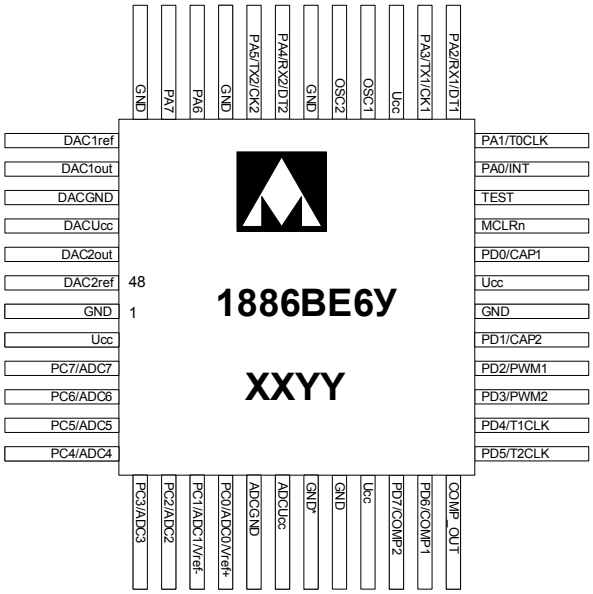
Высокопроизводительный универсальный 8-ми битный микроконтроллер с 12-ти разрядным АЦП, ЦАП и компаратором



XX – год выпуска
YY – неделя выпуска

Основные параметры микросхемы

- Тактовая частота до 32 МГц
- Минимальная длительность цикла выполнения команды 125 нс
- 58 однословных инструкций
- 8 х 8 битный аппаратный умножитель, за цикл
- Поддержка прямого, косвенного и относительного режимов адресации
- Аппаратная поддержка интерфейса LIN
- 8 канальное 12 разрядное АЦП
- 2 блока 12 разрядных ЦАП
- Диапазон напряжения питания 4,5 – 5,5 В
- Температурный диапазон:

Обозначение	Диапазон
1886ВЕ6У	минус 60 – 125 °С
К1886ВЕ6У	минус 60 – 125 °С
К1886ВЕ6АУ	0 – 70 °С
1886ВЕ6У1	минус 60 – 125 °С
К1886ВЕ6У1	минус 60 – 125 °С

Тип корпуса:

- 48-и выводной металлокерамический корпус Н16.48-1В

- 48-и выводной металлокерамический корпус 5142.48-А

Примечание
Микросхема К1886ВЕ6АУ является полным аналогом микросхемы К1886ВЕ6У с другим температурным диапазоном.

Особенности ядра микроконтроллера:

- 58 однословных инструкций
- Все инструкции выполняются за один цикл, за исключением инструкций переходов и инструкций чтения/записи таблиц выполняемых за два цикла
- Скорость работы: тактовая частота до 32 МГц, минимальная длительность цикла 114 нс.
- 8 x 8 битный аппаратный умножитель, за цикл
- Поддержка прерываний
- 16 словный аппаратный стек
- Прямая, косвенная и относительная модель адресации
- Внутренняя память программ типа EEPROM, размером 4Kx16

Особенности периферии:

- До 24 пользовательских выводов
- 16-ти битный таймер/счетчик с 8-ми битным предварительным делителем
- Таймер 1 и 2 (ШИМ/Захват/Таймеры)
- Два Универсальных Синхронных Асинхронных Приемника Передатчика (USART) с программируемой скоростью передачи и поддержкой режима LIN.
- Универсальный контроллер Внутренней памяти типа CMOS EEPROM размером 256x8.
- 8 канальное 12 разрядное АЦП последовательного приближения
- 2 блока 12 разрядных ЦАП
- Разрядность ШИМ на базе Таймер1 до 18 бит
- Компаратор

Специализированные особенности:

- Сброс по снижению питания
- Отложенный запуск по подаче питания и тактовой частоты
- Сторожевой таймер
- Защищенный режим
- Режим энергосбережения (SLEEP)
- Уровень напряжения питания микроконтроллера от 4,5 В до 5,5 В

Общее описание и области применения микросхемы

Микросхема предназначена для широкого применения в аппаратуре общего назначения, автомобильной технике, железнодорожном, водном и воздушном транспорте в качестве периферийного контроллера организующего сбор и первичную обработку информации.

Основные области применения:

- интеллектуальные датчики;
- автомобильная техника;
- промышленные системы управления;
- телекоммуникационное оборудование;
- системы безопасности;
- измерительное оборудование.

Содержание

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 12
Синхронизация выполнения команды 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 23
Прерывания 23
Регистр состояния прерываний (INTSTA) 24
Регистр состояния прерываний 25
Регистры разрешения периферийных прерываний PIE1 и PIE2 27
Регистры запроса периферийных прерываний PIR1 и PIR2 28
Обработка прерываний 31
Прерывание от вывода PA0/INT 32
Прерывание от вывода PA1/T0CLK 32
Периферийные прерывания 33
Сохранение регистров при прерывании 33
Организация памяти 37
Память программ 37
Память данных 39
Регистры общего назначения (GPR) 39
Регистры специального назначения (SFR) 39
Регистр статуса процессора (ALUSTA) 44
Регистр статуса ЦПУ (CPUSTA) 46
Регистр статуса управления TMR0 (T0STA) 47
Функционирование стека 48
Косвенная адресация 49
Регистры для чтения/записи таблиц 50
Модуль счетчика команд 51
Регистр выбора банка (BSR) 52
Считывание и запись таблиц данных 53
Запись таблиц во внутреннюю память 55
Чтение таблиц 56
Аппаратный умножитель 57
Порты ввода-вывода 60
Регистр порта A и регистр направления данных DDRA 60
Регистр порта C и регистр направления данных DDRC 61
Регистр порта D и регистр направления данных DDRD 62

Блок «таймер 0» ... 63
Таймер 1, таймер 2, ШИМ, захват (регистрация событий) ... 66
«Таймер 1» ... 69
Использование выходов широтно-импульсных модуляторов (ШИМ) ... 69
«Таймер 2» ... 72
Режим одного входа захвата и регистра периода для таймера ... 73
Режим двух входов захвата ... 74
Модуль универсального синхронно-асинхронных приемопередачика с поддержкой LIN интерфейса ... 76
Регистр режима и статуса работы приемника ... 77
Регистр режима и статуса работы передатчика ... 79
Регистр режима и статуса работы приемника LIN заголовка ... 80
Регистр данных приемника ... 81
Регистр данных передатчика ... 81
Регистр задания скорости приема и передачи ... 82
Регистр скорости поля SYNCH в LIN фрейме ... 82
Генератор скорости передачи данных ... 83
Асинхронный режим ... 83
Асинхронный передатчик ... 84
Асинхронный приемник ... 85
Режим автоматического приема LIN заголовка ... 87
Передача LIN фрейма ... 87
Синхронный ведущий режим ... 88
Передача данных в синхронном ведущем режиме ... 88
Прием данных в синхронном ведущем режиме ... 89
Синхронный ведомый режим ... 90
Передача данных в синхронном ведомом режиме ... 91
Прием данных в синхронном ведомом режиме ... 91
Аналогово-цифровой преобразователь ... 92
Блок Цифро-Аналогового Преобразователя ... 98
Регистр Управления и Состояния ЦАП ... 99
Блок Компаратор ... 100
Регистр Управления и Состояния Компаратора ... 100
Блок внутренней памяти данных EEPROM ... 101
Основные выполняемые функции и возможности ... 102
Регистр режима работы контроллера ... 102
Регистр контроля и тестирования ... 102
Регистр режима работы ... 103
Регистр данных ... 104
Регистр адреса обращения ... 105
Работа блока по стиранию, записи и чтению данных ... 105
Включение EEPROM ... 105
Очистка всей EEPROM ... 106
Запись всей EEPROM одним значением ... 106
Очистка строки EEPROM ... 107
Запись слова в EEPROM ... 107
Чтение слова из EEPROM ... 108
Описание блока управления EEPROM – памятью программ ... 109

Описание регистров ... 109
Регистр коэффициента деления частоты генератора ... 110
Регистр управления младшей половиной EEPROM памяти ... 110
Регистр управления старшей половиной EEPROM памяти ... 111
Регистр режима работы EEPROM памяти ... 112
Регистр старшего адреса EEPROM памяти ... 113
Регистр младшего адреса EEPROM памяти ... 114
Регистр конфигурационных бит ... 114
Регистр младшего байта данных EEPROM памяти ... 115
Регистр старшего байта данных EEPROM памяти ... 115
Функционирование и особенности работы с блоком ... 115
Чтение памяти ... 116
Конфигурационные биты ... 117
Специальные модули микроконтроллера ... 117
Регистры конфигурации микроконтроллера ... 117
Внутрисхемное программирование микроконтроллера ... 119
Сторожевой таймер ... 119
Режим энергосбережения (SLEEP) ... 120
Схема подключения напряжения питания ... 121
Система команд ... 122
Предельные и предельно-допустимые режимы работы ... 138
Электрические параметры микросхемы ... 140
Типовые зависимости ... 142
Габаритный чертеж микросхемы ... 143
Информация для заказа ... 144
Лист регистрации изменений ... 145

Описание выводов

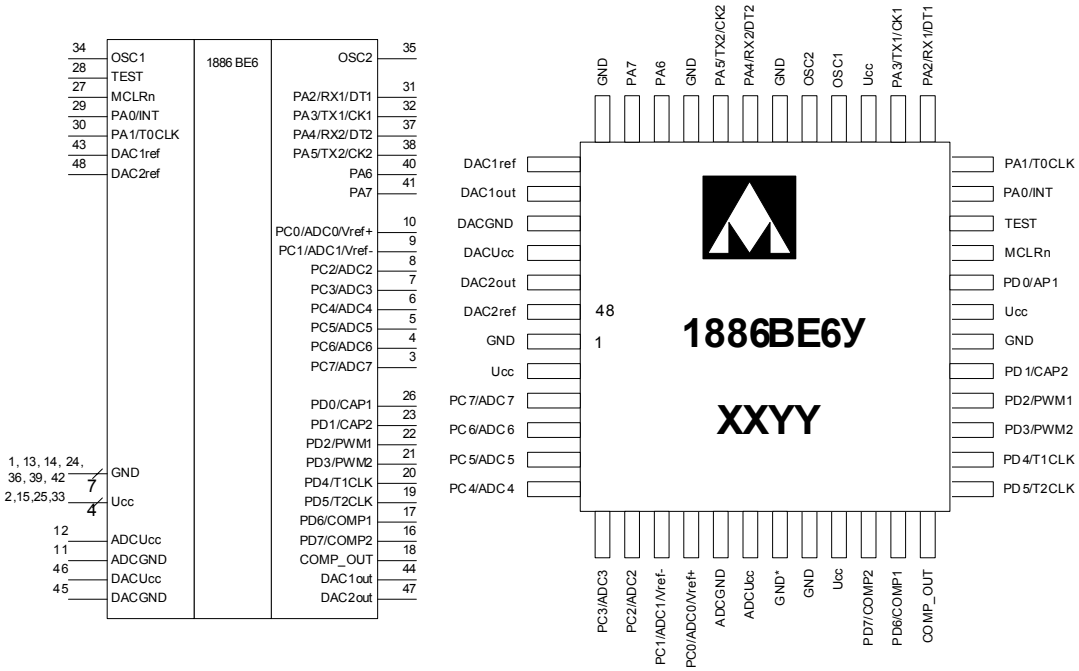


Рис. 1 Расположение выводов и условно-графическое обозначение микросхемы вкорпусе Н16.48-1В

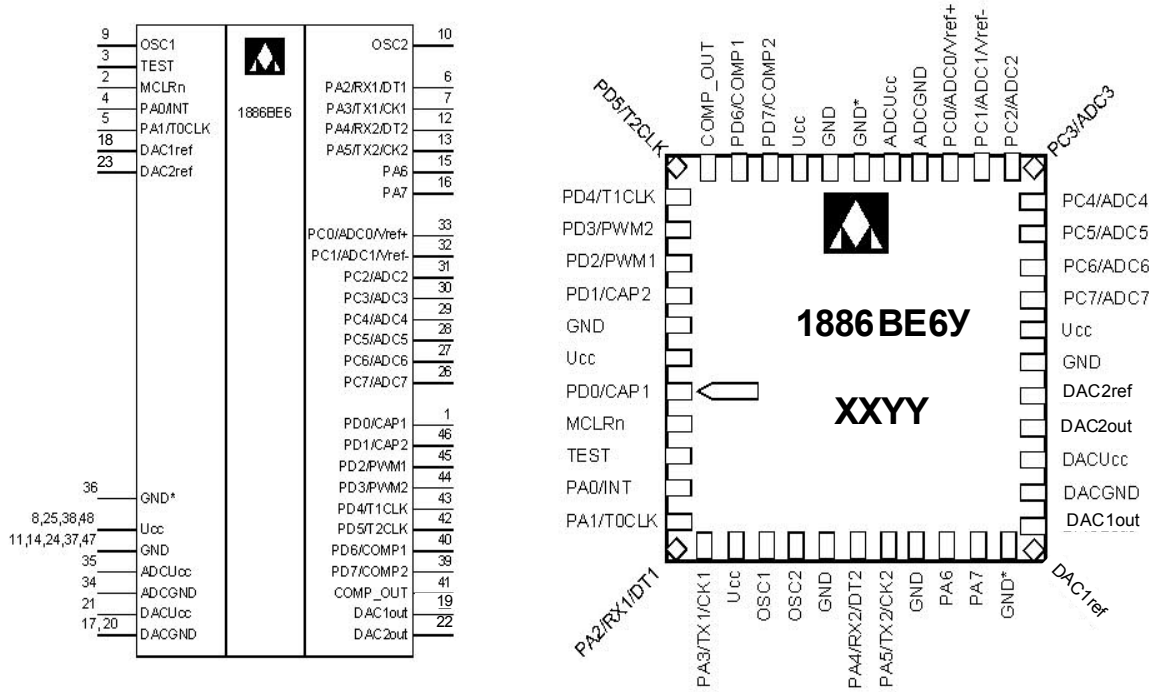


Рис. 2 Расположение выводов и условно-графическое обозначение микросхемы в корпусе 5142.48-А (вид сверху)

Таблица 1

Номер вывода		**Обозначение вывода**	**Тип вывода**	**Назначение вывода**
Корпус H16.48-1B	**Корпус 5142.48-A**			
34	9	OSC1	вход	Тактовая частота синхронизации / Вход внутреннего резонатора
35	10	OSC2	выход	Выход внутреннего резонатора
28	3	TEST	вход	Выбор режима работы
27	2	MCLRn	вход	Внешнего сигнала сброса
	Порт A			*Дополнительное назначение выводов:*
29	4	PA0/INT	вход	Пользовательский вывод / Внешнее прерывание
30	5	PA1/T0CLK	вход	Пользовательский вывод / Вход тактовой частоты Timer0
31	6	PA2/RX1/DT1	вход/ выход	Пользовательский вывод / Вывод RX интерфейса USART1
32	7	PA3/TX1/CK1	вход/ выход	Пользовательский вывод / Вывод TX интерфейса USART1
37	12	PA4/RX2/DT2	вход/ выход	Пользовательский вывод/ Вывод RX интерфейса USART2
38	13	PA5/TX2/CK2	вход/ выход	Пользовательский вывод/ Вывод TX интерфейса USART2
40	15	PA6	вход/ выход	Пользовательский вывод
41	16	PA7	вход/ выход	Пользовательский вывод
	Порт C			*Дополнительное назначение выводов:*
10	33	PC0/ADC0/ Uref+	вход/ выход	Пользовательский вывод / Канал АЦП 0/Верхнее опорное напряжение АЦП
9	32	PC1/ADC1/ Uref-	вход/ выход	Пользовательский вывод / Канал АЦП 1/Нижнее опорное напряжение АЦП
8	31	PC2/ADC2	вход/ выход	Пользовательский вывод / Канал АЦП 2
7	30	PC3/ADC3	вход/ выход	Пользовательский вывод / Канал АЦП 3
6	29	PC4/ADC4	вход/ выход	Пользовательский вывод / Канал АЦП 4
5	28	PC5/ADC5	вход/ выход	Пользовательский вывод / Канал АЦП 5
4	27	PC6/ADC6	вход/ выход	Пользовательский вывод / Канал АЦП 6

3	26	PC7/ADC7	вход/ выход	Пользовательский вывод / Канал АЦП 7
26	1	PD0/CAP1	вход/ выход	Пользовательский вывод / Вход схемы захвата 1
	Порт D			*Дополнительное назначение выводов:*
23	46	PD1/CAP2	вход/ выход	Пользовательский вывод / Вход схемы захвата 2
22	45	PD2/PWM1	вход/ выход	Пользовательский вывод / Выход схемы ШИМ 1
21	44	PD3/PWM2	вход/ выход	Пользовательский вывод / Выход схемы ШИМ 2
20	43	PD4/T1CLK	вход/ выход	Пользовательский вывод / Вход тактовой частоты Timer1
19	42	PD5/T2CLK	вход/ выход	Пользовательский вывод / Вход тактовой частоты Timer2
17	40	PD6/COMP1	вход/ выход	Пользовательский вывод / Вход 1 Компаратора
16	39	PD7/COMP2	вход/ выход	Пользовательский вывод / Вход 2 Компаратора
18	41	COMP_OUT	выход	Выход результата компаратора
44	19	DAC1out	выход	Выход DAC1
47	22	DAC2out	выход	Выход DAC2
43	18	DAC1ref	вход	Вход опорного напряжения DAC1
48	23	DAC2ref	вход	Вход опорного напряжения DAC2
2, 15, 25, 33	8, 25, 38, 48	Ucc	напряже-ние питания	Питание (4)
1, 13, 14, 24, 36, 39, 42	11, 14, 17, 24, 36, 37, 47	GND	общий	Общий (7)
12	35	ADCUcc	напряже-ние питания	Питание АЦП
11	34	ADCGND	общий	Общий АЦП
46	21	DACUcc	напряже-ние питания	Питание АЦП
45	20	DACGND	общий	Общий АЦП

Структурная блок-схема микросхемы

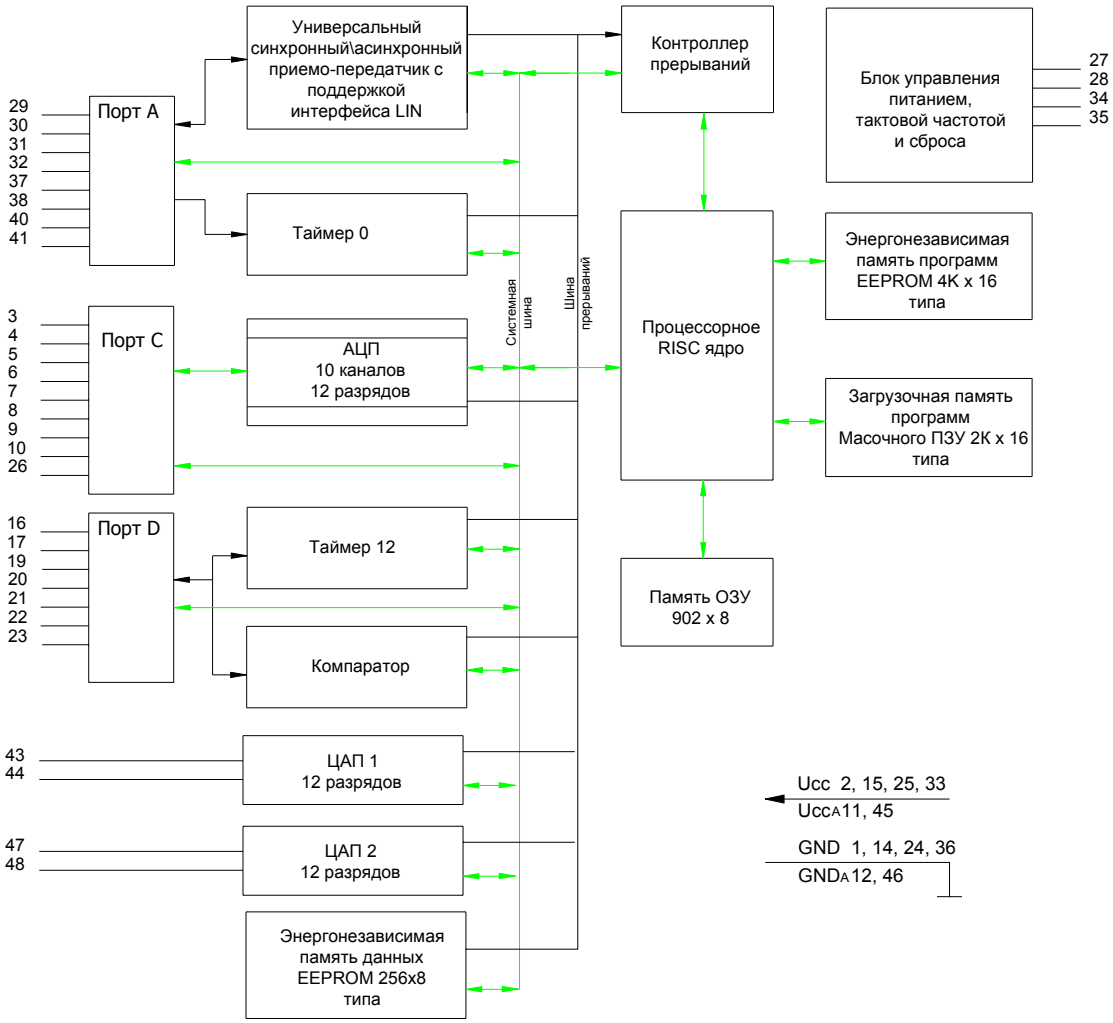


Рис. 3 Структурная блок-схема

Описание функционирования микросхемы

Встроенный тактовый генератор

Тактовая частота для микроконтроллера периферии может варьироваться в пределах до 32 МГц. И может подаваться как с 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 –генератор с внешним кварцевым или керамическим резонатором (частота до 2МГц);
XT – генератор с внешним кварцевым или керамическим резонатором (частота от 2МГц до 35 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 MCLRn при нормальной работе микроконтроллера тактовый генератор не выключается.

Использование кварцевого или керамического резонатора

В режиме тактового генератора XT и LF, кварцевый или керамический резонатор подсоединяется к выводам OSC1 и OSC2. 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. 4** 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 VDD) пока временная диаграмма тактового сигнала центрируется к VDD/2.

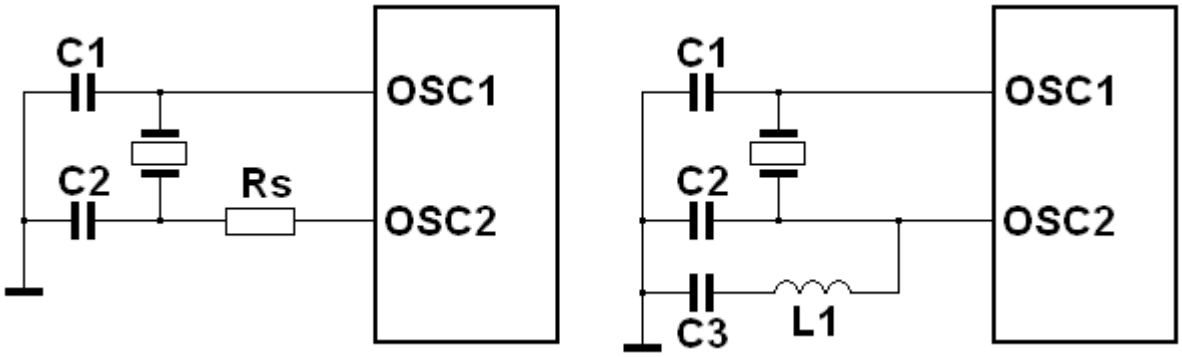


Рис. 4 Подключение резонатора.
Схема справа - для резонатора, работающего на гармониках

Примечания:

- Резистор Rs может потребоваться для некоторых типов резонаторов.
- Параллельный резонансный контур L1C2 отфильтровывает основную частоту: (2πf)2 = 1/(L1*C2).
- C3 (0.1 мкф) препятствует протеканию постоянного тока на землю.
- Для резонаторов необходимы внешние конденсаторы C1 и C2 (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 имеет высокое входное сопротивление, а вывод OSC2 является выходом CLKOUT (FOSC/4). В качестве генераторов могут быть использованы готовые модули генераторов, обеспечивающие широкий набор тактовых частот и стабильные параметры.

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. 5** показана схема подключения RC цепочки к микроконтроллеру. Для сопротивления резистора меньше 2.2 кОм частота тактового генератора может быть нестабильна 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 формируется тактовый сигнал с частотой FOSC/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.

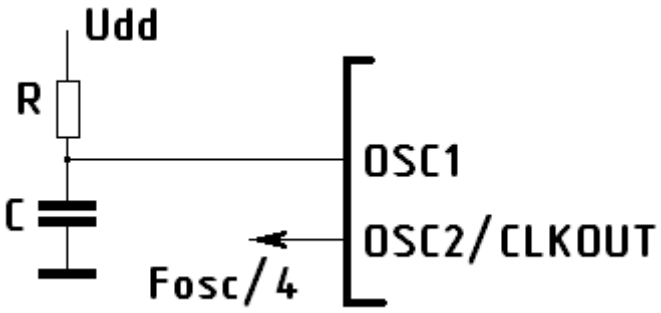


Рис. 5 Схема включения в режиме RC генератора

Синхронизация выполнения команды

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. 6**.

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. 7**). 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.

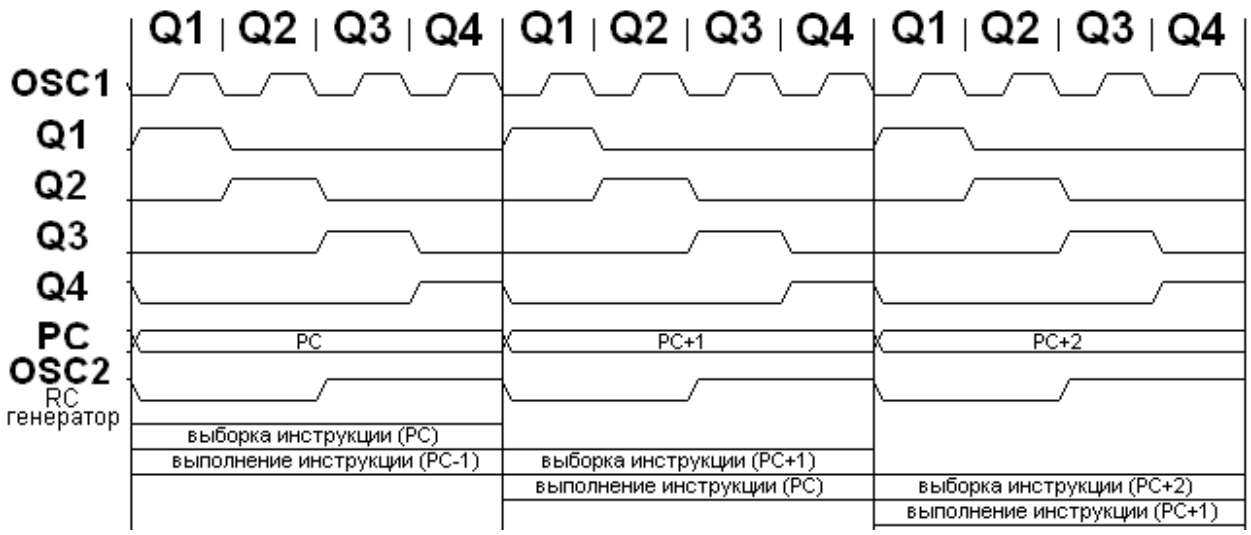


Рис. 6 Синхронизация выполнения команды



Рис. 7 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 **Таблице 3**. Эти биты в соединении с битами POR и BOR, используются в программном обеспечении для определения вида сброса. В **Таблице 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.

Упрощенная блок-схема схемы сброса показана на **Рис. 8**.

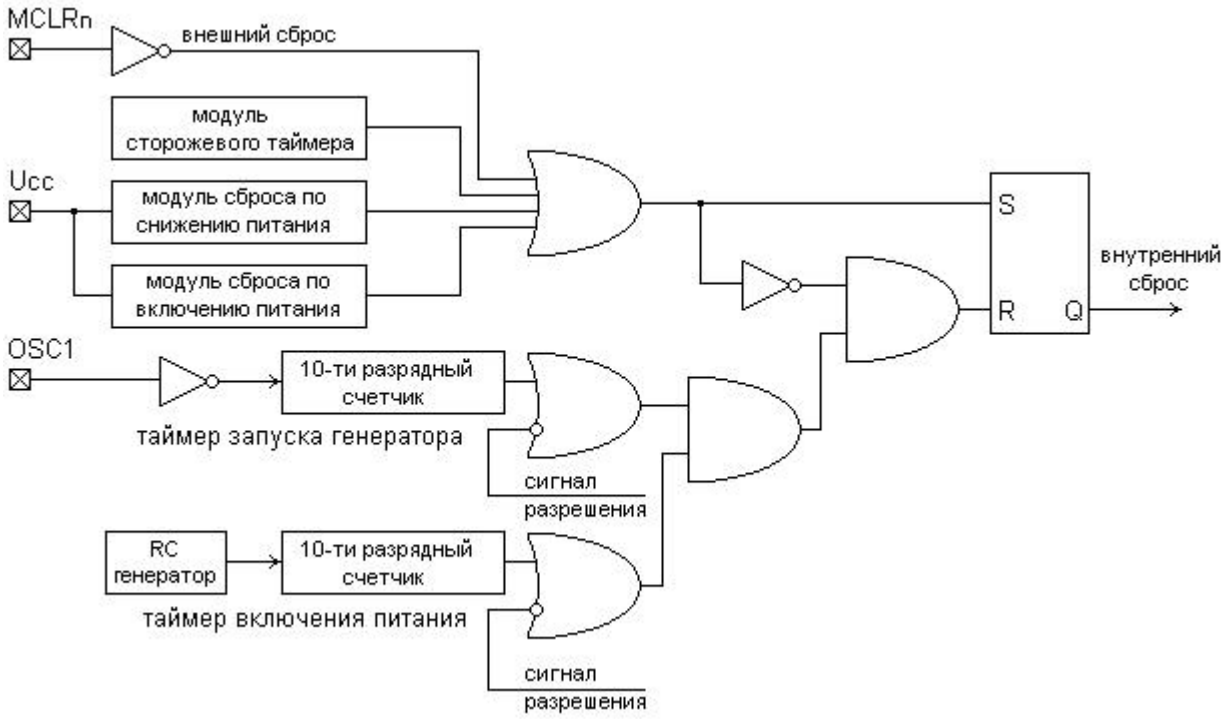


Рис. 8 Упрощенная блок-схема встроенной схемы сброса.

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 Вольта). Благодаря этой схеме, в ряде приложений можно обойтись без внешней RC цепочки, подключаемой к выводу MCLRn. В этом случае вывод MCLRn подключается через резистор или напрямую к напряжению питания. Внешняя схема «сброса» (**Рис. 9**) потребуется только в случае низкой скорости нарастания напряжения питания.

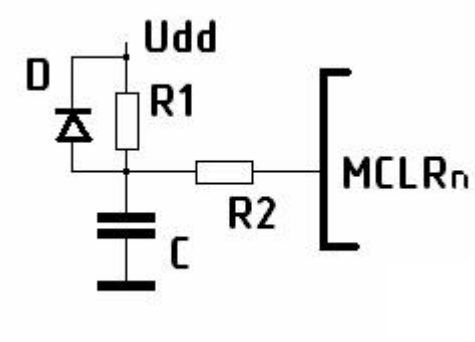


Рис. 9 Внешняя схема «сброса» по включению питания.

