**	**T0PS1**	**T0PS0**	**-**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**INTEDG**: бит выбора управляющего перепада сигнала на выводе PA0/INT для запроса прерывания. Этот бит выбирает, на фронте или спаде сигнала будет происходить запрос прерывания. 1 = фронт сигнала на выводе PA0/INT генерирует прерывание 0 = спад сигнала на выводе PA0/INT генерирует прерывание
бит 6	**T0SE:** бит выбора управляющего перепада сигнала при внешнем тактировании «таймера 0». Этот бит выбирает, на фронте или спаде сигнала «таймер 0» будет инкрементироваться. Когда T0CS=0 (внешнее тактирование): 1 = фронт сигнала на выводе PA1/T0CLK инкрементирует

	«таймер 0» и/или устанавливает T0CKIF - бит 0 = спад сигнала на выводе PA1/T0CLK инкрементирует «таймер 0» и/или устанавливает T0CKIF - бит Когда T0CS=1 (внутреннее тактирование): значение бита безразлично, установка бита T0CKIF не производится
бит 5	**T0CS:** бит выбора источника тактирования для «таймера 0». Этот бит выбирает источник синхронизации для «таймера 0». 1 = внутренняя тактовая частота с генератора циклов, прерывание от вывода RA1/T0CKI не формируется 0 = внешнее тактирование с вывода PA1/T0CLK, формирование прерывания от вывода RA1/T0CKI разрешено
бит 4-1	**T0PS3:T0PS0:** биты выбора предделителя для таймера 0. Эти биты позволяют выбрать величину деления входного тактового сигнала предделителем: 0000 - 1:1 0001 - 1:2 0010 - 1:4 0011 - 1:8 0100 - 1:16 0101 - 1:32 0110 - 1:64 0111 - 1:128 1xxx - 1:256
бит 0	не реализовано: читается значение равное нулю

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Если бит T0CS установлен в «1», то инкрементирование счетчика «таймера 0» происходит от тактовых импульсов внутреннего генератора, который является источником синхронизации для всего микроконтроллера. Если бит T0CS сброшен в «0», то инкрементирование счетчика происходит от тактовых импульсов с входа PA1/T0CLK (внешний источник тактовых импульсов). В случае использования внешнего источника тактовых импульсов, бит T0SE определяет полярность фронта, по которому изменяется состояние счетчика. Если бит T0SE установлен в «1», счетчик будет изменять свое состояние по переднему фронту сигнала PA1/T0CLK, а если бит T0SE сброшен в «0», то по заднему фронту (спаду) сигнала PA1/T0CLK. Предварительный 8-разрядный делитель с программируемым коэффициентом деления частоты (ПДПКД) осуществляет деление частоты тактовых импульсов в диапазоне от 1:1 до 1:256, в зависимости от состояния битов T0PS3:T0PS0. Таймер циклически изменяет свое состояние в диапазоне значений от 0000h до FFFFh с шагом 1. При достижении максимального значения (FFFFh) (состояние переполнения) устанавливается флаг (T0IF) запроса прерывания по переполнению «таймера 0». Этот запрос на обработку прерывания может быть замаскирован, путем сброса в «0» соответствующего бита разрешения запроса прерывания (T0IE) в регистре INTSTA. Флаг запроса на обработку прерывания от «таймера 0» (T0IF) сбрасывается программно программой обработки прерывания.

Если для «таймера 0» используется внешний источник тактовых импульсов, то осуществляется синхронизация внешнего тактового сигнала и внутренней тактовой частоты. Рис. 25 иллюстрирует механизм синхронизации. Тактовый сигнал синхронизируется после предделителя (ПДПКД). Сигнал с выхода предделителя (ПДПКД) сэмплируется два раза в каждом командном цикле с целью

детектирования появления переднего или заднего фронта. Требования к параметрам внешнего сигнала синхронизации приведены в таблице электрических параметров. Процедура синхронизации внешних тактовых импульсов, вносит задержку от времени прихода активного фронта до момента изменения таймером 0 своего состояния. На Рис. 25 показано, что эта задержка может составлять от $3*T_c$ до $7*T_c$.

Проблема считывания 16-разрядного значения регистров TMR0L и TMR0H заключается в том, что после считывания младшего (или старшего) байта, его значение может измениться от FFh к 00h. Для обеспечения однозначного считывания состояния счетчика рекомендуется маскировать сигнал запроса на обработку прерывания.

Запись в любой из регистров TMR0L и TMR0H блокирует изменение соответствующей части счетчика «таймера 0» в последующем после записи цикле микроконтроллера, при этом запись не оказывает влияния на другую часть счетчика. Поэтому рекомендуется последовательно производить запись сначала регистра TMR0L, а затем TMR0H. Запись в любой из регистров TMR0L или TMR0H сбрасывает в исходное состояние предделитель (ПДПКД).

Установка коэффициента деления предделителя (ПДПКД) полностью зависит от состояния регистра T0STA, то есть значение коэффициента может быть изменено «на лету» во время исполнения программы. Перед изменением коэффициента деления рекомендуется сбрасывать ПДПКД.

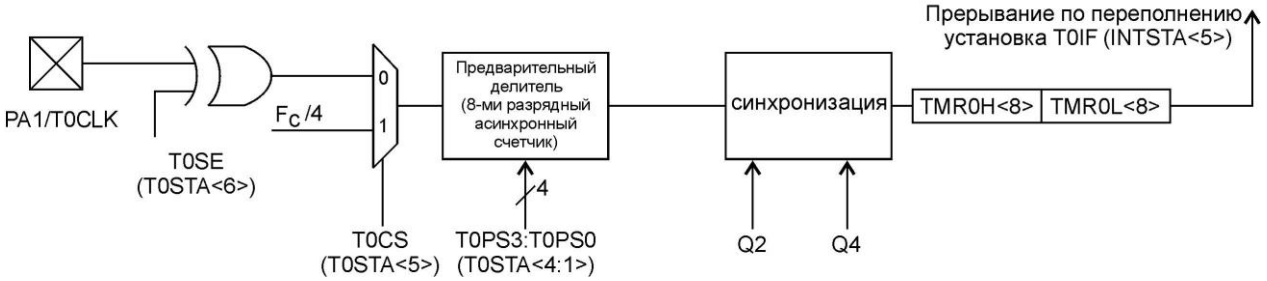


Рис. 24 Блок-схема модуля «таймер 0»

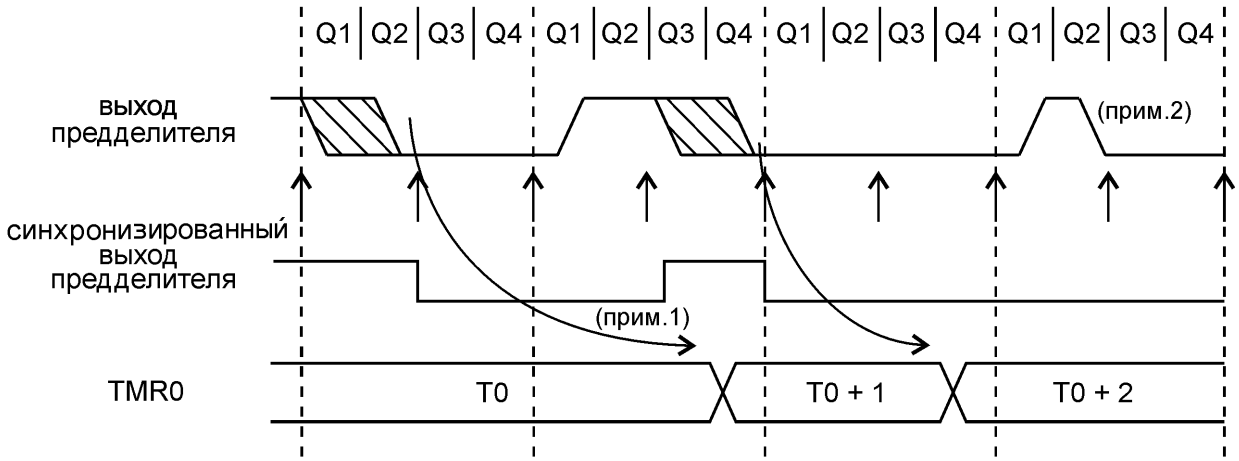


Рис. 25 Функционирование «таймер 0» от внешнего источника тактовых импульсов

Примечания:

1. Задержка от времени прихода активного фронта до момента изменения TMR0 своего состояния составляет от $3*T_c$ до $7*T_c$.
2. Длительность импульса на выходе ПДПКД меньше частоты синхронизации. В этом случае состояние счетчика TMR0 не изменится.

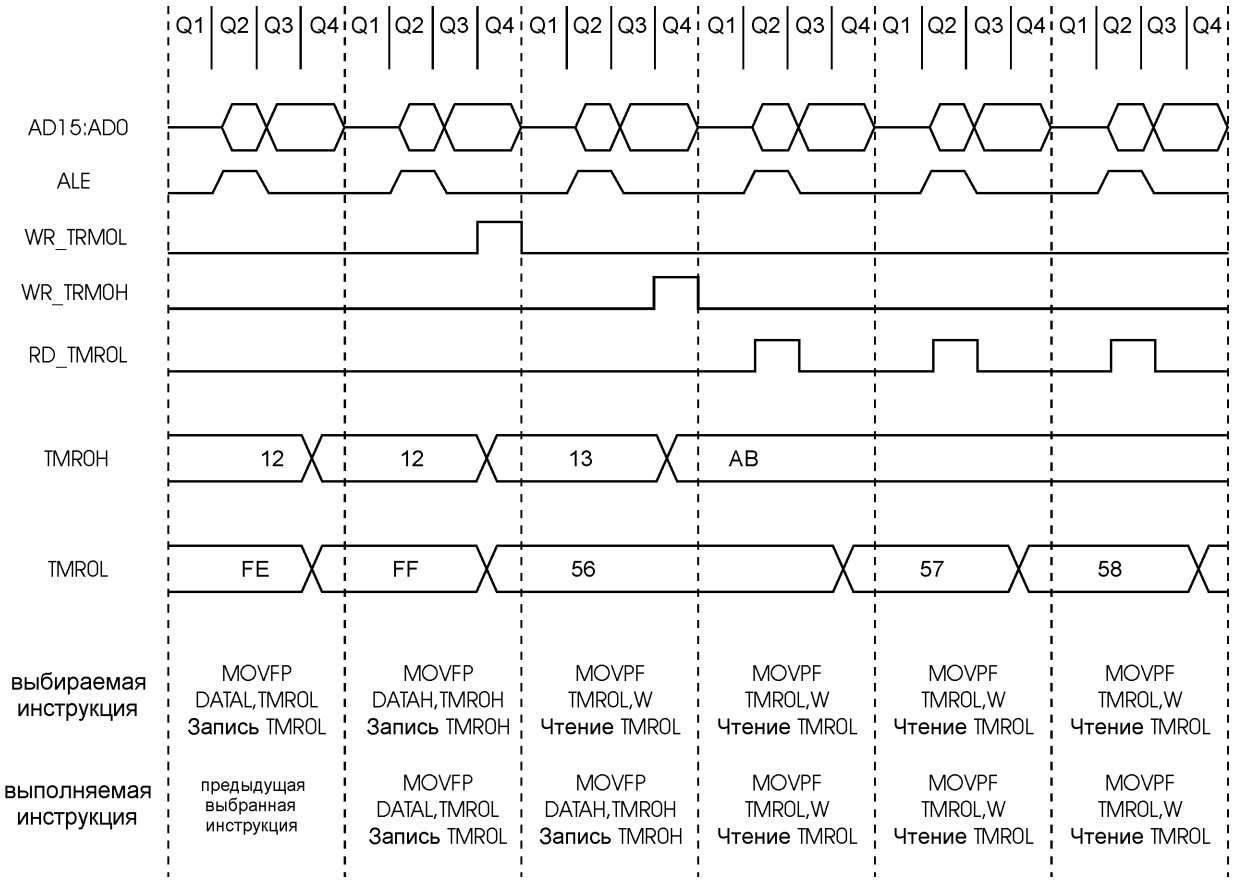


Рис. 26 Функционирование «таймер 0» при обращении к регистрам TMR0L и TMR0H

Примечание:
В примере записывается значение TMR0 равное AB56h.

Модуль синхронного последовательного порта SPI

Модуль ведущего синхронного последовательного порта представляет собой последовательный интерфейс SPI, применяемый для соединения с другими микроконтроллерами или периферийными микросхемами. Такими микросхемами могут быть последовательные EEPROM, сдвиговые регистры, драйверы дисплея, аналого-цифровые преобразователи и т. д.

Микропроцессорное ядро осуществляет работу с интерфейсом SPI посредством чтения и записи регистров блока, и обработки сигналов прерывания от блока. Блок содержит три программно доступных регистра:
SSPBUF - регистр принятых и передаваемых данных;
SSPSTAT - регистр состояния блока (см. Таблица 22);
SSPCON1 - регистр управления блоком (см. Таблица 23).

При окончании передачи данных модуль SPI может формировать сигнал запроса прерывания.

Вывод PE6/SCK служит для выдачи (в режиме ведущего) или приема (в режиме ведомого) сигналов синхронизации передаваемых данных. По линии PE7/SDO из модуля выгружаются данные, по линии PE5/SDI загружаются. Для выбора одного ведомого из нескольких может использоваться линия PE4/SS.

Таблица 22

SSPSTAT, регистр состояния блока SPI (адрес: 13h, банк 7)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	**CKE**	**-**	**-**	**-**	**-**	**-**	**BF**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**SMP:** выбор режима сэмплирования. Режим ведущего SPI: 1 = входные данные захватываются в конце периода выходной частоты синхронизации 0 = входные данные захватываются в середине периода выходной частоты синхронизации Режим ведомого SPI: SMP должен быть сброшен в «0» во время использования SPI в режиме ведомого
бит 6	**CKE:** Выбор фронта тактового сигнала SPI. При CKP=0: 1=Данные передаются по переднему фронту SCK 0=Данные передаются по заднему фронту SCK При CKP=1: 1=Данные передаются по переднему фронту SCK 0=Данные передаются по заднему фронту SCK
бит 5, 4, 3, 2, 1	зарезервировано
бит 0	**BF:** флаг заполнения буфера при приеме данных. 1 = Прием завершен, в SSPBUF находятся принятые данные 0 = Прием не завершен, SSPBUF пуст

Обозначения:

R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 23

SSPCON1, регистр управления блоком SPI (адрес: 11h, банк 7)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	**SSPOV**	**SSPEN**	**CKP**	**SSPM3**	**SSPM2**	**SSPM1**	**SSPM0**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**WCOL:** Флаг обнаружения конфликта записи. Режим ведущего: 1 = попытка записи в регистр SSPBUF в условиях не готовности блока к передаче данных. 0 = Нет конфликта. Режим ведомого: 1= попытка записи в SSPBUF в момент передачи ранее записанных данных (установка в «0» осуществляется программно). 0 = Нет конфликта

бит 6	**SSPOV:** Флаг обнаружения переполнения буфера при приеме данных. 1 = прием нового байта произошел до момента, когда ранее полученные данные не были считаны из регистра SSPBUF. В случае переполнения данные в SSPBUF теряются. Переполнение может происходить только в режиме ведомого. Чтобы предотвратить установку флага в режиме ведомого, пользователь должен считывать SSPBUF, даже если предполагается только передача данных. В режиме ведущего бит переполнения не устанавливается, т.к. каждый новый прием и передача данных инициализируется записью в SSPBUF. Установка в «0» осуществляется программно. 0 = переполнения нет
бит 5	**SSPEN:** включение модуля SPI. 1 = модуль SPI включен, выводы PE6/SCK, PE7/SDO, PE5/SDI и PE4/SS конфигурируются для использования модулем. 0 = модуль SPI выключен, выводы PE4 – PE7 работают как порты ввода/вывода
бит 4	**CKP:** Управление сигналом тактовой частоты. 1 = высокий уровень сигнала является не активным 0 = низкий уровень сигнала является не активным
биты 3, 2, 1, 0	**SSPM3 – SSPM0:** Выбор режима синхронного последовательного порта: 0000 = SPI режим ведущего, тактовый сигнал = Fc/4; 0001 = SPI режим ведущего, тактовый сигнал = Fc/16; 0010 = SPI режим ведущего, тактовый сигнал = Fc/64; 0011 = не используемая конфигурация; 0100 = SPI режим ведомого; тактовый сигнал с вывода PE6/SCK, разрешено управление выводом PE4/SS; 0101 = SPI режим ведомого, тактовый сигнал с вывода PE6/SCK, запрещено управление выводом PE4/SS, PE4/SS может использоваться как вывод порта входа/выхода ; 011x = не используемая конфигурация; 1xxx = не используемая конфигурация

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Режим SPI позволяет синхронно передавать и одновременно получать 8-разрядные данные. Поддерживаются 4 режима SPI. Для приема/передачи данных используются 3 вывода микроконтроллера: «выход последовательных данных» (SDO), «вход последовательных данных» (SDI) и «тактовый сигнал» (SCK). Для работы в режиме ведомого дополнительно может использоваться еще один вывод: «выбор ведомого» (SS).

Функционирование модуля SPI

На **Рис. 27** приведена блок-схема модуля SPI. Для инициализации режима SPI необходимо установить несколько контрольных разрядов регистра SSPCON1 (SSPCON1<5:0>) и SSPSTAT<7:6>. Выбранная конфигурация определяет следующие варианты функционирования:

- Режим ведущего (PE6/SCK - выход).
- Режим ведомого (PE6/SCK - вход).
- Активный уровень тактового сигнала (активный уровень SCK).
- Фаза выборки входных данных.
- Активный фронт тактового сигнала (передний/задний фронт SCK).
- Скорость передачи данных (только в режиме ведущего).
- Режим выбора ведомого (только в режиме ведомого).

Модуль состоит из приемо-передающего регистра сдвига SSPSR и буферного регистра SSPBUF. В SSPSR выполняется сдвиг данных в/из микроконтроллера старшим битом вперед. SSPBUF хранит данные, которые были записаны в SSPSR, до получения новых данных. Принятый байт данных перемещается в регистр SSPBUF. Далее устанавливаются флаг заполнения буфера BF (SSPSTAT<0>) и выставляется сигнал запроса на обработку прерывания SSPIF (PIR1<5>). Структура регистров SSPBUF и SSPSR позволяет хранить полученные данные в течение приема следующего байта данных. Любое обращение по записи к регистру SSPBUF во время передачи/приема данных будет игнорироваться, кроме того, будет установлен флаг обнаружения конфликта записи WCOL (SSPCON1<7>). Очистка флага WCOL в «0» должна быть осуществлена программно (обращением по записи к регистру SSPCON1).

При приеме данных в режиме SPI, регистр SSPBUF должен быть считан до момента окончания приема следующего байта. Установка флаг заполнения буфера, BF (SSPSTAT<0>), в «1» указывает, что SSPBUF загружен полученными данными (передача завершена). После обращения к регистру SSPBUF по чтению, бит BF автоматически сбрасывается в «0». Как правило, для определения окончания передачи или приема используется прерывание от модуля SPI. SSPBUF должен быть считан и записан в программе обработки прерываний.

Прямого доступа по чтению и записи к регистру SSPSR не существует, обращение к нему осуществляется через обращение к регистру SSPBUF.

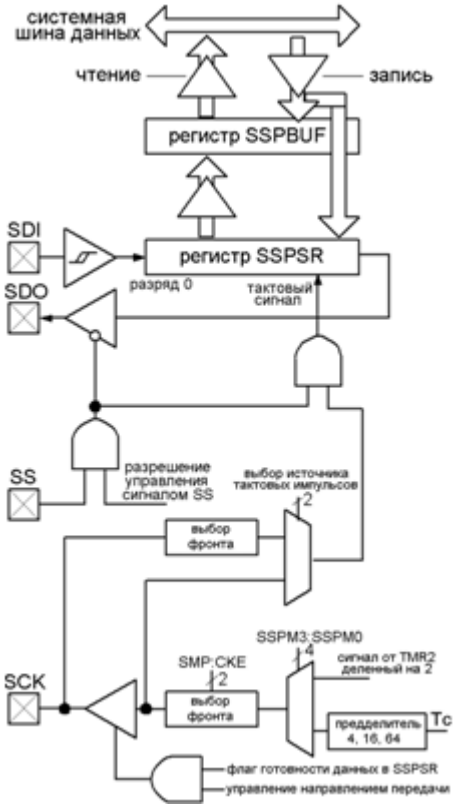


Рис. 27 Блок-схема модуля SPI

Настройка выводов при работе с модулем SPI. Типовое включение

Для включения модуля SPI необходимо установить бит SSPEN (регистр SSPCON1<5>) в «1». Для сброса или перенастройки модуля, необходимо сбросить в «0» бит SSPEN, перенастроить регистры SSPCON1, затем заново установить бит SSPEN в «1». Это позволит сконфигурировать выводы PE5/SDI, PE7/SDO, PA6/SCK и PA4/SS для использования в качестве выводов интерфейса SPI. Регистров управления направлением выводов (DDR) настраивать не требуется.

Главный контроллер инициализирует передачу данных, формируя тактовый сигнал SCK. Данные сдвигаются по установленному фронту тактового сигнала и принимаются на противоположном фронте тактового сигнала. Оба микроконтроллера должны иметь одинаковую конфигурацию режима SPI, в части состояния разряда CKP (управление сигналом тактовой частоты).

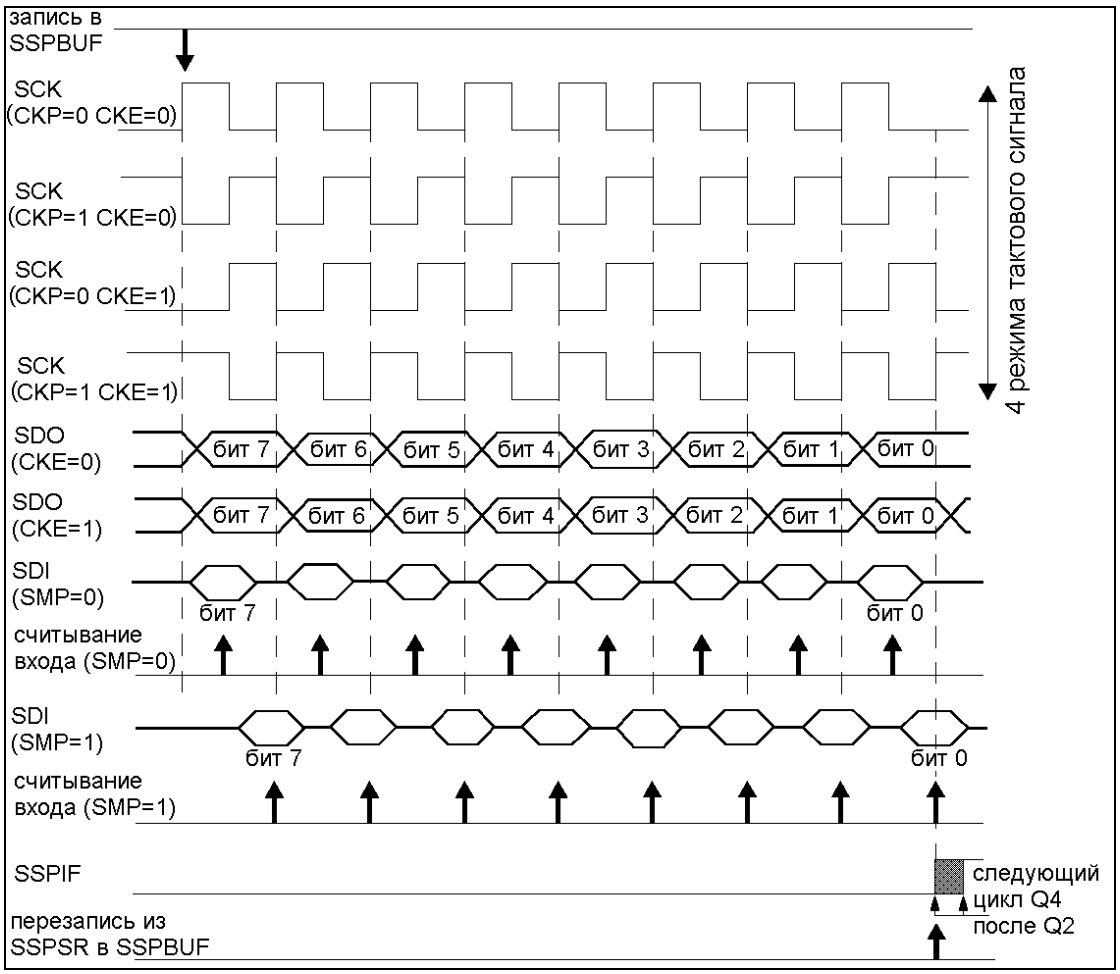


Рис. 28 Работа модуля SPI в режиме ведущего

Работа модуля SPI в режиме ведущего

Модуль SPI в режиме ведущего может инициировать передачу данных в любое время, т.к. в этом случае он контролирует линию SCK. Обмен данными инициируется путем обращения по записи к регистру SSPBUF. Ведущий определяет, когда ведомый может пересылать данные в соответствии с используемым протоколом. В режиме ведущего, данные считаются переданными/принятыми, как только произошла запись в регистр SSPBUF из сдвигового регистра. Данные со входа SDI последовательно (потактно) вдвигаются в регистр SSPSR со скоростью, определяемой регистрами управления. При получении 8 бит (байта), полученный байт загружается в регистр SSPBUF, при этом формируются сигнал запроса на обработку прерывания и определенные флаги.

Управление сигналом тактовой частоты, выбор активного фронта, моменты передачи и приема данных производится с помощью записи разрядов CKP регистра SSPCON1, CKE и SMP регистра SSPSTAT, возможные варианты передачи/приема данных показаны на

Рис. **28**, Рис. 30, Рис. 31.

Рис. **28** показывает варианты передачи данных для режима ведущего. Например, если бит CKE установлен в «1», то данные на выходе PE7/SDO устанавливаются еще до момента первого фронта на линии SCK. Момент приема входных данных определяется разрядом SMP. Кроме того, рисунок демонстрирует момент загрузки SSPBUF полученными данными. Данные всегда передаются

старшим разрядом вперед. Существуют следующие варианты выбора тактовой частоты SPI в режиме ведущего:

- Fc/4 (или Tic);
- Fc/16 (или 4 • Tic);
- Fc/64 (или 16 • Tic).

Таким образом, максимальная частота передачи данных составляет 8.25 МГц при тактовой частоте микроконтроллера 33 МГц.

Работа модуля SPI в режиме ведомого

В режиме ведомого данные передаются и принимаются по внешнему тактовому сигналу поступающего со входа PE6/SCK. При приеме последнего разряда, устанавливается флаг прерывания SSPIF (PIR1<5>). Параметры внешнего тактового сигнала должны удовлетворять требованиям, описанным в разделе электрических характеристик. Ведомый может передавать/получать данные, находясь в режиме покоя (Sleep режим). При получении байта данных микроконтроллер выходит из режима покоя.

Выбор синхронного режима ведомого

Наличие вывода PE4/SS позволяет использовать модуль SPI в режиме синхронного ведомого. В данном случае разряды регистра SSPCON1<3:0> должны иметь значение 04h, то есть установлена конфигурация «SPI режим ведомого, тактовый сигнал с вывода PE6/SCK, разрешено управление выводом PE4/SS». Если на выводе PE4 (SS) подан низкий логический уровень, передача и прием разрешены, вывод PE7/SDO управляется модулем SPI. Когда на вывод PE4/SS подан высокий уровень сигнала, вывод PE7/SDO отключен и находится в «третьем» состоянии. При таком использовании модуля рекомендуется применять внешние подтягивающие резисторы. Кроме того, подача логической единицы на вход PE4/SS фактически означает сброс модуля SPI. Сброс модуля SPI означает сброс счетчика битов сдвигового регистра.

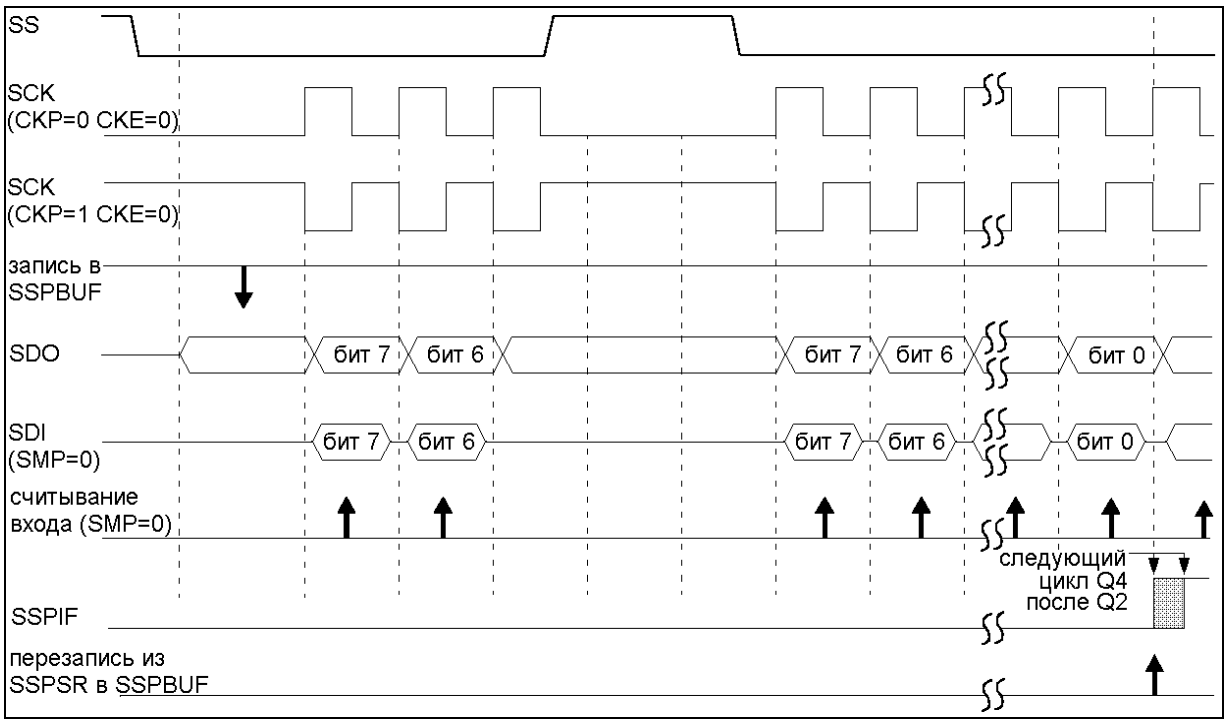


Рис. 29 Демонстрация работы в режиме синхронного ведомого

Функционирование в режиме покоя (Sleep режим). Эффект сброса.

