PIC18FXX2

Однокристальные 8-разрядные FLASH CMOS микроконтроллеры с 10 – разрядным АЦП компании Microchip Technology Incorporated

- **PIC18F242**
- **PIC18F252**
- **PIC18F442**
- **PIC18F452**

Перевод основывается на технической документации DS39564A компании Microchip Technology Incorporated, USA.

PIC18FXX2
Data Sheet
High Performance, Enhanced FLASH Microcontrollers with 10-Bit A/D

28/40-выводные высокоскоростные FLASH микроконтроллеры с 10-разрядным АЦП

Высокоскоростной RISC микроконтроллер:

- Оптимизированная архитектура и система команд для написания программ на языке С
- Система команд совместима с командами семейств PIC16C, PIC17C и PIC18C
- Линейное адресное пространство памяти программ 32кбайта
- Линейное адресное пространство памяти данных 1.5кбайт

Устройство	Память программ		Память данных (байт)	EEPROM память данных (байт)
	Flash (байт)	Команд		
PIC18F242	16к	8192	768	256
PIC18F252	32к	16384	1536	256
PIC18F442	16к	8192	768	256
PIC18F452	32к	16384	1536	256

- Быстродействие до 10MIPS:
  - Тактовая частота от DC до 4МГц
  - Тактовая частота в режиме PLL от 4МГц до 10МГц
- 16-разрядные команды, 8-разрядные данные
- Система приоритетов прерываний
- Аппаратное умножение 8x8 за один машинный цикл

Характеристика периферийных модулей:

- Высокая нагрузочная способность портов ввода/ вывода
- Три входа внешних прерываний
- Модуль TMR0: 8/16-разрядный таймер/счетчик с программируемым 8-разрядным предделителем
- Модуль TMR1: 16-разрядный таймер/счетчик
- Модуль TMR2: 8-разрядный таймер/счетчик с 8-разрядным регистром периода (основной для ШИМ)
- Модуль TMR3: 16-разрядный таймер/счетчик
- Вторичный генератор тактового сигнала на основе TMR1/TMR3
- Два модуля CCP
  - Выводы модуля CCP могут работать как:
  - 16-разрядный захват, максимальная разрешающая способность 6.25нс (TCY/16)
  - 16-разрядное сравнение, максимальная разрешающая способность 100нс (TCY)
  - ШИМ, разрядность от 1 до 10 бит, Максимальная частота ШИМ 156кГц@8 бит; 39кГЦ@10 бит

Характеристика периферийных модулей (продолжение):

- Модуль ведущего последовательного синхронного порта (MSSP)
  - 3-х проводной интерфейс SPITM (поддерживает 4 режима)
  - I2CTM (ведущий и ведомый режим)
- Адресуемый модуль USART, поддержка интерфейса RS-485 и RS-232
- Модуль PSP, ведомый параллельный порт

Аналоговые периферийные модули:

- Модуль 10-разрядного АЦП:
  - Высокая скорость преобразования
  - Работа модуля АЦП в SLEEP режиме микроконтроллера
  - DNL = ±1Lsb, INL = ±1Lsb
- Программируемый детектор пониженного напряжения (PLVD)
  - При обнаружении снижения напряжения возможна генерация прерываний
- Программируемый сброс по снижению напряжения питания

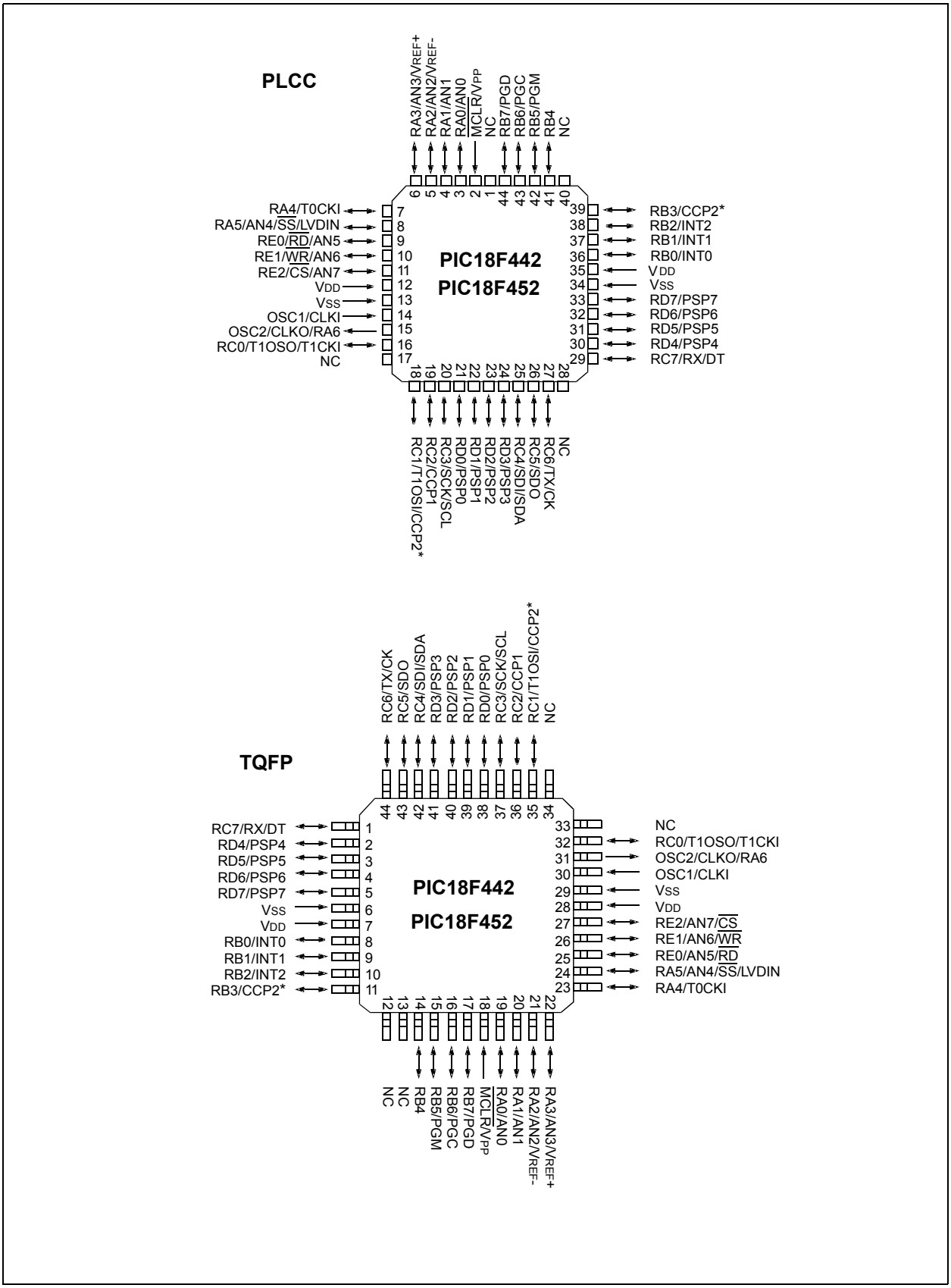
Особенности микроконтроллеров

- 100 000 гарантированных циклов стирание/запись памяти программ
- 1 000 000 гарантированных циклов стирание/запись EEPROM памяти данных
- Возможность самопрограммирования
- Сброс по включению питания (POR), таймер включения питания (PWRT), таймер запуска генератора (OST)
- Сторожевой таймер WDT с отдельным RC генератором
- Программируемая защита кода программы
- Режим пониженного энергопотребления и режим SLEEP
- Выбор режима работы тактового генератора, включая:
  - 4 x PLL (от основного генератора)
  - Вторичный генератор (32кГц)
- Внутрисхемное программирование по двухпроводной линии (ICSP) с одним напряжением питания 5В
- Внутрисхемная отладка по двухпроводной линии (ICD)

КМОП технология

- Высокоскоростная энергосберегающая КМОП технология
- Полностью статическая архитектура
- Широкий диапазон напряжений питания (от 2.0В до 5.5В)
- Промышленный и расширенный температурные диапазоны

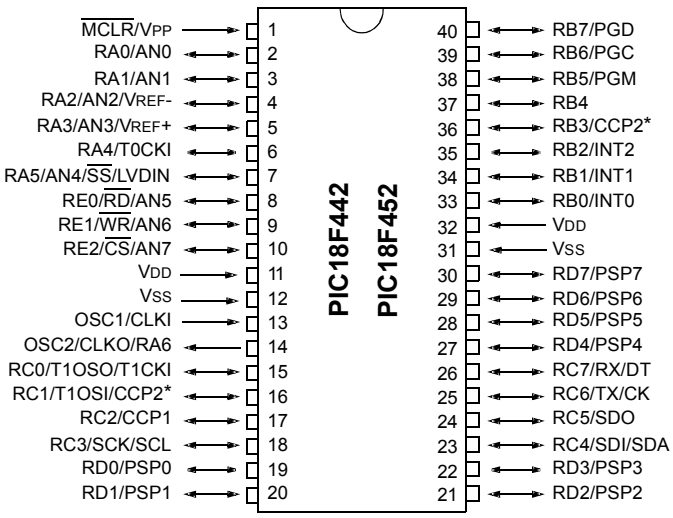
Расположение выводов



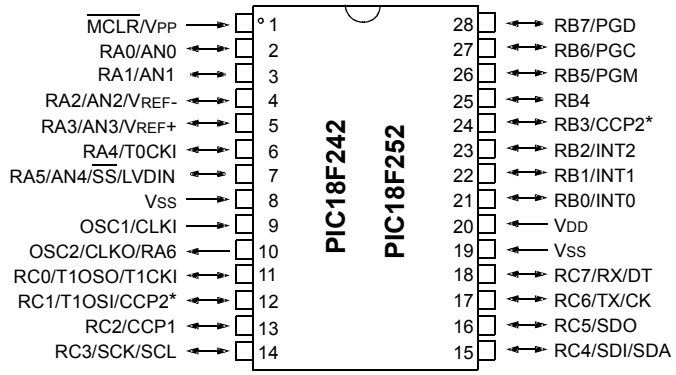
* RB3 – альтернативный вывод модуля CCP2

Расположение выводов (продолжение)

DIP



DIP, SOIC



Примечание. Микроконтроллеры PIC18F4X2 совместимы по выводам с 40-выводными PIC16C7X.

* RB3 – альтернативный вывод модуля CCP2

Содержание

1. Введение 10

2. Тактовый генератор 20

2.1 Режимы работы тактового генератора 20
2.2 Кварцевый/керамический резонатор 20
2.3 RC генератор 21
2.4 Внешний тактовый сигнал 22
2.5 HS/PLL 22
2.6 Переключение тактового генератора 23
2.6.1 Бит переключения тактового генератора 23
2.6.2 Переключение источника тактового сигнала 24
2.7 Влияние режима SLEEP на работу тактового генератора 26
2.8 Задержка старта после включения питания 26

3. Сброс 27

3.1 Сброс по включению питания POR 28
3.2 Таймер включения питания PWRT 28
3.3 Таймер запуска генератора OST 28
3.4 Таймер запуска PLL 28
3.5 Сброс по снижению напряжения питания BOR 28
3.6 Последовательность удержания микроконтроллера в состоянии сброса 29

4. Организация памяти 35

4.1 Организация памяти программ 35
4.2 Стек 36
4.2.1 Доступ к вершине стека 36
4.2.2 Указатель стека (регистр STKPTR) 36
4.2.3 Команды PUSH и POP 37
4.2.4 Сброс микроконтроллера при переполнении/исчерпании стека 37
4.3 Быстрые регистры стека 38
4.4 Регистры PCL, PCLATH и PCLATU 38
4.5 Синхронизация выполнения команд 38
4.6 Конвейерная выборка и выполнение команд 39
4.7 Размещение команд в памяти программ 39
4.7.1 Двухсловные команды 40
4.8 Таблицы 40
4.8.1 Вычисленный переход 40
4.8.2 Чтение/запись таблиц 40
4.9 Организация памяти данных 41
4.9.1 Регистры общего назначения GPR 41
4.9.2 Регистры специального назначения SFR 41
4.10 Банк памяти быстрого доступа 47
4.11 Регистр выбора банка памяти данных BSR 47
4.12 Косвенная адресация, регистры INDF и FSR 48
4.12.1 Операция косвенной адресации 48
4.13 Регистр STATUS 50
4.14 Регистр RCON 51

5. Flash память программ 52
5.1 Табличное чтение и табличная запись 52
5.2 Управляющие регистры 53
5.2.1 Регистры EECON1 и EECON2 53
5.2.2 Регистр TABLAT 55
5.2.3 Указатель таблицы, регистр TBLPTR 55
5.2.4 Границы указателя таблицы 55
5.3 Чтение Flash памяти программ 56
5.4 Стирание Flash памяти программ 57
5.4.1 Последовательность действий для стирания Flash памяти программ 57
5.5 Запись во Flash память программ 58
5.5.1 Последовательность записи во Flash память программ 58
5.5.2 Проверка записи 60
5.5.3 Выносливость ячеек памяти программ 60
5.5.4 Неожиданное завершение операции записи 60
5.5.5 Защита от случайной записи 60
5.6 Операции с Flash памятью программ при включенной защите кода 60

6. EEPROM память данных 61
6.1 Регистр EEADR 61
6.2 Регистры EECON1, EECON2 61
6.3 Чтение из EEPROM памяти данных 63
6.4 Запись в EEPROM память данных 63
6.5 Проверка записи 64
6.5.1 Выносливость ячеек EEPROM памяти данных 64
6.6 Защита от случайной записи 64
6.7 Операции с EEPROM памятью при включенной защите кода программы 64

7. Аппаратное умножение 8x8 65
7.1 Введение 65
7.2 Операции умножения 65

8. Прерывания 68
8.1 Регистры INTCON 70
8.2 Регистры PIR 73
8.3 Регистры PIE 75
8.4 Регистры IRP 77
8.5 Регистр RCON 79
8.6 Внешние прерывания INT 80
8.7 Прерывание от TMR0 80
8.8 Прерывание по изменению сигнала на входах PORTB 80
8.9 Сохранение контекста 80

9. Порты ввода/вывода 81
9.1 Регистры PORTA, TRISA, LATA 81
9.2 Регистры PORTB, TRISB, LATB 84
9.3 Регистры PORTC, TRISC, LATC 88
9.4 Регистры PORTD, TRISD, LATD 90
9.5 Регистры PORTE, TRISE, LATE 92
9.6 Ведомый параллельный порт PSP 95

10. Модуль таймера TMR0 97
10.1 Работа таймера TMR0 98
10.2 Предделитель 99
10.2.1 Переключение предделителя 99
10.3 Прерывание от TMR0 99
10.4 Чтение и запись таймера в 16-разрядном режиме 99

11. Модуль таймера TMR1 100
11.1 Работа таймера TMR1 101
11.2 Генератор TMR1 102
11.3 Прерывания от TMR1 102
11.4 Сброс TMR1 триггером модуля CCP 102
11.5 Чтение и запись таймера в 16-разрядном режиме 102

12. Модуль таймера TMR2 104
12.1 Работа таймера TMR2 104
12.2 Прерывания от TMR2 105
12.3 Выход TMR2 105

13. Модуль таймера TMR3 106
13.1 Работа таймера TMR3 107
13.2 Генератор TMR1 108
13.3 Прерывания от TMR3 108
13.4 Сброс TMR3 триггером модуля CCP 108

14. CCP модуль (Захват/Сравнение/ШИМ) 109
14.1 Модуль CCP1 110
14.2 Модуль CCP2 110
14.3 Режим захвата 111
14.3.1 Настройка вывода модуля CCP 111
14.3.2 Настройка таймера TMR1/TMR3 111
14.3.3 Обработка прерываний 111
14.3.4 Предварительный счетчик событий модуля CCP 111
14.4 Режим сравнения 112
14.4.1 Настройка вывода модуля CCP 112
14.4.2 Настройка таймера TMR1/TMR3 112
14.4.3 Обработка прерываний 112
14.4.4 Триггер специального события 112
14.5 Режим ШИМ 114
14.5.1 Период ШИМ 115
14.5.2 Длительность импульса ШИМ 115
14.5.3 Последовательность настройки модуля CCP в ШИМ режиме 115

15. Модуль MSSP 117

15.1 Введение 117
15.2 Управляющие регистры 117
15.3 Режим SPI 117
15.3.1 Регистры 118
15.3.2 Работа модуля MSSP в режиме SPI 120
15.3.3 Настройка выводов в режиме SPI 121
15.3.4 Типовое включение 121
15.3.5 Режим ведущего SPI 122
15.3.6 Режим ведомого SPI 123
15.3.7 Выбор ведомого в режиме SPI 123
15.3.8 Работа в SLEEP режиме микроконтроллера 125
15.3.9 Эффект сброса 125
15.3.10 Совместимость режимов шины 125
15.4 Режим I^2C 126
15.4.1 Регистры 126
15.4.2 Работа модуля MSSP в режиме I^2C 130
15.4.3 Режим ведомого I2C 130
15.4.4 Удержание тактового сигнала 136
15.4.5 Поддержка общего вызова 140
15.4.6 Режим ведущего I2C 141
15.4.7 Генератор скорости обмена 143
15.4.8 Формирование бита START в режиме ведущего I^2C 145
15.4.9 Формирование бита повторный START в режиме ведущего I2C 146
15.4.10 Передача данных в режиме ведущего I^2C 147
15.4.11 Прием данных в режиме ведущего I2C 147
15.4.12 Формирование бита подтверждения в режиме ведущего I^2C 150
15.4.13 Формирование бита STOP в режиме ведущего I2C 151
15.4.14 Работа в SLEEP режиме 151
15.4.15 Эффект сброса 151
15.4.16 Режим конкуренции 152
15.4.17 Режим конкуренции, арбитраж и конфликты шины 152

16. Адресуемый универсальный синхронно-асинхронный приемопередатчик (USART). 157

16.1 Генератор скорости обмена USART BRG 160
16.1.1 Выборка 160
16.2 Асинхронный режим USART 163
16.2.1 Асинхронный передатчик USART 163
16.2.2 Асинхронный приемник USART 165
16.2.3 Настройка 9-разрядного асинхронного приема с детектированием адреса 165
16.3 Синхронный ведущий режим USART 167
16.3.1 Передача синхронного ведущего 167
16.3.2 Прием синхронного ведущего 169
16.4 Синхронный ведомый режим USART 170
16.4.1 Передача синхронного ведомого 170
16.4.2 Прием синхронного ведомого 171

17. Модуль АЦП 172

17.1 Временные требования к подключению канала АЦП 175
17.2 Выбор источника тактовых импульсов для АЦП 177
17.3 Настройка аналоговых входов 177
17.4 Аналого-цифровое преобразование 178
17.5 Выравнивание результата преобразования 178
17.6 Использование триггера CCP2 179

18. Детектор пониженного напряжения LVD 180

18.1 Регистр управления 182

18.2 Работа модуля LVD 183

18.2.1 Внутренний источник опорного напряжения 184

18.2.2 Ток потребления 184

18.3 Работа модуля LVD в SLEEP режиме 184

18.4 Эффект сброса 184

19. Особенности микроконтроллеров PIC18FXX2 185

19.1 Биты конфигурации 185

19.2 Сторожевой таймер WDT 193

19.2.1 Регистр управления 193

19.2.2 Постделитель WDT 194

19.3 Режим энергосбережения SLEEP 195

19.3.1 Выход из режима SLEEP 195

19.3.2 Выход из режима SLEEP по прерыванию 195

19.4 Верификация и защита кода программы 197

19.4.1 Защита памяти программ 198

19.4.2 Защита EEPROM памяти данных 200

19.4.3 Защита регистров конфигурации 200

19.5 Размещение идентификатора ID 200

19.6 Внутрисхемное программирование ICSP 200

19.7 Внутрисхемный отладчик ICD 200

19.8 Режим низковольтного программирования 200

20. Описание системы команд 201

20.1 Подробное описание команд 206

21. Поддержка разработчиков 256

21.1 Интегрированная среда проектирования MPLAB-IDE 256

21.2 Ассемблер MPASM 257

21.3 С компиляторы MPLAB-C17 и MPLAB-C18 257

21.4 Линкер MPLINK, организатор библиотек MPLIB 257

21.5 Программный симулятор MPLAB-SIM 257

21.6 Универсальный эмулятор MPLAB-ICE 257

21.7 Внутрисхемный эмулятор ICEPIC 258

21.8 Внутрисхемный отладчик MPLAB-ICD 258

21.9 Универсальный программатор PRO MATE II 258

21.10 Программатор PICSTART Plus 258

21.11 Демонстрационная плата PICDEM-1 258

21.12 Демонстрационная плата PICDEM-2 258

21.13 Демонстрационная плата PICDEM-3 259

21.14 Демонстрационная плата PICDEM-17 259

21.15 KeeLoq (с функциями программатора) 259

22. Электрические характеристики 261

22.1 Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I 263

22.2 Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I 266

22.3 Временные диаграммы и спецификации 269

22.3.1 Символьное обозначение временных параметров 269

22.3.2 Условия временных диаграмм и параметров 270

22.3.3 Временные диаграммы и параметры 271

23. Характеристика микроконтроллеров 288

24. Корпуса микроконтроллеров 289

24.1 Описание обозначений на корпусах микроконтроллеров 289

24.2 Чертежи корпусов 291

24.2.1 Тип корпуса: 28-выводный PDIP - 300mil 291

24.2.2 Тип корпуса: 28-выводный SOIC - 300mil 292

24.2.3 Тип корпуса: 40-выводный PDIP - 600mil 293

24.2.4 Тип корпуса: 44-выводный TQFP 294

24.2.5 Тип корпуса: 44-выводный PLCC 295

24.3 Правила идентификации типа микроконтроллеров PIC18FXX2 296

1. Введение

В этом документе представлена информация по следующим микроконтроллерам:

1. PIC18F242
2. PIC18F252
3. PIC18F442
4. PIC18F452

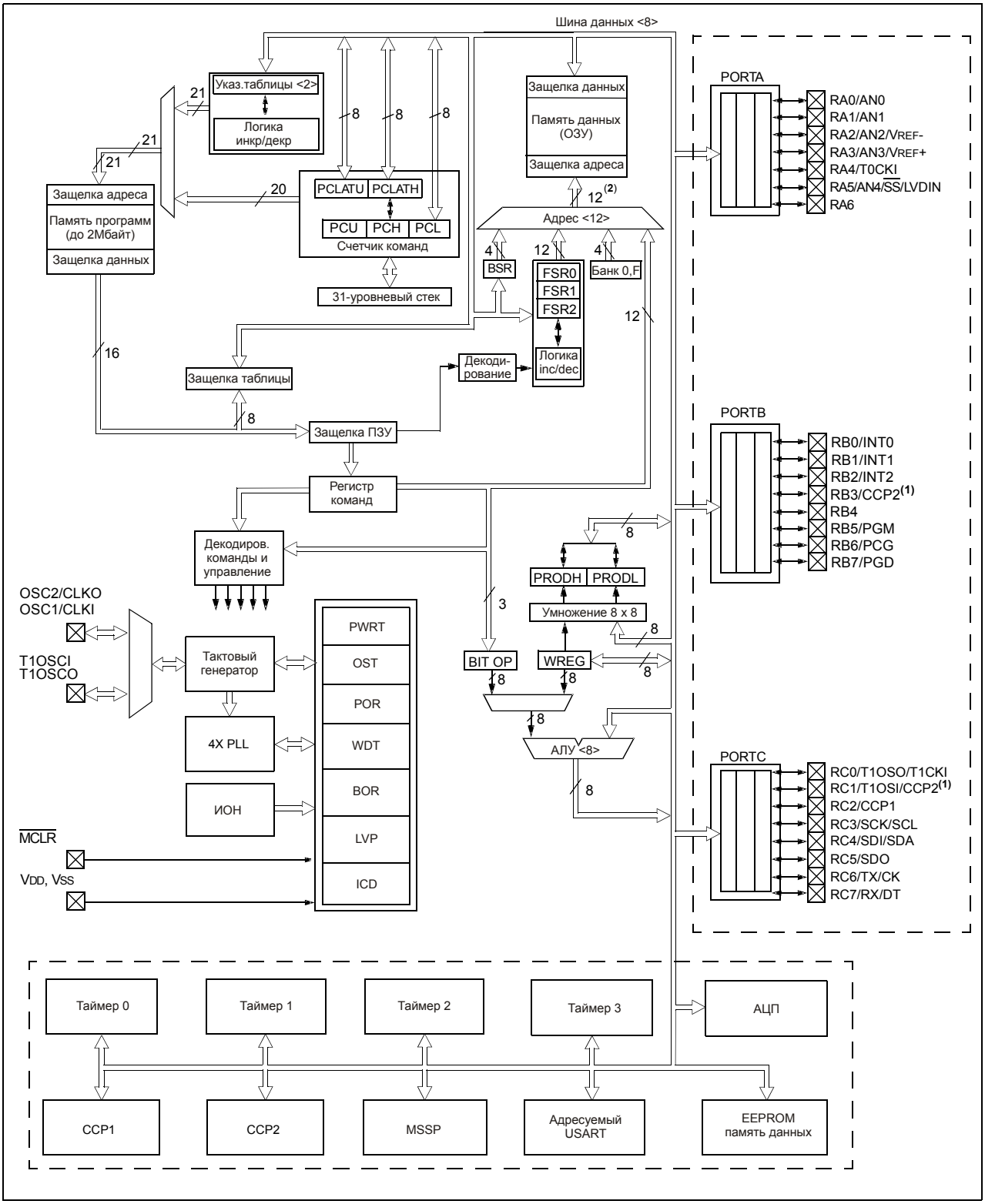
Микроконтроллеры выпускаются в 28-выводных и 40/44-выводных корпусах. 28-выводные микроконтроллеры не имеют модуля ведомого параллельного порта (PSP), а число реализованных входных каналов АЦП 5. Предварительную информацию смотрите в таблице 1-1.

На рисунке 1-1 представлена структурная схема 28-выводных микроконтроллеров, а на рисунке 1-2 показана структурная схема 40-выводных микроконтроллеров. В таблицах 1-2 и 1-3 соответственно представлено назначение выводов 28-выводных и 40-выводных микроконтроллеров.

Таблица 1-1. Основные характеристики микроконтроллеров

Параметр	**PIC18F242**	**PIC18F252**	**PIC18F442**	**PIC18F452**
Тактовая частота	DC-40МГц	DC-40МГц	DC-40МГц	DC-40МГц
Память программ (байт)	16К	32К	16К	32К
Память программ (команд)	8192	16384	8192	16384
Память данных (байт)	768	1536	768	1536
EEPROM память данных (байт)	256	256	256	256
Источников прерываний	17	17	17	17
Порты ввода/вывода	PORT A, B, C	PORT A, B, C	PORT A, B, C, D, E	PORT A, B, C, D, E
Таймеры	4	4	4	4
Модуль CCP	2	2	2	2
Последовательные интерфейсы	MSSP, адресуемый USART	MSSP, адресуемый USART	MSSP, адресуемый USART	MSSP, адресуемый USART
Параллельные интерфейсы	-	-	PSP	PSP
Модуль 10-разрядного АЦП	5 каналов	5 каналов	8 каналов	8 каналов
Сброс	POR, BOR, команда RESET, переполнение стека, исчерпание стека (PWRT, OST)	POR, BOR, команда RESET, переполнение стека, исчерпание стека (PWRT, OST)	POR, BOR, команда RESET, переполнение стека, исчерпание стека (PWRT, OST)	POR, BOR, команда RESET, переполнение стека, исчерпание стека (PWRT, OST)
Программируемый детектор пониженного напряжения	Есть	Есть	Есть	Есть
Программируемый сброс по снижению напряжения питания (BOR)	Есть	Есть	Есть	Есть
Команд микроконтроллера	75	75	75	75
Корпус	28DIP 28SOIC	28DIP 28SOIC	40DIP 44PLCC 44TQFP	40DIP 44PLCC 44TQFP

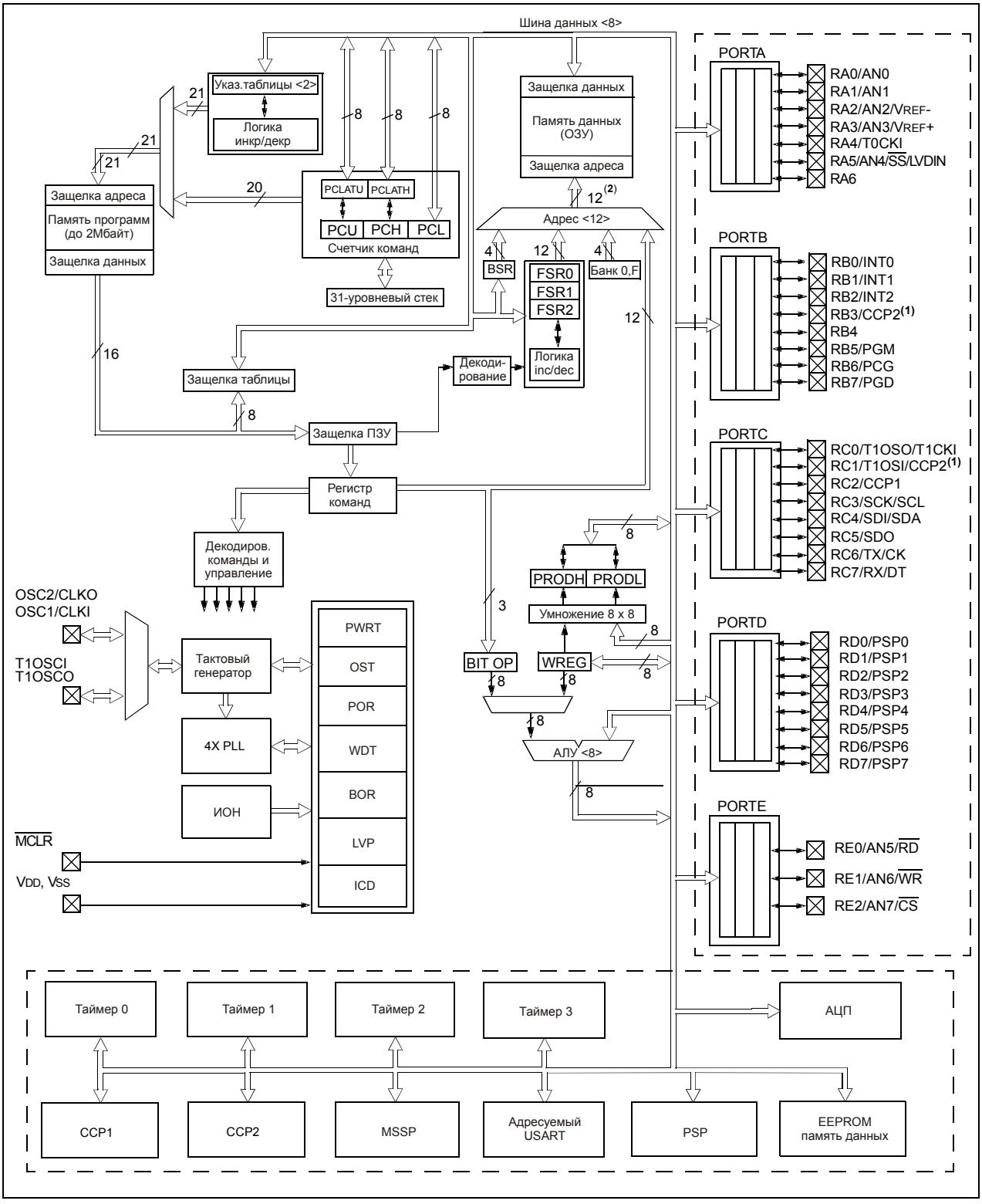
Рисунок 1-1. Структурная схема микроконтроллеров PIC18F2X2



Примечания:

1. Подключение вывода CCP2 к каналу порта ввода/вывода RB3 определяется битом конфигурации микроконтроллера.
2. Старшие биты адреса регистра ОЗУ берутся из регистра BSR (кроме команды MOVFF).
3. Большинство каналов ввода/вывода мультиплицируются с выводами периферийных модулей.

Рисунок 1-2. Структурная схема микроконтроллеров PIC18F4X2



Примечания:

1. Подключение вывода CCP2 к каналу порта ввода/вывода RB3 определяется битом конфигурации микроконтроллера.
2. Старшие биты адреса регистра ОЗУ берутся из регистра BSR (кроме команды MOVFF).
3. Большинство каналов ввода/вывода мультиплицируются с выводами периферийных модулей.