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 MCLRn, не превышало 0.2 В). Резистор R2 предназначен для ограничения тока через вывод MCLR, 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 MCLRn. 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 **Рис. 10** 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 рисунке: TOSC1 - время, требуемое кварцевому генератору для запуска.

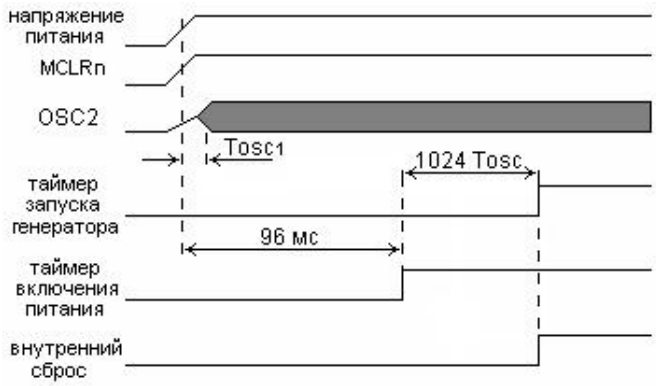


Рис. 10 Время запуска генератора.

Последовательность удержания микроконтроллера в состоянии сброса

При включении питания выполняется следующая последовательность удержания микроконтроллера в состоянии сброса: во-первых, внутренний сигнал «сброса» по включению питания увеличивается, пока не достигнет порогового уровня. Если сигнал MCLRn находится в высоком уровне, тогда начинает работать таймер включения питания, когда он отсчитает нужное время, начинает работать таймер запуска генератора, если MCLRn 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. 11** отображены последовательности задержек.

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

Если сигнал сброса MCLRn 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).

В **таблицах 3 и 4** показаны состояния после сброса для некоторых битов и специальных регистров, в то время как в **Таблица 4** 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 MCLRn
XT, LF	сумма 96 мс и 1024*T_C	1024*T_C	1 мс
EC, RC	сумма 96 мс и 1024*T_C	-	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 MCLRn (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.

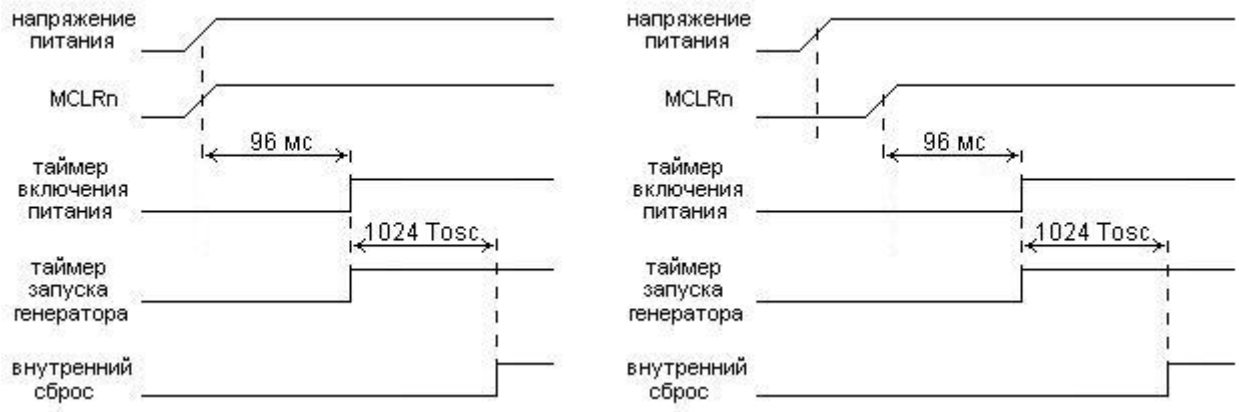
Таблица 4

Условия сброса программного счетчика и регистра CPUSTA

Тип «сброса»		PCH:PCL	CPUSTA(3)	Задержка включения
Сброс по включению питания		0000h	-011 1100	Сумма 96 мс и 1024*TC
Сброс по снижению напряжения питания		0000h	-011 1110	Сумма 96 мс и 1024*TC
Сброс от MCLRn в режиме нормальной работы		0000h	-011 1111(4)	1 мс
Сброс от MCLRn в режиме SLEEP		0000h	-011 1111(4)	Большее из 1 мс или (только для режимов XT и LF) 1024*TC
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*TC
Выход из режима SLEEP по прерыванию	GLINTD установлен	PC + 1	--11 1011(4)	Для режимов XT и LF: 1024*TC
	GLINTD сброшен	PC + 1(1)	--10 1011(4)	Для режимов XT и LF: 1024*TC

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.



а) **б)**

Рис. 11
Последовательность удержания в режиме сброса по включению питания:
а) MCLRn подсоединен к напряжению питания
б) MCLRn не подключен к напряжению питания

Таблица 5

Значения регистров при «сбросе»

Регистр	Адрес	Сброс по включению или снижению напряжения питания	Сброс от MCLRn или от сторожевого таймера	Выход из режима SLEEP по прерыванию
Вне банка				
INDF0	00h	N/A	N/A	N/A
FSR0	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h	0000h	0000h	PC+1(2)
PCLATH	03h	0000 0000	uuuu uuuu	uuuu uuuu
ALUSTA	04h	1111 xxxx	1111 uuuu	1111 uuuu
TOSTA	05h	0000 000-	0000 000-	0000 000-
CPUSTA(3)	06h	-011 11qq	-011 qquu	-uuu qquu
INSTA	07h	0000 0000	0000 0000	uuuu uuuu(1)
INDF1	08h	N/A	N/A	N/A
FSR1	09h	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	0Ah	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR0L	0Bh	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR0H	0Ch	xxxx xxxx	uuuu uuuu	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				
-	10h			
LIN1 CNTR	11h			
LIN1 BRG	12h			
RCSTA1	13h	0000 –000	0000 –000	uuuu -uuu
RCREG1	14h	0000 0000	0000 0000	uuuu uuuu
TXSTA1	15h	0000 --10	0000 --10	uuuu --uu
TXREG1	16h	0000 0000	0000 0000	uuuu uuuu
SPBRG1	17h	0000 0000	0000 0000	uuuu uuuu
Банк 1				
DDRA	10h	1111 11--	1111 11--	uuuu uu--
PORTA	11h	0-xx 11xx	0-uu 11uu	u-uu uuuu
DDRC	12h	1111 1111	1111 1111	uuuu uuuu
PORTC	13h	xxxx xxxx	uuuu uuuu	uuuu uuuu
DDRD	14h	1111 1111	1111 1111	uuuu uuuu
PORTD	15h	xxxx xxxx	uuuu uuuu	uuuu uuuu
-	16h			
-	17h			
Банк 2				
TMR1	10h	0000 0000	0000 0000	
TMR1H	11h	0000 0000	0000 0000	
TMR2	12h	0000 0000	0000 0000	
TMR2H	13h	0000 0000	0000 0000	
PR1	14h	1111 1111	1111 1111	
PR1H	15h	0000 0000	0000 0000	

Регистр	Адрес	Сброс по включению или снижению напряжения питания	Сброс от MCLRn или от сторожевого таймера	Выход из режима SLEEP по прерыванию
PR2/CA1L	16h	0000 0000	0000 0000	uuuu uuuu
PR2H/CA1H	17h	0000 0000	0000 0000	uuuu uuuu
Банк 3				
PW1DCL	10h	00-- ----	00-- ----	uu-- ----
PW2DCL	11h	000- ----	000- ----	uuu- ----
PW1DCH	12h			
PW2DCH	13h			
CA2L	14h	0000 0000	0000 0000	uuuu uuuu
CA2H	15h	0000 0000	0000 0000	uuuu uuuu
TCON1	16h	0000 0000	0000 0000	uuuu uuuu
TCON2	17h	0000 0000	0000 0000	uuuu uuuu
Банк 4				
PW1DCHH	10h	0000 0000	0000 0000	uuuu uuuu
PW2DCHH	11h	0000 0000	0000 0000	uuuu uuuu
DAC_CONT	12h	0000 0000	0000 0000	
DAC1L	13h	0000 0000	0000 0000	
DAC1H	14h	0000 0000	0000 0000	
DAC2L	15h	0000 0000	0000 0000	
DAC2H	16h	0000 0000	0000 0000	
COMPARE	17h	0000 0000	0000 0000	
Банк 5				
PIR1	10h	0000 0000	0000 0000	uuuu uuuu
PIE1	11h	0000 0000	0000 0000	
PIR2	12h	---- -000	---- -000	uuuu uuuu
PIE2	13h	---- -000	---- -000	
EE_CON	14h	0000 0000	0000 0000	
EE_MODE	15h	0-00 -000	0-00 -000	
EE_DATA	16h	0000 0000	0000 0000	
EE_ADR	17h	0000 0000	0000 0000	
Банк 6				
DBH	10h	0000 0000	0000 0000	uuuu uuuu
DBL	11h	0000 0000	0000 0000	uuuu uuuu
-	12h	xxxx xxxx	xxxx xxxx	xxxx xxxx
EEDIV	13h	0000 0101	0000 0101	uuuu uuuu
ADCON0	14h	0000 –0-0	0000 –0-0	uuuu uuuu
ADCON1	15h	0000- 0000	000- 0000	uuuu uuuu
ADRESL	16h	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESH	17h	xxxx xxxx	uuuu uuuu	uuuu uuuu
Банк 7				
-	10h			
LIN2 CNTR	11h			
LIN2 BRG	12h			
RCSTA2	13h	0000 –000	0000 –000	uuuu -uuu

Регистр	Адрес	Сброс по включению или снижению напряжения питания	Сброс от MCLRn или от сторожевого таймера	Выход из режима SLEEP по прерыванию
RCREG2	14h	0000 0000	0000 0000	uuuu uuuu
TXSTA2	15h	0000 --10	0000 --10	uuuu --uu
TXREG2	16h	0000 0000	0000 0000	uuuu uuuu
SPBRG2	17h	0000 0000	0000 0000	uuuu uuuu
Банк 14				
	10h	xxxx xxxx	xxxx xxxx	xxxx xxxx
EECONL	11h	0000 0000	0000 0000	
EECONH	12h	0000 0000	0000 0000	
	13h	xxxx xxxx	xxxx xxxx	xxxx xxxx
	14h	xxxx xxxx	xxxx xxxx	xxxx xxxx
	15h	xxxx xxxx	xxxx xxxx	xxxx xxxx
	16h	xxxx xxxx	xxxx xxxx	xxxx xxxx
	17h	xxxx xxxx	xxxx xxxx	xxxx xxxx
Банк15				
-	10h	xxxx xxxx	xxxx xxxx	xxxx xxxx
-	11h	xxxx xxxx	xxxx xxxx	xxxx xxxx
EDLSB	12h	0000 0000	0000 0000	
EDMSB	13h	0000 0000	0000 0000	
EEMOD	14h	0000 0000	0000 0000	
EAMSB	15h	0--- 0000	0--- 0000	
EALSB	16h	0000 0000	0000 0000	
CFREG	17h	-111 1111	-111 1111	
Вне банка				
PRODL	18h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODH	19h	xxxx xxxx	uuuu uuuu	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, PIR2 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.
5. При любом типе сброса устройства эти выводы конфигурируются как входы.

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 схема сброса: если напряжение питания опускается ниже U_{bor} (типовое значение 4.0 В, произойдет сброс по снижению напряжения питания. Если напряжение питания падает ниже, менее чем на этот период времени, то сброс не произойдет. Микроконтроллер находится в состоянии сброса, пока напряжение питания не поднимется выше U_{bor}. Затем включаются таймер включения питания и таймер запуска генератора (для режимов LP и XT). Это удерживает микроконтроллер в состоянии сброса более 96 мс и 1024*T_{OSC}. 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. 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.

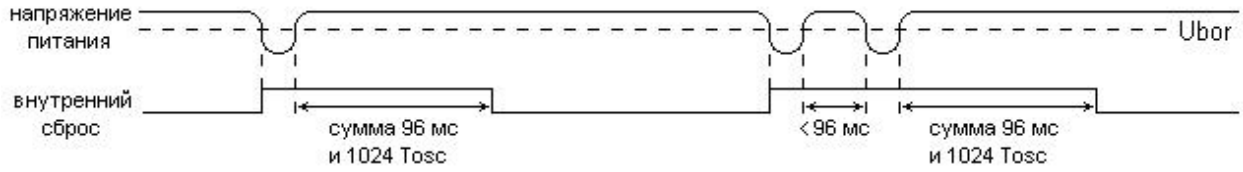


Рис. 12 Сброс по снижению напряжения питания

Прерывания

Микроконтроллеры имеют 15 источников прерывания:

- Внешнее прерывание от вывода PA0/INT;
- Переполнение таймера TMR0;
- Переполнение таймера TMR1;
- Переполнение таймера TMR2;
- Буфер передатчика USART1 пуст;
- Буфер приемника USART1 заполнен;
- Буфер передатчика USART2 пуст;
- Буфер приемника USART2 заполнен;
- Прерывания от модуля COMPARE;
- Прерывания от модуля EERPOM;
- Прерывания от модуля АЦП;
- Изменение сигнала на выводе PA1/T0CLK;

- Прерывание от схемы захвата 1;
- Прерывание от схемы захвата 2;
- Прерывание отладчика.

Прерываниями управляют шесть регистров: CPUSTA, INTSTA, PIE1, PIR1, PIE2, PIR2.

Регистр CPUSTA содержит бит глобального запрещения прерываний GLINTD (Global Interrupt Disable). Если этот бит установлен, то все прерывания запрещены. Этот бит является частью функциональных возможностей ядра микроконтроллера.

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.

Флаг запроса прерывания должен быть сброшен в программе 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 регистрах PIR1 и PIR2.

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.

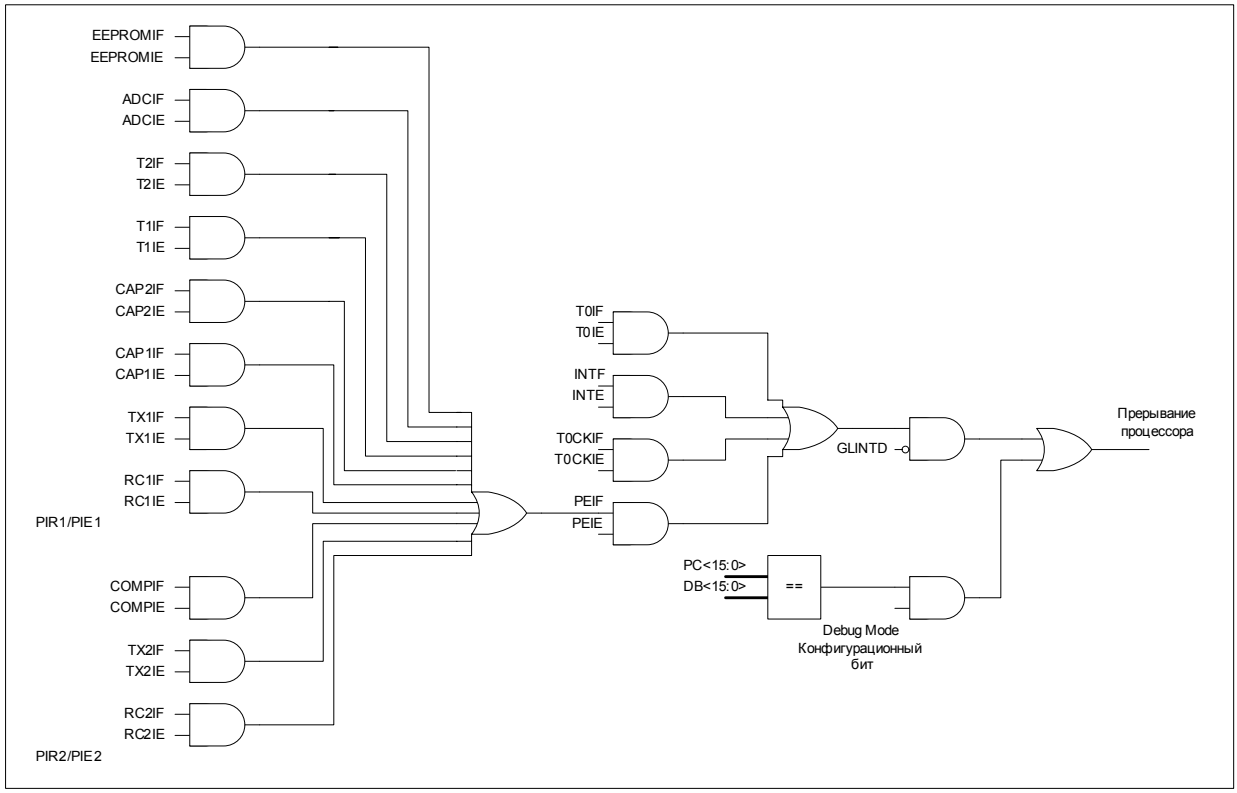


Рис. 13Структурная схема логики прерываний

Регистр состояния прерываний

Таблица 6

INTSTA.ADR = 0x07, Банк доступа BANK = UNBANKED

Номер	7	6	5	4	3	2	1	0
Доступ*	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	PEIF	T0CKIF	T0IF	INTF	PEIE	T0 CKIE	T0IE	INTE

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 7

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
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. 1 = есть не обработанное периферийное прерывание; 0 = нет не обработанного периферийного прерывания
6	T0CKIF	Флаг внешнего запроса прерывания от вывода T0CLK. Этот бит сбрасывается программно, когда логика прерывания направляет выполнение программы по адресу 18h. 1 = на выводе PA1/T0CLK обнаружен заданный фронт сигнала; 0 = на выводе PA1/T0CLK не обнаружен заданный фронт сигнала
5	T0IF	Флаг запроса прерывания по переполнению таймера 0. Этот бит сбрасывается программно, когда логика прерывания направляет выполнение программы по адресу 10h. 1 = таймер TMR0 переполнен; 0 = таймер TMR0 не переполнен
4	INTF	Флаг внешнего запроса прерывания от вывода INT. Этот бит сбрасывается программно, когда логика прерывания направляет выполнение программы по адресу 08h 1 = на выводе PA0/INT обнаружен заданный фронт сигнала; 0 = на выводе PA0/INT не обнаружен заданный фронт сигнала
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
2	T0CKIE	Бит разрешения внешнего прерывания от вывода T0CLK. 1 = прерывание разрешено; 0 = прерывание запрещено
1	T0IE	Бит разрешения прерывания по переполнению таймера 0. 1 = прерывание разрешено; 0 = прерывание запрещено
0	INTE	Бит разрешения внешнего прерывания на выводе PA0/INT. 1 = прерывание разрешено;

		0 = прерывание запрещено

Регистры разрешения периферийных прерываний PIE1 и PIE2

Таблица 8

PIE1. ADR = 0x11, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	EEPROMIE	ADCIE	T2IE	T1IE	CAP2IE	CAP1IE	TX1IE	RC1IE

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 9

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	EEPROMIE	Бит разрешения прерывания от EEPROM. 1 = прерывание разрешено; 0 = прерывание запрещено.
6	ADCIE	Бит разрешения прерывания от АЦП. 1 = прерывание разрешено; 0 = прерывание запрещено.
5	T2IE	Бит разрешения прерывания от Таймера 2. 1 = прерывание разрешено; 0 = прерывание запрещено.
4	T1IE	Бит разрешения прерывания от Таймера 1. 1 = прерывание разрешено; 0 = прерывание запрещено.
3	CAP2IE	Бит разрешения прерывания от схемы захвата 2.
2	CAP1IE	Бит разрешения прерывания от схемы захвата 1.
1	TXIE	Бит разрешения прерывания от передатчика USART1 (буфер передатчика пуст). 1 = прерывание разрешено; 0 = прерывание запрещено.
0	RCIE	Бит разрешения прерывания от приемника USART1 (в буфере приемника есть данные). 1 = прерывание разрешено; 0 = прерывание запрещено.

Таблица 10

PIE2. ADR = 0x13, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	U	U	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	-	-	-	-	-	COMPIE	TX2IE	RC2IE

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 11

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
3..7	-	Зарезервировано
2	COMPIE	Бит разрешения прерывания от Компаратора. 1 = прерывание разрешено; 0 = прерывание запрещено.
1	TX2IE	Бит разрешения прерывания от передатчика USART2 (буфер передатчика пуст). 1 = прерывание разрешено; 0 = прерывание запрещено.
0	RC2IE	Бит разрешения прерывания от приемника USART2 (в буфере приемника есть данные). 1 = прерывание разрешено; 0 = прерывание запрещено.

Регистры запроса периферийных прерываний PIR1 и PIR2

Эти регистры 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 12

PIR1. ADR = 0x10, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO
Значение после сброса**	0	0	0	0	0	0	1	0
	EEPROMIF	ADCIF	T2IF	T1IF	CAP2IF	CAP1IF	TXIF	RCIF

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 13

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	EEPROMIF	Флаг прерывания от EEPROM 1 = есть прерывание 0 = нет прерывания
6	ADCIF	Флаг прерывания от АЦП 1 = есть прерывание 0 = нет прерывания
5	T2IF	Флаг прерывания от Таймера 2 1 = есть прерывание 0 = нет прерывания
4	T1IF	Флаг прерывания от Таймера 1 1 = есть прерывание 0 = нет прерывания
3	CAP2IF	Флаг прерывания от схемы захвата 2 1 = есть прерывание 0 = нет прерывания
2	CAP1IF	Флаг прерывания от схемы захвата 1 1 = есть прерывание 0 = нет прерывания
1	TXIF	Флаг запроса прерывания от передатчика USART. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = буфер передатчика USART1 пуст; 0 = буфер передатчика USART1 заполнен.
0	RCIF	Флаг запроса прерывания от приемника USART1. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = в буфере приемника USART1 есть данные; 0 = буфер приемника USART1 пуст.

Таблица 14

PIR2. ADR = 0x12, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	U	U	RO	RO	RO	RO	RO	RO
Значение после сброса**	0	0	0	0	0	0	0	0
	-	-	-	-	-	COMPIF	TX2IF	RC2IF

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 15

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
3…7	-	Зарезервировано
2	COMPIF	Флаг прерывания от Компаратора 1 = есть прерывание 0 = нет прерывания
1	TX2IF	Флаг запроса прерывания от передатчика USART2. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = буфер передатчика USART2 пуст; 0 = буфер передатчика USART2 заполнен.
0	RC2IF	Флаг запроса прерывания от приемника USART2. Флаг устанавливается и сбрасывается аппаратно, доступен только для чтения. 1 = в буфере приемника USART2 есть данные; 0 = буфер приемника USART2 пуст.

Таблица 16

DBH. ADR = 0x10, Банк доступа BANK = 0x06

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	DB15	DB14	DB13	DB12	DB11	DB10	DB9	DB8

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 17

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7…0	DB<15:8>	Адрес точки остановки и перехода в программу отладчик

Таблица 18

DBH. ADR = 0x11, Банк доступа BANK = 0x06

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 19

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7…0	DB<7:0>	Адрес точки остановки и перехода в программу отладчик

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.

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

периферийных прерываний должны быть сброшены в программе перед разрешением прерываний, чтобы избежать повторного вызова.
Микроконтроллер 1886ВЕ6У имеет 5 векторов прерываний. Адреса векторов и их приоритеты показаны в **Таблица 20**. 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 20
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 на PA1/T0CLK (T0CKIF)	3
0020h	Периферийные прерывания (PEIF)	4 (самый низкий)
0FE0h	Область отладчика (DEBUGIF)	-

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

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 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. Прерывание может быть запрещено сбросом бита T0CKIE (INTSTA<2>). Прерывание 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.

Прерывание отладчика

В рабочем режиме, при установке бита Debug Mode в конфигурационных битах, микроконтроллер работает в режиме отладки. В регистрах DBH и DBL задается адрес остановки программы, когда счетчик команд PC равен адресу остановки процессор переходит в область отладчика. При запуске процессора с адреса 0000h, значение адреса остановки после сброса так же равно 0000h, таким образом процессор сразу переходит в область отладчика. В области отладчика должна быть реализована программа, возвращающая программисту необходимую ему для отладки информацию. После чего управление должно быть возвращено основной программе. Программа отладчик должна быть написана таким образом, что бы ее выполнение не влияло на контекст основной программы.

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 максимум до 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.

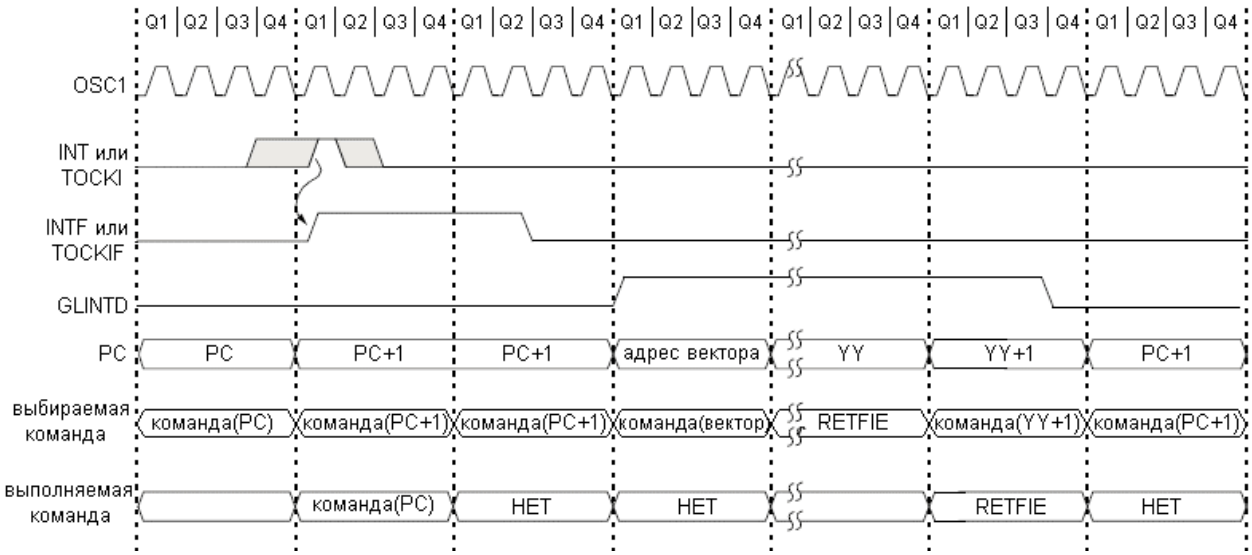


Рис. 14 Временная диаграмма обработки прерываний от выводов INT или T0CLK

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.

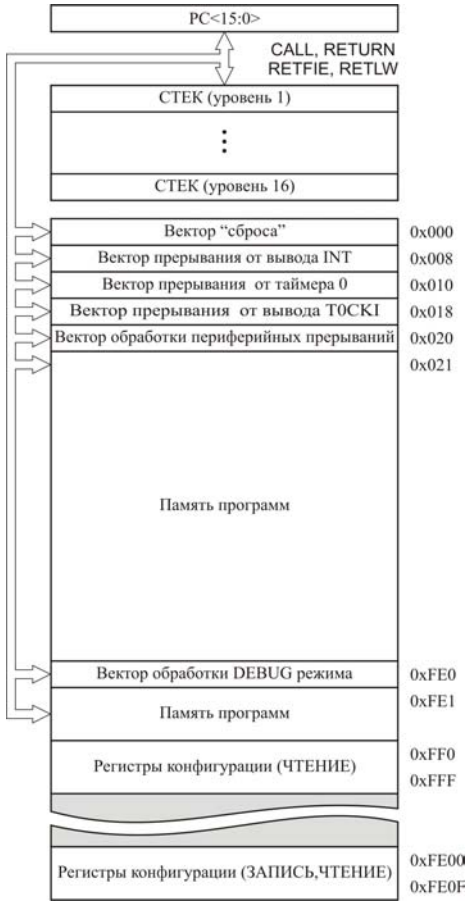


Рис. 15 Карта памяти программ и стека

Память программ

Микроконтроллеры имеют 16-ти битный счетчик команд, способный адресовать область памяти программ размером 64К x 16 бит. Вектор «сброса» имеет адрес 0000h, вектора прерывания находятся по адресам 0008h, 0010h, 0018h и 0020h (см. **Рис.15**).

Объем внутренней памяти программ составляет 4К x 16 бит. Память перепрограммируемая EEPROM. Микроконтроллер может функционировать в одной из 2 возможных конфигураций памяти программ. Конфигурация

выбирается битами конфигурации. Возможны следующие режимы: микроконтроллер и защищенный микроконтроллер.

Режимы «микроконтроллера» и «защищенного микроконтроллера» обеспечивают доступ только к внутренней памяти программ. Любая попытка доступа по адресам над памятью программ приводит к чтению не известного значения. Режим защищенного микроконтроллера еще имеет функцию защиты кода.

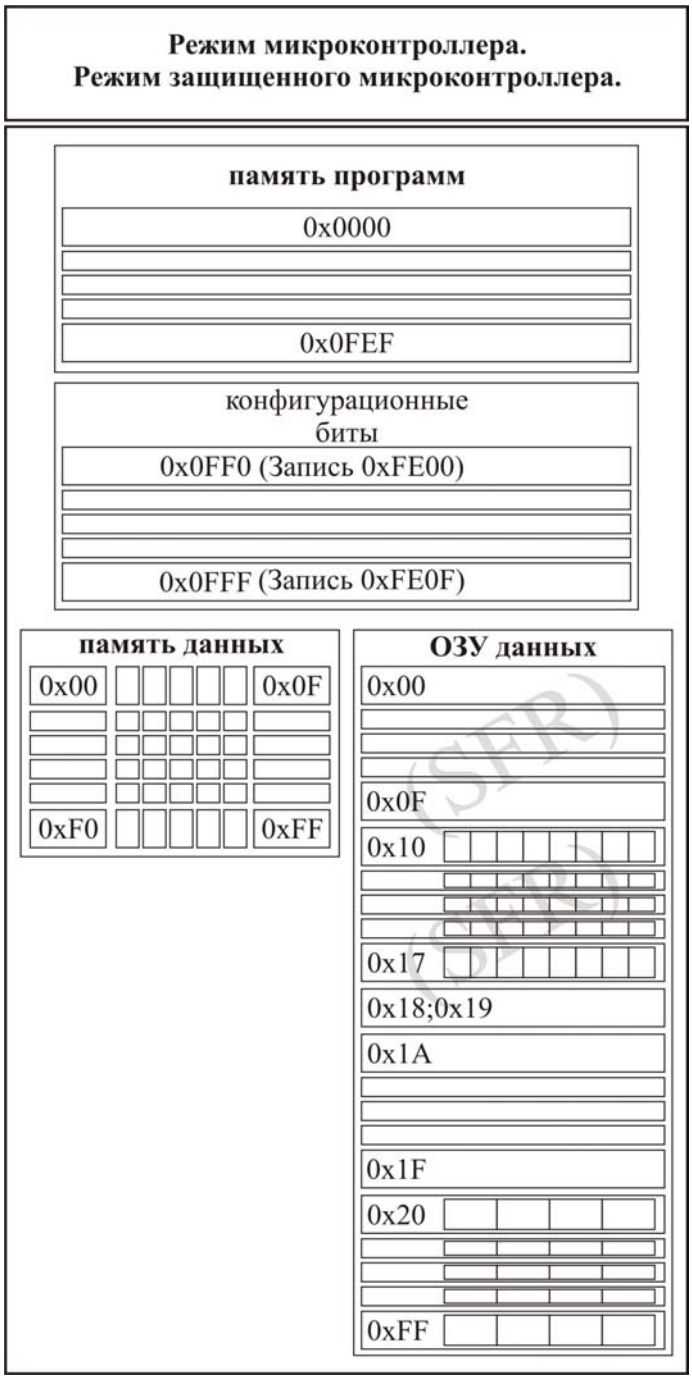


Рис. 16 Распределение памяти в различных режимах

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 байт.