Так как в режиме покоя системный тактовый сигнал отсутствует, модуль SPI, включенный в режиме ведущего, не может передавать или получать данные и будет сохранять свое текущее состояние до выхода микроконтроллера из спящего режима.

В режиме ведомого, сдвиговый регистр функционирует асинхронно по отношению к системной частоте микроконтроллера, что позволяет осуществлять прием/передачу данных в момент нахождения микроконтроллера в режиме покоя. По приему 8 бит, устанавливается флаг прерывания от модуля SPI, который выводит микроконтроллер из спящего режима.

Любой сброс отключает модуль SPI и прерывает текущую передачу.

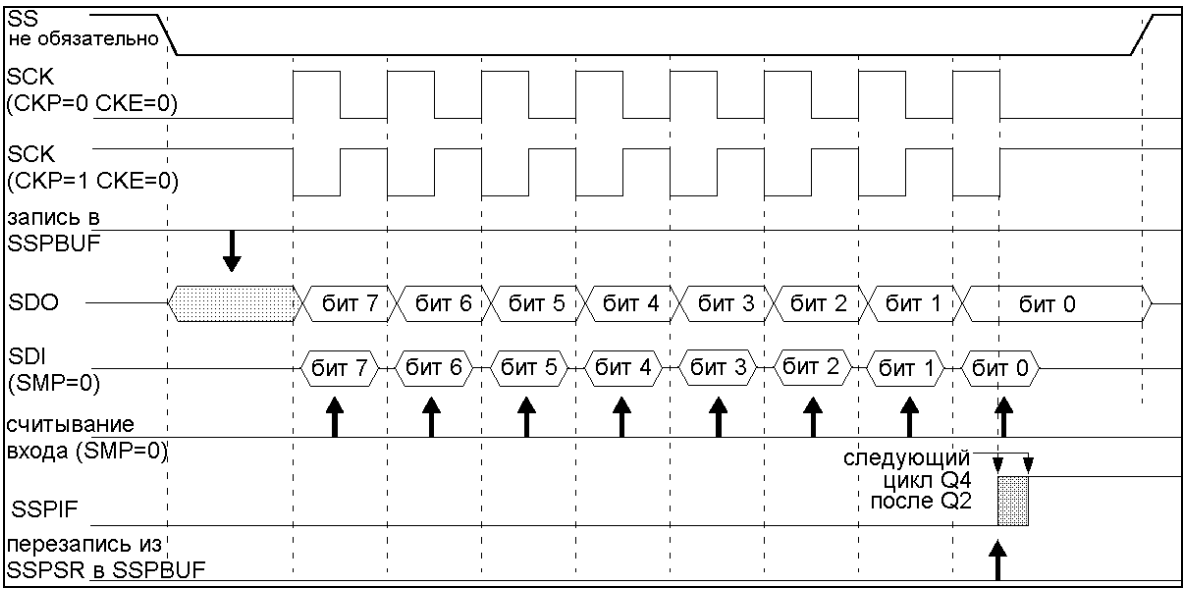


Рис. 30 Работа модуля SPI в режиме ведомый (CKE = 0)

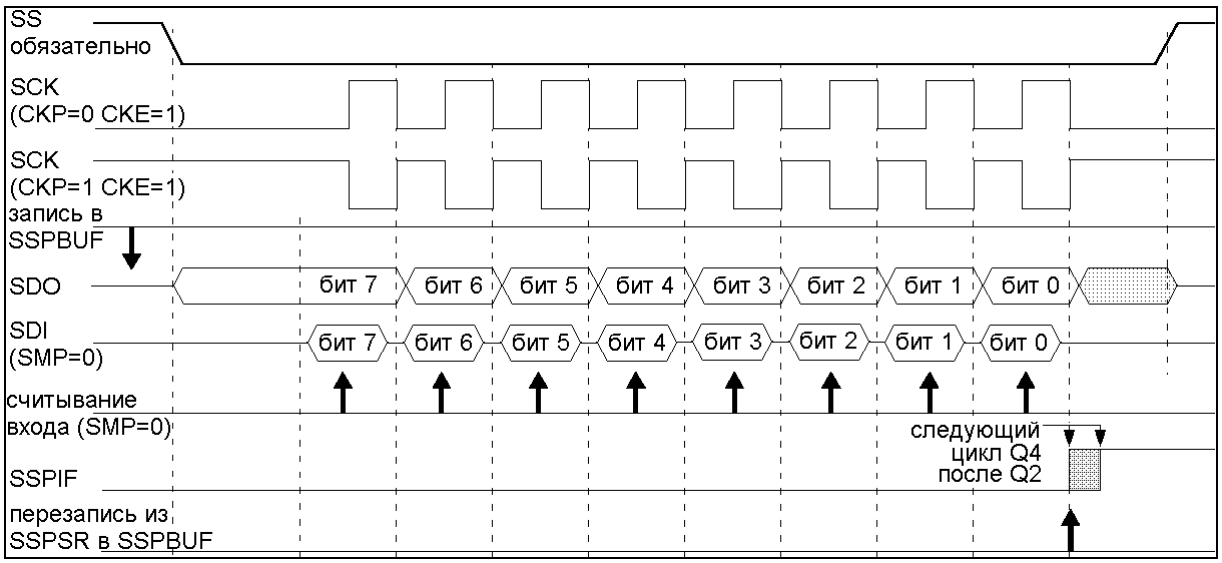


Рис. 31 Работа модуля SPI в режиме ведомый (CKE = 1)

Модуль универсального синхронно-асинхронного приемопередатчика

Микроконтроллер содержит один модуль синхронно-асинхронного приемопередатчика USART. Универсальный синхронно-асинхронный приемопередатчик может работать в следующих режимах:

- асинхронный (полный дуплекс);
- синхронный ведущий (полудуплекс);
- синхронный ведомый (полудуплекс).

Бит SPEN (RCSTA<7>) должен быть установлен, чтобы выводы PA2/RX/DT и PA3/TX/CK сконфигурировались как выводы последовательного интерфейса. Модуль USART будет управлять направлением выводов PA2/RX/DT и PA3/TX/CK, в зависимости от состояния битов конфигурации в регистрах RCSTA и TXSTA. Следующие биты контролируют направление выводов: SPEN, TXEN, SREN, CREN, CSRC.

Таблица 24

Регистр статуса и режима работы передатчика TXSTA (адрес: 15h, банк 5)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R-1	R/W-0
CSRC	**TX9**	**TXEN**	**SYNC**	**-**	**-**	**TRMT**	**TX9D**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**CSRC:** выбор источника тактовых импульсов Синхронный режим: 1 = режим ведущего (внутренний тактовый сигнал из BRG) 0 = режим ведомого (от внешнего источника тактового сигнала) Асинхронный режим: Не имеет значения
бит 6	**TX9:** выбор 9-ти разрядной передачи. 1 = выбирает 9-ти разрядную передачу 0 = выбирает 8-ми разрядную передачу
бит 5	**TXEN:** разрешение передачи 1 = передача разрешена 0 = передача отключена

	Бит SREN/CREN блокирует TXEN в синхронном режиме.
бит 4	**SYNC:** бит выбора режима USART (синхронный/асинхронный) 1 = синхронный режим 0 = асинхронный режим
бит 3,2	Не реализованы, читаются как «0»
бит 1	**TRMT:** флаг заполненности сдвигового регистра передатчика (TSR) 1 = регистр пуст 0 = регистр заполнен
бит 0	**TX9D:** 9-й бит передаваемых данных (может использоваться для программной реализации проверки четности)

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 25

Регистр статуса и режима работы приемника RCSTA (адрес: 13h, банк 5)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R-0	R-0	R-0
SPEN	**RX9**	**SREN**	**CREN**	**-**	**FERR**	**OERR**	**RX9D**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**SPEN:** бит разрешения работы последовательного порта 1 = конфигурирует PA3/TX/CK и PA2/RX/DT как выводы последовательного порта USART 0 = последовательный порт отключен
бит 6	**RX9:** выбор 9-ти разрядного приема 1 = выбирает 9-ти разрядный прием 0 = выбирает 8-ми разрядный прием
бит 5	**SREN:** бит разрешения однократного приема. Этот бит разрешает прием одного байта и после его приема автоматически сбрасывается. Синхронный режим: 1 = разрешить прием 0 = запретить прием Примечание: бит игнорируется в синхронном режиме ведомого. Асинхронный режим: Не имеет значения
бит 4	**CREN:** бит разрешения продолжительного приема. Этот бит разрешает непрерывный прием последовательно передаваемых данных. Асинхронный режим: 1 = разрешает непрерывный прием 0 = запрещает непрерывный прием Синхронный режим: 1 = разрешает непрерывный прием до момента сброса CREN (CREN отменяет SREN) 0 = отключает непрерывный прием

бит 3	Не реализовано, читается как «0»
бит 2	**FERR:** бит ошибки кадрирования 1 = есть ошибка (сбрасывается при чтении регистра RCREG) 0 = нет ошибки
бит 1	**OERR:** бит ошибки переполнения внутреннего буфера. 1 = есть ошибка (сбрасывается при сбросе бита CREN) 0 = нет ошибки
бит 0	**RX9D:** 9-й бит принятых данных (может использоваться для программной реализации проверки четности)

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 26

Регистр данных приемника RCREG (адрес: 14h, банк 5)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** данные, полученные по USART					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 27

Регистр данных передатчика TXREG (адрес: 16h, банк 5)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** данные, передаваемые по USART					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Генератор скорости передачи данных

Генератор скорости передачи данных (BRG) поддерживает асинхронный и синхронный режимы USART. Это отдельный 8-ми разрядный таймер/счетчик. Его период задается значением в регистре SPBRG. Скорость передачи рассчитывается по следующей формуле:

- **скорость передачи для асинхронного режима** = FC / (32*((SPBRG)+1)),
- **скорость передачи для синхронного режима** = FC / (4*((SPBRG)+1)), значение (SPBRG) от 0 до 255.

Запись нового значения в SPBRG приводит к сбросу таймера BRG. Это обеспечивает то, что генератор сразу переключается на новую скорость передачи. После сигнала «сброс» регистр SPBRG очищается, поэтому его необходимо загружать требуемым значением после каждого сброса.

Таблица 28

Регистр задания скорости приема и передачи SPBRG (адрес: 17h, банк 5)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BRG7	**BRG6**	**BRG5**	**BRG4**	**BRG3**	**BRG2**	**BRG1**	**BRG0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**BRG7…BRG0**: данные, определяющие скорость приема и передачи по USART					

Обозначения:

R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

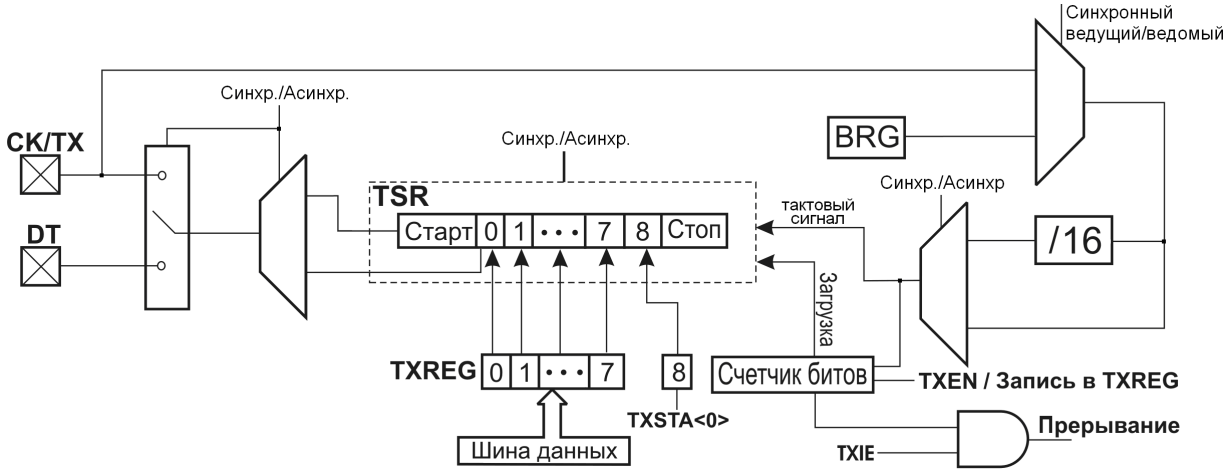


Рис. 32 Блок-схема передатчика

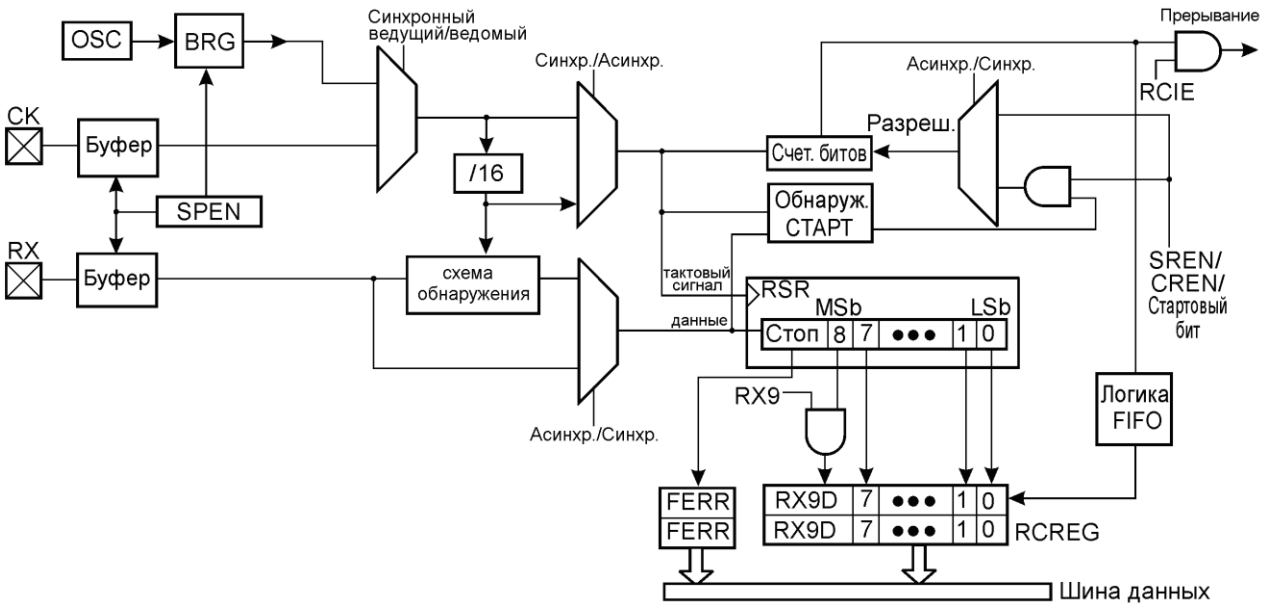


Рис. 33 Блок-схема приемника

Асинхронный режим

В этом режиме USART использует стандартный формат NRZ (один стартовый бит, восемь или девять информационных битов и один стоповый бит). Самый распространенный формат данных – это 8-ми битный. Внутрипроцессорный генератор скорости передачи может использоваться для получения стандартных частот скорости передачи. Приемник и передатчик USART являются функционально независимыми, но используют одинаковый формат данных и скорость передачи. Проверка четности не поддерживается аппаратными средствами, но может быть реализована программно (используя девятый бит данных). Модуль USART в асинхронном режиме останавливается во время SLEEP (в режиме ожидания). Асинхронный режим выбирается сбросом бита SYNC (TXSTA<4>).

Модуль USART в асинхронном режиме состоит из следующих компонентов:

- Генератор скорости передачи.
- Схема выборки.
- Асинхронный приемник.
- Асинхронный передатчик.

Асинхронный передатчик

Блок-схема передатчика представлена на Рис. **32**. Основой передатчика является сдвиговый регистр передачи (TSR). Он получает информацию из буфера передатчика (TXREG). В буфер TXREG данные загружаются программно. Сдвиговый регистр TSR загружается новыми данными из TXREG (если они есть) как только передастся стоповый бит предыдущей посылки данных. Это происходит в последнем командном цикле периода BRG. При этом устанавливается флаг запроса прерывания TXIF индицирующий, что TXREG пуст. Это прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита TXIE. Флаг запроса прерывания TXIF устанавливается независимо от значения TXIE. Он не может быть сброшен программно. Флаг сбрасывается аппаратно при загрузке новых данных в TXREG. Бит TRMT

(TXSTA<1>) индицирует состояние сдвигового регистра TSR. Бит устанавливается когда TSR пуст. TRMT доступен только для чтения и не может вызывать прерывания. Регистр TSR не отображается в памяти данных, т.е. не доступен для чтения/записи.

Передача разрешается, когда устанавливается бит TXEN (TXSTA<5>). Фактически передача не начнется: пока данные не будут загружены в TXREG и генератор скорости передачи не выдаст сдвиговый синхроимпульс (см. Рис. **34**). Передачу также можно начать сначала загрузив TXREG, а потом установив бит TXEN. Обычно, если передача начинается в первый раз, TSR пуст, поэтому запись данных в TXREG повлечет немедленную их передачу в TSR, освободив TXREG. Поэтому возможна неразрывная последовательная передача (см. Рис. 35). Сброс TXEN во время передачи вызовет отмену передачи, сброс передатчика.

Для выбора 9-ти битной передачи, бит TX9 (TXSTA<6>) должен быть установлен в единицу. Девятое значение бита записывается в TX9D (TXSTA<0>) перед записью 8-ми битных данных в TXREG, так как запись данных в TXREG может вызвать немедленную передачу данных в TSR (если TSR пуст).

Шаги для настройки асинхронной передачи следующие:

1. записать значение в регистр SPBRG для задания скорости передачи.
2. включить асинхронный последовательный порт (бит SYNC=0 и бит SPEN=1).
3. если требуются прерывания, тогда установите бит TXIE.
4. если требуется 9-ти битная передача, тогда установите бит TX9.
5. если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9D.
6. загрузите данные в регистр TXREG.
7. запустите передачу установкой бита TXEN.

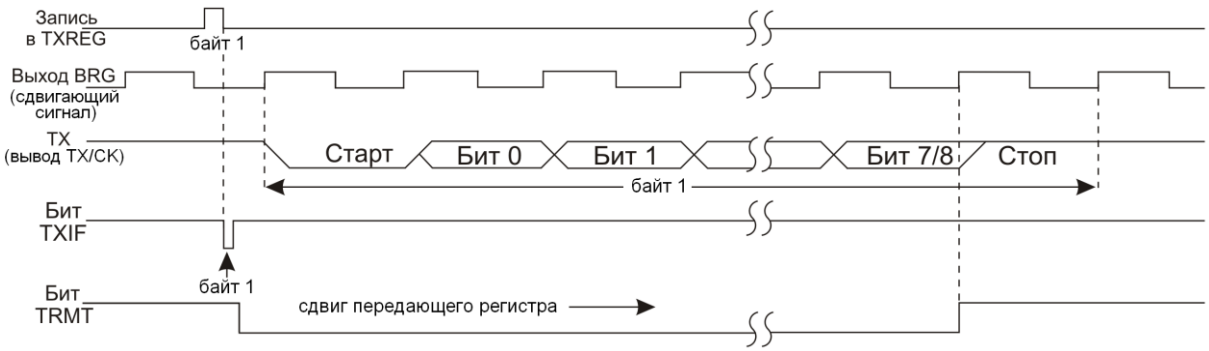


Рис. 34 Асинхронная передача

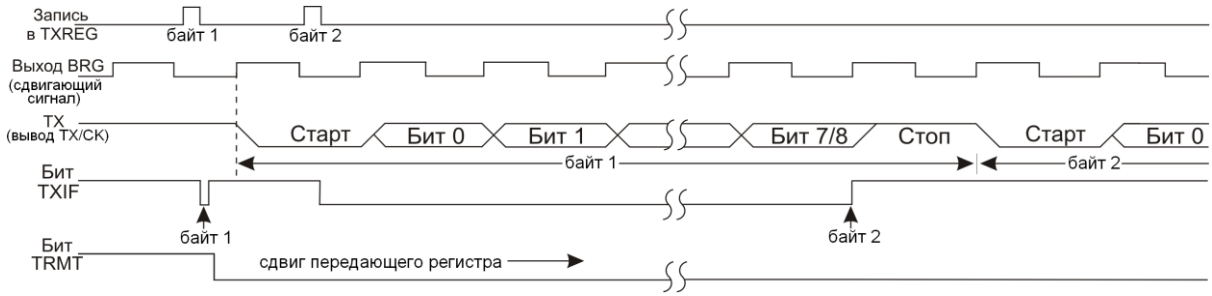


Рис. 35 Асинхронная неразрывная последовательная передача

Асинхронный приемник

Блок-схема приемника показана на

Рис. **33**. Данные с вывода PA2/RX/DT подаются в блок восстановления данных. Это высокоскоростной сдвиговый регистр, работающий на частоте в 16 раз выше скорости передачи, в то время как основной сдвиговый регистр принимаемых данных работает со скоростью передачи.
Если выбран асинхронный режим, то приемник включается установкой бита CREN (RCSTA<4>).

Основой приемника является сдвиговый регистр приема (RSR). После захвата стопового бита, принятые данные из RSR передаются в RCREG (если он пуст), после чего устанавливается флаг запроса прерывания RCIF. Прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита RCIE. RCIF доступен только для чтения, он сбрасывается аппаратно когда считываются данные из RCREG и регистр пуст. Регистр RCREG имеет двойную буферизацию, т.е. можно принять два байта данных в RCREG FIFO и третий байт начать принимать в RSR. При обнаружении стопового бита третьего байта, если RCREG по-прежнему не считан, устанавливается бит ошибки переполнения приемника OERR (RCSTA<1>). Данные в RSR будут потеряны. Для извлечения двух байт RCREG должен считываться дважды. Бит OERR должен быть очищен программно сбросом приема (сбросом бита CREN). Пока бит OERR установлен, приемник не работает. Флаг ошибки кадрирования FERR (RCSTA<2>) устанавливается, если невозможно обнаружить стоповый бит. Флаг FERR и девятый бит данных буферизуются также как и принятые данные. Поэтому необходимо считать регистр RCSTA перед считыванием RCREG, чтобы не потерять прежнюю информацию FERR и RX9D.

Данные с вывода PA2/RX/DT сканируются, три раза в каждом такте приема, мажоритарной схемой обнаружения, для выявления уровня сигнала на выводе PA2/RX/DT. Сканирование осуществляется на седьмом, восьмом и девятом заднем фронте (спаде) импульсов частотой в 16 раз превышающей частоту приема (См. Рис. 36).

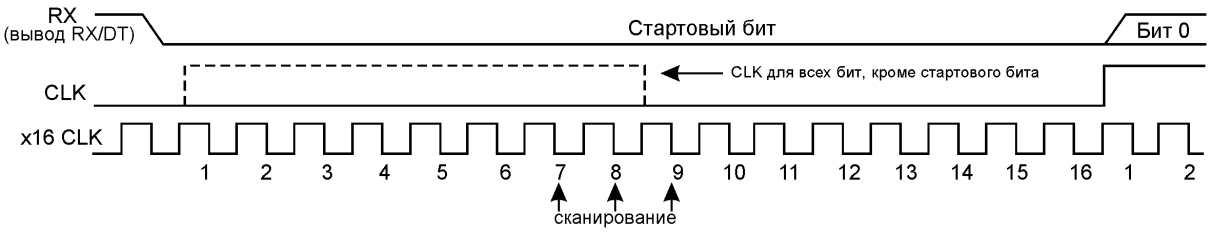


Рис. 36 Схема сканирования вывода PA2/RX/DT

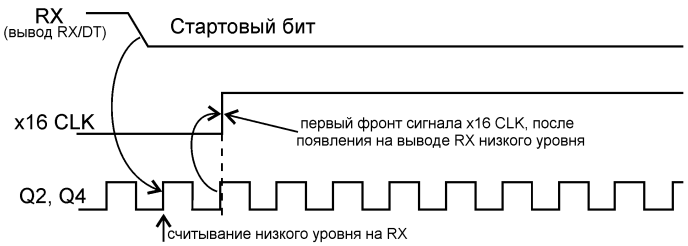


Рис. 37 Обнаружение стартового бита

Шаги для настройки асинхронного приема следующие:

1. записать значение в регистр SPBRG для задания скорости приема.
2. включить асинхронный последовательный порт (бит SYNC=0 и бит SPEN=1).
3. если требуются прерывания, тогда установите бит RCIE.
4. если требуется 9-ти битный прием, тогда установите бит RX9.
5. разрешите прием установкой бита CREN.

6. при завершении приема установится бит RCIF, и произойдет прерывание (если установлен бит RCIE).
7. считайте RCSTA для получения значения девятого бита (для 9-ти битного приема) и бит FERR для определения ошибки.
8. считайте 8-ми битные принятые данные из регистра RCREG.
9. если произошла ошибка переполнения, сбросьте бит OERR.

Для отмены приема, сбросьте либо биты SREN и CREN, либо бит SPEN. Это сбросит логику приема, но не изменит настройки.

Синхронный ведущий режим

В синхронном ведущем режиме данные передаются полудуплексным способом, то есть прием и передача происходят не одновременно: при передаче данных прием запрещен, и наоборот. Синхронный режим включается при установке бита SYNC (TXSTA<4>). Бит SPEN (RCSTA<7>) устанавливается, чтобы сконфигурировать выводы: PA3/CK - линия тактовых импульсов и PA2/DT - линия данных. Ведущий режим означает, что процессор формирует тактовые импульсы на линии PA3/CK. Ведущий режим выбирается установкой бита CSRC (TXSTA<7>).

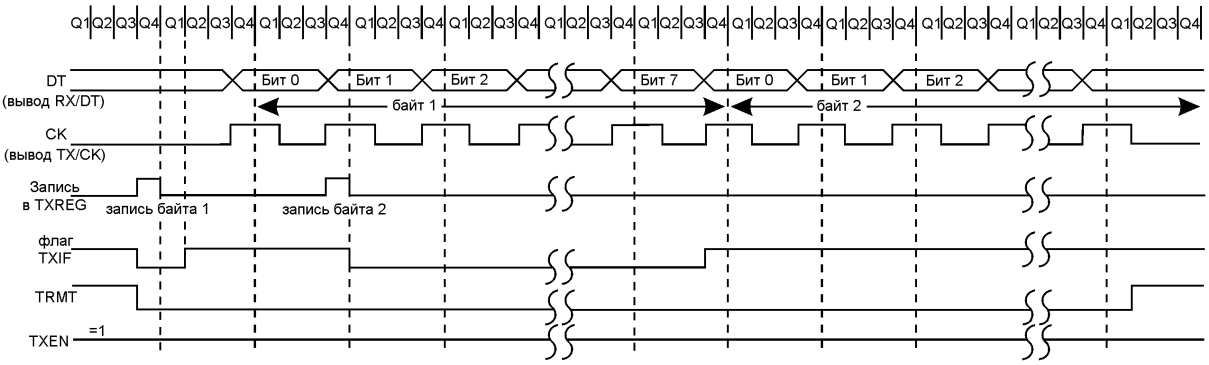


Рис. 38 Синхронная передача в ведущем режиме

Передача данных в синхронном ведущем режиме

Блок-схема передатчика показана на
Рис. **32**. Основой передатчика является сдвиговый регистр передачи (TSR). Сдвиговый регистр получает данные из буфера передатчика TXREG. Данные загружаются в TXREG программно. После передачи последнего бита предыдущей посылки, TSR загружается новыми данными из TXREG (если они есть). Это происходит в последнем командном цикле периода BRG. При этом устанавливается флаг запроса прерывания TXIF индицирующий, что TXREG пуст. Это прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита TXIE. Флаг запроса прерывания TXIF устанавливается независимо от значения TXIE. Он не может быть сброшен программно. Флаг сбрасывается аппаратно при загрузке новых данных в TXREG. Бит TRMT (TXSTA<1>) индицирует состояние сдвигового регистра TSR. Бит устанавливается когда TSR пуст. TRMT доступен только для чтения и не может вызывать прерывания. Регистр TSR не отображается в памяти данных, т.е. не доступен для чтения/записи.