Таблица 1-2. Назначение выводов в микроконтроллерах PIC18F2X2

Обозначение	Номер вывода		Тип вывода	Тип буфера	Описание
	DIP	SOIC			
-MCLR/V_{PP}	1	1			Вход сброса микроконтроллера или напряжение программирования
-MCLR			I	ST	Вход сброса микроконтроллера. Активный низкий логический уровень.
V_{PP}			P	-	Вход напряжения программирования.
OSC1/CLKIN	9	9			Кварцевый резонатор или вход внешнего тактового сигнала.
OSC1			I	ST	Вход для подключения кварцевого резонатора или внешнего тактового сигнала. ST буфер в RC режиме тактового генератора, CMOS в остальных режимах.
CLKIN			I	CMOS	Вход внешнего тактового сигнала. Всегда связан с функциями OSC1 (см. OSC1/CLKIN, OSC2/CLKO).
OSC2/CLKO/RA6	10	10			Кварцевый резонатор или выход тактового сигнала.
OSC2			O	-	Выход для подключения кварцевого резонатора в режиме кварцевого резонатора тактового генератора.
CLKO			O	-	В RC режиме тактового генератора на выводе CLKO присутствует сигнал с частотой $F_{OSC}/4$, синхронный выполнению команд микроконтроллером.
RA6			I/O	TTL	Канал порта ввода/вывода.
					PORTA – двунаправленный порт ввода/вывода.
RA0/AN0	2	2			
RA0			I/O	TTL	Цифровой канал порта ввода/вывода.
AN0			I	AN	Аналоговый вход 0.
RA1/AN1	3	3			
RA1			I/O	TTL	Цифровой канал порта ввода/вывода.
AN1			I	AN	Аналоговый вход 1.
RA2/AN2/V_{REF-}	4	4			
RA2			I/O	TTL	Цифровой канал порта ввода/вывода.
AN2			I	AN	Аналоговый вход 2.
V_{REF-}			I	AN	Вход опорного напряжения АЦП.
RA3/AN3/V_{REF+}	5	5			
RA3			I/O	TTL	Цифровой канал порта ввода/вывода.
AN3			I	AN	Аналоговый вход 3.
V_{REF+}			I	AN	Вход опорного напряжения АЦП.
RA4/T0CKI	6	6			
RA4			I/O	ST/OD	Цифровой канал порта ввода/вывода. Выход с открытым коллектором.
T0CKI			I	ST	Вход тактового сигнала для TMR0.
RA5/AN4/-SS/LVDIN	7	7			
RA5			I/O	TTL	Цифровой канал порта ввода/вывода.
AN4			I	AN	Аналоговый вход 4.
-SS			I	ST	Вход выбора ведомого SPI.
LVDIN			I	AN	Вход детектора пониженного напряжения.
RA6					Смотрите OSC2/CLKO/RA6.

Обозначения:

TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-2. Назначение выводов в микроконтроллерах PIC18F2X2 (продолжение)

Обозначение	Номер вывода		Тип вывода	Тип буфера	Описание
	DIP	SOIC			
					PORTB – двунаправленный порт ввода/вывода. На всех входах PORTB могут быть программно включены подтягивающие резисторы.
RB0/INT0	21	21			
RB0			I/O	TTL	Цифровой канал порта ввода/вывода.
INT0			I	ST	Вход внешнего прерывания 0.
RB1/INT1	22	22			
RB1			I/O	TTL	Цифровой канал порта ввода/вывода.
INT1			I	ST	Вход внешнего прерывания 1.
RB2/INT2	23	23			
RB2			I/O	TTL	Цифровой канал порта ввода/вывода.
INT2			I	ST	Вход внешнего прерывания 2.
RB3/CCP2	24	24			
RB3			I/O	TTL	Цифровой канал порта ввода/вывода.
CCP2			I/O	ST	Вход захвата 2, выход сравнения 2, выход ШИМ 2.
RB4	25	25	I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
RB5/PGM	26	26			
RB5			I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGM			I	ST	Включение режима низковольтного программирования ICSP.
RB6/PGC	27	27			
RB6			I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGC			I	ST	Вход тактового сигнала для внутрисхемной отладки и программирования ICSP
RB7/PGD	28	28			
RB7			I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGD			I	ST	Вывод данных для внутрисхемной отладки и программирования ICSP.

Обозначения:
TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-2. Назначение выводов в микроконтроллерах PIC18F2X2 (продолжение)

Обозначение	Номер вывода		Тип вывода	Тип буфера	Описание
	DIP	SOIC			
					PORTC – двунаправленный порт ввода/вывода.
RC0/T1OSO/T1CKI	11	11			
RC0			I/O	ST	Цифровой канал порта ввода/вывода.
T1OSO			O	-	Выход для подключения кварцевого резонатора TMR1.
T1CKI			I	ST	Вход тактового сигнала для TMR1/TMR3
RC1/T1OSI/CCP2	12	12			
RC1			I/O	ST	Цифровой канал порта ввода/вывода.
T1OSI			I	CMOS	Вход для подключения кварцевого резонатора TMR1.
CCP2			I/O	ST	Вход захвата 2, выход сравнения 2, выход ШИМ.
RC2/CCP1	13	13			
RC2			I/O	ST	Цифровой канал порта ввода/вывода.
CCP1			I/O	ST	Вход захвата 1, выход сравнения 1, выход ШИМ 1
RC3/SCK/SCL	14	14			
RC3			I/O	ST	Цифровой канал порта ввода/вывода.
SCK			I/O	ST	Вход/выход тактового сигнала в режиме SPI.
SCL			I/O	ST	Вход/выход тактового сигнала в режиме I^2C.
RC4/SDI/SDA	15	15			
RC4			I/O	ST	Цифровой канал порта ввода/вывода.
SDI			I	ST	Вход данных в режиме SPI.
SDA			I/O	ST	Вход/выход данных в режиме I^2C.
RC5/SDO	16	16			
RC5			I/O	ST	Цифровой канал порта ввода/вывода.
SDO			O	-	Выход данных в режиме SPI.
RC6/TX/CK	17	17			
RC6			I/O	ST	Цифровой канал порта ввода/вывода.
TX			O	-	Выход передатчика USART в асинхронном режиме.
CK			I/O	ST	Вывод синхронизации в синхронном режиме USART.
RC7/RX/DT	18	18			
RC7			I/O	ST	Цифровой канал порта ввода/вывода.
RX			I	ST	Вход приемника USART в асинхронном режиме.
DT			I/O	ST	Вывод данных USART в синхронном режиме.
V_{SS}	8, 19	8, 19	P	-	Общий вывод для логики ядра и портов ввода/вывода.
V_{DD}	20	20	P	-	Напряжение питания для логики ядра и портов ввода/вывода.

Обозначения:

TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-3. Назначение выводов в микроконтроллерах PIC18F4X2

Обозначение	Номер вывода			Тип вывода	Тип буфера	Описание
	DIP	PLCC	TQFP			
-MCLR/V_{PP}	1	2	18			Вход сброса микроконтроллера или напряжение программирования
-MCLR				I	ST	Вход сброса микроконтроллера. Активный низкий логический уровень.
V_{PP}				P	-	Вход напряжения программирования.
NC	-	1,17, 28,40	12,13, 33,34	-	-	Эти выводы внутри микросхемы не подключены.
OSC1/CLKIN	13	14	30			Кварцевый резонатор или вход внешнего тактового сигнала.
OSC1				I	ST	Вход для подключения кварцевого резонатора или внешнего тактового сигнала. ST буфер в RC режиме тактового генератора, CMOS в остальных режимах.
CLKIN				I	CMOS	Вход внешнего тактового сигнала. Всегда связан с функциями OSC1 (см. OSC1/CLKIN, OSC2/CLKO).
OSC2/CLKO/RA6	14	15	31			Кварцевый резонатор или выход тактового сигнала.
OSC2				O	-	Выход для подключения кварцевого резонатора в режиме кварцевого резонатора тактового генератора.
CLKO				O	-	В RC режиме тактового генератора на выводе CLKO присутствует сигнал с частотой $F_{OSC}/4$, синхронный выполнению команд микроконтроллером.
RA6				I/O	TTL	Канал порта ввода/вывода.
						PORTA – двунаправленный порт ввода/вывода.
RA0/AN0	2	3	19			
RA0				I/O	TTL	Цифровой канал порта ввода/вывода.
AN0				I	AN	Аналоговый вход 0.
RA1/AN1	3	4	20			
RA1				I/O	TTL	Цифровой канал порта ввода/вывода.
AN1				I	AN	Аналоговый вход 1.
RA2/AN2/V_{REF-}	4	5	21			
RA2				I/O	TTL	Цифровой канал порта ввода/вывода.
AN2				I	AN	Аналоговый вход 2.
V_{REF-}				I	AN	Вход опорного напряжения АЦП.
RA3/AN3/V_{REF+}	5	6	22			
RA3				I/O	TTL	Цифровой канал порта ввода/вывода.
AN3				I	AN	Аналоговый вход 3.
V_{REF+}				I	AN	Вход опорного напряжения АЦП.
RA4/T0CKI	6	7	23			
RA4				I/O	ST/OD	Цифровой канал порта ввода/вывода. Выход с открытым коллектором.
T0CKI				I	ST	Вход тактового сигнала для TMR0.
RA5/AN4/-SS/LVDIN	7	8	24			
RA5				I/O	TTL	Цифровой канал порта ввода/вывода.
AN4				I	AN	Аналоговый вход 4.
-SS				I	ST	Вход выбора ведомого SPI.
LVDIN				I	AN	Вход детектора пониженного напряжения.
RA6						Смотрите OSC2/CLKO/RA6.

Обозначения:
TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-3. Назначение выводов в микроконтроллерах PIC18F4X2 (продолжение)

Обозначение	Номер вывода			Тип вывода	Тип буфера	Описание
	DIP	PLCC	TQFP			
						PORTB – двунаправленный порт ввода/вывода. На всех входах PORTB могут быть программно включены подтягивающие резисторы.
RB0/INT0	33	36	8			
RB0				I/O	TTL	Цифровой канал порта ввода/вывода.
INT0				I	ST	Вход внешнего прерывания 0.
RB1/INT1	34	37	9			
RB1				I/O	TTL	Цифровой канал порта ввода/вывода.
INT1				I	ST	Вход внешнего прерывания 1.
RB2/INT2	35	38	10			
RB2				I/O	TTL	Цифровой канал порта ввода/вывода.
INT2				I	ST	Вход внешнего прерывания 2.
RB3/CCP2	36	39	11			
RB3				I/O	TTL	Цифровой канал порта ввода/вывода.
CCP2				I/O	ST	Вход захвата 2, выход сравнения 2, выход ШИМ 2.
RB4	37	41	14	I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
RB5/PGM	38	42	15			
RB5				I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGM				I	ST	Включение режима низковольтного программирования ICSP.
RB6/PGC	39	43	16			
RB6				I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGC				I	ST	Вход тактового сигнала для внутрисхемной отладки и программирования ICSP
RB7/PGD	40	44	17			
RB7				I/O	TTL	Цифровой канал порта ввода/вывода. Прерывания по изменению уровня сигнала на входе.
PGD				I	ST	Вывод данных для внутрисхемной отладки и программирования ICSP.

Обозначения:
TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-3. Назначение выводов в микроконтроллерах PIC18F4X2 (продолжение)

Обозначение	Номер вывода			Тип вывода	Тип буфера	Описание
	DIP	PLCC	TQFP			
						PORTC – двунаправленный порт ввода/вывода.
RC0/T1OSO/T1CKI	15	16	32			
RC0				I/O	ST	Цифровой канал порта ввода/вывода.
T1OSO				O	-	Выход для подключения кварцевого резонатора TMR1.
T1CKI				I	ST	Вход тактового сигнала для TMR1/TMR3
RC1/T1OSI/CCP2	16	18	35			
RC1				I/O	ST	Цифровой канал порта ввода/вывода.
T1OSI				I	CMOS	Вход для подключения кварцевого резонатора TMR1.
CCP2				I/O	ST	Вход захвата 2, выход сравнения 2, выход ШИМ.
RC2/CCP1	17	19	36			
RC2				I/O	ST	Цифровой канал порта ввода/вывода.
CCP1				I/O	ST	Вход захвата 1, выход сравнения 1, выход ШИМ 1
RC3/SCK/SCL	18	20	37			
RC3				I/O	ST	Цифровой канал порта ввода/вывода.
SCK				I/O	ST	Вход/выход тактового сигнала в режиме SPI.
SCL				I/O	ST	Вход/выход тактового сигнала в режиме I^2C.
RC4/SDI/SDA	23	25	42			
RC4				I/O	ST	Цифровой канал порта ввода/вывода.
SDI				I	ST	Вход данных в режиме SPI.
SDA				I/O	ST	Вход/выход данных в режиме I^2C.
RC5/SDO	24	26	43			
RC5				I/O	ST	Цифровой канал порта ввода/вывода.
SDO				O	-	Выход данных в режиме SPI.
RC6/TX/CK	25	27	44			
RC6				I/O	ST	Цифровой канал порта ввода/вывода.
TX				O	-	Выход передатчика USART в асинхронном режиме.
CK				I/O	ST	Вывод синхронизации в синхронном режиме USART.
RC7/RX/DT	26	29	1			
RC7				I/O	ST	Цифровой канал порта ввода/вывода.
RX				I	ST	Вход приемника USART в асинхронном режиме.
DT				I/O	ST	Вывод данных USART в синхронном режиме.

Обозначения:

TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

Таблица 1-3. Назначение выводов в микроконтроллерах PIC18F4X2 (продолжение)

Обозначение	Номер вывода			Тип вывода	Тип буфера	Описание
	DIP	PLCC	TQFP			
						PORTD – двунаправленный порт ввода/вывода или параллельный ведомый порт для подключения к шине микропроцессора. В режиме PSP подключены входные буферы ТТЛ.
RD0/PSP0	19	21	38	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD1/PSP1	20	22	39	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD2/PSP2	21	23	40	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD3/PSP3	22	24	41	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD4/PSP4	27	30	2	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD5/PSP5	28	31	3	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD6/PSP6	29	32	4	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
RD7/PSP7	30	33	5	I/O	ST TTL	Цифровой канал порта ввода/вывода. Данные параллельного ведомого порта.
						PORTE – двунаправленный порт ввода/вывода.
RE0/-RD/AN5	8	9	25			
RE0				I/O	ST	Цифровой канал порта ввода/вывода.
-RD				I	TTL	Вход сигнала чтения ведомого параллельного порта (см. –WR и –CS).
AN5				I	AN	Аналоговый вход 5.
RE1/-WR/AN6	9	10	26			
RE1				I/O	ST	Цифровой канал порта ввода/вывода.
-WR				I	TTL	Вход сигнала записи в ведомый параллельный порт (см. –RD и –CS).
AN6				I	AN	Аналоговый вход 6.
RE2/-CS/AN7	10	11	27			
RE2				I/O	ST	Цифровой канал порта ввода/вывода.
-CS				I	TTL	Вход сигнала выбора ведомого параллельного порта (см. –RD и –WR).
AN7				I	AN	Аналоговый вход 7.
V_{SS}	12,31	13,34	6,29	P	-	Общий вывод для логики ядра и портов ввода/вывода.
V_{DD}	11,32	12,35	7,28	P	-	Напряжение питания для логики ядра и портов ввода/вывода.

Обозначения:

TTL = ТТЛ совместимый вход
ST = вход с триггером Шмитта и КМОП уровнями
O = выход
OD = выход с открытым коллектором (нет диода, подключенного к V_{DD})
CMOS = КМОП совместимый вход/выход
I = вход
P = питание
AN = аналоговый вход

2. Тактовый генератор

2.1 Режимы работы тактового генератора

Тактовый генератор PIC18FXX2 может работать в восьми режимах. Пользователь может выбрать один из восьми режимов тактового генератора в битах конфигурации микроконтроллера (FOSC2, FOSC1 и FOSC0):

1. LP – низкочастотный кварцевый резонатор (малое энергопотребление)
2. XT – кварцевый/керамический резонатор
3. HS – высокочастотный кварцевый/керамический резонатор
4. HS+PLL – высокочастотный резонатор с включенным PLL модулем
5. RC – внешний резистор/конденсатор
6. RCIO – внешний резистор/конденсатор с включенным каналом порта ввода/вывода
7. EC – внешний тактовый сигнал
8. ECIO – внешний тактовый сигнал с включенным каналом порта ввода/вывода

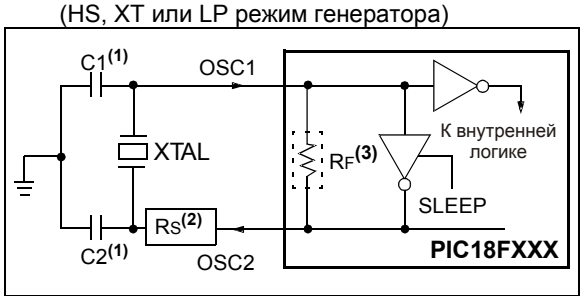
2.2 Кварцевый/керамический резонатор

Для формирования тактового сигнала в XT, LP, HS или HS+PLL режиме тактового генератора к выводам OSC1, OSC2 подключается кварцевый/керамический резонатор. Схему подключения смотрите на рисунке 2-1.

Для микроконтроллеров PIC18FXX2 необходимо использовать кварцевые/керамические резонаторы с параллельным резонатором.

Примечание. Использование резонаторов с последовательным резонансом может привести к получению тактовой частоты, не соответствующей параметрам резонатора.

Рисунок 2-1. Схема подключения кварцевого/керамического резонатора (HS, XT или LP режим генератора)



Примечания:

1. Смотрите таблицы 2-1 и 2-2 для выбора емкости конденсаторов C1, C2.
2. Для некоторых типов резонаторов может потребоваться последовательно включенный резистор R_S.
3. Значение сопротивления R_F зависит от выбранного режима тактового генератора.

Таблица 2-1. Емкость конденсаторов для керамического резонатора (оценочные значения)

Режим	Частота	C1	C2
XT	455кГц	68 – 100пФ	68 – 100пФ
	2.0МГц	15 – 68пФ	15 – 68пФ
	4.0МГц	15 – 68пФ	15 – 68пФ
HS	8.0МГц	10 – 68пФ	10 – 68пФ
	16.0МГц	10 – 22пФ	10 – 22пФ
Смотрите примечания к таблице.			
Протестированные резонаторы			
455кГц	Panasonic EFO-A455K04B	±0.3%	
2.0МГц	Murata Erie CSA2.00MG	±0.5%	
4.0МГц	Murata Erie CSA4.00MG	±0.5%	
8.0МГц	Murata Erie CSA8.00MT	±0.5%	
16.0МГц	Murata Erie CSA16.00MX	±0.5%	
Все резонаторы не имели паразитной емкости.			

Примечания:

1. Большая емкость увеличивает стабильность генератора, но увеличивается и время запуска.
2. При напряжении питания менее 3В или при использовании некоторых керамических резонаторов может быть необходимо использование HS режима тактового генератора для низкочастотных резонаторов (или использовать кварцевый резонатор).
3. Значения емкости конденсаторов, указанные в таблице, являются оценочными, т.к. каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного выбора внешних компонентов.

Таблица 2-2. Емкость конденсаторов для кварцевого резонатора (оценочные значения)

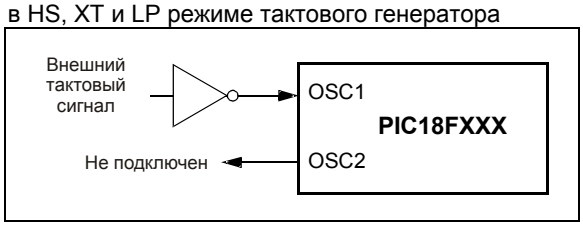
Режим	Частота	C1	C2
LP	32кГц 200кГц	33пФ 15пФ	33пФ 15пФ
XT	200кГц 1.0МГц 4.0МГц	47 – 68пФ 15пФ 15пФ	47 – 68пФ 15пФ 15пФ
HS	4.0МГц 8.0МГц 20МГц 25МГц	15пФ 15 – 33пФ 15 – 33пФ TBD	15пФ 15 – 33пФ 15 – 33пФ TBD
Смотрите примечания к таблице.			
Протестированные резонаторы			
32кГц	Epson C-001R32.768K-A		±20PPM
200кГц	STD XTL 200.000kHz		±20PPM
1.0МГц	ECS ECS-10-13-1		±50PPM
4.0МГц	ECS ECS-40-20-1		±50PPM
8.0МГц	Epson CA-301 8.000M-C		±30PPM
20МГц	Epson CA-301 20.000M-C		±30PPM
Все резонаторы не имели паразитной емкости.			

Примечания:

1. Большая емкость увеличивает стабильность генератора, но увеличивается и время запуска.
2. Последовательный резистор R_S может потребоваться в HS, XT режиме тактового генератора для предотвращения возбуждения резонатора на низкой частоте.
3. Значения емкости конденсаторов, указанные в таблице, являются оценочными, т.к. каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного выбора внешних компонентов.

В режимах HS, XT и LP микроконтроллер может работать от внешнего источника тактового сигнала, подключенного к выводу OSC1 (см. рисунок 2-2).

Рисунок 2-2. Подключение внешнего тактового сигнала в HS, XT и LP режиме тактового генератора

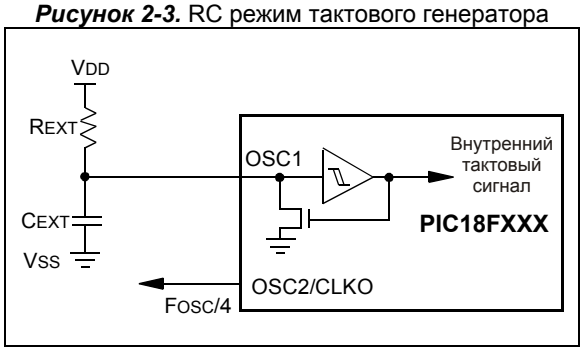


2.3 RC генератор

В приложениях, не требующих высокостабильной тактовой частоты, возможно использовать RC и RCIO режим тактового генератора, что уменьшает общую стоимость устройства. Частота RC генератора зависит от напряжения питания, сопротивления резистора (R_{EXT}), емкости конденсатора (C_{EXT}) и температуры. Дополнительно тактовая частота микроконтроллера будет варьироваться в небольших приделах из-за технологического разброса параметров Различные паразитные емкости также будут влиять на частоту тактового генератора, особенно при малой емкости C_{EXT}. Необходимо учитывать технологический разброс параметров внешних компонентов R_{EXT}, C_{EXT}. На рисунке 2-3 показана схема подключения RC цепочки.

В RC режиме тактового генератора на выводе OSC2 присутствует тактовый сигнал с частотой $F_{OSC}/4$. Этот сигнал может использоваться для синхронизации другой логики устройства.

Рисунок 2-3. RC режим тактового генератора



Рекомендованные значения: 3кОм ≤ R_{EXT} ≤ 100кОм

C_{EXT} > 20пФ

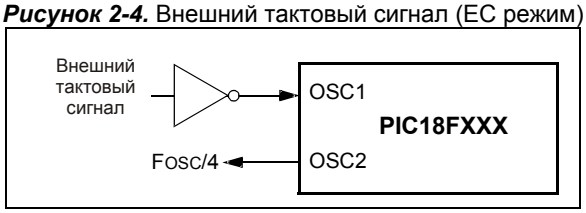
В RCIO режиме тактовый генератор работает также как и в RC режиме, но вывод OSC2 используется в качестве дополнительного канала ввода/вывода RA6, управляемый битом 6 в регистре PORTA.

2.4 Внешний тактовый сигнал

В EC и ECIO режимах тактового генератора внешний тактовый сигнал подается на вывод OSC1. Обратная связь между выводами OSC1 и OSC2 выключена, чтобы снизить энергопотребление. Нет задержки запуска тактового генератора при сбросе по включению питания (POR) и выходу микроконтроллера из режима SLEEP.

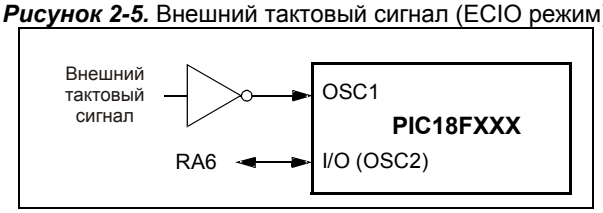
В EC режиме тактового генератора на выводе OSC2 присутствует тактовый сигнал с частотой $F_{OSC}/4$. Этот сигнал может использоваться с целью диагностики или синхронизации другой логики устройства. Схема включения в EC режиме тактового генератора показана на рисунке 2-4.

Рисунок 2-4. Внешний тактовый сигнал (EC режим)



В ECIO режиме тактовый генератор работает также как и в EC режиме, но вывод OSC2 используется в качестве дополнительного канала ввода/вывода RA6, управляемый битом 6 в регистре PORTA. Схема включения в ECIO режиме тактового генератора показана на рисунке 2-4.

Рисунок 2-5. Внешний тактовый сигнал (ECIO режим)



2.5 HS/PLL

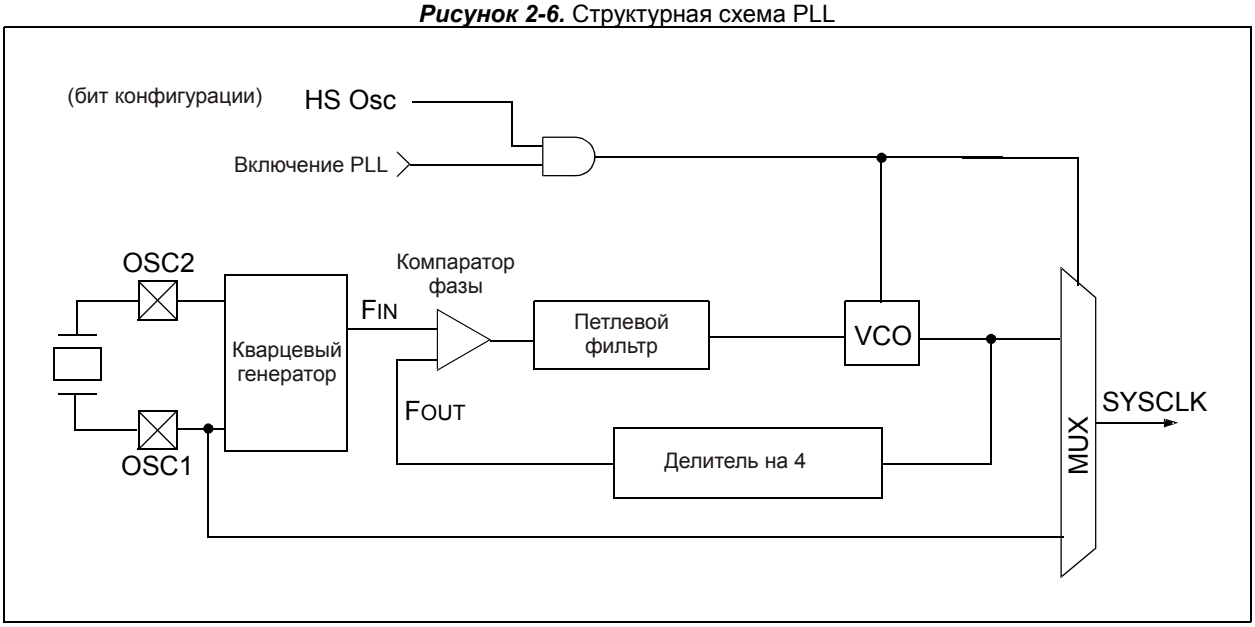
Внутренняя схема PLL (Phase Locked Loop), включаемая при программировании микроконтроллера, позволяет умножить тактовую частоту на 4. При входной тактовой частоте 10МГц внутренняя тактовая частота микроконтроллера будет 40МГц. Подобное решение имеет более высокую защищенность от электромагнитных помех по сравнению с использованием кварцевого резонатора высокой частоты.

Работа PLL возможна только в HS режиме тактового генератора. В других режимах тактового генератора работа схемы PLL заблокирована, внутренний тактовый сигнал будет сниматься непосредственно с вывода OSC1.

PLL – один из режимов тактового генератор, устанавливаемый битами конфигурации FOSC2:FOSC0 во время программирования микроконтроллера.

Таймер задержки PLL используется для стабилизации работы схемы PLL перед началом работы микроконтроллера. Задержка запуска схемы PLL обозначается как T_{PLL}.

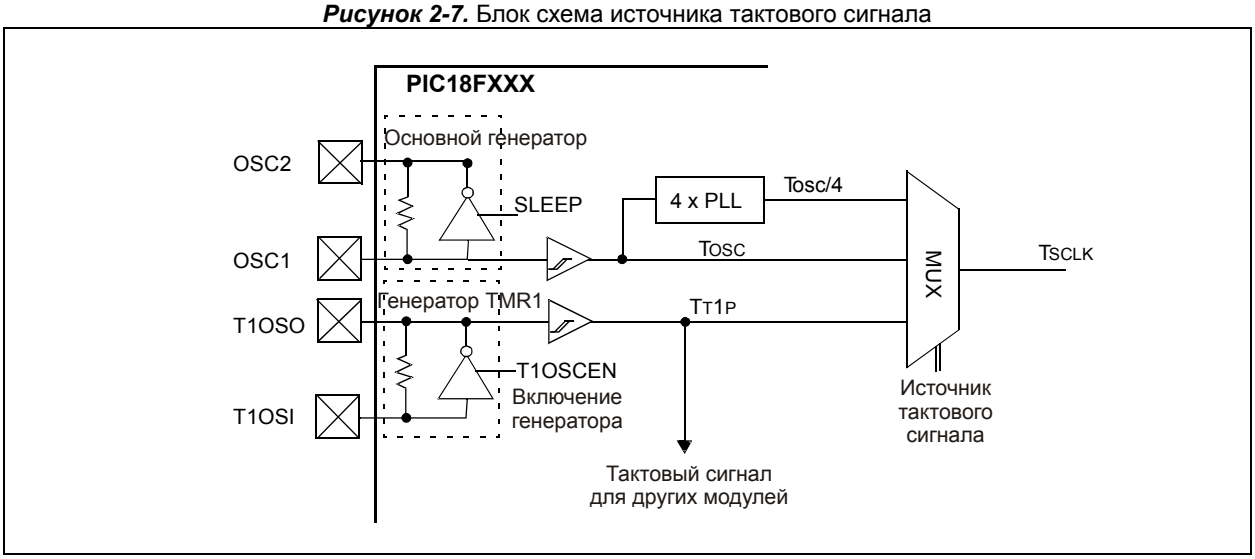
Рисунок 2-6. Структурная схема PLL



2.6 Переключение тактового генератора

Микроконтроллеры PIC18FXX2 содержат особенность, которая позволяет переключать источник тактового сигнала от основного к дополнительному источнику с более низкой частотой. Для PIC18FXX2 дополнительным источником тактового сигнала является генератор таймера TMR1. Если низкочастотный резонатор (например, 32кГЦ) был подключен к выводам генератора TMR1, то микроконтроллер может перейти в режим работы с малым энергопотреблением. На рисунке 2-7 показана структурная схема источника тактового сигнала. Разрешение переключения тактового генератора устанавливается в битах конфигурации (-OSCSEN=0, регистр 1H) при программировании микроконтроллера. Переключение тактового сигнала заблокировано в «чистом» микроконтроллере. Описание работы генератора таймера TMR1 смотрите в разделе 11.0. Описание регистров конфигурации микроконтроллера смотрите в разделе 19.0.

Рисунок 2-7. Блок схема источника тактового сигнала



2.6.1 Бит переключения тактового генератора

Переключение источника тактового сигнала выполняется командами микроконтроллера, изменяющими состояние бита SCS (OSCCON<0>). Когда SCS=0, тактовый сигнал берется с основного генератора, который настраивается битами конфигурации FOSC в регистре 1H. Когда SCS=1 источником тактового сигнала является генератор таймера TMR1. При любом виде сброса бит SCS сбрасывается в ‘0’.

Примечание. Для использования дополнительного источника тактового сигнала генератор TMR1 должен быть включен, установкой бита T1OSCEN регистра T1CON в ‘1’. Если генератор TMR1 выключен, то любая запись в регистр SCS будет игнорироваться (принудительно сбрасываться), микроконтроллер продолжит работать от основного генератора тактового сигнала.

Регистр 2-1. Регистр OSCCON

U - 0	U - 0	U - 0	U - 0	U - 0	U - 0	U - 0	R/W - 0
-	-	-	-	-	-	-	SCS
Бит 7							Бит 0

Бит 7-1 **Не используется:** Читается как ‘0’

Бит 0 **SCS:** Переключение источника тактового сигнала
Если -OSCSEN=0, T1OSCEN=1
1 = тактовый сигнал от генератора таймера TMR1
0 = тактовый сигнал от основного генератора

Другое состояние битов -OSCSEN, T1OSCEN
Принудительно сбрасывается в ‘0’

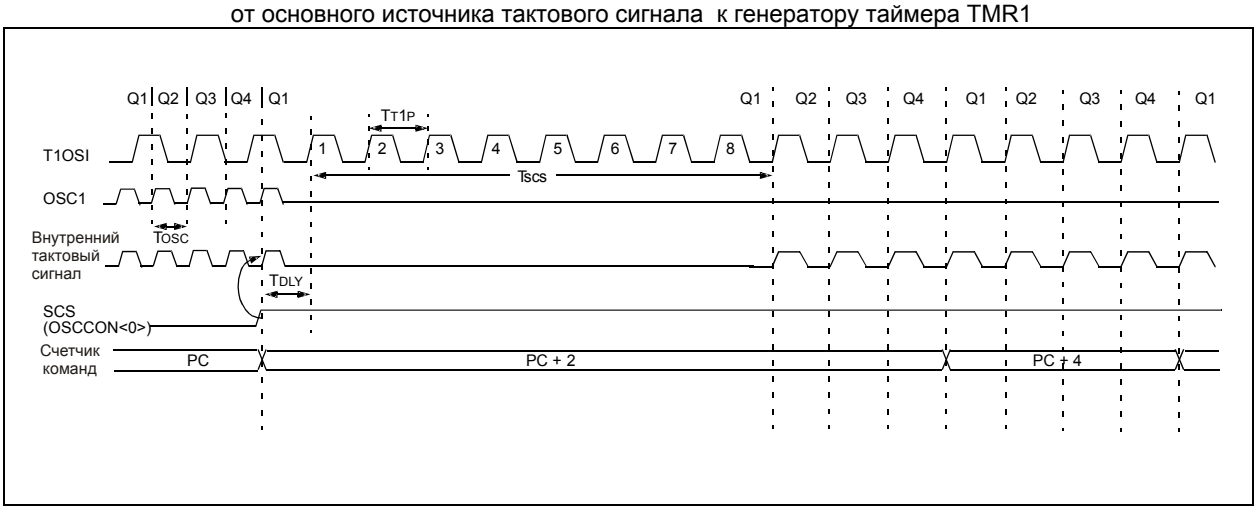
Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

2.6.2 Переключение источника тактового сигнала

Микроконтроллеры PIC18FXX2 содержит схему, которая предотвращает сбои при переключении источника тактового сигнала. Фактически схема синхронизации ожидает восемь передних фронтов тактового сигнала, на который выполняется переключение. Это гарантирует, что частота тактового сигнала стабильна и не будет импульса длительностью меньше, чем минимальная длительность импульса двух источников.