Организация банками требует использования управляющих битов для выбора 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. **Рис. 16** показывает организацию карты памяти данных.

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 FFh. 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 «Косвенная адресация»).

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 21**). 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 21
Схема адресов регистров

Адрес	**Не зависит от номера адресуемого банка.**									
00h	INDF0									
01h	FSR0									
02h	PCL									
03h	PCLATH									
04h	ALUSTA									
05h	TOSTA									
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**	**Банк 14**	**Банк 15**
10h	-	DDRA	TMR1	PW1DCL	PW1DCHH	PIR1	DBH	-	-	-
11h	LIN1 CNTR	PORTA	TMR1H	PW2DCL	PW2DCHH	PIE1	DBL	LIN2 CNTR	EECONL	-
12h	LIN1 BRG	DDRC	TMR2	PW1DCH	DAC_CONT	PIR2	-	LIN2 BRG	EECONH	EDLSB
13h	RCSTA1	PORTC	TMR2H	PW2DCH	DAC1L	PIE2	EEDIV	RCSTA2	-	EDMSB
14h	RCREG1	DDRD	PR1	CA2L	DAC1H	EE_CONT	ADCON0	RCREG2	-	EEMOD
15h	TXSTA1	PORTD	PR1H	CA2H	DAC2L	EE_MODE	ADCON1	TXSTA2	-	EAMSB
16h	TXREG1	-	PR2/CA1L	TCON1	DAC2H	EE_DATA	ADRESL	TXREG2	-	EALSB
17h	SPBRG1	-	PR2H/CA1H	TCON2	COMPARE	EE_ADR	ADRESH	SPBRG2	-	CFREG
	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.		

Примечания:

1. 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.
2. 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.

3. Чтение любого не существующего регистра дает значение равное нулю.

Таблица 22

Регистры специального назначения

Адрес	Название	**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 (не физический регистр)								---- ----	---- ----
01h	FSR0	указатель адреса 0, для косвенной адресации памяти данных								xxxx xxxx	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	uuuu uuuu
04h	ALUSTA	FS3	FS2	FS1	FS0	OV	Z	DC	C	1111 xxxx	1111 uuuu
05h	TOSTA	INTEDG	TOSE	TOCS	TOPS3	TOPS2	TOPS1	TOPS0	-	0000 000-	0000 000-
06h	CPUSTA	-	ESLP	STKAV	GLINTD	TO	PD	POR	BOR	--11 11qq	--11 qquu
07h	INTSTA	PEIF	TOCKIF	TOIF	INTF	PEIE	TOCKIE	TOIE	INTE	0000 0000	0000 0000
08h	INDF1	использует содержимое FSR1 для адресации памяти данных (не физический регистр)								---- ----	---- ----
09h	FSR1	указатель адреса 1, для косвенной адресации памяти данных								xxxx xxxx	uuuu uuuu
0Ah	WREG	рабочий регистр								xxxx xxxx	uuuu uuuu
0Bh	TMR0L	младший байт регистра таймера 0								xxxx xxxx	uuuu uuuu
0Ch	TMR0H	старший байт регистра таймера 0								xxxx xxxx	uuuu uuuu
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	-									xxxx xxxx	uuuu uuuu
11h	LIN1 CNTR	BRK CNT 3	BRK CNT 2	BRK CNT 1	BRK CNT 0	BRK	SYNCH	ERR	LINEN		
12h	LIN1 BRG	LINBRG[7:0]									
13h	RCSTA1	SPEN	RX9	SREN	CREN	-	FERR	OERR	RX9D		
14h	RCREG1	RCREG[7:0]									
15h	TXSTA1	CSRC	TX9	TXEN	SYNC	-	-	TRMT	TX9D		
16h	TXREG1	TXREG[7:0]									
17h	SPBRG1	SPBRG[7:0]									
Банк 1											
10h	DDRA	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	-	-	1111 11--	1111 11--
11h	PORTA	PORTA[7:0]								xxxx xxxx	uuuu uuuu
12h	DDRC	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0	1111 1111	1111 1111
13h	PORTC	PORTC[7:0]								xxxx xxxx	uuuu uuuu
14h	DDRD	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0	1111 1111	1111 1111
15h	PORTD	PORTD[7:0]								xxxx xxxx	uuuu uuuu
16h	-									xxxx xxxx	uuuu uuuu
17h	-									xxxx xxxx	uuuu uuuu
Банк 2											
10h	TMR1	TMR1[7:0]								**0000 0000**	**0000 0000**

11h	TMR1H	TMR1[15:8]								**0000 0000**	**0000 0000**
12h	TMR2	TMR2[7:0]								**0000 0000**	**0000 0000**
13h	TMR2H	TMR2[15:8]								**0000 0000**	**0000 0000**
14h	PR1	PR1[7:0]								**1111 1111**	**1111 1111**
15h	PR1H	PR1[15:8]								**0000 0000**	**0000 0000**
16h	PR2/CA1L	PR2[7:0]/ CA1[7:0]								**0000 0000**	**0000 0000**
17h	PR2H/CA1H	PR2[15:8]/ CA1[15:8]								**0000 0000**	**0000 0000**
Банк 3											
10h	PW1DCL	DC1	DC0							**0000 0000**	**0000 0000**
11h	PW2DCL	DC1	DC0							**0000 0000**	**0000 0000**
12h	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	**0000 0000**	**0000 0000**
13h	PW2DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	**0000 0000**	**0000 0000**
14h	CA2L	младший байт регистра захвата 2								0000 0000	0000 0000
15h	CA2H	старший байт регистра захвата 2								0000 0000	0000 0000
16h	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	--	TMR2CS	--	TMR1CS	**0000 0000**	**0000 0000**
17h	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/PR2	TMR2ON	--	TMR1ON	**0000 0000**	**0000 0000**
Банк 4											
10h	PW1DCHH	DC17	DC16	DC15	DC14	DC13	DC12	DC11	DC10	0000 0000	0000 0000
11h	PW2DCHH	DC17	DC16	DC15	DC14	DC13	DC12	DC11	DC10	0000 0000	0000 0000
12h	DAC_CONT	SYNC_A	DACON1	ENOUT1	MREF1	DACON0	ENOUT0	MREF0	DACFM	0000 0000	0000 0000
13h	DAC1L	DAC1[7:0]								0000 0000	0000 0000
14h	DAC1H	DAC1[7:0]								0000 0000	0000 0000
15h	DAC2L	DAC2[15:8]								0000 0000	0000 0000
16h	DAC2H	DAC2[15:8]								0000 0000	0000 0000
17h	COMPARE	RESULT	STAGE	C_IF	E_L	PH_NL	CONTR1	CONTR0	COMP_ON	0000 0000	0000 0000
Банк 5											
10h	PIR1	EEPROMIF	ADCIF	T2IF	T1IF	CAP2IF	CAP1IF	TX1IF	RC1IF	0000 0000	0000 0000
11h	PIE1	EEPROMIE	ADCIE	T2IE	T1IE	CAP2IE	CAP1IE	TX1IE	RC1IE	0000 0000	0000 0000
12h	PIR2	-	-	-	-	-	COMPIF	TX2IF	RC2IF	---- -000	---- -000
13h	PIE2	-	-	-	-	-	COMPIE	TX2IE	RC2IE	---- -000	---- -000
14h	EE_CON	TESTp	EE TEST	CP TEST	VEE2	VEE1	EEBRG[2:0]			0000 0000	0000 0000
15h	EE_MODE	EN	-	IFBUSY	BUSY	-	MODE[2:0]			0-00 -000	0-00 -000
16h	EE_DATA	EEDATA[7:0]								0000 0000	0000 0000
17h	EE_ADR	EEADR[7:0]								0000 0000	0000 0000
Банк 6											
10h	DBH	DB[15:8]								0000 0000	0000 0000
11h	DBL	DB[7:0]								0000 0000	0000 0000
12h	-	-								xxxx xxxx	uuuu uuuu
13h	EEDIV	EEDIV[7:0]								0000 0101	0000 0101

14h	ADCON0	-	CHS2	CHS1	CHS0	-	GO/ DONE	-	ADON	-000 -0-0	-000 -0-0
15h	ADCON1	ADCS1	ADCS0	ADFM	-	PCFG3	PCFG2	PCFG1	PCFG0	000-0000	000-0000
16h	ADRESL	ADRESL[7:0]								0000 0000	0000 0000
17h	ADRESH	ADRESH[7:0]								0000 0000	0000 0000
Банк 7											
10h	-	-								xxxx xxxx	uuuu uuuu
11h	LIN2 CNTR	BRK CNT 3	BRK CNT 2	BRK CNT 1	BRK CNT 0	BRK	SYNCH	ERR	LINEN	0000 0000	0000 0000
12h	LIN2 BRG	LINBRG[7:0]								0000 0000	0000 0000
13h	RCSTA2	SPEN	RX9	SREN	CREN	-	FERR	OERR	RX9D	0000 -000	0000 -000
14h	RCREG2	RCREG[7:0]								0000 0000	0000 0000
15h	TXSTA2	CSRC	TX9	TXEN	SYNC	-	-	TRMT	TX9D	0000 --00	0000 --00
16h	TXREG2	TXREG[7:0]								0000 0000	0000 0000
17h	SPBRG2	SPBRG[7:0]								0000 0000	0000 0000
Банк 14											
10h	-	-								xxxx xxxx	uuuu uuuu
11h	EECONL	LOCKL	ERRL	EETL	EBWL	EBEL	EERL	EWRL	ERDL	0000 0000	0000 0000
12h	EECONH	LOCKH	ERRH	EETH	EBWH	EBEH	EERH	EWRH	ERDH	0000 0000	0000 0000
13h	-	-								xxxx xxxx	uuuu uuuu
14h	-	-								xxxx xxxx	uuuu uuuu
15h	-	-								xxxx xxxx	uuuu uuuu
16h	-	-								xxxx xxxx	uuuu uuuu
17h	-	-								xxxx xxxx	uuuu uuuu
Банк 15											
10h	-	-								xxxx xxxx	uuuu uuuu
11h	-	-								xxxx xxxx	uuuu uuuu
12h	EDLSB	ED[7:0]								0000 0000	0000 0000
13h	EDMSB	ED[15:8]								0000 0000	0000 0000
14h	EEMOD	State	Reset	CPen	HWs	AM	TM2	TM1	ESTBY	0000 0000	0000 0000
15h	EAMSB	CPrdy	-	-	-	EA[11]	EA[10]	EA[9]	EA[8]	0--- 0000	0--- 0000
16h	EALSB	EA[7:0]								0000 0000	0000 0000
17h	CFREG	-	DBG_En	BODEN	PM0	WDT1	WDT0	FOSC1	FOSC0	-000 0000	-000 0000
Не зависит от номера адресуемого банка.											
18h	PRODL	младший байт 16-ти битного результата (8x8 битное аппаратное умножение)								xxxx xxxx	uuuu uuuu
19h	PRODH	старший байт 16-ти битного результата (8x8 битное аппаратное умножение)								xxxx xxxx	uuuu uuuu

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>, его содержимое обновляется от, или записывается в старший байт счетчика команд.
2. Внешний сброс от вывода MCLRn 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.
5. При любом сбросе прибора эти выводы конфигурируются как входы.

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 23

ALUSTA. ADR = 0x04, Банк доступа BANK = UNBANKED

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	1	1	1	1	0	0	0	0
	FS3	FS2	FS1	FS0	OV	Z	DC	C

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 24

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..6	FS[3:2]	биты выбора режима FSR1 00 = автодекремент величины FSR1 после выполнения команды 01 = автоинкремент величины FSR1 после выполнения команды 1x = значение FSR1 не изменяется
5..4	FS[1:0]	биты выбора режима FSR0 00 = автодекремент величины FSR0 после выполнения команды 01 = автоинкремент величины FSR0 после выполнения команды 1x = значение FSR0 не изменяется
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
2	Z	флаг нуля 1 = результат арифметической или логической операции равен 0 0 = результат арифметической или логической операции не равен 0
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).
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).

Регистр статуса ЦПУ (CPUSTA)

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 (PIE). 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 MCLRn 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 25

CPUSTA. ADR = 0x06, Банк доступа BANK = UNBANKED

Номер	7	6	5	4	3	2	1	0
Доступ*	U	R/W	RO	R/W	RO	RO	R/W	R/W
Значение после сброса**	0	0	1	1	1	1	0	0
	-	ESLP	STKAV	GLINTD	TO	PD	POR	BOR

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 26

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	-	Зарезервировано
6	ESLP	Бит управления EEPROM памяти программ в режиме SLEEP. 0 = память не отключается при переходе в SLEEP режим, вызывает повышение потребления, при выходе из режима память стазу готова к работе. 1 = память отключается при переходе в SLEEP режим, снижает потребление, при выходе из режима требуется дополнительное время порядка 10 мкс для запуска схемы памяти.
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)
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
3	TO	флаг переполнения сторожевого таймера 1 = устанавливается после включения питания или выполнения команд CLRWDT или SLEEP 0 = после переполнения сторожевого таймера
2	PD	флаг включения питания 1 = устанавливается после включения питания или выполнения команды CLRWDT 0 = после выполнения команды SLEEP
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)
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

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), при котором на выводе PA0/INT будет устанавливаться флаг запроса прерывания PA0/INT. Остальные биты конфигурируют таймер 0, его предделитель и источник тактовых сигналов.

Таблица 27

T0STA. ADR = 0x05, Банк доступа BANK = UNBANKED

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	U
Значение после сброса**	0	0	0	0	0	0	0	0
	INTEDG	T0SE	T0CS	T0PS3	T0PS2	T0PS1	T0PS0	-

* R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 28

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
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
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
5	T0CS	бит выбора источника тактирования для таймера 0. Этот бит выбирает источник синхронизации для таймера 0. 1 = внутренняя тактовая частота с генератора циклов (T_{cy}) 0 = внешнее тактирование с вывода T0CLK
4..1	T0PS[3:0]	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
0	-	Зарезервировано

Функционирование стека

Микроконтроллеры имеют 16x16 бит аппаратный стек (см. **Рис. 15**). 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. **17** показан принцип косвенной адресации. Там показана модификация значения адреса памяти данных, значением регистра.

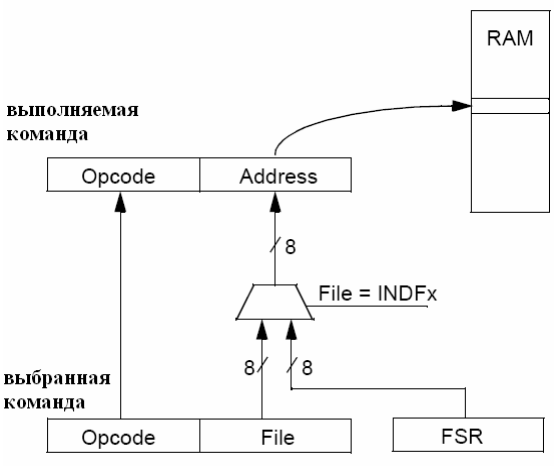


Рис. 17 Косвенная адресация

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 «Считывание и запись таблиц данных».

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.

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. 1 и Рис. 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>

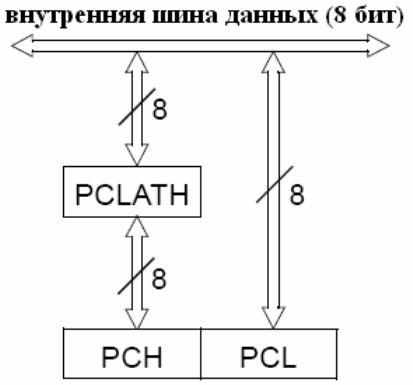


Рис. 1 Функционирование счетчика команд

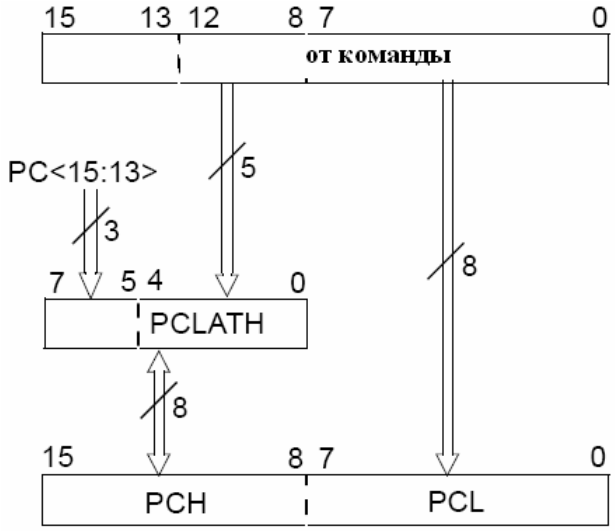


Рис. 18 Счетчик команд при выполнении инструкций Call и GOTO

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 с 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 (Рис. **)**. 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.

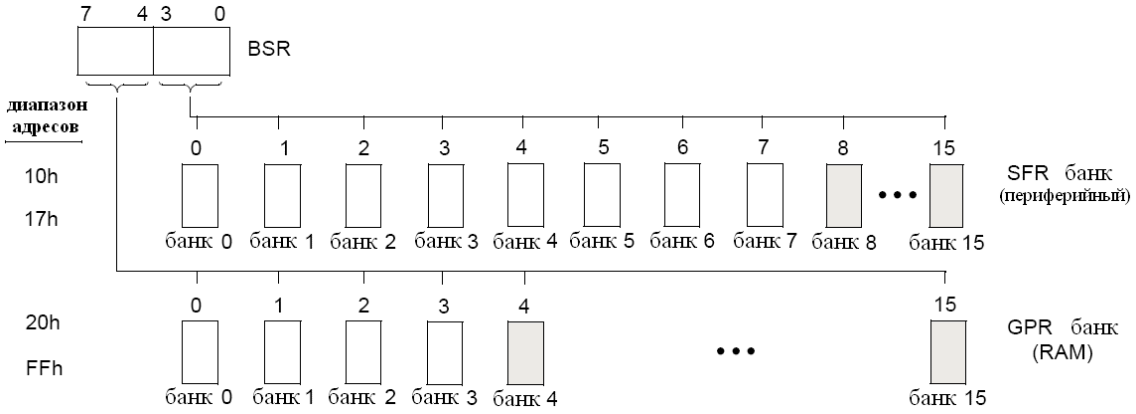
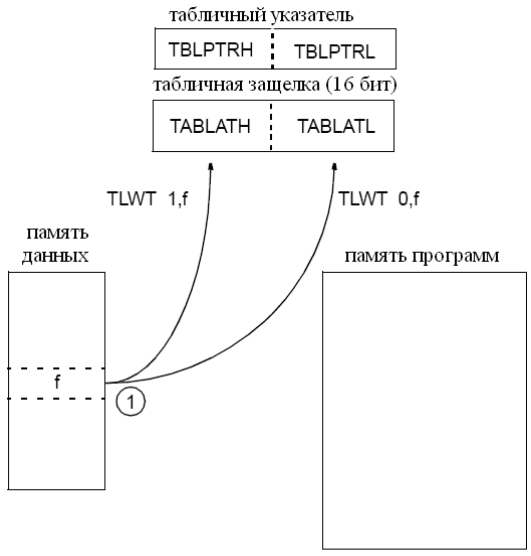


Рис. 19 Функционирование BSR

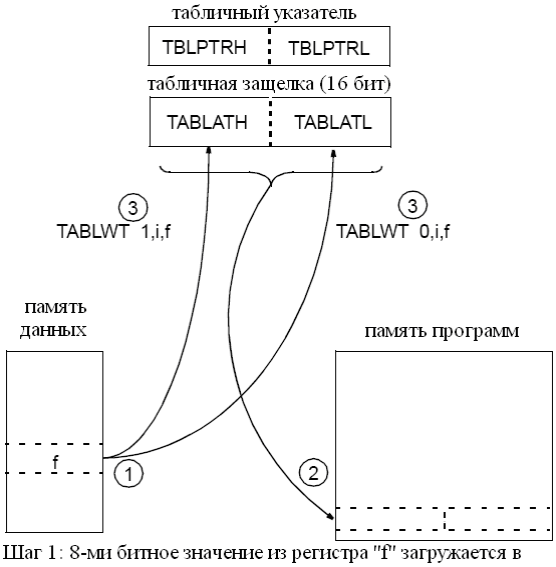
Считывание и запись таблиц данных

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). Регистр TBLPTR автоматически не инкрементируется. Рис. показывает выполнение этих 4 команд. Шаги показывают последовательность операций. Память программ может быть как внутренней, так и внешней (для режима «микропроцессор» или «расширенный микроконтроллер»).



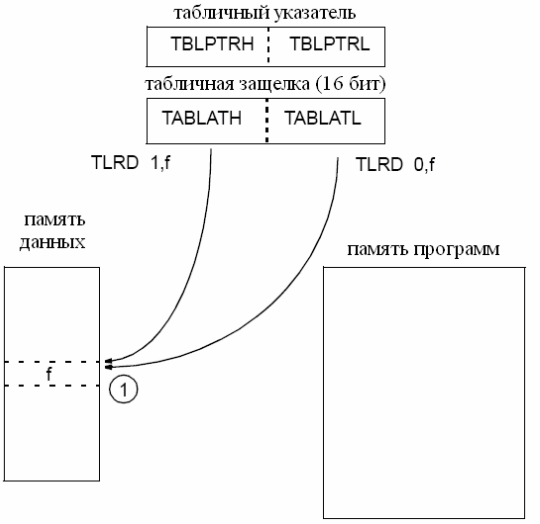
Шаг 1: 8-ми битное значение из регистра "f" загружается в старший или младший байт в TABLAT (16 бит).

а)



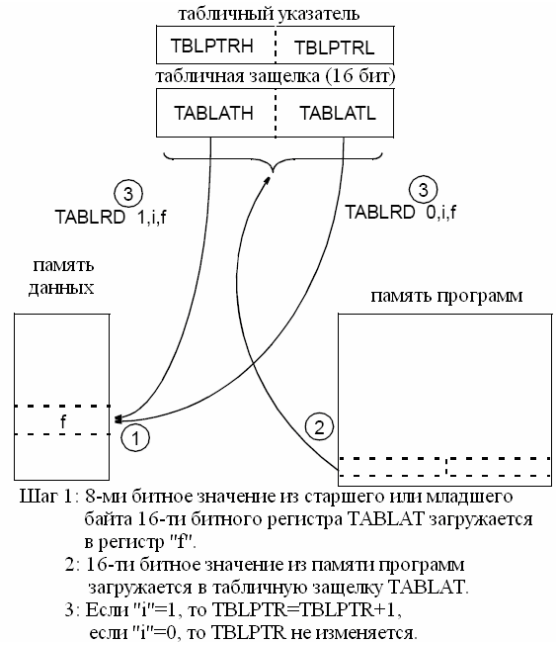
Шаг 1: 8-ми битное значение из регистра "f" загружается в старший или младший байт TABLAT (16 бит).
2: 16-ти битное значение TABLAT записывается в память программ по адресу табличного указателя (TBLPTR).
3: Если "i"=1, то TBLPTR=TBLPTR+1, если "i"=0, то TBLPTR не изменяется.

б)



Шаг 1: 8-ми битное значение из старшего или младшего байта 16-ти битного регистра TABLAT загружается в регистр "f".

в)



Шаг 1: 8-ми битное значение из старшего или младшего байта 16-ти битного регистра TABLAT загружается в регистр "f".
2: 16-ти битное значение из памяти программ загружается в табличную защелку TABLAT.
3: Если "i"=1, то TBLPTR=TBLPTR+1, если "i"=0, то TBLPTR не изменяется.

г)

Рис. 20 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 EEPROM 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 MCLRn/U_{pp} 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 может (включая температуру) привести к тому, что ячейка памяти будет запрограммирована не полностью и со временем может стереться. Если напряжение программирования (U_{pp}) 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 в память в EEPROM задается 8-ми разрядным регистром **EEDIV** (коэффициент деления частоты генератора). Требуемое значение длительности 5 мс. Коэффициент деления выбирается: K = 5 мс / T_c, где T_c в мс. Значение регистра после сброса равно 05h. Если предполагается производить запись в EEPROM память и частота генератора отличается от 1 МГц, то необходимо предварительно загрузить в этот регистр необходимое значение. Регистр доступен по чтению и записи.

Таблица 29

Воздействие прерываний на операцию «длинной записи»

источник прерываний	GLINTD	бит разрешения	флаг запроса	действие
PA0/INT, PA1/T0CLK, TMR0(прим.)	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.
периферийные прерывания	0	1	1	Заканчивает длинную запись таблиц, 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.

Чтение таблиц

Операция чтения таблиц осуществляет чтение памяти программ. Это позволяет хранить константы в памяти программ и считывать их в память данных при необходимости. В **Примере 4** показано 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 4

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
```

Аппаратный умножитель

Микроконтроллеры имеют 8x8 битный аппаратный умножитель, включенный в АЛУ прибора. Из-за того, что умножение реализовано аппаратно, оно выполняется за один цикл. 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. 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.

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 30**).

Пример 5 показывает последовательность действий при 8x8 беззнаковом умножении. Требуется только одна команда, когда один аргумент для умножения уже загружен в регистр WREG.

Пример 6 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 7 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 8 приводит последовательность действий при 16x16 знаковом умножении. В **уравнении 2** 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 30

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	без аппаратного умножителя	-	-	-	-	-

со знаком	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 знаком	без аппаратного умножителя	52	254	30.788	63.50	127.0
	аппаратный умножитель	36	36	4.36	9.0	18.0

Пример 5

Программа 8 x 8 битного беззнакового умножения

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

Пример 6

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
```