Передача разрешается, при установке бита TXEN (TXSTA<5>). Фактически передача не начнется пока данные не будут загружены в TXREG. Первый бит данных появится на первом переднем фронте тактовых импульсов с вывода PA3/TX/CK. Данные стабилизируются к заднему фронту тактовых импульсов (см

Рис. 38). Передачу также можно начать сначала загрузив TXREG, а потом установив бит TXEN. Это удобно, когда выбраны низкие скорости передачи. Генератор BRG остановлен, когда биты TXEN, CREN, SREN сброшены. Установка бита TXEN запустит генератор BRG, который сразу же выдаст сдвиговые тактовые импульсы. Обычно, при первом разрешении передачи регистр TSR пуст, поэтому записанные в TXREG данные будут сразу переданы в TSR, опустошая TXREG. Поэтому возможна неразрывная последовательная передача данных. Сброс TXEN во время передачи вызовет отмену передачи, сброс передатчика. Если во время передачи будут установлены биты CREN и SREN, передача будет отменена, и вывод PA2/RX/DT вернется в третье состояние (для приема). Вывод PA3/TX/CK останется выходом, если установлен бит CSRC (внутренний источник тактовых импульсов от BRG). Логика передатчика не сбрасывается. Чтобы сбросить передатчик - необходимо сбросить бит TXEN. Если бит SREN установлен для прерывания осуществляемой передачи и получения одного байта, то после получения одного байта SREN сбросится и последовательный порт снова вернется к передаче, так как бит TXEN по прежнему установлен. Линия DT сразу же будет переключена из третьего состояния в режим выхода. Для избежание этого, TXEN должен быть сброшен.

Чтобы выбрать 9-ти битную передачу, необходимо установить бит TX9 (TXSTA<6>). Девятый бит должен записываться (в TX9D (TXSTA<0>)) до записи 8-ми битных данных в TXREG, так как запись данных в TXREG может вызвать немедленную передачу данных в TSR (если TSR пуст).

Шаги для настройки передачи в синхронном ведущем режиме:

1. Записать значение в регистр SPBRG для задания скорости передачи.
2. Включить синхронный последовательный порт в ведущем режиме (биты SYNC=1, SPEN=1 и CSRC=1).
3. Проверить что биты CREN и SREN сброшены, если эти биты установлены, то они отменяют передачу.
4. Если требуются прерывания, тогда установите бит TXIE.
5. Если требуется 9-ти битная передача, тогда установите бит TX9.
6. Если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9D.
7. Загрузите данные в регистр TXREG.
8. Запустите передачу установкой бита TXEN.

Для отмены передачи необходимо сбросить бит SPEN или бит TXEN. Это сбросит логику передачи, но сохранит настройки, когда передача возобновится.

Прием данных в синхронном ведущем режиме

Если выбран синхронный режим, прием разрешается установкой бита SREN (RCSTA<5>) или бита CREN (RCSTA<4>). Данные с вывода PA2/RX/DT опрашиваются на заднем фронте (спаде) тактовых импульсов. Если установлен SREN, то принимается только один байт. Если установлен CREN, то прием продолжается пока CREN не сбросится. Если установлены оба бита, то CREN имеет приоритет. После приема последнего бита полученные данные из сдвигового регистра приема (RSR) передаются в RCREG (если он пуст), и устанавливается флаг запроса прерывания RCIF. Прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита RCIE. RCIF доступен только для чтения, и сбрасывается аппаратно, когда RCREG считан и пуст.

Регистр RCREG имеет двойную буферизацию, т.е. можно принять два байта данных в RCREG FIFO и третий байт начать принимать в RSR. При приеме последнего бита третьего байта, если RCREG по-прежнему не считан, устанавливается бит ошибки переполнения приемника OERR (RCSTA<1>). Данные в RSR будут потеряны. Для извлечения двух байт RCREG должен считываться

дважды. Бит OERR должен быть очищен программно сбросом приема (сбросом бита CREN). Пока бит OERR установлен, приемник не работает. Девятый бит данных буферизуется также как и принятые данные. Поэтому необходимо считать регистр RCSTA перед считыванием RCREG, чтобы не потерять прежнее значение RX9D.

Шаги для настройки приема в синхронном ведущем режиме:

1. Записать значение в регистр SPBRG для задания скорости приема.
2. Включить синхронный последовательный порт в ведущем режиме (биты SYNC=1, SPEN=1 и CSRC=1).
3. Если требуются прерывания, тогда установите бит RCIE.
4. Если требуется 9-ти битный прием, тогда установите бит RX9.
5. Если требуется прием единичного байта установите бит SREN, для продолжительного приема установите бит CREN.
6. При завершении приема установится бит RCIF, и произойдет прерывание (если установлен бит RCIE).
7. Считайте RCSTA для получения значения девятого бита (для 9-ти битного приема) и бит определения ошибки.
8. Считайте 8-ми битные принятые данные из регистра RCREG.
9. Если произошла ошибка переполнения, сбросьте бит OERR.

Для отмены приема, сбросьте либо биты SREN и CREN, либо бит SPEN. Это сбросит логику приема, но не изменит настройки.

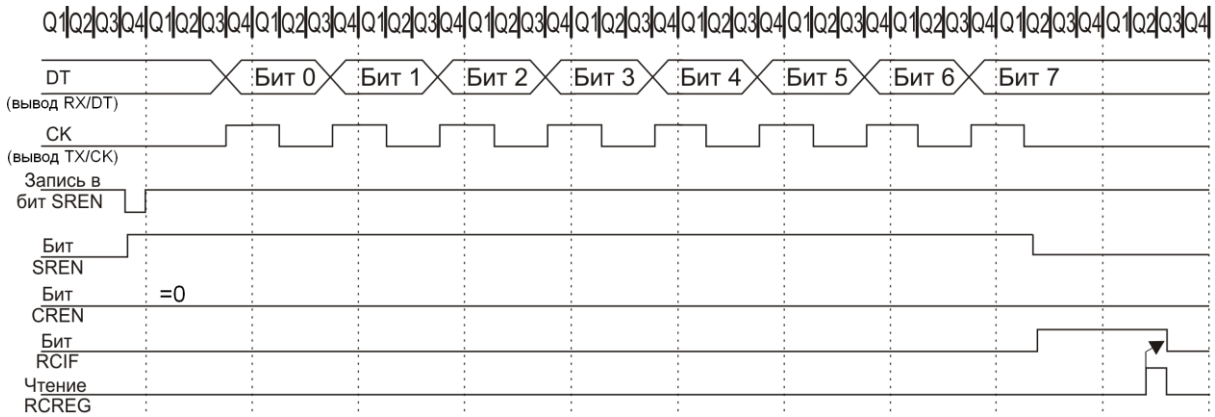


Рис. 39 Синхронный прием

Синхронный ведомый режим

Синхронный ведомый режим отличается от ведущего тем, что тактовые импульсы подаются от внешнего источника. Это позволяет устройству передавать или получать данные в режиме SLEEP. Ведомый режим включается сбросом бита CSRC (TXSTA<7>).

Передача данных в синхронном ведомом режиме

Работа ведущего и ведомого режимов идентична, за исключением работы в режиме SLEEP. Если 2 байта записываются в TXREG и затем исполняется команда SLEEP, то произойдет следующее. Первый байт сразу переносится в TSR и будет передаваться по тактовым импульсам. Второй байт останется в TXREG. Флаг TXIF не будет установлен. Когда закончится передача первого байта, второй байт будет перенесен из TXREG в TSR и установится флаг TXIF. Если TXIE=1, прерывание

выведет чип из режима SLEEP, и если разрешено периферийное прерывание, тогда программа переходит к вектору прерывания (0020h).

Шаги для настройки передачи в синхронном ведомом режиме:

1. Записать значение в регистр SPBRG для задания скорости передачи.
2. Включить синхронный последовательный порт в ведомом режиме (биты SYNC=1, SPEN=1 и CSRC=0).
3. Сбросить бит CREN.
4. Если требуются прерывания, тогда установите бит TXIE.
5. Если требуется 9-ти битная передача, тогда установите бит TX9.
6. Если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9D.
7. Загрузите данные в регистр TXREG.
8. Запустите передачу установкой бита TXEN.

Для отмены передачи необходимо сбросить бит SPEN или бит TXEN. Это сбросит логику передачи, но сохранит настройки, когда передача возобновится.

Прием данных в синхронном ведомом режиме

Работа ведущего и ведомого режимов идентична, за исключением работы в режиме SLEEP. Также безразлично значение бита SREN.

Если прием разрешен (CREN=1) до команды SLEEP, то данные могут быть получены в режиме SLEEP. При завершении приема, данные из RSR передаются в RCREG и устанавливается флаг запроса прерывания RCIF, а если бит RCIE=1 прерывание выведет чип из режима SLEEP. Если разрешено периферийное прерывание, программа перейдет к вектору прерывания (0020h).

Шаги для настройки приема в синхронном ведомом режиме:

1. Записать значение в регистр SPBRG для задания скорости приема.
2. Включить синхронный последовательный порт в ведомом режиме (биты SYNC=1, SPEN=1 и CSRC=0).
3. Если требуются прерывания, тогда установите бит RCIE.
4. Если требуется 9-ти битный прием, тогда установите бит RX9.
5. Для разрешения приема установите бит CREN.
6. При завершении приема установится бит RCIF, и произойдет прерывание (если установлен бит RCIE).
7. Считайте RCSTA для получения значения девятого бита (для 9-ти битного приема) и бит определения ошибки.
8. Считайте 8-ми битные принятые данные из регистра RCREG.
9. Если произошла ошибка переполнения, сбросьте бит OERR.

Для отмены приема, сбросьте либо бит CREN, либо бит SPEN. Это сбросит логику приема, но не изменит настройки.

Блок внутренней EEPROM памяти данных

Микроконтроллер содержит блок внутренней EEPROM памяти данных объемом 256 8-ми разрядных слов (байт). EEPROM память имеет организацию 16 строк по 16 байт. Минимально доступным для чтения и записи является один байт. Минимально доступной для стирания является одна строка (16 байт). Перед записью необходимо провести стирание строк, в которых будет расположена записываемая информация. Запись возможна только «1» поверх «0». В стертом состоянии память содержит все «0».

Структурная схема организации EEPROM памяти представлена на Рис. 40.

При переходе в SLEEP режим блок контроллера должен быть программным образом выключен для обеспечения энергосберегающего режима. После выхода из SLEEP режима блок должен включаться заново.
После выполнения каждой операции, контроллер EEPROM должен быть переведен в режим работы "нет" - EEPROM_MODE<2..0>=000.

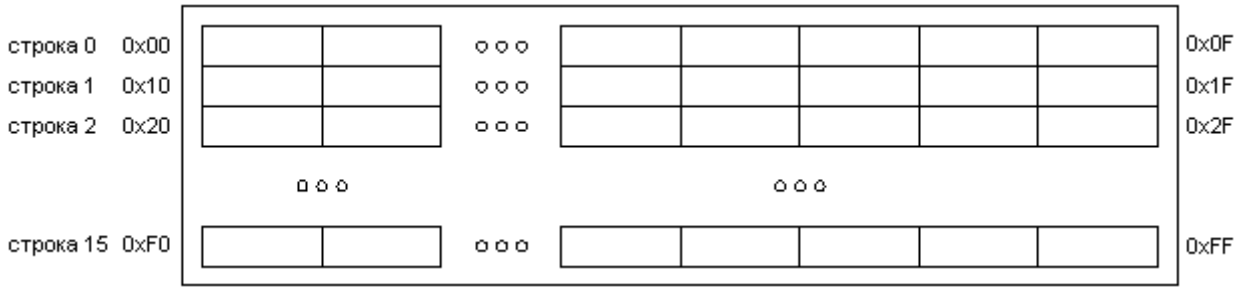


Рис. 40 Структурная схема организации EEPROM памяти

Операции, выполняемые контроллером внутренней EEPROM памяти:

- **Запись 8-ми разрядного слова (байта) в EEPROM** - позволяет записать заданный байт. Запись битов памяти производится только из состояния логического нуля в логическую единицу. Для перевода ячеек из состояния единицы в ноль необходимо использовать операцию стирания.
- **Чтение 8-ми разрядного слова (байта) из EEPROM** - позволяет считать заданный байт из EEPROM памяти.
- **Стирание всей EEPROM памяти** - позволяет стереть весь объем EEPROM памяти.
- **Стирание строки EEPROM памяти** - позволяет стереть заданную строку EEPROM памяти.
- **Запись всей EEPROM памяти одним значением** - позволяет записать весь объем EEPROM памяти заданным одинаковым значением.

Регистры управления блоком внутренней EEPROM памяти данных

Работа с EEPROM памятью осуществляется с помощью чтения или записи регистров контроллера. Описание регистров приведено ниже.

Таблица 29

Регистр контроля и тестирования EEPROM_CONT (адрес: 14h, банк 6)

R-1	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**
-	**EETEST**	**CPTEST**	**VEE2**	**VEE1**	**BRG2**	**BRG1**	**BRG0**
бит 7	6	5	4	3	2	1	бит 0
бит 7		зарезервировано, всегда считывается «1».					
бит 6		**EETEST:** сигнал для отбраковочных испытаний микросхемы.					
бит 5		**CPTEST:** сигнал для отбраковочных испытаний микросхемы.					
бит 4		**VEE2:** сигнал для отбраковочных испытаний микросхемы.					
бит 3		**VEE1:** сигнал для отбраковочных испытаний микросхемы.					
бит 2, 1, 0		**BRG2, BRG1, BRG0:** задание временных характеристик. В зависимости от частоты работы ядра микроконтроллера, необходимо указать контроллеру EEPROM памяти эту частоту, исходя из которой, он будет формировать необходимые					

	длительности сигналов «программирования» и «стирания» заданных длительностей: 000 = частота Fc = 30 – 33 МГц, 001 = частота Fc = 20 – 30 МГц, 010 = частота Fc = 10 – 20 МГц, 011 = частота Fc = 1 – 10 МГц, 100 = частота Fc = 500 кГц – 1 МГц, 101 = частота Fc = 250 кГц – 500 кГц, 110 = частота Fc = 0 – 250 кГц, 111 = зарезервировано

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 30

Регистр режима работы EEPROM_MODE (адрес: 15h, банк 6)

R/W-0	R-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
EN_EE	**-**	**IEBUSY_EE**	**BUSY_EE**	**-**	**MODE2_EE**	**MODE1_EE**	**MODE0_EE**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**EN_EE:** разрешение работы контроллера EEPROM памяти: 0 = контроллер выключен, 1 = контроллер включен
бит 6	зарезервировано
бит 5	**IEBUSY_EE:** флаг разрешения прерывания по окончанию занятости контроллера EEPROM памяти: 0 = прерывание запрещено, 1 = прерывание разрешено.
бит 4	**BUSY_EE:** флаг занятости контроллера EEPROM памяти: 0 = контроллер свободен, 1 = контроллер занят
бит 3	зарезервировано
бит 2, 1, 0	**MODE2_EE, MODE1_EE, MODE0_EE:** режим работы контроллера EEPROM памяти: 000 = нет, 001 = чтение байта из EEPROM, 010 = запись слова в EEPROM, 011 = стирание строки EEPROM, 100 = стирание всей EEPROM, 101 = запись всей EEPROM одним значением, 11x = запрещенная комбинация

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;

0 = сброшен;
x = значение не известно.

Таблица 31

Регистр данных EEPROM_DATA (адрес: 16h, банк 6).

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** записываемые или считываемые данные					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 32

Регистр адреса обращения EEPROM_ADDR (адрес: 17h, банк 6)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADDR7	**ADDR6**	**ADDR5**	**ADDR4**	**ADDR3**	**ADDR2**	**ADDR1**	**ADDR0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**ADDR7…ADDR0:** Регистр адреса данных: При чтении и записи одного байта используются все разряды адреса, при стирании строки имеют значения только ADDR[7:4], при блочной очистке и записи содержимое регистра значения не имеет					

Работа блока по стиранию, записи и чтению данных

Включение EEPROM памяти

После выставления бита EN_EE в регистре EEPROM_MODE производится включение EEPROM памяти. Процесс включения при тактовой частоте микроконтроллера 33 МГц занимает порядка 45 мкс (Tsuy). Последовательность действий при включении EEPROM памяти представлена на Рис. 41.

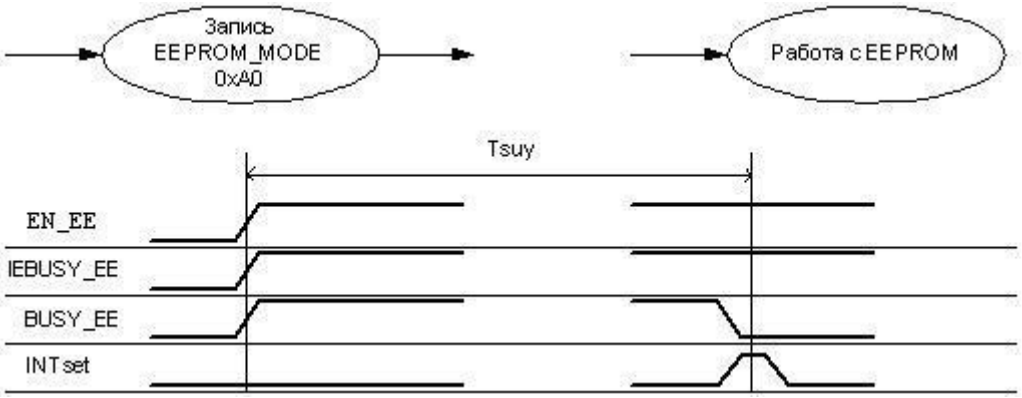


Рис. 41 Включение EEPROM памяти

Стирание всей EEPROM памяти

Для стирания всего содержимого EEPROM памяти используется команда «стирание всей EEPROM памяти». Для выполнения операции необходимо чтобы был включен контроллер EEPROM памяти. Процесс стирания всей EEPROM памяти при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp). Последовательность действий при стирании всей EEPROM памяти представлена на Рис. 42.

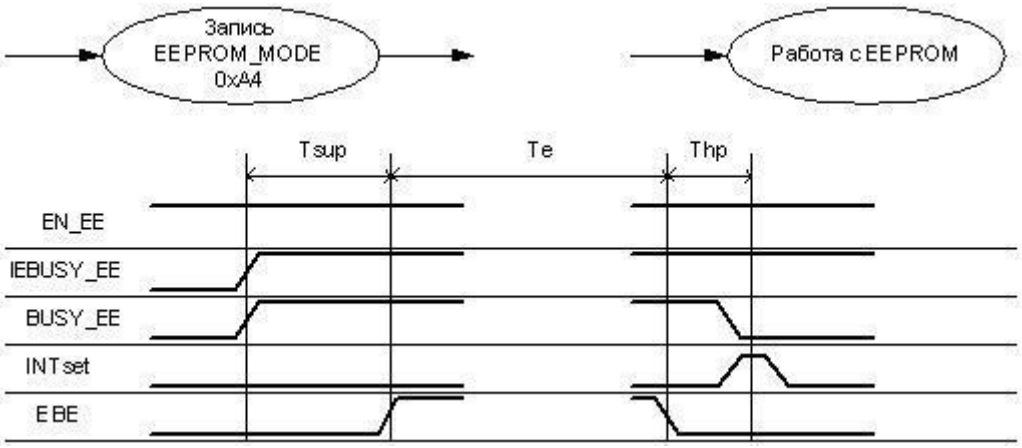


Рис. 42 Стирание всей EEPROM памяти в TEST режиме

Запись всей EEPROM памяти одним значением

Для заполнения всего объема EEPROM памяти используется команда «запись всей EEPROM памяти». Перед подачей этой команды в регистр EEPROM_DATA необходимо записать значение, которым будет заполнена вся память. Для заполнения всей памяти необходимо, чтобы был включен контроллер EEPROM памяти. Процесс заполнения всей EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp). Последовательность действий при заполнении всей EEPROM памяти одним значением представлена на Рис. 43.

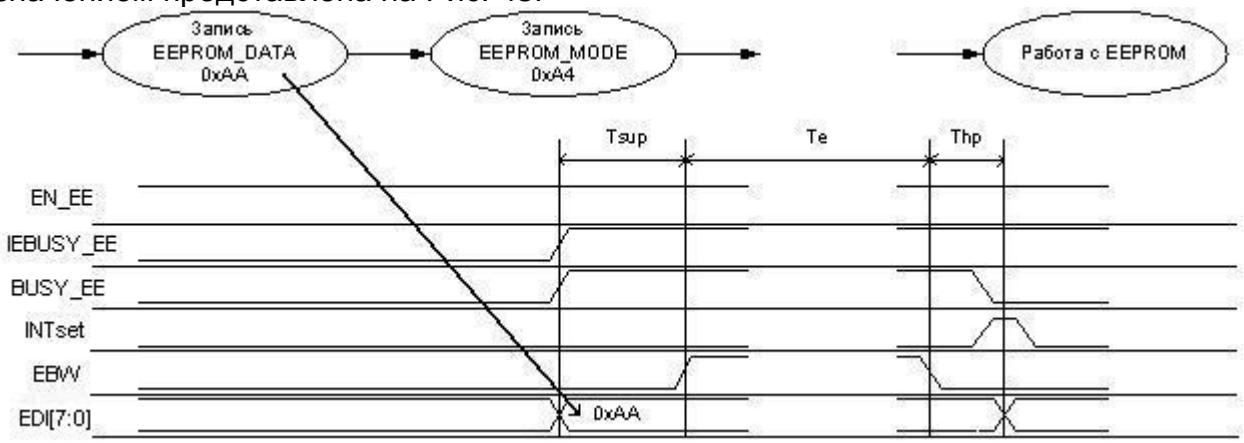


Рис. 43 Запись всей EEPROM памяти одним значением

Стирание строки EEPROM памяти

Для стирания содержимого одной строки EEPROM памяти используется команда «стирание строки EEPROM памяти». Перед подачей этой команды в регистр EEPROM_ADDR[7:4] необходимо записать адрес строки. Младшие разряды регистра EEPROM_ADDR значения не имеют. Для стирания заданной строки необходимо, чтобы был включен контроллер EEPROM памяти. Процесс стирания одной строки EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp). Последовательность действий при стирании одной строки EEPROM представлена на Рис. 44.

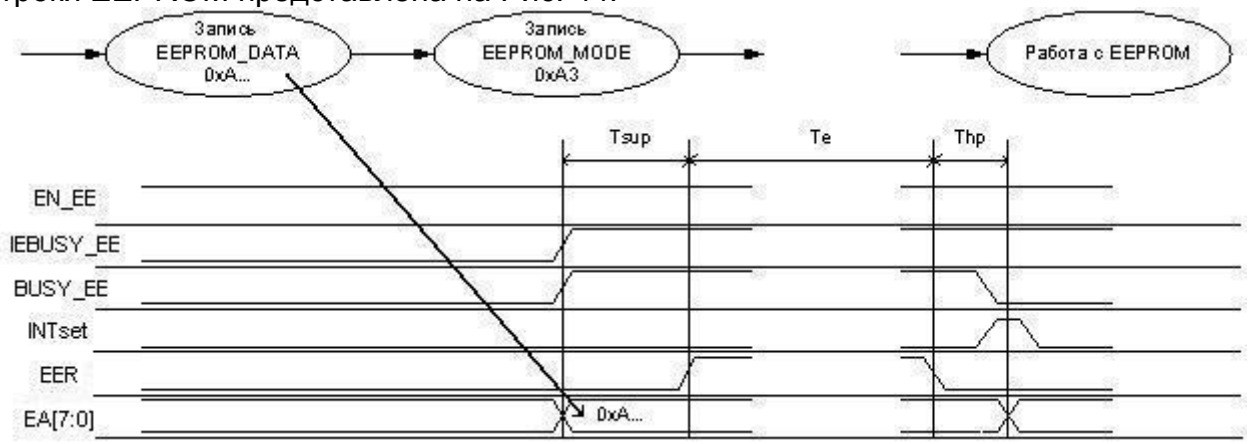


Рис. 44 Стирание строки EEPROM памяти

Запись байта в EEPROM память

Для записи одного 8-ми разрядного слова (байта) в EEPROM память используется команда «запись байта в EEPROM память». Перед подачей этой команды в регистр EEPROM_ADDR[7:0] необходимо записать адрес для записи, а в регистр EEPROM_DATA записать значение для сохранения в EEPROM память. Для записи байта в память необходимо, чтобы был включен контроллер EEPROM памяти. Процесс записи байта в EEPROM память при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp). Последовательность действий при записи одного байта в EEPROM память представлена на Рис. 45.

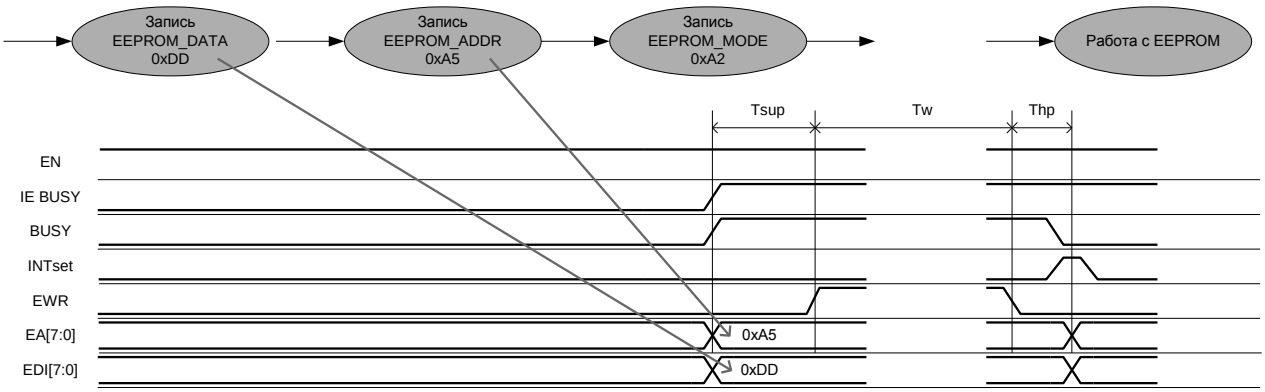


Рис. 45 Запись байта в EEPROM память

Чтение байта из EEPROM памяти

Для чтения одного 8-ми разрядного слова (байта) из EEPROM памяти используется команда «чтение байта из EEPROM памяти». Перед подачей этой команды в регистр EEPROM_ADDR[7:0] необходимо записать адрес для чтения. Для чтения слова из памяти необходимо, что бы контроллер EEPROM памяти был включен. Процесс чтения байта из EEPROM памяти при тактовой частоте микроконтроллера 33 МГц занимает порядка 1 мкс (Tsux, Tacc и Thx). Последовательность действий при чтении одного байта из EEPROM памяти представлена на Рис. 46.

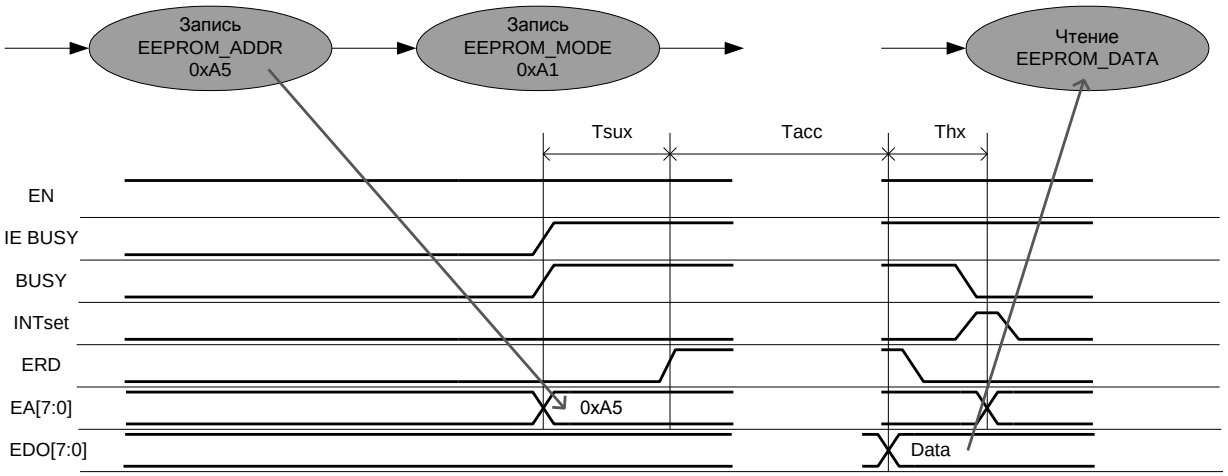


Рис. 46 Чтение байта из EEPROM памяти

Описание блока контроллера внешней NAND Flash памяти

Блок контроллера внешней NAND Flash памяти предназначен для обеспечения работы ядра микроконтроллера с периферийной памятью типа NAND Flash фирмы Samsung и других. Микросхемы памяти подсоединяются к микроконтроллеру через порты «PORTC» и «PORTD» (см. Рис. 47). При этом если микросхема памяти работает от напряжения питания в 3.3В, данные порты так же должны быть запитаны этим напряжением. Для этого можно использовать встроенный регулятор напряжения и с вывода U33out завести питание на выводы $U_{cc1port}$ и $U_{cc2port}$. Нагрузочная способность встроенного регулятора обеспечивает ток до 40 мА. Таким образом, от этого источника может быть запитана и сама микросхема NAND Flash памяти.