На рисунке 2-8 показана временная диаграмма переключения от основного источника тактового сигнала к генератору таймера TMR1. Генератор таймера TMR1 постоянно включен. После установки бита SCS в ‘1’ выполнение программы приостанавливается на следующем такте Q1, отсчитывается 8 тактов генератора таймера TMR1, затем продолжится выполнение программы. Нет никаких дополнительных задержек после отсчета синхронизирующих импульсов.

Рисунок 2-8. Временная диаграмма переключения от основного источника тактового сигнала к генератору таймера TMR1

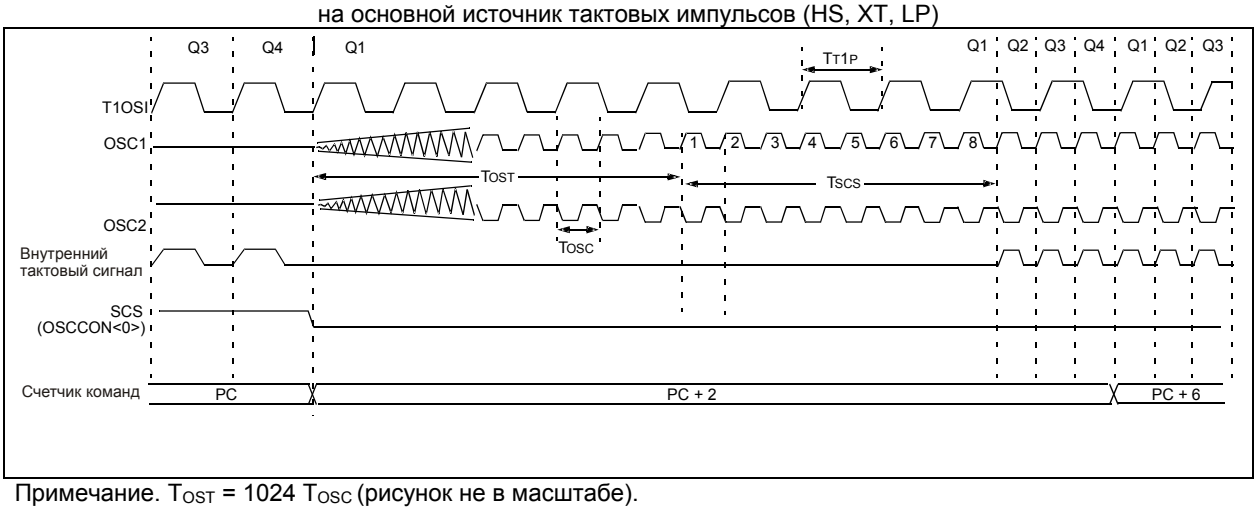


Примечание. Нет задержки после отсчета восьми тактовых импульсов.

Последовательность действий, выполняемых при переключении с генератора TMR1 на основной источник тактовых импульсов, зависит от режима работы основного генератора. Помимо отсчета восьми тактовых импульсов могут быть добавлены дополнительные задержки.

Если основной генератор работает в режиме кварцевого/керамического резонатора (HS, XT, LP), то переход произойдет после запуска генератора (задержка T_{OST}). На рисунке 2-9 показана временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов, работающий в режиме HS, XT или LP тактового генератора.

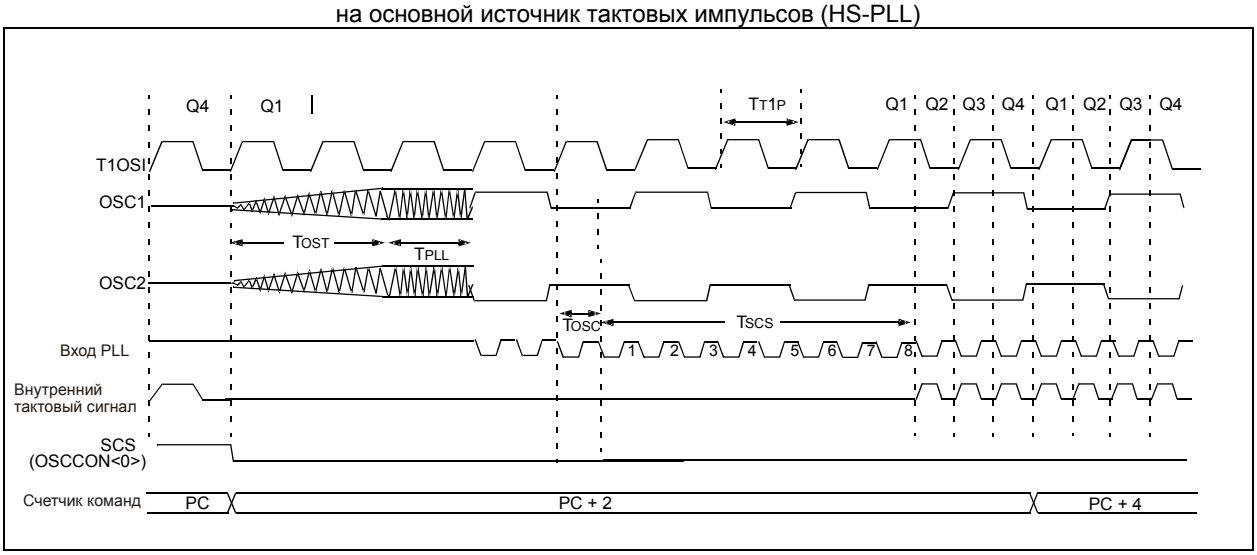
Рисунок 2-9. Временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов (HS, XT, LP)



Примечание. T_{OST} = 1024 T_{OSC} (рисунок не в масштабе).

Если основной генератор работает в режиме HS-PLL, то время запуска генератора T_{OST} плюс задержка запуска схемы PLL T_{PLL}. Типовое время задержки старта схемы PLL 2мс. На рисунке 2-10 показана временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов в режиме HS-PLL.

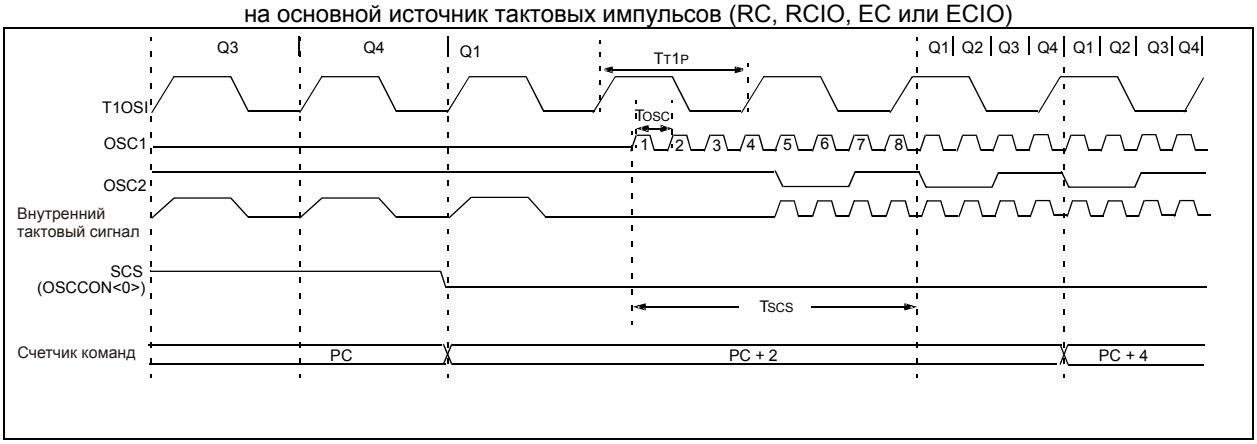
Рисунок 2-10. Временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов (HS-PLL)



Примечание. T_{OST} = 1024 T_{OSC} (рисунок не в масштабе).

Если основной генератор настроен в RC, RCIO, EC или ECIO режиме тактового генератора нет никакой задержки перед запуском генератора. Работа микроконтроллера возобновляется после отсчета восьми тактовых импульсов. На рисунке 2-11 показана временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов, работающий в режиме RC, RCIO, EC или ECIO.

Рисунок 2-11. Временная диаграмма переключения с генератора TMR1 на основной источник тактовых импульсов (RC, RCIO, EC или ECIO)



Примечание. RC режим генератора.

2.7 Влияние режима SLEEP на работу тактового генератора

После выполнения команды SLEEP интегрированный тактовый генератор выключен, ядро микроконтроллера переведено в начало цикла команды (такт Q1). Генератор прекратит формирование тактовых импульсов. Т.к. все ключи схему микроконтроллера закрыты, он потребляет минимальный ток (только токи утечки). Любой периферийный модуль, работающий в SLEEP режиме, увеличит ток потребления. Выход из режима SLEEP происходит при внешнем сбросе микроконтроллера, сбросе от сторожевого таймера и генерации прерывания.

Таблица 2-3. Состояние выводов OSC1, OSC2 в SLEEP режиме микроконтроллера

Режим тактового генератора	Вывод OSC1	Вывод OSC2
RC	Свободный, внешний резистор должен притягивать к высокому логическому уровню	Низкий логический уровень
RCIO	Свободный, внешний резистор должен притягивать к высокому логическому уровню	Управляется битом 6 регистра PORTA
ECIO	Свободный	Управляется битом 6 регистра PORTA
EC	Свободный	Низкий логический уровень
LP, XT или HS	Обратная связь инвертора выключена, неизменяемый уровень напряжения	Обратная связь инвертора выключена, неизменяемый уровень напряжения

Примечание. Длительность задержек старта тактового генератора после сброса –MCLR и выхода из режима SLEEP смотрите в таблице 3-1 раздела «Сброс»

2.8 Задержка старта после включения питания

Два таймера управляют задержкой старта выполнения программы микроконтроллера после включения питания, что не требует применения внешних дополнительных схем сброса. Задержки гарантируют, что микроконтроллер будет находиться в состоянии сброса пока напряжение питания и частота тактового генератора не стабилизировались. Дополнительную информацию по сбросу микроконтроллера смотрите в разделе 3 «Сброс».

Первый таймер – таймер включения питания (PWRT), который обеспечивает задержку 72мс (типовое значение) при сбросе по включению питания (POR) или снижению напряжения питания (BOR). Второй таймер – таймер запуска тактового генератора (OST), удерживающий микроконтроллер в состоянии сброса пока не стабилизируется частота тактового генератора.

С включенным режимом PLL (HS/PLL) последовательность задержек старта программы после сброса POR несколько иная: счет таймера PWRT после сброса POR; счет таймера запуска генератора. Этих задержек недостаточно для нормального запуска схемы PLL, поэтому на базе таймера PWRT выполняется дополнительная задержка длительностью 2мс (типовое значение).

3. Сброс

В PIC18FXX2 различаются следующие виды сброса:

a) Сброс по включению питания (POR)
b) Сброс по сигналу –MCLR в нормальном режиме
c) Сброс по сигналу –MCLR в режиме SLEEP
d) Сброс от сторожевого таймера WDT в нормальном режиме
e) Сброс по снижению напряжения питания (BOR)
f) Выполнение команды RESET
g) Сброс по переполнению стека
h) Сброс по исчерпанию стека

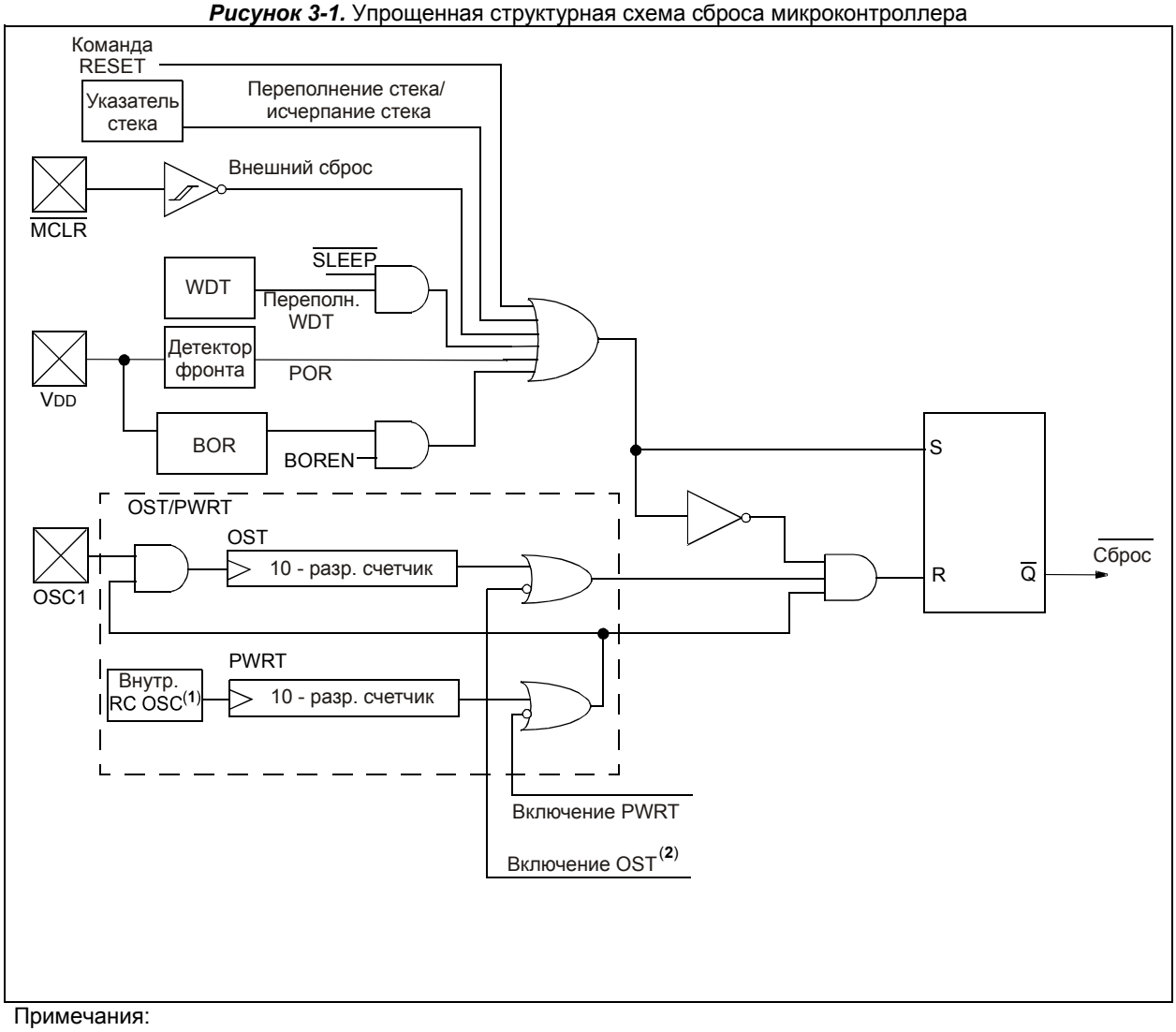
Большинство регистров не изменяют своего содержимого после любого вида сброса, а при сбросе POR содержат неизвестное значение. Другие регистры сбрасываются в начальное состояние при сбросе POR, -MCLR, BOR, переполнение WDT в нормальном режиме и выполнении команды RESET.

Сброс по переполнению WDT в SLEEP режиме микроконтроллера рассматривается как возобновление нормальной работы и на большинство регистров не влияет. В регистре RCON содержаться биты (-RI, -TO, -PD, -POR, -BOR), с помощью которых можно определить причину сброса микроконтроллера (смотрите таблицу 3-2). Эти биты могут использоваться в программе пользователя для определения причины сброса микроконтроллера. В таблице 3-3 представлено состояние всех регистров после различных видов сброса.

Упрощенная структурная схема сброса микроконтроллера показана на рисунке 3-1. На входе –MCLR есть внутренний фильтр, не пропускающие короткие импульсы.

Необходимо отметить, что сброс WDT не управляет выводом –MCLR.

Рисунок 3-1. Упрощенная структурная схема сброса микроконтроллера



Примечания:

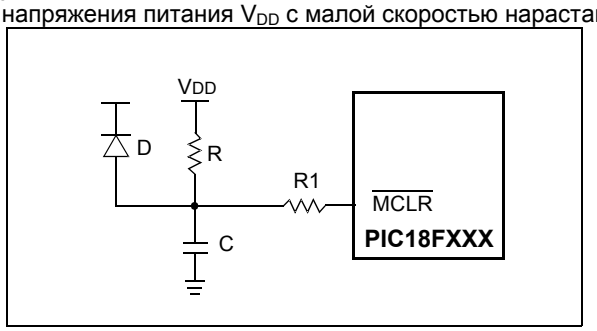
1. Это отдельный RC генератор.
2. Длительность задержек смотрите в таблице 3-1.

3.1 Сброс по включению питания POR

Интегрированная схема POR удерживает микроконтроллер в состоянии сброса, пока напряжение V_{DD} не достигнет требуемого уровня. Для включения схемы POR необходимо соединить вывод –MCLR с V_{DD}, не требуя внешней RC цепочки, обычно используемой для сброса. Минимальную скорость нарастания напряжения питания смотрите в разделе «Электрические характеристики» параметр D004. Для случая с малой скоростью напряжения питания воспользуйтесь схемой, показанной на рисунке 3-2.

Когда микроконтроллер переходит в режим нормальной работы из состояния сброса, рабочие параметры (напряжение питания, тактовая частота, температура и т.д.) должны соответствовать указанным в разделе «Электрические характеристики». Если рабочие параметры не удовлетворяют требованиям, то микроконтроллер должен находиться в состоянии сброса.

Рисунок 3-2. Внешняя схема сброса по включению питания
(для напряжения питания V_{DD} с малой скоростью нарастания)



Примечания:

1. Внешняя схема сброса по включению питания требуется только, если скорость нарастания напряжения питания очень мало. Диод D позволяет быстро разрядить конденсатор C при выключении напряжения питания VDD.
2. Рекомендованное значение R < 40кОм. Это необходимо, чтобы выполнить требования электрических характеристик (падение напряжение на резисторе).
3. Резистор R1 с сопротивлением от 100Ом до 1кОм позволяет предотвратить большой ток вывода –MCLR от конденсатора C в случае электростатического (или от перенапряжения) повреждения.

3.2 Таймер включения питания PWRT

Таймер по включению питания обеспечивает фиксированную задержку (смотрите параметр 33) по сигналу схемы POR. Таймер включения питания работает от отдельного внутреннего RC генератора и удерживает микроконтроллер в состоянии сброса по активному сигналу от PWRT. Задержка PWRT позволяет достигнуть напряжению питания V_{DD} номинального значения. Включение схемы PWRT осуществляется соответствующей настройкой бита конфигурации PWRT.

Время задержки PWRT варьируется в каждом микроконтроллере, зависит от напряжения питания и температуры (смотрите раздел «Электрические характеристики» параметр 33).

3.3 Таймер запуска генератора OST

Таймер запуска генератора обеспечивает задержку в 1024 такта генератора (OSC1) после окончания задержки от таймера PWRT (если он включен). Это гарантирует, что частота кварцевого/керамического резонатора стабилизировалась. Задержка OST только в режимах HS, XT и LP тактового генератора после сброса POR или выхода микроконтроллера из режима SLEEP.

3.4 Таймер запуска PLL

С включенным режимом PLL (HS/PLL) последовательность задержек старта программы после сброса POR несколько иная: счет таймера PWRT после сброса POR; счет таймера запуска генератора. Этих задержек недостаточно для нормального запуска схемы PLL, поэтому на базе таймера PWRT выполняется дополнительная задержка длительностью T_{PLL} = 2мс (типовое значение), выполняемая после окончания задержки OST.

3.5 Сброс по снижению напряжения питания BOR

Битом BODEN в слове конфигурации можно включить (BODEN=0) или выключить (BODEN=1) детектор снижения напряжения питания. Если напряжение питания V_{DD} снижается ниже параметра D005 на время больше или равное T_{BOR} (смотрите параметр 35), то произойдет сброс по снижению напряжения питания. Микроконтроллер останется в состоянии сброса пока напряжение питания V_{DD} не станет выше BV_{DD}. После нормализации напряжения питания микроконтроллер находится в состоянии сброса в течение задержки PWRT (параметр 33), если она включена. Если напряжение V_{DD} стало ниже BV_{DD} во время счета таймера PWRT, то микроконтроллер возвратится в состояние сброса по снижению напряжения питания. Каждый переход напряжения питания V_{DD} через уровень BV_{DD} инициализирует PWRT, создавая дополнительную задержку.

3.6 Последовательность удержания микроконтроллера в состоянии сброса

При включении питания выполняется следующая последовательность удержания микроконтроллера в состоянии сброса: сброс POR, задержка PWRT (если она разрешена), задержка OST (после завершения задержки PWRT). Полное время задержки изменяется в зависимости от режима тактового генератора и состояния бита –PWRT. На рисунках 3-3, 3-4, 3-5, 3-6 и 3-7 представлены последовательности удержания микроконтроллера в состоянии сброса после включения питания.

Если сигнал –MCLR удерживается в низком логическом уровне достаточно долго (дольше времени всех задержек), то после перехода –MCLR в высокий уровень программ начнет выполняться немедленно (см. рисунок 3-5). Это может быть полезно для одновременного запуска нескольких микроконтроллеров, работающих параллельно.

В таблице 3-2 показано состояние некоторых регистров специального назначения после различных видов сброса, а в таблице 3-3 состояние всех регистров специального назначение.

Таблица 3-1. Длительность задержек в различных ситуациях

Режим генератора	**Сброс POR**		**Сброс BOR**	**Выход из SLEEP или переключение генератора**
	-PWRTR=0	**-PWRTE=1**		
HS/PLL(1)	72мс + 1024T_{OSC} + 2мс	1024T_{OSC} + 2мс	72мс(2) + 1024T_{OSC} +2мс	1024T_{OSC} + 2мс
HS, XT, LP	72мс + 1024T_{OSC}	1024T_{OSC}	72мс(2) + 1024T_{OSC}	1024T_{OSC}
EC	72мс	-	72мс(2)	-
Внешний RC	72мс	-	72мс(2)	-

Примечания:

1. Дополнительная задержка в 2мс необходима для запуска схемы PLL.
2. 72мс – типовое время задержки PWRT, если она включена.

Регистр 3-1. Регистр RCON

R/W - 0	U - 0	U - 0	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
IPEN	-	-	-RI	-TO	-PD	-POR	-BOR
Бит 7							Бит 0

Примечание. Описание битов регистра смотрите в разделе 4.14.

Таблица 3-2. Состояние некоторых битов и регистров специального назначения после сброса

Вид сброса	**Счетчик команд PC**	**Регистр RCON**	**-RI**	**-TO**	**-PD**	**-POR**	**-BOR**	**STKFUL**	**STKUNF**
Сброс POR	0000h	0--1 1100	1	1	1	0	0	u	u
Сброс по сигналу –MCLR в нормальном режиме	000h	0--u uuuu	u	u	u	u	u	u	u
Программный сброс	0000h	0--u uuuu	0	u	u	u	u	u	u
Сброс по переполнению стека	0000h	0--u uu11	u	u	u	u	u	u	1
Сброс по исчерпанию стека	0000h	0--u uu11	u	u	u	u	u	1	u
Сброс по сигналу –MCLR в SLEEP режиме	0000h	0--u 10uu	u	1	0	u	u	u	u
Сброс по переполнению WDT	0000h	0--u 01uu	1	0	1	u	u	u	u
Выход из режима SLEEP по переполнению WDT	PC + 2	u--u 00uu	u	0	0	u	u	u	u
Сброс BOR	0000h	0--1 11u0	1	1	1	1	0	u	u
Выход из режима SLEEP по прерыванию	PC + 2(1)	u--u 00uu	u	1	0	u	u	u	u

Обозначения: u = не изменяется; x = неизвестно; - = не используется, читается как “0’

Примечание. Когда происходит выход из режима SLEEP по прерыванию и бит GIEH или GIEL установлен в ‘1’, в счетчик команд загружается вектор прерывания (0x000008h или 0x000018h).

Таблица 3-3. Состояние регистров специального назначения после сброса

Регистр					Сброс POR, BOR	Сброс –MCLR, WDT, команда RESET, сброс от стека	Выход из режима SLEEP по переполнению WDT или прерываниям
TOSU	242	442	252	452	---0 0000	---0 0000	---0 uuuu(3)
TOSH	242	442	252	452	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	242	442	252	452	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	242	442	252	452	00-0 0000	00-0 0000	uu-u uuuu(3)
PCLATU	242	442	252	452	---0 0000	---0 0000	---u uuuu
PCLATH	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
PCL	242	442	252	452	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	242	442	252	452	--00 0000	--00 0000	--uu uuuu
TBLPTRH	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
TABLAT	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
PRODH	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	242	442	252	452	0000 000x	0000 000u	0000 000u(1)
INTCON2	242	442	252	452	1111 -1-1	1111 -1-1	uuuu -u-u(1)
INTCON3	242	442	252	452	11-0 0-00	11-0 0-00	uu-u u-uu(1)
INDF0	242	442	252	452	N/A	N/A	N/A
POSTINC0	242	442	252	452	N/A	N/A	N/A
POSTDEC0	242	442	252	452	N/A	N/A	N/A
PREINC0	242	442	252	452	N/A	N/A	N/A
PLUSW0	242	442	252	452	N/A	N/A	N/A
FSR0H	242	442	252	452	---- xxxx	---- uuuu	---- uuuu
FSR0L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	242	442	252	452	N/A	N/A	N/A
POSTINC1	242	442	252	452	N/A	N/A	N/A
POSTDEC1	242	442	252	452	N/A	N/A	N/A
PREINC1	242	442	252	452	N/A	N/A	N/A
PLUSW1	242	442	252	452	N/A	N/A	N/A
FSR1H	242	442	252	452	---- xxxx	---- uuuu	---- uuuu
FSR1L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	242	442	252	452	---- 0000	---- 0000	---- uuuu
INDF2	242	442	252	452	N/A	N/A	N/A
POSTINC2	242	442	252	452	N/A	N/A	N/A
POSTDEC2	242	442	252	452	N/A	N/A	N/A
PREINC2	242	442	252	452	N/A	N/A	N/A
PLUSW2	242	442	252	452	N/A	N/A	N/A
FSR2H	242	442	252	452	---- xxxx	---- uuuu	---- uuuu
FSR2L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu

Обозначения: u = не изменяется; x = неизвестно; - = бит не используется, читается как ‘0’; q = зависит от условий; N/A = регистр физически не реализован; Затененная ячейка = не использовать для данного микроконтроллера.

Примечания:

1. Один или несколько битов в регистрах INTCONx, PIRx будут инициировать выход микроконтроллера из режима SLEEP.
2. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в счетчик команд PC загружается вектор прерываний (0008h или 0018h).
3. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в вершину стека записывается текущее значение PC (TOSU, TOSH и TOSL). Изменяется значение указателя аппаратного стека STKPTR.
4. В таблице 3-2 смотрите значения при различных видах сброса.
5. Бит 6 регистров PORTA, LATA и TRISA используется только EICO и RCIO режимах тактового генератора. В других режимах тактового генератора этот бит не используется и читается как ‘0’.
6. Бит 7 регистров PORTA, LATA и TRISA не реализован во всех микроконтроллерах. Не реализованный бит читается как ‘0’.

Таблица 3-3. Состояние регистров специального назначения после сброса (продолжение)

Регистр					Сброс POR, BOR	Сброс –MCLR, WDT, команда RESET, сброс от стека	Выход из режима SLEEP по переполнению WDT или прерываниям
STATUS	242	442	252	452	---x xxxx	---u uuuu	---u uuuu
TMR0H	242	442	252	452	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
OSCCON	242	442	252	452	---- ---0	---- ---0	---- ---u
LVDCON	242	442	252	452	--00 0101	--00 0101	--uu uuuu
WDTCON	242	442	252	452	---- ---0	---- ---0	---- ---u
RCON(4)	242	442	252	452	0—q 11qq	0—q qquu	u—u qquu
TMR1H	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	242	442	252	452	0-00 0000	u-uu uuuu	u-uu uuuu
TMR2	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
PR2	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
T2CON	242	442	252	452	-000 0000	-000 0000	-uuu uuuu
SSPBUF	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
SSPCON1	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
SSPCON2	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
ADRESH	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	242	442	252	452	0000 00-0	0000 00-0	uuuu uu-u
ADCON1	242	442	252	452	00—0000	00—0000	uu—uuuu
CCPR1H	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	242	442	252	452	--00 0000	--00 0000	--uu uuuu
CCPR2H	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	242	442	252	452	--00 0000	--00 0000	--uu uuuu
TMR3H	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	242	442	252	452	0000 0000	uuuu uuuu	uuuu uuuu
SPBRG	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
RCREG	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
TXREG	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
TXSTA	242	442	252	452	0000 -010	0000 -010	uuuu -uuu
RCSTA	242	442	252	452	0000 000x	0000 000x	uuuu uuuu
EEADR	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
EEDATA	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
EECON1	242	442	252	452	xx-0 x000	uu-0 u000	uu-0 u000
EECON2	242	442	252	452	N/A	N/A	N/A

Обозначения: u = не изменяется; x = неизвестно; - = бит не используется, читается как ‘0’; q = зависит от условий; N/A = регистр физически не реализован; Затененная ячейка = не использовать для данного микроконтроллера.

Примечания:

1. Один или несколько битов в регистрах INTCONx, PIRx будут инициировать выход микроконтроллера из режима SLEEP.
2. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в счетчик команд PC загружается вектор прерываний (0008h или 0018h).
3. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в вершину стека записывается текущее значение PC (TOSU, TOSH и TOSL). Изменяется значение указателя аппаратного стека STKPTR.
4. В таблице 3-2 смотрите значения при различных видах сброса.
5. Бит 6 регистров PORTA, LATA и TRISA используется только EICO и RCIO режимах тактового генератора. В других режимах тактового генератора этот бит не используется и читается как ‘0’.
6. Бит 7 регистров PORTA, LATA и TRISA не реализован во всех микроконтроллерах. Не реализованный бит читается как ‘0’.

Таблица 3-3. Состояние регистров специального назначения после сброса (продолжение)

Регистр					Сброс POR, BOR	Сброс –MCLR, WDT, команда RESET, сброс от стека	Выход из режима SLEEP по переполнению WDT или прерываниям
IPR2	242	442	252	452	---1 1111	---1 1111	---u uuuu
PIR2	242	442	252	452	---0 0000	---0 0000	---u uuuu(1)
PIE2	242	442	252	452	---0 0000	---0 0000	---u uuuu
IPR1	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
	242	442	252	452	-111 1111	-111 1111	-uuu uuuu
PIR1	242	442	252	452	0000 0000	0000 0000	uuuu uuuu(1)
	242	442	252	452	-000 0000	-000 0000	-uuu uuuu(1)
PIE1	242	442	252	452	0000 0000	0000 0000	uuuu uuuu
	242	442	252	452	-000 0000	-000 0000	-uuu uuuu
TRISE	242	442	252	452	0000 -111	0000 -111	uuuu -uuu
TRISD	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISC	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISB	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISA(5,6)	242	442	252	452	-111 1111(5)	-111 1111(5)	-uuu uuuu(5)
LATE	242	442	252	452	---- -xxx	---- -uuu	---- -uuu
LATD	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5,6)	242	442	252	452	-xxx xxxx(5)	-uuu uuuu(5)	-uuu uuuu(5)
PORTE	242	442	252	452	---- -000	---- -000	---- -uuu
PORTD	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5,6)	242	442	252	452	-x0x 0000(5)	-u0u 0000(5)	-uuu uuuu(5)

Обозначения: u = не изменяется; x = неизвестно; - = бит не используется, читается как ‘0’; q = зависит от условий; N/A = регистр физически не реализован; Затененная ячейка = не использовать для данного микроконтроллера.

Примечания:

1. Один или несколько битов в регистрах INTCONx, PIRx будут инициировать выход микроконтроллера из режима SLEEP.
2. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в счетчик команд PC загружается вектор прерываний (0008h или 0018h).
3. Когда происходит выход из режима SLEEP по прерыванию и бит GIEL или GIEH установлен в ‘1’, в вершину стека записывается текущее значение PC (TOSU, TOSH и TOSL). Изменяется значение указателя аппаратного стека STKPTR.
4. В таблице 3-2 смотрите значения при различных видах сброса.
5. Бит 6 регистров PORTA, LATA и TRISA используется только EICO и RCIO режимах тактового генератора. В других режимах тактового генератора этот бит не используется и читается как ‘0’.
6. Бит 7 регистров PORTA, LATA и TRISA не реализован во всех микроконтроллерах. Не реализованный бит читается как ‘0’.