Уравнение 1

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 7

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
```

Уравнение 2

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 8

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. Все порты 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» 8-и разрядный. По сигналу «сброс», выводы «порта A» принудительно конфигурируются, как «вход» с высоким входным сопротивлением. Выводы PA0 и PA1 всегда сконфигурированы как вход. Направление данных на выводах PA2, PA3, PA4 и PA5, контролируется периферийными модулями USART или регистром DDRA. Направление данных на выводах PA6 и PA7 контролируется периферийным модулем DAC или регистром DDRA. 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 и PA3 мультиплексированы с периферийным модулем USART1. Выводы PA4 и PA5 мультиплексированы с периферийным модулем USART2.

Не рекомендуется использовать команды чтение-модификация-запись (например, BCF, BSF, BTG) над регистром «порта A». Такие операции могут изменить состояние выходного триггера-защелки, что приведет к переключению от состояния выхода на вход или наоборот. Для того чтобы избежать этого, используйте дополнительный регистр, а затем переместите его значение в регистр «порта A».

Таблица 31

Название	Бит	Тип входного буфера	Функция
PA0/INT	0	Триггер Шмитта	Вход или вход внешнего прерывания.
PA1/T0CLK	1		Вход или вход тактового сигнала «таймера 0».
PA2/RX1/DT1	2		Пользовательский вывод/Вход или вход приемника асинхронного USART1, или вход/выход данных синхронного USART1.
PA3/TX1/CK1	3		Пользовательский вывод/Вход или выход асинхронного передатчика USART1, или вход/выход синхросигнала синхронного USART1.
PA4/RX2/DT2	4		Пользовательский вывод/Вход или вход приемника асинхронного USART2, или вход/выход данных синхронного USART2.
PA5/TX2/CK2	5		Пользовательский вывод/Вход или выход асинхронного передатчика USART2, или вход/выход синхросигнала синхронного USART2.
PA6	6		Пользовательский вывод
PA7	7		Пользовательский вывод

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. Выводы порта C мультиплексированы с аналоговыми входами АЦП.

Таблица 32

Название	Бит	Тип входного буфера	Функция
PC0/ADC0/Vref+	0	**ТТЛ**	Вход/выход или 0 канал АЦП
PC1/ADC1/Vref-	1		Вход/выход или 1 канал АЦП
PC2/ADC2	2		Вход/выход или 2 канал АЦП
PC3/ADC3	3		Вход/выход или 3 канал АЦП
PC4/ADC4	4		Вход/выход или 4 канал АЦП
PC5/ADC5	5		Вход/выход или 5 канал АЦП
PC6/ADC6	6		Вход/выход или 6 канал АЦП
PC7/ADC7	7		Вход/выход или 7 канал АЦП

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.

Выводы порта D мультиплексированы с выводами блока Таймер 1, 2 и Компаратора, поэтому при работе с портом необходимо обратить внимание на значение регистров управления конфигурацией блоков Таймер 1, 2 и Компаратора.

Таблица 33

Название	Бит	Тип входного буфера	Функция
PD0/CAP1	0	**ТТЛ**	Вход/выход или вход 1 схемы регистрации событий
PD1/CAP2	1		Вход/выход или вход 2 схемы регистрации событий
PD2/PWM1	2		Вход/выход или выход 1 схемы ШИМ
PD3/PWM2	3		Вход/выход или выход 2 схемы ШИМ
PD4/T1CLK	4		Вход/выход или вход внешнего сигнала синхронизации Таймера 1.
PD5/T2CLK	5		Вход/выход или вход внешнего сигнала синхронизации Таймера 2.
PD6/COMP1	6		Вход/выход или Вход компаратора
PD7/COMP2	7		Вход/выход или Вход компаратора

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 34

T0STA.ADR = 0x05, банк доступа BANK = UNBANKED.

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	U
Значение после сброса**	0	0	0	0	0	0	0	0
	INTEDG	T0SE	T0CS	T0PS3	T0PS2	T0PS1	T0PS0	-

* R/W - бит доступен на чтение и запись
RO - бит доступен только на чтение
U - бит физически не реализован или зарезервирован.

Таблица 35

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений.
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
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
5	T0CS	бит выбора источника тактирования для «таймера 0». Этот бит выбирает источник синхронизации для «таймера 0». 1 = внутренняя тактовая частота с генератора циклов

		0 = внешнее тактирование с вывода PA1/T0CLK
4..1	T0PS[3:0]	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
0	-	Зарезервировано

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). 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. 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. 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.

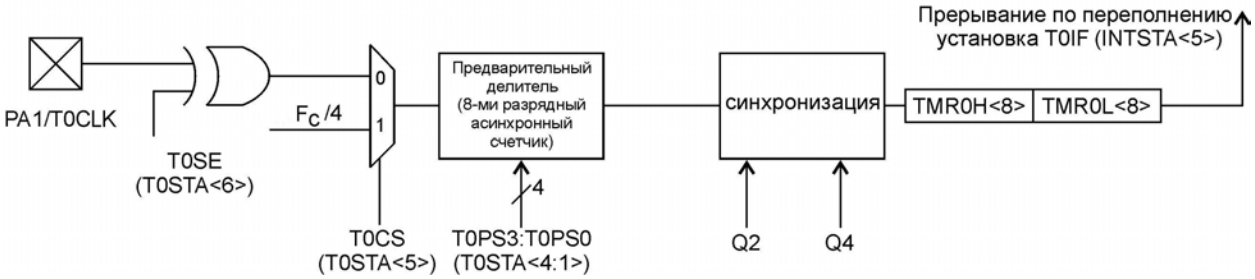


Рис. 21 Блок-схема модуля «таймер 0»

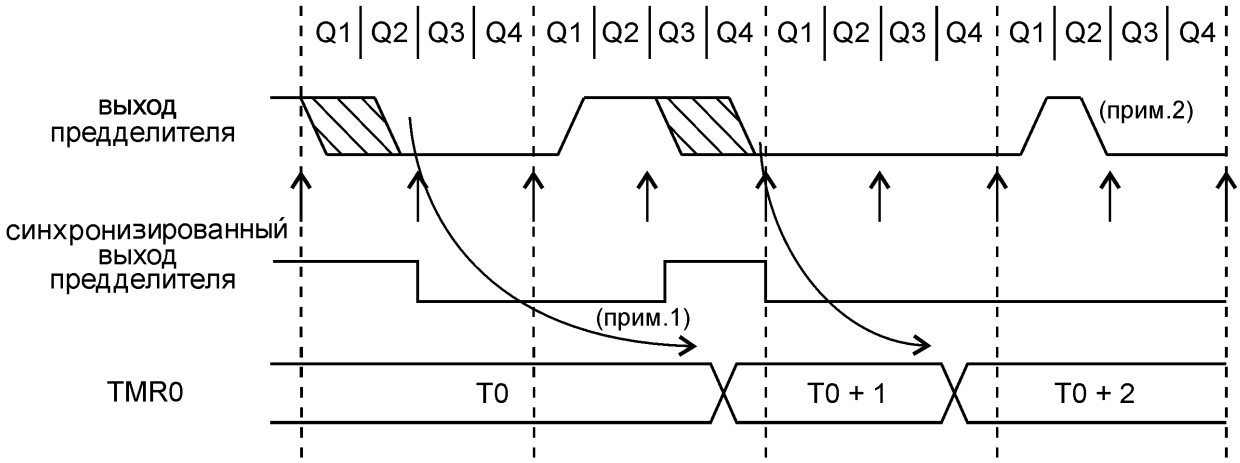


Рис. 22 Функционирование «таймер 0» от внешнего источника тактовых импульсов

Примечания:

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

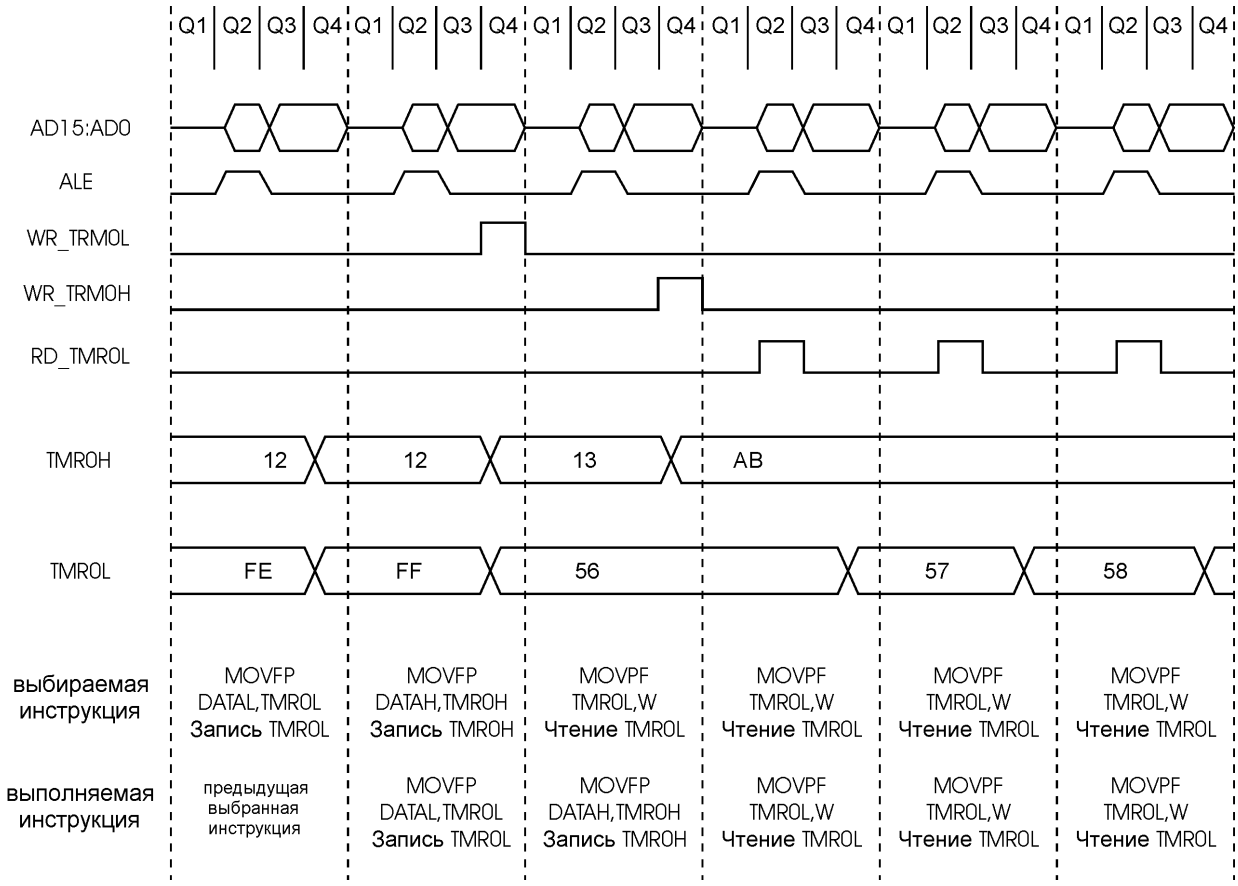


Рис. 23 Функционирование «таймер 0» при обращении к регистрам TMR0L и TMR0H

Примечание:
В примере записывается значение TMR0 равное AB56h.

Таймер 1, таймер 2, ШИМ, захват (регистрация событий)

«Таймер 1» представляет собой 16-ти разрядный таймер/счетчик (TMR1 и TMR1H), с 16-ти разрядным регистром периода (PR1) и флагом запроса прерывания при переполнении. Таймер 1 может работать как таймер, инкрементирующийся от внутренних тактовых импульсов Fc/4, либо инкрементирующийся на заднем фронте внешнего тактового сигнала с вывода PD4/T1CLK. Этот таймер также используется как опорный для модулей широтно-импульсных модуляторов.

«Таймер 2» является 16-разрядным таймером/счетчиком (регистры TMR2H и TMR2L). Он содержит два дополнительных регистра (PR2H/CA1H:PR2L/CA1L), которые используются как 16-ти разрядный регистр периода, или как 16-ти разрядный регистр «захвата 1». Для приращения «таймера 2» может использоваться или внутренний тактовый сигнал Fc/4, или внешний тактовый сигнал с вывода PD5/T2CLK. Этот таймер используется как опорный для всех 16-ти разрядных захватов (регистраций событий). Два других дополнительных регистра включают регистры захвата (регистрации событий): 2 (CA2H:CA2L).

Микроконтроллер имеет два выхода ШИМ (широтно-импульсных модуляторов) и два входа захвата (регистрации событий). **Таблица 36**

показывает использование ресурсов таймеров для реализации функций ШИМ и захватов.

Таблица 36

Ресурсов таймеров для реализации функций ШИМ и захватов.

Функция	**Ресурсы таймера**
ШИМ 1	Таймер 1
ШИМ 2	Таймер 1
Захват 1	Таймер 2
Захват 2	Таймер 2

Таблица 37

TCON1.ADR = 0x16, банк доступа BANK = 0x03

R/W-0	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**	**R/W-0**
CA2ED1	**CA2ED0**	**CA1ED1**	**CA1ED0**	**-**	**TMR2CS**	**-**	**TMR1CS**
бит 7	6	5	4	3	2	1	бит 0
бит 7, 6	**CA2ED1:CA2ED0:** биты выбора режима «захвата 2». 00 = Захват на каждом заднем фронте (спаде) 01 = Захват на каждом переднем фронте 10 = Захват на каждом 4-м переднем фронте 11 = Захват на каждом 16-м переднем фронте						
бит 5, 4	**CA1ED1:CA1ED0:** биты выбора режима «захвата 1». 00 = Захват на каждом заднем фронте (спаде) 01 = Захват на каждом переднем фронте 10 = Захват на каждом 4-м переднем фронте 11 = Захват на каждом 16-м переднем фронте						
бит 3	Не используется						
бит 2	**TMR2CS:** бит выбора источника тактовых импульсов «таймера 2» 1 = внешний тактовый сигнал с вывода PD5/T2CLK (приращение по заднему фронту сигнала) 0 = внутренний тактовый сигнал FC/4						
бит 1							
бит 0	**TMR1CS:** бит выбора источника тактовых импульсов «таймера 1». 1 = внешний тактовый сигнал с вывода PD4/T1CLK (приращение по заднему фронту сигнала) 0 = внутренний тактовый сигнал FC/4						

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 38

TCON2.ADR = 0x17, банк доступа BANK = 0x03 Регистр

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CA2OVF	**CA1OVF**	**PWM2ON**	**PWM1ON**	**CA1_PR2**	**TMR2ON**	**-**	**TMR1ON**
бит 7	6	5	4	3	2	1	бит 0

бит 7	**CA2OVF:** бит переполнения «захвата 2». Этот бит означает, что значение регистров захвата (CA2H:CA2L) не было считано до того как произошел следующий захват. Регистр захвата сохраняет старое несчитанное значение захвата (последний захват перед переполнением). Последующие захваты не обновят значение регистра захвата до тех пор, пока регистр захвата не будет считан (оба байта). 1 = произошло переполнение 0 = нет переполнения
бит 6	**CA1OVF:** бит переполнения «захвата 1». Этот бит означает, что значение регистров захвата (PR3H/CA1H:PR3L/CA1L) не было считано до того как произошел следующий захват. Регистр захвата сохраняет старое несчитанное значение захвата (последний захват перед переполнением). Последующие захваты не обновят значение регистра захвата до тех пор, пока регистр захвата не будет считан (оба байта). 1 = произошло переполнение 0 = нет переполнения
бит 5	**PWM2ON:** бит включения «ШИМ 2». 1 = «ШИМ 2» включен, вывод PD3/PWM2 игнорирует состояние бита DDRD<3> 0 = «ШИМ 2» выключен, вывод PD3/PWM2 использует состояние бита DDRD<3> для направления передачи данных.
бит 4	**PWM1ON:** бит включения «ШИМ 1». 1 = «ШИМ 1» включен, вывод PD2/PWM1 игнорирует состояние бита DDRD<2> 0 = «ШИМ 1» выключен, вывод PD2/PWM1 использует состояние бита DDRD<2> для направления передачи данных.
бит 3	**CA1_PR2:** бит выбора режима 1 = активирует «захват 1», регистр PR2H/CA1H:PR2L/CA1L является регистром «захвата 1». «Таймер 2» работает без регистра периода. 0 = включает регистр периода. PR2H/CA1H:PR2L/CA1L является регистром периода для «таймера 2».
бит 2	**TMR2ON:** бит включения «таймера 2». 1 = «таймер 2» включен 0 = «таймер 2» остановлен
бит 1	Не используется
бит 0	**TMR1ON:** бит включения «таймера 1». 1 = запускает 16-ти разрядный «таймер 1» 0 = останавливает 16-ти разрядный «таймер 1»

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.

«Таймер 1»

Источник тактовых сигналов для таймера может быть настроен, это или внутренняя тактовая частота FC/4, или внешний сигнал с вывода PD4/T1CLK. Источник импульсов таймера конфигурируется битом TMR1CS. Если бит TMR1CS=0, то источник тактовых импульсов внутренний и таймер инкрементируется каждый командный цикл (частота $F_C/4$). Если бит TMR1CS установлен, то источник импульсов внешний сигнал с вывода PD4/T1CLK и таймер инкрементируется на каждом заднем фронте (спаде) сигнала T1CLK. Сигнал с входа T1CLK синхронизируется с внутренним тактовым сигналом, это вызывает задержку между спадом сигнала на выводе и приращением таймера. Временные требования к внешнему тактовому сигналу смотрите в «электрических параметрах».

Значение таймера инкрементируется от 00h до момента равенства с значением регистра периода (PR1). В следующем цикле приращения он сбрасывается в 00h. Флаг запроса прерывания от таймера устанавливается, когда таймер сбрасывается. «Таймер 1» имеет индивидуальный бит флага запроса прерывания (T1IF).

Таймер имеет индивидуальный бит разрешения прерываний (T1IE). Прерывание от таймера может быть разрешено установкой и запрещено очисткой этого бита. Также для разрешения прерывания должен быть установлен бит разрешения прерываний от периферийных устройств (PEIE=1) и сброшен бит глобального запрещения прерываний (GLINTD=0).

Таймер может включаться и выключаться программно установкой или сбросом соответствующего управляющего бита TMR1ON.

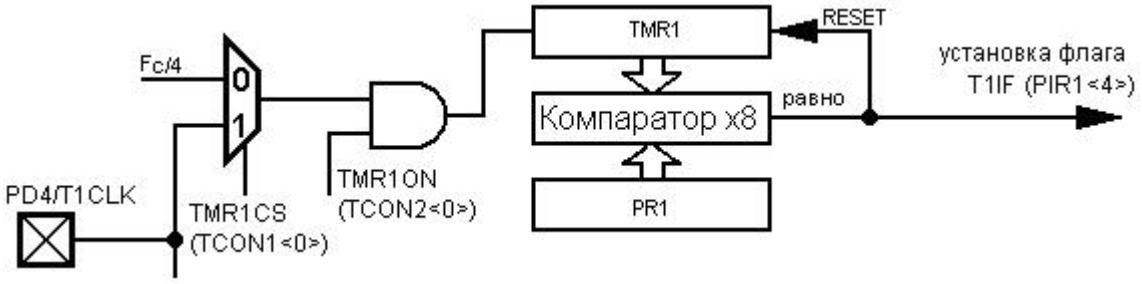


Рис. 24 «Таймер 1»

Использование выходов широтно-импульсных модуляторов (ШИМ)

Микроконтроллер содержит два высокоскоростных выхода ШИМ. Выходы «ШИМ 1» и «ШИМ 2» используют как опорный счетчик «таймер 1». Выходы ШИМ подключены к выводам PD2/PWM1, PD3/PWM2.

Каждый выход ШИМ имеет максимальное разрешение 18 бит. При 18-ти битовом разрешении выходная частота ШИМ равняется 125,9 Гц (при тактовой частоте 33 МГц), а при 10-ти битовом разрешении частота выхода ШИМ равна 128.9 кГц. Скважность сигнала на выходе может варьироваться от 0% до 100%. На Рис. показана упрощенная блок-схема модуля ШИМ.

Регистры скважности имеют двойную буферезацию для работы без помех. На Рис. продемонстрировано появление помех для случая отсутствия двойной буферезации регистров скважности. Пунктирная линия показывает выход ШИМ, для случая отсутствия двойной буферезации. Если новое значение скважности записывается после того, как таймер прошел данное значение, ШИМ не сбрасывается в текущем цикле. В этом примере период ШИМ равен 50. прежнее значение скважности равно 30, новое значение равно 10.

Для включения «ШИМ 1» необходимо установить бит PWM1ON (TCON2<4>). Когда бит PWM1ON=1, вывод PD2/PWM1 становится выходом «ШИМ 1» и независимо от бита направления передачи данных (DDRD<2>) конфигурируется как выход. Если бит PWM1ON=0, то направление передачи данных вывода задается битом направления передачи (DDRD<2>). Подобным образом, бит PWM2ON (TCON2<5>) задает конфигурацию вывода PD3/PWM2.

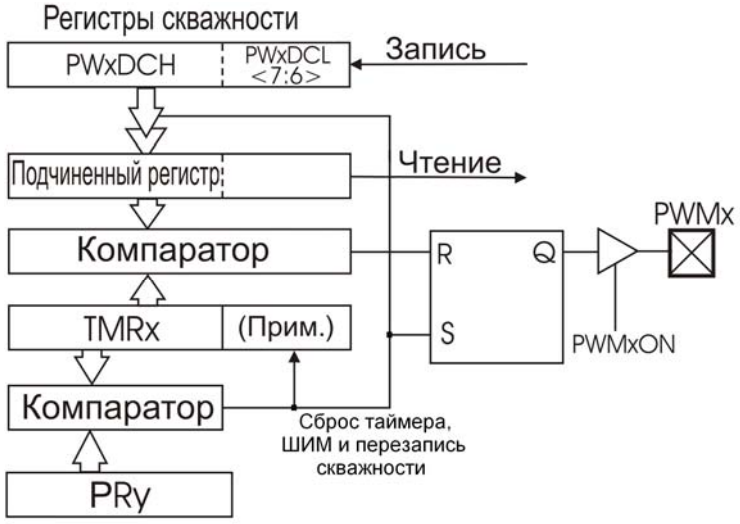


Рис. 25 Упрощенная блок-схема модуля ШИМ

Примечание:
16-ти разрядный таймер объединен с 2-х разрядным значением фазы Q для создания 18-ти разрядного значения.

Период выхода «ШИМ 1» определяется «таймером 1» и его регистром периода PR1H и PR1. Для «ШИМ 2» период также определяется «таймером 1» и PR1H и PR1. Периоды ШИМ могут высчитываться следующим образом:
период «ШИМ 1» = [(PR1H и PR1) + 1] * 4*T_C
период «ШИМ 2» = [(PR1H и PR1) + 1] * 4*T_C

Скважность ШИМ определяется 18-ти битным значением DCx<17:0>. Старшие 8 бит находятся в регистре PWxDCHH, а младшие 2 бита в регистре PWxDCL<7:6>. **Таблица 39** демонстрирует максимальную частоту ШИМ в зависимости от значения в регистре периода. Колличество битов разрешения, которого может достигнуть ШИМ, зависит от тактовой частоты микроконтроллера и частоты ШИМ.

Максимальное разрешение ШИМ (биты) для заданной частоты ШИМ: = log(F_C/F_{PWM}) / log(2),

где: F_{PWM} = 1/ период ШИМ.

Длительность импульса ШИМ = (DCx)*T_C (где DCx представляет 18-ти битное значение из PWxDCH:PWxDCL).

Если DCx = 0, тогда длительность импульса равена 0. Если PRx = PWxDCH, тогда выход ШИМ будет низким от одного до четырех тактов тактового генератора (в зависимости от состояния битов PWxDCL<7:6>). Чтобы скважность была 100%, значение PWxDCH должно быть больше, чем значение PRx.

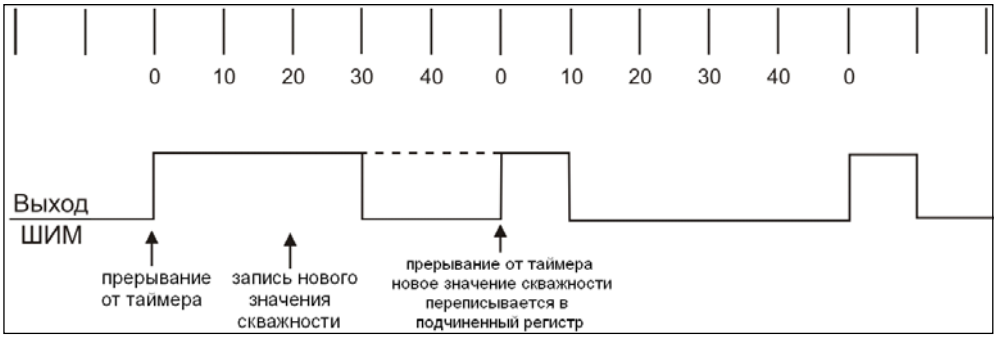


Рис. 26 Выход ШИМ

Регистры скважности для всех ШИМ имеют двойную буферизацию. При записи в регистры значение сохраняется в основных защелках, а при переполнении опорного таймера и начале нового периода ШИМ, значения из основной защелки переписываются в регистры подчиненной защелки, и вывод ШИМ переходит в высокий уровень.

Для регистров PW1DCH, PW1DCL, PW2DCH, PW2DCL операция записи записывает в «основные защелки», в то время как операция чтения считывает «подчиненные защелки». Поэтому желательно избегать операций типа чтение-модификация-запись с регистрами скважности.

Модули ШИМ используют прерывания от «таймера 1» (флаг запроса прерывания T1IF). Прерывание от таймера генерируется, когда значение регистра таймера совпадет со значением регистра периода и во время следующего приращения сбросится в 0. Это прерывание также отмечает начало цикла ШИМ. В этот момент можно записать новые значения скважности. Флаги запроса прерывания должны быть сброшены программно.

ШИМ может работать с внешним генератором тактовых импульсов, но при этом необходимо учесть ряд особенностей. Так как внешний тактовый сигнал с входа PD4/T1CLK синхронизируется с внутренним (выбирается один раз за командный цикл), то задержка между изменением сигнала T1CLK и увеличением таймера, будет изменяться в пределах T_{cy} (одного командного цикла). Это вызовет дрожание скважности и ошибку в периоде ШИМ. Дрожание будет приблизительно $+1T_{cy}$, если внешний генератор не синхронизирован с генератором процессора. При использовании внешнего источника тактовых импульсов для ШИМ, его частота должна быть много меньше, чем тактовая частота микроконтроллера (F_C).

Использование внешнего генератора тактовых импульсов для тактирования опорного таймера ШИМ (таймер 1) ограничивает максимальное разрешение ШИМ до 16 бит. Биты PWxDCL<7:6> должны быть сброшены. Использование любого другого значения вызовет искажение выхода ШИМ. Максимально достижимая частота ШИМ также более низкая. Максимальная

частота ШИМ, когда источником тактовых импульсов является вывод PD4/T1CLK, показана в **Таблица 39** (режим стандартного разрешения).

Таблица 39

Частота ШИМ (при тактовой частоте 33МГц)

Частота ШИМ	32.2 КГц	64.5 КГц	90.66 КГц	128.9 КГц	515.6 КГц
Значение PR	0xFF	0x7F	0x5A	0x3F	0x0F
Высокое разрешение	10 бит	9 бит	8.5 бит	8 бит	6 бит
Стандартное разрешение	8 бит	7 бит	6.5 бит	6 бит	4 бит

«Таймер 2»

«Таймер 2» является 16-ти разрядным таймером, состоящим из регистров TMR2H (старший байт) и TMR2 (младший байт). Таймер имеет 16-ти разрядный регистр периода, который может также быть 16-ти разрядным регистром захвата (регистрации событий) PR2H/CA1H:PR2L/CA1L. Таймер может тактироваться внутренними импульсами FC/4 (если бит TMR2CS (TCON1<2>) = 0), или внешними – задним фронтом (спадом) сигнала на выводе PD5/T2CLK (если TMR2CS=1). Для работы таймера должен быть установлен бит TMR2ON.

При внешнем тактировании таймера, сигнал с вывода PD5/T2CLK синхронизируется внутренними тактовыми импульсами дважды во время каждого цикла команды. Это вызывает задержку от момента появления заднего фронта на T2CLK до фактического приращения таймера. Рис. показывает временную диаграмму при работе таймера от внешнего генератора тактовых импульсов.

«Таймер 2» является 16-ти разрядным, поэтому необходимо осторожно считывать или записывать его во время работы, так как эти операции 8-ми разрядные. Лучше остановить таймер на время выполнения считывания/записи, а затем вновь запустить (используя бит TMR2ON). Однако, если нет возможности остановить таймер, можно использовать **Пример 7** для записи и **Пример 8** для считывания (во время данного процесса прерывания должны быть запрещены).
Таймер может работать в двух режимах (зависит от состояния бита CA1_PR2(TCON2<3>)):

- Режим одного входа захвата, таймер работает с регистром периода.
- Режим двух входов захвата.

Всего два 16-ти разрядных регистров захвата, которые захватывают 16-ти разрядное значение таймера, при фиксации события на выводах захвата. Есть два входа захвата PD0/CAP1, PD1/CAP2 по одному для каждой пары регистров захвата. Событиями захвата могут быть (определяется битами CAxED1 и CAxED0):

- Передний фронт сигнала на входе захвата.
- Задний фронт (спад).
- Каждый четвертый передний фронт.
- Каждый шестнадцатый передний фронт.

Каждый 16-ти разрядный регистр захвата имеет связанный с ним флаг запроса прерывания, который устанавливается, когда происходит захват. Модули захвата являются частью блока «таймера 2».

Режим одного входа захвата и регистра периода для таймера

Этот режим выбирается, если управляющий бит CA1_PR2=0. В режиме одного входа захвата регистры PR2H/CA1H и PR2L/CA1L составляют 16-ти разрядный регистр периода, и соответственно «захват 1» отключен и бит прерывания CAP1IF никогда не устанавливается. Блок-схема показана на Рис. . Таймер инкрементируется до тех пор, пока не сравняется с значением регистра периода, а затем сбрасывается в 0000h в следующем цикле приращения. При этом устанавливается флаг запроса прерывания от таймера (T2IF). Это прерывание может быть запрещено сбросом бита разрешения прерываний (T2IE). Флаг T2IF должен быть сброшен программно.

Биты CAxED1 и CAxED0 определяют событие, по которому произойдет захват. Когда происходит захват, устанавливается флаг запроса прерывания (CAPxIF). Прерывание будет разрешено, если бит маски CAPxIE установлен, бит разрешения прерывания от периферийного устройства (PEIE) установлен, а бит глобального запрета прерываний (GLINTD) должен быть сброшен. Флаг запроса прерывания CAPxIF сбрасывается программно.

При изменении выбраного предварительного делителя, содержимое делителя частоты не сбрасывается. Поэтому, первый захват после подобного изменения будет неопределенным. Однако последующие захваты будут верными. Предварительный делитель частоты сбрасывается сигналом «сброс».

При использовании вывода захвата PDx/CAPx в качестве выхода порта режим захвата не отключается. Можно просто отключить прерывание от захвата посредством сброса CAPxIE. Если вывод PDx/CAPx используется как выход, то можно активировать захват записью в порт. Это может быть полезным во время разработки для эмуляции прерывания от захвата.

Сигнал на входе захвата синхронизируется с внутренними тактовыми импульсами. Это накладывает определенные ограничения на форму входного сигнала.

Флаг переполнения имеет двойную буферезацию. Основной флаг устанавливается если одно захваченное значение уже находится в регистре захвата CAxH:CAxL, и произошло другое событие захвата на выводе PDx/CAPx. Новое значение не будет записываться в регистр захвата, защищая предыдущее несчитанное значение. При считывании старшего и младшего байта регистра захвата (в любом порядке), основной флаг переполнения передается в подчиненный флаг переполнения CAxOVF, и затем основной флаг сбрасывается.

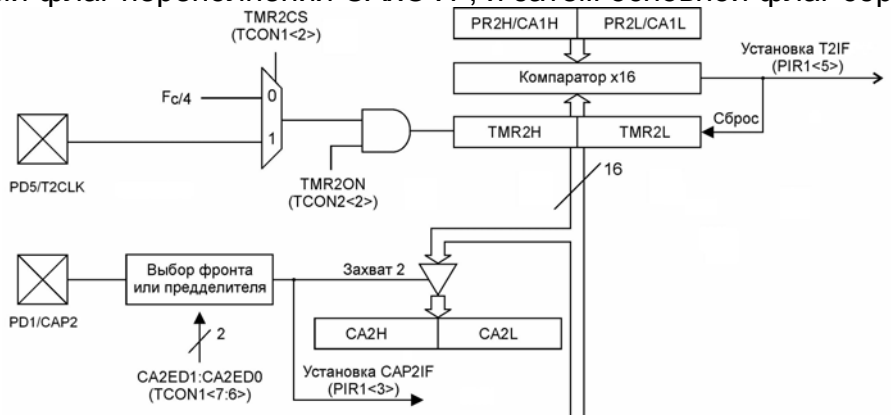


Рис. 27 Блок-схема «таймера 2»
в режиме с тремя входами захвата и регистром периода

После этого можно считать регистр TCONx для определения значения CAxOVF. Рекомендуемая последовательность для считывания регистров захвата и флагов переполнения захвата показана в **Пример 7**.

Пример 7

Последовательность для чтения регистров захвата

```
MOVLB  3                        ; выбрать банк 3
MOVPF  CA2L, LO_BYTE            ; считать младший байт регистра «захвата 2»
MOVPF  CA2H, HI_BYTE            ; считать старший байт регистра «захвата 2»
MOVPF  TCON2, STAT_VAL          ; считать регистр TCON2
```

Пример 8

Запись в «таймер 2»

```
BSF       CPUSTA,  GLINTD   ; отключить прерывания
MOVFP     RAM_L,   TMR2L
MOVFP     RAM_H,   TMR2H
BCF       CPUSTA,  GLINTD   ; сделано, активировать прерывания
```

Пример 9

Считывание из «таймера 2»