Контроллер внешней NAND Flash памяти аппаратно формирует сигналы управления RBn, REn, CLE, ALE, WEn и управляет шиной данных IO. Сигналы управления CE и WP для микросхемы памяти не формируются. Разработчик сам обязан выбрать какие либо свободные линии ввода-вывода микроконтроллера для формирования этих сигналов. Например, можно использовать линию PD2 (AD10) для формирования сигнала CE для первой микросхемы памяти, а линию PD3 (AD11) для формирования сигнала CE для второй микросхемы памяти. Линия PD7 (AD15) может использоваться для формирования или контроля сигнала WP (Write Protect). Кроме того, линия RBn должна быть подтянута к уровню логической единицы. При включении блока контроллера внешней NAND Flash памяти, порт «PORTC» полностью переходит под управление этого блока. Порт «PORTD» частично. Линии PD0, PD1, PD4, PD5 и PD6 управляются блоком, а линии PD2, PD3 и PD7 остаются под управлением регистров порта «PORTD» и разработчик может их использовать по своему усмотрению. При этом питание, подаваемое на выводы $U_{cc1port}$ и $U_{cc2port}$, действует для всех линий портов «PORTC» и «PORTD», независимо

от выбранного режима работы. При переходе в SLEEP режим разработчик должен завершить все операции работы с NAND Flash памятью, и после этого программно выключить внутренний регулятор напряжения для обеспечения энергосберегающего режима. После выхода из SLEEP, аналогичным образом необходимо включить источник и проинициализировать микросхему памяти.

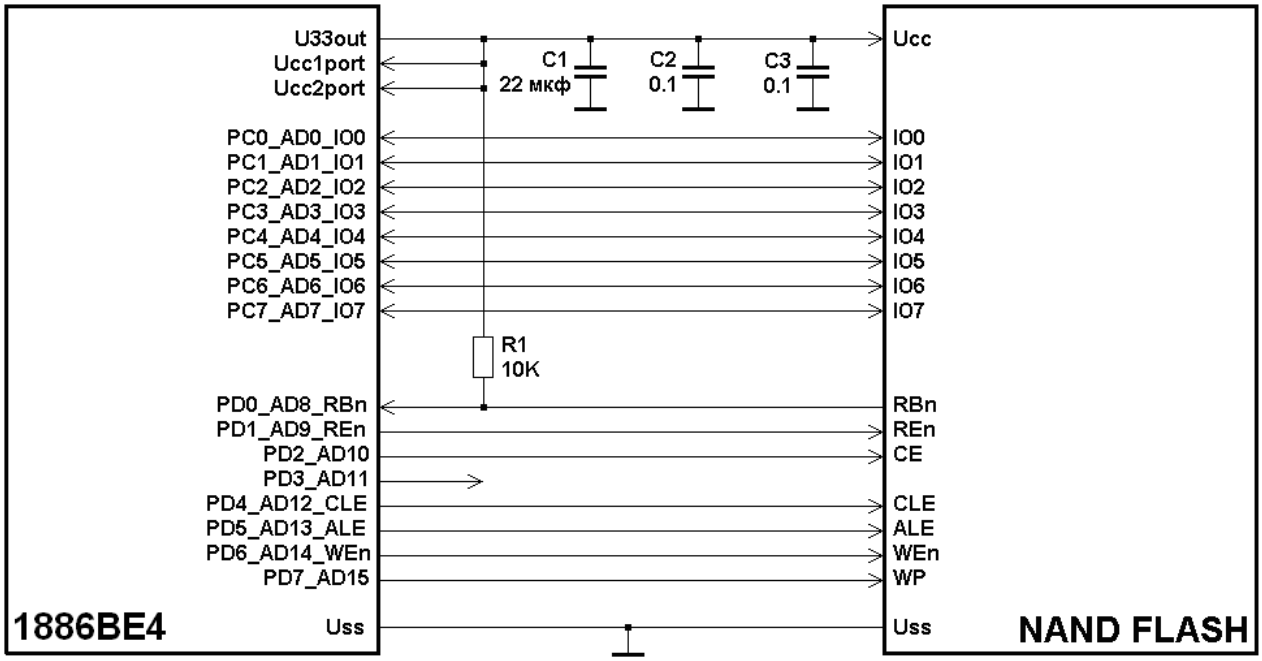


Рис. 47 Подключение к микроконтроллеру внешней NAND Flash памяти

Описание программного доступа к NAND Flash памяти

Семейство микросхем NAND Flash памяти имеет достаточно большую номенклатуру. Микросхемы отличаются друг от друга временными характеристиками, размерами памяти и организацией. Но принципы работы с ними у всех одинаковые. Для работы с конкретной микросхемой предварительно изучите ее описание. В данном описании приводятся общие принципы работы.

Запись команды в NAND Flash память

Для задания команды NAND Flash памяти необходимо в регистре NAND_MODE установить режим «выдача командной информации». Затем в регистр NAND_DATA записать значение команды. Различные типы микросхем NAND Flash памяти могут иметь различные наборы команд. Данная документация будет построена на базе рассмотрения микросхемы K9K1G08U0M фирмы Samsung. Эта микросхема имеет следующий набор команд (Таблица 33). Диаграмма сигналов представлена на
Рис. **48**.

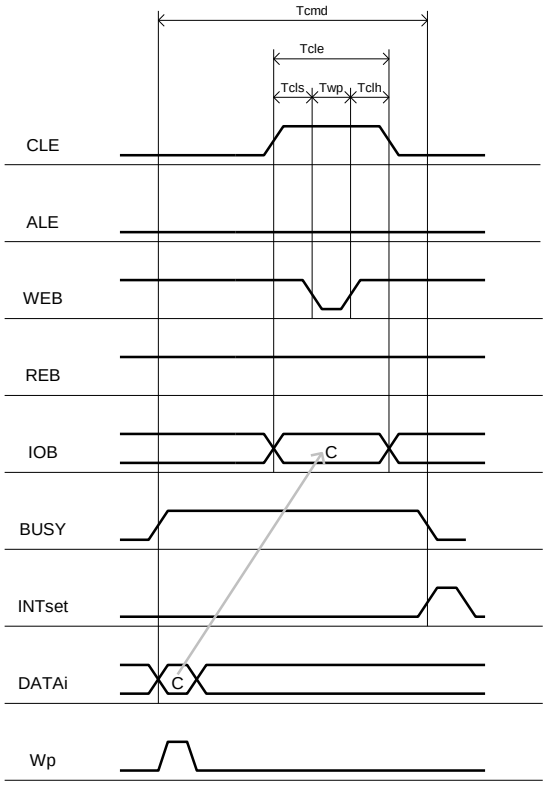


Рис. 48 Диаграмма сигналов записи команды в NAND Flash память

Таблица 33

Набор команд микросхемы NAND Flash памяти K9K1G08U0M

Функция	Первый Цикл	Второй Цикл	Третий Цикл	Примечания
Read 1	0x00/0x01(1)	-	-	
Read 2	0x50	-	-	
Read ID	0x90	-	-	
Reset	0xFF	-	-	
Page Program (True)	0x80	0x10	-	
Page Program (Dummy)	0x80	0x11	-	
Copy-Back Program (True)	0x00	0x8A	0x10	
Copy-Back Program (Dummy)	0x03	0x8A	0x11	
Block Erase	0x60	0xD0	-	
Multi-Plane Block Erase	0x60—0x60	0xD0	-	
Read Status	0x70	-	-	
Read Multi-Plane Status	0x71	-	-	

Примечание:

(1) Команда 0x00 определяет, что обращение идет начиная с первой половины регистровой памяти, команда 0x01 определяет что обращение идет со второй половины регистровой памяти.

Запись адреса в NAND Flash память

Для задания адреса обращения в NAND Flash необходимо в регистре NAND_MODE установить режим «выдача адресной информации». Затем в регистр NAND_DATA последовательно записать адресную информацию. Для окончания выдачи адресной информации необходимо в регистре NAND_MODE установить режим «нет работы» либо любой другой режим отличный от «выдача адресной информации». Диаграмма сигналов представлена на Рис. 49.
Для микросхемы K9K1G08U0M фирмы Samsung адрес выдается в четыре цикла:

Таблица 34

	IO0	IO1	IO2	IO3	IO4	IO5	IO6	IO7
1 Цикл	A0	A1	A2	A3	A4	A5	A6	A7
2 Цикл	A9	A10	A11	A12	A13	A14	A15	A16
3 Цикл	A17	A18	A19	A20	A21	A22	A23	A24
4 Цикл	A25	A26	Low	Low	Low	Low	Low	Low

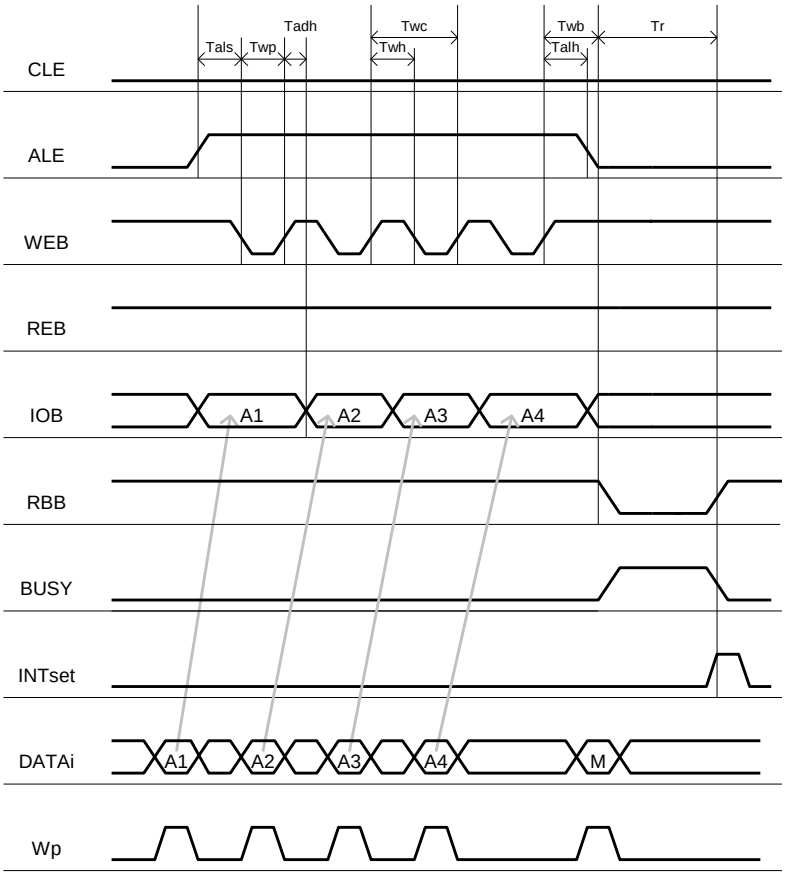


Рис. 49 Диаграмма сигналов записи адреса в NAND Flash память

Стирание страницы NAND Flash памяти

Перед записью Flash памяти необходимо предварительно ее стереть. NAND Flash позволяет стирать минимум одну страницу. Диаграмма работы при стирании представлена на Рис. 50.

Запись данных в NAND Flash память

Для записи данных в NAND Flash память необходимо подготовить микросхему к приему данных: очистить область для записи данных, установить адрес и функцию записи. После этого установите в регистре NAND_MODE режим «программная запись данных» и последовательно в регистр NAND_DATA записывайте данные, предназначенные для сохранения в NAND Flash памяти. Диаграмма сигналов представлена на Рис. 50.

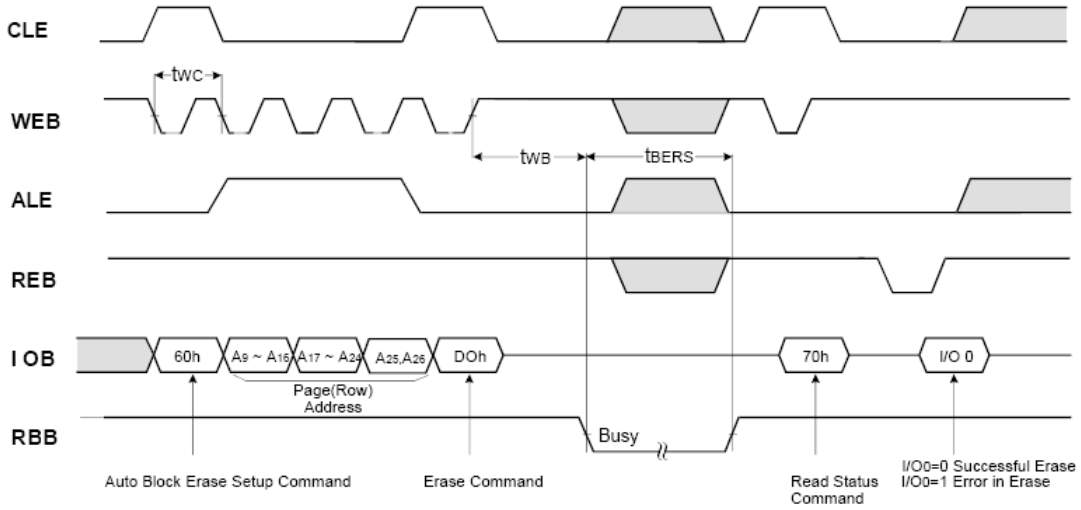


Рис. 50 Диаграмма сигналов при стирании страницы NAND Flash памяти

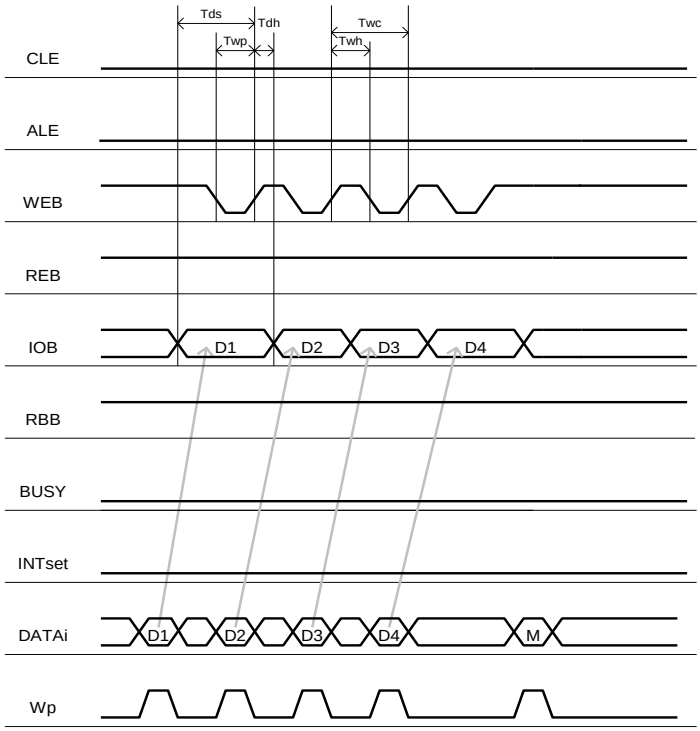


Рис. 51 Диаграмма сигналов записи данных в NAND Flash память

Чтение данных из NAND Flash памяти

Для чтения данных из NAND Flash памяти необходимо подготовить микросхему к выдаче данных: установить адрес чтения и функцию чтения, дождаться готовности, и далее установить в регистре NAND_MODE режим

«программное чтение данных» и последовательно из регистра NAND_DATA считывать данные. Диаграмма сигналов представлена на Рис. 52.

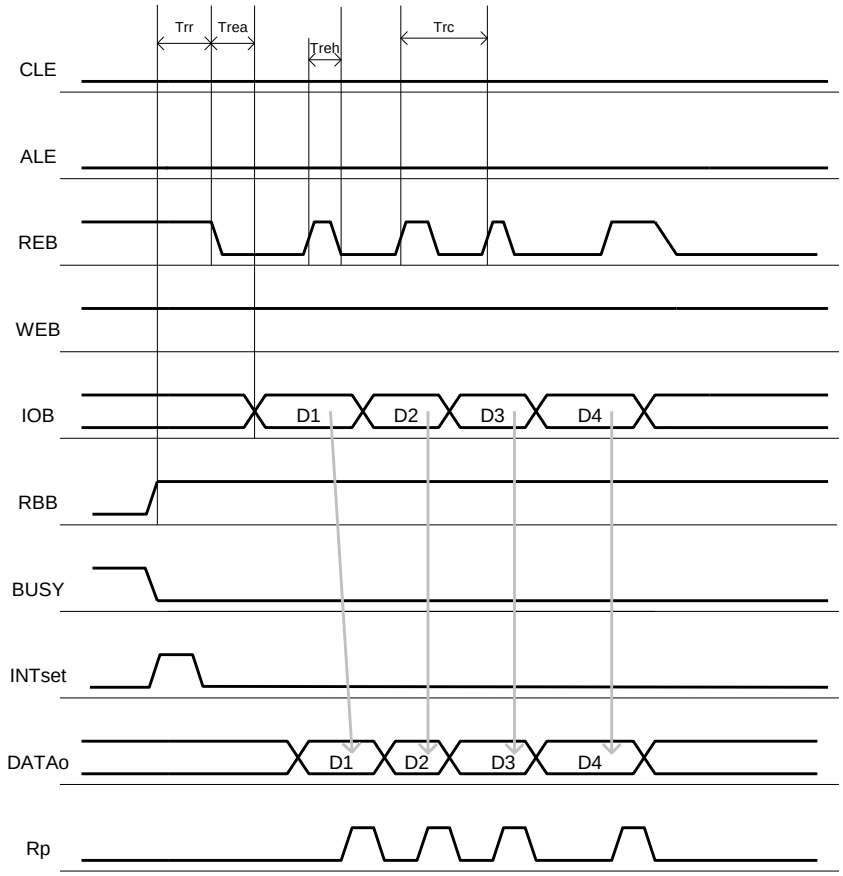


Рис. 52 Диаграмма сигналов чтения данных из NAND Flash памяти

Регистры управления контроллером NAND Flash памяти

Управление контроллером внешней NAND Flash памяти осуществляется с помощью двух регистров: регистра режима работы контроллера (см. Таблица 35) и регистра данных (см. Таблица 36).

Таблица 35

Регистр режима работы контроллера NAND Flash памяти NAND_MODE (адрес: 10h, банк 6)

R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
EN_NAND	**LOW_SPEED_NAND**	**IE_BUSY_NAND**	**BUSY_NAND**	**MODE3_NAND**	**MODE2_NAND**	**MODE1_NAND**	**MODE0_NAND**
бит 7	6	5	4	3	2	1	бит 0
бит 7		**EN_NAND:** разрешение работы контроллера NAND Flash памяти и конфигурирование портов: 0 = контроллер выключен, 1 = контроллер включен					
бит 6		**LOW_SPEED_NAND:** скорость работы контроллера: 0 = нормальная скорость, 1 = медленная скорость. Примечание: этот бит применяется для обеспечения доступа к					

	медленным микросхемам NAND Flash памяти. В данном режиме все времена увеличиваются вдвое, при этом дополнительно устанавливается флаг BUSY_NAND
бит 5	**IE_BUSY_NAND:** флаг разрешения прерывания по окончанию занятости контроллера NAND Flash памяти: 0 = прерывание запрещено, 1 = прерывание разрешено
бит 4	**BUSY_NAND:** флаг занятости контроллера NAND Flash памяти: 0 = контроллер свободен, 1 = контроллер занят
бит 3, 2, 1, 0	**MODE3_NAND, MODE2_NAND, MODE1_NAND, MODE0_NAND:** режим работы контроллера NAND Flash памяти: 0000 – нет работы, 0001 – выдача адресной информации, 0010 – выдача командной информации, 0011 – чтение данных из NAND Flash память, 0100 – запись данных в NAND Flash память, 1011 – зарезервировано, 1100 – зарезервировано

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 36

Регистр данных NAND Flash памяти NAND_DATA (адрес: 11h, банк 6)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** записываемые и считываемые данные, передаваемые команды и адреса					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Контроллер USB интерфейса

Цифровой интерфейсный блок USB предназначен для реализации контроллера интерфейса USB версии 1.1 со скоростью передачи до 12 Мбит/с. Блок обеспечивает работу на низкой 1.5 Мбит/с (Low Speed) и полной скорости 12 Мбит/с

(Full Speed). Блок поддерживает до 5 оконечных точек (Endpoint): одна оконечная точка управления работой блока контроллера USB и 4 пользовательских оконечных точки. На кристалле микроконтроллера реализован аналоговый приемопередатчик PHY, включающий линейные драйверы линий D+ и D-, обеспечивающий физический стык с шиной USB. Схемы подключения микроконтроллера к USB шине представлены на Рис. 53 и
Рис. **54**. Так же контроллер позволяет работать с внешним приемопередатчиком, например микросхемой PDIUSBP11A.

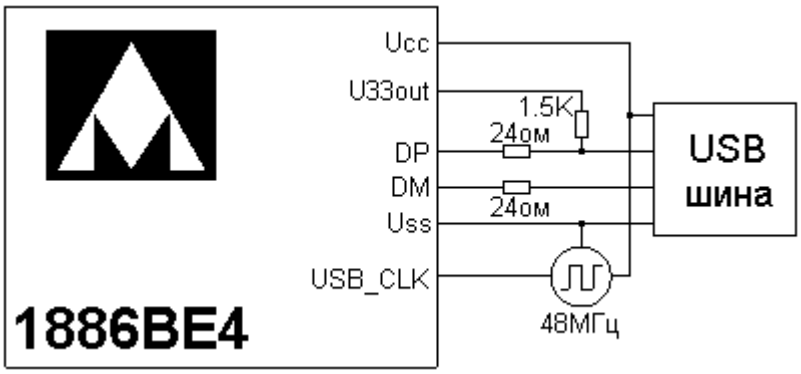


Рис. 53 Схема подключения микроконтроллера к USB шине в режиме Full Speed

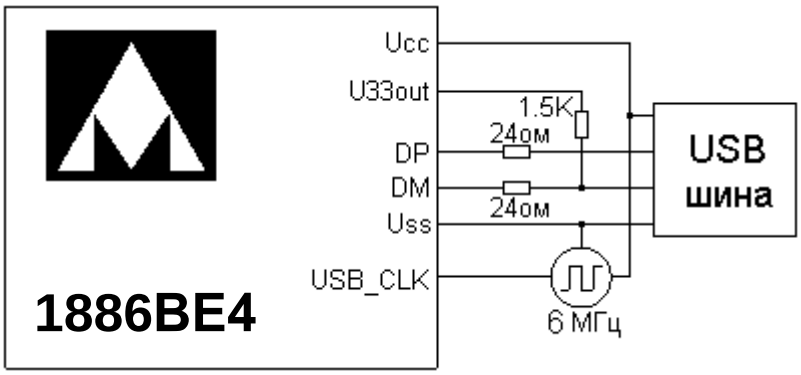


Рис. 54 Схема подключения микроконтроллера к USB шине в режиме Low Speed

Описание принципов работы и функционирования контроллера USB интерфейса

Блок контроллера USB предназначен для приема и передачи данных по USB интерфейсу версии 1.1. между микроконтроллером (functional device) и хост-контроллером (USB host). Блок контроллера USB может функционировать в режимах Low Speed (1.5 Мбит/с) и Full Speed (12 Мбит/с). В составе блока реализовано 5 оконечных точек. Нулевая оконечная точка предназначена для инициализации контроллера USB после подсоединения его к хост-контроллеру. Четыре пользовательские оконечные точки могут быть настроены для обмена пользовательскими данными с хост-контроллером. Пользовательские оконечные точки могут быть сконфигурированы в режимы IN или OUT. В режиме IN оконечная точка обеспечивает передачу данных от микроконтроллера к хост-контроллеру. В режиме OUT оконечная точка обеспечивает прием данных от хост-контроллера к микроконтроллеру. В зависимости от выбранного режима, блок контроллера USB обеспечивает следующие типы обмена с хост-контроллером: Interrupt transfer, Bulk transfer и Isochronous transfer. При работе в режиме Low Speed обеспечивается тип

обмена Interrupt Transfer. При работе в режиме Full Speed обеспечивается тип обмена Interrupt Transfer, Bulk Transfer и Isochronous Transfer. Блок контроллера USB позволяет задать пользовательский описатель устройства (Descriptor).

Структурная схема блока USB приведена на **Рис. 55**, перечень основных выполняемых функций в Таблица 37.

Таблица 37

Основные выполняемые функции

Наименование	Краткое описание
Автоматическая инициализация USB интерфейса	При разрешении работы из программы, выполняемой на микроконтроллере, и подсоединении его к хост-контроллеру, будет произведена автоматическая инициализация блока контроллера USB. Дополнительное участие ядра микроконтроллера при инициализации блока не требуется. Программа должна обеспечить мониторинг сообщений и ошибок, которые могут возникнуть при инициализации
Пользовательский описатель устройства USB	При реализации USB интерфейса пользователь может задать произвольный описатель USB устройства, включая Vendor ID, Product ID, до 3 строковых описателей и прочее
Тип обмена: Interrupt Transfer	Тип обмена с подтверждением приема, обеспечивает передачу пакетов длиной до 8 байт в режиме Low Speed и длиной до 64 байт в режиме Full Speed. Пакет может передаваться не чаще одного раза во фрейм (1 миллисекунду)
Тип обмена: Bulk Trans	Тип обмена с подтверждением приема, обеспечивает передачу пакетов длиной до 64 байт только в режиме Full Speed. Пакеты могут передаваться многократно во время одного фрейма
Тип обмена: Isochronous Transfer	Тип обмена без подтверждения приема, обеспечивает передачу пакетов длиной до 64 байт только в режиме Full Speed. Пакеты могут передаваться многократно во время одного фрейма
Режим передачи: IN	При конфигурировании одной из оконечных точек в режим передачи IN, контроллер будет автоматически осуществлять передачу данных хост-контроллеру. Программа, выполняемая на микроконтроллере, должна обеспечить корректную запись в блок контроллера USB данных, предназначенных для передачи. Также программа должна обеспечить мониторинг сообщений и ошибок, которые могут возникнуть при передаче
Режим приема: OUT	При конфигурировании одной из оконечных точек в режим приема OUT, контроллер будет автоматически осуществлять прием данных от хост-контроллера. Программа, выполняемая на микроконтроллере, должна обеспечить корректное чтение принятых данных. Так же программа должна обеспечить мониторинг сообщений и ошибок, которые могут возникнуть при приеме

Основным блоком контроллера является Protocol Layer, обеспечивающий обработку принимаемых пакетов и формирование отсылаемых пакетов. Protocol Layer обеспечивает расчет и контроль четности, bit staffing, обработку запросов при конфигурировании, инициализацию интерфейса. Обработку принимаемых пакетов и

формирование передаваемых пакетов. При обработке конфигурационных запросов, получаемый пакет размещается в принимающей очереди нулевой оконечной точки EP0 OUT. Описание устройства записанное в блок Descriptor переписывается в передающую очередь EP0 IN, после чего блоком Protocol Layer формируется в пакеты и передается хост-контроллеру. Принимаемые и отправляемые пакеты передаются через UTMI интерфейс специальному аналоговому приемопередатчику. При работе по передаче пользовательских данных, принимаемые от хост-контроллера данные могут быть размешены в очереди одной из четырех пользовательских оконечных точек, отправляемые данные могут записываться в очередь другой оконечной точки, после чего из этих данных будут сформированы пакеты Protocol Layer и переданы хост-контроллеру. Работа с блоком контроллера USB интерфейса со стороны ядра микроконтроллера осуществляется через интерфейс Core Interface.

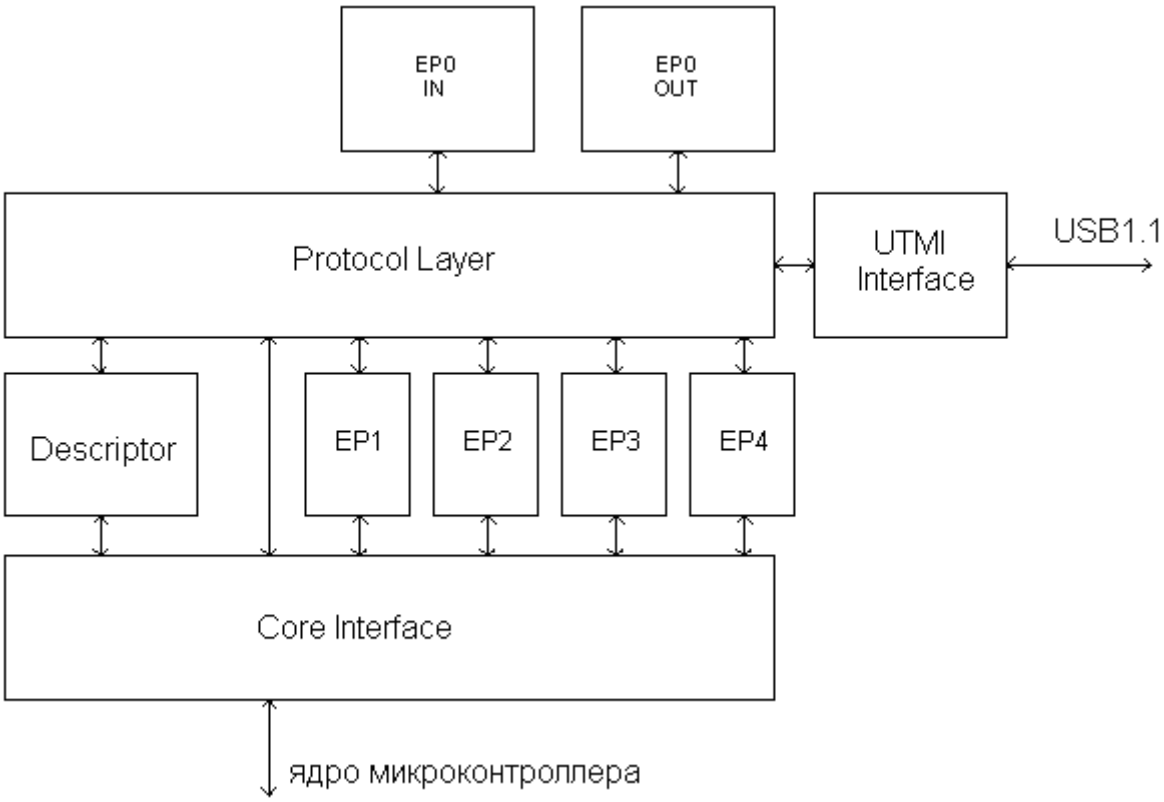


Рис. 55 Структурная схема блока USB

Работа с контроллером USB интерфейса

Работа микроконтроллера с блоком контроллера USB осуществляется с помощью записи и чтения регистров блока. Описание регистров приведено далее.