Рисунок 3-3. Временная диаграмма сброса по включению питания (-MCLR подключен к V_{DD})

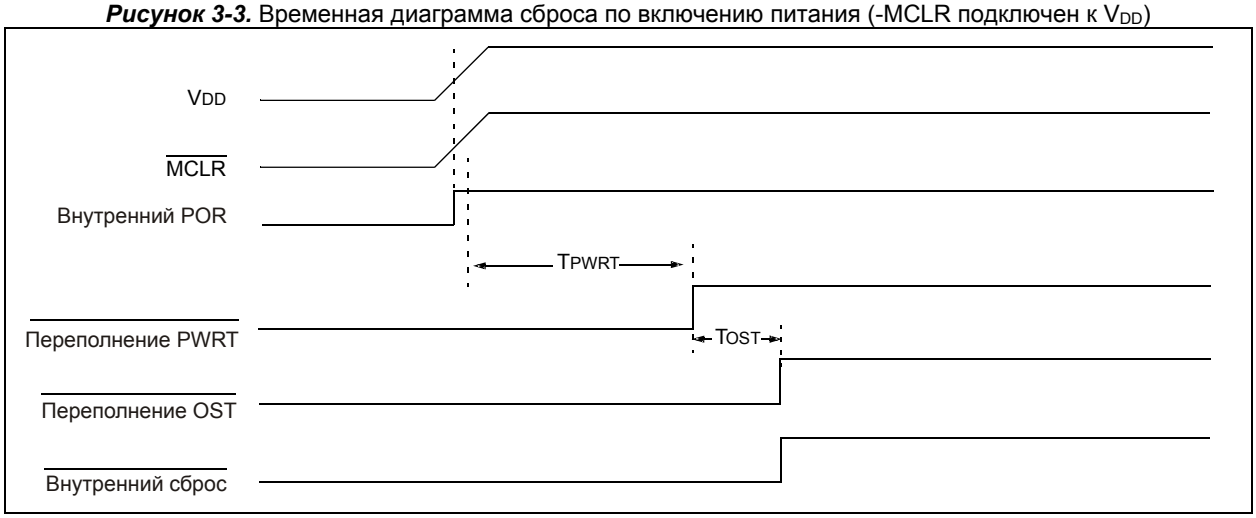


Рисунок 3-4. Временная диаграмма сброса по включению питания (-MCLR не подключен к V_{DD}), случай 1

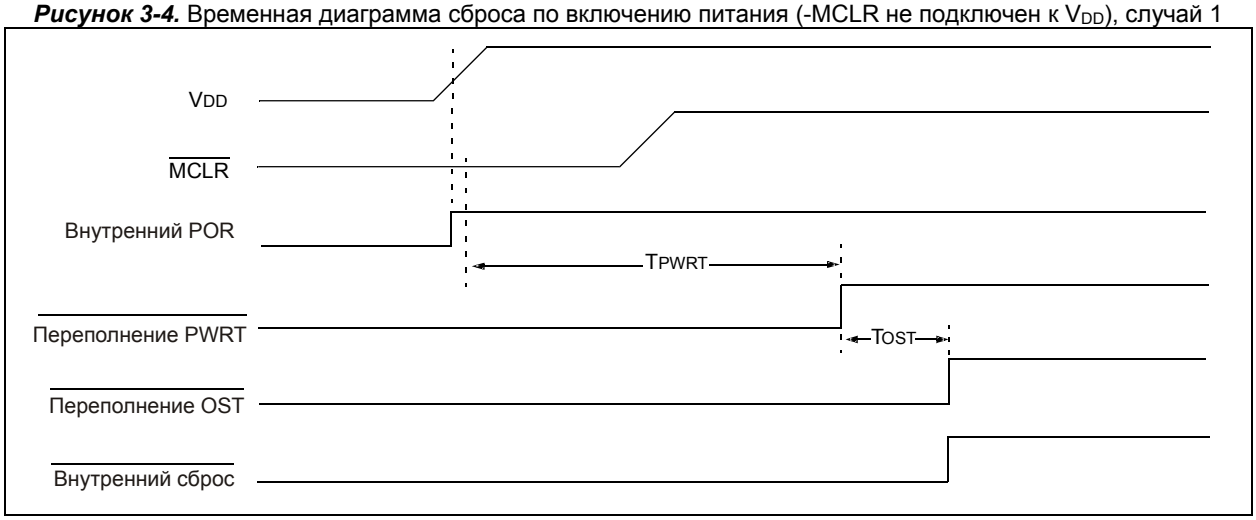


Рисунок 3-5. Временная диаграмма сброса по включению питания (-MCLR не подключен к V_{DD}), случай 2

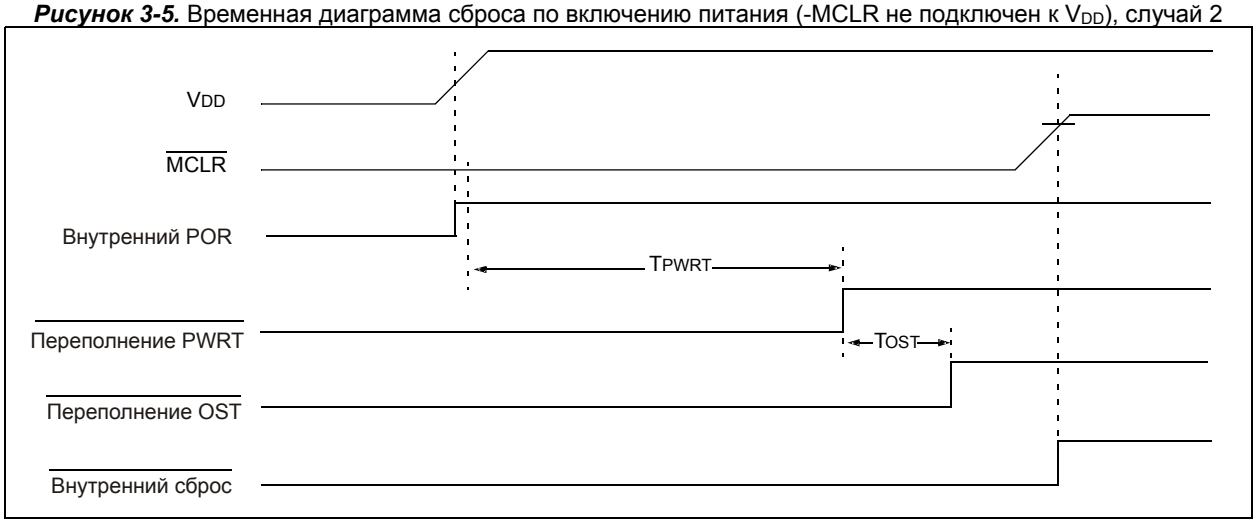


Рисунок 3-6. Временная диаграмма сброса
при медленном нарастании напряжения питания (-MCLR подключен к V_{DD})

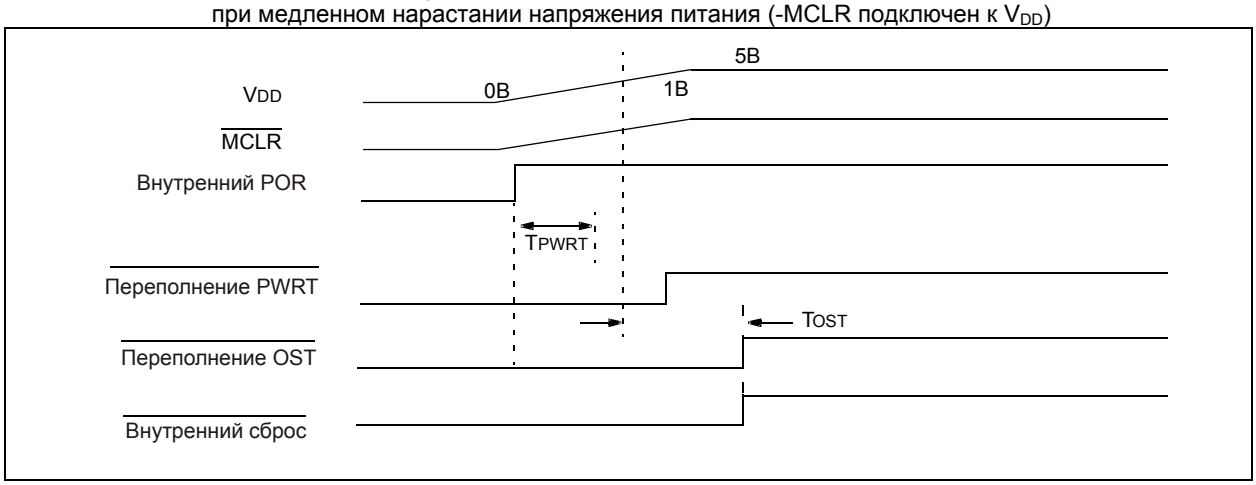
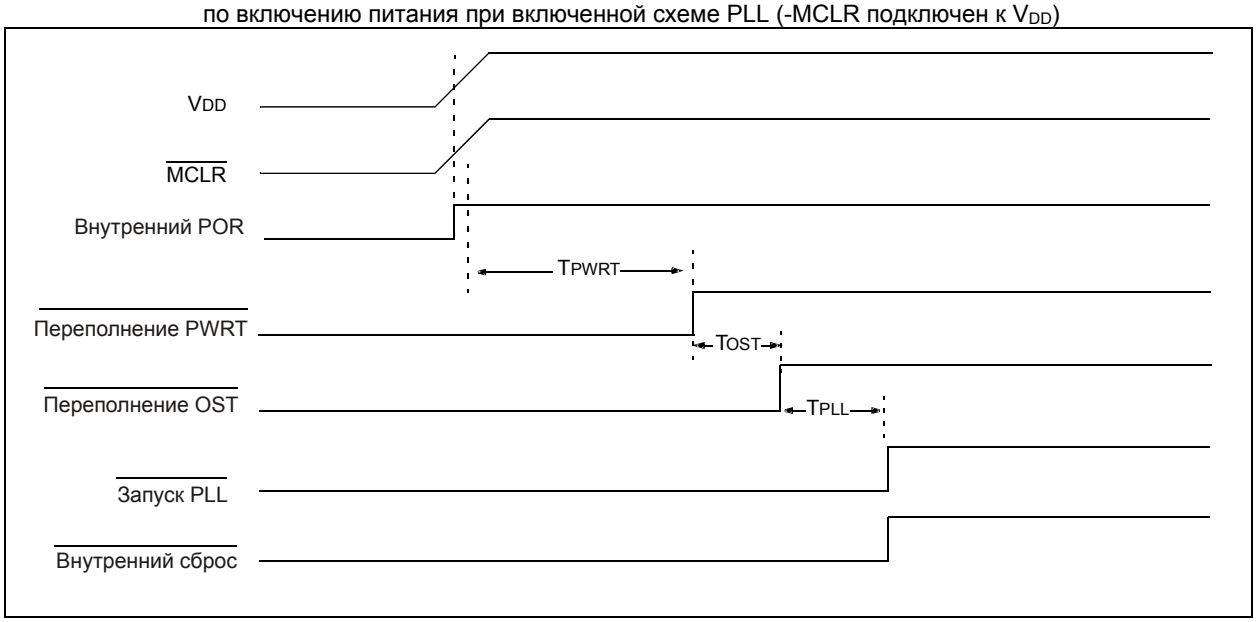


Рисунок 3-7. Временная диаграмма сброса
по включению питания при включенной схеме PLL (-MCLR подключен к V_{DD})



Примечание. $T_{OST} = 1024T_{OSC}$. $T_{PLL} \approx 2$мс.

4. Организация памяти

В микроконтроллерах PIC18FXX2 реализовано три типа памяти:

- Память программ
- Память данных
- EEPROM память данных

Обращение к памяти программ и памяти данных выполняется по отдельным шинам, что позволяет организовать параллельный доступ к этим видам памяти.

Дополнительную информацию по Flash памяти программ и EEPROM памяти данных смотрите соответственно в разделах 5 и 6.

4.1 Организация памяти программ

21-разрядный счетчик команд PC позволяет адресовать 2Мбайта памяти программ. Физически не реализованная память программ читается как ‘0’ (команда NOP).

Микроконтроллеры PIC18F252, PIC18F452 содержат по 32кбайта Flash памяти программ, а PIC18F242 и PIC18F442 имеют по 16кбайт Flash памяти программ. Это означает, что микроконтроллеры PIC18FX52 могут иметь до 16к отдельных команд, а PIC18FX42 – до 8к отдельных команд.

Адрес вектора сброса – 0000h. Адреса векторов прерываний – 0008h и 00018h.

На рисунках 4-1 и 4-2 показана карта памяти микроконтроллеров PIC18F242/442 и PIC18F252/452.

Рисунок 4-1. Карта памяти программ и стека микроконтроллеров PIC18F242/442

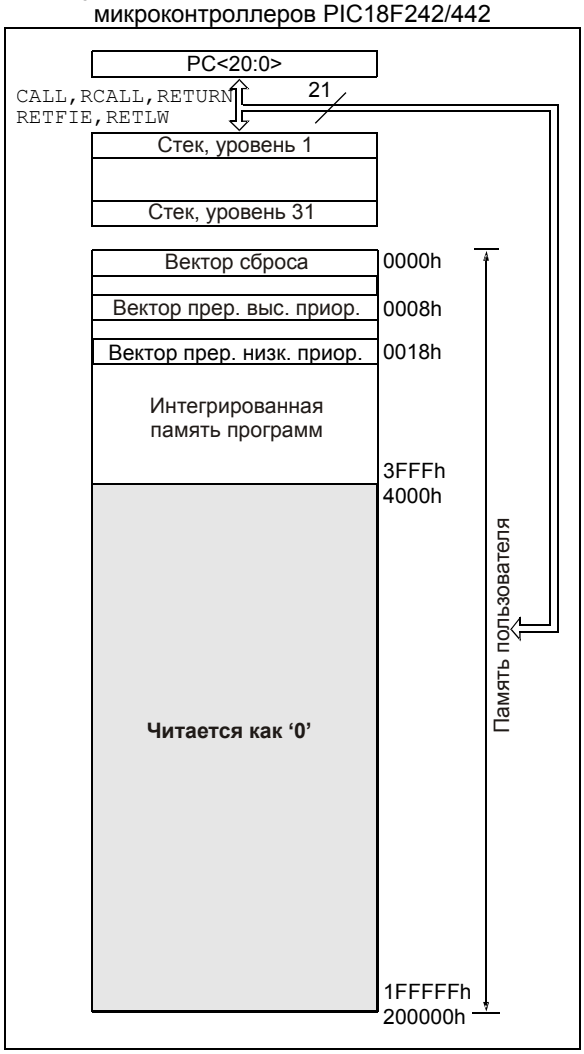
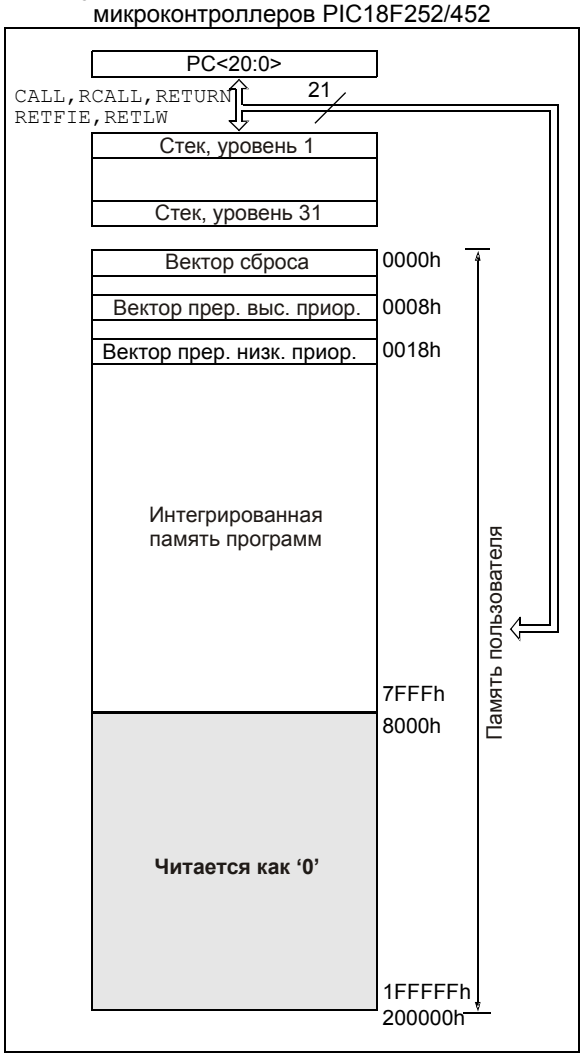


Рисунок 4-2. Карта памяти программ и стека микроконтроллеров PIC18F252/452



4.2 Стек

Стек позволяет сохранить до 31 адреса возврата из подпрограммы или обработки прерываний. Значение счетчика команд PC помещается в стек при выполнении команд CALL, RCALL или переходе на подпрограмму обработки прерываний. По команде RETURN, RETLW или RETFIE значение из стека загружается в счетчик команд PC. При выполнении любой команды перехода или возврата из подпрограммы (прерываний) значение регистров PCLATU, PCLATH не изменяется.

Стек выполнен в виде 21-разрядного ОЗУ объемом 31 слово. 5-разрядный указатель стека принимает значение 00000b после любого вида сброса микроконтроллера. Нет никакой связи с памятью данных и значением указателя стека 00000b. При выполнении команды типа CALL сначала увеличивается указатель стека, а затем значение счетчика команд PC помещается в вершину стека. При выполнении команды типа RETURN значение с вершины стека загружается в счетчик команд PC, затем указатель стека декрементируется.

Стек не является частью памяти программ или памяти данных. Указатель стека доступен для записи и чтения, он фактически является адресатом вершины стека, которая может быть прочитана и изменена через регистры специального назначения. Данные могут быть загружены/прочитаны в стек выполняя операции с вершиной стека. Биты статуса отображают состояние указателя стека (переполнение, исчерпание стека).

4.2.1 Доступ к вершине стека

Вершина доступна для записи и чтения. Три регистра специального назначения TOSU, TOSH и TOSL отображают состояние вершины стека, указанной в регистре STKPTR. Это позволяет в случае необходимости выполнять операции со стеком командами микроконтроллера. После выполнения команд CALL, RCALL или перехода на обработку прерываний, пользователь может прочитать вершину стека через регистры TOSU, TOSH и TOSL. Эти значения могут быть помещены в определенный пользователем программный стек. При возвращении из процедуры можно программным способом изменить значение регистров TOSU, TOSH, TOSL и выполнить выход из подпрограммы.

При изменении значений стека рекомендуется выключать прерывания, чтобы предотвратить возможное некорректное изменение содержимого стека.

4.2.2 Указатель стека (регистр STKPTR)

Регистр STKPTR содержит: биты указателя стека; бит STKFUL - флаг переполнения стека; бит STKUNF - флаг исчерпания стека. Указатель стека может принимать значения от 0 до 31. Указатель стека увеличивается, когда помещается новое значение в стек, а при чтении вершины стека декрементируется. При любом сбросе микроконтроллера указатель стека становится равным 0. Указатель стека доступен для записи и чтения. Эта особенность может использоваться системами RTOS для сохранения адресов возврата из процедур.

После записи в стек более 31 раза (без чтения содержимого стека) устанавливается бит STKFUL, который может быть сброшен в ‘0’ только программным способом или сбросом по включению питания POR.

Действие, выполняемое при переполнении стека, зависит от состояния бита конфигурации STVREN (разрешение сброса микроконтроллера при переполнении стека). Подробное описание битов конфигурации смотрите в разделе 20. Если бит STVREN установлен в ‘1’ (значение по умолчанию), то при переходе на процедуру (обработку прерываний) в 31-ю ячейку стека помещается адрес возврата (PC+2), устанавливается в ‘1’ бит STKFUL, выполняется сброс микроконтроллера (при этом бит STKFUL остается равным ‘1’, а указатель стека равен 0).

Если STVREN=0, STKFUL будет установлен в ‘1’ при записи в 31-ю ячейку стека, указатель стека будет иметь значение 31. Любая дополнительная запись в стек не будет изменять значение стека, а указатель стека по-прежнему будет иметь значение 31.

При исчерпании стека (выполнялся возврат больше число раз, чем переходов на подпрограммы/обработку прерываний) в счетчик команд PC загружается 0000h, устанавливается в ‘1’ бит STKUNF, указатель стека остается равным 0. Бит STKUNF сбрасывается в ‘0’ программным способом и при сбросе по включению питания POR.

Примечание. При исчерпании стека происходит переход по вектору сброса 0000h, где может быть проверено состояние стека и выполнены необходимые действия.

Регистр 4-1. Регистр STKPTR

R/C - 0	R/C - 0	U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
STKFUL	STKUNF	-	SP4	SP3	SP2	SP1	SP0
Бит 7							Бит 0

Бит 7 **STKFUL:** Флаг переполнения стека
1 = стек полон или произошло переполнения стека
0 = стек не полон, нет переполнения стека

Бит 6 **STKUNF:** Флаг исчерпания стека
1 = произошло исчерпание стека
0 = исчерпание стека не происходило

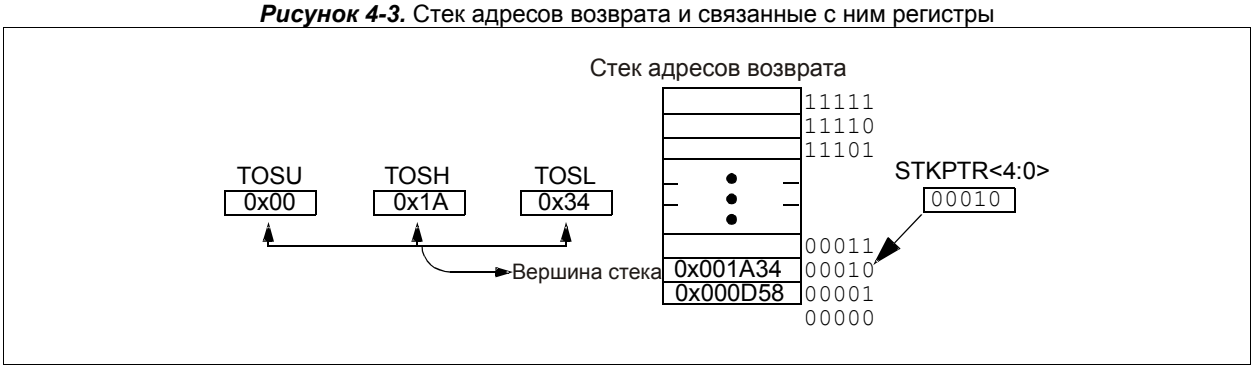
Бит 5 **Не используется:** Читается как ‘0’

Бит 4-0 **SP4:SP0:** Биты указателя стека

Примечание. Биты 7 и 6 программно могут быть только сброшены в ‘0’ (биты равны нулю после сброса POR).

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Рисунок 4-3. Стек адресов возврата и связанные с ним регистры



4.2.3 Команды PUSH и POP

Возможность записи и чтения вершины стека (TOS) предоставляет дополнительную гибкость в написании программ без нарушения нормальной работы микроконтроллера. Для записи текущего значения счетчика команд PC может быть выполнена команда PUSH. Выполнение этой команды приведет к увеличению указателя стека и записи текущего значения PC в вершину стека. Изменять значение вершины стека можно с помощью регистров TOSU, TOSH, TOSL, что дает возможность изменить адрес возврата.

С помощью команды POP указатель стека декрементируется, пропуская текущее значение в вершине стека и помещая в нее предыдущее значение без нарушения нормальной работы микроконтроллера.

4.2.4 Сброс микроконтроллера при переполнении/исчерпании стека

Разрешение этих видов сброса микроконтроллера устанавливается битом конфигурации STVREN. Когда бит STVREN=0, при переполнении/исчерпании стека будет установлен соответствующий флаг (STKFUL или STKUNL), но сброса микроконтроллера не произойдет. Если STVREN=1, то при переполнении/исчерпании стека устанавливается соответствующий флаг и выполняется сброс микроконтроллера. Биты STKFUL и STKUNL программно могут быть только сброшены в ‘0’. Эти биты равны нулю при сбросе по включению питания POR.

4.3 Быстрые регистры стека

«Быстрый возврат из прерываний» - опция, доступная для прерываний. Быстрые регистры стека предназначены для однократного сохранения регистров STATUS, WREG, BSR. Стек не доступен для записи и чтения, в него загружается текущие значения регистров при переходе по вектору прерываний. Значение регистров восстанавливается при выполнении команды возврата из прерываний FAST RETURN.

Сохранение регистров в стеке может происходить при возникновении прерывания низкого и высокого приоритета. Если прерывания с низким и высоким приоритетом используют функцию сохранения регистров в стеке, то эта функция может работать некорректно для обработчика прерываний с низким приоритетом. Если происходит прерывание с высоким приоритетом при обработке прерывания с низким приоритетом, то значение регистров, сохраненное при переходе на обработку прерываний с низким приоритетом, будет потеряно.

Если прерывания с высоким приоритетом не отключаются в обработчике прерываний с низким приоритетом, то пользователь при обслуживании прерывания с низким приоритетом должен сохранять основные регистры программным способом.

Если прерывания не используется, то пользователь может использовать функцию сохранения значения регистров STATUS, WREG, BSR при выполнении обычных подпрограмм. Для этого необходимо использовать команду FAST CALL.

В примере 4-1 показано использование сохранения значения в быстрых регистрах стека.

Пример 4-1. Использование быстрых регистров стека

```
   CALL SUB1, FAST     ;STATUS, WREG, BSR
                       ;Сохраняются в быстрых регистрах
                       ;стека
         •
         •
SUB1
         •
         •
   RETURN FAST         ;Восстановление значения регистров
                       ;сохраненных в быстрых регистрах стека
```

4.4 Регистры PCL, PCLATH и PCLATU

21-разрядный счетчик команд PC указывает адрес выполняемой команды в памяти программ. Младший счетчик команд PCL доступен для записи и чтения. Старший байт PCH содержит биты PC<15:8> и не доступен для записи и чтения. Обновление значение регистра PCH может быть выполнено через регистр PCLATH. Верхний байт PCU содержит биты PC<20:16> и не доступен для записи и чтения. Обновление значение регистра PCU может быть выполнено через регистр PCLATU.

Регистр счетчика команд PC адресует байты в памяти программ. Чтобы предотвратить смещение счетчика команд на один байт относительно команд микроконтроллера в памяти программ младший бит регистра PCL всегда равен ‘0’. Для адресации команд в памяти программ к счетчику команд всегда прибавляется число 2.

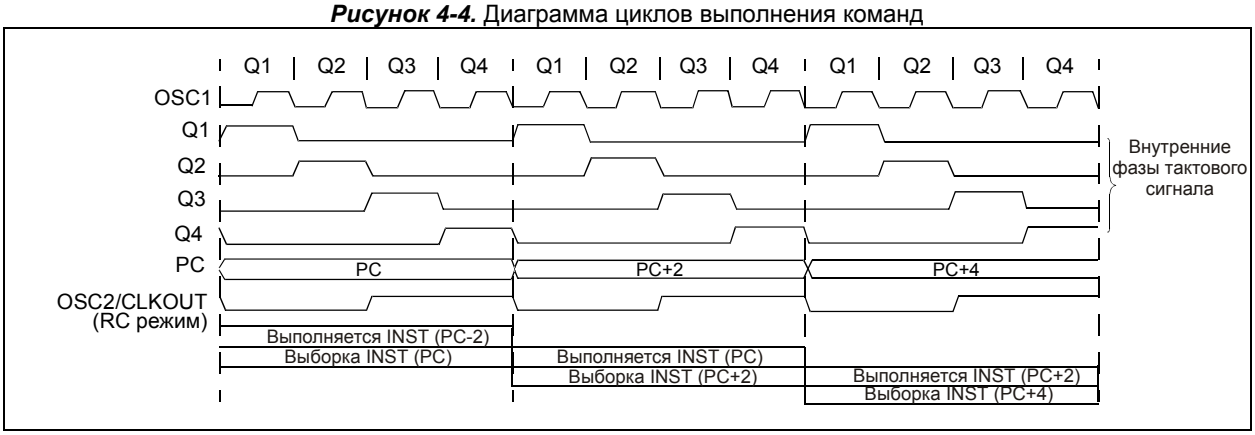
Команды CALL, RCALL, GOTO и команды возврата вызывают непосредственную запись в счетчик команд (значение регистров PCLATH, PCLATU не передается в счетчик команд).

Содержимое регистров PCLATH, PCLATU передается в счетчик команд при выполнении команды, выполняющей запись в регистр PCL. Значение регистров PCH, PCU соответственно помещается в регистры PCLATH, PCLATU при выполнении чтения регистра PCL, что может быть полезно при вычислении смещений счетчика команд PC (смотрите раздел 4.8.1).

4.5 Синхронизация выполнения команд

Входной тактовый сигнал (вывод OSC1) внутренней схемой микроконтроллера разделяется на четыре последовательных неперекрывающихся такта Q1, Q2, Q3 и Q4. Внутренний счетчик команд (PC) увеличивается на каждом такте Q1, а выборка команды из памяти программ происходит на каждом такте Q4. Декодирование и выполнение команды происходит с такта Q1 по Q4. На рисунке 4-4 показаны циклы выполнения команд.

Рисунок 4-4. Диаграмма циклов выполнения команд



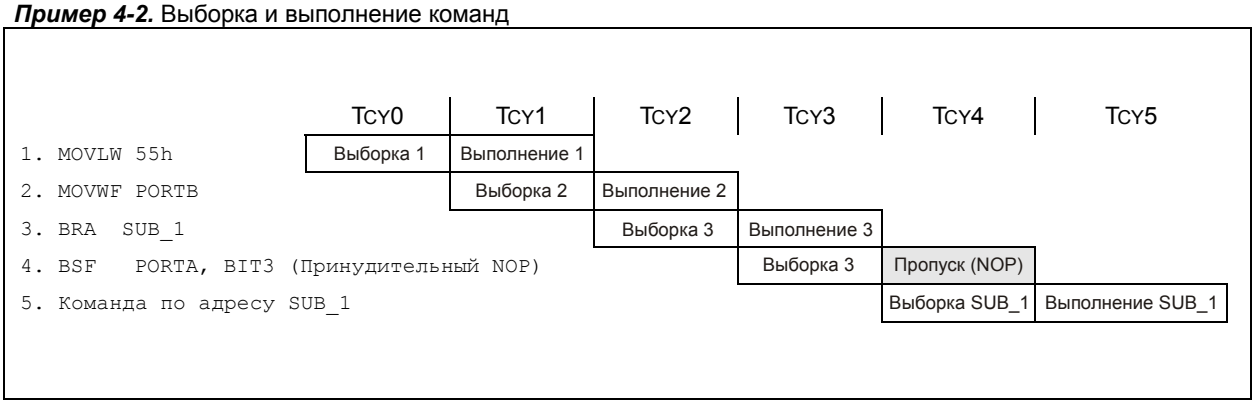
4.6 Конвейерная выборка и выполнение команд

Цикл выполнения команды состоит из четырех тактов Q1, Q2, Q3 и Q4. Выборка следующей команды и выполнение текущей совмещены по времени, таким образом, выполнение команды происходит за один цикл. Если команда изменяет счетчик команд PC (команды ветвления, например GOTO), то необходимо два машинных цикла для выполнения команды (см. пример 4-2).

Цикл выборки команды начинается с приращения счетчика команд PC 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 и Q4. Операнд из памяти данных читается в такте Q2, а результат выполнения команды записывается в такте Q4

Пример 4-2. Выборка и выполнение команд



Все команды выполняются за один цикл, кроме команд ветвления. Команды ветвления требуют два машинных цикла, т.к. необходимо удалить предварительно выбранную команду из конвейера. Во время удаления выбирается новая команда, а затем она исполняется в следующем машинном цикле.

4.7 Размещение команд в памяти программ

Память программ микроконтроллеров PIC18FXX2 адресуется побайтно. Команды в памяти программ сохраняются как два или четыре байта. Старший байт команды всегда располагается первым в памяти программ (младший бит адреса равен ‘0’). На рисунке 4-5 показан пример размещения команд в памяти программ. Чтобы всегда правильно делать выборку кода команды из памяти программ счетчик команд имеет приращение 2, а младший бит PC всегда читается как ‘0’ (смотрите раздел 4.4).

Команды CALL и GOTO имеют абсолютный адрес перехода в памяти программ, входящий в код команды. В качестве адреса перехода в коде команды используется адрес слова PC<20:1>, а не байта. На рисунке 4-5 показано как кодируется команда GOTO 000006h в памяти программ. Команды перехода, которые используют относительное смещение адреса, работают по аналогичному принципу. Значение смещения сохраняется в словах памяти программ. Дополнительное описание команд микроконтроллера смотрите в разделе 19.

4.7.1 Двухсловные команды

PIC18FXX2 имеет 4 двухсловных команды: MOVFF, CALL, GOTO и LFSR. Четыре старших бита второго слова подобных команд всегда имеют значение ‘1’, что соответствует команде NOP. Остальные 12 бит второго слова команды содержат данные, используемые командой. Если выполнено первое слово команды, то происходит обращение ко второму слову. Если второе слово команды выполняется отдельно (первое слово команды было пропущено), то оно будет выполнено как NOP. Эта предосторожность необходима, когда выполняется переход на двухсловную команду. В примере 4-3 показано использование двухсловных команд. Дополнительное описание команд микроконтроллера смотрите в разделе 19.