```
MOVPF     TMR2L,  TMPLO  ; считывать младший байт TMR2
MOVPF     TMR2H,  TMPHI  ; считывать старший байт TMR2
MOVFP     TMPLO,   WREG  ; младший байт в WREG
CPFSLT     TMR2L              ; сравнить TMR2L < WREG
RETURN               ; возврат
MOVPF     TMR2L,  TMPLO   ; считать младший байт TMR2
MOVPF     TMR2H,  TMPHI   ; считать старший байт TMR2
RETURN               ; возврат
```

Режим захвата задается битами CA1ED1 и CA1ED0. Все захваты работают аналогично (смотрите раздел «Режим одного входа захвата и регистра периода для таймера»).

Режим двух входов захвата

Этот режим выбирается если бит CA1_PR2=1(TCON2 <3>). Блок-схема показана на
Рис. . В данном режиме таймер работает без регистра периода, инкрементируясь от 0000h до FFFFh, и затем сбрасываясь в 0000h (с установкой флага запроса прерывания T2IF). Бит T2IF должен быть сброшен программно. Регистры PR2H/CA1H и PR2L/CA1L образуют 16-ти разрядный регистр «захвата 1». Соответствующий ему вход захвата - вывод PD0/CAP1.

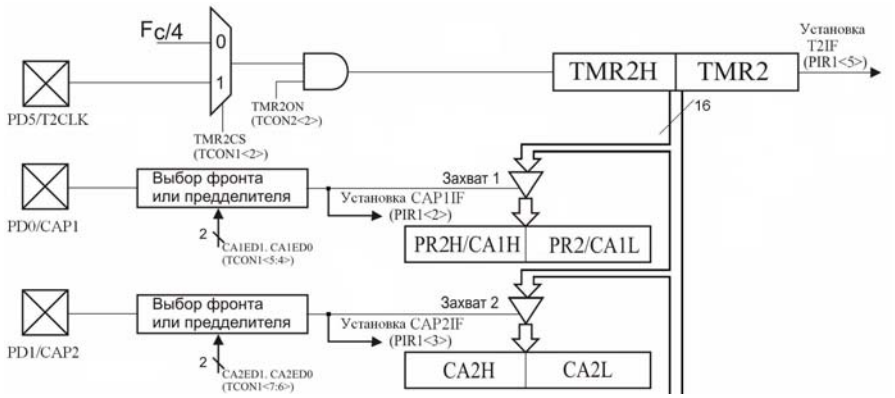


Рис. 28 Блок-схема «таймера 2» в режиме с четырьмя входами захвата.

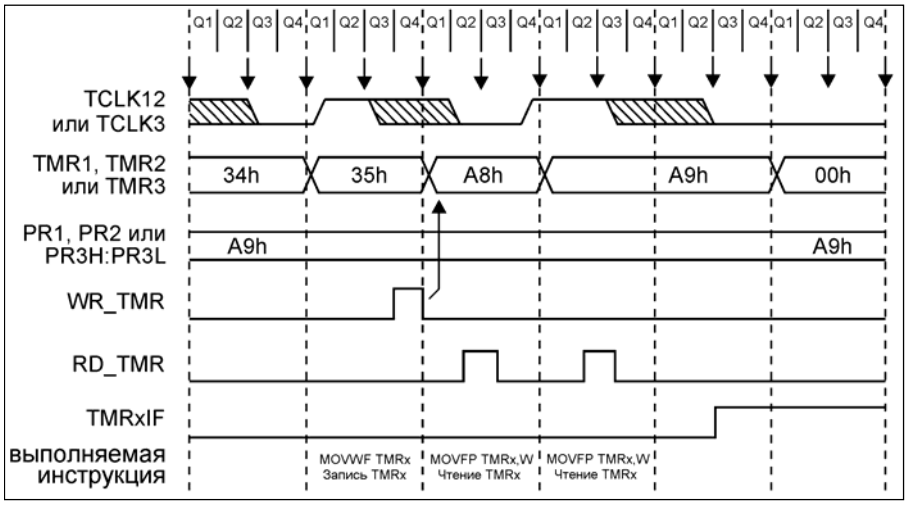


Рис. 29 Работа таймеров 1 и 2 от внешнего сигнала синхронизации

Примечания:

- T1CLK (T2CLK) выбирается в Q2 и Q4, «стрелка вниз» обозначает точку выборки.
- Задержка от спада сигнала на T1CLK (T2CLK) до приращения таймера от $2*T_c$ до $6*T_c$.

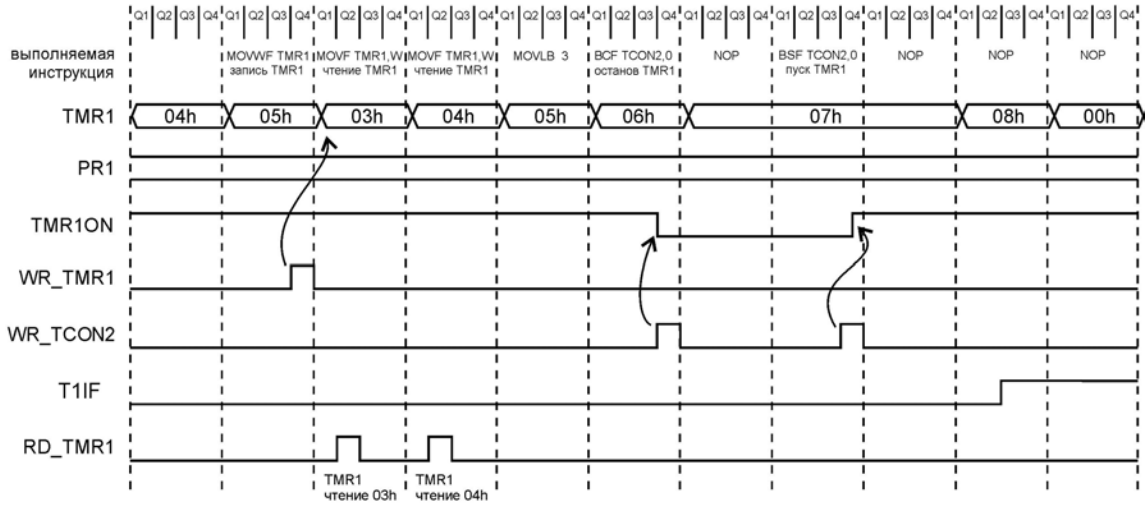


Рис. 30 Работа таймеров 1 и 2 от внутреннего сигнала синхронизации.

Таблица 40
Регистры блока

Адрес	Назва-ние	бит 7	бит 6	бит 5	бит 4	бит 3	бит 2	бит 1	бит 0	POR, BOR	MCLRn, WDT
Банк 2											
10h	TMR1	младший байт регистра таймера 1								0000 0000	0000 0000
11h	TMR1H	старший байт регистра таймера 1								0000 0000	0000 0000
12h	TMR2	младший байт регистра таймера 2								0000 0000	0000 0000
13h	TMR2H	старший байт регистра таймера 2								0000 0000	0000 0000
14h	PR1	младший байт регистра периода таймера 1								1111 1111	1111 1111
15h	PR1H	старший байт регистра периода таймера 1								0000 0000	0000 0000
16h	PR2/ CA1L	младший байт регистра периода таймера 2 / младший байт регистра захвата 1								0000 0000	0000 0000
17h	PR2H/ CA1H	старший байт регистра периода таймера 2 / старший байт регистра захвата 1								0000 0000	0000 0000
Банк 3											
10h	PW1DCL	PW1_DC1	PW1_DC0							0000 0000	0000 0000
11h	PW2DCL	PW2_DC1	PW2_DC0							0000 0000	0000 0000
12h	PW1DCH	PW1_DC9	PW1_DC8	PW1_DC7	PW1_DC6	PW1_DC5	PW1_DC4	PW1_DC3	PW1_DC2	0000 0000	0000 0000
13h	PW2DCH	PW2_DC9	PW2_DC8	PW2_DC7	PW2_DC6	PW2_DC5	PW2_DC4	PW2_DC3	PW2_DC2	0000 0000	0000 0000
14h	CA2L	младший байт регистра захвата 2								0000 0000	0000 0000
15h	CA2H	старший байт регистра захвата 2								0000 0000	0000 0000
16h	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	--	TMR2CS	--	TMR1CS	0000 0000	0000 0000
17h	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1_PR2	TMR2ON	--	TMR1ON	0000 0000	0000 0000
Банк 4											
10h	PW1DCHH	PW1_DC17	PW1_DC16	PW1_DC15	PW1_DC14	PW1_DC13	PW1_DC12	PW1_DC11	PW1_DC10	0000 0000	0000 0000
11h	PW2DCHH	PW2_DC17	PW2_DC16	PW2_DC15	PW2_DC14	PW2_DC13	PW2_DC12	PW2_DC11	PW2_DC10	0000 0000	0000 0000

Модуль универсального синхронно-асинхронных приемопередачика с поддержкой LIN интерфейса

Микроконтроллер содержит один модуль синхронно-асинхронных приемопередатчика с поддержкой LIN интерфейса USART/LIN1 и USART2/LIN2.

Универсальный синхронно-асинхронный приемопередатчик может работать в следующих режимах:

- асинхронный (полный дуплекс);
- асинхронный (полный дуплекс) с автоматическим приемом LIN заголовка;
- синхронный ведущий (полудуплекс);
- синхронный ведомый (полудуплекс).

Бит SPEN (RCSTA1<7>) должен быть установлен, чтобы выводы PAx/RXx/DTx и PAx/TXx/CKx сконфигурировались как выводы последовательного интерфейса. Модуль USARTx/LINx будет управлять направлением выводов PAx/RXx/DTx и PAx/TXx/CKx, в зависимости от состояния битов конфигурации в регистрах RCSTAx и TXSTAx. Следующие биты контролируют направление выводов: SPENx, TXENx, SRENx, CRENx, CSRCx. При установки бита LINx_EN в асинхронном режиме работы, модуль автоматически принимает заголовок пакета (LINx_BRK и LINx_SYNCH) с вычислением скорости передачи.

Регистр режима и статуса работы приемника

Таблица 41

RCSTA1. ADR = 0x13, Банк доступа BANK = 0x00
RCSTA2. ADR = 0x13, Банк доступа BANK = 0x07

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	U	RO	RO	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	SPEN	RX9	SREN	CREN	-	FERRx	OERRx	RX9Dx

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 42

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	SPENx	бит разрешения работы последовательного порта 1 = конфигурирует PAx/TXx/CKx и PAx/RXx/DTx как выводы последовательного порта USARTx 0 = последовательный порт отключен
6	RX9x	выбор 9-ти разрядного приема 1 = выбирает 9-ти разрядный прием 0 = выбирает 8-ми разрядный прием
5	SRENx	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
4	CRENx	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
3	-	Не реализовано, читается как «0»
2	FERRx	бит ошибки кадрирования 1 = есть ошибка (сбрасывается при чтении регистра RCREG) 0 = нет ошибки
1	OERRx	бит ошибки переполнения внутреннего буфера. 1 = есть ошибка (сбрасывается при сбросе бита CREN) 0 = нет ошибки
0	RX9Dx	9-й бит принятых данных (может использоваться для программной реализации проверки четности)

Регистр режима и статуса работы передатчика

Таблица 43

TXSTA1. ADR = 0x15, Банк доступа BANK = 0x00
TXSTA2. ADR = 0x15, Банк доступа BANK = 0x07

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	U	U	RO	R/W
Значение после сброса**	0	0	0	0	0	0	1	X
	CSRCx	TX9x	TXENx	SYNCx	-	-	TRMTx	TX9Dx

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 44

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	CSRCx	выбор источника тактовых импульсов *Синхронный режим:* 1 = режим ведущего (внутренний тактовый сигнал из BRG) 0 = режим ведомого (внешнего источника тактового сигнала) *Асинхронный режим:* Не имеет значения
6	TX9x	выбор 9-ти разрядной передачи. 1 = выбирает 9-ти разрядную передачу 0 = выбирает 8-ми разрядную передачу
5	TXENx	разрешение передачи 1 = передача разрешена 0 = передача отключена Бит SREN/CREN подменяет TXEN в синхронном режиме.
4	SYNCx	бит выбора режима USART (синхронный/асинхронный) 1 = синхронный режим 0 = асинхронный режим
3..2	-	Не реализованы, читаются как «0»
1	TRMTx	флаг заполненности сдвигового регистра передатчика (TSR) 1 = регистр пуст 0 = регистр заполнен
0	TX9Dx	9-й бит передаваемых данных (может использоваться для программной реализации проверки четности)

Регистр режима и статуса работы приемника LIN заголовка

Таблица 45

LIN1_CNTR. ADR = 0x11, Банк доступа BANK = 0x00
LIN2_CNTR. ADR = 0x11, Банк доступа BANK = 0x07

Номер	7	6	5	4	3	2	1	0
Доступ*	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	LINx_BRKCNT3	LINx_BRKCNT2	LINx_BRKCNT1	LINx_BRKCNT0	LINx_BRK	LINx_SYNCH	LINx_ERR	LINx_EN

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 46

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..4	LINx_BRKCNT[3:0]	Длительность принятого поля BREAK, При приеме поля BREAK короче, чем 11 битовых интервалов вырабатывается сигнал ошибки (ERR)
3	LINx_BRK	Флаг окончания приема поля BREAK 1 = BREAK получен 0 = не получен
2	LINx_SYNCH	Флаг окончания приема поля SYNCH 1 = SYNCH получен 0 = не получен
1	LINx_ERR	Флаг возникновения ошибки при приеме заголовка LIN пакета. 1 = есть ошибка 0 = нет ошибки
0	LINx_EN	Сигнал разрешения работы приемника LIN заголовка Асинхронный режим: 1 = принимает заголовок LIN пакета 0 = отключен Синхронный режим: Не важно.

Регистр данных приемника

Таблица 47

RCREG1. ADR = 0x14, Банк доступа BANK = 0x00
RCREG2. ADR = 0x14, Банк доступа BANK = 0x07

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	DATA 7	DATA 6	DATA 5	DATA 4	DATA 3	DATA 2	DATA 1	DATA 0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 48

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	DATA[7:0]	Регистр данных: Данные полученные по USART.

Регистр данных передатчика

Таблица 49

TXREG1. ADR = 0x16, Банк доступа BANK = 0x00
TXREG2. ADR = 0x16, Банк доступа BANK = 0x07

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	DATA 7	DATA 6	DATA 5	DATA 4	DATA 3	DATA 2	DATA 1	DATA 0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 50

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	DATA[7:0]	Регистр данных: Данные полученные по USART.

Регистр задания скорости приема и передачи

Таблица 51

SPBRG. ADR = 0x17, Банк доступа BANK = 0x00

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реалиан.

Таблица 52

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	BRG[7:0]	SPBRG регистр определяет период (скорость) работы приемопередатчика

Регистр скорости поля SYNCH в LIN фрейме

Таблица 53

LIN1_BRG. ADR = 0x12, Банк доступа BANK = 0x00
LIN2_BRG. ADR = 0x12, Банк доступа BANK = 0x00

Номер	7	6	5	4	3	2	1	0
Доступ*	RO	RO	RO	RO	RO	RO	RO	RO
Значение после сброса**	0	0	0	0	0	0	0	0
	LINx_ BRG7	LINx_ BRG6	LINx_ BRG5	LINx_ BRG4	LINx_ BRG3	LINx_ BRG2	LINx_ BRG1	LINx_ BRG0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 54

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	LINx_BRG[7:0]	LINx_BRG регистр содержит значение скорости определенное при приеме поля SYNCH заголовка LIN фрейма

Генератор скорости передачи данных

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

- скорость передачи для асинхронного режима (в том числе LIN) = FOSC / (32*((SPBRGx)+1)), значение (SPBRGx) от 0 до 255;
- скорость передачи для синхронного режима = FOSC / (4*((SPBRGx)+1)), значение (SPBRGx) от 0 до 255.

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

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 (TXSTA1<4>).

Модуль USART в асинхронном режиме состоит из следующих компонентов:

- Генератор скорости передачи.
- Схема выборки.
- Асинхронный приемник.
- Асинхронный передатчик.
- Детектор LIN заголовка.

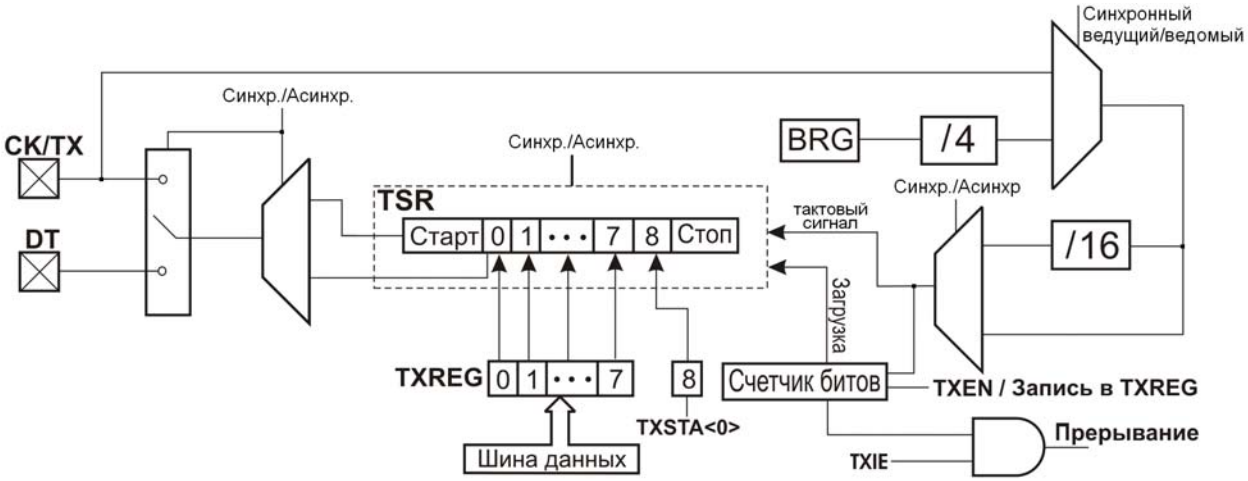


Рис. 31 Блок-схема передатчика

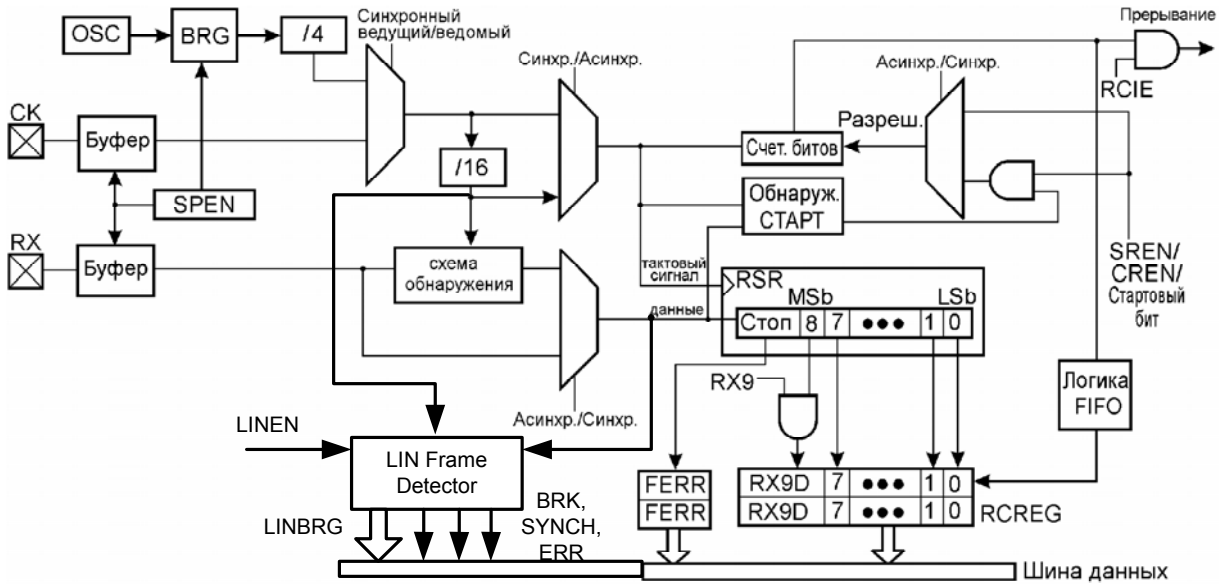


Рис. 32 Блок-схема приемника

Асинхронный передатчик

Блок-схема передатчика представлена на
Рис. . Основой передатчика является сдвиговый регистр передачи (TSR). Он получает информацию из буфера передатчика (TXREGx). В буфер TXREGx данные загружаются программно. Сдвиговый регистр TSR загружается новыми данными из TXREGx (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 TXxIE. Флаг запроса прерывания TXxIF устанавливается независимо от значения TXIE. Он не может быть сброшен программно. Флаг сбрасывается аппаратно при загрузке новых данных в TXREGx. Бит TRMTx (TXSTAx<1>) индицирует состояние сдвигового регистра TSR. Бит устанавливается когда TSR пуст. TRMTx 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.). 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.). Сброс TXEN во время передачи вызовет отмену передачи, сброс передатчика.

Для выбора 9-ти битной передачи, бит TX9x (TXSTAx<6>) должен быть установлен в единицу. Девятое значение бита записывается в TX9Dx (TXSTAx<0>) перед записью 8-ми битных данных в TXREGx, так как запись

данных в TXREGx может вызвать немедленную передачу данных в TSR (если TSR пуст).

Шаги для настройки асинхронной передачи следующие:

- записать значение в регистр SPBRGx для задания скорости передачи.
- включить асинхронный последовательный порт (бит SYNCx=0 и бит SPENx=1).
- если требуются прерывания, тогда установите бит TXxIE.
- если требуется 9-ти битная передача, тогда установите бит TX9x.
- если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9Dx.
- загрузите данные в регистр TXREGx.
- запустите передачу установкой бита TXENx.

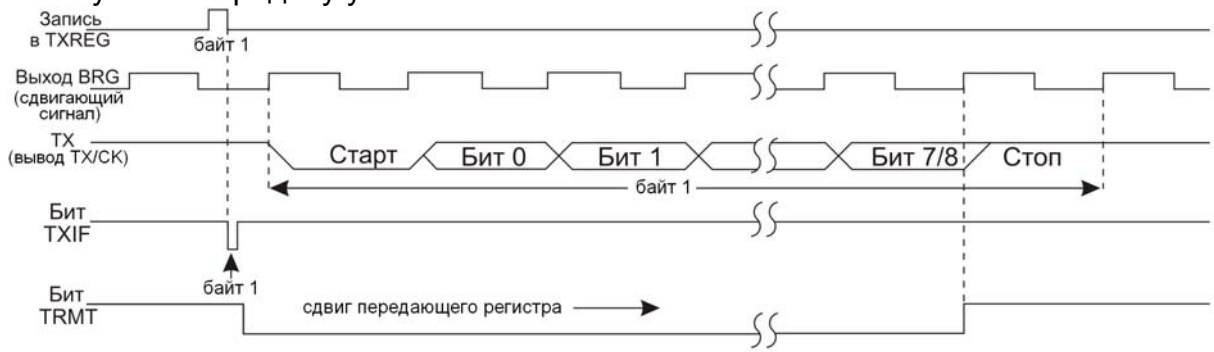


Рис. 33 Асинхронная передача

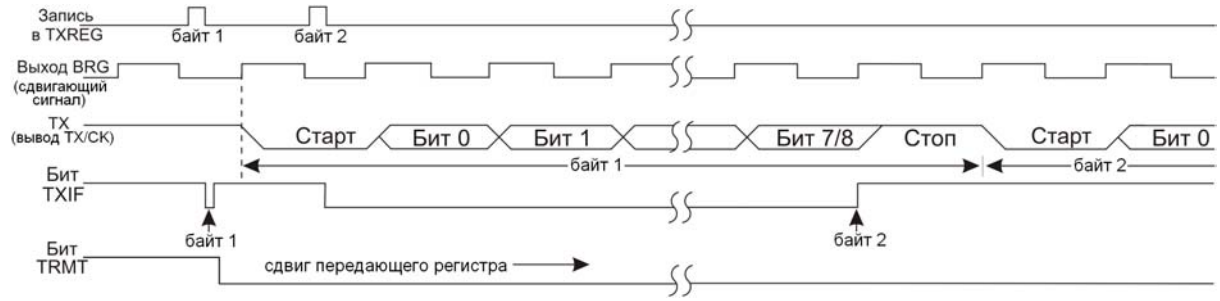


Рис. 34 Асинхронная неразрывная последовательная передача

Асинхронный приемник

Блок-схема приемника показана на
Рис. . 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 CRENx (RCSTAx<4>).

Основой приемника является сдвиговый регистр приема (RSR). После обнаружения стопового бита, принятые данные из RSR передаются в RCREGx (если он пуст), после чего устанавливается флаг запроса прерывания RCxIF. Прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита RCxIE. RCxIF доступен только для чтения, он сбрасывается аппаратно когда считываются данные из RCREGx и регистр пуст. Регистр

RCREGx имеет двойную буферизацию, т.е. можно принять два байта данных в RCREGx FIFO и третий байт начать принимать в RSR. При обнаружении стопового бита третьего байта, если RCREGx по-прежнему не считан, устанавливается бит ошибки переполнения приемника OERRx (RCSTA<1>). Данные в RSR будут потеряны. Для извлечения двух байт RCREGx должен считываться дважды. Бит OERRx должен быть очищен программно сбросом приема (сбросом бита CRENx). Пока бит OERRx установлен, приемник не работает. Флаг ошибки кадрирования FERRx (RCSTAx<2>) устанавливается, если невозможно обнаружить стоповый бит. Флаг FERR и девятый бит данных буферизуются также как и принятые данные. Поэтому необходимо считать регистр RCSTAx перед считыванием RCREGx, чтобы не потерять прежнюю информацию FERRx и RX9Dx.

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

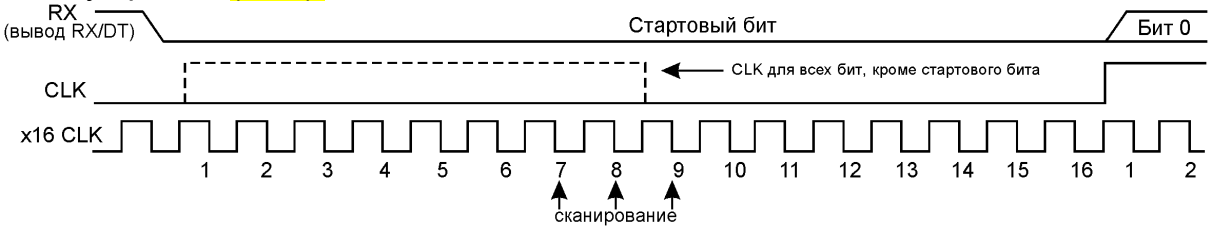


Рис.35 Схема сканирования вывода PAx/RXx/DTx

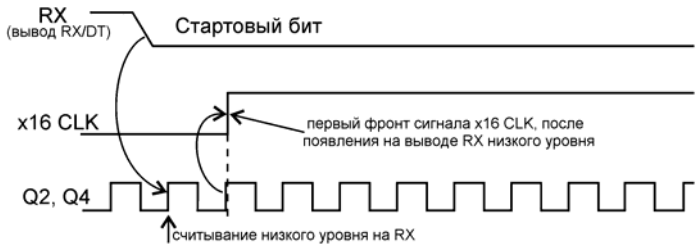


Рис. 36 Обнаружение стартового бита

Шаги для настройки асинхронного приема следующие:

- записать значение в регистр SPBRGx для задания скорости приема;
- включить асинхронный последовательный порт (бит SYNCx=0 и бит SPENx=1);
- если требуются прерывания, тогда установите бит RCxIE;
- если требуется 9-ти битный прием, тогда установите бит RX9x;
- разрешите прием установкой бита CRENx;
- при завершении приема установится бит RCxIF, и произойдет прерывание (если установлен бит RCxIE);
- считайте RCSTAx для получения значения девятого бита (для 9-ти битного приема) и бит FERRx для определения ошибки;
- считайте 8-ми битные принятые данные из регистра RCREGx;
- если произошла ошибка переполнения, сбросте бит OERRx.

Для отмены приема, сбросте либо биты SRENx и CRENx, либо бит SPENx. Это сбросит логику приема, но не изменит настройки.

Режим автоматического приема LIN заголовка

При работе в асинхронном режиме модуль USART может автоматически принимать и распознавать заголовок LIN фрейма. Для этого необходимо разрешить работу блока LIN Frame Detector установкой бита LINx_EN. Прием поля BREAK производится на основании скорости передачи задаваемой в регистре SPBRx. При приеме поля BREAK вырабатывается прерывание RCxIF и выставляется флаг LINx_BRK. В регистре LINx_CNTR биты LINx_BRKCNT<3:0> содержат длительность принятого поля BREAK. Если длительность поля BREAK меньше 11, автоматически вырабатывается флаг возникновения ошибки LINx_ERR. Максимальное значение BRKCNT – 16, независимо от реальной длительности поля BREAK. Для сброса сигнала прерывание RCxIF после приема поля BREAK в бит LINx_BRK регистр LIN_CNTR необходимо записать «0». При этом прерывание снимается, но значение бита не изменяется.

После приема поля BREAK автоматически принимается поле SYNCH. При приеме поля SYNCH автоматически вычисляется скорость передачи. При приеме поля SYNCH вырабатывается прерывание RCxIF и выставляется флаг LINx_SYNCH. В регистре LINx_BRG содержится скорость передачи поля SYNCH. В случае необходимости эта скорость может быть использована для задания скорости передачи SPBRGx для всего блока USARTx.

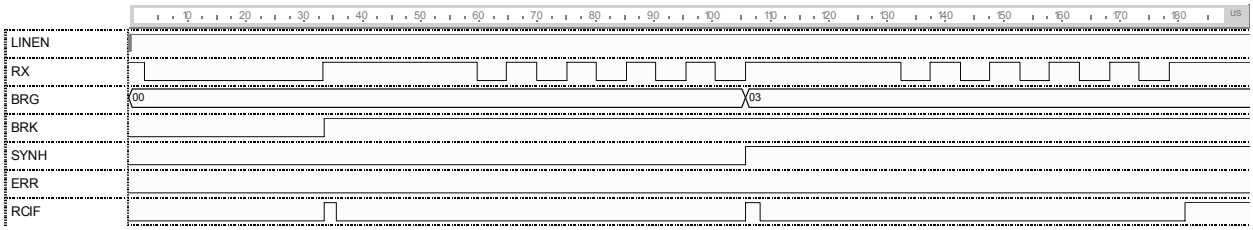


Рис. 37 Автоматический прием LIN заголовка

Для сброса сигнала прерывание RCxIF после приема поля LINx_SYNCH в бит LINx_SYNCH регистр LIN_CNTR необходимо записать «0». При этом прерывание снимается, но значение бита не изменяется. В случае если при приеме поля SYNCH распознаваемая скорость будет ниже, чем скорость при SPBRGx == 0xFF, автоматически вырабатывается флаг возникновения ошибки LINx_ERR. После приема поля SYNCH модуль USARTx переходит в обычный режим работы в асинхронном режиме, все последующие принимаемые данные отображаются в регистре RCREGx. После приема всего LIN фрейма для подготовки к приему следующего необходимо сбросить блок LIN Frame Detector. Для этого необходимо установить бит LINx_EN сначала в «0», а затем в «1».

Передача LIN фрейма

Формирование LIN фрейма осуществляется программным путем. Для передачи поля BREAK в регистр SPBRGx должна быть записана скорость меньшая чем реальна скорость передачи, таким образом, что бы при отправке байта 0x00 приемник воспринял минимум 11 битов равных 0 (с учетом старт бита). После передачи поля BREAK в регистре SPBRGx задается нормальная

скорость работы и для отправки поля SYNCH отправляется байт 0x55. Затем остальные байты фрейма.

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 SYNCx (TXSTAx<4>). Бит SPENx (RCSTAx<7>) устанавливается, чтобы сконфигурировать выводы: PAx/TXx/CKx - линия тактовых импульсов и PAx/RXx/DTx - линия данных. Ведущий режим означает, что процессор формирует тактовые импульсы на линии PAx/TXx/CKx. Ведущий режим выбирается установкой бита CSRCx (TXSTAx<7>).

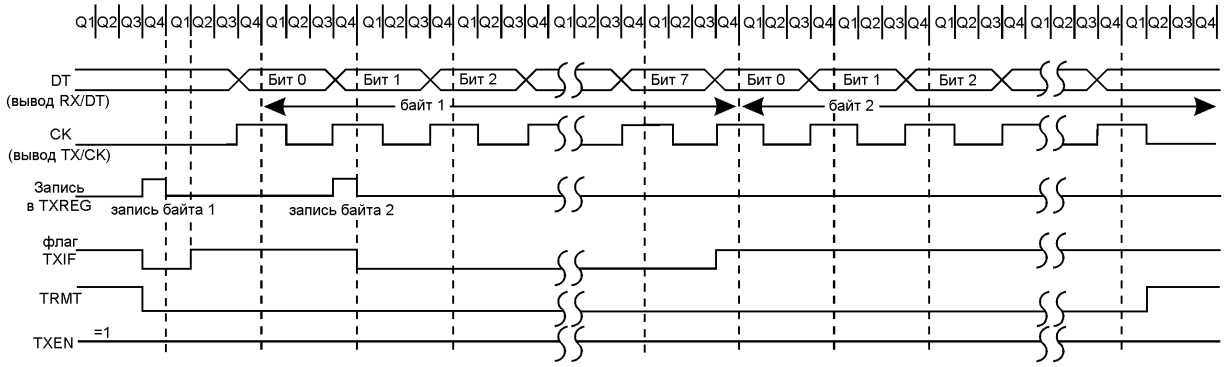


Рис. 38 Синхронная передача в ведущем режиме

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 TXREGx. Данные загружаются в TXREGx программно. После передачи последнего бита предыдущей посылки, TSR загружается новыми данными из TXREGx (если они есть). Это происходит в последнем командном цикле периода BRG. При этом устанавливается флаг запроса прерывания TXxIF индицирующий, что TXREGx пуст. Это прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита TXxIE. Флаг запроса прерывания TXxIF устанавливается независимо от значения TXxIE. Он не может быть сброшен программно. Флаг сбрасывается аппаратно при загрузке новых данных в TXREGx. Бит TRMTx (TXSTAx<1>) индицирует состояние сдвигового регистра TSR. Бит устанавливается когда TSR пуст. TRMTx 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 TXENx (TXSTAx<5>). Фактически передача не начнется пока данные не будут загружены в TXREGx. Первый бит данных появится на первом переднем фронте тактовых импульсов с вывода PAx/TXx/CKx. Данные стабилизируются по заднему фронту тактовых

импульсов (**Рис. 39**). Передачу также можно начать сначала загрузив TXREGx, а потом установив бит TXENx. Это удобно, когда выбраны низкие скорости передачи. Генератор BRG остановлен, когда биты TXENx, CRENx, SRENx сброшены. Установка бита TXENx запустит генератор BRG, который сразу же выдаст сдвиговые тактовые импульсы. Обычно, при первом разрешении передачи регистр TSR пуст, поэтому записанные в TXREGx данные будут сразу переданы в TSR, опустошая TXREGx. Поэтому возможна неразрывная последовательная передача данных. Сброс TXENx во время передачи вызовет отмену передачи, сброс передатчика и переведет выводы PAx/RXx/DTx и PAx/TXx/CKx в третье состояние. Если во время передачи будут установлены биты CRENx и SRENx, передача будет отменена, и вывод PAx/RXx/DTx вернется в третье состояние (для приема). Вывод PAx/TXx/CKx останется выходом, если установлен бит CSRCx (внутренний источник тактовых импульсов от BRG). Логика передатчика не сбрасывается, хотя и отсоединяется от выводов. Чтобы сбросить передатчик - необходимо сбросить бит TXENx. Если бит SRENx установлен для прерывания осуществляемой передачи и получения одного байта, то после получения одного байта SRENx сбросится и последовательный порт снова вернется к передаче, так как бит TXENx попрежнему установлен. Линия PAx/RXx/DTx сразу же будет переключена из третьего состояния в режим выхода. Для избежания этого, TXENx должен быть сброшен.

Чтобы выбрать 9-ти битную передачу, необходимо установить бит TX9x (TXSTAx<6>). Девятый бит должен записываться (в TX9Dx (TXSTAx<0>)) до записи 8-ми битных данных в TXREGx, так как запись данных в TXREGx может вызвать немедленную передачу данных в TSR (если TSR пуст).

Шаги для настройки передачи в синхронном ведущем режиме:

- записать значение в регистр SPBRGx для задания скорости передачи;
- включить синхронный последовательный порт в ведущем режиме (биты SYNCx=1, SPENx=1 и CSRCx=1);
- проверить что биты CRENx и SRENx сброшены, если эти биты установлены, то они отменяют передачу;
- если требуются прерывания, тогда установите бит TXxIE;