Регистр функционального адреса

В процессе инициализации хост-контроллером каждому USB устройству назначается уникальный функциональный адрес. Значение присвоенного USB устройству адреса отображается в данном регистре. После присвоения адреса так же взводится бит ADR_SET в регистре USB_STAT.

Таблица 38

Регистр функционального адреса USB_ADR (адрес: 10h, банк 4)

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
-	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
бит 7	6	5	4	3	2	1	бит 0
бит 7		зарезервировано					
бит 6…0		**ADR6…ADR0:** Функциональный адрес USB устройства. Назначается хост-контроллером в процессе инициализации устройства					

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Регистр статуса ошибок

В процессе работы блока контроллера USB, при передаче данных или управляющих пакетов, могут возникать ошибки. Ошибки могут возникать в результате недостаточного качества линии связи, ошибках в программном обеспечении драйвера хост-контроллера, ошибок в программе выполняемой на ядре микроконтроллера и т.д. Данные ошибки исправляются контроллером USB автоматически, но разработчик может при возникновении данных ошибок принять меры по улучшению надежности работы схемы.

Таблица 39

Регистр статуса ошибок USB_ERROR (адрес: 11h, банк 4)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUS_ ERR	DROP_ FRM	TO_ERR	SEQ_ER R	NSE_ER R	PID_ER R	CRC16_ ERR	CRC5_ ERR
бит 7	6	5	4	3	2	1	бит 0
бит 7		**BUS_ERR:** Bus Error. На линиях D+, D- обнаружена ошибка. Может быть вызвана неправильным подсоединением Full Speed устройства в Low Speed линии, или ошибка bit staffing, byte error и т.д.					
бит 6		**DROP_FRM:** Dropped Frame. Контроллер отказа в приеме пакета от хост-контроллера по причине отсутствия свободного месте в очереди оконечной точки. Хост контроллеру отправлен отказ в подтверждении приема					
бит 5		**TO_ERR:** Time Out Error. Хост контроллер не прислал подтверждения приема данных и они будут посланы контроллером снова, либо после пакета DATA OUT не получен пакет с данными и контроллер вернулся в режим ожидания					
бит 4		**SEQ_ERR:** Sequential Error. Получен пакет с тем же DATAx что и предыдущий пакет. Пакет проигнорирован, хост контроллеру выдано подтверждение приема					
бит 3		**NSE_ERR:** No Such Endpoint Error. Хост контроллер обратился к устройству с нашим функциональным адресом, но к несуществующей оконечной точке					

бит 2	**PID_ERR:** PID Error. Инверсия поля PID не соответствует прямой части
бит 1	**CRC16_ERR:** CRC16 Error. CRC16 полученных данных не соответствует принятому полю, данных отброшены, хост контроллеру отправлен отказ в подтверждении приема
бит 0	**CRC5_ERR:** CRC5 Error. CRC5 полученного пакета не соответствует рассчитанному, пакет проигнорирован

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистр статуса блока

В процессе работы блока контроллера USB с хост-контроллером, статус текущего состояния и событий, происходящих при работе, отображается в регистре USB_STAT.

Таблица 40

Регистр статуса блока USB_STAT (адрес: 12h, банк 4)

U-0	R-0	R-0	R/W-0	R-0	R-0	R-0	R-0
-	**ADR_SET**	**CONF_SET**	**USB_RST**	**EP2_SEL**	**EP1_SEL**	**EP0_SEL**	**USB_BUSY**
бит 7	6	5	4	3	2	1	бит 0

бит 7	зарезервировано.
бит 6	**ADR_SET:** Addressed. Контролер получил адрес от хост-контроллера
бит 5	**CONF_SET**: Configured. Контролер сконфигурирован хост-контроллером
бит 4	**USB_RST:** Reset on USB lines. Хост-контроллер выставил команду сброса на USB линию
бит 3, 2, 1	**EP2_SEL; EP1_SEL; EP0_SEL:** Endpoint Select. Текущая, выбранная хост-контроллером, оконечная точка для работы: 000 – Нулевая оконечная точка либо работа не ведется. 001 – Первая оконечная точка. 010 – Вторая оконечная точка. 011 – Третья оконечная точка. 100 – Четвертая оконечная точка. Остальные – недопустим
бит 0	**USB_BUSY:** USB BUSY. Хост-контроллер работает с контроллером USB

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистр управления блока

Для обеспечения возможности работы контроллера USB с хост-контроллером, необходимо задать режим работы контроллера и разрешить его работу. Кроме того, необходимо задать режимы работы для четырех оконечных точек (регистры EPx_CFGx). Изменение режимов работы после инициализации контроллера могут привести к сбоям в работе.

Таблица 41

Регистр управления блока USB_CTRL (адрес: 13h, банк 4)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	**-**	**-**	**-**	**USB_ TEST2**	**USB_ TEST1**	**FULL_ LOW**	**USB_EN**
бит 7	6	5	4	3	2	1	бит 0

бит 7, 6, 5, 4	зарезервировано.
бит 3	**USB_TEST2:** Тестирование цифровой части. При выставлении данного бита, на порт E выводятся цифровые сигналы USB. И делает возможным тестирование с внешним аналоговым драйвером шины или тестовым оборудованием. 0 = тестовый режим выключен, 1 = тестовый режим включен
бит 2	**USB_TEST1:** Тестирование буфера. Используется в процессе диагностических и отбраковочных тестов. 0 = нет тестирования, 1 = на TX_DP и TX_DN выдается меандр для оценки характеристик передатчика. USB должен быть разрешен и отсоединен от хост-контроллера
бит 1	**FULL_LOW:** Full Low Select. Определяет, в каком режиме будет функционировать контроллер USB: 0 = Low Speed (1.5 Мбит/с), 1 = Full Speed (12 Мбит/с)
бит 0	**USB_EN:** USB Enable. Разрешение работы контроллера USB

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Регистры разрешения прерываний от блока USB

При заполнении буферов, их опустошении и других событиях возможно формирование прерывания микроконтроллера. Для выбора событий, генерирующих прерывания, используется регистр разрешения прерываний.

Таблица 42

Регистр разрешения прерываний от USB блока USB_IE1 (адрес: 14h, банк 4)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EP4_ FULL_IE	**EP4_ EMPTY_ IE**	**EP3_ FULL_IE**	**EP3_ EMPTY_ IE**	**EP2_ FULL_IE**	**EP2_ EMPTY_ IE**	**EP1_ FULL_IE**	**EP1_ EMPTY_ IE**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**EP4_FULL_IE:** Endpoint 4 Fifo Full Interupt Enable. 1 = разрешено прерывание при заполнении ФИФО четвертой оконечной точки, 0 = прерывание запрещено
бит 6	**EP4_EMPTY_IE:** Endpoint 4 Fifo Empty Interupt Enable. 1 = разрешено прерывание при опустошении ФИФО четвертой оконечной точки, 0 = прерывание запрещено
бит 5	**EP3_FULL_IE:** Endpoint 3 Fifo Full Interupt Enable. 1 = разрешено прерывание при заполнении ФИФО третьей оконечной точки, 0 = прерывание запрещено
бит 4	**EP3_EMPTY_IE:** Endpoint 3 Fifo Empty Interupt Enable. 1 = разрешено прерывание при опустошении ФИФО третьей оконечной точки, 0 = прерывание запрещено
бит 3	**EP2_FULL_IE:** Endpoint 2 Fifo Full Interupt Enable. 1 = разрешено прерывание при заполнении ФИФО второй оконечной точки, 0 = прерывание запрещено
бит 2	**EP2_EMPTY_IE:** Endpoint 2 Fifo Empty Interupt Enable. 1 = разрешено прерывание при опустошении ФИФО второй оконечной точки, 0 = прерывание запрещено
бит 1	**EP1_FULL_IE:** Endpoint 1 Fifo Full Interupt Enable. 1 = разрешено прерывание при заполнении ФИФО первой оконечной точки, 0 = прерывание запрещено
бит 0	**EP1_EMPTY_IE:** Endpoint 1 Fifo Empty Interupt Enable. 1 = разрешено прерывание при опустошении ФИФО первой оконечной точки, 0 = прерывание запрещено

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Таблица 43

Регистр разрешения прерываний от USB блока USB_IE2 (адрес: 15h, банк 4)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
USB_ RST_IE	**USB_ BUSY_I E**	**-**	**-**	**-**	**-**	**-**	**-**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**USB_RST_IE:** USB Reset Interrupt Enable. 1 = при возникновении на линиях USB состояния RESET генерируется прерывание, 0 = прерывание запрещено
бит 6	**USB_BUSY_IE:** USB Busy Interrupt Enable. 1 = при передаче по линиям USB генерируется прерывание, 0 = прерывание запрещено
бит 5	зарезервировано. Примечание: бит должен всегда равняться нулю
бит 4	зарезервировано. Примечание: бит должен всегда равняться нулю
бит 3, 2, 1, 0	зарезервировано

Обозначения:

R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;

-n = значение бита после сброса по включению питания:

1 = установлен;

0 = сброшен;

x = значение не известно.

Таблица 44

Регистр разрешения прерываний от USB блока USB_IE3 (адрес: 16h, банк 4)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUS_ ERR_IE	**DROP_F RM_IE**	**TO_ ERR_IE**	**SEQ_ ERR_IE**	**NSE_ ERR_IE**	**PID_ ERR_IE**	**CRC16_ ERR_IE**	**CRC5_ ERR_IE**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**BUS_ERR_IE:** BUS Error Interrupt Enable. 1 = разрешение прерывания при ошибке на линиях D+, D- или ошибке bit staffing, byte error и т.д. 0 = прерывание запрещено
бит 6	**DROP_FRM_IE:** Packet Dropped Interrupt Enable. 1 = разрешение прерывания для случая отказа в приеме пакета от хост-контроллера по причине отсутствия свободного месте в очереди оконечной точки. 0 = прерывание запрещено
бит 5	**TO_ERR_IE:** Time Out Interrupt Enable. 1 = разрешение прерывания для случая, если хост-контроллер не прислал подтверждения приема данных и они будут посланы контроллером снова, либо после пакета DATA OUT не получен пакет с данными и контроллер вернулся в режим ожидания. 0 = прерывание запрещено
бит 4	**SEQ_ERR_IE:** Sequential Interrupt Enable

	1 = разрешение прерывания для случая, если получен пакет с тем же DATAx что и предыдущий пакет. 0 = прерывание запрещено
бит 3	**NSE_ERR_IE:** Not Such Endpoint Interrupt Enable. 1 = разрешение прерывания в случае, если хост-контроллер обратился к устройству с нашим функциональным адресом, но к несуществующей оконечной точке. 0 = прерывание запрещено
бит 2	**PID_ERR_IE:** PID Field Error Interrupt Enable. 1 = разрешение прерывания в случае, если инверсия поля PID не соответствует прямой части. 0 = прерывание запрещено
бит 1	**CRC16_ERR_IE:** CRC16 Error Interrupt Enable. 1 = разрешение прерывания в случае, если CRC16 полученных данных не соответствует принятому полю. 0 = прерывание запрещено
бит 0	**CRC5_ERR_IE:** CRC5 Error Interrupt Enable. 1 = разрешение прерывания в случае, если CRC5 полученного пакета не соответствует вычисленному значению. 0 = прерывание запрещено

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 45

Регистр разрешения прерываний от USB блока USB_IE4 (адрес: 17h, банк 4)

U-0	U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0
-	-	-	-	USB_ ALL_IE	-	ADR_ SET_IE	CONF_ SET_IE
бит 7	6	5	4	3	2	1	бит 0

бит 7, 6, 5, 4	зарезервировано
бит 3	**USB_ALL_IE:** USB All Interrupt Enable. 1 = глобальное разрешение прерываний от контроллера USB. 0 = прерывания от контроллера USB запрещены
бит 2	зарезервировано
бит 1	**ADR_SET_IE:** Addressed Interrupt Enable. 1 = разрешение прерываний в случае, если контролер получил адрес от хост-контроллера, 0 = прерывание запрещено
бит 0	**CONF_SET_IE:** Configured Interrupt Enable. 1 = разрешение прерывания в случае, если контролер сконфигурирован хост-контроллером, 0 = прерывание запрещено

Обозначения:
R = бит для чтения; W = бит с возможностью записи;

U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры статуса оконечной точки

Таблица 46

Регистры статуса оконечных точек EPx_STAT:
первой-EP1_STAT (адрес: 10h, банк 3), второй-EP2_STAT (адрес: 11h, банк 3),
третьей-EP3_STAT (адрес: 12h, банк 3), четвертой-EP4_STAT (адрес: 13h, банк 3)

U-0	U-0	R-0	R-0	U-0	U-0	R-0	R-0
-	-	BLKx_ DATA	ACTIVE x	-	-	FULLx	EMPTYx
бит 7	6	5	4	3	2	1	бит 0

бит 7, 6	зарезервировано
бит 5	**BLKx_DATA:** 1 = работа ФИФО соответствующей оконечной точки запрещена
бит 4	**ACTIVEx:** 1 = работа ФИФО соответствующей оконечной точки разрешена
бит 3, 2	зарезервировано
бит 1	**FULLx:** 1 = ФИФО соответствующей оконечной точки заполнено. 0 = ФИФО соответствующей оконечной точки не заполнено
бит 0	**EMPTYx:** 1 = ФИФО соотвествующей оконечной точки пусто. 0 = ФИФО соотвествующей оконечной точки не пусто

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры количества слов в очереди оконечной точки

Таблица 47

Регистры количества слов в очередях оконечных точек EPx_CNT:
первой-EP1_CNT (адрес: 14h, банк 3), второй-EP2_CNT (адрес: 15h, банк 3),
третьей-EP3_CNT (адрес: 16h, банк 3), четвертой-EP4_CNT (адрес: 17h, банк 3).

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
-	CNT6	CNT5	CNT4	CNT3	CNT2	CNT1	CNT0
бит 7	6	5	4	3	2	1	бит 0

бит 7	зарезервировано
бит 6…0	**CNT6… CNT0:** Counter of Word in FIFO. Счетчик слов в очереди соответствующей оконечной точки. 00h = очередь полностью пуста, 40h = очередь полностью заполнена

Обозначения:

R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры конфигурации оконечных точек

Таблица 48

Регистры максимального размера пакета оконечных точек EPx_CFG1: первой-EP1_CFG1 (адрес: 10h, банк 0), второй-EP2_CFG1 (адрес: 12h, банк 0), третьей-EP3_CFG1 (адрес: 14h, банк 0), четвертой-EP4_CFG1 (адрес: 16h, банк 0)

U-0	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**
-	**MAX SIZE6**	**MAX SIZE5**	**MAX SIZE4**	**MAX SIZE3**	**MAX SIZE2**	**MAX SIZE1**	**MAX SIZE0**
бит 7	6	5	4	3	2	1	бит 0
бит 7		зарезервировано					
бит 6…0		**MAX SIZE6… MAX SIZE0:** Максимальный размер пакета соответствующей оконечной точки					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 49

Регистры конфигурации оконечных точек EPx_CFG2: первой-EP1_CFG2 (адрес: 11h, банк 0), второй-EP2_CFG2 (адрес: 13h, банк 0), третьей-EP3_CFG2 (адрес: 15h, банк 0), четвертой-EP4_CFG2 (адрес: 17h, банк 0)

U-0	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**U-0**
-	**BLOCK_ EPx**	**TYPE1_ EPx**	**TYPE0_ EPx**	**MODE2_ EPx**	**MODE1_ EPx**	**MODE0_ EPx**	**-**
бит 7	6	5	4	3	2	1	бит 0
бит 7		зарезервировано					
бит 6		**BLOCK_EPx:** 1 = блокирование работы соответствующей оконечной точки					
бит 5, 4		**TYPE1_EPx, TYPE0_EPx:** Transfer Type. Тип обмена соответствующей оконечной точки: 00 = Interrupt Transfer, 01 = Isochronous Transfer, 10 = Bulk Transfer, 11 = недопустимо					
бит 3, 2, 1		**MODE2_EPx…MODE0_EPx:** Endpoint Mode. Режим работы соответствующей оконечной точки: 000 = оконечная точка отключена, 001 = IN оконечная точка, 010 = OUT оконечная точка, остальные – недопустимо					
бит 0		зарезервировано					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистр данных оконечных точек.

Таблица 50

Регистры данных оконечных точек EPx_REG: первой-EP1_REG (адрес: 10h, банк 2), второй-EP2_REG (адрес: 11h, банк 2), третьей-EP3_REG (адрес: 12h, банк 2), четвертой-EP4_REG (адрес: 13h, банк 2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** USB Data. Регистр для записи данных передаваемых по USB, или чтения данных полученных по USB, соответствующей оконечной точки					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Регистры адреса поля дескриптора и данных дескриптора

Таблица 51

Регистр адреса поля дескриптора DESC_ADR (адрес: 10h, банк 1)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	**ADR6**	**ADR5**	**ADR4**	**ADR3**	**ADR2**	**ADR1**	**ADR0**
бит 7	6	5	4	3	2	1	бит 0
бит 7		зарезервировано					
бит 6…0		**ADR6…ADR0:** Descriptor Field Address. Адрес поля дескриптора для доступа на запись или чтение					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Таблица 52

Регистр данных поля дескриптора DESC_DATA (адрес: 11h, банк 1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DATA7	**DATA6**	**DATA5**	**DATA4**	**DATA3**	**DATA2**	**DATA1**	**DATA0**
бит 7	6	5	4	3	2	1	бит 0
бит 7…0		**DATA7…DATA0:** Descriptor Data. Значение поля дескриптора для записи или чтения					

Обозначения:
R = бит для чтения; W = бит с возможностью записи;
U = бит не реализован, читается как 0;
-n = значение бита после сброса по включению питания:
1 = установлен;
0 = сброшен;
x = значение не известно.

Автоматическая инициализация USB интерфейса

Для автоматической инициализации USB интерфейса при подключении устройства к хост-контроллеру необходимо:

- выбрать конфигурацию оконечных точек,
- заполнить в соответствии с конфигурацией оконечных точек дескриптор,
- задать режим работы USB устройства,
- разрешить работу блока USB.

Инициализация устройства, включающая в себя выдачу хост-контроллеру информации о дескрипторе устройства, получения от хост-контроллера функционального адреса и адресации устройства, конфигурирование устройства, происходит автоматически. В ходе инициализации блок контроллера USB отражает текущее состояние посредством регистров статуса и ошибок. Программа, выполняемая на микроконтроллере, должна отслеживать состояние блока контроллера USB, а в случае возникновения ошибок, принять меры по их исправлению. Блок-схема работы микроконтроллера при запуске USB интерфейса приведена на
Рис. **56**.

С момента подсоединения микроконтроллера к хост-контроллеру до момента начала автоматической инициализации пройдет не менее 500 мс. За это время должно включиться питание микроконтроллера, запуститься ядро и начать выполняться программа (в случае если микроконтроллер получает питание от шины USB). Программа должна успеть задать пользовательский дескриптор, сконфигурировать оконечные точки, задать режим работы блока и разрешить его работу. Изначально блок контроллера USB адресуется хост-контроллером по нулевому функциональному адресу. В процессе автоматической инициализации хост-контроллер задает блоку контроллера USB уникальный функциональный адрес. После чего обращается к блоку только по этому адресу. После назначения функционального адреса, его значение отображается в регистре USB_ADR, а в регистре USB_STAT выставляется флаг ADR_SET. После автоматической инициализации драйвер USB устройства, работающий на стороне хост-контроллера, должен задать конфигурацию и интерфейс блоку контроллера USB.

После задания конфигурации в регистре USB_STAT выставляется флаг CONF_SET. После выставления данного флага программа, выполняемая на микроконтроллере, может приступить к приему и передаче данных в соответствии с общей логикой работы устройства.

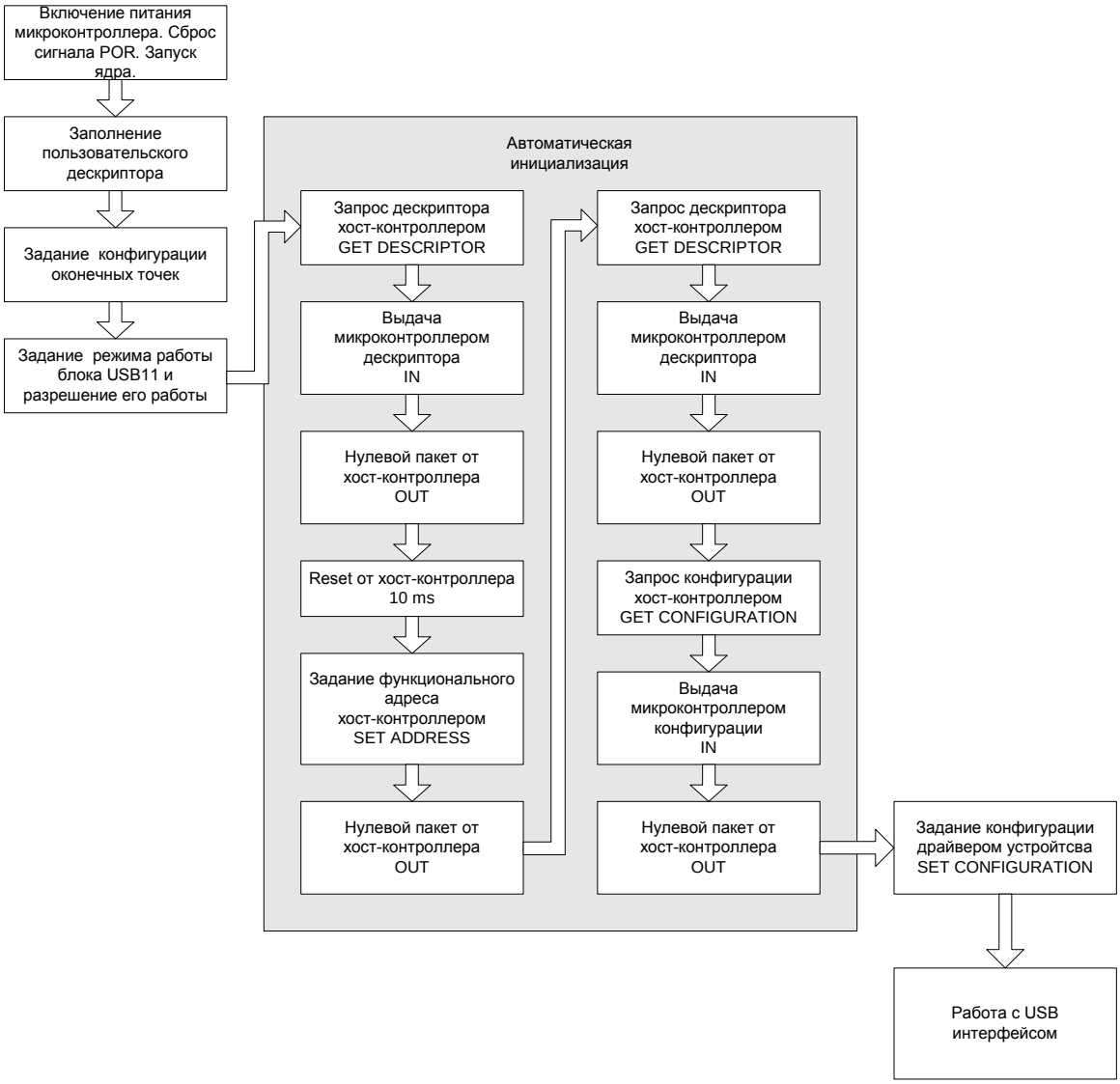


Рис. 56 Блок-схема работы микроконтроллера при запуске USB интерфейса

Задание пользовательского описателя USB устройства

Каждое USB устройство имеет собственное описание (Descriptor). Описание служит для того, что при подключении устройства к хост контроллеру, хост смог получить основную информацию о устройстве, о числе оконечных точек, о размерах буферов и прочее. Для этого разработчик в соответствии со спецификацией шины USB и заложенными в разрабатываемое устройство возможностями должен заполнить описатель в соответствии с Таблица 53. Заполнение описателя производится посредствам записи в регистры DESC_ADR и DESC_DATA. Установив в регистре DESC_ADR адрес нужного поля, в регистр DESC_DATA записывается его значение. Заполнение описания должно производиться до включения блока контроллера USB.

Таблица 53
Дескриптор USB устройства

Адрес поля	Обозначение	Обяза-тельное значение	Рекомен-дуемое значение	Описание
Standard Device Descriptor				
0	bLength	h18	-	Размер этого дескриптора в байтах
1	bDescriptorType	h01	-	Тип дескриптора – DEVICE
2	bcdUSB	h00	-	Номер версии USB спецификации в двоично-десятичном кодировании (т.е., 2.10 обозначается как 210H). Это поле обозначает релиз USB спецификации которому соответствует устройство
3	bcdUSB	h10	-	
4	bDeviceClass	-	hFF	Код класса (назначается USB). Если это поле сбрасывается в ноль, каждый интерфейс в рамках данной конфигурации определяется своим собственным классом и различные интерфейсы работают независимо. Если поле имеет значение от 1 до FEH, то устройство поддерживает различные классы спецификации на различные интерфейсы и различные интерфейсы не могут функционировать независимо. Это значение определяет класс, используемый для объединенных интерфейсов. Если значение поля FFH, то класс устройства - vendor
5	bDeviceSubClass	-	h00	Код подкласса (назначается USB). Эти коды определяются значением поля bDeviceClass field. Если поле bDeviceClass нулевое, это поле так же должно быть нулевым. Если bDeviceClass не установлено в FFH, все значения зарезервированы для назначения через USB
6	bDeviceProtocol	-	hFF	Код протокола (назначается USB).