Пример 4-3. Двухсловные команды

Случай 1:

Код	**Исходный код**
0110 0110 0000 0000	`TSTFSZ REG1 ; Значение регистра ОЗУ 0?`
1100 0001 0010 0011	`MOVFF REG1, REG2 ; Нет, выполнить двухсловную команду`
1111 0100 0101 0110	`; во 2-м операнде адрес регистра REG2`
0010 0100 0000 0000	`ADDWF REG3 ; продолжение кода`

Случай 2:

Код	**Исходный код**
0110 0110 0000 0000	`TSTFSZ REG1 ; Значение регистра ОЗУ 0?`
1100 0001 0010 0011	`MOVFF REG1, REG2 ; Да`
1111 0100 0101 0110	`; 2-й операнд выполняется как NOP`
0010 0100 0000 0000	`ADDWF REG3 ; продолжение кода`

4.8 Таблицы

Таблицы в памяти программ могут быть реализованы двумя методами:

- Вычисленный переход
- Чтение/запись таблиц

4.8.1 Вычисленный переход

Вычисляемый переход выполняется, добавляя смещение к счетчику команд (ADDWF PCL).

Таблица может быть реализована на основе одной команды ADDWF PCL и группе команд RETLW 0xNN. Перед запросом таблицы в регистр WREG загружается смещение в таблице. Как правило, первой командой является ADDWF PCL. Следующей будет одна из команд RETLW 0xNN, которая возвращает значение 0xNN вызываемой процедуре.

Значение смещения (регистр WREG) указывает на какое число слов необходимо сместить счетчик команд.

Данным методом в одном слове программы можно сохранить только один байт данных. Необходимо учитывать, что в стеке должен размещаться адрес возврата.

4.8.2 Чтение/запись таблиц

Этот метод является наиболее предпочтительным, поскольку позволяет сохранить два байта данных в одном слове памяти программ.

Данные таблицы могут быть сохранены (прочитаны) в памяти программ используя функции табличной записи (чтения). Указатель таблицы (TBLPTR) содержит адрес байта в памяти программ, а защелка TABLAT данные, которые прочитаны или должны быть сохранены в памяти программ. Единовременно может быть записан/прочитан только один байт данных.

Дополнительную информацию по операциям табличного чтения/записи смотрите в разделе 5.

4.9 Организация памяти данных

Память данных реализована как статическое ОЗУ. Каждый регистр в памяти данных имеет 12-разрядный адрес, что позволяет адресовать до 4096 байт памяти данных. На рисунках 4-6, 4-7 показана организация памяти данных микроконтроллеров PIC18FXX2.

Память данных разделена на 16 банков, каждый из которых содержит по 256 байт. Младшие 4 бита регистра BSR используются для выбора текущего банка памяти (BSR<3:0>). Старшие 4 бита регистра BSR не реализованы.

Память данных содержит регистры специального (SFR) и общего (GPR) назначения. Регистры SFR используются для управления ядром и периферийных модулей микроконтроллера, в то время как GPR используются для хранения данных пользователя. Регистры SFR начинаются с последнего байта 15-го банка памяти данных (0xFFF) и распространяются вниз по карте памяти. Любой незадействованный регистр в области SFR может использоваться как регистр общего назначения. Регистры GPR начинаются в первом байте 0-го банка памяти данных и распространяются вверх по карте памяти. Чтение не реализованной памяти данных будет давать результат ‘0’.

К любому регистру памяти данных можно обратиться непосредственно или косвенно. При прямой адресации может потребоваться настройка регистра BSR. Косвенная адресация требует настройки регистров FSRn и обращение к памяти через соответствующий регистр INDFn. Каждый регистр FSR содержит 12-разрядный адрес регистра в памяти программ, что позволяет выполнять косвенную адресацию без переключения банков памяти данных.

Система команд PIC18FXX2 позволяет выполнять операции с регистрами во всей области памяти программ. Это может быть выполнено с помощью косвенной адресации или командой MOVFF. Команда MOVFF является двухсловной и двухцикловой, она перемещает значение одного регистра к другому.

Для обращения за один машинный цикл к регистрам специального и части регистров общего назначения был реализован банк памяти быстрого доступа. Независимо от текущего значения регистра BSR происходит обращение к части банка 0 и банка 15. Подробное описание памяти быстрого доступа смотрите в разделе 4.10.

4.9.1 Регистры общего назначения GPR

К регистрам общего назначения можно обратиться непосредственно или косвенно. Косвенная адресация требует настройки регистров FSR и обращение через регистр INDF. Описание операции косвенной адресации смотрите в разделе 4.12.

Регистры общего назначения имеют организацию в памяти данных по банкам, они не инициализируются при сбросе по включению питания, а при остальных видах сброса не изменяют своего значения.

Память данных доступна для обращения всеми командами микроконтроллера. Старшая часть банка 15 содержит регистры SFR, все остальные банки содержат регистры GPR (начиная с банка 0).

4.9.2 Регистры специального назначения SFR

Регистры специального назначения предназначены для управления ядром микроконтроллера и периферийными модулями. Эти регистры реализованы как статическое ОЗУ. Список регистров специального назначения представлен в таблицах 4-1, 4-2.

Регистры SFR разделяются на две основные группы: управление ядром микроконтроллера; управление периферийными модулями микроконтроллера. Регистры, которые управляют ядром микроконтроллера, описаны в этом разделе. Описание регистров, связанных с работой периферийных модулей, смотрите в соответствующем разделе документации.

Не реализованные регистры SFR будут читаться как ‘0’. Адреса регистров специального назначения смотрите в таблице 4-1.

Рисунок 4-6. Карта памяти данных микроконтроллеров PIC18F242/442

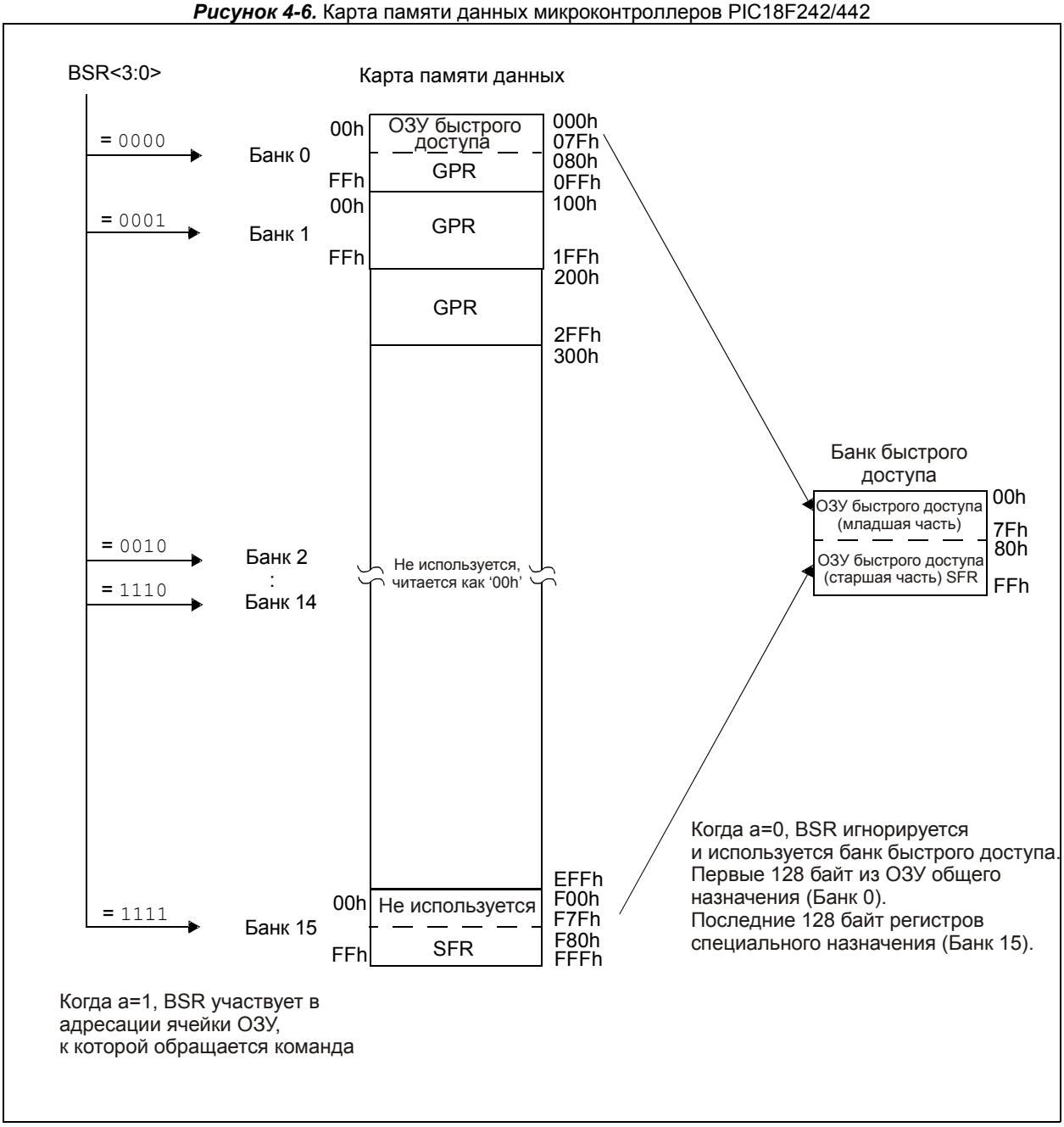


Рисунок 4-7 Карта памяти данных микроконтроллеров PIC18F252/452

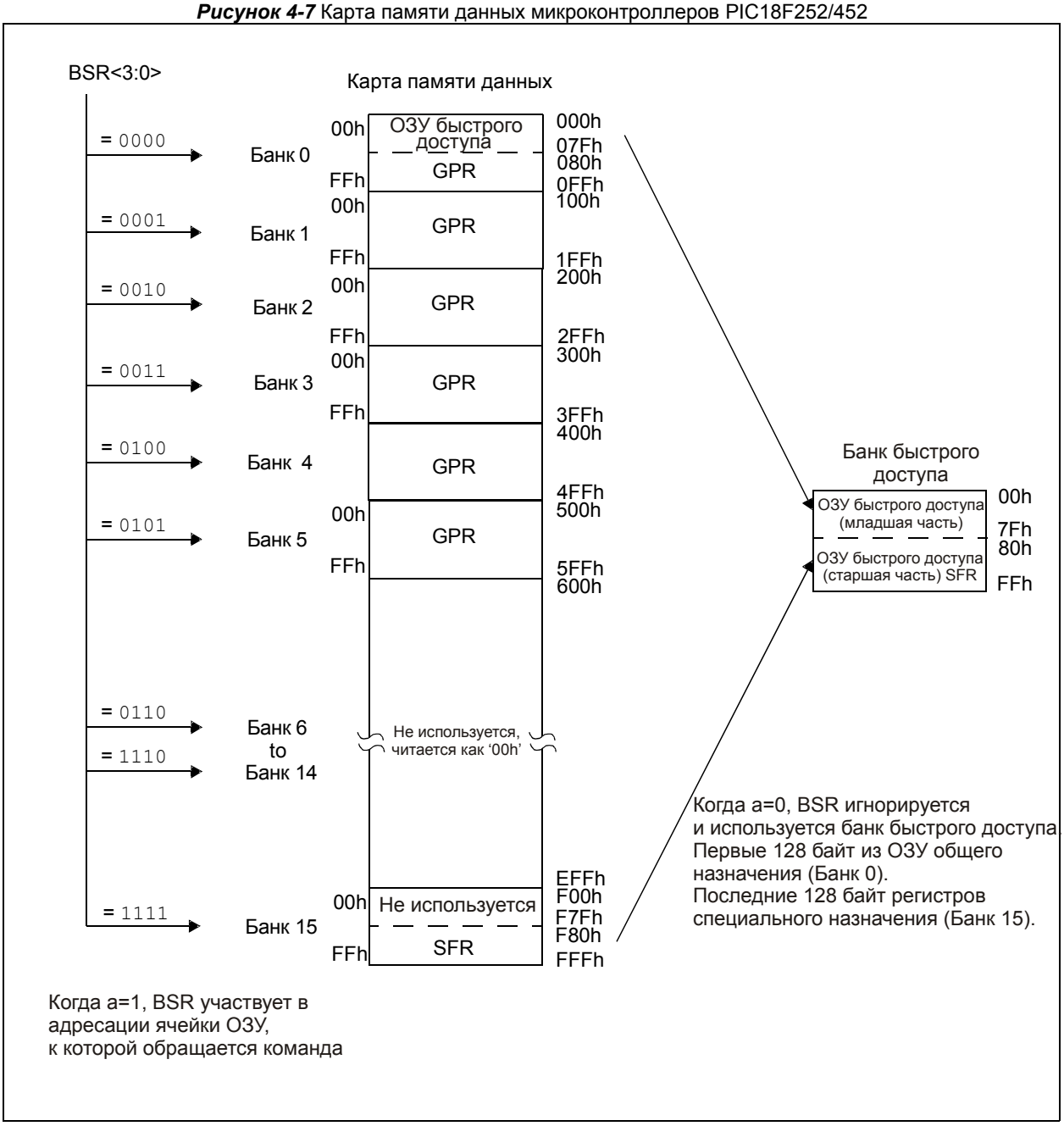


Таблица 4-1. Карта памяти регистров специального назначения

Адрес	Имя	Адрес	Имя	Адрес	Имя	Адрес	Имя
FFFh	TOSU	FDFh	INDF2(3)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(3)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(3)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(3)	FBCh	CCPR2H	F9Ch	—
FFBh	PCLATU	FDBh	PLUSW2(3)	FBBh	CCPR2L	F9Bh	—
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—
FF9h	PCL	FD9h	FSR2L	FB9h	—	F99h	—
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—	F98h	—
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—	F97h	—
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—	F96h	TRISE(2)
FF5h	TABLAT	FD5h	T0CON	FB5h	—	F95h	TRISD(2)
FF4h	PRODH	FD4h	—	FB4h	—	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	LVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—
FF0h	INTCON3	FD0h	RCON	FB0h	—	F90h	—
FEFh	INDF0(3)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—
FEEh	POSTINC0(3)	FCEh	TMR1L	FAEh	RCREG	F8Eh	—
FEDh	POSTDEC0(3)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE(2)
FECh	PREINC0(3)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD(2)
FEBh	PLUSW0(3)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	—	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	—
FE7h	INDF1(3)	FC7h	SSPSTAT	FA7h	EECON2	F87h	—
FE6h	POSTINC1(3)	FC6h	SSPCON1	FA6h	EECON1	F86h	—
FE5h	POSTDEC1(3)	FC5h	SSPCON2	FA5h	—	F85h	—
FE4h	PREINC1(3)	FC4h	ADRESH	FA4h	—	F84h	PORTE(2)
FE3h	PLUSW1(3)	FC3h	ADRESL	FA3h	—	F83h	PORTD(2)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	—	FA0h	PIE2	F80h	PORTA

Примечания:
1. Не реализован, читается как ‘0’.
2. Этот регистр не реализован в микроконтроллерах PIC18F2X2.
3. Не физический регистр.

Таблица 4-2. Регистры специального назначения

Обозначение	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
TOSU	-	-	-	Вершина стека верхний байт (TOS<20:16>)					---0 0000
TOSH	Вершина стека старший байт (TOS<20:16>)								0000 0000
TOSL	Вершина стека младший байт (TOS<20:16>)								0000 0000
STKPTR	STKFUL	STKUNF	-	Указатель стека возврата					00-0 0000
PCLATU	-	-	-	Регистр защелка для PC<20:16>					---0 0000
PCLATH	Регистр защелка для PC<15:8>								0000 0000
PCL	Младший байт PC (PC<7:0>)								0000 0000
TBLPTRU	-	-	Бит 21(2)	Указ. табл. памяти программ верхний байт (TBLPTR<20:16>)					--00 0000
TBLPTRH	Указатель таблицы памяти программ старший байт (TBLPTR<15:8>)								0000 0000
TBLPTRL	Указатель таблицы памяти программ младший байт (TBLPTR<7:0>)								0000 0000
TABLAT	Защелка таблицы памяти программ								0000 0000
PRODH	Результат умножения старший байт								xxxx xxxx
PRODL	Результат умножения младший байт								xxxx xxxx
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	-	TMR0IP	-	RBIP	1111 -1-1
INTCON3	INT2IP	INT1IP	-	INT2IE	INT1IE	-	INT2IF	INT1IF	11-0 0-00
INDF0	Используется для обращения к регистру с адресом в FSR0 – FSR0 не изменяется (нефизический регистр)								-
POSTINC0	Используется для обращения к регистру с адресом в FSR0 – FSR0 пост-инкремент (нефиз. регистр)								-
POSTDEC0	Используется для обращения к регистру с адресом в FSR0 – FSR0 пост-декремент (нефиз. регистр)								-
PREINC0	Используется для обращения к регистру с адресом в FSR0 – FSR0 пред-инкремент (нефиз. регистр)								-
PLUSW0	Используется для обращения к регистру с адресом в FSR0 – FSR0 пред -инкремент (нефиз. регистр) к значению FSR0 добавляется смещение из WREG								-
FSR0H	-	-	-	-	Указатель косвенной адресации 0 старший байт				---- xxxx
FSR0L	Указатель косвенной адресации 0 младший байт								xxxx xxxx
WREG	Рабочий регистр								xxxx xxxx
INDF1	Используется для обращения к регистру с адресом в FSR1 – FSR1 не изменяется (нефизический регистр)								-
POSTINC1	Используется для обращения к регистру с адресом в FSR1 – FSR1 пост-инкремент (нефиз. регистр)								-
POSTDEC1	Используется для обращения к регистру с адресом в FSR1 – FSR1 пост-декремент (нефиз. регистр)								-
PREINC1	Используется для обращения к регистру с адресом в FSR1 – FSR1 пред-инкремент (нефиз. регистр)								-
PLUSW1	Используется для обращения к регистру с адресом в FSR1 – FSR1 пред -инкремент (нефиз. регистр) к значению FSR1 добавляется смещение из WREG								-
FSR1H	-	-	-	-	Указатель косвенной адресации 1 старший байт				---- xxxx
FSR1L	Указатель косвенной адресации 1 младший байт								xxxx xxxx
BSR	-	-	-	-	Регистр выбора банка памяти				---- 0000
INDF2	Используется для обращения к регистру с адресом в FSR2 – FSR2 не изменяется (нефизический регистр)								-
POSTINC2	Используется для обращения к регистру с адресом в FSR2 – FSR2 пост-инкремент (нефиз. регистр)								-
POSTDEC2	Используется для обращения к регистру с адресом в FSR2 – FSR2 пост-декремент (нефиз. регистр)								-
PREINC2	Используется для обращения к регистру с адресом в FSR2 – FSR2 пред-инкремент (нефиз. регистр)								-
PLUSW2	Используется для обращения к регистру с адресом в FSR2 – FSR2 пред -инкремент (нефиз. регистр) к значению FSR2 добавляется смещение из WREG								-
FSR2H	-	-	-	-	Указатель косвенной адресации 2 старший байт				---- xxxx
FSR2L	Указатель косвенной адресации 2 младший байт								xxxx xxxx
STATUS	-	-	-	N	OV	Z	DC	C	---x xxxx
TMR0H	Регистр таймера 0 старший байт								0000 0000
TMR0L	Регистр таймера 0 младший байт								xxxx xxxx
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111
OSCCON	-	-	-	-	-	-	-	SCS	---- ---0
LVDCON	-	-	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0	--00 0101
WDTCON	-	-	-	-	-	-	-	SWDTE	---- ---0
RCON	IPEN	-	-	-RI	-TO	-PD	-POR	-BOR	0--1 11qq
TMR1H	Регистр таймера 1 старший байт								xxxx xxxx
TMR1L	Регистр таймера 1 младший байт								xxxx xxxx
T1CON	RD16	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON	0-00 0000
TMR2	Регистр таймера 2								0000 0000
PR2	Регистр периода таймера 2								1111 1111
T2CON	-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000

Примечания:

1. RA6 доступен только в RICO и ECIO режиме тактового генератора, в остальных режимах генератора читается как ‘0’.
2. Бит 21 в регистре TBLPTRU определяет доступ к битам конфигурации.
3. Эти биты и регистры в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

Таблица 4-2. Регистры специального назначения (продолжение)

Обозначение	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
SSPBUF	SSP приемный буфер / регистр передатчика								xxxx xxxx
SSPADD	SSP регистр адреса в режиме ведомого I^2C. SSP регистр скорости обмена в режиме ведущего I^2C.								0000 0000
SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000
ADRESH	Старший байт результата преобразования АЦП								xxxx xxxx
ADRESL	Младший байт результата преобразования АЦП								xxxx xxxx
ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/-DONE	-	ADON	0000 00-0
ADCON1	ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000
CCPR1H	Регистр 1 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
CCPR1L	Регистр 1 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
CCP1CON	-	-	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000
CCPR2H	Регистр 2 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
CCPR2L	Регистр 2 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
CCP2CON	-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000
TMR3H	Регистр таймера 3 старший байт								xxxx xxxx
TMR3L	Регистр таймера 3 младший байт								xxxx xxxx
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	-T3SYNC	TMR3CS	TMR3ON	0000 0000
SPBRG	Регистр скорости обмена USART								0000 0000
RCREG	Регистр приемника USART								0000 0000
TXREG	Регистр передатчика USART								0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 -010
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
EEADR	Регистр адреса EEPROM памяти								0000 0000
EEDATA	Регистр данных EEPROM памяти								0000 0000
EECON1	EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD	xx-0 x000
EECON2	Управляющий регистр 2 EEPROM памяти (нефизический регистр)								---- ----
IPR2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000
IRP1	PSPIP(3)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111
PIR1	PSPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
PIE1	PSPIE(3)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
TRISE(3)	IBF	OBF	IBOV	PSPMODE	-	Направление данных для PORTE			0000 -111
TRISD(3)	Направление данных для PORTD								1111 1111
TRISC	Направление данных для PORTC								1111 1111
TRISB	Направление данных для PORTB								1111 1111
TRISA	-	TRISA6(1)	Направление данных для PORTA						-111 1111
LATE(3)	-	-	-	-	-	Чтение защелки PORTE, запись в защелку PORTE			---- -xxx
LATD(3)	Чтение защелки PORTD, запись в защелку PORTD								xxxx xxxx
LATC	Чтение защелки PORTC, запись в защелку PORTC								xxxx xxxx
LATB	Чтение защелки PORTB, запись в защелку PORTB								xxxx xxxx
LATA	-	LATA6(1)	Чтение защелки PORTA, запись в защелку PORTA						-xxx xxxx
PORTE(3)	Чтение с выводов PORTE, запись в защелку PORTE								---- -000
PORTD(3)	Чтение с выводов PORTD, запись в защелку PORTD								xxxx xxxx
PORTC	Чтение с выводов PORTC, запись в защелку PORTC								xxxx xxxx
PORTB	Чтение с выводов PORTB, запись в защелку PORTB								xxxx xxxx
PORTA	-	RA6(1)	Чтение с выводов PORTA, запись в защелку PORTA						-x0x 0000

Примечания:

1. RA6 доступен только в RICO и ECIO режиме тактового генератора, в остальных режимах генератора читается как ‘0’.
2. Бит 21 в регистре TBLPTRU определяет доступ к битам конфигурации.
3. Эти биты и регистры в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

4.10 Банк памяти быстрого доступа

Банк памяти быстрого доступа – архитектурное решение, которое является особенно полезно для оптимизации кода при написании программ на языке С. Методы, используемые компилятором С, могут быть также полезны для программ, написанных на ассемблере.

Эта область памяти может использоваться для:

- Хранение промежуточных значений вычислений
- Отдельные служебные переменные
- Быстрого доступа к отдельным переменным
- Обычные переменные
- Быстрый доступ к регистрам специального назначения

Банк памяти быстрого доступа содержит старших 128 байт банка 15 (регистры специального назначения) и нижних 128 байт банка 0 памяти данных. Две секции в банке памяти быстрого доступа называются нижняя и верхняя область банка. На рисунках 4-6, 4-7 показана область в памяти данных для банка быстрого доступа. В слове команды определяется, как должна выполняться адресация к памяти данных – банк выбирается с учетом регистра BSR или обращение к банку быстрого доступа. Бит, определяющий правило доступа к памяти, обозначается как ‘a’.

Когда необходимо выполнить обращение к банку прямого доступа, бит a=0. Обращение к регистрам специального назначения можно выполнять без изменения текущего банка памяти данных, что очень удобно при проверки флагов и изменении управляющих битов.

4.11 Регистр выбора банка памяти данных BSR

Потребность в большем объеме памяти данных определяет наличие разделения ОЗУ на банки памяти. Вся память данных разделена на 16 банков. При использовании непосредственной адресации необходимо настроить регистр BSR для обращения к нужному банку.

BSR<3:0> содержит 4 старших бита 12-разрядного адреса регистра в памяти данных. BSR<7:4> всегда читаются как ‘0’, а запись не будет иметь никакого эффекта.

С помощью команды MOVLB можно выбрать необходимый банк памяти данных.

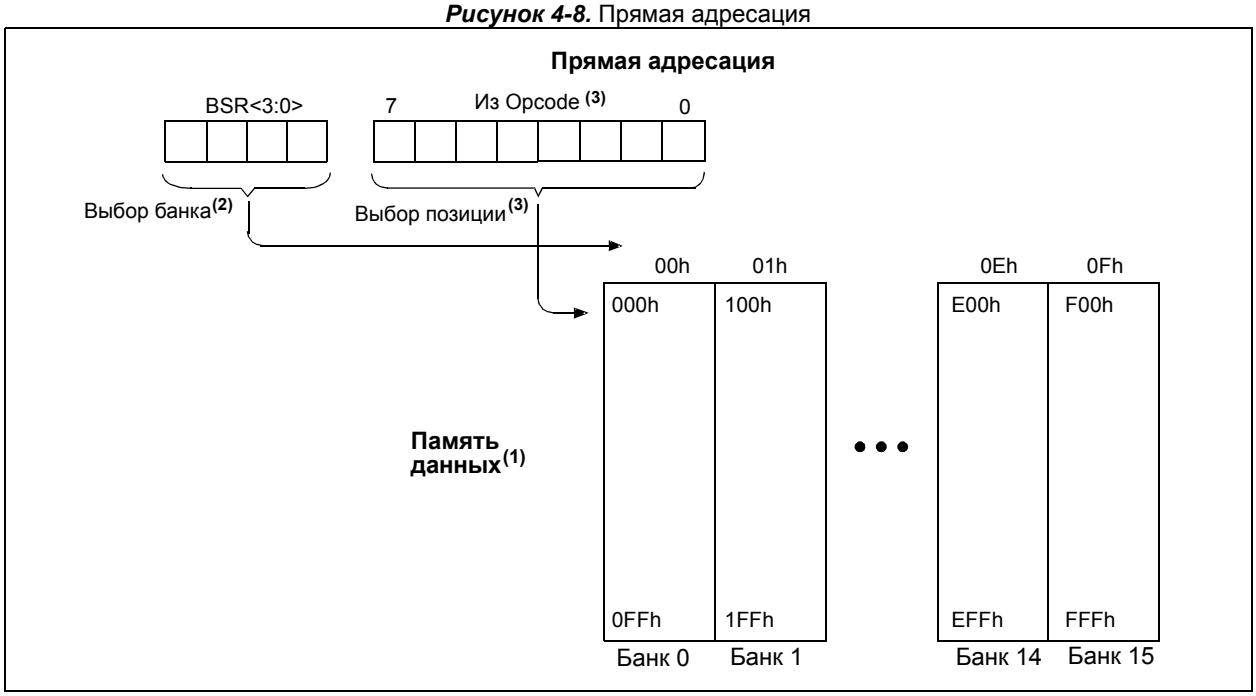
Если выбранный банк не реализован, то чтение будет давать результат ‘0’, а любая запись игнорируется. Биты регистра STATUS будут изменяться в соответствии с выполняемой командой.

Каждый банк памяти данных имеет внутренние адреса от 00h до FFh (256 байт). Вся память данных реализована как статическое ОЗУ.

Команда MOVFF игнорирует содержимое регистра BSR, т.к. значение 12-разрядных адресов содержится в коде команды.

Метод косвенной адресации (смотрите раздел 4.12) позволяет линейно адресовать всю область памяти данных.

Рисунок 4-8. Прямая адресация



Примечания:

1. Детальную карту памяти данных смотрите в таблице 4-1.
2. В команде может использоваться бит быстрого доступа, чтобы игнорировать содержимое регистра BSR(<3:0>) и получить доступ к банку быстрого доступа.
3. Команда MOVFF игнорирует содержимое регистра BSR, т.к. значение 12-разрядных адресов содержится в коде команды.

4.12 Косвенная адресация, регистры INDF и FSR

Косвенная адресация – режим адресации памяти данных, когда адрес регистра не включается в код команды. Регистр FSR используется как указатель ячейки в памяти программ, которая должна быть прочитана или в которую должно быть записано новое значение. Указатель размещается в памяти данных, поэтому может быть изменен программным способом. Этот метод адресации может быть полезен для операций с таблицами, размещенными в памяти данных. Механизм косвенной адресации показан на рисунке 4-9 (запись данных в регистр, адрес которого указан в FSR).

Косвенная адресация возможна при использовании одного из регистров INDF. Любая команда, использующая регистр INDF фактически обращается к регистру, указанному в FSR. Косвенное чтение регистра INDF будет давать результат 00h. Косвенная запись в регистр INDF не вызовет никаких действий. Регистр FSR содержит 12-разрядный адрес ячейки в памяти данных (смотрите рисунок 4-10).

Регистр INDFn – не физический регистр. Обращение к регистру INDFn фактически вызовет действие с регистром, адрес которого указан в FSRn (принцип косвенной адресации).

Последовательность очистки памяти данных в банке 1 с адреса 100h по 1FFh минимальным числом команд смотрите в примере 4-4.

Пример 4-4. Очистка памяти данных с использованием косвенной адресации

```
            LFSR   FSR0 ,0x100  ;
NEXT        CLRF   POSTINC0     ; Очистить регистр INDF
                                ; и инкрементировать
                                ; указатель
            BTFSS  FSR0H, 1     ; Все регистры очищены
                                ; в банке 1?
            GOTO   NEXT         ; Нет, очистить следующий регистр
CONTINUE                        ; Да, продолжить программу
```

В микроконтроллерах PIC18FXX2 реализовано три 12-разрядных регистра косвенной адресации, для обращения ко всей области памяти данных (4096 байт):

1. FSR0 состоит из FSR0H:FSR0L
2. FSR1 состоит из FSR1H:FSR1L
3. FSR2 состоит из FSR2H:FSR2L

Дополнительно есть регистры INDF0, INDF1 и INDF2, которые физически не реализованы. Обращение к этим регистрам фактически вызовет действие с регистром, адрес которого указан в FSR. Если команда выполняет запись в регистр INDF0, то данные будут записаны в регистр, адрес которого указан в FSR0H:FSR0L. Чтение регистра INDF1 возвратит значение регистра, адрес которого указан в FSR1H:FSR1L. Регистры INDFn могут использоваться как операнды команд.

Если INDF0, INDF1, INDF2 читаются косвенно через FSR, то чтение будет давать результат ‘0’. Операция косвенной записи в регистры INDF0, INDF1, INDF2 будет эквивалентна команде NOP, и на биты регистра STATUS влияния не окажет.

4.12.1 Операция косвенной адресации

Каждому регистру FSR соответствует регистр INDF, плюс еще четыре дополнительных регистра, которые определяют, как изменится FSR при выполнении косвенной адресации:

- При косвенной адресации регистр FSRn не изменяется (обращение к INDFn)
- Автодекремент FSRn после косвенной адресации (обращение к POSTDECn)
- Автоинкремент FSRn после косвенной адресации (обращение к POSTINCn)
- Автоинкремент FSRn перед косвенной адресацией (обращение к PREINCn)
- Значение в регистре WREG используется как смещение к FSRn. После косвенной адресации значение WREG и FSR не изменяется (обращение к PLUSWn)

Состояние FSR не отображается в регистре STATUS при использовании автоинкремента или декремента при косвенной адресации. Например, если значение в FSR становиться равным ‘0’, то бит Z не будет установлен в ‘1’.

Инкремент и декремент FSR затрагивает все 12 разрядов адреса. Например, переполнение FSRnL вызовет автоматическое увеличение FSRnH. Эти особенности позволяют использовать FSRn как указатель программного стека, в дополнение к операциям с таблицами в памяти данных.

Обращение к PLUSWn позволяет реализовать индексированную косвенную адресацию. К регистру FSR добавляется значение регистра WREG, чтобы сформировать адрес ячейки. Значение регистра при этом не изменяется.

Если регистр FSR содержит значение, которое указывает на один из регистров INDFn, косвенное чтение будет давать результат ‘0’, а запись эквивалентна команде NOP (биты регистра STATUS не изменяются).

Если адресатом при косвенной адресации являются регистры FSRnH или FSRnL, то операция записи имеет более высокий приоритет, чем автоинкремент и автодекремент.