- если требуется 9-ти битная передача, тогда установите бит TX9x;
- если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9Dx;
- загрузите данные в регистр TXREGx;
- запустите передачу установкой бита TXENx;

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

Прием данных в синхронном ведущем режиме

Если выбран синхронный режим, прием разрешается установкой бита SRENx (RCSTAx<5>) или бита CRENx (RCSTAx<4>). Данные с вывода PAx/RXx/DTx опрашиваются на заднем фронте (спаде) тактовых импульсов. Если установлен SRENx, то принимается только один байт. Если установлен CRENx, то прием продолжается пока CRENx не сбросится. Если установлены оба бита, то CRENx имеет приоритет. После приема последнего бита полученные данные из сдвигового регистра приема (RSR) передаются в RCREGx (если он пуст), и

устанавливается флаг запроса прерывания RCxIF. Прерывание может быть разрешено или запрещено соответственно установкой или сбросом бита RCxIE. RCxIF доступен только для чтения, и сбрасывается аппаратно, когда RCREGx считан и пуст.
Регистр RCREGx имеет двойную буферизацию, т.е. можно принять два байта данных в RCREGx FIFO и третий байт начать принимать в RSR. При приеме последнего бита третьего байта, если RCREGx по-прежнему не считан, устанавливается бит ошибки переполнения приемника OERRx (RCSTAx<1>). Данные в RSR будут потеряны. Для извлечения двух байт RCREGx должен считываться дважды. Бит OERRx должен быть очищен программно сбросом приема (сбросом бита CRENx). Пока бит OERRx установлен, приемник не работает. Девятый бит данных буферизуется также как и принятые данные. Поэтому необходимо считать регистр RCSTAx перед считыванием RCREGx, чтобы не потерять прежнее значение RX9Dx.

Шаги для настройки приема в синхронном ведущем режиме:

- записать значение в регистр SPBRG для задания скорости приема;
- включить синхронный последовательный порт в ведущем режиме (биты SYNCx=1, SPENx=1 и CSRCx=1).
- если требуются прерывания, тогда установите бит RCxI;.
- если требуется 9-ти битный прием, тогда установите бит RX9x;
- если требуется прием единичного байта установите бит SRENx, для продолжительного приема установите бит CRENx;
- при завершении приема установится бит RCxIF, и произойдет прерывание (если установлен бит RCxIE);
- считайте RCSTA для получения значения девятого бита (для 9-ти битного приема) и бит определения ошибки;
- считайте 8-ми битные принятые данные из регистра RCREGx;
- если произошла ошибка переполнения, сбросьте бит OERRx.
- Для отмены приема, сбросьте либо биты SRENx и CRENx, либо бит SPENx. Это сбросит логику приема, но не изменит настройки.

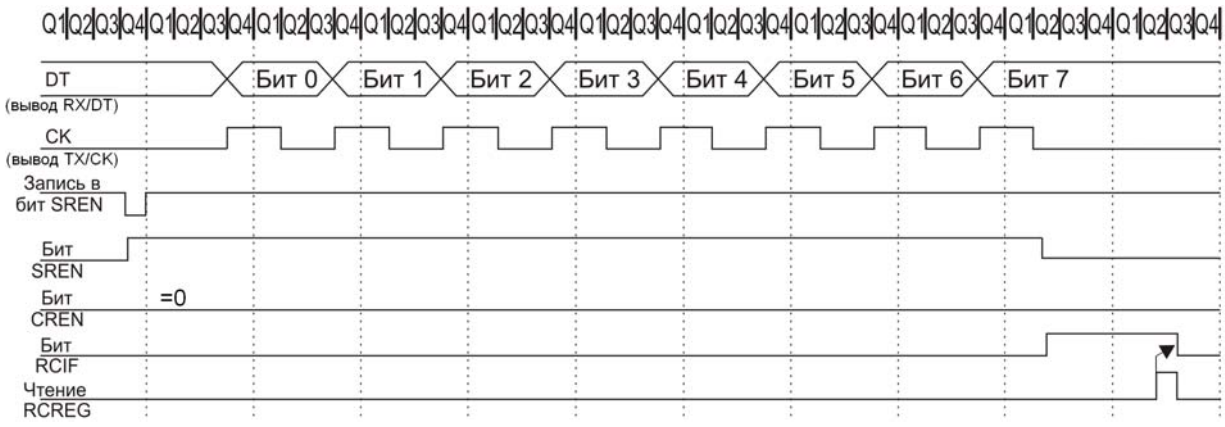


Рис. 39 Синхронный прием

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 CSRCx (TXSTAx<7>).

Передача данных в синхронном ведомом режиме

Работа ведущего и ведомого режимов идентична, за исключением работы в режиме SLEEP. Если 2 байта записываются в TXREGx и затем исполняется команда SLEEP, то произойдет следующее. Первый байт сразу переносится в TSR и будет передаваться по тактовым импульсам. Второй байт останется в TXREGx. Флаг TXxIF не будет установлен. Когда закончится передача первого байта, второй байт будет перенесен из TXREGx в TSR и установится флаг TXxIF. Если TXxIE=1, прерывание выведет микроконтроллер из режима SLEEP, и если разрешено переферийное прерывание, тогда программа переходит к вектору прерывания (0020h).

Шаги для настройки передачи в синхронном ведомом режиме:

- записать значение в регистр SPBRG для задания скорости передачи;
- включить синхронный последовательный порт в ведомом режиме (биты SYNCx=1, SPENx=1 и CSRCx=0);
- сбросить бит CRENx;
- если требуются прерывания, тогда установите бит TXxIE;
- если требуется 9-ти битная передача, тогда установите бит TX9x;
- если выбрана 9-ти битная передача, девятый бит должен быть загружен в TX9Dx;
- загрузите данные в регистр TXREGx;
- запустите передачу установкой бита TXENx.

Для отмены передачи необходимо сбросить бит SPENx или бит TXENx. 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 SRENx.

Если прием разрешен (CRENx=1) до команды SLEEP, то данные могут быть получены в режиме SLEEP. При завершении приема, данные из RSR передаются в RCREGx и устанавливается флаг запроса прерывания RCxIF, а если бит RCxIE=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:

- записать значение в регистр SPBRGx для задания скорости приема;
- включить синхронный последовательный порт в ведомом режиме (биты SYNCx=1, SPENx=1 и CSRCx=0);
- если требуются прерывания, тогда установите бит RCxIE;
- если требуется 9-ти битный прием, тогда установите бит RX9x;
- для разрешения приема установите бит CRENx;
- при завершении приема установится бит RCIF, и произойдет прерывание (если установлен бит RCxIE);
- считайте RCSTA для получения значения девятого бита (для 9-ти битного приема) и бит определения ошибки;
- считайте 8-ми битные принятые данные из регистра RCREGx;

- если произошла ошибка переполнения, сбросте бит OERRx.

Для отмены приема, сбросте либо бит CRENx, либо бит SPENx. Это сбросит логику приема, но не изменит настройки.

Аналогово-цифровой преобразователь

Аналогово-цифровой преобразователь (АЦП) имеет 8 аналоговых входов. Входной аналоговый сигнал через коммутатор каналов поступает в модуль АЦП. Модуль АЦП преобразует напряжение методом последовательного приближения. Результат преобразования 12-ти разрядный. Положительный и отрицательный входы опорного напряжения могут быть выбраны программно или с выводов AUCC и AUSS, или с выводов PC0/AN0/UREF+ и PC1/AN1/UREF-. АЦП может работать в спящем режиме микроконтроллера, при этом в качестве источника тактовых импульсов для АЦП должен быть выбран RC генератор. Модуль АЦП имеет 4 регистра:
регистр результата, старший байт (ADRESH);
регистр результата, младший байт (ADRESL);
регистр управления 0 (ADCON0);
регистр управления 1 (ADCON1).

Регистр ADCON0 используется для настройки работы модуля АЦП. С помощью регистра ADCON1 устанавливается, какие входы микроконтроллера будут использоваться модулем АЦП и в каком режиме. Входы настраиваются в качестве аналоговых входов (PC0/AN0/UREF+ и PC1/AN1/UREF- могут быть входами опорного напряжения) или цифровых входов/выходов.

Таблица 55

ADCON0.ADR = 0x14, банк доступа BANK = 0x06

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0	R/W-0
---	**CHS2**	**CHS1**	**CHS0**	**—**	**GO_DONE**	**—**	**ADON**
бит 7	6	5	4	3	2	1	бит 0

бит 7	Не используется, читается как «0».
бит 6-4	**CHS2 - CHS0:** биты выбора аналогового канала: 000 = канал 0, (AN0) 001 = канал 1, (AN1) 010 = канал 2, (AN2) 011 = канал 3, (AN3) 100 = канал 4, (AN4) 101 = канал 5, (AN5) 110 = канал 6, (AN6) 111 = канал 7, (AN7)
бит 3	Не используется, читается как «0».
бит 2	**GO_DONE:** бит состояния модуля АЦП. Если ADON = 1, т.е. АЦП включен: 1 = модуль АЦП выполняет преобразование (установка бита вызывает начало преобразования, аппаратно сбрасывается по завершению преобразования), 0 = в модуле АЦП преобразования нет.
бит 1	Не используется, читается как «0».

бит 0	**ADON:** включение модуля АЦП. 1 = модуль АЦП включен, 0 = модуль АЦП выключен (снижает общий ток потребления микроконтроллера).
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 56

ADCON1.ADR = 0x15, банк доступа BANK = 0x06

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS1	**ADCS0**	**ADFM**	**—**	**PCFG3**	**PCFG2**	**PCFG1**	**PCFG0**
бит 7	6	5	4	3	2	1	бит 0

бит 7, 6	**ADCS1:ADCS0:** выбор источника импульсов преобразования модуля АЦП: 00 = $F_C/8$ 01 = $F_C/32$ 10 = $F_C/64$ 11 = источник импульсов внутренний RC генератор (F_{RC})
бит 5	**ADFM:** Формат сохранения 12-битного результата: 1 = правое выравнивание, 4 старших бит ADRESH читаются как «0». 0 = левое выравнивание, 4 младших бит ADRESL читаются как «0».
бит 4	Не используется, читается как «0».
бит 3-1	**PCFG3:PCFG1:** биты управления конфигурацией порта АЦП.
бит 0	**PCFG0:** Выбор источника опорного напряжения: 1 = опорное напряжение берется с выводов U_{REF+} и U_{REF-}. 0 = опорное напряжение берется с выводов AU_{DD} и AU_{SS}.
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 57

PCFG3:PCFG1	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
000	**A**	**A**	**A**	**A**	**A**	**A**	**A**	**A**
001	**D**	**A**	**A**	**A**	**A**	**A**	**A**	**A**
010	**D**	**D**	**A**	**A**	**A**	**A**	**A**	**A**
011	**D**	**D**	**D**	**A**	**A**	**A**	**A**	**A**
100	**D**	**D**	**D**	**D**	**A**	**A**	**A**	**A**
101	**D**	**D**	**D**	**D**	**D**	**A**	**A**	**A**
110	**D**	**D**	**D**	**D**	**D**	**D**	**A**	**A**
111	**D**	**D**	**D**	**D**	**D**	**D**	**D**	**D**
Обозначения: A - аналоговый вход, D - цифровой вход/выход.								

Когда преобразование завершено, 12-ти разрядный результат преобразования записывается в регистры ADRESH:ADRESL, бит GO_DONE (ADCON0<2>) сбрасывается и устанавливается флаг запроса прерывания ADCIF. Блок-схема модуля АЦП показана на **Рис. 41 и 45**.

Рекомендованная последовательность действий для работы с АЦП:

1. Настроить модуль АЦП: выбрать аналоговые входы, источник опорного напряжения, цифровые входы/выходы (ADCON1); выбрать источник импульсов преобразования (ADCON1); выбрать входной канал АЦП (ADCON0); включить модуль АЦП (ADCON0). С помощью регистров DDR все выводы, которые будут использоваться как аналоговые входы, должны быть настроены на вход.

2. Настроить прерывание от модуля АЦП (если необходимо): сбросить флаг запроса прерывания по окончанию преобразования ADCIF; установить бит ADCIE (маска прерывания от АЦП); сбросить бит GLINTD.

3. Выдержать паузу, необходимую для зарядки конденсатора C_{HOLD} (до уровня входного напряжения).

4. Начать аналогово-цифровое преобразование, установив бит GO_DONE (ADCON0).

5. Ожидать окончание преобразования (длительность преобразования $12 \cdot T_{AD}$), т.е. ожидать или сброс бита GO_DONE, или (если разрешено) прерывание по окончании преобразования.

6. Считать результат преобразования из регистров ADRESH:ADRESL и сбросить бит ADCIF (если необходимо).

7. Для запуска следующего преобразования необходимо выполнить шаги, начиная с пункта 1, 2 или 3. Время преобразования одного бита определяется как время T_{AD}. Время ожидания перед следующим преобразованием должно быть не менее $2 \cdot T_{AD}$.

Для обеспечения необходимой точности преобразования конденсатор C_{HOLD} должен успевать заряжаться до уровня входного напряжения. Схема модели входа АЦП приведена на **Рис. 42**. Выходное сопротивление R_S и сопротивление ключа выборки R_{SS} влияют на время зарядки емкости C_{HOLD}. Величина сопротивления ключа выборки (R_{SS}) зависит от напряжения питания U_{CC} (**Рис. 42**). Значение выходного сопротивления источника аналогового сигнала R_S влияет на значение входного смещения напряжения из-за тока утечки вывода. Выходное сопротивление R_S должно быть не более 10 КОм. При его уменьшении время заряда емкости C_{HOLD} уменьшается. После выбора аналогового входного канала до начала преобразования, должно пройти определенное время для заряда емкости C_{HOLD}. Для расчета этого времени воспользуетесь уравнением, описанным далее. Уравнение дает ошибку в пределах ½ LSb (шага АЦП). Ошибка в ½ LSb, это максимальная погрешность, позволяющая работать модулю АЦП с необходимой точностью вычисления.

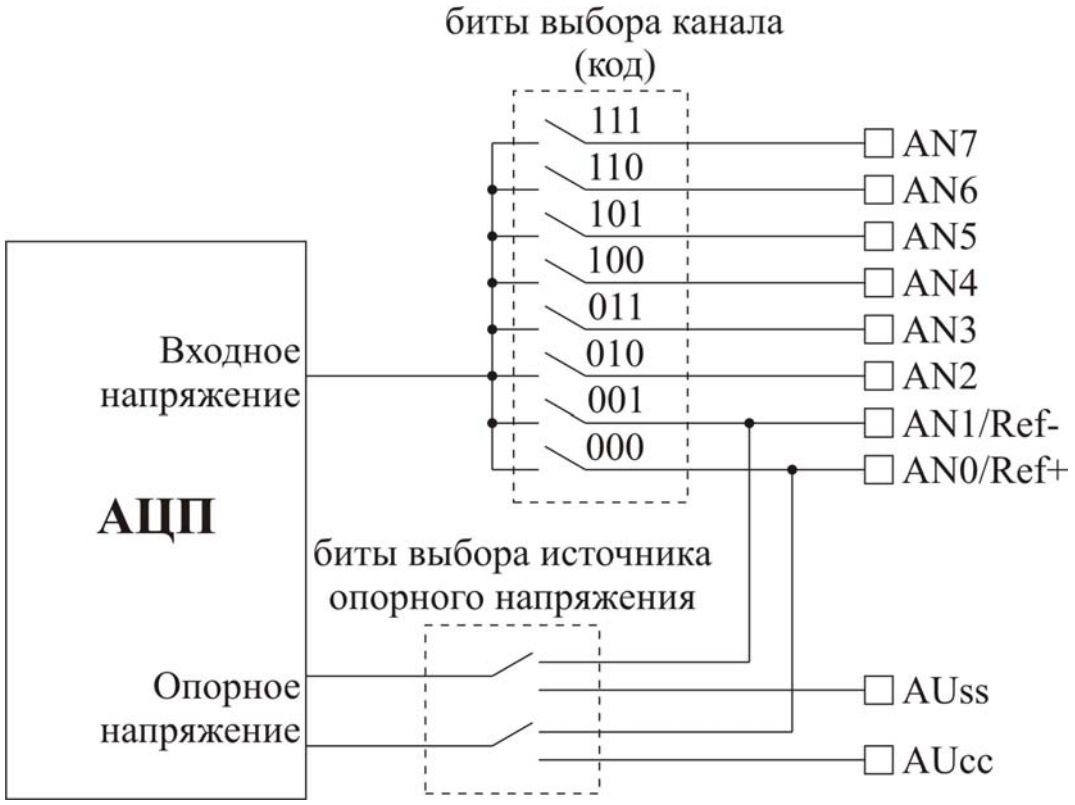


Рис. 40 Блок-схема одуля АЦП

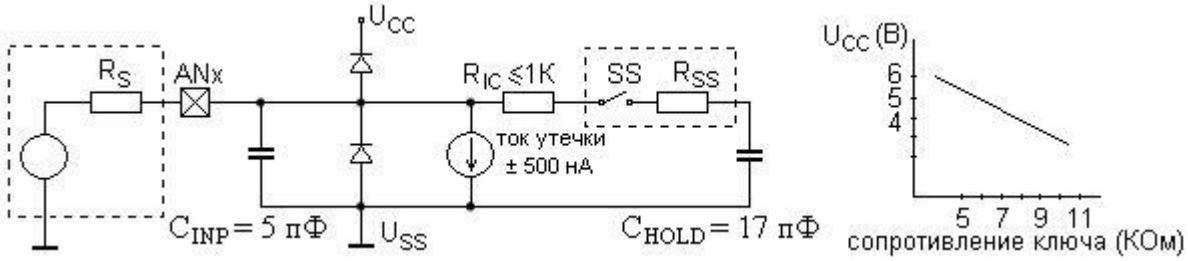


Рис. 41 Схема аналогового входа АЦП

Вычисление минимального времени задержки

T_{ACQ} = время установки схемы + время зарядки C_{HOLD} + температурный коэффициент
$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

Уравнение вычисления минимального времени заряда емкости C_{HOLD}.

$U_{HOLD} = (U_{REF} - (U_{REF} / 2048)) \cdot (1 - e^{(-TC / CHOLD (RIS + RSS + RS))})$
или: $T_C = C_{HOLD} (R_{IS} + R_{SS} + R_S) \cdot Ln(1/2047)$

Для C_{HOLD} = 17 пФ, R_S = 10 Ком, ошибки ½ LSB, U_{CC} = 5 В (R_{SS} = 6 КОм), Temp = 50ºС, U_{HOLD} = 0:

$T_C = -17 \cdot (1 + 6 + 10) \cdot Ln(1/2047) = 2.2$ мкс

$T_{ACQ} = 2$ мкс + 2.2 мкс + [(Temp - 25ºС) · (0.05 мкс/ºС)] = 5.45 мкс

Примечание:
1. Опорное напряжение U_{REF} не влияет на уравнение.
2. Конденсатор C_{HOLD} после каждого преобразования не разряжается.

3. После того как преобразование завершено необходимо программно обеспечить задержку не менее $2 \cdot T_{AD}$, прежде чем начать следующее преобразование. В течение этого времени конденсатор C_{HOLD} не подключен к выбранному входному каналу АЦП.

Время преобразования АЦП зависит от частоты тактовых импульсов преобразования (T_{AD}). Для 12-ти разрядного аналогово-цифрового преобразования требуется время $14 \cdot T_{AD}$. Источники импульсов тактирования АЦП выбираются программно. Для достоверного аналогово-цифрового преобразования должен быть выбран источник импульсов, обеспечивающий **время T_{AD} не менее 1.6 мкс**. Возможны четыре варианта:

- $F_C/8$ ($T_{AD}=8 \cdot T_C$) - используется при тактовой частоте микроконтроллера до 5 МГц;
- $F_C/32$ ($T_{AD}=32 \cdot T_C$) - используется при тактовой частоте микроконтроллера до 20 МГц;
- $F_C/64$ ($T_{AD}=64 \cdot T_C$) - используется при тактовой частоте микроконтроллера до 33 МГц;
- внутренний RC генератор - если тактовая частота микроконтроллера больше 1 МГц, то RC генератор рекомендуется использовать только в SLEEP режиме.

Регистры ADCON1 и DDR управляют настройкой выводов АЦП. Если выводы микросхемы конфигурируются как аналоговые входы, то необходимо установить в единицу соответствующие биты в регистре DDR. Если соответствующий бит сброшен, то вывод настраивается как цифровой выход. Модуль АЦП работает независимо от установленного состояния битов CHS2:CHS0 и битов регистра DDR. При считывании порта, результат чтения разрядов, соответствующих выводам, настроенным как аналоговые входы, будет всегда равен нулю. Аналоговые уровни на цифровом входе не влияют на корректность преобразования. Аналоговый сигнал, подаваемый на цифровой вход (включая AN0-AN7), влияет на ток потребления входного буфера, это приводит к повышению энергопотребления микроконтроллера.

Для запуска аналогово-цифрового преобразования необходимо включить АЦП и установить в «1» бит GO_DONE. Эти биты должны устанавливаться разными командами. После определенного времени на выбранном канале начнется преобразование (**Рис.43**), длительность первого такта от T_{CY} до T_{AD}. Сброс в «0» бита GO_DONE во время преобразования останавливает преобразование. При этом регистры ADRESH:ADRESL не изменяют своего значения. После досрочного завершения преобразования необходимо обеспечить временную задержку $2 \cdot T_{AD}$, т.к. выбранный канал не подключен к внутреннему конденсатору в течение этого времени.

12-ти разрядный результат преобразования АЦП записывается в 16-ти разрядный регистр ADRESH:ADRESL. Модуль АЦП позволяет преобразовывать 12-ти разрядное значение в 16-ти разрядное, выравниванием влево или вправо (см. **Рис. 44**). Выбор формата преобразования осуществляется программно. Не задействованные биты имеют значение «0». При выключенном АЦП значения регистров ADRESH:ADRESL не изменяются, и регистры могут быть использованы как универсальные 8-разрядные регистры.

от T_{CY} до T_{AD}	T_{AD} 1	T_{AD} 2	T_{AD} 3	T_{AD} 4	T_{AD} 5	T_{AD} 6	T_{AD} 7	T_{AD} 8	T_{AD} 9	T_{AD} 10	T_{AD} 11	
		b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	

← начало преобразования

← отключение конденсатора от аналогового входа (около 100 нс)

← установка бита GO

в цикле Q4: загрузка результата вА DRES, сброс бита GO, установка флага ADCIF, подключение конденсатора к аналоговому входу.

Рис.43 Работа модуля АЦП по тактам

Выравнивание вправо:

ADRESH								ADRESL							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	0	0	12-ти разрядный результат преобразования											

Выравнивание влево:

ADRESH								ADRESL							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
12-ти разрядный результат преобразования												0	0	0	0

Рис.44 Выравнивание результата преобразования АЦП

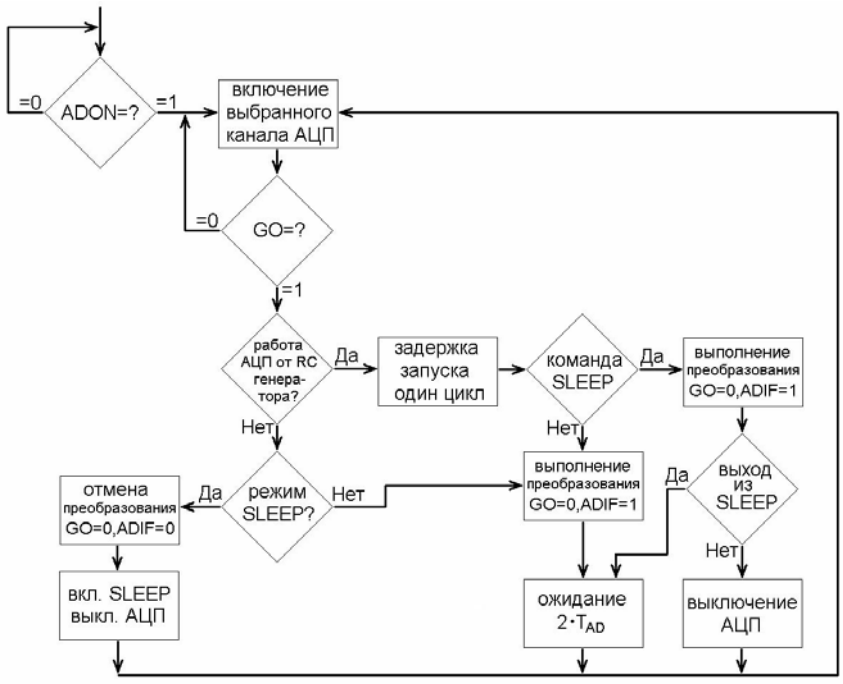


Рис.42 Блок-схема работы АЦП

Модуль АЦП может работать в спящем режиме микроконтроллера в случае если источником тактовых импульсов для АЦП будет внутренний RC генератор (ADCS1:ADCS0=11). Если выбран этот режим тактирования, то модуль АЦП, прежде чем начать преобразование, произведет задержку в течение одного

цикла команд. Это позволяет программе выполнить команду SLEEP, для уменьшения цифрового шума во время преобразования (команда SLEEP должна следовать непосредственно за командой, устанавливающей бит GO_DONE). После завершения преобразования бит GO_DONE сбрасывается, результат преобразования записывается в регистры ADRESH:ADRESL. Если разрешено прерывание от АЦП, то микроконтроллер будет выведен из режима SLEEP. Если прерывание запрещено, то после преобразования модуль АЦП будет выключен, хотя бит ADON останется установленным в «1». Если был выбран не RC генератор, а другой источник тактовых импульсов для АЦП, то выполнение команды SLEEP прервет преобразование и выключит модуль АЦП, уменьшив ток потребления микроконтроллера, оставив ADON=1.

При сбросе микроконтроллера модуль АЦП выключается и останавливается преобразование (если оно было начато). Регистры ADRESH:ADRESL не меняют значения, а после сброса по включению питания их значение не определено.

Предпочтительно использовать АЦП с **T_{AD} не больше 8 мкс**, но не меньше рекомендованного нижнего предела. В системах с низкой рабочей частотой и в случае использования АЦП в режиме SLEEP микроконтроллера, источником тактового сигнала должен быть встроенный RC генератор. В других случаях используется тактовый сигнал от основного тактового генератора. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. Если каналы цифрового ввода/вывода постоянно активны, потеря точности из-за шумов при переключении может быть значительной. При работе АЦП в SLEEP режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой.

Если значение входного напряжения АЦП превышает на 0.3 В величину питающих напряжений (U_{SS} и U_{CC}),то точность преобразования выйдет за пределы значений, оговоренных в спецификации.

Для сглаживания пульсаций входного сигнала на вход АЦП может добавляться внешняя RC цепочка. Значение сопротивления R_S должно выбираться, чтобы общее сопротивление источника сигнала было в пределах рекомендованной величины 10 КОм. Любой внешний компонент, подключенный к аналоговому входу, должен иметь низкий ток утечки через выводы.

Блок Цифро-Аналогового Преобразователя

Блок ЦАП имеет 2 канала и предназначен для формирования аналогового сигнала по цифровому коду, записываемому в него.

Блок обеспечивает как синхронную, так и асинхронную выработку нового значения напряжения при записи в младший байт первого канала (в синхронном режиме) и выработку нового значения напряжения при записи в младший байт (в асинхронном режиме).

Важно! - для корректной работы блока ЦАП, необходимо производить последовательную запись кода в регистр старшего разряда (DAC0H или DAC1H), а затем в регистр младшего разряда (DAC0L или DAC1L) для каждого канала. Для работы в синхронном режиме необходимо задать значение регистра DAC1L

в последнюю очередь, после чего произойдет выработка нового значения напряжения по двум каналам.

Регистр Управления и Состояния ЦАП

Таблица 58

DAC_CONT.ADR = 0x12, банк доступа BANK = 0x04

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	SYNC_A	DACON1	ENOUT1	MREF1	DACON0	ENOUT0	MREF0	DACFM

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 59

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	SYNC_A	Флаг синхронизации канала 1 и 0: 0 = каналы асинхронны, преобразование начинается после записи в младший байт; 1 = каналы синхронизованы, преобразование обоих каналов начинается после записи в младший байт канала 1.
6	DACON1	Флаг включения канала 1: 0 = канал выключен 1 = канал включен
5	ENOUT1	Флаг разрешения выдачи сигнала канала 1: 0 = на DACout2 == 0 1 = выдача преобразованного сигнала
4	MREF1	Флаг выбора опорного напряжения канала 1: 0 = сигнал внутреннего источника опорного напряжения 1 = сигнал с DAC2ref
3	DACON0	Флаг включения канала 0: 0 = канал выключен 1 = канал включен
2	ENOUT0	Флаг разрешения выдачи сигнала канала 0: 0 = на DACout1 == 0 1 = выдача преобразованного сигнала