				Этот код определяется значениями полей bDeviceClass и bDeviceSubClass. Если устройство поддерживает class-specific protocols on a device basis as opposed to an interface basis, этот код определяет протокол, который устройство использует как определено в спецификации класса устройства. Если это поле сброшено в ноль, то устройство не использует class-specific protocols on a device basis. В любом случае оно может использовать class-specific protocols on an interface basis. Если поле установлено в FFH, то устройство использует vendor-specific protocol on a device basis
7	bMaxPacketSize0	h08	-	Максимальный размер пакета для нулевой Оконечной точки.
8	idVendor	-	h34	Vendor ID (назначенный USB)
9	idVendor	-	h12	
10	idProduct	-	h78	Product ID (назначенный производителем)
11	idProduct	-	h56	
12	bcdDevice	-	h00	Номер версии устройства в двоично-десятичном коде.
13	bcdDevice	-	h00	
14	iManufacturer	-	h01	Указатель строки дескриптора определяющего производителя
15	iProduct	-	h02	Указатель строки дескриптора определяющего продукт
16	iSerialNumber	-	h03	Указатель строки дескриптора определяющего серийный номер устройства
17	bNumConfigurations	h01	-	Количество возможных конфигураций
Standard Configuration Descriptor				
18	bLength	h09	-	Размер этого дескриптора в байтах.
19	bDescriptorType	h02	-	Тип дескриптора – CONFIGURATION
20	wTotalLength	h20	-	Общая длина данных возвращаемых данной конфигурацией. Включая длину всех дескрипторов
21	wTotalLength	h00	-	
22	bNumInterfaces	h01	-	Количество интерфейсов,

				поддерживаемых данной конфигурацией.
23	bConfigurationValue	h01	-	Значение, используемое как аргумент для SetConfiguration запроса для выбора этой конфигурации.
24	iConfiguration	h00	-	Указатель на строку дескриптора, описывающую эту конфигурацию.
25	bmAttributes	-	hC0	Характеристики конфигурации: D7: Зарезервировано (уст. в единицу) D6: =1, если есть собственное питание D5: =0 D4...0: = 0
26	MaxPower	-	hA0	Максимальная мощность, потребляемая устройством, от шины USB. Указывается в 2 мА единицах, т.е. 50=100 мА
Standard Interface Descriptor				
27	bLength	h09	-	Размер этого дескриптора в байтах
28	BDescriptorType	h04	-	Тип дескриптора – INTERFACE
29	bInterfaceNumber	h00		Номер интерфейса. Zero-based значение, указывающее на используемый интерфейс из множества поддерживаемых конфигурацией интерфейсов
30	bAlternateSetting	h00	-	Значение, используемое для выбора альтернативного интерфейса
31	bNumEndpoints	-	h04	Количество оконечных точек, используемых данным интерфейсом (исключая нулевую оконечную точку). Если это значение нулевое, этот интерфейс может использоваться только как Default Control Pipe
32	bInterfaceClass	-	hFF	Код класса (назначается USB). Нулевое значение зарезервировано для будущей стандартизации. Если поле установлено в FFH, класс интерфейса - vendor. Все остальные значения зарезервированы для назначения через USB
33	bInterfaceSubClass	-	h01	Код подкласса (назначается USB). Этот код определяется полем

				bInterfaceClass. Если поле bInterfaceClass сброшено в ноль, это поле должно также быть сброшено в 0. Если поле bInterfaceClass не установлено в FFH, все значения этого поля зарезервированы для назначения через USB
34	bInterfaceProtocol	-	hFF	Код протокола (назначается USB). Этот код определяется полями bInterfaceClass и bInterfaceSubClass. Если интерфейс поддерживает class-specific запросы, этот код определяет протоколы, которые использует устройство как определено спецификацией класса устройства. Если это поле сброшено в ноль, то устройство не использует class-specific протокол на этом интерфейсе. Если это поле установлено в FFH, то устройство использует vendor-specific протокол для этого интерфейса
35	iInterface	h00		Указатель на строку дескриптора, описывающую этот интерфейс.
Standard Endpoint 1 Descriptor				
36	bLength	h07	-	Размер этого дескриптора в байтах.
37	bDescriptorType	h05	-	Тип дескриптора – ENDPOINT
38	bEndpointAddress	-	h81	Адрес оконечной точки устройства USB, описанного в этом дескрипторе: Bit 3...0: Номер оконечной точки. Bit 6...4: Зарезервировано, сбрасывается в ноль. Bit 7: Направление передачи, игнорируется для control точек (0 = OUT оконечная точка, 1 = IN оконечная точка)
39	bmAttributes	-	h03	Это поле содержит атрибуты оконечных точек, когда они сконфигурированы в bConfigurationValue. Бит 1..0: Тип передачи: 00 = Control 01 = Isochronous

				10 = Bulk 11 = Interrupt Остальные биты зарезервированы
40	wMaxPacketSize	-	h08	Максимальный размер пакета, который может принять или передать эта точка, не более 64
41	wMaxPacketSize	h00	-	
42	Binterval	-	h01	Интервал опроса оконечной точки для передачи данных (в миллисекундах). Это поле игнорируется для bulk and control оконечных точек. Для isochronous точки должно быть установлено в 1. Для interrupt точек может иметь значение от 1 до 255
Standard Endpoint 2 Descriptor				
43	bLength	h07	-	Размер этого дескриптора в байтах
44	bDescriptorType	h05	-	Тип дескриптора – ENDPOINT
45	bEndpointAddress	-	h02	Адрес оконечной точки устройства USB, описанного в этом дескрипторе: Bit 3...0: Номер оконечной точки. Bit 6...4: Зарезервировано, сбрасывается в ноль. Bit 7: Направление передачи, игнорируется для control точек (0 = OUT оконечная точка, 1 = IN оконечная точка)
46	bmAttributes	-	h03	Это поле содержит атрибуты оконечных точек, когда они сконфигурированы в bConfigurationValue. Бит 1..0: Тип передачи: 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt Остальные биты зарезервированы
47	wMaxPacketSize	-	h08	Максимальный размер пакета, который может принять или передать эта точка, не более 64
48	wMaxPacketSize	h00	-	
49	Binterval	-	h01	Интервал опроса оконечной точки для передачи данных (в миллисекундах). Это поле

				игнорируется для bulk and control оконечных точек. Для isochronous точки должно быть установлено в 1. Для interrupt точек может иметь значение от 1 до 255
Standard Endpoint 3, 4 Descriptor				
50-56	…	…	…	Аналогично для дескриптора 3 оконечной точки
57-63	…	…	…	Аналогично для дескриптора 4 оконечной точки
Codes Representing Languages Supported by the Device				
64	bLength	h08	-	Размер этого дескриптора в байтах
65	bDescriptorType	h03	-	Тип дескриптора – STRING
66	wLANGID[0]	-	h09	LANGID code 0
67	wLANGID[1]	-	h04	
68	wLANGID[2]	-	h09	LANGID code 1
69	wLANGID[3]	-	h04	
70	wLANGID[4]	-	h09	LANGID code 2
71	wLANGID[5]	-	h04	
UNICODE String 1 Descriptor				
72	bLength	h0A	-	Размер этого дескриптора в байтах.
73	bDescriptorType	h03	-	Тип дескриптора – STRING.
74	BString[0]	-	h00	UNICODE encoded string
	…			
81	BString[7]	-	h00	
UNICODE String 2 Descriptor				
82	bLength	h0A	-	Размер этого дескриптора в байтах.
83	bDescriptorType	h03	-	Тип дескриптора – STRING.
84	BString[0]	-	h00	UNICODE encoded string
	…			
91	BString[7]	-	h00	
UNICODE String 3 Descriptor				
92	bLength	h0A	-	Размер этого дескриптора в байтах.
93	bDescriptorType	h03	-	Тип дескриптора – STRING.
94	BString[0]	-	h00	UNICODE encoded string
	…			
101	BString[7]	-	h00	
UNICODE String 4 Descriptor				
102	bLength	h1A	-	Размер этого дескриптора в байтах.
103	bDescriptorType	h03	-	Тип дескриптора – STRING.
104	BString[0]	-	h00	UNICODE encoded string
	…			
127	BString[23]	-	h00	

Режим передачи IN

В режиме передачи IN, данные передаются от микроконтроллера к хост-контроллеру (см.
Рис. **57**). Хост-контроллер выдает на шину USB пакет типа IN, с указанием функционального адреса устройства и номера оконечной точки к которой обращается. В ответ устройство, которому в процессе инициализации, был присвоен данный адрес должно выставить пакет типа DATA0 или DATA1 и в поле данных передать данные, которые были заранее записаны в FIFO очередь оконечной точки. В ответ, после получения данных, хост-контроллер выдает пакет типа ACK. Данные считаются переданными.

Во время передачи могут возникнуть какие либо коллизии и ошибки, например, ошибки контрольных сумм CRC5 или CRC16, обращения к несуществующим оконечным точкам, отсутствие устройства, к которому обращается хост-контроллер, данных для передачи и т.п. В этом случае, данные считаются не переданными, и ситуации такого рода обрабатываются блоком контроллера USB автоматически в зависимости от типа обмена.

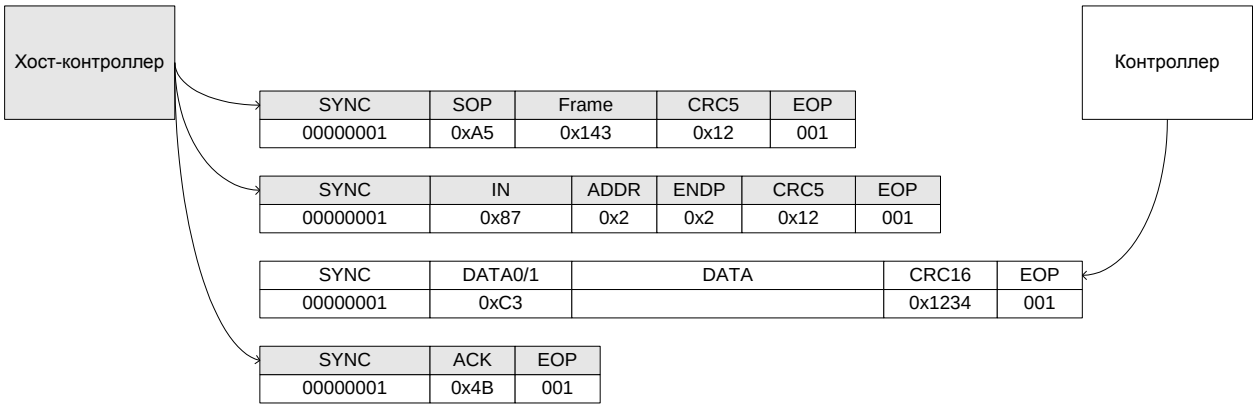


Рис. 57 Режим передачи IN

При получении блоком контроллера USB от хост-контроллера пакета типа IN. Контроллер USB, начинает последовательно извлекать из FIFO очереди оконечной точки данные и, кодируя передавать их, до тех пор пока либо не будут извлечены все данные, либо не будет превышен максимальный размер пакета. После чего передача пакета прекратится. В случае принятия данных хост-контроллером, им будет выставлен пакет подтверждения ACK. После получения от хост-контроллера пакета типа ACK, блок контроллера USB зафиксирует факт передачи, и данные будут считаться переданными. Предположим, что при передаче данных хост-контроллеру произошла ошибка, при которой данные были не переданы. В этом случае блок контроллера USB осуществляет «откат» передачи, и восстановит данные в FIFO очереди. Поскольку в процессе передачи данных по USB, ядро микроконтроллера может продолжать заполнять FIFO очередь оконечной точки, то может возникнуть ситуация когда в FIFO очереди может не остаться места для восстановления непреданных данных. В этом случае, если во время передачи данных число свободных ячеек FIFO будет меньше максимального размера пакета, блок контроллера USB заблокирует возможность записи в FIFO данных со стороны ядра и периферийных устройств, выставив в регистр статуса оконечной точки EPx_STAT флаги BLKx_DATA и FULLx.

Режим передачи OUT

В режиме передачи OUT данные передаются от хост-контроллера к микроконтроллеру (см.
Рис. **58**). Хост-контроллер выдает на шину USB пакет типа OUT, с указанием функционального адреса устройства и номера оконечной точки к которой обращается. Вслед за этим выдается пакет типа DATA0 или DATA1 и в поле данных передаются данные. В ответ, после получения данных, микроконтроллер выдает пакет типа ACK. Данные считаются переданными.

В процессе передачи могут возникнуть какие либо коллизии или ошибки, например, ошибки контрольных сумм CRC5 или CRC16, обращения к несуществующим оконечным точкам и т.п. В этом случае блок контроллера USB не выдаст подтверждения передачи хост-контроллеру, и данные не будут считаться переданными. В этом случае хост-контроллер повторит передачу пакета с тем же типом DATA0 или DATA1, что был у не переданного пакета. Однако, может возникнуть ситуация, когда блок контроллера USB принял данные, и ответил хост-контроллеру пакетом типа ACK. Но хост-контроллер не принял данного пакета. То, в этом случае, хост-контроллер так же повторит передачу, но на основании того, что блоку контроллера USB пришел пакет с тем же типом что и предыдущий принятый пакет (DATA0 или DATA1), то данные, передаваемые в этом пакете, будут проигнорированы, но хост-контроллеру повторно будет выдан пакет подтверждения ACK.

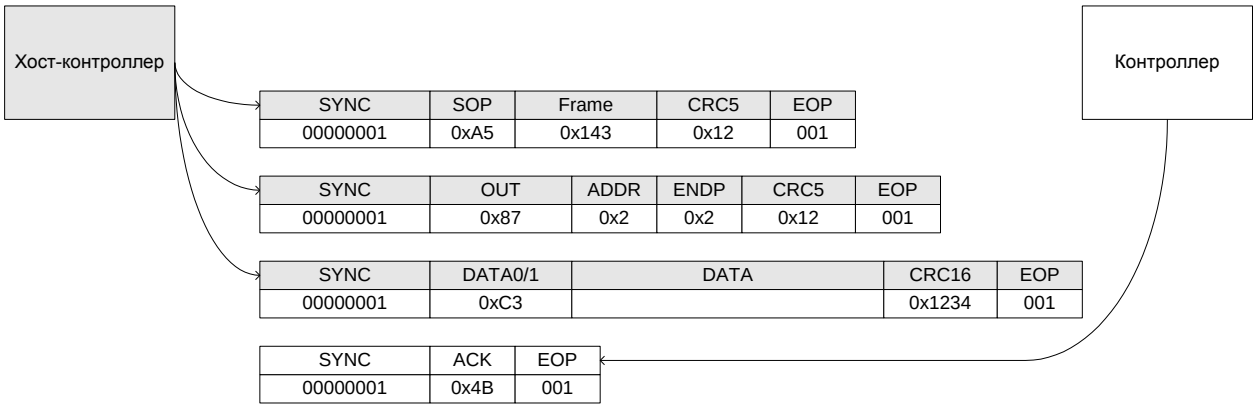


Рис. 58 Режим передачи OUT

При получении блоком контроллера USB от хост-контроллера пакета типа OUT, блок переходит в режим приема данных. И при получении пакета типа DATA0 или DATA1 с данными, последовательно записывает их в FIFO очередь оконечной точки. После получения контрольной суммы, если будет обнаружена ошибка, блок USB выполнит «откат» и удалит из FIFO очереди данные, полученные с ошибкой. При этом хост-контроллеру не будет выставлен пакет подтверждения ACK. Кроме того, в процессе приема данных, может возникнуть переполнение FIFO очереди оконечной точки, в этом случае блок контроллера USB так же выполнит «откат» уже принятых данных, а хост-контроллеру не выдаст подтверждения ACK. В этой ситуации предполагается, что ядро микроконтроллера к началу повторной передачи считает из FIFO ранее принятые данные, что позволит корректно обработать повторную передачу. Поскольку ядро микроконтроллера может читать данные из FIFO очереди в процессе приема пакета, то может возникнуть ситуация, когда блок контроллера USB записал в очередь данные, корректность которых не может быть установлена до окончания пакета, а ядро микроконтроллера уже начинает их

считывать. Для предотвращения этой коллизии, на время получения пакета типа OUT, при количестве данных в FIFO очереди меньше чем максимальный размер пакета, блоком контроллера USB будет заблокирован доступ к очереди со стороны ядра и в регистре EPx_STAT будет выставлен флаг BLKx_DATA и EMPTYx.

Тип обмена: Interrupt Transaction

Передачи типа Interrupt могут выполнятся в режимах передачи IN и OUT. Передача осуществляется один раз за фрейм (1 мс), что минимально загружает шину USB. При передаче доставка данных гарантируется. Передачи типа Interrupt Transaction могут выполняться в режимах Low Speed и Full Speed.

Когда приходит пакет типа IN, блок USB может возвратить хост-контроллеру данные, NAK или STALL. Если блок USB не имеет данных, необходимых для отправки хост-контроллеру, он возвращает NAK. Если блок USB переведен в режим Halt, то он возвращает STALL. Если у блока USB есть данные, которые можно передать, то он возвращает пакет данных. После получения данных хост-контроллером без ошибок, он возвращает блоку USB пакет ACK. Данные считаются переданными. Если ACK от хост-контроллера не был получен, то блок USB восстановит в FIFO очереди переданные данные и повторит их передачу в следующем пакете. Когда приходит пакет типа OUT, блок USB начинает ожидать пакет с данными. При получении пакета данных с корректным DATA0 или DATA1 и контрольной суммой, и все полученные данные смогли разместиться в FIFO очереди, то блок USB возвращает хост-контроллеру ACK. Если же произошла какая-либо ошибка в принимаемом пакете, или FIFO очереди оконечной точки не хватило для всех данных, то блок USB вернет хост-контроллеру NAK. Хост-контроллер, получив от блока USB NAK, будет повторять передачу до тех пор пока передача не будет выполнена или остановлена работа. Если блок USB переведен в режим Halt, то он возвращает STALL. Таким образом, обеспечивается гарантированная доставка данных.

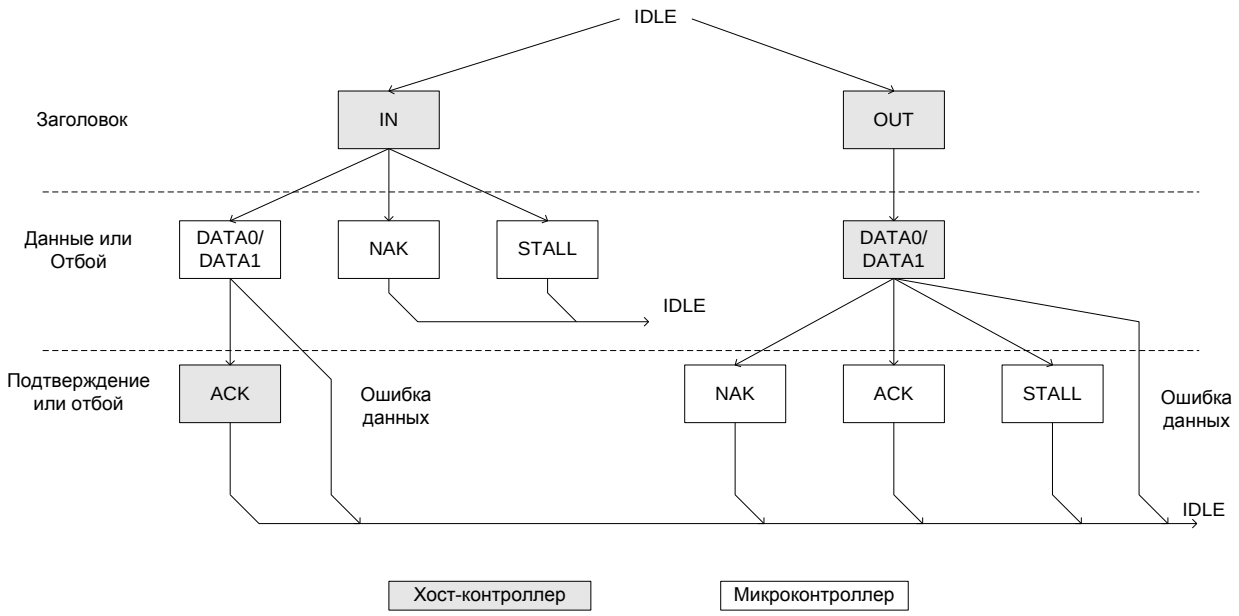


Рис. 59 Тип обмена Interrupt Transaction

Тип обмена: Bulk Transaction

Передачи типа Bulk могут выполняться в режимах передачи IN и OUT. Передача может осуществляться многократно за один фрейм (1 мс) насколько позволяет общая загруженность шины. При передаче доставка данных гарантируется. Передачи типа Bulk Transaction могут выполняться только в режиме Full Speed. По алгоритму работы и подтверждениям принятия данных Bulk Transaction полностью аналогичны Interrupt Transactions.

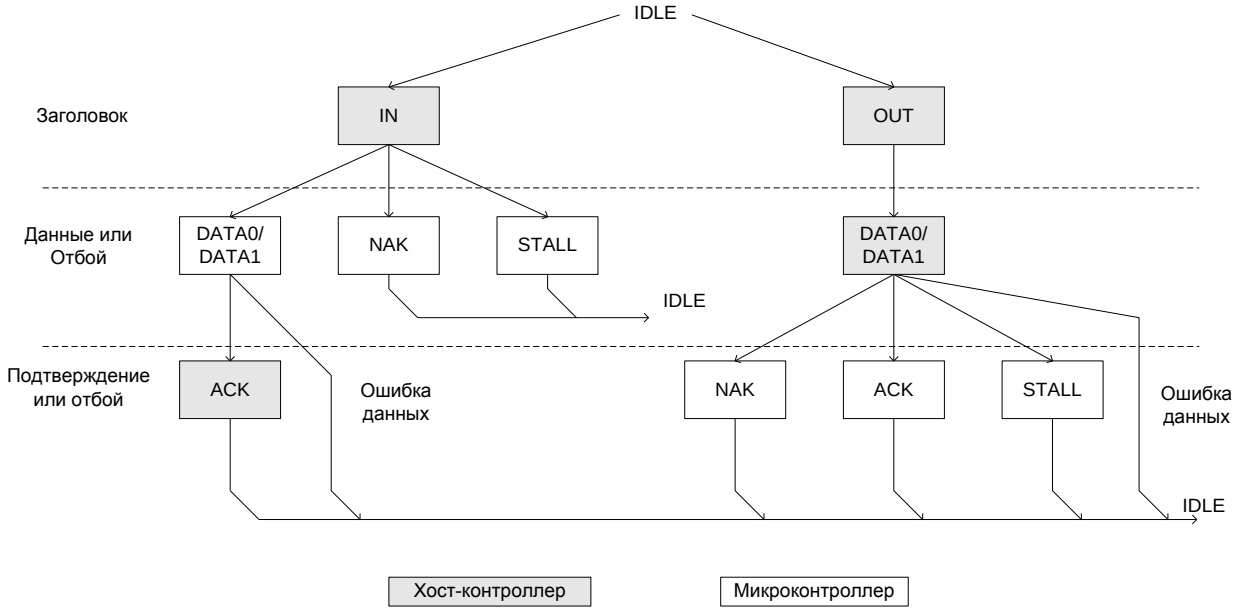


Рис. 60 Тип обмена Bulk Transaction

Тип обмена: Isochronous Transaction

Передачи типа Isochronous могут выполняться в режимах передачи IN и OUT. Передача может осуществляться многократно за один фрейм насколько позволяет общая загруженность шины. При передаче доставка данных не гарантируется, так как не происходит подтверждений. При этом обеспечивается максимальная латентность при передаче. Передачи типа Isochronous Transaction могут выполняться только в режиме Full Speed. По алгоритму работы Isochronous Transaction не предполагают подтверждение приема данных, что не позволяет гарантировать доставку данных и их целостность. Если при приеме данных выявлена ошибка контрольной суммы, то все данные пакета будут отброшены. Если во время приема данных произошло переполнение FIFO очереди, то данные не поместившиеся в FIFO будут отброшены.

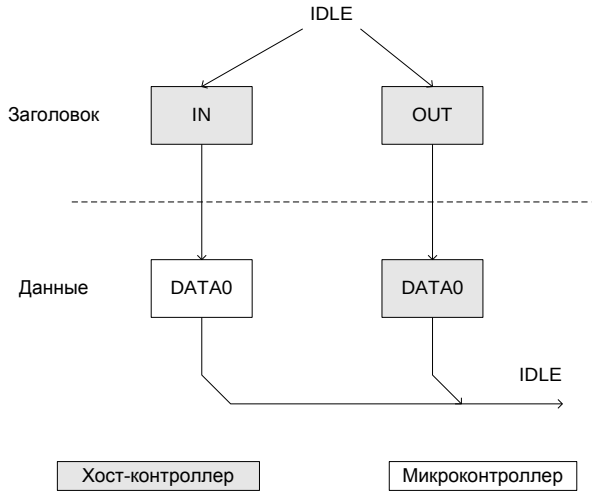


Рис. 61 Тип обмена Isochronous Transaction

Специальные модули микроконтроллера

Регистры конфигурации микроконтроллера

Регистры конфигурации микроконтроллера находятся в памяти программ и записываются при программировании микроконтроллера (смотрите спецификацию по программированию).

Запись битов конфигурации производится побитно, по соответствующим этим битам адресам слов (см. Таблица 54). Чтение битов PM1, PM0, WDTPS1, WDTPS0, FOSC1 и FOSC0 возможно по любому адресу в диапазоне FE00h-FE07h, а битов PM2 и BODEN по любому адресу в диапазоне FE08h-FE0Fh.

Значения битов конфигурации дублируются в ПЗУ программ по адресам 7FF0h-7FFFh. Значению бита =0 соответствует значение ячейки ПЗУ равное FFFFh, а значению бита =1 - значение ячейки равное 0000h. Соответствие адресов ячеек ПЗУ битам конфигурации указано в таблице. Область памяти программ 7FF0h-7FFFh доступна только для чтения.

Примечание:

Если в ячейки с адресами 7FF4h, 7FF6h и 7FFFh записано значение FFFFh, то при «сбросе» микроконтроллер перейдет в режим защиты кода программ, т.е. считывание и стирание памяти программ заблокируется. Для предотвращения этого необходимо произвести стирание ячеек памяти до поступления сигнала «сброс».

Режим микроконтроллера с защитой кодов программы (режим «защищенного микроконтроллера») позволяет защитить содержимое внутренней памяти программ микроконтроллера от считывания или модификации внешним устройством (программатором). После установки этого режима блокируется доступ программатора к внутренней памяти программ, т.е. выполнение операций стирания, чтения, верификации и записи памяти программ, а также регистров конфигурации, становится не возможным. Микроконтроллер, находящийся в «защищенном» режиме, игнорирует все команды, поступающие от программатора, и не отвечает на них. Программа управления программатором в этом случае не может определить наличие программируемого микроконтроллера и выдает соответствующую ошибку.

Установка режима «защищенного микроконтроллера» не влияет на выполнение команд микроконтроллера, осуществляющих чтение/запись таблиц в памяти программ (TABLRD и TABLWT). Установка режима «защищенного микроконтроллера» не влияет на функционирование программы, записанной в память программ микроконтроллера.

Перевод микроконтроллера в режим «защищенного микроконтроллера» производится записью соответствующей комбинации битов в регистры конфигурации микроконтроллера (см. Таблица 54).

Внимание!
Перед программированием микроконтроллера проверьте правильность установленной конфигурации микроконтроллера. Установка режима «защищенного микроконтроллера» заблокирует возможность изменения содержимого внутренней памяти программ микроконтроллера и его конфигурации.

Таблица 54

Регистры конфигурации микроконтроллера

	FE0Fh	**FE0Eh**						
бит 15 - 8	бит 7	6	5	4	3	2	1	бит 0
-	**PM2**	**BODEN**	**-**	**-**	**-**	**-**	**-**	**-**

		FE06h		**FE04h**	**FE03h**	**FE02h**	**FE01h**	**FE00h**
бит 15 - 8	бит 7	6	5	4	3	2	1	бит 0
-	**-**	**PM1**	**-**	**PM0**	**WDTPS1**	**WDTPS0**	**FOSC1**	**FOSC0**

PM2: бит 7, адрес FE0Fh (ячейка ПЗУ: 7FFFh) **PM1:** бит 6, адрес FE06h (ячейка ПЗУ: 7FF6h) **PM0:** бит 4, адрес FE04h (ячейка ПЗУ: 7FF4h)	PM2, PM1, PM0 - биты выбора режима микроконтроллера: 111 = режим микропроцессора 110 = режим микроконтроллера 101 = расширенный режим микроконтроллера 000 = режим микроконтроллера с защитой кодов программы (т.е. запрет считывания содержимого внутренней памяти программ)
BODEN: бит 6,адрес FE0Eh (ячейка ПЗУ: 7FFEh)	BODEN - включение схемы сброса по снижению напряжения питания: 1 = схема включена 0 = схема выключена
WDTPS1:бит 3,адрес FE03h (ячейка ПЗУ: 7FF3h) WDTPS0:бит 2,адрес FE02h (ячейка ПЗУ: 7FF2h)	WDTPS1, WDTPS0 - выбор предделителя «сторожевого таймера»: 11 = «сторожевой таймер» включен, предделитель = 1 10 = «сторожевой таймер» включен, предделитель = 256 01 = «сторожевой таймер» включен, предделитель = 64 00 = «сторожевой таймер» выключен, работает как 16-ти разрядный таймер переполнения
FOSC1: бит 1, адрес FE01h (ячейка ПЗУ: 7FF1h) **FOSC0:** бит 0, адрес FE00h (ячейка ПЗУ: 7FF0h)	FOSC1, FOSC0 - выбор режима тактового генератора: 11 = EC – режим подачи внешнего тактового сигнала, 10 = XT – генератор с внешним кварцевым или керамическим резонатором (частота от 2МГц до 33 МГц), 01 = RC генератор с частотой до 4 МГц, 00 = LF – работа на частоте USBCLK/2

Обозначение: - = зарезервировано, читается как 0.

Обозначения: - = зарезервировано, читается как 0.

Внутрисхемное программирование микроконтроллера

Для программирования внутренней FLASH памяти микроконтроллера используется последовательный интерфейс. Для этого задействуются семь выводов микросхемы (см. Таблица 55). Это позволяет производить программирование микроконтроллеров установленных на печатную плату. Подробнее смотрите в спецификации по программированию.

Таблица 55

Выводы, используемые для программирования

Обозначение	В режиме программирования		
	Назначение	**Тип**	**Описание**
PA2/RX/DT	DT	I/O	последовательные данные
PA3/TX/CK	CK	I	последовательные тактовые импульсы
PA1/T0CLK	OSCI	I	вход синхронизации микроконтроллера
TEST	TEST	I	вход выбора тестового режима
MCLR_UPP	MCLR/UPP	P	напряжение программирования
Ucc	Ucc	P	напряжение питания
Uss	Uss	P	«земля»

Сторожевой таймер

Сторожевой таймер (WDT) предназначен для восстановления микроконтроллера при сбоях Сторожевой таймер (WDT) предназначен для восстановления микроконтроллера при сбоях или сброса устройства во время его нахождения в режиме SLEEP. Для повышения надежности сторожевой таймер имеет собственный RC генератор. Он работает даже при отсутствии тактовой частоты микроконтроллера. Режим сторожевого таймера программируется битами конфигурации в регистрах конфигурации микроконтроллера. Во время нормальной работы WDT должен очищаться через определенные интервалы времени. Это время должно быть меньше, чем минимальное время переполнения WDT, в противном случае переполнение WDT произведет сброс устройства.