Рисунок 4-9. Механизм косвенной адресации

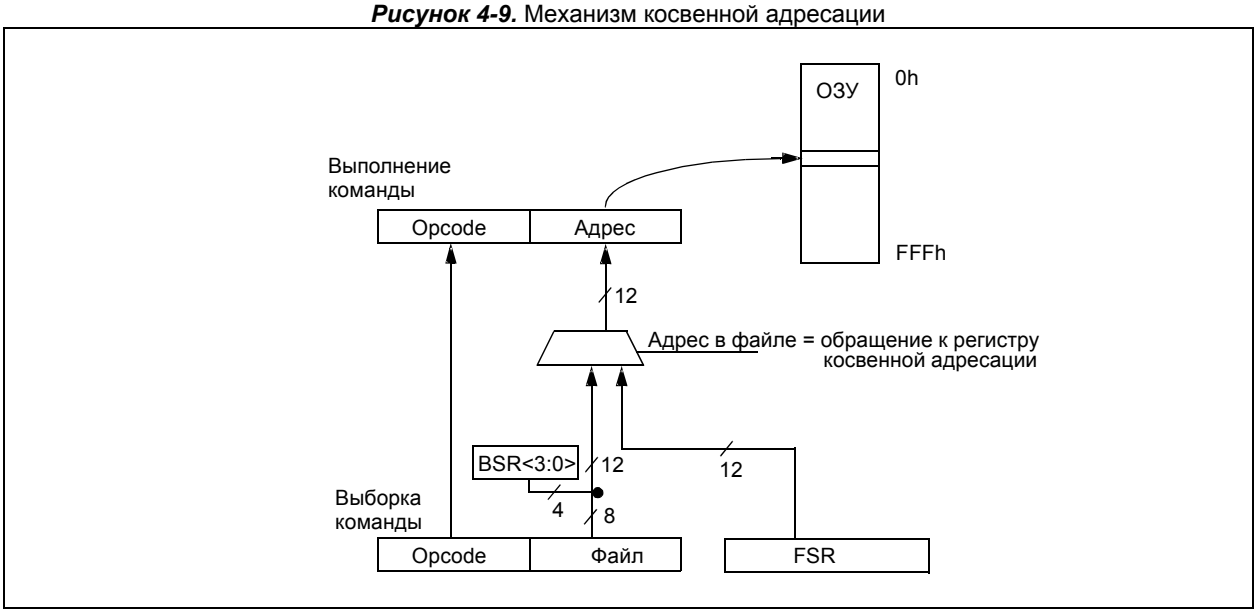
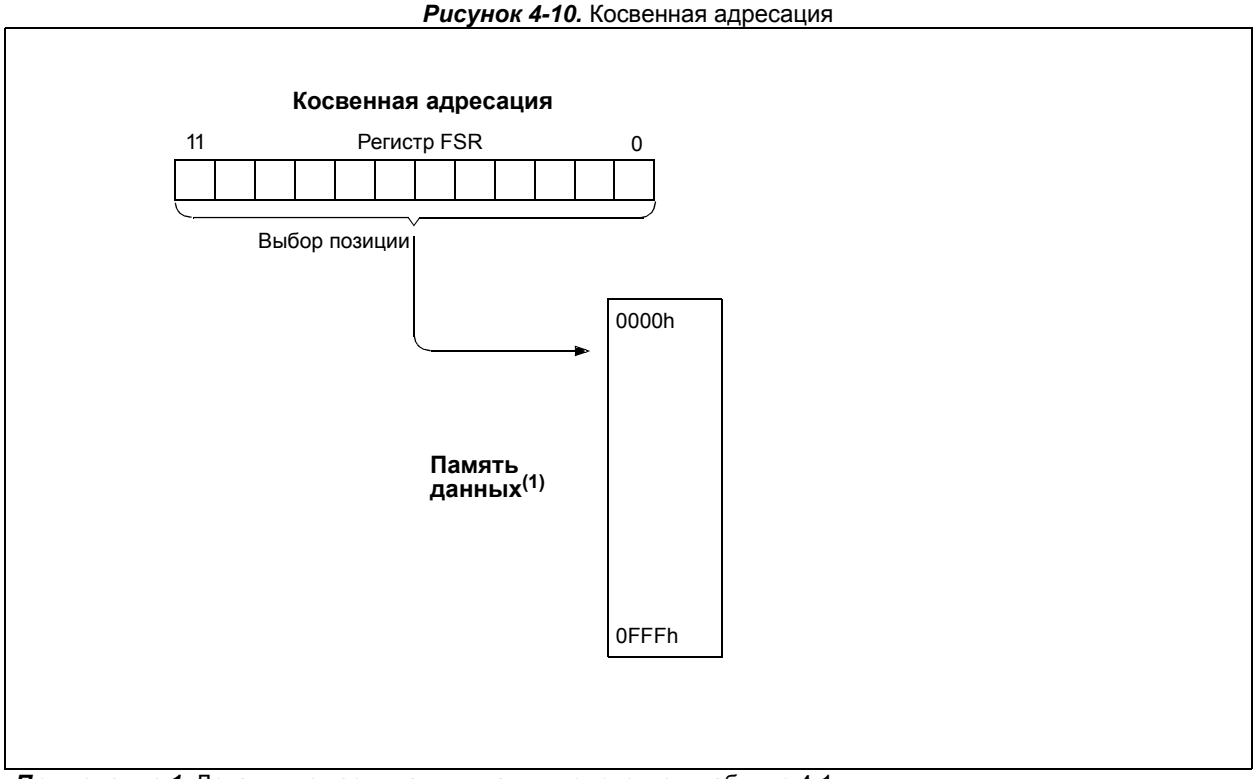


Рисунок 4-10. Косвенная адресация



Примечание 1. Детальную карту памяти данных смотрите в таблице 4-1.

4.13 Регистр STATUS

Регистр STATUS содержит флаги состояния АЛУ. Регистр STATUS может быть адресован любой командой, как и любой другой регистр памяти данных. Если обращение к регистру STATUS выполняется командой, которая воздействует на флаги Z, DC, C, OV или N, то изменение этих битов командой заблокировано. Эти биты изменяются согласно логике ядра микроконтроллера. Поэтому, результат выполнения команды с регистром STATUS может отличаться от ожидаемого.

Например, команда CLRF STATUS только установит в ‘1’ бит Z (состояние регистра STATUS после выполнения команды 000u u1uu, где u – не изменяемый бит).

При изменении битов регистра STATUS рекомендуется использовать команды, не влияющие на флаги АЛУ (BCF, BSF, SWAPF, MOVWF и MOVFF). Полный список команд, не влияющих на флаги АЛУ (Z, C, DC, OV и N), смотрите в таблице 20-2.

Примечание. Флаги C и DC используются как биты заема и десятичного заема соответственно, например, при выполнении команд вычитания.

Регистр 4-2. Регистр STATUS

U - 0	U - 0	U - 0	R/W - x	R/W - x	R/W - x	R/W - x	R/W - x
-	-	-	N	OV	Z	DC	C
Бит 7							Бит 0

Бит 7-5 **Не используется:** Читается как ‘0’

Бит 4 **N:** Флаг отрицательного результата
Этот бит используется для арифметики дополнения до 2. Флаг указывает на отрицательный результат (АЛУ MSB=1)
1 = отрицательный результат
0 = положительный результат

Бит 3 **OV:** Флаг переполнения
Этот бит используется для арифметики дополнения до 2. Флаг указывает на переполнение 7-разрядного значения, что привело к изменению старшего бита байта
1 = произошло переполнение в арифметической операции
0 = переполнения не было

Бит 2 **Z:** Флаг нулевого результата
1 = нулевой результат арифметической или логической операции
0 = результат арифметической или логической операции не нулевой

Бит 1 **DC:** Флаг десятичного переноса/заема
1 = был перенос из младшего полубайта
0 = не было переноса из младшего полубайта

Примечание. Флаг заема имеет инверсное значение. Вычитание выполняется путем прибавления дополнительного кода второго операнда. При выполнении команд сдвига (RRF, RLF) бит DC загружается 3-м или 4-м битом сдвигаемого регистра.

Бит 0 **C:** Флаг переноса/заема
1 = был перенос из старшего бита
0 = не было переноса из старшего бита

Примечание. Флаг заема имеет инверсное значение. Вычитание выполняется путем прибавления дополнительного кода второго операнда. При выполнении команд сдвига (RRF, RLF) бит C загружается старшим или младшим битом сдвигаемого регистра.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

4.14 Регистр RCON

В регистре RCON содержатся биты, с помощью которых можно определить причину сброса микроконтроллера (-TO, -PD, -POR, -BOR и -RI). Регистр доступен для записи и чтения.

Примечания:

1. Если бит BODEN в слове конфигурации установлен в ‘1’, то бит -BOR будет устанавливаться в ‘1’ при сбросе по включению питания. После сброса по снижению напряжения питания бит -BOR=0, он должен быть установлен программой пользователя для обнаружения последующих сбросов BOR.
2. Рекомендуется устанавливать в ‘1’ бит -POR после обнаружения сброса по включению питания, чтобы была возможность обнаружить последующие сбросы POR.

Регистр 4-3. Регистр RCON

R/W - 0	U - 0	U - 0	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
IPEN	-	-	-RI	-TO	-PD	-POR	-BOR
Бит 7							Бит 0

Бит 7 **IPEN:** Разрешение приоритетной системы прерываний
1 = приоритетная система прерываний разрешена
0 = приоритетная система прерываний выключена (для совместимости с PIC16CXXX)

Бит 6-5 **Не используется:** Читается как ‘0’

Бит 4 **-RI:** Флаг выполнения команды RESET
1 = команда RESET не выполнялась
0 = сброс микроконтроллера произошел по выполнению команды RESET
(бит должен быть установлен в ‘1’ после сброса BOR)

Бит 3 **-TO:** Флаг переполнения сторожевого таймера WDT
1 = после сброса POR, выполнения команды CLRWDT или SLEEP
0 = произошло переполнение WDT

Бит 2 **-PD:** Флаг детектора выключения питания
1 = после сброса POR или выполнения команды CLRWDT
0 = после выполнения команды SLEEP

Бит 1 **-POR:** Флаг сброса по включению питания POR
1 = сброса по включению питания не происходило
0 = произошел сброс по включению питания
(бит должен быть установлен в ‘1’ после сброса POR)

Бит 0 **-BOR:** Флаг сброса по снижению напряжения питания
1 = сброса по снижению напряжения питания не происходило
0 = произошел сброс по снижению напряжения питания
(бит должен быть установлен в ‘1’ после сброса BOR)

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

5. Flash память программ

Flash память программ доступна для записи, чтения и стирания во время нормальной работы микроконтроллера во всем диапазоне допустимого напряжения питания V_{DD}.

За одну операцию из памяти программ можно прочитать один байт данных. Операция записи выполняется по блочно, 8 байт в одном блоке. Стирание памяти программ выполняется блоками по 64 байта. Операция стирания всей памяти одной командой не может быть сформирована программой пользователя.

При выполнении записи и стирания памяти программ выполнение текущей программы приостанавливается, пока не закончится цикл записи/стирания. К памяти программ нельзя обращаться во время записи/стирания, поэтому выполнение программы приостанавливается. Операция записи/стирания выполняется от отдельного таймера.

Значения, сохраняемые в памяти программ, не обязательно должны иметь силу команды (может быть сохранена таблица данных). Если при выполнении программы встречается некорректный код команды, то эта команда будет выполнена как NOP.

5.1 Табличное чтение и табличная запись

Для передачи данных между памятью данных и памятью программ предусмотрено две операции:

- Табличное чтение (TBLRD)
- Табличная запись (TBLWT)

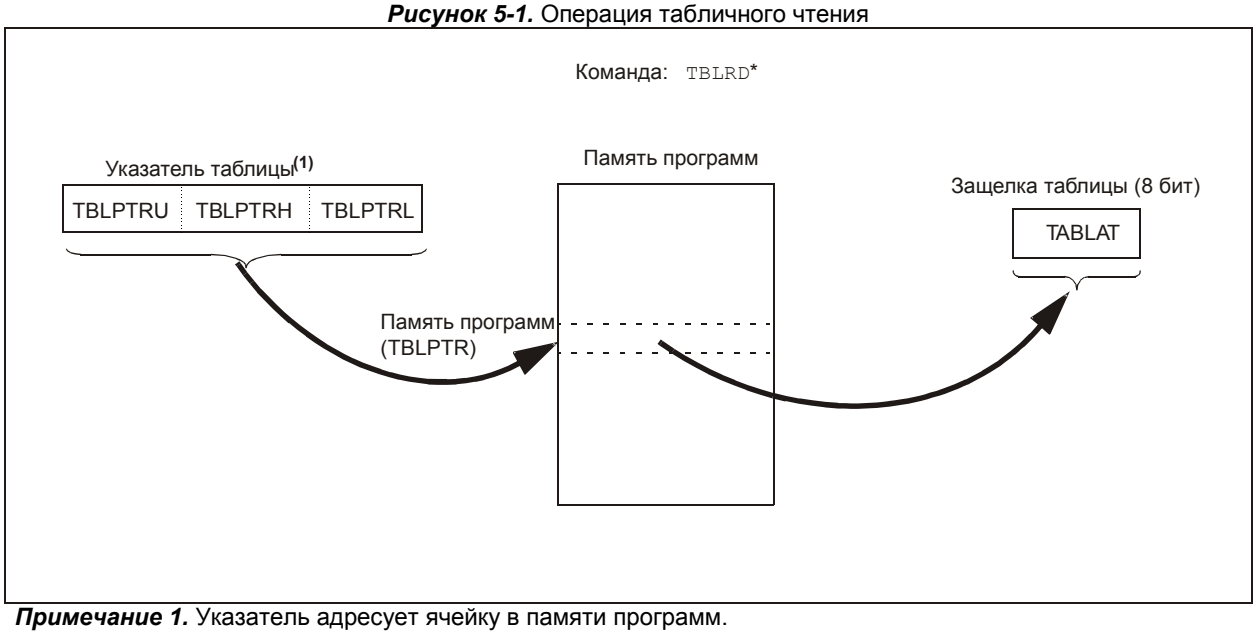
Память программ имеет 16-разрядную шину, в то время как память данных имеет 8-разрядную шину. Передача данных между памятью программ и памятью данных выполняется через 8-разрядный регистр TABLAT.

При табличном чтении данные выбираются из памяти программ и помещаются в память данных. На рисунке 5-1 показана операция табличного чтения из памяти программ.

При табличной записи информация из памяти данных переписываются в область памяти программ. Подробное описание операции записи смотрите в разделе 5.5. Операция записи информации из памяти данных в память программ показана на рисунке 5-2.

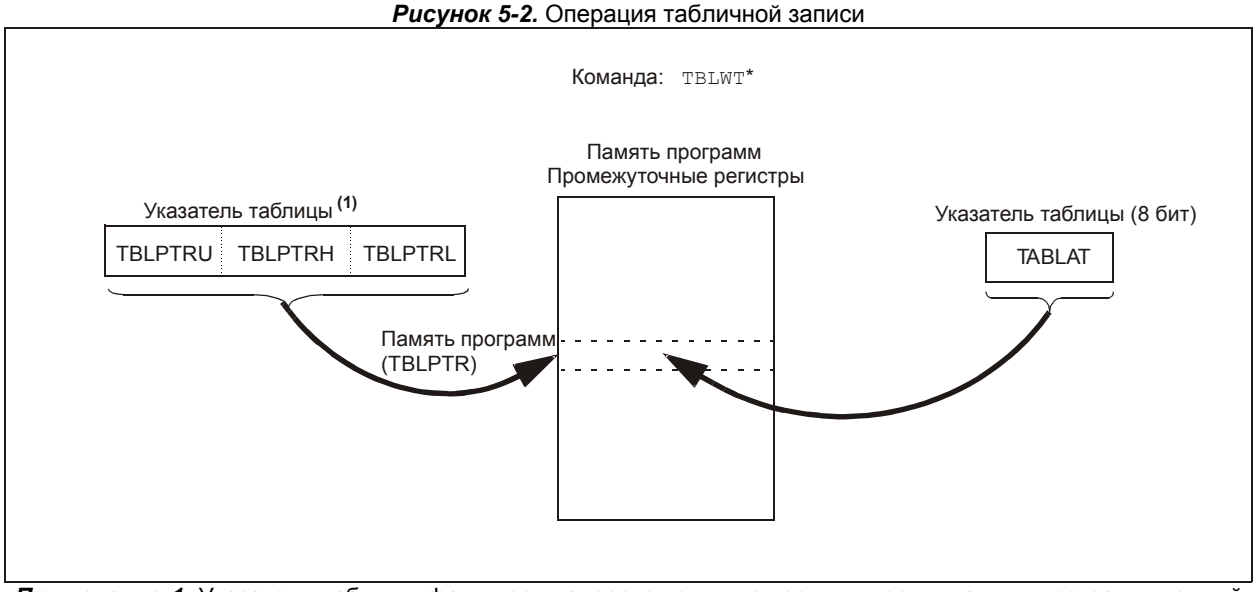
Табличное чтение и запись оперирует с байтами. Блоки таблиц чаще всего содержат данные, нежели коды команд, поэтому выравнивание слов не требуется. Блок таблицы может начинаться и заканчиваться в любом месте адресного пространства памяти программ. Если выполняется запись кодов программы, то необходимо делать выравнивание по границе слов команд.

Рисунок 5-1. Операция табличного чтения



Примечание 1. Указатель адресует ячейку в памяти программ.

Рисунок 5-2. Операция табличной записи



Примечание 1. Указатель таблицы фактически адресует одну из восьми ячеек в памяти программ, точный адрес которой определяется битами TBLPTRL<2:0>. Подробное описание процедуры записи данных во flash память программ смотрите в разделе 5.5.

5.2 Управляющие регистры

Совместно с командами TBLRD и TBLWT используется несколько регистров управления:

- EECON1
- EECON2
- TABLAT
- TBLPTR

5.2.1 Регистры EECON1 и EECON2

EECON1 – регистр управления для доступа к памяти.

EECON2 – не физический регистр. Чтение EECON2 будет давать результат 00h. Регистр EECON2 используется только с операциями записи и стирания (выполнение обязательной последовательности).

Выбор, к какому типу памяти выполняется обращение (к EEPROM памяти данных или Flash памяти программ), осуществляется битом EEPGD. Если EEPGD =1, то происходит обращение к памяти программ.

Управляющий бит CFGS определяет - будет происходить обращение к регистрам конфигурации/калибровки или к памяти программ/EEPROM памяти данных. Если CFGS =1, то все последующие операции будут относиться к регистрам конфигурации/калибровки независимо от состояния бита EEPGD (смотрите раздел 19). Когда бит CFGS=0, тип памяти, к которой выполняется обращение, определяется битом EEPGD.

Если бит FREE установлен в ‘1’, то выполняется операция стирания памяти программ при инициализации записи (бит WR). При сброшенном в ‘0’ бите FREE разрешена только запись.

Запись в память допускается только при установленном в ‘1’ бите WREN. При сбросе POR бит WREN=0. Бит WRERR устанавливается в ‘1’, когда операция записи прервана сбросом микроконтроллера -MCLR или переполнением WDT в нормальном режиме работы. В этом случае пользователь может проверить состояние бита WRERR и повторить запись (необходимо повторно загрузить данные в регистры EEDATA и EEADR).

С помощью битов RD и WR инициируется соответственно операция чтения и записи. Эти биты не могут быть сброшены программно, они сбрасываются аппаратно по завершению операции чтения или записи. Запрет сброса бита WR программным способом предотвращает случайное (преждевременное) завершение операции записи. Бит RD не может быть установлен в ‘1’ при обращении к памяти программ (EEPGD=1).

Примечание. Флаг прерывания EEIF в регистре PIR2 устанавливается в ‘1’ по завершении операции записи. Этот бит должен быть сброшен в ‘0’ программно.

Регистр 5-1. Регистр EECON1

R/W - x	R/W - x	U - 0	R/W - 0	R/W - x	R/W - 0	R/S - 0	R/S - 0
EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD
Бит 7							Бит 0

Бит 7 **EEPGD:** Обращение к Flash памяти программ или EEPROM памяти данных
1 = обращение к Flash памяти программ
0 = обращение к EEPROM памяти данных

Бит 6 **CFGS:** Обращение к Flash памяти программ/EEPROM памяти данных или к регистрам конфигурации
1 = обращение к регистрам конфигурации
0 = обращение к Flash памяти программ/EEPROM памяти данных

Бит 5 **Не используется:** Читается как ‘0’

Бит 4 **FREE:** Разрешение стирания Flash памяти программ
1 = стереть блок в памяти программ начиная с адреса TBLPTR при следующей команде WR (сбрасывается аппаратно при завершении операции стирания)
0 = только запись данных

Бит 3 **WRERR:** Флаг ошибки записи в память
1 = запись прервана (произошел один из сбросов во время выполнения записи)
0 = запись завершена

Примечание. При установке бита WRERR биты EEPGD, CFGS не сбрасываются, что позволяет определить условие ошибки.

Бит 2 **WREN:** Разрешение записи в память
1 = запись разрешена
0 = запись запрещена

Бит 1 **WR:** Управляющий бит записи
1 = инициализация цикла стирание/запись в EEPROM память данных. Для памяти программ инициализация цикла записи или стирания
(Бит сбрасывается аппаратно по завершении операции стирания/записи, программно он может быть только установлен в ‘1’)
0 = цикл стирание/запись завершен

Бит 0 **RD:** Управляющий бит чтения
1 = инициализация чтения EEPROM памяти данных
(Чтение выполняется за один цикл. Бит RD сбрасывается аппаратно, программно он может быть только установлен в '1’. RD не устанавливается в ‘1’, если EEPGD=1)
0 = чтение EEPROM памяти данных не инициализировалось

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

5.2.2 Регистр TABLAT

8-разрядный регистр TABLAT отображается на память данных в области регистров специального назначения. Этот регистр используется для передачи байта (8 бит) данных между памятью программ и памятью данных.

5.2.3 Указатель таблицы, регистр TBLPTR

С помощью указателя таблицы адресуется байт в области памяти программ. TBLPTR состоит из трех регистров специального назначения: верхний байт TBLPTRU; старший байт TBLPTRH; младший байт TBLPTRL. Эти регистры объединены для формирования 22-разрядного указателя в памяти программ. Младшие биты (21 бит) используются для адресации памяти программ (область до 2Мбайт). 22-й бит позволяет обращаться к ID микроконтроллера; ID пользователя и битам конфигурации.

Указатель таблицы TBLPTR используется командами TBLRD и TBLWT. Команды табличного чтения/записи способны изменять значение указателя TBLPTR одним из четырех способов (смотрите таблицу 5-1), они не распространяют свое действие на 22-й бит указателя.

5.2.4 Границы указателя таблицы

TBLPTR используется в операциях чтения, записи и стирания Flash памяти программ.

При выполнении команды TBLRD в адресации байта в памяти программ участвуют все 22 бита указателя. Байт с указанным адресом помещается в регистр TABLAT.

При выполнении команды TBLWT три младших бита указателя (TBLPTR<2:0>) определяют, какой из восьми байтов блока памяти программ записывается. Когда инициализируется запись в память программ (длинная запись) 19 старших битов (TBLPTR<21:3>) определяют, в какой блок памяти данных выполняется запись. Дополнительную информацию по записи данных во Flash память программ смотрите в разделе 5.5.

При выполнении стирания памяти программ используются 16 старших битов указателя (TBLPTR<21:6>) 64 байтного блока, а младшие биты (TBLPTR<5:0>) игнорируются.

На рисунке 5-3 отображены границы указателя TBLPTR при выполнении операций с Flash памятью программ.

Таблица 5-1. Изменение указателя таблицы командами TBLRD и TBLWT

Пример	Операция с указателем таблицы
TBLRD* TBLWT*	TBLPTR не изменяется
TBLRD*+ TBLWT*+	TBLPTR инкрементируется после чтения/записи
TBLRD*- TBLWT*-	TBLPTR декрементируется после чтения/записи
TBLRD+* TBLWT+*	TBLPTR инкрементируется перед чтением/записью

Рисунок 5-3. Границы указателя таблиц при различных операциях с памятью программ

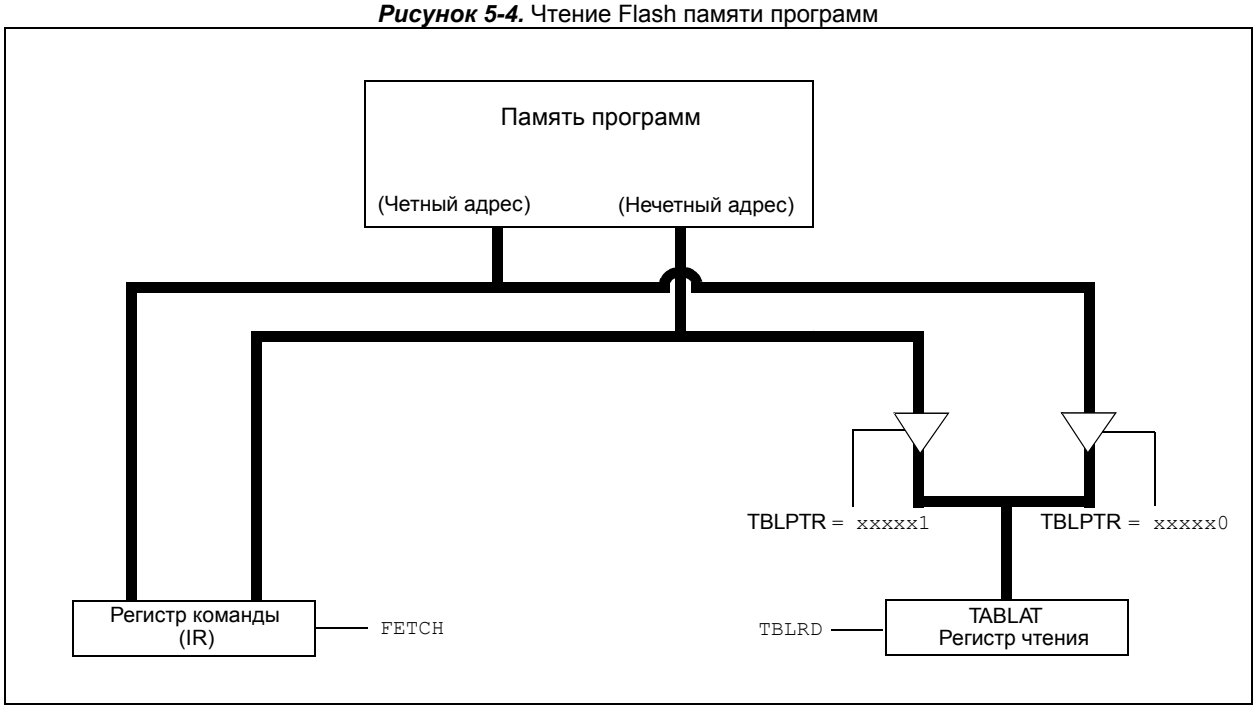
5.3 Чтение Flash памяти программ

Команда TBLRD используется для передачи байта из памяти программ в память данных. За одну операцию передается один байт данных.

В регистре TBLPTR содержится адрес ячейки в памяти программ. По команде TBLRD значение ячейки памяти программ передается в регистр TABLAT. Значение указателя TBLPTR может быть автоматически изменено для чтения следующего байта таблицы.

Внутренняя память программ имеет организацию хранения данных в виде слов. Младший бит адреса указывает, какую часть слова переписывать в регистр TABLAT (младшую или старшую). На рисунке 5-4 изображена связь памяти программ и регистра TABLAT.

Рисунок 5-4. Чтение Flash памяти программ



Пример 5-1. Чтение слова из Flash памяти программ

```
            MOVLW     CODE_ADDR_UPPER     ; Загрузка в TBLPTR
            MOVWF     TBLPTRU             ; адреса слова
            MOVLW     CODE_ADDR_HIGH
            MOVWF     TBLPTRH
            MOVLW     CODE_ADDR_LOW
            MOVWF     TBLPTRL
READ_WORD
            TBLRD*+                       ; Чтение TABLAT
            MOVFW     TABLAT              ; и инкремент указателя
            MOVWF     WORD_EVEN
            TBLRD*+                       ; Чтение TABLAT
            MOVFW     TABLAT              ; и инкремент указателя
            MOVWF     WORD_ODD
```

5.4 Стирание Flash памяти программ

Минимальный блок стираемой Flash памяти программ – 32 слова или 64 байта. Только с помощью внешнего программатора или через интерфейс ICSP можно стирать блоки большего объема. Стирание слова во Flash памяти программ не поддерживается.

При выполнении операции стирания командами микроконтроллера стирается блок Flash памяти программ в 64 байта. 16 старших битов указателя TBLPTR<21:6> используются для адресации блока, а младшие биты TBLPTR<5:0> игнорируются.

В регистре EECON1: бит EEPGD должен быть установлен в ‘1’ для выбора Flash памяти программ; бит WREN должен быть установлен в ‘1’ для разрешения операции записи; бит FREE должен быть установлен в ‘1’ для разрешения стирания блока Flash памяти программ.

Для защиты от случайного стирания необходимо выполнить обязательную последовательность действий с регистром EECON2.

Инициализация длинной записи необходима для стирания блока Flash памяти программ. Выполнение программы во время цикла стирания приостановлено. Завершение цикла стирания блока Flash памяти программ определяется внутренним таймером.

5.4.1 Последовательность действий для стирания Flash памяти программ

Рекомендованная последовательность действий для стирания блока Flash памяти программ:

1. Загрузить в указатель адрес стираемого блока
2. Установить бит EEPGD для выбора Flash памяти программ; установить бит WREN для разрешения записи; установить бит FREE для разрешения стирания
3. Выключить прерывания
4. Записать 55h в регистр EECON2
5. Записать AAh в регистр EECON2
6. Установить бит WR для инициализации цикла стирания
7. CPU остановит выполнение программы до завершения цикла стирания (ориентировочно 2мс)
8. Выполнить команду NOP
9. Разрешить прерывания

Пример 5-2. Стирание блока Flash памяти программ

```
            MOVLW   CODE_ADDR_UPPER          ; загрузить в TBLPTR
            MOVWF   TBLPTRU                  ; адрес стираемого блока
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
ERASE_ROW
            BSF     EECON1,EEPGD             ; выбрать Flash память программ
            BSF     EECON1,WREN              ; разрешить запись в память
            BSF     EECON1,FREE              ; разрешить операцию стирания
            BCF     INTCON,GIE               ; запретить прерывания
            MOVLW   55h                      ; Обязательная последовательность
            MOVWF   EECON2                   ; запись 55H
            MOVLW   AAh
            MOVWF   EECON2                   ; запись AAH
            BSF     EECON2,WR                ; старт стирания (CPU остановлен)
            NOP
            BSF     INTCON,GIE               ; re-enable interrupts
```

5.5 Запись во Flash память программ

Минимальный программируемый блок – 4 слова или 8 байт. Программирование отдельного слова или байта не поддерживается.

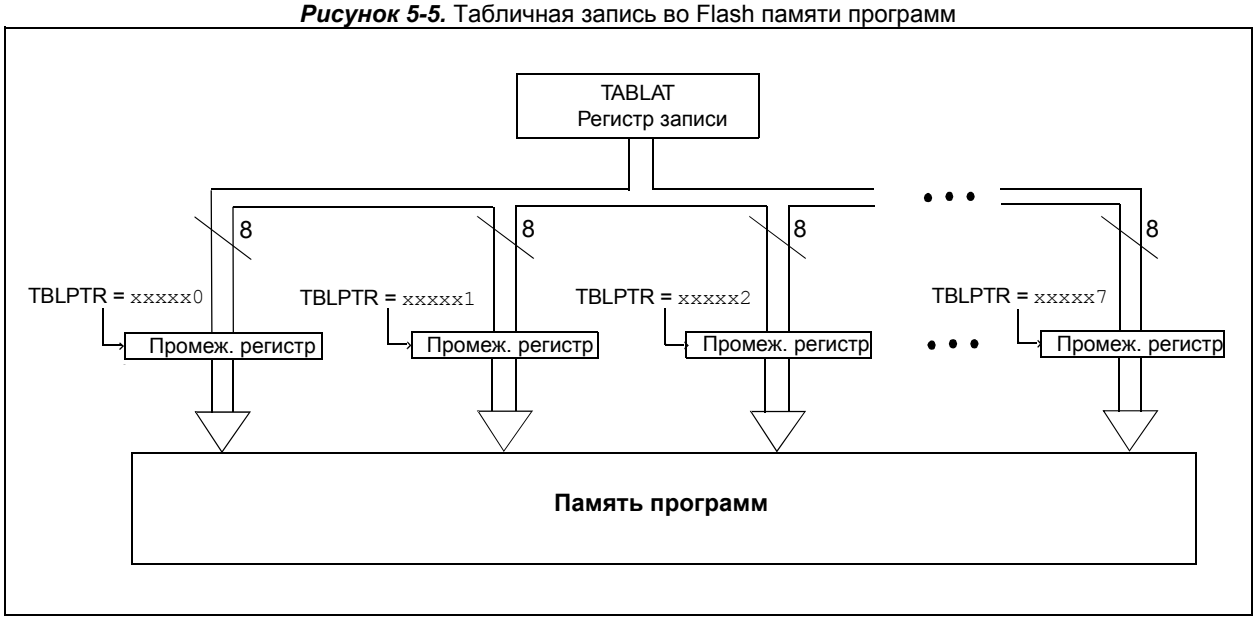
В табличной записи используется 8 промежуточных регистров, в которых храниться подготовленные данные для записи во Flash Память программ.

Регистр TABLAT может сохранить только один байт, поэтому команда TBLWT должна быть выполнена 8 раз для каждого записываемого байта в отдельности. Все команды TBLWT будут выполнены быстро, поскольку данные передаются в промежуточные регистры. После записи 8-го байта регистр EECON1 должен быть настроен для выполнения длинной записи.

Длинная запись необходима для программирования блока во Flash память программ. Во время записи выполнение команд приостановлено, пока не завершится цикл записи. Завершение цикла записи блока Flash памяти программ определяется внутренним таймером.

Интегрированный таймер записи в EEPROM память данных управляет циклом записи во Flash память программ. Напряжения, необходимые для циклов стирания/записи, генерируются внутренним источником, чтобы нормально выполнять операции во всем диапазоне допустимых напряжений питания V_{DD}.