1	MREF0	Флаг выбора опорного напряжения канала 0: 0 = сигнал внутреннего источника опорного напряжения 1 = сигнал с DAC1ref

0	DACFM	Смещение слова в регистре данных: 0 = левое смещение 1 = правое смещение

Блок Компаратор

Блок компаратор предназначен для сравнения уровней сигналов. Блок позволяет сравнить сигналы: PD7/COMP2 с PD6/COMP1 или со внутренним источником опорного напряжения U_{op}.

Схема контроллера блока компаратора позволяет обработать непосредственно сигнал с выхода компаратора, либо сгенерировать прерывание по факту изменения состояния генератора. Флаг C_IF может вырабатываться при наличии заданного уровня, либо при появлении заданного фронта на выходе компаратора. На **Рис. 46** представлена блок-схема компаратора.

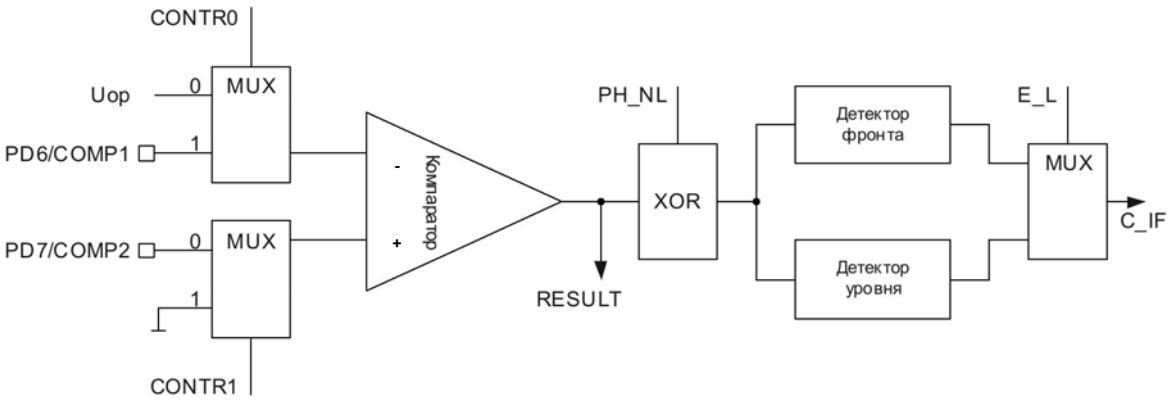


Рис.46 Структурная-схема блока компаратора

Регистр Управления и Состояния Компаратора

Таблица 60

COMPARE.ADR = 0x17, банк доступа BANK = 0x04

Номер	7	6	5	4	3	2	1	0
Доступ*	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**								
	RESULT	STAGE	C_IF	E_L	PH_NL	CONTR 1	CONTR 0	COMP_ ON

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 61

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	RESULT	Состояние выхода компаратора
6	STAGE	Состояние выхода компаратора
5	C_IF	Флаг возникновения события компаратора 0 = нет события 1 = есть событие
4	E_L	Флаг выбора режима работы Фронт/Уровень: 0 = схема детектирует уровень 1 = схема детектирует фронт
3	PH_NL	Флаг выбора режима работы Фронт (E_L=1): 0 = схема детектирует отрицательный фронт 1 = схема детектирует положительный фронт Флаг выбора режима работы Уровень (E_L=0): 0 – схема детектирует низкий уровень 1 – схема детектирует высокий уровень
2	CONTR1	Флаг выбора отрицательного входа Компаратора: 0 – вывод PD7/COMP1 функционирует как аналоговый вход модуля компаратора (вне зависимости включен модуль компаратора или нет) 1 – вывод PD7/COMP1 функционирует как разряд порта (вход/выход)
1	CONTR0	Флаг выбора положительного входа Компаратора: 0 – сигнал внутреннего источника опорного напряжения 1 – сигнал с PD6/COMP0
0	COMP_ON	Разрешение работы блока Компаратор: 0 – блок отключен 1 – блок включен

Блок внутренней памяти данных EEPROM

Блок контроллера EEPROM памяти предназначен для возможности работы ядра микроконтроллера с встроенной памятью типа EEPROM размером 256 8-ми битных слов. Память EEPROM имеет организацию 16 строк по 16 8-ми битных слов. Минимально доступной для чтения и запись является одно 8-ми битное слово. Минимально доступной для стирания является одна строка. Перед записью необходимо провести стирание строк, в которых будет расположена записываемая информация. Запись возможна только «1» поверх «0». В очищенном состоянии память содержит все «0».

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.

Основные выполняемые функции и возможности

Таблица 62

Наименование	Краткое описание, качественно-количественные характеристики
Запись 8-ми битного слова в EEPROM	Позволяет записать в заданный байт, значение битов отличное от нуля. Запись в TEST режиме возможна всегда
Чтение 8-ми битного слова из EEPROM	Позволяет считать из памяти байт
Очистка всей EEPROM	Позволяет стереть содержимое всей памяти
Очистка строки EEPROM	Позволяет очистить заданную строку памяти.
Запись всей EEPROM одним значением	Позволяет проинициализировать всю память одним значением

Регистр режима работы контроллера

Работа с памятью EEPROM осуществляется по средствам чтения и записи регистров контроллера. Описание регистров приведено ниже.

Регистр контроля и тестирования

Таблица 63

EE_CONT. ADR = 0x14, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	1	0	0	0	0	0	0	0
	TEST_P	EETEST	CPTEST	VEE2	VEE1	EEBRG2	EEBRG1	EEBRG0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 64

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	TEST_P	Сигналы для отбраковочных испытаний микросхемы.
6	EETEST	Сигналы для отбраковочных испытаний микросхемы.
5	CPTEST	
4	VEE2	
3	VEE1	
2..0	EEBRG[2:0]	Режим определения временных характеристик: В зависимости от частоты работы ядра, необходимо указать контроллеру эту частоту, исходя из которой, контроллер будет формировать необходимые длительности сигналов Программирования и Стирания заданных длительностей. 000 – частота Fc = 30 – 32 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.

Регистр режима работы

Таблица 65

EE_MODE. ADR = 0x15, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	RO	R/W	RO	RO	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	EE_EN	-	IEBUSY	BUSY	-	MODE2	MODE1	MODE0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 66

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7	EN_EE	Разрешение работы контроллера EEPROM 0 = Контроллер выключен 1 = Контроллер включен
6	-	Зарезервировано
5	IEBUSY	Разрешения прерывания по окончанию занятости контроллера EEPROM
4	BUSY	Флаг занятости контроллера EEPROM
3	-	Зарезервировано
2..0	MODE[2:0]	Режим работы контроллера: 000 – Нет работы; 001 – Чтение слова из EEPROM; 010 – Запись слова в EEPROM; 011 – Отчистка строки EEPROM; 100 – Отчистка всей EEPROM; 101 – Запись всей EEPROM одним значением; 110 – Запрещено; 111 – Запрещено. **После каждой операции, блок должен быть переведен в режим «Нет работы»**

Регистр данных

Таблица 67

EE_DATA. ADR = 0x16, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 68

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	DATA[7:0]	Регистр данных: При выдаче команды в этот регистр записывается команда, при выдаче адреса в регистр записывается адрес, при выдаче или записи данные

Регистр адреса обращения

Таблица 69

EE_ADR. ADR = 0x17, Банк доступа BANK = 0x05

Номер	7	6	5	4	3	2	1	0
Доступ*	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Значение после сброса**	0	0	0	0	0	0	0	0
	EEADR7	EEADR6	EEADR5	EEADR4	EEADR3	EEADR2	EEADR1	EEADR0

*

R/W - бит доступен на чтение и запись;
RO - бит доступен только на чтение;
U - бит физически не реализован или зарезервирован.

Таблица 70

№	Функциональное имя бита	Расшифровка функционального имени бита, краткое описание назначения и принимаемых значений
7..0	EEADR [7:0]	Регистр адреса данных: При чтении и записи одного слова используются все разряды адреса, при очистке строки имеют значения только EEADR[7:4], при блочной очистке и записи (в TEST режиме) содержимое регистра значения не имеет. При выполнении операций установления флагов защиты от записи или стирания, номер строки для которой будет устанавливаться флаг определяется данным регистром

Работа блока по стиранию, записи и чтению данных

Включение EEPROM

После выставления бита EE_EN в регистре EE_MODE производится включение памяти EEPROM. Процесс включения при тактовой частоте микроконтроллера 33 МГц занимает порядка 45 мкс (Tsuy)
Последовательность действий при включении EEPROM представлена на **Рис. 47**.

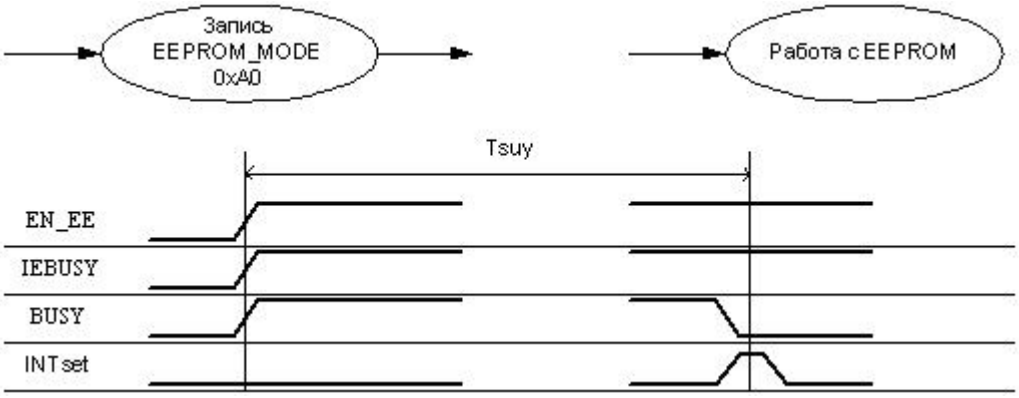


Рис. 43 Последовательность действий при включении EEPROM.

Очистка всей EEPROM

Для очистки всего содержимого EEPROM используется команда очистка всей EEPROM. Для очистки всей памяти необходимо чтобы был включен контроллер EEPROM. Процесс очистки всей EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp)

Последовательность действий при очистке всей EEPORM представлена на **Рис. 48**.

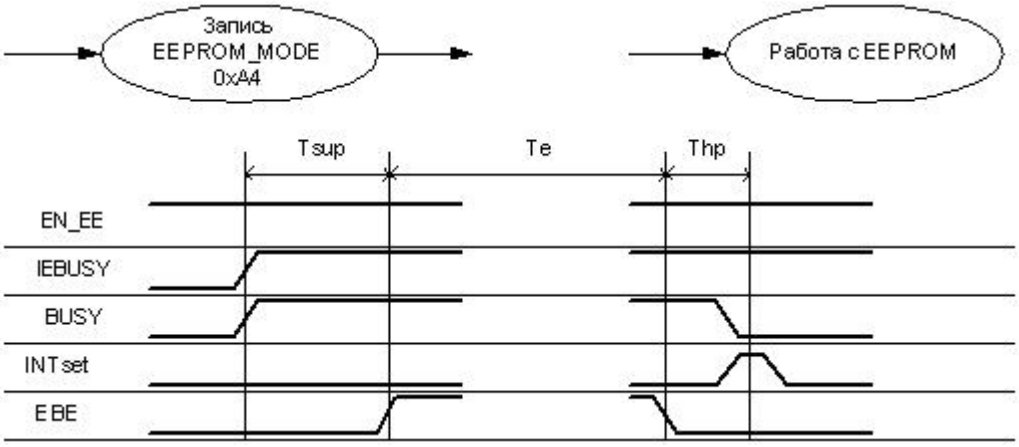


Рис. 44 Последовательность действий при очистке всей EEPORM

Запись всей EEPROM одним значением

Для заполнения всего содержимого EEPROM используется команда записи всей EEPROM. Перед подачей этой команды в регистр EE_DATA необходимо записать значение, которым будет заполнена вся память. Для заполнения всей памяти необходимо чтобы был включен контроллер EEPROM. Процесс заполнения всей EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp)

Последовательность действий при заполнении всей EEPORM представлена на **Рис. 45**.

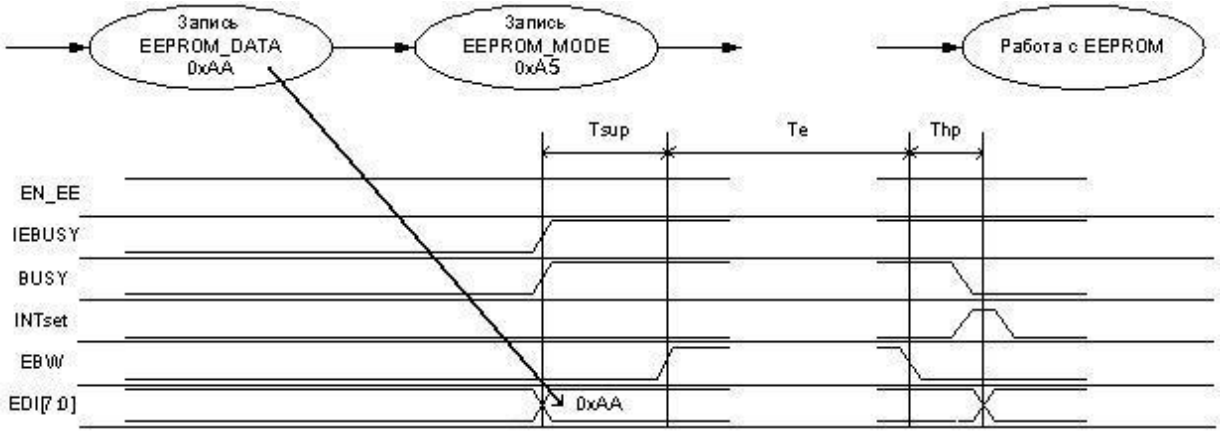


Рис. 45 Последовательность действий при заполнении всей EEPORM

Очистка строки EEPROM

Для очистки содержимого одной строки EEPROM используется команда очистки одной строки EEPROM. Перед подачей этой команды в регистр EE_ADR[7:4] необходимо записать адрес строки значение для очистки (младшие разряды регистра EE_ADR значения не имеют). Для очистки заданной строки памяти необходимо чтобы контроллер EEPROM был включен. Процесс очистки одной строки EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp)

Последовательность действий при очистке одной строки EEPORM представлена на **Рис.46**.

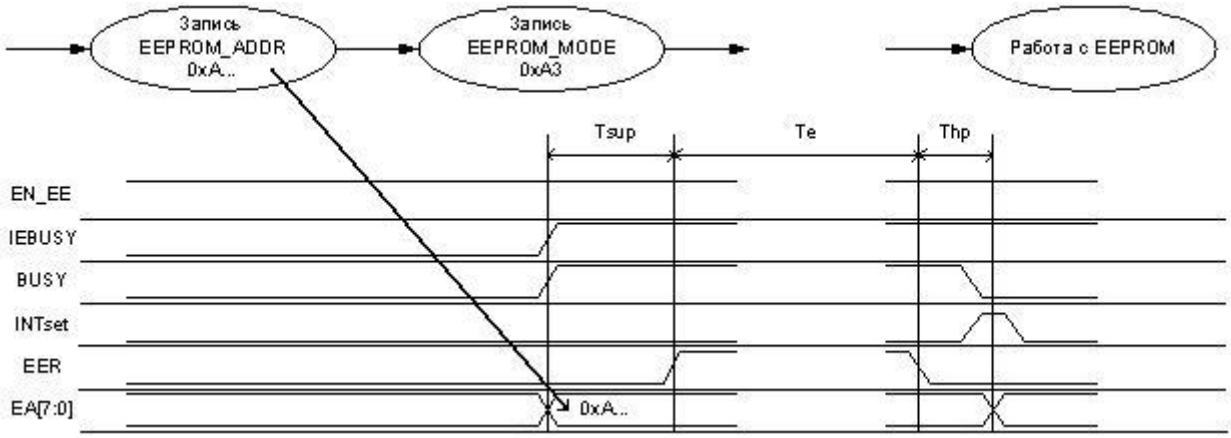


Рис. 46 Последовательность действий при очистке одной строки EEPORM

Запись слова в EEPROM

Для записи одного 8-ми битного слова в память EEPROM используется команда запись слова в EEPROM. Перед подачей этой команды в регистр EE_ADR[7:0] необходимо записать адрес для записи, а в регистр EE_DATA записать значение для сохранения в памяти. Для записи слова в память

необходимо, чтобы контроллер EEPROM был включен. Процесс записи строки EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 5 мс (Tsup, Te и Thp).

Последовательность действий при записи одного слова в EEPORM представлена на **Рис. 47**.

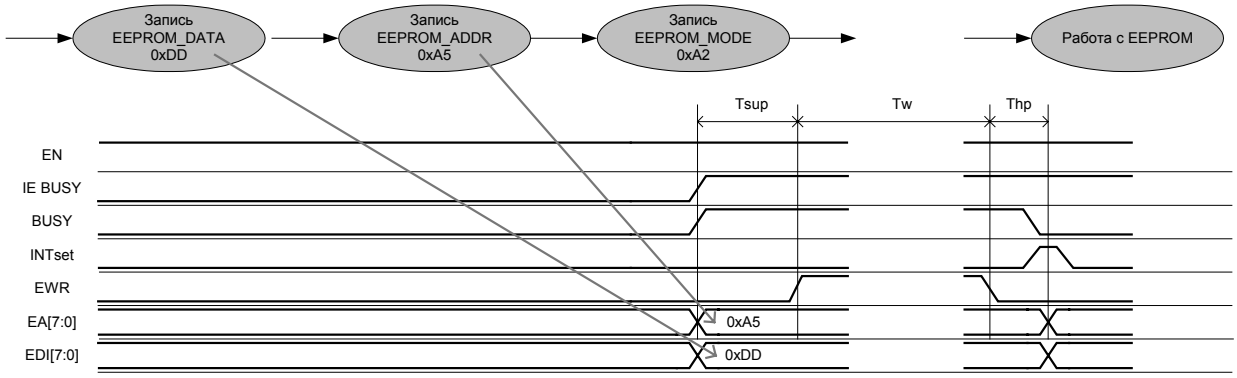


Рис. 47 Последовательность действий при записи одного слова в EEPORM.

Чтение слова из EEPROM

Для чтения одного 8-ми битного слова из памяти EEPROM используется команда чтение слова из EEPROM. Перед подачей этой команды в регистр EE_ADR[7:0] необходимо записать адрес для чтения. Для чтения слова из памяти необходимо, чтобы контроллер EEPROM был включен. Процесс чтения строки EEPROM при тактовой частоте микроконтроллера 33 МГц занимает порядка 1 мкс (Tsux, Tacc и Thx).

Последовательность действий при чтении одного слова из EEPORM представлена на **Рис. 48**.

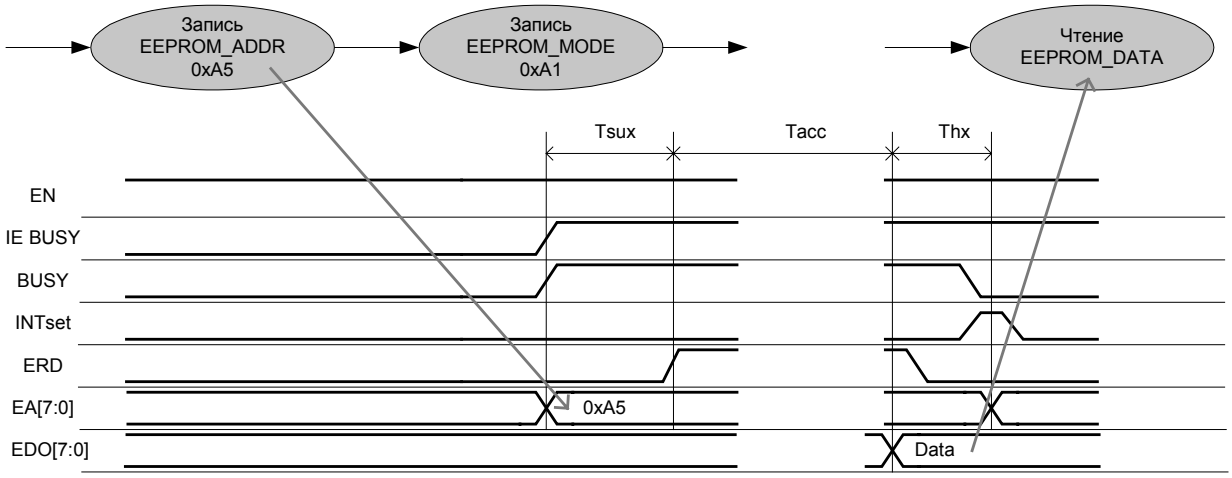


Рис. 48 Последовательность действий при чтении одного слова из EEPORM.

Описание блока управления EEPROM – памятью программ

Блок позволяет осуществлять чтение, запись и стирание памяти различными способами, в частности через:

- команды ядра TABLWD, TABLRD (в этом случае используются сигналы чтения и записи от ядра);
- установку/снятие битов управления периферийных регистров.

Адресное пространство 4К слов или от 16`h0000 до 16`h0FFF. Блок фактически управляет двумя блоками EEPROM памяти. Младшая половина от 16`h0000 до 16`h07FF, соответственно старшая половина от 16`h0800 до 16`h0FFF.

При работе через установку/снятие битов управления периферийных регистров возможно использование функции аппаратной поддержки. Функция реализует автоматическую выработку длительностей сигналов записи/стирания, а также автоматический сброс бит записи/стирания периферийных регистров. По умолчанию включен режим аппаратной поддержки.

Блок осуществляет загрузку и хранение конфигурационных бит из памяти.

В блоке реализованы следующие защитные функции:

- при доступе по чтению данных по адресу выходящему за пределы адресного пространства (0000 – 0FFF) получаемые данные будут равны 0000. Исключением являются адреса FE00 – FE0F. При обращении по чтению к этим адресам данные будут содержать текущее конфигурационное слово (см. раздел «Схема сброса микроконтроллера»);
- все попытки записи по адресам выходящим за пределы адресного пространства блока будут блокированы. Исключения: попытки записи по адресам 16`h0FF0 – 16`h0FFF блокируются, так как это область хранения конфигурационных бит. Запись конфигурационных бит осуществляется по адресам FE00 – FE0F;
- изменение содержимого регистра хранения конфигурации в рабочем режиме не возможно, т.е. доступ по записи заблокирован;
- реализован контроль правильности адресации при выполнении операций записи слова и стирания строки, в случае если режим аппаратной поддержки включен.

Описание регистров

Таблица 71

Общее описание регистров блока EEPROM_interface

Обозначение	Адрес	Доступ	Значение после сброса
EEDIV	13h, банк 6	RW RW RW RW RW RW RW RW	0000 0101
EECONL	11h, банк 14	R- R- RW RW RW RW RW RW	0000 0000
EECONH	12h, банк 14	R- R- RW RW RW RW RW RW	0000 0000
EDLSB	12h, банк 15	R- R- R- R- R- R- R- R-	0000 0000
EDMSB	13h, банк 15	R- R- R- R- R- R- R- R-	0000 0000
EEMOD	14h, банк 15	R- RW RW RW RW RW RW RW	0000 0000
EAMSB	15h, банк 15	R- U- U- U- RW RW RW RW	0--- 0000
EALSB	16h, банк 15	RW RW RW RW RW RW RW RW	0000 0000
CFREG	17h, банк 15	U- RW RW RW RW RW RW RW	-111 1111

Регистр коэффициента деления частоты генератора

Таблица 72

EE_DIV.ADR = 0x13, банк доступа BANK = 0x06

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EEDIV	**EEDIV**	**EEDIV**	**EEDIV**	**EEDIV**	**EEDI2**	**EEDIV**	**EEDIV**
бит 7	6	5	4	3	2	1	бит 0
бит 7-0		Коэффициента деления частоты генератора с целью выработки импульсов записи и стирания длительностью ≥ 5 мс. Делитель состоит из прескалера и постскалера. Значение коэффициента деления прескалера фиксировано и равно 906. Значение коэффициента постскалера равно значению регистра. Таким образом при измени значения регистра на 1, реальный коэффициент деления изменится на 906. Правило выбора коэффициента деления: K = 5 мс / 906·Tosc В случае дробного результата, округление проводить в большую сторону, для получения времени импульса больше 5 мс					

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 младшей половиной EEPROM памяти

Таблица 73

Регистр EECONL (адрес: 11h, банк 14)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LOCKL	**ERRL**	**EETL**	**EBWL**	**EBEL**	**EERL**	**EWRL**	**ERDL**
бит 7	6	5	4	3	2	1	бит 0
бит 7 LOCKL		Флаг режима младшей половины EEPROM 1 – режим standby 0 – рабочий режим					
бит 6 ERRL		Флаг ошибки данных младшей половины EEPROM 1 – указывает на наличие ошибки 0 – указывает на отсутствие ошибки					
бит 5 EETL		Установка тестового режима работы младшей половины EEPROM. Не доступен для записи в рабочем режиме микроконтроллера (Op_mode=1) 1 – тестовый режим 0 – рабочий режим					
бит 4 EBWL		Одновременная запись всей младшей половины EEPROM данными со входа памяти.					

бит 3 EBEL	Одновременное стирание всей младшей половины EEPROM. Для описания работы с этим разрядом смотри п. 3.1
бит 2 EERL	Стирание строки младшей половины EEPROM. Адрес строки определяется 10-4 разрядами адресной шины. Таким образом, строка содержит 16 слов. Для описания работы с этим разрядом смотри п. 3.1
бит 1 EWRL	Запись одного слова младшей половины EEPROM данными со входа памяти. Адрес слова определяется 10-0 разрядами адресной шины. Для описания работы с этим разрядом смотри п. 3.1
бит 0 ERDL	Чтение слова младшей половины EEPROM. Адрес слова определяется 10-0 разрядами адресной шины. Для описания работы с этим разрядом смотри п. 3.3

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 старшей половиной EEPROM памяти

Таблица 74

EECONH.ADR = 0x12, банк доступа BANK = 0x0E)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LOCKH	**ERRH**	**EETH**	**EBWH**	**EBEH**	**EERH**	**EWRH**	**ERDH**
бит 7	6	5	4	3	2	1	бит 0

бит 7 LOCKH	Флаг режима старшей половины EEPROM 1 – режим standby 0 – рабочий режим
бит 6 ERRH	Флаг ошибки данных старшей половины EEPROM 1 – указывает на наличие ошибки 0 – указывает на отсутствие ошибки
бит 5 EETH	Установка тестового режима работы старшей половины EEPROM. Не доступен для записи в рабочем режиме микроконтроллера (Op_mode=1) 1 – тестовый режим 0 – рабочий режим
бит 4 EBWH	Одновременная запись всей старшей половины EEPROM данными со входа памяти. Для описания работы с этим разрядом смотри п. 3.1
бит 3 EBEH	Одновременное стирание всей старшей половины EEPROM.
бит 2 EERH	Стирание строки старшей половины EEPROM. Адрес строки определяется 10-4 разрядами адресной шины. Таким образом, строка содержит 16 слов.
бит 1 EWRH	Запись одного слова старшей половины EEPROM данными со входа памяти. Адрес слова определяется 10-0 разрядами адресной шины.

бит 0 ERDH	Чтение слова старшей половины EEPROM. Адрес слова определяется 10-0 разрядами адресной шины.

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 режима работы EEPROM памяти

Таблица 75

EEMOD.ADR = 0x14, банк доступа BANK = 0x0F

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STATE	**RESET**	**CPEN**	**HWS**	**AM**	**TM2**	**TM1**	**ESTBY**
бит 7	6	5	4	3	2	1	бит 0

бит 7 STATE	Флаг состояния автомата аппаратной поддержки 1 – автомат находится в состоянии работы 0 – автомат выключен
бит 6 RESET	Сброс автомата аппаратной поддержки и делителя. 1 – схемы находятся в состоянии сброса 0 – схемы находятся в рабочем состоянии
бит 5 CPEN	Разрешение работы блока Charge Pumpe. 1 – блок включен. Важно: установка высокого напряжения необходимого для операций записи и стирания происходит через 10 - 20 мкс после установки этого бита в 1. Признаком появления высокого напряжения является EAMSB[7] = 1. 0 – блок выключен
бит 4 HWS	Отключение режима аппаратной поддержки. Для описания работы с этим разрядом смотри п. 3.1 1 – режим отключен 0 – режим включен (значение по умолчанию)
бит 3 AM	Режим адресации EEPROM памяти. Для описания работы с этим разрядом смотри п. 3.2. 1 – адрес поступает с регистров EAMSB[3:0] и EALSB[7:0] 0 – адрес поступает с входа address (от ядра)
бит 2 TM2	Выбор тестового подрежима обеих половин EEPROM памяти. Состояние разряда имеет значение, только если EECONH[5] или EECONL[5] равны 1. Включение тестового режима относится к той половине EEPROM памяти для которой установлен бит EECONx[5]. Не доступен для записи в рабочем режиме микроконтроллера (Op_mode=1). 1 – выбран тестовый подрежим «High». 0 – выбран тестовый подрежим «Low».

бит 1 TM1	Выбор тестового режима обеих половин EEPROM памяти. Состояние разряда имеет значение, только если EECONH[5] или EECONL[5] равны 1. Включение тестового режима относится к той половине EEPROM памяти для которой установлен бит EECONx[5]. Не доступен для записи в рабочем режиме (Op_mode=1). 1 – выбран тестовый режим «Internal» 0 – выбран тестовый режим «External».
бит 0 ESTBY	Отключение (перевод в режим standby) EEPROM памяти. Не доступен для записи в рабочем режиме микроконтроллера (Op_mode=1). 1 – EEPROM отключена. Признаком этого состояния служат EECONH[7] и EECONL[7] равные 1. 0 – EEPROM находится в рабочем состоянии. Значение по умолчанию.

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 старшего адреса EEPROM памяти

Таблица 76
EAMSB.ADR = 0x15, банк доступа BANK = 0x0F

R-0	U-0	U-0	U-0	R/W -0	R/W-0	R/W-0	R/W-0
CPRDY	**-**	**-**	**-**	**EA [11]**	**EA [10]**	**EA [9]**	**EA [8]**
бит 7	6	5	4	3	2	1	бит 0

бит 7 CPRDY	Флаг состояния блока Charge Pumpe 1 – блок включен и на выходе установлено высокое напряжение. Память готова к операции запись или стирание 0 – блок выключен или на выходе еще нет высокого напряжения
биты 6 - 4	Не реализованы.
бит 3 EA[11]	Старший разряд шины адреса EEPROM памяти. Адресует старшую (1) или младшую (0) половину. Состояние имеет значение только при EEMOD[3] = 1. Для описания работы с этим разрядом смотри п. 3.2.
бит 2-0 EA[10:8]	Разряды 10 – 8 шины адреса EEPROM памяти. Адресуют слова в пределах одной из половин EEPROM памяти. Состояние имеет значение только при EEMOD[3] = 1.

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 младшего адреса EEPROM памяти

Таблица 77

Регистр EALSB (адрес: 16h, банк 15)

R/W -0	R/W -0	R/W -0	R/W -0	R/W -0	R/W-0	R/W-0	R/W-0
EA[7]	**EA[6]**	**EA[5]**	**EA[4]**	**EA[3]**	**EA[2]**	**EA[1]**	**EA[0]**
бит 7	6	5	4	3	2	1	бит 0
бит 7-0 EA[7:0]		Разряды 7 – 0 шины адреса EEPROM памяти. Адресуют слова в пределах одной из половин EEPROM памяти. Состояние имеет значение только при EEMOD[3] = 1.					

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 конфигурационных бит

Таблица 78

CFREG.ADR = 0x17, банк доступа BANK = 0x0F

U -0	R/W -1	R/W -1	R/W -1	R/W -1	R/W-1	R/W-1	R/W-1
-	**DBG_EN**	**BODEN**	**PM0**	**WDT1**	**WDT0**	**FOSC1**	**FOSC0**
бит 7	6	5	4	3	2	1	бит 0
бит 7		Не реализован					
бит 6 DBG_EN		Конфигурационный бит разрешения режима отладки.					
бит 5 BODEN		Конфигурационный бит разрешения работы BOR.					
бит 4 PM0		Конфигурационный бит установки защиты памяти программ.					
бит 3 WDT1		Конфигурационный бит режима WDT.					
бит 2 WDT0		Конфигурационный бит режима WDT.					
бит 1 FOSC1		Конфигурационный бит режима генератора.					
бит 0 FOSC0		Конфигурационный бит режима генератора.					

Примечание:

Регистр не доступен для записи в операционном режиме микроконтроллера (Op_mode=1).

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 младшего байта данных EEPROM памяти