Номинальный период сторожевого таймера (с предделителем = 1) составляет около 12 мс. Это время зависит от температуры и напряжения питания. Для увеличения периода таймера можно включить предделители с большим коэффициентом деления. Сторожевой таймер и его предделитель сбрасываются командами CLRWDT и SLEEP, сигналом «сброса» и при выходе из режима SLEEP по прерыванию. Таймер начинает счет сразу же после окончания сигнала «сброс». Бит TO в регистре CPUSTA будет сброшен при переполнении сторожевого таймера. Если сторожевой таймер включен в режиме обычного таймера, то на него подаются импульсы с генератора тактовой частоты микроконтроллера. Время переполнения составляет 65536 тактов T_C. При переполнении сбрасывается бит TO в регистре CPUSTA, но устройство не сбрасывается. Команда CLRWDT устанавливает этот бит. Регистры сторожевого таймера и его предделителя не доступны для чтения/записи. Таймер в этом режиме останавливается в режиме SLEEP.

Режим энергосбережения (SLEEP)

Микроконтроллер переходит в режим энергосбережения при выполнении команды SLEEP. При этом сбрасывается сторожевой таймер и его предделитель (если они включены), бит PD сбрасывается, бит TO устанавливается (регистр CPUSTA), выключается генератор тактовой частоты микроконтроллера, порты ввода/вывода сохраняют свое состояние.

Следующие события могут вывести микроконтроллер из режима SLEEP:

- Сброс при включении или сброс при снижении питания.
- Подача сигнала сброс на внешний вход сброса MCLR.
- Сброс от сторожевого таймера (если он включен).
- Прерывание с вывода PA0/INT, прерывание от PA1/T0CLK, или некоторые периферийные прерывания (USART, USB, NAND и т.п.).

Другие периферийные устройства не могут генерировать прерывания в режиме SLEEP, так как выключен тактовый генератор микроконтроллера. Кроме того, ряд периферийных блоков такие, как контроллер внутренней EERPOM памяти и регулятор напряжения в 3.3 В могут отключаться при переводе микроконтроллера в режим SLEEP. Это обусловлено наличием источников опорных напряжений, которые потребляют энергию постоянно и их потребление не зависит от режима работы микроконтроллера. После выхода из режима SLEEP данные блоки должны быть включены снова.

Любой сигнал «сброса» вызовет сброс устройства. Прерывания продолжат выполнение программы. Биты TO и PD в регистре CPUSTA могут быть использованы для определения причины сброса устройства. Устанавливаемый при включении бит PD, сбрасывается при переходе в режим SLEEP. Бит TO сбрасывается при переполнении сторожевого таймера.

При выполнении команды SLEEP, предварительно выбирается следующая команда (PC+1). Чтобы пробудить устройство прерыванием, должен быть установлен соответствующий бит разрешения прерывания. Это происходит независимо от состояния бита GLINTD. Если бит GLINTD установлен, устройство продолжает выполнение программы. Если бит GLINTD сброшен, то устройство выполняет команду, следующую за SLEEP, и затем переходит к адресу вектора прерывания. В случаях, где исполнение команды после SLEEP нежелательно, необходимо ставить NOP после команды SLEEP. Сторожевой таймер сбрасывается при выходе устройства из режима SLEEP, независимо от источника пробуждения.

Если установлен XT или LF режим генератора тактовой частоты микроконтроллера, то при выходе из SLEEP происходит запуск «таймера запуска генератора», который будет держать устройство в состоянии «сброса» в течение 1024 тактов T_C.

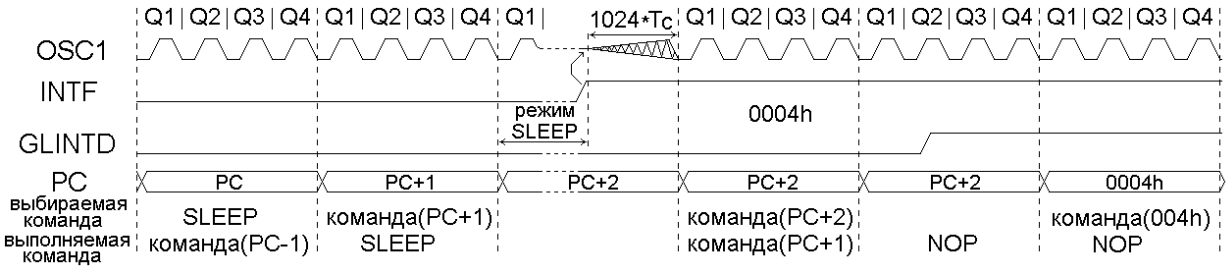


Рис. 62 Выход из режима SLEEP по прерыванию

Примечания:

1. Режим генератора XT или LF. В RC режиме задержки запуска генератора ($1024*T_C$) не будет.
2. Если GLINTD=0, процессор переходит к программе обработки прерываний, если GLINTD=1, то продолжится выполнение программы.

Регулятор питания на 3.3 вольта

Для обеспечения питания блока аналогового приемопередатчика USB и для возможности питания пользовательских портов ввода/вывода от напряжения 3.3 вольта, внутри микроконтроллера реализован регулятор напряжения, обеспечивающий выработку напряжения 3.3 вольта из 5 вольт питания ядра микроконтроллера. Нагрузочная способность регулятора напряжения до 40 мА. Стабильность вырабатываемого напряжения ±5%. Вырабатываемое напряжение выдается на вывод U_{33out}.

Блок аналогового приемопередатчика USB получает питание напрямую от регулятора. Для подачи питания на порты ввода/вывода с вывода U_{33out} питание подается по печатной плате на выводы питания портов $U_{cc1port}$ и $U_{cc2port}$. Если ток потребления портов превосходит 50 мА, то для питания портов рекомендуется использовать внешний регулятор напряжения. Выводы питания $U_{cc1port}$ и $U_{cc2port}$ не объединяются внутри микросхемы, но они должны быть объединены на печатной плате.

Подключение напряжения питания

Все шины питания U_{CC} (положительный вывод напряжения питания микроконтроллера) и U_{SS} («земляной» вывод напряжения питания микроконтроллера) должны быть подключены к соответствующим шинам питания на печатной плате. Непосредственно у выводов микроконтроллера между шинами питания устанавливаются керамические конденсаторы (не менее 5 шт.) емкостью не менее 0.1 мкФ и электролитический конденсатор емкостью не менее 33 мкФ. Между «земляной» шиной и выходом регулятора напряжения U_{33out} подключаются керамический конденсатор емкостью не менее 0.1 мкФ и электролитический конденсатор емкостью не менее 33 мкФ. Выводы напряжения питания портов $U_{cc1port}$ и $U_{cc2port}$ соединяются между собой и подключаются либо к напряжению питания микроконтроллера, либо к выходу встроенного регулятора напряжения на 3.3 вольта, либо к другому источнику напряжения в диапазоне от 3 вольт до напряжения питания микроконтроллера. Между каждым из выводов $U_{cc1port}$, $U_{cc2port}$ и «земляной» шиной питания рекомендуется устанавливать по керамическому конденсатору емкостью не менее 0.1 мкФ.

Система команд

Микроконтроллер поддерживает 58 команд (см. Таблица 54). Все команды 16-ти разрядные. Команды выполняются за один цикл, состоящий из четырех периодов тактовой частоты, за исключением выполняемых за два цикла команд переходов и

команд изменяющих значение программного счетчика PC (т.е. результат операции записывается в PCL), а также команд чтения/записи таблиц в памяти программ (запись во внутреннюю FLASH память имеет большую длительность). Коды команд приведены в Таблица 55. Неиспользуемые коды команд зарезервированы, их применение не рекомендуется. Есть некоторые особенности использования команд:

- если результат выполнения операции записывается в регистр ALUSTA, то флаги Z, C, DC и OV меняя свое значение после выполнения команды, изменят записанный результат.
- операции с регистром PCL: чтение PCL приводит к загрузке в PCLATH значения регистра PCH, запись и чтение-модификация-запись приводит к загрузке в PCH значения регистра PCLATH.
- необходимо учитывать, что команды битовых операций производят операцию «чтение-модификация-запись» целого регистра.

Таблица 56
Набор команд

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
ADDLW k	Содержимое регистра WREG складывается с 8-ми битной константой «k» (k=0…255) и результат помещается в регистр WREG	OV, C, DC, Z	1
ADDWF f,d	Сложение содержимого регистров WREG и «f» (f = 0…255). Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f»	OV, C, DC, Z	1
ADDWFC f,d	Сложение содержимого регистров WREG, бита переноса и содержимого регистра «f» (f = 0…255). Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f»	OV, C, DC, Z	1
ANDLW k	Логическая операция «И» 8-ми битной константы «k» (k = 0…255) и содержимого регистра WREG. Результат помещается в регистр WREG	Z	1
ANDWF f,d	Логическая операция «И» содержимого регистров WREG и «f». Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	Z	1
BCF f,b	Сброс бита «b» в регистре «f» (b = 0…7, f = 0…255)	-	1
BSF f,b	Установка в единицу бита «b» в регистре «f» (b=0…7, f = 0…255)	-	1
BTFSC f,b	Если бит «b» в регистре «f» равен 0, тогда следующая команда пропускается, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла (b = 0…7, f = 0…255)	-	1(2)

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
BTFSS f,b	Если бит «b» в регистре «f» равен 1, тогда следующая команда пропускается, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла (b = 0…7, f = 0…255)	-	1(2)
BTG f,b	Инвертирование бита «b» в регистре «f» (b = 0…7, f = 0…255)	-	1
CALL k	Вызов подпрограммы находящейся в пределах страницы 8Кслов. Адрес следующей после CALL команды (PC+1) помещается в стек. 13-ти битный адрес, содержащийся в коде команды, загружается в счетчик команд PC <12:0>. Затем старшие 8 бит PC копируются в PCLATH. Команда CALL выполняется за два цикла. Для вызова подпрограмм за пределами 8Кслов, смотрите команду LCALL	-	2
CLRF f,s	Сбрасывает (обнуляет) регистр «f» (f = 0…255). Если s=0: сбрасываются регистры «f» и WREG, если s=1: сбрасывается регистр «f»	-	1
CLRWDT	Сбрасывает «Сторожевой таймер» и его предделитель. Устанавливает биты TO, PD в «1»	TO=1, PD=1	1
COMF f,d	Инвертирование битов регистра «f» (f = 0…255). Если d=0, то результат сохраняется в регистре WREG, если d=1, то результат сохраняется в регистре «f»	Z	1
CPFSEQ f	Сравнение содержимого регистра «f» (f = 0…255) с содержимым регистра WREG путем беззнакового вычитания. Если (f) = (WREG), то вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла	-	1(2)
CPFSGT f	Сравнение содержимого регистра «f» (f = 0…255) с содержимым регистра WREG путем беззнакового вычитания. Если (f) > (WREG), то вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла	-	1(2)

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
CPFSLT f	Сравнение содержимого регистра «f» (f = 0…255) с содержимым регистра WREG путем беззнакового вычитания. Если (f) < (WREG), то вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла	-	1(2)
DAW f,s	Команда производит десятичную коррекцию результата сложения (регистр WREG) двух чисел в упакованном формате BCD. Если s=0, то результат десятичной коррекции помещается в «f» и в WREG, если s=1, то результат помещается в «f» (f = 0…255). Выполнение операции: Если: [[WREG<7:4> > 9].OR.[C = 1]].AND.[WREG<3:0> > 9], то: WREG<7:4> + 7→ f<7:4>, s<7:4>. Если: [WREG<7:4> > 9].OR.[C = 1], то WREG<7:4> + 6→ f<7:4>, s<7:4>, иначе WREG<7:4>→ f<7:4>, s<7:4>. Если: [WREG<3:0> > 9].OR.[DC = 1], то WREG<3:0> + 6→ f<3:0>, s<3:0>, иначе WREG<3:0>→ f<3:0>, s<3:0>	C	1
DECF f,d	Уменьшение значения регистра «f» на единицу (f = 0…255). Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f»	OV, C, DC, Z	1
DECFSZ f,d	Уменьшение значения регистра «f» на единицу и пропуск следующей команды если результат равен нулю, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение занимает 2 цикла. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1(2)
DCFSNZ f,d	Уменьшение значения регистра «f» на единицу и пропуск следующей команды если результат не равен нулю, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение занимает 2 цикла. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1(2)

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
GOTO k	Безусловный переход по программе в пределах страницы 8Кслов. 13-ти битный адрес, содержащийся в коде команды, загружается в счетчик команд PC<12:0>. Затем старшие 8 бит PC копируются в PCLATH. Команда выполняется за 2 цикла	-	2
INCF f,d	Увеличение значения регистра «f» на единицу (f = 0…255). Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f»	OV, C, DC, Z	1
INCFSZ f,d	Увеличение значения регистра «f» на единицу и пропуск следующей команды если результат равен нулю, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение занимает 2 цикла. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1(2)
INFSNZ f,d	Увеличение значения регистра «f» на единицу и пропуск следующей команды если результат не равен нулю, т.е. вместо следующей команды выполняется операция NOP, в этом случае выполнение занимает 2 цикла. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1(2)
IORLW k	Логическая операция включающего «ИЛИ» 8-ми битной константы «k» (k = 0…255) и содержимого регистра WREG. Результат помещается в регистр WREG	Z	1
IORWF f,d	Логическая операция включающего «ИЛИ» содержимого регистров WREG и «f». Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	Z	1
LCALL k	Вызов подпрограммы находящейся в любом месте памяти в диапазоне 64Кслов. Адрес следующей после LCALL команды (PC+1) помещается в стек. 16-ти битный адрес загружается в счетчик команд PC. Младшие 8 бит загружаются из кода команды, а старшие 8 бит из регистра PCLATCH. LCALL выполняется за два цикла	-	2

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
MOVFP f,p	Пересылка данных из области памяти «f» в область памяти «p». Адрес области «f» может быть от 00h до FFh, а области «p» от 00h до 1Fh. И «f» и «p» могут быть регистром WREG, а также косвенно адресованы	-	1
MOVLB k	Загрузка 4х битной константы «k» в младшие 4 бита регистра выбора банка (BSR). Старшие 4 бита регистра BSR не изменяются	-	1
MOVLR k	Загрузка 4-х битной константы «k» в старшие 4 бита Регистра выбора банка (BSR). Младшие 4 бита регистра BSR не изменяются	-	1
MOVLW k	8-ми битная константа «k» загружается в регистр WREG	-	1
MOVPF p,f	Пересылка данных из области памяти «p» в область памяти «f». Адрес области «f» может быть от 00h до FFh, а области «p» от 00h до 1Fh. И «f» и «p» могут быть регистром WREG, а также косвенно адресованы	Z	1
MOVWF f	Пересылка содержимого регистра WREG в регистр «f» (f = 0…255)	-	1
MULLW k	Беззнаковое перемножение 8-ми битной константы «k» и содержимого регистра WREG. 16-ти битный результат записывается в паре регистров PRODH:PRODL. Регистр WREG не изменяется. Операция не изменяет флаги	-	1
MULWF f	Беззнаковое перемножение содержимого регистров «f» (f = 0…255) и WREG. 16-ти битный результат записывается в паре регистров PRODH:PRODL. Регистры WREG и «f» не изменяются. Операция не изменяет флаги	-	1
NEGW f,s	Изменение знака содержимого регистра WREG путем двоичного дополнения. Если s=0, то результат помещается в «f» и в WREG, если s=1, то результат помещается в «f» (f = 0…255)	OV, C, DC, Z	1
NOP	Нет операции	-	1
RETFIE	Возврат из прерывания. Содержимое счетчика команд восстанавливается из стека. Разрешаются глобальные прерывания сбросом бита GLINTD(CPUSTA<4>). PCLATCH не изменяется. Команда выполняется за 2 цикла	GLINTD=0	2

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
RETLW k	Возврат из подпрограммы. В регистр WREG загружается значение 8-ми битной константы «k». В счетчик команд загружается из стека адрес возврата. PCLATCH не изменяется. Команда выполняется за два цикла	-	2
RETURN	Возврат из подпрограммы. В счетчик команд загружается из стека адрес возврата. PCLATCH не изменяется. Команда выполняется за два цикла	-	2
RLCF f,d	Операция циклического сдвига содержимого регистра «f» влево через флаг переноса «C». Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	C	1
RLNCF f,d	Операция циклического сдвига содержимого регистра «f» влево. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1
RRCF f,d	Операция циклического сдвига содержимого регистра «f» вправо через флаг переноса «C». Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	C	1
RRNCF f,d	Операция циклического сдвига содержимого регистра «f» вправо. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1
SETF f,s	Установка всех битов регистра «f» в единицы. Если s=0, то значение 0FFh помещается в «f» и в WREG, если s=1, то результат помещается только в «f» (f = 0…255)	-	1
SLEEP	Сбрасывает «сторожевой таймер» и его предделитель. Бит «TO» устанавливается, а «PO» сбрасывается. Процессор переходится в режим «сна» (SLEEP) с остановкой тактового генератора	TO=1, PD=0	1
SUBLW k	Содержимое регистра WREG вычитается из 8-ми битной константы «k». Результат помещается в регистр WREG	OV, C, DC, Z	1
SUBWF f,d	Вычитание содержимого регистра WREG из содержимого регистра «f». Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	OV, C, DC, Z	1

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
SUBWFB f,d	Вычитание содержимого регистра WREG и флага переноса (заема) из содержимого регистра «f» (метод двоичного дополнения). Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	OV, C, DC, Z	1
SWAPF f,d	Обмен местами полубайтов регистра «f». Верхняя половина регистра и нижняя меняются местами. Если d=0, то результат операции сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	-	1
TABLRD t,i,f	1. Содержимое байта «табличной защелки» (TBLAT) пересылается в регистр «f» (f = 0…255). Если t=1, то пересылается старший байт, если t=0, то младший байт. 2. Содержимое области памяти программ, указываемой 16-ти битным «табличным указателем» (TBLPTR) загружается в 16-ти битную «табличную защелку» (TBLAT). 3. Если i=1, то значение TBLPTR увеличивается на единицу, если i=0, то значение TBLPTR не изменяется. Команда выполняется 2 цикла, а если регистр «f» является регистром PCL, то 3 цикла	-	2(3)
TABLWT t,i,f	1. Загрузка содержимого регистра «f» (f = 0…255) в 16-ти битную «табличную защелку» (TBLAT). Если t=1, то загружается в старший байт, если t=0, то в младший байт. 2. Содержимое «табличной защелки» (TBLAT) записывается в область памяти программ, указываемой «табличным указателем» (TBLPTR). Если TBLPTR указывает на область внешней памяти программ, то команда выполнится за 2 цикла. Если TBLPTR указывает на внутреннюю область FLASH памяти, то выполнение команды происходит до прерывания (т.е. >2 циклов). 3. Если i=1, то значение TBLPTR увеличивается на единицу, если i=0, то значение TBLPTR не изменяется. Примечание: Для записи во внутреннюю память программ должно быть подано напряжение программирования. Если это условие не выполнено, то команда выполнится за 2 цикла и значение памяти не изменится	-	2(>2)

Мнемоника команды	Описание команды	Изменяемые флаги	Кол-во циклов
TLRD t,f	Считывание данных из 16-ти битной «табличной защелки» в регистр «f» (f = 0…255). «Табличная защелка» при этом не изменяется. Команда используется совместно с TABLRD для пересылки данных из памяти программ в память данных. Если t=1, то считывается старший байт, если t=0, то младший байт	-	1
TLWT t,f	Содержимое регистра «f» (f = 0…255) записывается в 16-ти битную «табличную защелку» (TBLAT). Команда используется совместно с командой TABLWT для пересылки данных из памяти данных в память программ. Если t=1, то записывается старший байт, если t=0, то младший байт	-	1
TSTFSZ f	Сравнение содержимого регистра «f» (f = 0…255) с нулем. Если (f) = 0, то вместо следующей команды выполняется операция NOP, в этом случае выполнение команды занимает 2 цикла	-	1(2)
XORLW k	Логическая операция исключающего «ИЛИ» 8-ми битной константы «k» (k = 0…255) и содержимого регистра WREG. Результат помещается в регистр WREG	Z	1
XORWF f,d	Логическая операция исключающего «ИЛИ» содержимого регистров WREG и «f». Если d=0, то результат сохраняется в регистре WREG, если d=1, то в регистре «f» (f = 0…255)	Z	1

Таблица 57

Коды команд

Мнемоника команды	Код команды	Мнемоника команды	Код команды	Мнемоника команды	Код команды
ADDLW k	1011 0001 kkkk kkkk	DCFSNZ f,d	0010 011d ffff ffff	RETURN	0000 0000 0000 0010
ADDWF f,d	0000 111d ffff ffff	GOTO k	110k kkkk kkkk kkkk	RLCF f,d	0001 101d ffff ffff
ADDWFC f,d	0001 000d ffff ffff	INCF f,d	0001 010d ffff ffff	RLNCF f,d	0010 001d ffff ffff
ANDLW k	1011 0101 kkkk kkkk	INCFSZ f,d	0001 111d ffff ffff	RRCF f,d	0001 100d ffff ffff
ANDWF f,d	0000 101d ffff ffff	INFSNZ f,d	0010 010d ffff ffff	RRNCF f,d	0010 000d ffff ffff
BCF f,b	1000 1bbb ffff ffff	IORLW k	1011 0011 kkkk kkkk	SETF f,s	0010 101s ffff ffff
BSF f,b	1000 0bbb ffff ffff	IORWF f,d	0000 100d ffff ffff	SLEEP	0000 0000 0000 0011

BTFSC f,b	1001 1bbb ffff ffff	LCALL k	1011 0111 kkkk kkkk	SUBLW k	1011 0010 kkkk kkkk
BTFSS f,b	1001 0bbb ffff ffff	MOVFP f,p	011p pppp ffff ffff	SUBWF f,d	0000 010d ffff ffff
BTG f,b	0011 1bbb ffff ffff	MOVLB k	1011 1000 uuuu kkkk	SUBWFB f,d	0000 001d ffff ffff
CALL k	111k kkkk kkkk kkkk	MOVLR k	1011 101x kkkk uuuu	SWAPF f,d	0001 110d ffff ffff
CLRF f,s	0010 100s ffff ffff	MOVLW k	1011 0000 kkkk kkkk	TABLRD t,i,f	1010 10ti ffff ffff
CLRWDT	0000 0000 0000 0100	MOVPF p,f	010p pppp ffff ffff	TABLWT t,i,f	1010 11ti ffff ffff
COMF f,d	0001 001d ffff ffff	MOVWF f	0000 0001 ffff ffff	TLRD t,f	1010 00tx ffff ffff
CPFSEQ f	0011 0001 ffff ffff	MULLW k	1011 1100 kkkk kkkk	TLWT t,f	1010 01tx ffff ffff
CPFSGT f	0011 0010 ffff ffff	MULWF f	0011 0100 ffff ffff	TSTFSZ f	0011 0011 ffff ffff
CPFSLT f	0011 0000 ffff ffff	NEGW f,s	0010 110s ffff ffff	XORLW k	1011 0100 kkkk kkkk
DAW f,s	0010 111s ffff ffff	NOP	0000 0000 0000 0000	XORWF f,d	0000 110d ffff ffff
DECF f,d	0000 011d ffff ffff	RETFIE	0000 0000 0000 0101		
DECFSZ f,d	0001 011d ffff ffff	RETLW k	1011 0110 kkkk kkkk		

Обозначения:

f - адрес регистра от 00h до FFh.

p - адрес периферийного регистра от 00h до 1Fh.

k - поле константы (данные или адрес).

b - адрес бита в 8-ми разрядном регистре.

d - выбор места назначения для размещения результата: если =0 - результат помещается в регистр WREG, если =1 - в указанный регистр.

s - выбор места назначения для размещения результата: если =0 - результат помещается в указанный регистр и регистр WREG, если =1 - только в указанный регистр.

i - управление «табличным указателем»: если =1 - значение указателя инкрементируется после выполнения операции, если =0 - не изменяется.

t - выбор байта в 16-ти разрядной «табличной защелке»: если =1 - старший байт, если =0 - младший байт.

x,u - не используются, имеют значение 0.

Предельно допустимые и предельные режимы работы

Таблица 58

Наименование параметра	Буквенное обозначение	Норма параметра				Единица измерения
		Предельно допустимый режим		Предельный режим		
		не менее	не более	не менее	не более	
Напряжение источника питания,	U_{CC}	4,5	5,5	-	6,5	В
Напряжение питания портов PC и PD	U_{CCP}	3,0	U_{CC}	-	6,5	В
Входное напряжение высокого уровня на выводах PA(17, 20), OSC1_CLKIN (47), USBCLK (6), MCLR_Upp (45)	U_{IH}	0,8•U_{CC}	U_{CC}	-	U_{CC}+0,3	В
на выводах PE (7…11, 14…16)		2,0	U_{CC}	-	U_{CC}+0,3	
на выводах DM, DP (3, 4)		2,0	3,6	-	U_{CC}+0,3	
на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42)		2,0	U_{CCP}	-	U_{CCP}+0,3	
Входное напряжение низкого уровня на выводах PA(17, 20), OSC1_CLKIN (47), USBCLK (6), MCLR_Upp (45)	U_{IL}	0	0,2•U_{CC}	минус 0,3	-	В
на выводах PE (7…11, 14…16)		0	0,8	минус 0,3	-	
на выводах DM, DP (3, 4)		0	1,1	минус 0,3	-	
на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42)		0	0,6	минус 0,3	-	
Выходной ток высокого уровня, на выводах PA (19, 20), PE (7…11, 14…16) PC, PD (21…23, 25, 28…31, 33…36, 39…42)	I_{OH}	-	6	-	10	мА
на выводе OSC2 (48)		-	1	-	2	
на выводах DM, DP (3, 4)		-	4	-	10	
Выходной ток низкого уровня, на выводах PA (19, 20),PE (7…11, 14…16) PC, PD (21…23, 25, 28…31, 33…36, 39…42)	I_{OL}	минус 6	-	минус 10	-	мА
на выводе OSC2 (48)		минус 1	-	минус 2	-	
на выводах DM, DP (3, 4)		минус 4	-	минус 10	-	

Наименование параметра	Буквенное обозначение	Норма параметра				Единица измерения
		Предельно допустимый режим		Предельный режим		
		не менее	не более	не менее	не более	
Напряжение программирования и стирания на входах: MCLR_Upp (45), TEST (46)	U_{PR}	11,75	12,25	-	-	В
Частота следования импульсов тактовых сигналов микроконтроллера (OSC1_CLKIN (47))	f_C	-	33	-	-	МГц
Частота следования импульсов тактовых сигналов контроллера USB (USBCLK(6))	f_{USB}	-	48	-	-	МГц
Длительность программирования одного слова	t_{CYW}	-	500	-	-	мкс
Длительность сигнала высокого уровня синхронизации на входах PA1/T0CLK (18) и OSC1_CLKIN (47),	$t_{WH(TOCLK)}$ $t_{WH(CLKIN)}$	15	-	-	-	нс
Длительность сигнала высокого уровня синхронизации на входе USBCLK (6)	$t_{WH(USBCLK)}$	10	-	-	-	нс
Длительность сигнала высокого уровня прерывания PA0/INT (17)	$t_{WH(INT)}$	40	-	-	-	нс
Длительность сигнала низкого уровня системного сброса, MCLR_Upp (45)	$t_{WL(MCLR)}$	100	-	-	-	нс
Параметры универсального последовательного синхронно-асинхронного приемопередатчика						
Время установления данных RX/DT (19) относительно синхронизации TX/CK (20)	$t_{SU(TX-RX)}$	17	-	-	-	нс
Время удержания данных RX/DT (19) относительно синхронизации TX/CK (20)	$t_{H(TX-RX)}$	17	-	-	-	нс
Параметры аналогового приемо-передатчика USB						
Длительность EOP при приеме USB на линиях DM (3) и DP(4) (full speed)	$t_{EOP\ F}$	140	180			нс
Длительность EOP при приеме USB на линиях DM(3) и DP(4) (low speed)	$t_{EOP\ L}$	1,17	1,50			мкс

Наименование параметра	Буквенное обозначение	Норма параметра				Единица измерения
		Предельно допустимый режим		Предельный режим		
		не менее	не более	не менее	не более	
Параметры синхронного последовательного приемопередатчика в режиме SPI						
Длительность сигнала высокого уровня на линии SCK (8)	$t_{WH(SCK)}$	40	-	-	-	нс
Длительность сигнала низкого уроня на линии SCK (8)	$t_{WL(SCK)}$	40	-	-	-	нс
Время установления данных SDI (9) относительно SCK (8)	$t_{SU(SCK-SDI)}$	100	-	-	-	нс
Время удержания данных SDI (9) относительно SCK (8)	$t_{H(SCK-SDI)}$	100	-	-	-	нс
Параметры Таймера 0						
Длительность сигнала высокого уровня на линии T0CLK (18), прескалер включен	$t_{WH(PA1)}$	15	-	-	-	нс
Длительность сигнала низкого уровня на линии T0CLK (18), прескалер включен	$t_{WL(PA1)}$	15	-	-	-	нс
Длительность фронта и спада входного сигнала на выводе OSC1_CLKIN (47), при: f_C=33 МГц	τ_r τ_f	-	5	-	-	нс
Длительность фронта и спада входного сигнала на выводе USBCLK (6), при: f_{USB}=48 МГц	τ_{r_USBCLK} τ_{f_USBCLK}	-	5	-	-	нс
Емкость нагрузки на выводах (7…11, 14…16, 19…23, 25, 28…31, 33…36, 39…42, 48)	C_L	-	50	-	-	пФ