Рисунок 5-5. Табличная запись во Flash памяти программ



5.5.1 Последовательность записи во Flash память программ

Рекомендованная последовательность действий для записи блока Flash памяти программ:

1. Прочитать 64 байта блока в память данных
2. Обновить необходимые регистры с информацией из памяти программ
3. Загрузить в указатель адрес стираемого блока Flash памяти программ
4. Выполнить процедуру стирания
5. Загрузить в указатель адрес первого байта
6. Записать первые 8 байт с автоинкрементом указателя в промежуточные регистры
7. Установить бит EEPGD для выбора Flash памяти программ; установить бит WREN для разрешения записи
8. Выключить прерывания
9. Записать 55h в регистр EECON2
10. Записать AAh в регистр EECON2
11. Установить бит WR для инициализации цикла записи блока
12. CPU остановит выполнение программы до завершения цикла записи (ориентировочно 2мс)
13. Выполнить команду NOP
14. Разрешить прерывания
15. Повторить шаги 6-14 семь раз для записи 64 байт
16. Выполнить контрольное чтение

Процедура записи будет занимать ориентировочно 18мс, т.к. необходимо записать 64 байта. Исходный текст программы записи во Flash память программ показан в примере 5-3.

Пример 5-3. Запись блока во Flash память программ

```
                MOVLW   D'64                    ; число байт в стираемом блоке
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; указатель буфера
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   CODE_ADDR_UPPER         ; загрузка в TBLPTR адреса
                MOVWF   TBLPTRU                 ; блока памяти
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
READ_BLOCK
                TBLRD*+                         ; чтение TABLAT
                MOVFW   TABLAT                  ; и инкремент указателя
                MOVWF   POSTINC0                ; сохранение данных
                DECFSZ  COUNTER                 ; завершено?
                GOTO    READ_BLOCK              ; повтор
MODIFY_WORD
                MOVLW   DATA_ADDR_HIGH          ; указатель буфера
                MOVWF   FSR0H
                MOVLW   DATA_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   NEW_DATA_LOW            ; обновление слова в буфере
                MOVWF   POSTINC0
                MOVLW   NEW_DATA_HIGH
                MOVWF   INDF0
ERASE_BLOCK
                MOVLW   CODE_ADDR_UPPER         ; загрузка в TBLPTR адреса
                MOVWF   TBLPTRU                 ; блока памяти
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
                BSF     EECON1,EEPGD            ; выбрать Flash память программ
                BSF     EECON1,WREN             ; разрешить запись в память
                BSF     EECON1,FREE             ; разрешить операцию стирания
                BCF     INTCON,GIE              ; запретить прерывания
                MOVLW   55h
                MOVWF   EECON2                  ; запись 55H
                MOVLW   AAh
                MOVWF   EECON2                  ; запись AAH
                BSF     EECON1,WR               ; старт стирания (CPU остановлен)
                NOP
                BSF     INTCON,GIE              ; разрешить прерывания
                TBLRD*-                         ; пустое чтение с декрементом
WRITE_BUFFER_BACK
                MOVLW   8                       ; количество записей по 8 байт
                MOVWF   COUNTER_HI
                MOVLW   BUFFER_ADDR_HIGH        ; указатель буфера
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
PROGRAM_LOOP
                MOVLW   8                       ; число записываемых байт в промежуточные
                                                ; регистры
                MOVWF   COUNTER
                WRITE_  WORD_TO_HREGS
                MOVFW   POSTINC0                ; байт из буфера
                MOVWF   TABLAT                  ; поместить в защелку таблицы
                TBLWT+*                         ; короткая запись данных во
                                                ; внутренние промежуточные регистры.
                DECFSZ  COUNTER                 ; буфер из промежуточных регистров полон
                GOTO    WRITE_WORD_TO_HREGS
```

Пример 5-3. Запись блока во Flash память программ (продолжение)

```
PROGRAM_MEMORY
              BSF     EECON1,EEPGD            ; выбрать Flash память программ
              BSF     EECON1,WREN             ; разрешить запись в память
              BCF     INTCON,GIE              ; запретить прерывания
              MOVLW   55h
              MOVWF   EECON2                  ; запись 55H
              MOVLW   AAh
              MOVWF   EECON2                  ; запись AAH
              BSF     EECON1,WR               ; старт программирования (CPU остановлен)
              NOP
              BSF     INTCON,GIE              ; разрешить прерывания
              DECFSZ  COUNTER_HI              ; цикл завершен
              GOTO    PROGRAM_LOOP
              BCF     EECON1,WREN             ; запретить запись в память
```

5.5.2 Проверка записи

Хорошим тоном программирования считается проверка записанных данных. Особенно проверка записанных данных должна выполняться в приложениях, в которых допускается исчерпание гарантированного числа циклов стирание/запись.

5.5.3 Выносливость ячеек памяти программ

Приложения, в которых допускается превышение 10% использования гарантированного числа циклов стирание/запись (параметры D120, D120A), число обновлений для каждой ячейки должно быть не более 1/10 указанных значений. Дополнительную информацию смотрите в документе AN790 (DS00790).

5.5.4 Неожиданное завершение операции записи

При неожиданном завершении операции записи (выключение питания, сброс микроконтроллера), программируемая область должна быть проверена и при необходимости перепрограммирована. Бит WRERR устанавливается в ‘1’, если произошел сброс -MCLR или переполнение WDT в нормальном режиме при выполнении цикла записи. В этом случае пользователь может проверить состояние бита WRERR и повторить запись.

5.5.5 Защита от случайной записи

Для защиты от случайной записи во Flash память программ предусмотрена обязательная последовательность действий. Дополнительную информацию смотрите в разделе 19.

5.6 Операции с Flash памятью программ при включенной защите кода

Детальное описание защиты кода программы смотрите в разделе 19.

Таблица 5-2. Регистры и биты, связанные с Flash памятью программ

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF8h	TBLPTRU	-	-	Бит 21(2)	Указ. табл. памяти прогр. верхний байт (TBLPTR<20:16>)					--00 0000
FF7h	TBLPTRH	Указатель таблицы памяти программ старший байт (TBLPTR<15:8>)								0000 0000
FF6h	TBLPTRL	Указатель таблицы памяти программ младший байт (TBLPTR<7:0>)								0000 0000
FF5h	TABLAT	Защелка таблицы памяти программ								0000 0000
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FA7h	EECON2	Управляющий регистр 2 EEPROM памяти (нефизический регистр)								---- ----
FA6h	EECON1	EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD	xx-0 x000
FA2h	IPR2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
FA1h	PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
FA0h	PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

6. EEPROM память данных

EEPROM память данных доступна для записи/чтения в нормальном режиме работы микроконтроллера во всем диапазоне рабочего напряжения питания V_{DD}. EEPROM память данных не отображается на адресное пространство памяти данных, а доступна через регистры специального назначения.

Для косвенного доступа к EEPROM памяти данных используются 4 регистра специального назначения:

- EECON1
- EECON2
- EEDATA
- EEADR

Чтение и запись EEPROM памяти выполняется по байтно. В регистре EEDATA сохраняются 8-разрядные данные записи/чтения, а регистр EEADR содержит адрес ячейки EEPROM памяти данных. С помощью 8 - разрядного регистра EEADR можно адресовать 256 байт EEPROM памяти данных (диапазон адресов 00h-FFh).

EEPROM память позволяет выполнить циклы чтения и записи байта данных. При записи байта происходит автоматическое стирание ячейки и запись новых данных (стирание перед записью). Время записи управляется интегрированным таймером и зависит от напряжения питания, температуры и технологического разброса параметров кристалла (смотрите параметр D122 в разделе 22).

6.1 Регистр EEADR

Регистр адреса ячейки в EEPROM памяти данных, с помощью которого можно адресовать 256 байт (максимум).

6.2 Регистры EECON1, EECON2

Регистр EECON1 содержит биты управления EEPROM памяти данных.

Регистр EECON2 не реализован физически, читается как 00h. Он используется в операциях записи в EEPROM память данных для реализации обязательной последовательности команд.

Управляющие биты RD и WR инициализируют соответственно чтение и запись данных. Программно эти биты могут быть только установленные в ‘1’, сброс в ‘0’ происходит аппаратно по завершению операции чтения/записи. Защита от программного сброса бита WR позволяет предотвратить преждевременное завершение операции записи.

Если бит WREN=1, то разрешена запись в EEPROM память данных. После сброса по включению питания (POR) бит WREN равен ‘0’. Бит WRERR устанавливается в ‘1’, если во время выполнения записи в EEPROM память данных произошел сброс по сигналу -MCLR или по переполнению сторожевого таймера WDT в нормальном режиме. Проверив состояние бита WREER пользователь может повторить запись (регистры EEDATA и EEADR не изменяют своего значения).

Примечание. После завершения записи в EEPROM память данных устанавливается флаг EEIF в регистре PIR2. Бит EEIF должен быть сброшен в ‘0’ программно.

Регистр 6-1. Регистр EECON1

R/W - x	R/W - x	U - 0	R/W - 0	R/W - x	R/W - 0	R/S - 0	R/S - 0
EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD
Бит 7							Бит 0

Бит 7 **EEPGD:** Обращение к Flash памяти программ или EEPROM памяти данных
1 = обращение к Flash памяти программ
0 = обращение к EEPROM памяти данных

Бит 6 **CFGS:** Обращение к Flash памяти программ/EEPROM памяти данных или к регистрам конфигурации
1 = обращение к регистрам конфигурации
0 = обращение к Flash памяти программ/EEPROM памяти данных

Бит 5 **Не используется:** Читается как ‘0’

Бит 4 **FREE:** Разрешение стирания Flash памяти программ
1 = стереть блок в памяти программ начиная с адреса TBLPTR при следующей команде WR (сбрасывается аппаратно при окончании операции стирания)
0 = только запись данных

Бит 3 **WRERR:** Флаг ошибки записи в память
1 = запись прервана (произошел один из сбросов во время выполнения записи)
0 = запись завершена

Примечание. При установке бита WRERR биты EEPGD, CFGS не сбрасывается, что позволяет определить условие ошибки.

Бит 2 **WREN:** Разрешение записи в память
1 = запись разрешена
0 = запись запрещена

Бит 1 **WR:** Управляющий бит записи
1 = инициализация цикла стирание/запись в EEPROM память данных. Для памяти программ инициализация цикла записи или стирания
(Бит сбрасывается аппаратно по завершении операции стирания/записи, программно он может быть только установлен в ‘1’)
0 = цикл стирание/запись завершен

Бит 0 **RD:** Управляющий бит чтения
1 = инициализация чтения EEPROM памяти данных
(Чтение выполняется за один цикл. Бит RD сбрасывается аппаратно, программно он может быть только установлен в '1’. RD не устанавливается в ‘1’, если EEPGD=1)
0 = чтение EEPROM памяти данных не инициализировалось

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

6.3 Чтение из EEPROM памяти данных

Для чтения EEPROM памяти данных необходимо записать адрес в регистр EEADR, сбросить бит EEPGD (EECON1<7>) и установить бит RD (EECON1<0>) в ‘1’. В следующем машинном цикле данные доступны для чтения из регистра EEDATA. Прочитанное значение из EEPROM памяти данных будет храниться в регистре EEDATA до следующего чтения или записи в этот регистр по команде микроконтроллера.

Пример 6-1 Чтение из EEPROM памяти данных

```
MOVLW    CONFIG_ADDR        ;
MOVWF    EEADR              ; Адрес считываемого регистра
BCF      EECON1, EEPGD      ; Выбрать EEPROM память данных
BSF      EECON1, RD         ; Чтение
MOVF     EEDATA, W          ; W = EEDATA
```

6.4 Запись в EEPROM память данных

Для записи в EEPROM память данных необходимо записать адрес в регистр EEADR, данные в регистр EEDATA и выполнить обязательную последовательность команд, показанных в примере 6-2.

Запись байта не будет произведена, если не выполнена указанная последовательность (запись 55h в EECON2, запись AAh в EECON2, установка бита WR в ‘1’ для каждого байта). Рекомендуется запрещать прерывания при выполнении обязательной последовательности команд. Если во время выполнения указанной последовательности произойдет переход по вектору прерывания, запись байта выполнена не будет.

Чтобы разрешить запись в EEPROM память данных, необходимо установить бит WREN (EECON1<2>) в ‘1’, защищающий от случайной записи. Пользователь должен установить бит WREN в ‘1’ перед началом записи, а после окончания записи сбросить его в ‘0’ (аппаратно бит WREN в ‘0’ не сбрасывается).

После инициализации записи значения регистров EECON1, EEADR и EDATA не может быть изменено. Установка бита WR заблокирована, если бит WREN=0. Бит WR не может быть установлен в ‘1’ при одновременной установки бита WREN (одной командой), бит WREN должен быть предварительно установлен.

По окончанию записи бит WR аппаратно сбрасывается в ‘0’, а флаг прерывания EEIF устанавливается в ‘1’. Пользователь может использовать прерывания для проверки окончания записи в EEPROM память данных. Флаг EEIF сбрасывается в ’0’ программно.

Пример 6-2. Запись в EEPROM память данных

```
MOVLW   DATA_EE_ADDR       ;
MOVWF   EEADR              ; адрес записываемой ячейки
MOVLW   DATA_EE_DATA       ;
MOVWF   EEDATA             ; записываемые данные
BCF     EECON1, EEPGD      ; операция с EEPROM памятью
BSF     EECON1, WREN       ; разрешить запись
BCF     INTCON,GIE         ; запретить прерывания
MOVLW   55h                ; Обязательная последовательность
MOVWF   EECON2             ; запись 55H
MOVLW   AAh
MOVWF   EECON2             ; запись AAH
BSF     EECON1,WR          ; установить бит WR для начала записи
BSF     INTCON, GIE        ; разрешить прерывания
SLEEP                      ; ожидать прерывания завершения цикла записи
BCF     EECON1, WREN       ; запретить запись
```

6.5 Проверка записи

Рекомендуется после выполнения операции записи в EEPROM память данных произвести контрольное чтение. Выполнять контрольное чтение особенно рекомендуется, если возможно исчерпание гарантированных циклов стирание/запись.

6.5.1 Выносливость ячеек EEPROM памяти данных

Приложения, в которых допускается превышение 10% использования гарантированного числа циклов стирание/запись (параметры D130, D130A), число обновлений для каждой ячейки должно быть не более 1/10 указанных значений. Дополнительную информацию смотрите в документе AN790 (DS00790).

6.6 Защита от случайной записи

Существует несколько условий, когда запись байта в EEPROM память данных не выполняется:

1. После сброса по включению питания POR бит WREN = 0.
2. Таймер включения питания (в течение 72мс) запрещает запись в EEPROM память данных.
3. Обязательная последовательность инициализации записи и бит WREN предотвращают случайную запись.

Все эти меры предотвращают случайную запись в EEPROM память данных при сбое программы, снижении напряжения питания и других ненормальных режимах работы микроконтроллера.

6.7 Операции с EEPROM памятью при включенной защите кода программы

EEPROM память данных имеет собственный механизм защиты. При включенной защите запрещена запись/чтение EEPROM памяти внешними устройствами (программаторами). Программа пользователя может нормально читать/записывать данные EEPROM память вне зависимости от состояния бита защиты в регистрах конфигурации. Дополнительную информацию смотрите в разделе 19.

Таблица 6-1. Регистры и биты, связанные с EEPROM памятью данных

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FA9h	EEADR	Регистр адреса EEPROM памяти								0000 0000
FA8h	EEDATA	Регистр данных EEPROM памяти								0000 0000
FA7h	EECON2	Управляющий регистр 2 EEPROM памяти (нефизический регистр)								---- ----
FA6h	EECON1	EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD	xx-0 x000
FA2h	IPR2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
FA1h	PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
FA0h	PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

7. Аппаратное умножение 8x8

7.1 Введение

АЛУ микроконтроллеров PIC18FXX2 содержит модуль аппаратного умножения 8x8. Операция умножения выполняется за один машинный цикл. Результатом является беззнаковое 16-разрядное число, которое сохраняется в спаренном регистре PRODH:PRODL. Умножение не изменяет состояние флагов регистра STATUS.

Использование аппаратного умножения 8x8 дает следующие преимущества:

- Более высока вычислительная мощность
- Уменьшение кода программы на алгоритмы умножения

Увеличение вычислительной мощности позволяет использовать микроконтроллеры PIC18FXX2 в приложениях, в которых применяются DSP.

В таблице 7-1 представлено сравнение выполнения некоторых функций при использовании аппаратного умножения и реализации умножения программным способом.

Таблица 7-1. Сравнение операций умножения

Умножение	Метод умножения	Память программ (слов)	Циклов (макс.)	Длительность		
				@ 40МГц	@ 10МГц	@ 4МГц
8 x 8 unsigned	Без аппаратного умножения	13	69	6.9мкс	27.6мкс	69мкс
	С аппаратным умножением	1	1	100нс	400нс	1мкс
8 x 8 signed	Без аппаратного умножения	33	91	9.1мкс	36.4мкс	91мкс
	С аппаратным умножением	6	6	600нс	2.4мкс	6мкс
16 x 16 unsigned	Без аппаратного умножения	21	242	24.2мкс	96.8мкс	242мкс
	С аппаратным умножением	24	24	2.4мкс	9.6мкс	24мкс
16 x 16 signed	Без аппаратного умножения	52	254	25.4мкс	102.6мкс	254мкс
	С аппаратным умножением	36	36	3.6мкс	14.4мкс	36мкс

7.2 Операции умножения

В примере 7-1 показана последовательность действий для выполнения беззнакового умножения 8x8. Для этой операции необходимо только одна команда микроконтроллера, если один из параметров уже загружен в WREG.

В примере 7-2 показана последовательность команд для выполнения знакового умножения 8x8. Чтобы получить знак результата, необходимо проверить старший бит каждого байта.

Пример 7-1. Последовательность команд для выполнения беззнакового умножения 8x8

```
MOVF    ARG1, W         ;
MULWF   ARG2            ; ARG1 * ARG2 ->
                        ; PRODH:PRODL
```

Пример 7-2. Последовательность команд для выполнения знакового умножения 8x8

```
MOVF    ARG1, W
MULWF   ARG2            ; ARG1 * ARG2 ->
                        ; PRODH:PRODL
BTFSC   ARG2, SB        ; Проверка знакового бита
SUBWF   PRODH, F        ; PRODH = PRODH
                        ; - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB        ; Проверка знакового бита
SUBWF   PRODH, F        ; PRODH = PRODH
                        ; - ARG2
```

Операция беззнакового умножения 16x16 представлена в примере 7-3. В уравнении 7-1 показан алгоритм вычислений. 32-разрядный результат сохраняется в четырех регистрах RES3:RES0.

Уравнение 7-1. Алгоритм беззнакового умножения 16x16

$$
\begin{aligned}
RES3{:}RES0 &= ARG1H{:}ARG1L \times ARG2H{:}ARG2L \\
&= (ARG1H \times ARG2H \times 2^{16}) + \\
&\quad (ARG1H \times ARG2L \times 2^{8}) + \\
&\quad (ARG1L \times ARG2H \times 2^{8}) + \\
&\quad (ARG1L \times ARG2L)
\end{aligned}
$$

Пример 7-3. Последовательность команд для выполнения беззнакового умножения 16x16

```
MOVF    ARG1L, W
MULWF   ARG2L          ; ARG1L * ARG2L ->
                       ; PRODH:PRODL
MOVFF   PRODH, RES1    ;
MOVFF   PRODL, RES0    ;
                       ;
MOVF    ARG1H, W
MULWF   ARG2H          ; ARG1H * ARG2H ->
                       ; PRODH:PRODL
MOVFF   PRODH, RES3    ;
MOVFF   PRODL, RES2    ;
                       ;
MOVF    ARG1L, W
MULWF   ARG2H          ; ARG1L * ARG2H ->
                       ; PRODH:PRODL
MOVF    PRODL, W       ;
ADDWF   RES1, F        ; Прибавить к
MOVF    PRODH, W       ; результату
ADDWFC  RES2, F        ;
CLRF    WREG           ;
ADDWFC  RES3, F        ;
                       ;
MOVF    ARG1H, W       ;
MULWF   ARG2L          ; ARG1H * ARG2L ->
                       ; PRODH:PRODL
MOVF    PRODL, W       ;
ADDWF   RES1, F        ; Прибавить к
MOVF    PRODH, W       ; результату
ADDWFC  RES2, F        ;
CLRF    WREG           ;
ADDWFC  RES3, F        ;
```

Последовательность команд для операции знакового умножения 16x16 показана в примере 7-4. В уравнении 7-1 смотрите алгоритм вычислений. 32-разрядный результат сохраняется в четырех регистрах RES3:RES0. Чтобы получить знак результата, необходимо проверить старший бит каждого 16-разрядного слова.

Уравнение 7-2. Алгоритм знакового умножения 16x16

```
RES3:RES0
  =     ARG1H:ARG1L x ARG2H:ARG2L
  =     (ARG1H x ARG2H x 2 16 )+
        (ARG1H x ARG2L x 2 8 )+
        (ARG1L x ARG2H x 2 8 )+
        (ARG1L x ARG2L)+
        (-1 x ARG2H<7> x ARG1H:ARG1L x 2 16 )+
        (-1 x ARG1H<7> x ARG2H:ARG2L x 2 16 )
```

Пример 7-4. Последовательность команд для выполнения знакового умножения 16x16

```
      MOVF     ARG1L, W
      MULWF    ARG2L          ; ARG1L * ARG2L ->
                              ; PRODH:PRODL
      MOVFF    PRODH, RES1    ;
      MOVFF    PRODL, RES0    ;
                              ;
      MOVF     ARG1H, W
      MULWF    ARG2H          ; ARG1H * ARG2H ->
                              ; PRODH:PRODL
      MOVFF    PRODH, RES3    ;
      MOVFF    PRODL, RES2    ;
                              ;
      MOVF     ARG1L, W
      MULWF    ARG2H          ; ARG1L * ARG2H ->
                              ; PRODH:PRODL
      MOVF     PRODL, W       ;
      ADDWF    RES1, F        ; Прибавить к
      MOVF     PRODH, W       ; результату
      ADDWFC   RES2, F        ;
      CLRF     WREG           ;
      ADDWFC   RES3, F        ;
                              ;
      MOVF     ARG1H, W       ;
      MULWF    ARG2L          ; ARG1H * ARG2L ->
                              ; PRODH:PRODL
      MOVF     PRODL, W       ;
      ADDWF    RES1, F        ; Прибавить к
      MOVF     PRODH, W       ; результату
      ADDWFC   RES2, F        ;
      CLRF     WREG           ;
      ADDWFC   RES3, F        ;
                              ;
      BTFSS    ARG2H, 7       ; ARG2H:ARG2L отрицательный?
      BRA      SIGN_ARG1      ; нет, проверка ARG1
      MOVF     ARG1L, W       ;
      SUBWF    RES2           ;
      MOVF     ARG1H, W       ;
      SUBWFB   RES3
                              ;
SIGN_ARG1
      BTFSS    ARG1H, 7       ; ARG1H:ARG1L отрицательный?
      BRA      CONT_CODE      ; нет, завершить
      MOVF     ARG2L, W       ;
      SUBWF    RES2           ;
      MOVF     ARG2H, W       ;
      SUBWFB   RES3
                              ;
CONT_CODE
      :
```

8. Прерывания

Микроконтроллеры PIC18FXX2 имеют несколько источников прерываний и функцию приоритетной системы прерываний, которая позволяет для каждого источника прерываний назначить высокий или низкий приоритет. При возникновении прерывания с высоким приоритетом происходит переход по вектору 000008h, а при возникновении прерывания с низким приоритетом – 000018h. Прерывание с высоким приоритетом приостанавливают обработку прерываний с низким приоритетом.

В PIC18FXX2 предусмотрено 10 регистров специального назначения для управления прерываниями:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2
- PIE1, PIE2
- IPR1, IPR2

Рекомендуется использовать файлы *.inc, которые входят в состав MPLAB-IDE, для символьного обозначения регистров. Это позволяет ассемблеру/компилятору автоматически корректировать расположение управляющих битов в регистрах специального назначения.

Каждому источнику прерываний соответствует три управляющих бита:

- Флаг прерываний, указывает на то, что выполнено условие возникновения прерывания
- Бит разрешения прерывания, разрешает переход по вектору прерывания при установке соответствующего флага
- Бит приоритета, выбор низкого или высокого приоритета прерывания

Приоритетная система прерываний включена, если бит IPEN(RCON<7>) установлен в ‘1’. Для приоритетной системы прерываний предусмотрено два бита глобального разрешения прерываний. Установка в ‘1’ бита GIEH(INTCON<7>) разрешает прерывания с высоким приоритетом (бит приоритета этих прерываний должен быть установлен). Если бит GIEL(INTCON<6>) установлен в ‘1’, то разрешены все прерывания с низким приоритетом (бит приоритета этих прерываний должен быть сброшен). Когда флаг разрешенного прерывания установлен в ‘1’ и разрешены прерывания соответствующего приоритета, происходит переход по вектору 000008h или 000018h в зависимости от приоритетности прерывания. Отдельные прерывания могут быть запрещены сбросом соответствующего бита разрешения.

Когда бит IPEN=0 (состояние по умолчанию), приоритетная система прерываний выключена (система прерываний совместима с микроконтроллерами PICmicro среднего семейства). В этом режиме биты приоритета прерываний не имеют никакого значения. INTCON<6> - PEIE, разрешает/запрещает все периферийные прерывания. INTCON<7> - GIE, бит глобального разрешения прерываний. При возникновении прерывания всегда происходит переход по вектору 000008h.

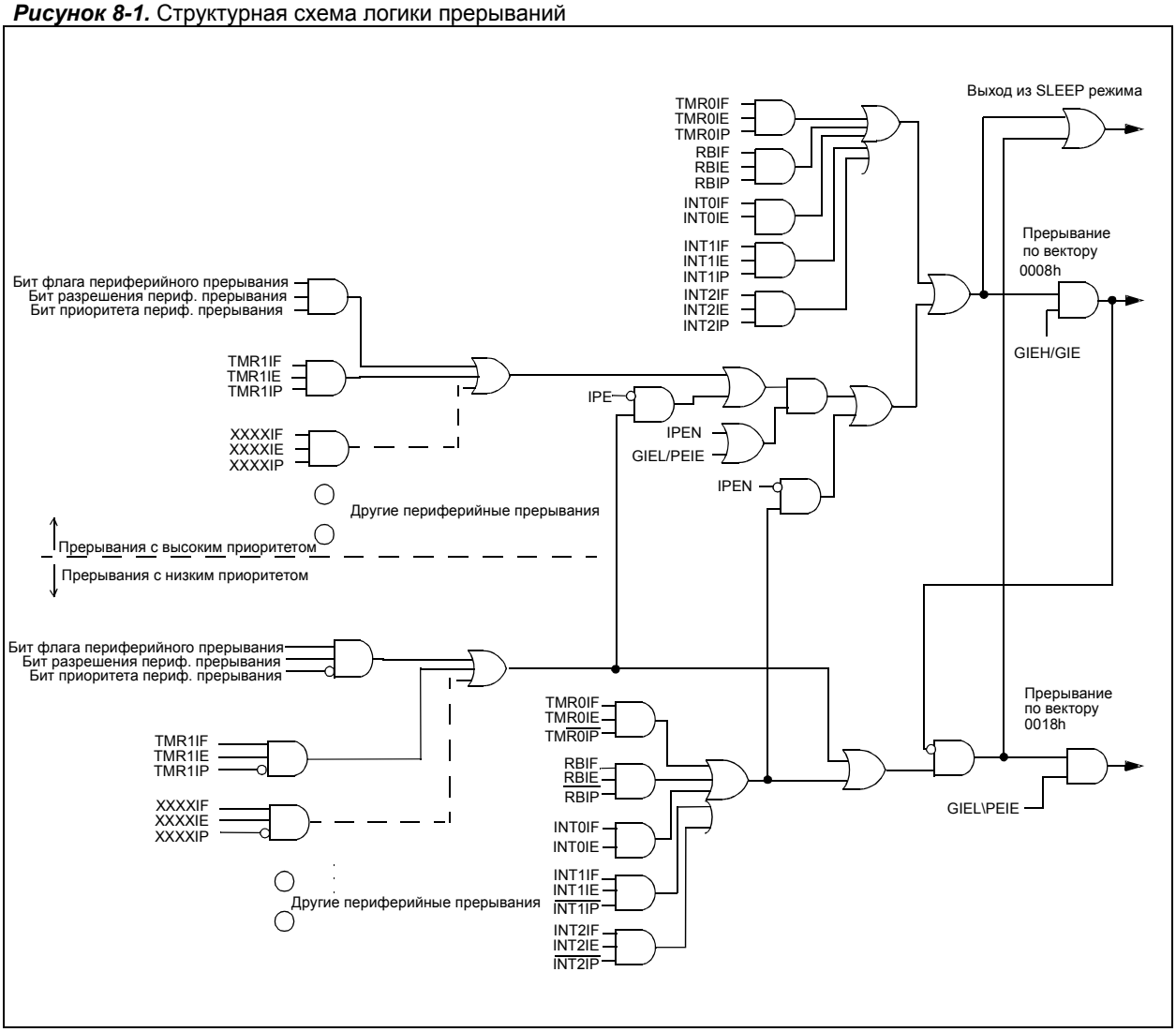
При переходе на обработку прерываний бит глобального разрешения прерываний сбрасывается, чтобы запретить прерывания соответствующего приоритета. Если бит IPEN=0, то это бит GIE. Если приоритетная система прерываний включена, то это один из битов GIEH или GIEL. Прерывания с высоким приоритетом могут приостанавливать обработку прерываний с низким приоритетом.

Адрес возврата помещается в вершину стека, а в счетчик команд PC помещается вектор прерываний (000008h или 000018h). В обработчике прерываний конкретный источник прерываний может быть определен проверкой соответствующих флагов. Флаги прерываний должны быть сброшены в обработчике прерываний для предотвращения повторного перехода на обработку прерывания.

Выход из обработки прерываний необходимо выполнять командой RETFIE, по которой будет установлен соответствующих бит глобального разрешения прерываний (GIE, GIEH или GIEL).

Время перехода на обработку прерываний от внешних источников (прерывания INT, изменение уровня сигнала на входах PORTB и др.) составляет три-четыре цикла команд. Время перехода не зависит от типа выполняемой команды (однословная или двухсловная). Флаги прерываний устанавливаются вне зависимости от состояния битов глобального и индивидуального разрешения прерываний.

Рисунок 8-1. Структурная схема логики прерываний



8.1 Регистры INTCON

Регистры INTCON доступны для записи и чтения, они содержат биты разрешения прерываний, флаги прерываний и биты приоритета.

Примечание. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита общего разрешения прерываний. Это позволяет выполнять программный контроль возникновения условия прерываний. Необходимо заботиться о том, чтобы флаг прерывания был сброшен перед разрешением прерывания.

Регистр 8-1. Регистр INTCON

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
Бит 7							Бит 0

Бит 7 **GIE/GIEH:** Бит глобального разрешения прерываний
IPEN=0
1 = разрешены все немаскированные прерывания
0 = все прерывания запрещены

IPEN=1
1 = разрешены прерывания с высоким приоритетом
0 = все прерывания с высоким приоритетом запрещены

Бит 6 **PEIE/GIEL:** Разрешение периферийных прерываний
IPEN=0
1 = разрешены все периферийные немаскированные прерывания
0 = все периферийные прерывания запрещены

IPEN=1
1 = разрешены прерывания с низким приоритетом
0 = все прерывания с низким приоритетом запрещены

Бит 5 **TMR0IE:** Разрешение прерывания по переполнению TMR0
1 = разрешено прерывание по переполнению TMR0
0 = прерывание по переполнению TMR0 запрещено

Бит 4 **INT0IE:** Разрешение внешнего прерывания INT0
1 = внешнее прерывание INT0 разрешено
0 = внешнее прерывание INT0 запрещено

Бит 3 **RBIE:** Разрешение прерывания по изменению уровня сигнала на входах PORTB
1 = разрешено прерывание по изменению уровня сигнала на входах PORTB
0 = запрещено прерывание по изменению уровня сигнала на входах PORTB

Бит 2 **TMR0IF:** Флаг прерывания переполнения таймера TMR0
1 = произошло переполнение таймера TMR0 (сбрасывается программно)
0 = переполнение таймера TMR0 не происходило

Бит 1 **INT0IF:** Флаг внешнего прерывания INT0
1 = выполнено условие внешнего прерывания INT0 (сбрасывается программно)
0 = условие внешнего прерывания INT0 не выполнено

Бит 0 **RBIF:** Флаг прерывания по изменению уровня сигнала на входах PORTB
1 = зафиксировано изменение уровня сигнала на одном из входов RB7:RB4 (сбрасывается программно)
0 = уровень сигнала на входах RB7:RB4 не изменялся

Примечание. Несоответствие входного сигнала и сохраненного значения будет устанавливать флаг RBIF в ‘1’. Чтение регистра PORTB снимет условие несоответствия и позволит сбросить флаг RBIF.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 8-2. Регистр INTCON2

R/W - 1	R/W - 1	R/W - 1	R/W - 1	U - 0	R/W - 1	U - 0	R/W - 1
-RBPU	INTEDG0	INTEDG1	INTEDG2	-	TMR0IP	-	RBIP
Бит 7							Бит 0

Бит 7 **-RBPU:** Включение подтягивающих резисторов на входах PORTB
1 = все подтягивающие резисторы выключены
0 = подтягивающие резисторы включены на выводах PORTB, настроенных как вход

Бит 6 **INTEDG0:** Выбор активного фронта внешнего прерывания INT0
1 = прерывание по переднему фронту сигнала
0 = прерывание по заднему фронту сигнала

Бит 5 **INTEDG1:** Выбор активного фронта внешнего прерывания INT1
1 = прерывание по переднему фронту сигнала
0 = прерывание по заднему фронту сигнала

Бит 4 **INTEDG2:** Выбор активного фронта внешнего прерывания INT2
1 = прерывание по переднему фронту сигнала
0 = прерывание по заднему фронту сигнала

Бит 3 **Не используется:** Читается как ‘0’

Бит 2 **TMR0IP:** Выбор приоритета прерывания по переполнению таймера TMR0
1 = высокий приоритет
0 = низкий приоритет

Бит 1 **Не используется:** Читается как ‘0’

Бит 0 **RBIP:** Выбор приоритета прерывания по изменению уровня сигнала на входах PORTB
1 = высокий приоритет
0 = низкий приоритет

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Примечание. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита общего разрешения прерываний. Это позволяет выполнять программный контроль возникновения условия прерываний. Необходимо заботиться о том, чтобы флаг прерывания был сброшен перед разрешением прерывания.