Таблица 79

EDLSB.ADR = 0x12, банк доступа BANK = 0x0F

R -0	**R -0**	**R -0**	**R -0**	**R -0**	**R-0**	**R-0**	**R-0**
ED[7]	**ED[6]**	**ED[5]**	**ED[4]**	**ED[3]**	**ED[2]**	**ED[1]**	**ED[0]**
бит 7	6	5	4	3	2	1	бит 0
бит 7-0		Разряды 7 – 0 шины данных EEPROM памяти (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.

Регистр старшего байта данных EEPROM памяти

Таблица 80

Регистр EDMSB (адрес: 13h, банк 15)

R -0	**R -0**	**R -0**	**R -0**	**R -0**	**R-0**	**R-0**	**R-0**
ED[15]	**ED[14]**	**ED[13]**	**ED[12]**	**ED[11]**	**ED[10]**	**ED[9]**	**ED[8]**
бит 7	6	5	4	3	2	1	бит 0
бит 7-0 ED[15:8]		Разряды 15 – 8 шины данных EEPROM памяти (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.

Функционирование и особенности работы с блоком

Существует 2 режима работы с данным сигналом:

1. Режим аппаратной поддержки (EEMOD[4]=0)

Для записи необходимо установить бит в 1. При этом включение и отключение блока Charge Pumpe, отсчет длительности импульса (обязательна предварительная инициализация регистра EEDIV), а также сброс бита в 0 будут произведены автоматически. Признаком окончания операции могут служить:

- - равенство данного бита 0;
- - бит EEMOD[7] = 0.

Важно: в данном режиме реализована функция защиты от неверной адресации. Если адресована, например, младшая половина памяти, и в то же время пользователь устанавливает бит управления записью или стирания старшей половины, то данное действие будет заблокировано.

2. Режим без аппаратной поддержки (EEMOD[4]=1)

Перед установкой бита в 1 необходимо:

- установить адрес и данные (см. п. 3.2)
- включить Charge Pumpe (EEMOD[5]=1)
- убедиться в наличии высокого напряжения на выходе Charge Pumpe (EAMSB[7] должен быть в состоянии 1)

После установки бита в 1 необходимо отсчитать временной интервал равный 5 мс. Далее сбросить бит в 0. Затем отключить Charge Pumpe (EEMOD[5]=0) если не требуется повторных операций записи или стирания.

Адресация памяти может осуществляться 2-мя способами.

1. EEMOD[3]=0 (режим по умолчанию)

В этом режиме адресация происходит со входа address[15:0], то есть фактически от ядра. Физическое адресное пространство – 12 бит или 4096 слов (11-0). Старший бит фактически выбирает старшую или младшую половины памяти.

Адреса от 0FF0 до 0FFF заняты конфигурационными битами. При прямой адресации к ним возможно обращение только по чтению. Запись в эти адреса блокируется.

Область адресов FE00 – FE0F – виртуальное пространство конфигурационных бит. При обращении к этим адресам по чтению блок выдаст конфигурационное слово, как это описано в п. 3.4. При обращении по записи в любой из указанных адресов, будет записано FFFF, не зависимо от состояния шины данных.

При попытке чтения адресов выходящих за пределы адресного пространства блока (кроме FE00 – FE0F) выходная шина данных примет состояние 0000.

Запись в адреса выходящие за пределы адресного пространства блока (кроме FE00 – FE0F) будет блокироваться.

2. EEMOD[3]=1

В этом режиме адресация происходит от регистров EAMSB и EALSB. Этот режим рекомендуется для выполнения операций стирания.

Адреса от 0FF0 до 0FFF заняты конфигурационными битами. При прямой адресации к ним возможно обращение только по чтению. Запись в эти адреса блокируется.

Чтение памяти

Для чтения содержимого старшей или младшей половин памяти при доступе через регистры, достаточно установить сигнал EECONH[0] или EECONL[0] соответственно в 1 на один системный такт (после чего сбросить в 0). Содержимое адреса памяти будет находиться в регистрах EDMSB и EDLSB. При выполнении данной операции необходимо следить за соответствием адреса и использованием бит регистров EECONH и EECONL.

Конфигурационные биты

Конфигурационные биты физически находятся по адресам 0FF0 – 0FFF. Кроме того, существует виртуальная область памяти конфигурационных бит (FE00 – FE0F).
При чтении конфигурационных битов по любому адресу в диапазоне FE00h-FE0Fh, блок будет возвращать конфигурационное слово, как оно представлено в опсании регистра конфигурации микроконтроллера.
При чтении конфигурационных битов по любому адресу в диапазоне 0FF0h-0FFFh, блок будет возвращать фактическое содержимое ячейки памяти по соответствующему адресу.
Как было сказано ранее, значения битов конфигурации дублируются в памяти программ по адресам 0FF0h-0FFFh. Значению бита = 0 соответствует значение ячейки ПЗУ равное FFFFh, а значению бита =1 - значение ячейки равное 0000h. Соответствие адресов ячеек паямти битам конфигурации указано в опсании регистра конфигурации микроконтроллера.
Область памяти программ 0FF0 - 0FFF доступна только для чтения. Для записи конфигурационных бит необходимо обращаться к адресам FE00 – FE0F.
Для оперативной работы с конфигурационными битами (чтение и запись) в области периферийных регистров реализован регистр CFREG (см. описание регистров). При работе в операционном режиме доступ по записи к нему заблокирован.

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 81**). Чтение битов DBG_EN, BODEN, PM0, WDTPS1, WDTPS0, FOSC1 и FOSC0 возможно по любому адресу в диапазоне FE00h-FE07h.

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

Примечание:

Если в ячейки с адресами 0FF4h 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, за исключением команды* «СТЕРЕТЬ ВСЮ ПАМЯТЬ» из расширенного набора команд.

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. (смотрите **таблицу 82**).

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 81

Регистры конфигурации микроконтроллера

		FE06h	**FE05h**	**FE04h**	**FE03h**	**FE02h**	**FE01h**
-	**-**	**DBG_EN**	**BODEN**	**PM0**	**WDTPS1**	**WDTPS0**	**FOSC1**
бит 15 - 8	бит 7	6	5	4	3	2	1

Таблица 82

Описание регистров конфигурации микроконтроллера

DBGEN: бит 6, адрес FE06h (ячейка ПЗУ: 0FF6h)	**DBGEN** – включение отладочного режима 1 – обычный режим 0 – отладочный режим
BODEN: бит 5,адрес FE05h (ячейка ПЗУ: 0FF5h)	**BODEN** - включение схемы сброса по снижению напряжения питания: 1 = схема включена 0 = схема выключена
PM0: бит 4, адрес FE04h (ячейка ПЗУ: 0FF4h)	**PM0** - биты выбора режима микроконтроллера: 1 = режим микроконтроллера 0 = режим микроконтроллера с защитой кодов программы (т.е. запрет считывания содержимого внутренней памяти программ)
WDTPS1:бит 3, адрес FE03h (ячейка ПЗУ: 0FF3h) **WDTPS0:**бит 2,адрес FE02h (ячейка ПЗУ: 0FF2h)	**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: 0FF1h) **FOSC0:** бит 0, адрес FE00h (ячейка ПЗУ: 0FF0h)	**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 – генератор с внешним низкочастотным кварцевым резонатором (≦ 2 МГц)

Обозначение:
- = зарезервировано, читается как 0.

Внутрисхемное программирование микроконтроллера

Для программирования внутренней EEPROM памяти программ микроконтроллеров 1886ВЕ6У используется последовательный интерфейс ISP (Interface Serial Program). В этом режиме программирования задействованы выводы микроконтроллера и напряжения питания, приведённые в **таблице 83**.

Таблица 83
Выводы ISP интерфейса

Обозначение	**В режиме программирования**		
	Назначение	**Тип**	**Описание**
PA2/RX1/DT1	DT	вход/выход	Последовательные данные
PA3/TX1/CK1	CK	вход	Последовательный синхросигнал
PA1/T0CLK	CLK	вход	Источник синхронизации микроконтроллера
TEST	TEST	вход	Вход для выбора тестового режима
MCLRn	MCLRn	питание	Внешний сброс
UCC	UCC	питание	Напряжение питания
GND	GND	общий	Общий
AUCC	AUCC	питание	Напряжение питания АЦП
AGND	AGND	общий	Общий АЦП

Сторожевой таймер

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 MCLRn/Upp.
- Сброс от сторожевого таймера (если он включен).
- Прерывание с вывода PA0/INT, прерывание от изменения сигналов на порте D, прерывание с PA1/T0CLK, или некоторые перифирийные прерывания (от модуля захвата (регистрации событий), от приемника и передатчика USART в синхронном ведомом режиме, завершение преобразования АЦП).

Другие периферийные устройства не могут генерировать прерывания в режиме SLEEP, так как выключен тактовый генератор микроконтроллера.

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 переходе в режим SLEEP необходимо отключить повышенное питание EEPROM памяти программ и памяти данных. Для отключения EEPROM памяти программ необходимо установить бит ESLP в регистре CPUSTA. Для отключения EEPROM памяти данных необходимо очистить бит EN_EE в регистре EE_MODE. При выполнении команды SLEEP, предварительно выбирается следующая команда (PC+1). Чтобы пробудить устройство прерыванием, должен быть установлен соответсвующий 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.

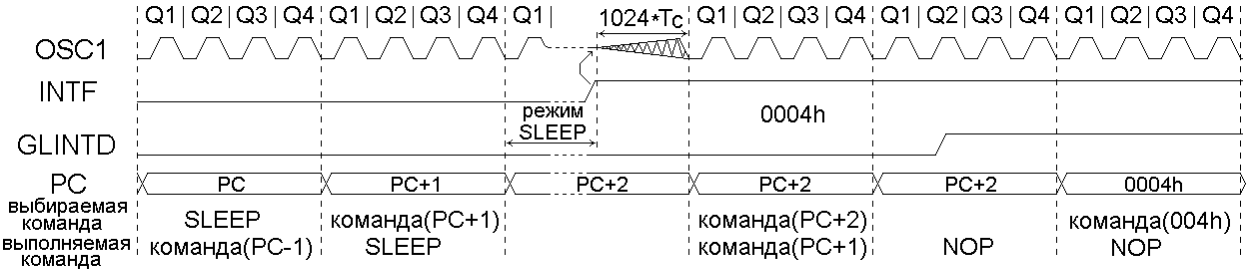


Рис. 49 Выход из режима SLEEP по прерыванию

Примечания:

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

Схема подключения напряжения питания

Схема представлена на **Рис. 50**. Линии питания U_{CC} (положительный вывод напряжения питания цифровой части микроконтроллера) и U_{SS} («земляной» вывод напряжения питания микроконтроллера) подключаются к соответствующим линиям питания на печатной плате. Линии питания AU_{CC} (положительный вывод напряжения питания АЦП микроконтроллера) и AU_{SS} («земляной» вывод напряжения питания АЦП микроконтроллера) для снижения влияния помех рекомендуется подключать к соответствующим аналоговым линиям питания на печатной плате (если таковые имеются). Потенциалы положительных выводов напряжения питания U_{CC} и AU_{CC} должны быть одинаковы. Аналогично потенциалы «земляных» выводов U_{SS} и AU_{SS} также должны быть одинаковыми.

Рекомендуемые значения номиналов конденсаторов, установленных в цепи питания: C1-C6 не менее 0.1 мкф., C7 - не менее 22 мкф.

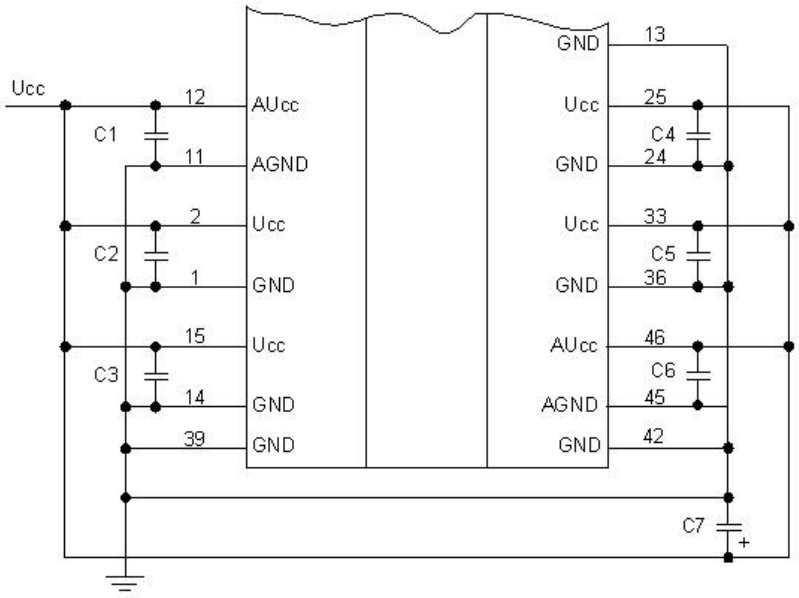


Рис. 50Схема подключения напряжения питания

- C1…C5 - не менее 0,1 мкф, не менее 7В, конденсаторы должны располагаться максимально близко к соответствующим выводам питания.
- C6 - не менее 22 мкф, не менее 7 В.

Система команд

Микроконтроллер поддерживает 58 команд (см. **таблица 84**). 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 EEPROM память программ имеет большую длительность). Коды команд приведены в таблица набора команд. 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 84
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)
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)
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)
GOTO k	Безусловный переход по программе в пределах страницы 8 Кслов. 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
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

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

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)

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
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)
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)

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)
GOTO k	Безусловный переход по программе в пределах страницы 8 Кслов. 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
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
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
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)
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 85

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.

Предельные и предельно-допустимые режимы работы

Таблица 86

Наименование параметра, единица измерения	Буквенное обозначение	Норма параметра			
		Предельно-допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
Напряжение источника питания, В	U_{CC} U_{CCA}	4,5	5,5	-	7,0
Опорное напряжение ЦАП, В	U_{REF_ADC}	3,0	U_{CCA}	-	-
Входное напряжение высокого уровня, В на выводах PA, PC, PD, PE, OSC1, nMCLR, TEST	U_{IH}	0,8•U_{CC}	U_{CC}	-	U_{CC}+0,3
Входное напряжение низкого уровня, В на выводах PA, PC, PD, PE, OSC1, nMCLR, TEST	U_{IL}	0	0,2•U_{CC}	минус 0,3	-
Выходной ток высокого уровня, мА на выводах PA, PC, PD, PE, COMP_OUT	I_{OH}	-	4	-	10
на выводе OSC2		-	1	-	2
Выходной ток низкого уровня, мА на выводах PA, PC, PD, PE, COMP_OUT	I_{OL}	минус 4	-	минус 10	-
на выводе OSC2		минус 1	-	минус 2	-
Частота следования импульсов тактовых сигналов микроконтроллера PA1 и OSC1, МГц	f_C	-	24	-	-
Длительность программирования одного слова, мс	t_{CYW}	-	4	-	-
Длительность сигнала высокого уровня синхронизации на входах PA1 и OSC1, нс при: U_{CC}=4,5 В	$t_{WH(PA_OSC)}$	20	-	-	-
Длительность сигнала высокого уровня прерывания INT/PA0, нс, при: U_{CC}=4,5 В	$t_{WH(INT)}$	50	-	-	-
Длительность сигнала низкого уровня системного сброса, nMCLR, нс при: U_{CC}=4,5 В	$t_{WL(MCLR)}$	100	-	-	-
Параметры универсального последовательного синхронно-асинхронного приемопередатчика					
Время установления данных RX/DT относительно синхронизации TX/CK , нс при: U_{CC}=4,5 В	$t_{SU(TX-RX)}$	17	-	-	-
Время удержания данных RX/DT относительно синхронизации TX/CK, нс при: U_{CC}=4,5 В	$t_{H(TX-RX)}$	17	-	-	-
Параметры Таймера 0					
Длительность сигнала высокого уровня на линии PA1/T0CKI , нс, прескалер включен при: U_{CC}=4,5 В	$t_{WH(PA1)}$	20	-	-	-

Продолжение таблицы 2

Наименование параметра, единица измерения	Буквенное обозначение	Норма параметра			
		Предельно-допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
Длительность сигнала низкого уровня на линии PA1/T0CKI , нс прескалер включен при: U_{CC}=4,5 В	$t_{WL(PA1)}$	20	-	-	-
Параметры Таймера 1, 2					
Период сигнала на линии PD4/T1CLK, нс при: U_{CC}=4,5 В	$T_{C(PD)}$	$4 \bullet T_C+25$	-	-	-
Период сигнала на линии PD5/T2CLK, нс при: U_{CC}=4,5 В	$T_{C(PD)}$	$4 \bullet T_C+25$	-	-	-
Параметры АЦП					
Время выборки АЦП, мкс (интервал времени от включенияАЦП (ADON=1) до начала преобразования (GO=1)	t_{A_ADC}	2,0	-	-	-
Период тактовой частоты преобразования АЦП, мкс	t_{AO_ADC}	1,0	-	-	-
Параметры ЦАП					
Выходной ток ЦАП, мА	I_{DAC}	-	$\|\pm5\|$	-	$\|\pm10\|$
Параметры схем регистрации событий					
Длительность сигнала высокого уровня на линиях CAP1 и CAP2, нс при: U_{CC}=4,5 В	$t_{WH(PB,PG,PE)}$	10	-	-	-
Длительность сигнала низкого уровня на линиях CAP1 и CAP2, нс при: U_{CC}=4,5 В	$t_{WL(PB,PG,PE)}$	10	-	-	-
Период сигнала на линиях CAP1 и CAP2, нс при: U_{CC}=4,5 В	$T_{C(PB,PG,PE)}$	$2 \bullet (4 \bullet T_C)/N1$	-	-	-
Длительность фронта нарастания и спада входного сигнала на выводе OSC1 (15), нс при: f_C=35 МГц	τ_r τ_f	-	5	-	-
Емкость нагрузки, пФ	C_L	-	40	-	60

Примечание:

1. Не допускается одновременное воздействие двух и более предельных режимов.
2. n - в названии вывода - обозначает инверсию.
3. Напряжение питания АЦП (U_{CCA}) должно осуществляться от общего источника питания микросхемы.

Электрические параметры микросхемы

Таблица 87

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды (корпуса), °С
		не менее	не более	
Выходное напряжение низкого уровня, В, на выводах PA (29…32, 37, 38, 40, 41), PC (3…10, 26), PD (16, 17, 20…23) при: U_{CCA} =U_{CC}= 4,5 В, I_{OL}= 4,0 мА	U_{OL}	-	0,45	25, 125, минус 60
на выводе COMP_OUT (18) при: I_{OL}= 4,0 мА		-	0,45	
на выводе OSC2 (35) при: I_{OL}= 1,0 мА		-	0,45	
Выходное напряжение высокого уровня, В, на выводах PA (29…32, 37, 38, 40, 41), PC (3…10, 26), PD (16, 17, 20…23) при: U_{CCA}=U_{CC}= 4,5 В, I_{OH}= 4,0 мА	U_{OH}	4,05	-	25, 125, минус 60
на выводе COMP_OUT (18) при: I_{OH}= 4,0 мА		4,05	-	
на выводе OSC2 (35) при: I_{OH}= 1,0 мА		4,05	-	
Уровень напряжения для срабатывания схемы генерации сброса, В	U_{BOR}	3,6	4,3	25, 125, минус 60
Статический ток потребления в режиме покоя, память программ отключена, мкА, при: U_{CCA}= U_{CC}= 5,5 В	I_{CCS1}	-	40	25, 125, минус 60
Статический ток потребления в режиме покоя, память программ включена, мА, при: U_{CCA}= U_{CC}= 5,5 В	I_{CCS2}	-	1,5	25, 125, минус 60
Динамический ток потребления, мА, при: U_{CCA}= U_{CC}= 5,5 В, f_C= 24 МГц	I_{OCC}	-	50	25, 125, минус 60
Входной ток утечки высокого уровня, мкА, на выводах PA (29…32, 37, 38, 40, 41), PC (3…10, 26), PD (16, 17, 20…23), DAC1ref (43), DAC2ref (48), OSC1 (34), nMCLR (27), TEST (28) при: U_{CCA}= U_{CC}= 5,5 В, U_I= 5,5 В	I_{ILH}	-	|±1,0|*	25, 125, минус 60
Входной ток утечки низкого уровня, мкА, на выводах PA (29…32, 37, 38, 40, 41), PC (3…10, 26), PD (16, 17, 20…23), DAC1ref (43), DAC2ref (48), OSC1 (34), nMCLR (27), TEST (28) при: U_{CCA}= U_{CC}= 5,5 В, U_I= 0 В	I_{ILL}	-	|±1,0|*	25, 125, минус 60
Период работы внутреннего RC - генератора сторожевого таймера, мкс при: U_{CCA}= U_{CC}= 4,5 В	T_{RC}	60	140	25, 125, минус 60

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды (корпуса), °С
		не менее	не более	
Параметры универсального последовательного синхронно-асинхронного приемопере-датчика				
Время задержки данных RX/DT (31) от фронта TX/CK (32), нс при: U_{CCA}= U_{CC}= 4,5 В	$t_{d(TX-RX)}$	-	50	25, 125, минус 60
Параметры АЦП				
Дифференциальная нелинейность АЦП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,12 В	E_{DL_ADC}	-	|±1|	25, 125, минус 60
Интегральная нелинейность АЦП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,12 В	E_{IL_ADC}	-	|±2|	25, 125, минус 60
Разрешающая способность АЦП, разрядов при: U_{CCA}= U_{CC}= 5,12 В	E_{N_ADC}	-	12	25, 125, минус 60
Ошибка смещения АЦП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,12 В	E_{OFF_ADC}	-	|±5|	25, 125, минус 60
Параметры ЦАП				
Дифференциальная нелинейность ЦАП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,5 В	E_{DL_DAC}	-	|±1|	25, 125, минус 60
Интегральная нелинейность ЦАП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,5 В	E_{IL_DAC}	-	|±4|	25, 125, минус 60
Разрешающая способность ЦАП, разрядов при: U_{CCA}= U_{CC}= 5,5 В	E_{N_DAC}	-	12	25, 125, минус 60
Ошибка смещения ЦАП, единица младшего разряда при: U_{CCA}= U_{CC}= 5,5 В	E_{OFF_DAC}	-	|±10|	25, 125, минус 60
Диапазон выходного напряжения ЦАП, В, при: U_{CCA}= U_{CC}= 5,5 В	ΔU_{O_DAC}	0,01	5,1	25, 125, минус 60
Время включения ЦАП, мкс, при: U_{CCA}= U_{CC}= 5,5 В	t_{ON_DAC}	-	10	25, 125, минус 60
Параметры компаратора				
Напряжение внутреннего опорного источника компаратора, В	U_{REF_C}	1,04	1,36	25, 125, минус 60
Время включения компаратора, мкс, при: U_{CC}= 4,5 В	t_{ON_C}	-	8	25, 125, минус 60

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды (корпуса), °С
		не менее	не более	
Время задержки компаратора, нс, при: U_{CC}= 4,5 В, $\|U_{(PD6)} - U_{(PD7)}\|$ = 400мВ	t_{P_C}	-	125	25, 125, минус 60

Примечание

1 . n - в названии вывода - обозначает инверсию.

2 * - допускается измерение суммарного тока по всем выводам одновременно.

Справочные данные

Таблица 88

Наименование параметра, единица измерения, режим измерения	**Буквенное обозначение параметра**	**Норма параметра**		**Температура среды (корпуса), °С**
		не менее	**не более**	
Гистерезис напряжения компаратора, мВ	ΔU_{TH_C}	6	15	25, 125, минус 60
Напряжение смещения компаратора, мВ	U_{CM}		±15	25, 125, минус 60
Время установления выходных сигналов ЦАП, мкс при: U_{CCA}= U_{CC}= 5,5 В, 0 ≤U_{IN}≤ U_{CCA}, I_{DAC}= ±5 мА, C_L= 40 пФ	t_{SU_DAC}	-	10	25, 125, минус 60

Типовые зависимости

Зависимость частоты RC генератора от внешних времязадающих элементов(R, C) на входе OSC1.

Условия измерения: U_{CC}=5 В, T= 25°С

Таблица 89

CEXT	**REXT**	**fC**
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

Габаритный чертеж микросхемы

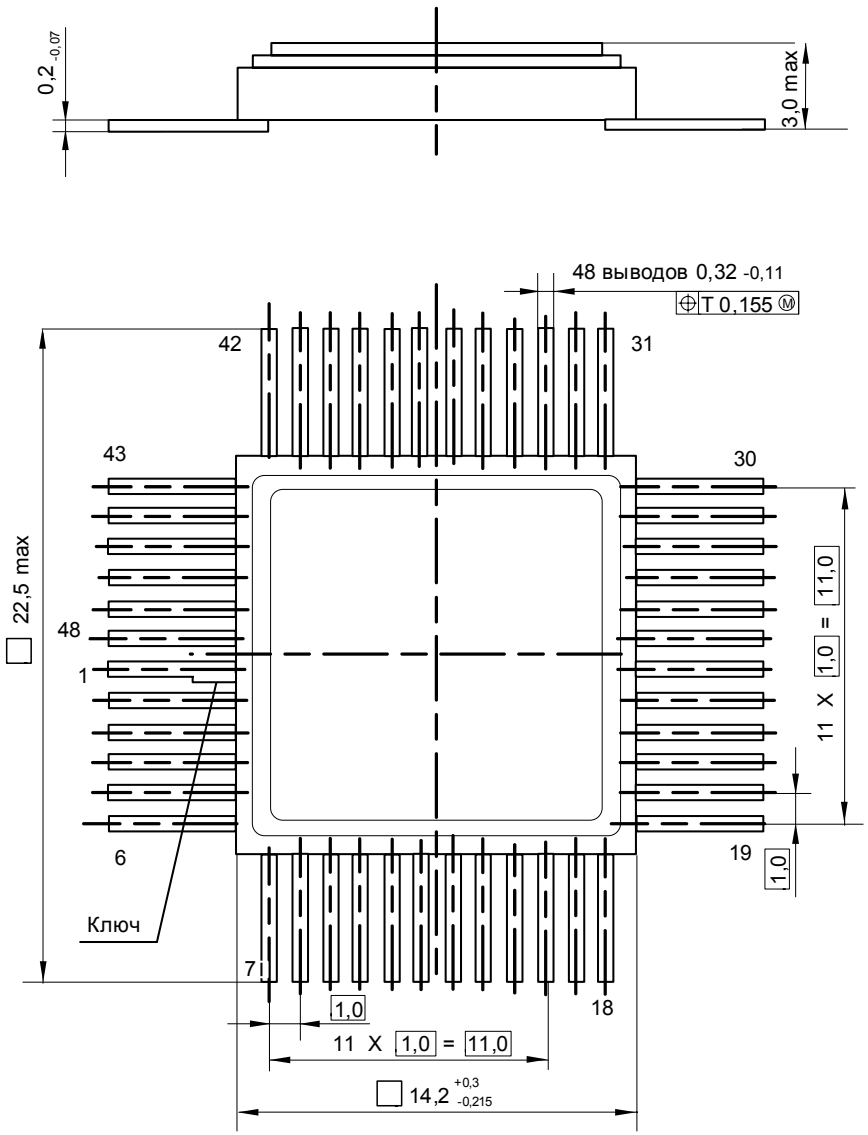


Рис. 51 Габаритный чертеж корпуса Н16.48-1В

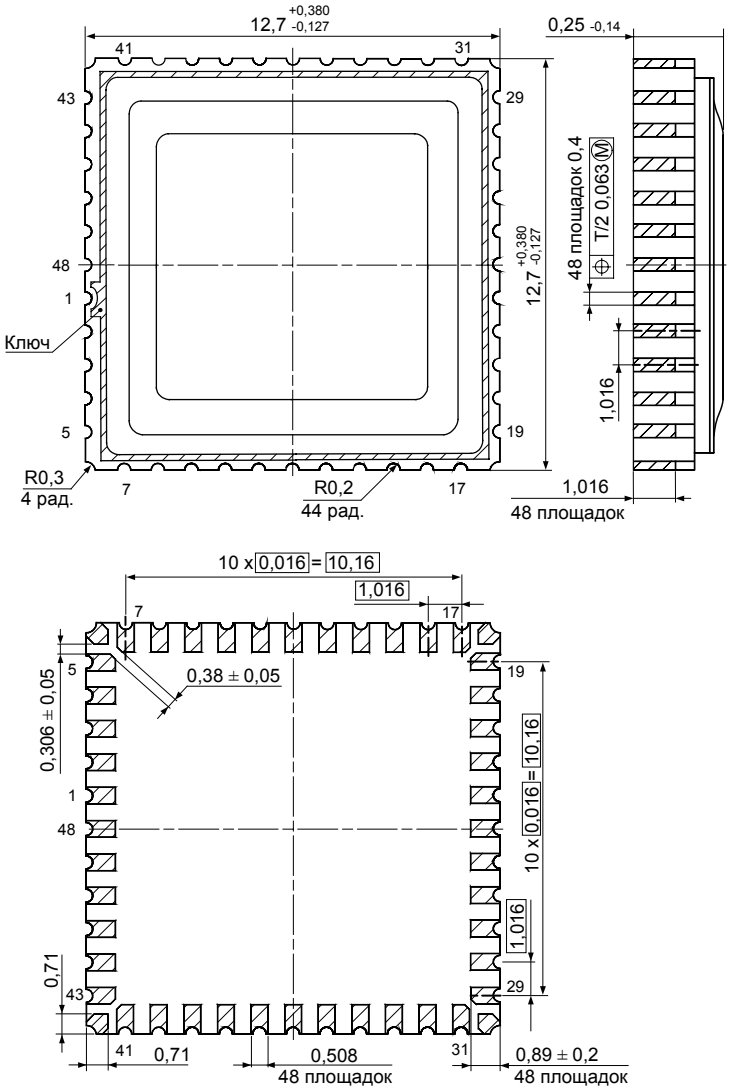


Рис. 52 Габаритный чертеж корпуса 5142.48-А

Информация для заказа

Обозначение	Маркировка	Тип корпуса	Температурный диапазон
1886ВЕ6У	1886ВЕ6У	Н16.48-1В	минус 60 – 125 °С
К1886ВЕ6У	К1886ВЕ6У	Н16.48-1В	минус 60 – 125 °С
К1886ВЕ6АУ	К1886ВЕ6У•	Н16.48-1В	0 – 70 °С
1886ВЕ6У1	1886ВЕ6У1	5142.48-А	минус 60 – 125 °С
К1886ВЕ6У1	К1886ВЕ6У1	5142.48-А	минус 60 – 125 °С

Лист регистрации изменений

№ п/п	Дата	Версия	Краткое содержание изменения	№№ изменяемых листов	№№ новых листов
1	03.02.2010	1.2	1. Исправлены названий выводов, регистров, битов, адресов. 2. Приведены в соответствие с ТУ табл. 116, 117 3. Введен лист регистрации изменений	2, 5, 8-23, 24, 25, 30 - 33, 34, 35, 36, 41-43, 47, 48, 54, 55, 59, 60 - 73, 79, 80, 83, 90 -92, 95, 96, 98, 100-110	146
2	12.04.2010	1.3	Корректировка на основании планового пересмотра документации	1 - 145	-
3	27.04.2010	1.4	Замена логотипа	1	-
4	02.09.2010	1.5	Приведение в соответствие с ТУ и ТЗ	1	-
5	22.10.2010	1.6	Приведение в соответствие с ТУ	7-9, 100, 138-144	Всего листов 145