Электрические параметры микросхемы

Таблица 58

Наименование параметра, режим измерения	Буквенное обозначение параметра	Норма параметра		единица измерения
		не менее	не более	
Выходное напряжение низкого уровня на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42) при: U_{CCP}= 3,0 В, U_{CC}= 4,5 В, I_{OL}= 6,0 мА	U_{OL}	-	0,4	В
на выводах PE (7…11, 14…16) при: I_{OL}= 6,0 мА		-	0,4	
на выводах PA (19, 20) при: I_{OL}= 6,0 мА		-	0,45	
на выводах DM, DP (3, 4) при: I_{OL}= 4,0 мА		-	0,3	
на выводе OSC2_CLKOUT (48) при: I_{OL}= 1,0 мА		-	0,45	
Выходное напряжение высокого уровня на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42) при: U_{CCP}= 3,0 В, U_{CC}= 4,5 В, I_{OH}= 6,0 мА	U_{OH}	2,4	-	В
на выводах PE (7…11, 14…16) при: I_{OH}= 6,0 мА		2,4	-	
на выводах PA (19, 20) при: I_{OH}= 6,0 мА		4,05	-	
на выводах DM, DP (3, 4) при: I_{OH}= 4,0 мА		2,8	-	
на выводе OSC2_ CLKOUT (48) при: I_{OH}= 1,0 мА		4,05	-	
Уровень напряжения для срабатывания схемы генерации сброса	U_{BOR}	3,6	4,3	В
Уровень выходного напряжения внутреннего регулятора напряжения при: U_{CC}= 4,5 В, I_{OH}= 40 мА	U_{33OUT}	3,0	3,6	В
Статический ток потребления в режиме покоя при: U_{CC}= U_{CCP}= 5,5 В, MCLR_Upp= 0 В	I_{CCS}	-	50	мкА
Динамический ток потребления при: U_{CC}= U_{CCP}= 5,5 В, f_C= 33 МГц, f_{USB}= 48 МГц	I_{OCC}	-	100	мА
Входной ток утечки высокого уровня, мкА, на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 5,5 В	I_{ILH}	минус 1,0*	1,0	мкА
на выводах PE (7…11, 14…16) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 5,5 В		минус 1,0*	1,0	
на выводах PA (17…20) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 5,5 В		минус 1,0*	1,0	
на выводах DM, DP (3, 4) при: U_{CC}= U_{CCP}= 5,5 В, U_I=3,6 В		минус 1,0*	1,0	
на выводах OSC1_CLKIN, USBCLK (47, 6) при: U_{CC}= U_{CCP}= 5,5 В U_I= 5,5 В		минус 1,0*	1,0	

Наименование параметра, режим измерения	Буквенное обозначение параметра	Норма параметра		единица измерения
		не менее	не более	
на выводах MCLR_Upp, TEST (45, 46) при: U_{CC}= U_{CCP}= 5,5 В U_I= 5,5 В		минус 1,0*	1,0	
Входной ток утечки высокого уровня, на выводах MCLR_Upp, TEST (45, 46) при: U_{CC}= U_{CCP}= 5,5 В U_I= 12 В	I_{ILH1}	минус 90,0*	90,0	мкА
Входной ток утечки низкого уровня, на выводах PC, PD (21…23, 25, 28…31, 33…36, 39…42) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 0 В	I_{ILL}	минус 1,0*	1,0	мкА
на выводах PE (7…11, 14…16) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 0 В		минус 1,0*	1,0	
на выводах PA (17…20) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 0 В		минус 1,0*	1,0	
на выводах DM, DP (3, 4) при: U_{CC}= U_{CCP}= 5,5 В, U_I= 0 В		минус 1,0*	1,0	
на выводах OSC1_CLKIN, USBCLK (47, 6) при: U_{CC}= U_{CCP}= 5,5 В U_I= 0 В		минус 1,0*	1,0	
на выводах MCLR_Upp, TEST (45, 46) при: U_{CC}= U_{CCP}= 5,5 В U_I= 0 В		минус 1,0*	1,0	
Период работы внутреннего RC - генератора сторожевого таймера при: U_{CC}= U_{CCP}= 4,5 В	T_{RC}	60	140	мкс
Параметры синхронного последовательного приемопередатчика в режиме SPI				
Время задержки данных SDO (7) относительно SCK (8) при: U_{CC}= 4,5 В	$t_{d(SCK-SDO)}$	-	50	нс
Параметры универсального последовательного синхронно-асинхронного приемопередатчика				
Время задержки данных RX/DT (19) от фронта TX/CK (20), при: U_{CC}= U_{CCP}= 4,5 В	$t_{d(TX-RX)}$	-	50	нс
Параметры аналогового приемо-передатчика USB				
Время нарастания и спада выходного сигнала, на выводах DM, DP (3, 4) (Full Speed) при: U_{CC}= U_{CCP}= 4,5 В, C_L=100 пФ	t_{r_FS} t_{f_FS}	-	40	нс
на выводах DM, DP (3, 4) (Low Speed) при: U_{CC}= U_{CCP}= 4,5 В, C_L=400 пФ	t_{r_LS} t_{f_LS}	-	300	

Примечание: *n - в названии вывода - обозначает инверсию.*

Электрические параметры микросхемы, контролируемые на общей пластине (бескорпусное исполнение)

Таблица 59

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С
		не менее	не более	
Выходное напряжение низкого уровня, В на выводах DM, DP при: I_{OL}=4,0 мА	U_{OL}	-	0,29	25
Выходное напряжение высокого уровня, В, на выводах DM, DP при: I_{OL}=4,0 мА	U_{OH}	-	2,88	25
Уровень напряжения внутреннего регулятора напряжения, В, при: I_{OH}=40 мА	$U_{33\ OUT}$	3,1	3,5	25
Статический ток потребления в режиме покоя, мкА, при: U_{CCP}=5,5 В, nMCLR=0 В	I_{CCS}	-	48,5	25
Функциональный контроль при: U_{CC} = 5,5 В, U_{CCP} = U_{CC}, f_C=1 МГц	-	-	-	25
Функциональный контроль при: U_{CC} = 5,0 В, U_{CCP} = U_{CC}, f_C=1 МГц	-	-	-	25

Типовые зависимости

Зависимость частоты генератора, в режиме RC, от внешних времязадающих элементов (R, C), при: U_{CC}=5 В, T= 25°C

C_{EXT}	R_{EXT}	f_C
22 пФ	10 кОм	2676 кГц
	100 кОм	278 кГц
100 пФ	3.3 кОм	2856 кГц
	5.1 кОм	2000 кГц
	10 кОм	1080 кГц
	100 кОм	116 кГц
300 пФ	3.3 кОм	1120 кГц
	5.1 кОм	756 кГц
	10 кОм	404 кГц
	100 кОм	27,6 кГц

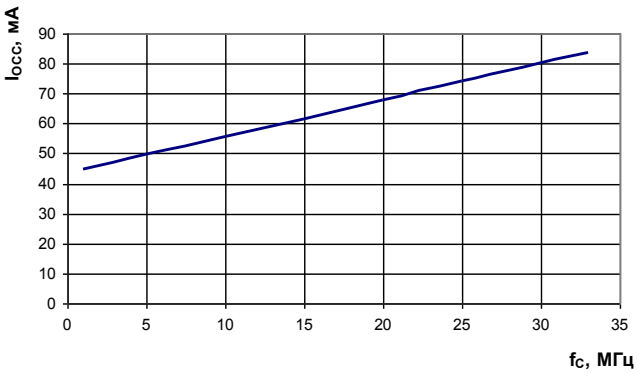


Рис. 63 Зависимость динамического тока потребления от частоты следования синхроимпульсов (f_C), при: T= 25°C, U_{CC} = 5,5 В, f_{USB}= 48 МГц

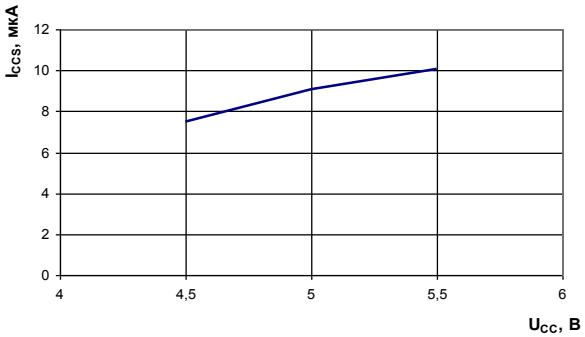


Рис. 64 Зависимость статического тока потребления от напряжения питания при: T= 25°C

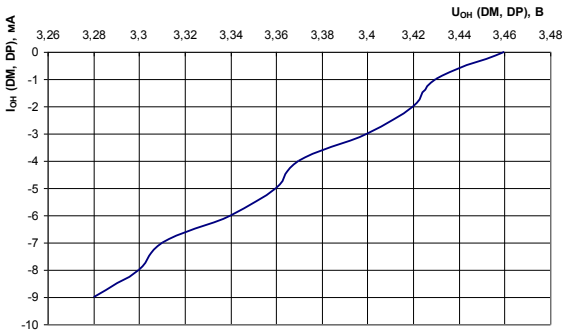


Рис. 65 Зависимость выходного напряжения высокого уровня на выводах DM, DP от тока нагрузки
при: T= 25°C, U_{CC} = 4,5 В

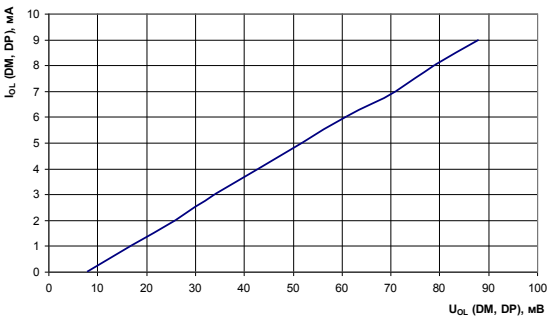


Рис. 66 Зависимость выходного напряжения низкого уровня на выводах DM, DP от тока нагрузки
при: T= 25°C, U_{CC} = 4,5 В

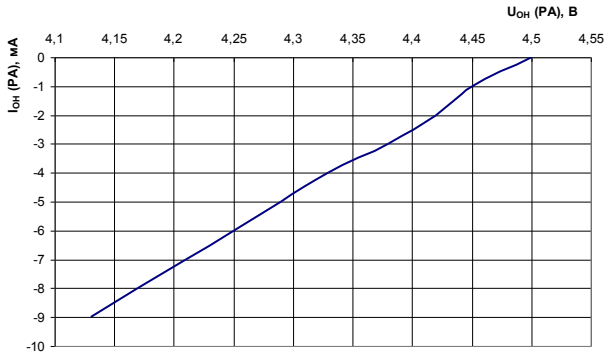


Рис. 67 Зависимость выходного напряжения высокого уровня на выводах PA от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В

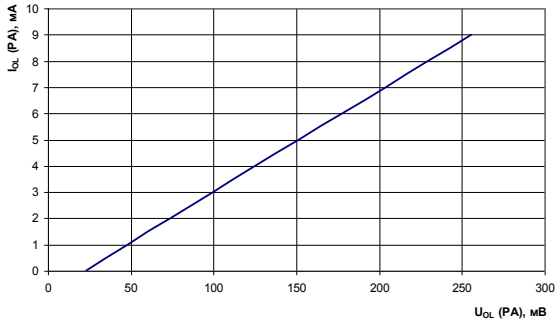


Рис. 68 Зависимость выходного напряжения низкого уровня на выводах PA от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В

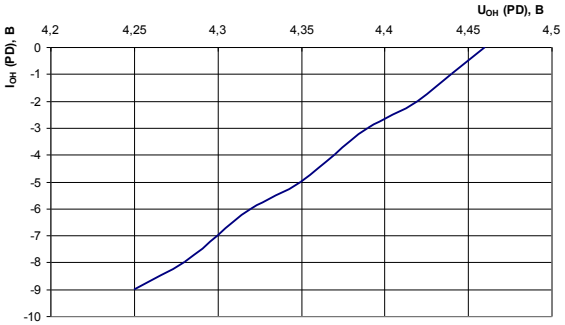


Рис. 69 Зависимость выходного напряжения высокого уровня на выводах PD от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 4,5 В

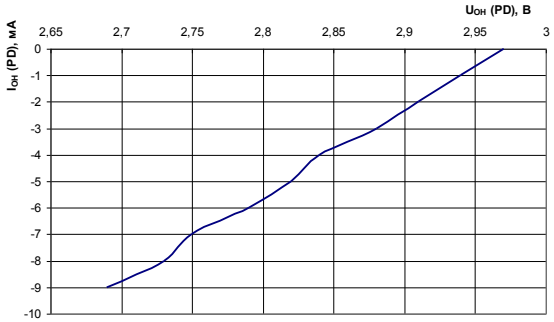


Рис. 70 Зависимость выходного напряжения высокого уровня на выводах PD от тока нагрузки
ри: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 3,0 В

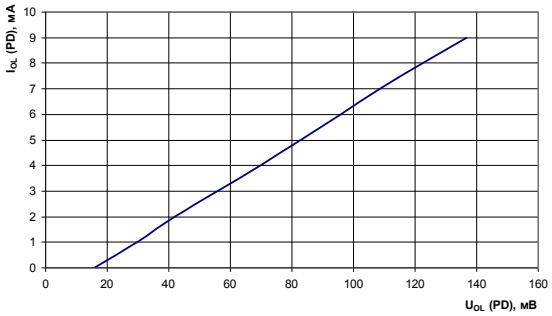


Рис. 71 Зависимость выходного напряжения низкого уровня на выводах PD от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 3,0 В

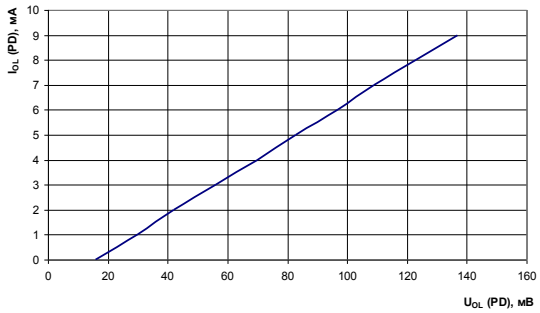


Рис. 72 Зависимость выходного напряжения низкого уровня на выводах PD от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 4,5 В

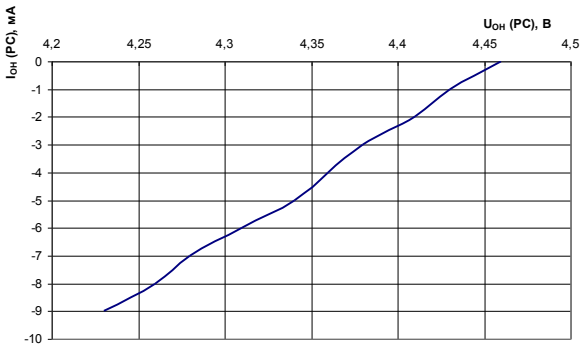


Рис. 73 Зависимость выходного напряжения высокого уровня на выводах PC от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 4,5 В

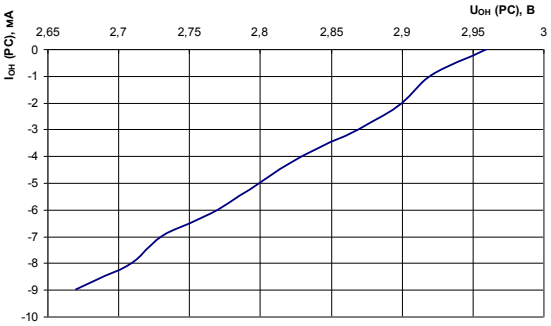


Рис. 74 Зависимость выходного напряжения высокого уровня на выводах PC от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 3,0 В

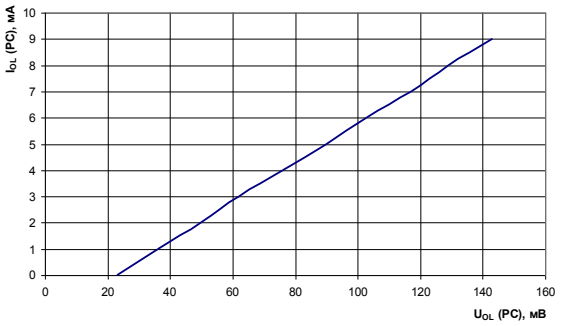


Рис. 75 Зависимость выходного напряжения низкого уровня на выводах PC от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 3,0 В

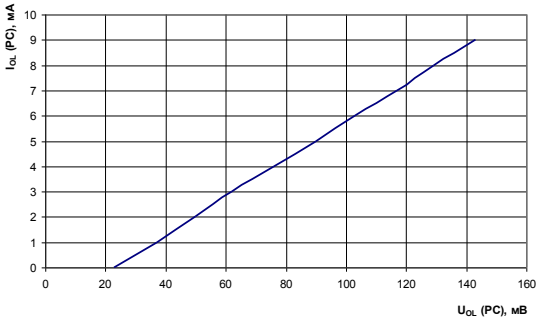


Рис. 76 Зависимость выходного напряжения низкого уровня на выводах PC от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В, U_{CCP}= 4,5 В

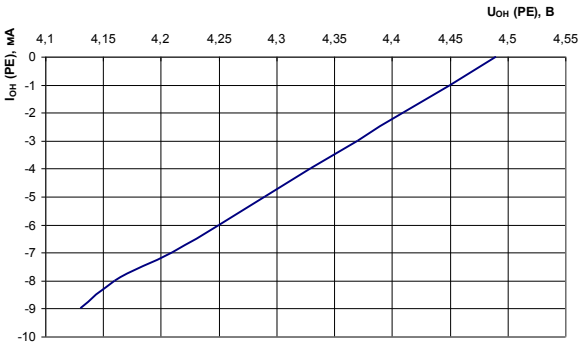


Рис. 77 Зависимость выходного напряжения высокого уровня на выводах PE от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В

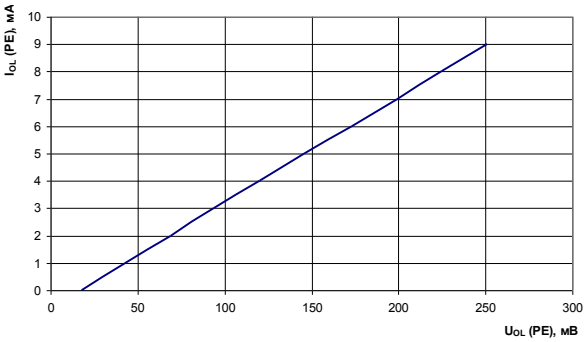


Рис. 78 Зависимость выходного напряжения низкого уровня на выводах PE от тока нагрузки
при: T= 25°C, U_{CC}= 4,5 В

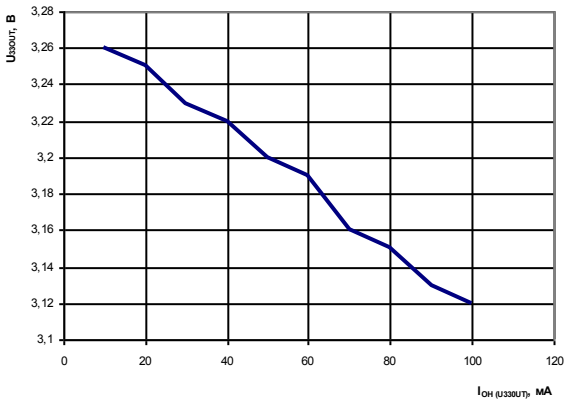


Рис. 79 Зависимость уровня внутреннего регулятора напряжения от тока нагрузки при: T= 25°C, U_{CC}= 4,5 В

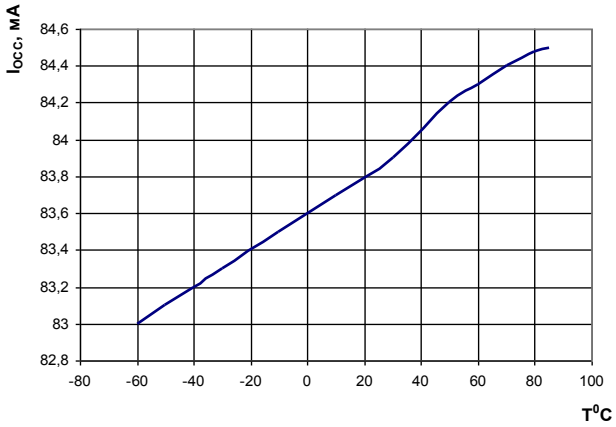


Рис. 80 Зависимость динамического тока потребления от температуры U_{CC}= 5,5 В, fc=33 МГц, , f_{USB}= 48 МГц

Габаритный чертеж микросхемы

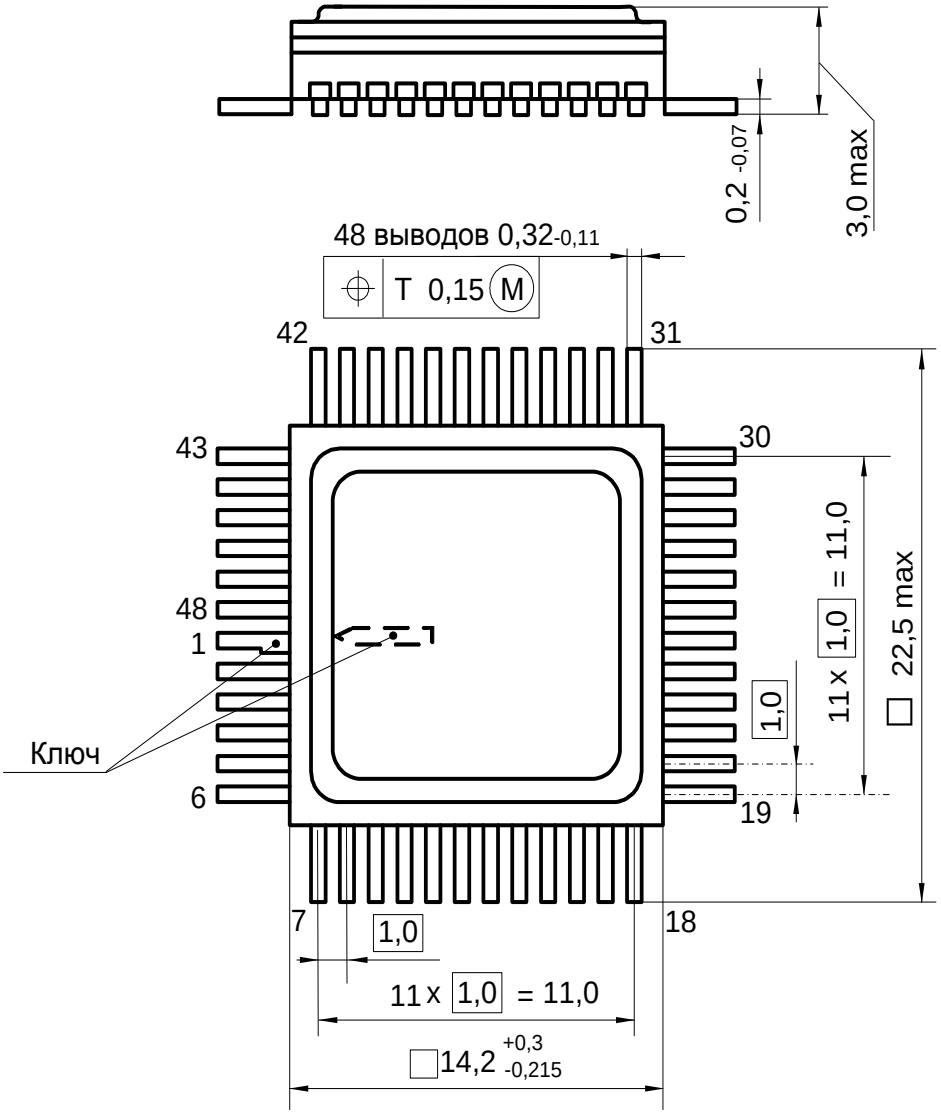


Рис. 81 Корпус Н16.48-1В

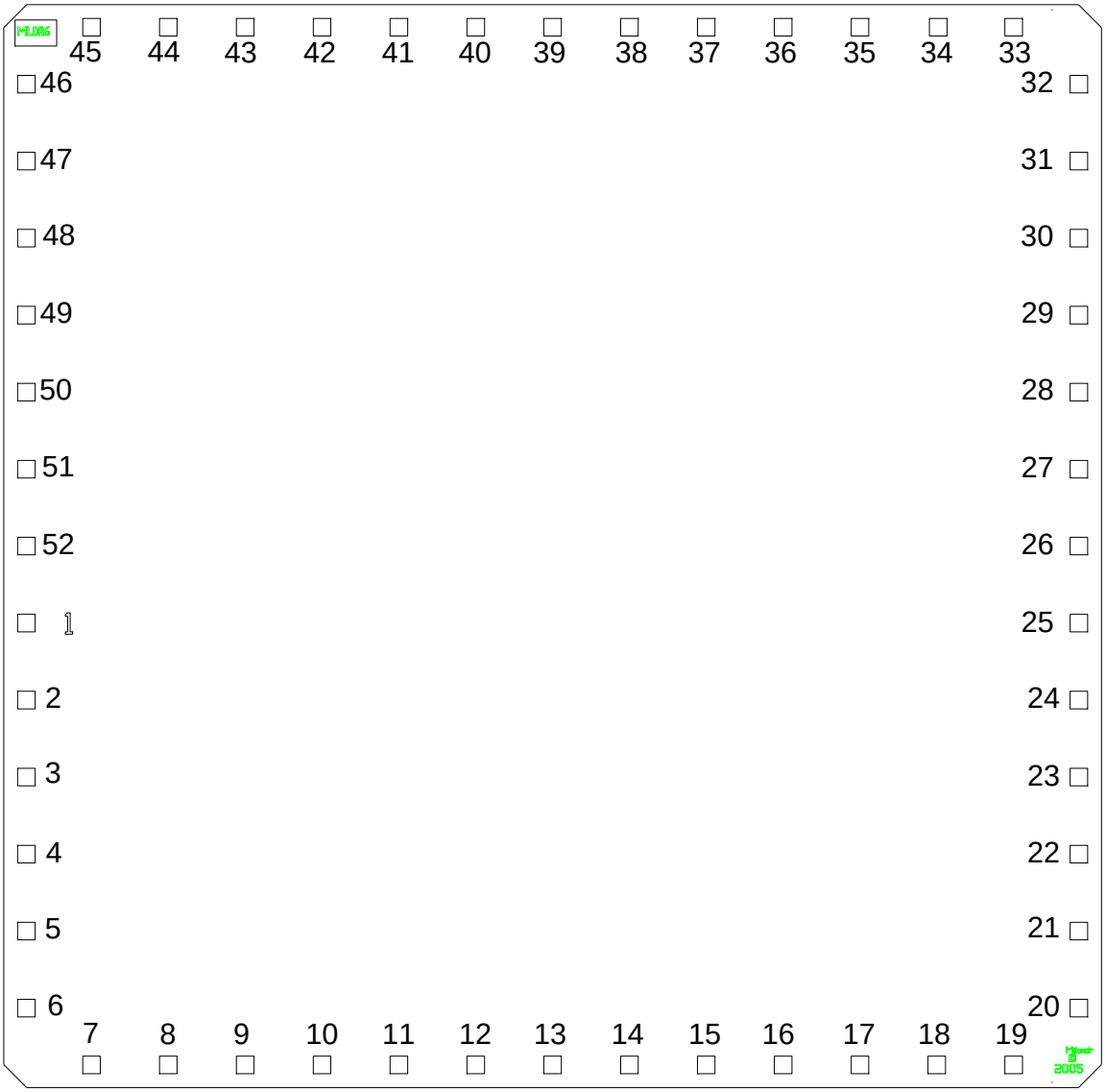


Рис. 82 Кристалл 6,25 max x 6,35 max (мм)

Примечание

Номера контактным площадкам (кроме первой) присвоены условно.

Информация для заказа

Обозначение	Маркировка	Тип корпуса	Температурный диапазон
1886ВЕ4У	1886ВЕ4У	Н16.48-1В	минус 60 – 85 °С
К1886ВЕ4У	К1886ВЕ4У	Н16.48-1В	минус 60 – 85 °С
К1886ВЕ4УК	К1886ВЕ4У•	Н16.48-1В	0 – 70 °С

Примечание:

Микросхемы в бескорпусном исполнении поставляются в виде отдельных кристаллов, получаемых разделением пластины. Микросхемы поставляются в таре (кейсах) без потери ориентации. Маркировка микросхемы – К1886ВЕ4Н4, наносится на тару.

Микросхемы с приемкой «ВП» маркируются ромбом.

Микросхемы с приемкой «ОТК» маркируются буквой «К».

Лист регистрации изменений

№ п/п	Дата	Версия	Краткое содержание изменения	№№ изменяе-мых листов
1	27.02.2010	2.2	1. Исправлен рисунок 10; 2. Исправлены названия регистров, битов; 3. Исправлены описания блока «Таймер 0», модуля SPI. 4. Введен лист регистрации изменений	4, 19-21, 23, 25, 26, 35-40, 59, 60, 62-68, 75, 81, 82, 91, 92
2	26.03.2010	2.3	1. Корректировка на основании планового пересмотра документации	1, 146
3	27.04.2010	2.4	Замена логотипа	1
4	04.04.2011	2.5	Уточнение описания работы микросхемы	22, 28, 60
5	07.10.2011	2.6	Уточнение наименования микросхем	По тексту
6	10.04.2012	2.7.0	Введена микросхема в бескорпусном исполнении	По тексту