Регистр 8-3. Регистр INTCON3

R/W - 1	R/W - 1	U - 0	R/W - 0	R/W - 0	U - 0	R/W - 0	R/W - 0
INT2IP	INT1IP	-	INT2IE	INT1IE	-	INT2IF	INT1IF
Бит 7							Бит 0

Бит 7 **INT2IP:** Выбор приоритета внешнего прерывания INT2
1 = высокий приоритет
0 = низкий приоритет

Бит 6 **INT1IP:** Выбор приоритета внешнего прерывания INT1
1 = высокий приоритет
0 = низкий приоритет

Бит 5 **Не используется:** Читается как ‘0’

Бит 4 **INT2IE:** Разрешение внешнего прерывания INT2
1 = внешнее прерывание INT2 разрешено
0 = внешнее прерывание INT2 запрещено

Бит 3 **INT1IE:** Разрешение внешнего прерывания INT1
1 = внешнее прерывание INT1 разрешено
0 = внешнее прерывание INT1 запрещено

Бит 2 **Не используется:** Читается как ‘0’

Бит 1 **INT2IF:** Флаг внешнего прерывания INT2
1 = выполнено условие внешнего прерывания INT2 (сбрасывается программно)
0 = условие внешнего прерывания INT2 не выполнено

Бит 0 **INT1IF:** Флаг внешнего прерывания INT1
1 = выполнено условие внешнего прерывания INT1 (сбрасывается программно)
0 = условие внешнего прерывания INT1 не выполнено

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Примечание. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита общего разрешения прерываний. Это позволяет выполнять программный контроль возникновения условия прерываний. Необходимо заботиться о том, чтобы флаг прерывания был сброшен перед разрешением прерывания.

8.2 Регистры PIR

Регистры PIR содержат индивидуальные флаги периферийных прерываний. В соответствии с числом периферийных прерываний реализовано два регистра PIR1 и PIR2.

Примечания:
1. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита глобального разрешения прерываний GIE(INTCON<7>).
2. Пользователь может выполнять программный контроль возникновения условия прерываний. Необходимо заботиться о том, чтобы флаг прерывания был сброшен перед разрешением прерывания и после обработки прерывания.

Регистр 8-4. Регистр флагов периферийных прерываний PIR1

R/W - 0	R/W - 0	R - 0	R - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
PSPIF[1]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
Бит 7							Бит 0

Бит 7 **PSPIF[1]:** Флаг прерывания ведомого параллельного порта PSP
1 = произошла операция чтения или записи (сбрасывается программно)
0 = операции чтения или записи не выполнялось

Бит 6 **ADIF:** Флаг прерывания от модуля АЦП
1 = преобразование АЦП завершено (сбрасывается программно)
0 = преобразование АЦП не завершено

Бит 5 **RCIF:** Флаг прерывания от приемника USART
1 = буфер приемника USART полон (сбрасывается чтением регистра RCREG)
0 = буфер приемника USART пуст

Бит 4 **TXIF:** Флаг прерывания от передатчика USART
1 = буфер передатчика USART пуст (сбрасывается записью в регистр TXREG)
0 = буфер передатчика USART полон

Бит 3 **SSPIF:** Флаг прерываний от модуля MSSP
1 = выполнено условие возникновения прерывания от модуля SSP (сбрасывается программно)
0 = условие возникновения прерывания от модуля SSP не выполнено

Бит 2 **CCP1IF:** Флаг прерывания от модуля CCP1
Режим захвата
1 = выполнен захват значения TMR1 (сбрасывается программно)
0 = захвата значения TMR1 не происходило

Режим сравнения
1 = значение TMR1 достигло указанного в регистрах сравнения (сбрасывается программно)
0 = значение TMR1 не достигло указанного в регистрах сравнения

ШИМ режим
Не используется

Бит 1 **TMR2IF:** Флаг прерывания переполнения таймера TMR2
1 = произошло переполнение таймера TMR2 (сбрасывается программно)
0 = переполнение таймера TMR2 не происходило

Бит 0 **TMR1IF:** Флаг прерывания переполнения таймера TMR1
1 = произошло переполнение таймера TMR0 (сбрасывается программно)
0 = переполнение таймера TMR1 не происходило

Примечание 1. Бит PSPIF в микроконтроллерах PIC18F2X2 не реализован, при записи должен равняться ‘0’.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 8-5. Регистр флагов периферийных прерываний PIR2

U - 0	U - 0	U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF
Бит 7							Бит 0

Бит 7-5 **Не используется:** Читается как ‘0’

Бит 4 **EEIF:** Флаг прерывания по окончании записи в EEPROM данных / Flash памяти программ
1 = запись данных завершена (сбрасывается программно)
0 = запись данных не завершена или не была начата

Бит 3 **BCLIF:** Флаг прерываний возникновения коллизий на шине
1 = на шине обнаружены коллизии (сбрасывается программно)
0 = коллизий не обнаружено

Бит 2 **LVDIF:** Флаг прерывания от детектора пониженного напряжения
1 = обнаружено снижение напряжения питания (сбрасывается программно)
0 = напряжение питания выше установленного значения

Бит 1 **TMR3IF:** Флаг прерывания переполнения таймера TMR3
1 = произошло переполнение таймера TMR3 (сбрасывается программно)
0 = переполнение таймера TMR3 не происходило

Бит 0 **CCP2IF:** Флаг прерывания от модуля CCP2
Режим захвата
1 = выполнен захват значения TMR1 (сбрасывается программно)
0 = захвата значения TMR1 не происходило

Режим сравнения
1 = значение TMR1 достигло указанного в регистрах сравнения (сбрасывается программно)
0 = значение TMR1 не достигло указанного в регистрах сравнения

ШИМ режим
Не используется

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

8.3 Регистры PIE

Регистры PIE содержат индивидуальные биты разрешения периферийных прерываний. В соответствии с числом периферийных прерываний реализовано два регистра PIE1 и PIE2. Если бит IPEN=0, то для разрешения периферийных прерываний необходимо установить бит PEIE.

Регистр 8-6. Регистр разрешения периферийных прерываний PIE1

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
PSPIE[1]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
Бит 7							Бит 0

Бит 7 **PSPIE[1]:** Разрешение прерывания от ведомого параллельного порта PSP
1 = прерывание разрешено
0 = прерывание запрещено

Бит 6 **ADIE:** Разрешение прерывания от модуля АЦП
1 = прерывание разрешено
0 = прерывание запрещено

Бит 5 **RCIE:** Разрешение прерывания от приемника USART
1 = прерывание разрешено
0 = прерывание запрещено

Бит 4 **TXIE:** Разрешение прерывания от передатчика USART
1 = прерывание разрешено
0 = прерывание запрещено

Бит 3 **SSPIE:** Разрешение прерываний от модуля MSSP
1 = прерывание разрешено
0 = прерывание запрещено

Бит 2 **CCP1IE:** Разрешение прерывания от модуля CCP1
1 = прерывание разрешено
0 = прерывание запрещено

Бит 1 **TMR2IE:** Разрешение прерывания по переполнению таймера TMR2
1 = прерывание разрешено
0 = прерывание запрещено

Бит 0 **TMR1IE:** Разрешение прерывания по переполнению таймера TMR1
1 = прерывание разрешено
0 = прерывание запрещено

Примечание 1. Бит PSPIE в микроконтроллерах PIC18F2X2 не реализован, при записи должен равняться ‘0’.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 8-7. Регистр разрешения периферийных прерываний PIE2

U - 0	U - 0	U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE
Бит 7							Бит 0

Бит 7-5 **Не используется:** Читается как ‘0’

Бит 4 **EEIE:** Разрешение прерывания по окончании записи в EEPROM данных / Flash памяти программ
1 = прерывание разрешено
0 = прерывание запрещено

Бит 3 **BCLIE:** Разрешение прерываний при возникновении коллизий на шине
1 = прерывание разрешено
0 = прерывание запрещено

Бит 2 **LVDIE:** Разрешение прерывания от детектора пониженного напряжения
1 = прерывание разрешено
0 = прерывание запрещено

Бит 1 **TMR3IE:** Разрешение прерывания по переполнению таймера TMR3
1 = прерывание разрешено
0 = прерывание запрещено

Бит 0 **CCP2IE:** Разрешение прерывания от модуля CCP2
1 = прерывание разрешено
0 = прерывание запрещено

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

8.4 Регистры IRP

Регистры IRP содержат индивидуальные биты приоритета периферийных прерываний. В соответствии с числом периферийных прерываний реализовано два регистра IRP1 и IRP2. Для включения приоритетной системы прерываний необходимо, чтобы бит IPEN был установлен в ‘1’.

Регистр 8-8. Регистр приоритета периферийных прерываний IRP1

R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
PSPIP[1]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP
Бит 7							Бит 0

Бит 7 **PSPIP[1]:** Выбор приоритета прерывания от ведомого параллельного порта PSP
1 = высокий приоритет
0 = низкий приоритет

Бит 6 **ADIP:** Выбор приоритета прерывания от модуля АЦП
1 = высокий приоритет
0 = низкий приоритет

Бит 5 **RCIP:** Выбор приоритета прерывания от приемника USART
1 = высокий приоритет
0 = низкий приоритет

Бит 4 **TXIP:** Выбор приоритета прерывания от передатчика USART
1 = высокий приоритет
0 = низкий приоритет

Бит 3 **SSPIP:** Выбор приоритета прерываний от модуля MSSP
1 = высокий приоритет
0 = низкий приоритет

Бит 2 **CCP1IP:** Выбор приоритета прерывания от модуля CCP1
1 = высокий приоритет
0 = низкий приоритет

Бит 1 **TMR2IP:** Выбор приоритета прерывания по переполнению таймера TMR2
1 = высокий приоритет
0 = низкий приоритет

Бит 0 **TMR1IP:** Выбор приоритета прерывания по переполнению таймера TMR1
1 = высокий приоритет
0 = низкий приоритет

Примечание 1. Бит PSPIP в микроконтроллерах PIC18F2X2 не реализован, при записи должен равняться ‘1’.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 8-9. Регистр приоритета периферийных прерываний IRP2

U - 0	U - 0	U - 0	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP
Бит 7							Бит 0

Бит 7-5 **Не используется:** Читается как ‘0’

Бит 4 **EEIP:** Выбор приоритета прерывания по окончании записи в EEPROM данных / Flash памяти программ
1 = высокий приоритет
0 = низкий приоритет

Бит 3 **BCLIP:** Выбор приоритета прерываний при возникновении коллизий на шине
1 = высокий приоритет
0 = низкий приоритет

Бит 2 **LVDIP:** Выбор приоритета прерывания от детектора пониженного напряжения
1 = высокий приоритет
0 = низкий приоритет

Бит 1 **TMR3IP:** Выбор приоритета прерывания по переполнению таймера TMR3
1 = высокий приоритет
0 = низкий приоритет

Бит 0 **CCP2IP:** Выбор приоритета прерывания от модуля CCP2
1 = высокий приоритет
0 = низкий приоритет

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

8.5 Регистр RCON

Регистр RCON содержит бит включения приоритетной системы прерываний (IPEN).

Регистр 8-10. Регистр RCON

R/W - 0	U - 0	U - 0	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
IPEN	-	-	-RI	-TO	-PD	-POR	-BOR
Бит 7							Бит 0

Бит 7 **IPEN:** Разрешение приоритетной системы прерываний
1 = приоритетная система прерываний разрешена
0 = приоритетная система прерываний выключена (для совместимости с PIC16CXXX)

Бит 6-5 **Не используется:** Читается как ‘0’

Бит 4 **-RI:** Флаг выполнения команды RESET
1 = команда RESET не выполнялась
0 = сброс микроконтроллера произошел по выполнению команды RESET
(бит должен быть установлен в ‘1’ после сброса BOR)

Бит 3 **-TO:** Флаг переполнения сторожевого таймера WDT
1 = после сброса POR, выполнения команды CLRWDT или SLEEP
0 = произошло переполнение WDT

Бит 2 **-PD:** Флаг детектора выключения питания
1 = после сброса POR или выполнения команды CLRWDT
0 = после выполнения команды SLEEP

Бит 1 **-POR:** Флаг сброса по включению питания POR
1 = сброса по включению питания не происходило
0 = произошел сброс по включению питания
(бит должен быть установлен в ‘1’ после сброса POR)

Бит 0 **-BOR:** Флаг сброса по снижению напряжения питания
1 = сброса по снижению напряжения питания не происходило
0 = произошел сброс по снижению напряжения питания
(бит должен быть установлен в ‘1’ после сброса BOR)

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

8.6 Внешние прерывания INT

Внешние прерывания с входов RB0/INT0, RB1/INT1 и RB2/INT2 происходит: по переднему фронту сигнала, если бит INTEDGx (регистр INTCON2) установлен в ‘1’; по заднему фронту, если бит INTEDGx сброшен в ‘0’. При появлении ожидаемого фронта сигнала на входе RBx/INTx устанавливается флаг прерываний INTxF. Прерывание может быть запрещено сбросом бита INTxE. Флаг прерывания INTxF должен быть сброшен программно в обработчике прерываний, перед разрешением прерываний. Все внешние прерывания (INT0, INT1 и INT2) могут вывести микроконтроллер из режима SLEEP, если бит INTxE был установлен в ‘1’ до перехода в режим SLEEP. Если разрешены прерывания (GIE=1), то произойдет переход по вектору прерывания.

Приоритет прерываний INT1, INT2 определяется битами INT1IP(INTCON3<6>), INT2IP(INTCON3<7>) соответственно. Обратите внимание, что внешние прерывание INT0 не имеет бита приоритета. Прерывание INT0 всегда с высоким приоритетом.

8.7 Прерывание от TMR0

В 8-разрядном режиме таймера TMR0 (режим по умолчанию) при переполнении регистра TMR0 (переход от FFh к 00h) происходит установка флага прерываний TMR0IF. В 16-разрядном режиме флаг прерывания TMR0IF устанавливается в ‘1’, когда происходит переполнение сдвоенного регистра TMR0H:TMR0L (переход от FFFFh к 0000h). Прерывание может быть разрешено/запрещено битом TMR0IE (INTCON<5>). Приоритет прерывания по переполнению таймера TMR0 определяется битом TMR0IP (INTCON2<2>). Дополнительную информацию по работе таймера смотрите в разделе 10.

8.8 Прерывание по изменению сигнала на входах PORTB

Изменение логического уровня сигнала на входах RB7:RB4 вызывает установки флага прерываний RBIF(INTCON<0>). Прерывание может быть разрешено/запрещено битом RBIE (INTCON<3>). Приоритет прерывания определяется битом RBIP (INTCON2<0>).

8.9 Сохранение контекста

При переходе на обработку прерываний в стеке сохраняется только адрес возврата. Дополнительно в стеке могут быть сохранены значения регистров WREG, STATUS, BSR. Если быстрое возвращение из прерываний не используется (смотрите раздел 4.3), то значения регистров WREG, STATUS, BSR сохраняется программным способом. В зависимости от приложения могут сохраняться и другие регистры. В примере 8-1 представлены операции сохранения и восстановления значений регистров WREG, STATUS, BSR при обработке прерываний.

Пример 8-1. Сохранение и восстановление значений регистров WREG, STATUS, BSR при обработке прерываний.

```
MOVWF    W_TEMP                ; Сохранение W_
MOVFF    STATUS, STATUS_TEMP   ; Сохранение STATUS_TEMP
MOVFF    BSR, BSR_TEMP         ; Сохранение BSR;
; Код пользователя
;
MOVFF    BSR_TEMP, BSR         ; Восстановление BSR
MOVF     W_TEMP, W             ; Восстановление WREG
MOVFF    STATUS_TEMP,STATUS    ; Восстановление STATUS
```

9. Порты ввода/вывода

В зависимости от типа микроконтроллера реализовано пять или три порта ввода/вывода. Некоторые каналы портов ввода/вывода мультиплексированы с дополнительными функциями периферийных модулей микроконтроллера. В общем случае, когда используется периферийная функция, вывод не может использоваться как канал порта ввода/вывода.

Каждому порту соответствует три управляющих регистра:

- TRIS – регистр выбора направления данных в каналах порта ввода/вывода
- PORT – регистр порта (результатом чтения является логический уровень сигнала на выводах)
- LAT – защелка порта ввода/вывода

Защелка порта ввода/вывода LAT особенно полезна при использовании команд со структурой «чтение - модификация - запись».

9.1 Регистры PORTA, TRISA, LATA

PORTA – 7-разрядный порт ввода/вывода. Все каналы PORTA имеют соответствующие биты направления в регистре TRISA, позволяющие настраивать канал как вход или выход. Запись ‘1’ в TRISA переводит соответствующий выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISA определяет соответствующий канал как выход, содержимое защелки PORTA передается на вывод микроконтроллера.

Чтение регистра PORTA возвращает состояние на выводах порта, а запись производится в защелку PORTA.

Регистр защелки LATA отображается на память данных. Операция типа «чтение – модификация – запись» с регистром LATA будет выполнена с данными, записанными в порт ввода/вывода PORTA.

Вывод RA4/T0CKI имеет триггер Шмитта на входе и открытый сток на выходе, мультиплексирован с тактовым входом таймера TMR0. Все остальные каналы PORTA имеют TTL буфер на входе и полнофункциональные выходные КМОП буферы.

Другие каналы PORTA мультиплексированы с аналоговыми входами АЦП и аналоговым входом источника опорного напряжения V_{REF+} и V_{REF-}. Биты управления режимом работы каналов порта ввода/вывода PORTA находятся в регистре ADCON1 (управляющий регистр АЦП).

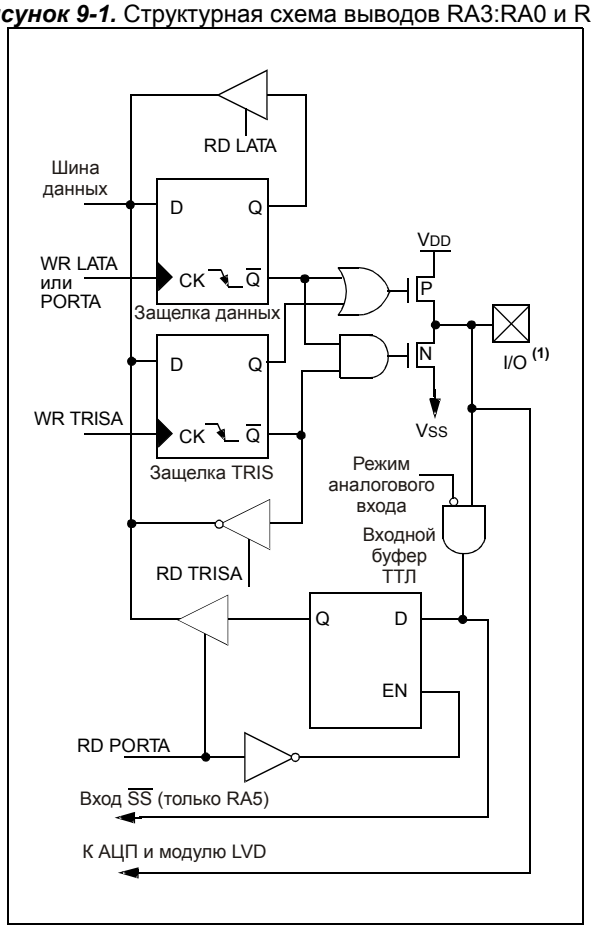
Примечание. После сброса по включению питания выводы RA5, RA3:RA0 настраиваются как аналоговые входы, чтение дает результат '0'. Выводы RA4, RA6 при сбросе POR настраиваются как цифровые входы.

Биты регистра TRISA управляют направлением каналов PORTA, даже когда они используются как аналоговые входы. Пользователь должен удостовериться, что соответствующие каналы PORTA настроены на вход при использовании их в качестве аналоговых входов.

Пример 9-1. Инициализация PORTA

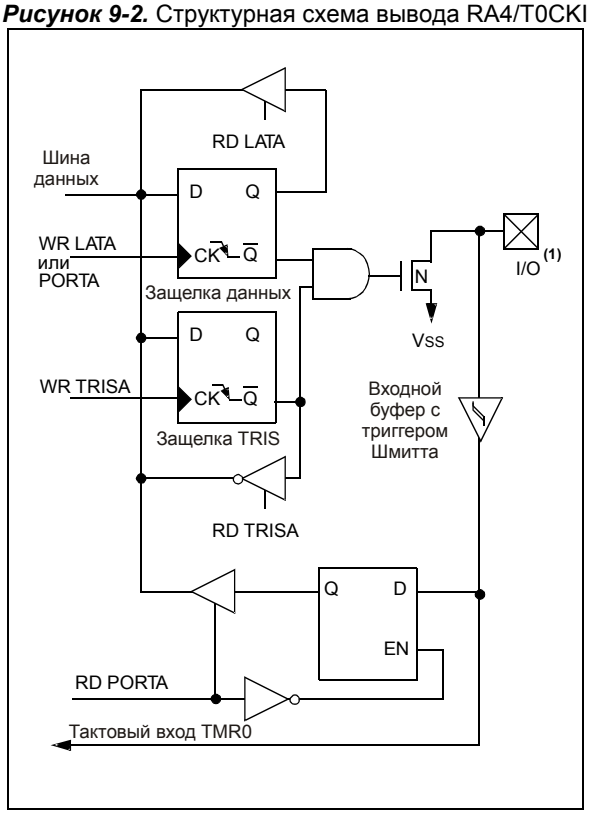
```
CLRF    PORTA          ; Инициализация PORTA
                       ; с очисткой выходной
                       ; защелки данных
CLRF    LATA           ; Альтернативный метод
                       ; очистки выходной
                       ; защелки данных
MOVLW   0x07           ; Настройка АЦП
MOVWF   ADCON1         ; как цифровых входов
MOVLW   0xCF           ; Значение
                       ; инициализации
                       ; направления данных
MOVWF   TRISA          ; Установить RA<3:0> входами
                       ; RA<5:4> выходами
```

Рисунок 9-1. Структурная схема выводов RA3:RA0 и RA5



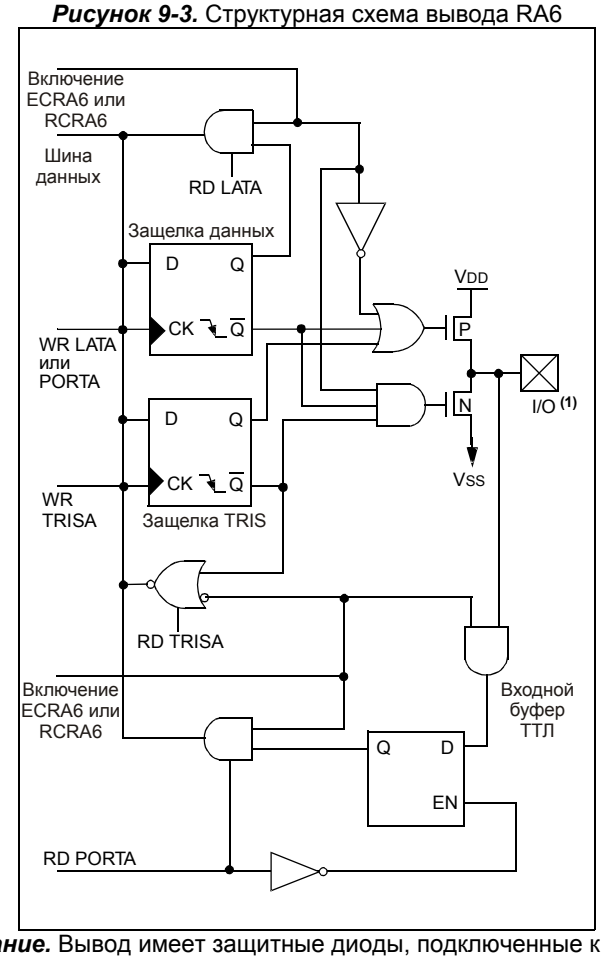
Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Рисунок 9-2. Структурная схема вывода RA4/T0CKI



Примечание. Вывод имеет защитный диод, подключенный только к V_{SS}.

Рисунок 9-3. Структурная схема вывода RA6



Примечание. Вывод имеет защитные диоды, подключенные к V_{DD} и V_{SS}.

Таблица 9-1. Функциональное назначение выводов PORTA

Обозначение	№ бита	Буфер	Описание
RA0/AN0	0	TTL	Вход/выход или аналоговый вход.
RA1/AN1	1	TTL	Вход/выход или аналоговый вход.
RA2/AN2/V_{REF-}	2	TTL	Вход/выход, аналоговый вход или V_{REF-}.
RA3/AN3/V_{REF+}	3	TTL	Вход/выход, аналоговый вход или V_{REF+}.
RA4/T0CKI	4	ST	Вход/выход или вход тактового сигнала для TMR0. Выход с открытым коллектором.
RA5/AN4/-SS/LVDIN	5	TTL	Вход/выход, вход выбора ведомого SPI, аналоговый вход или вход детектора пониженного напряжения.
OSC2/CLKO/RA6	6	TTL	OSC2, выход тактового сигнала или канал порта ввода/вывода.

Обозначение: ST = вход с триггером Шмитта; TTL = входной буфер ТТЛ

Таблица 9-2. Регистры и биты, связанные с работой PORTA

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F80h	PORTA	-	RA6	RA5	RA4	RA3	RA2	RA1	RA0	-x0x 0000
F89h	LATA	-	Регистр выходных данных							-xxx xxxx
F92h	TRISA	-	Регистр направления данных							-111 1111
FC1h	ADCON1	ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

9.2 Регистры PORTB, TRISB, LATB

PORTB – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISB определяют направление каналов порта. Установка битва в ’1’ регистра TRISB переводит выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISB настраивает соответствующий канал как выход, содержимое защелки PORTB передается на вывод микроконтроллера.

Регистр защелки LATB отображается на память данных. Операция типа «чтение – модификация – запись» с регистром LATB будет выполнена с данными, записанными в порт ввода/вывода PORTB.

Пример 9-2. Инициализация PORTB

```
CLRF    PORTB          ; Инициализация PORTB
                       ; с очисткой выходной
                       ; защелки данных
CLRF    LATB           ; Альтернативный метод
                       ; очистки выходной
                       ; защелки данных
MOVLW   0xCF           ; Значение
                       ; инициализации
                       ; направления данных
MOVWF   TRISB          ; Установить RB<3:0> входами
                       ; RB<5:4> выходами
                       ; RB<7:6> входами
```

К каждому выводу PORTB подключен внутренний подтягивающий резистор. Бит -RBPU (INTCON2<7>) определяет, подключены (-RBPU=0) или нет (-RBPU=1) подтягивающие резисторы. Подтягивающие резисторы автоматически отключаются после сброса по включению питания POR, и когда каналы порта настраиваются на выход.

Примечание. При сбросе POR каналы порта ввода/вывода PORTB настраиваются как цифровые входы.

Четыре канала PORTB RB7:RB4, настроенные на вход, могут генерировать прерывания по изменению логического уровня сигнала на входе. Если один из каналов RB7:RB4 настроен на выход, то он не может быть источником прерываний. Сигнал на выводах RB7:RB4 сравнивается со значением, сохраненным при последнем чтении PORTB. В случае несовпадения одного из значений устанавливается флаг RBIF (INTCON<0>), и если разрешено, генерируется прерывание.

Это прерывание может вывести микроконтроллер из режима SLEEP. В подпрограмме обработки прерываний необходимо сделать следующие действия:

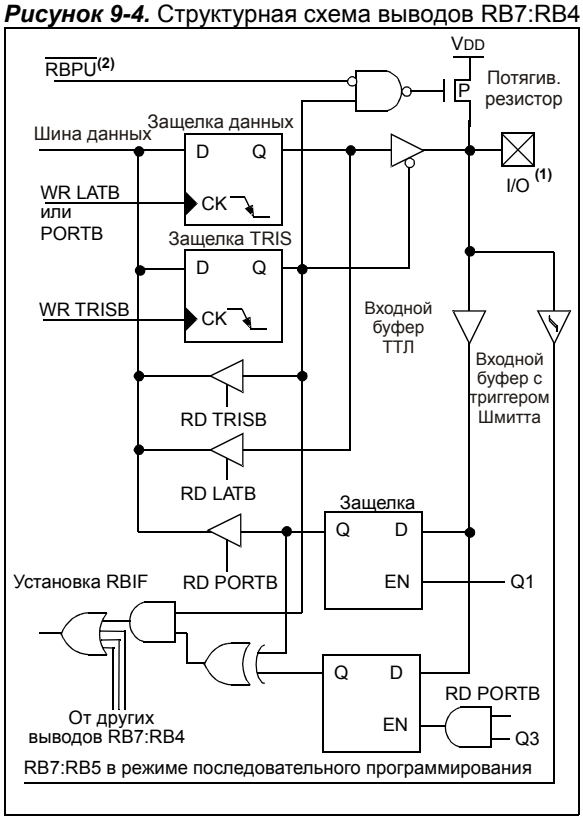
- Выполнить чтение или запись в PORTB, исключив несоответствие
- Сбросить флаг RBIF в ’0’

Несоответствие сохраненного значения с сигналом на входе PORTB всегда устанавливает бит RBIF в ‘1’. Чтение из PORTB прервет условие несоответствия и позволит сбросить флаг RBIF в ‘0’.

Прерывания по изменению сигнала на входах рекомендуется использовать для определения нажатия клавиш, когда PORTB полностью задействован для реализации клавиатуры. Не рекомендуется опрашивать PORTB при использовании прерываний по изменению входного сигнала.

Вывод RB3 может быть настроен битом конфигурации CCP2MX как дополнительный периферийный вывод модуля CCP2 (CCP2MX=0).

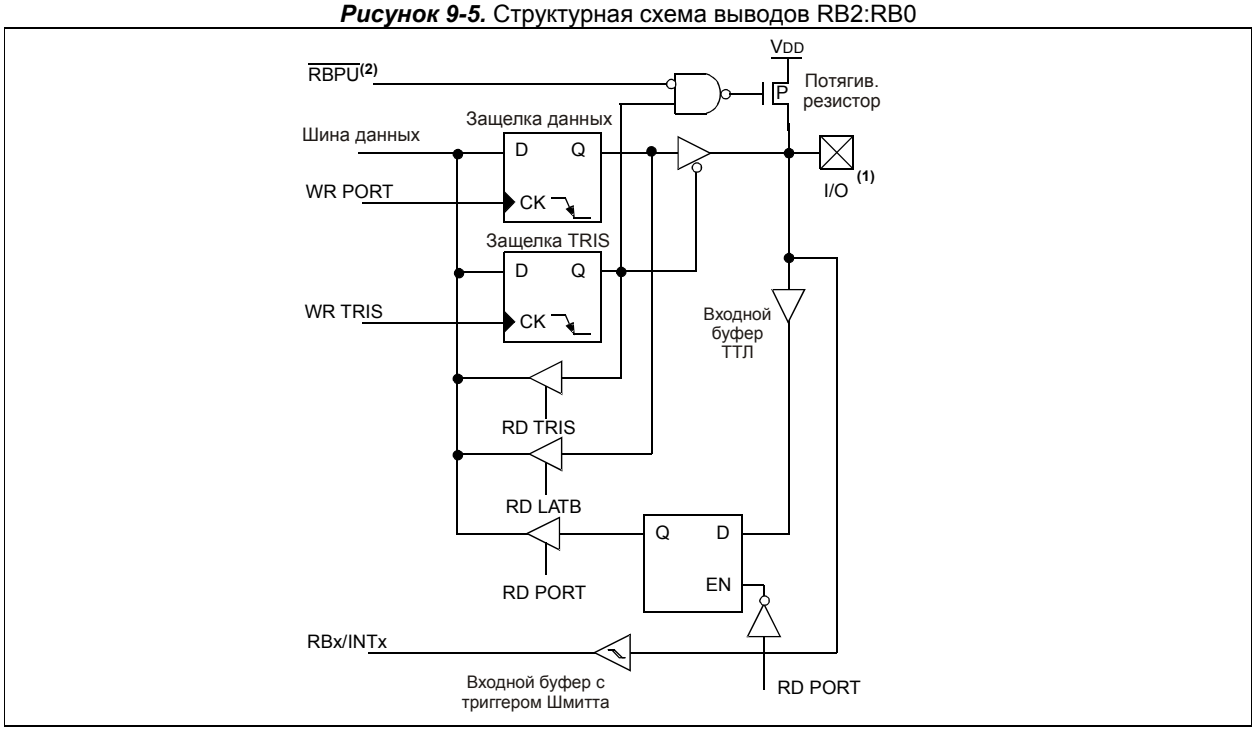
Рисунок 9-4. Структурная схема выводов RB7:RB4



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Для включения подтягивающих резисторов необходимо установить в '1' соответствующий бит TRISB и сбросить в '0' бит -RBPU (INTCON2<7>).

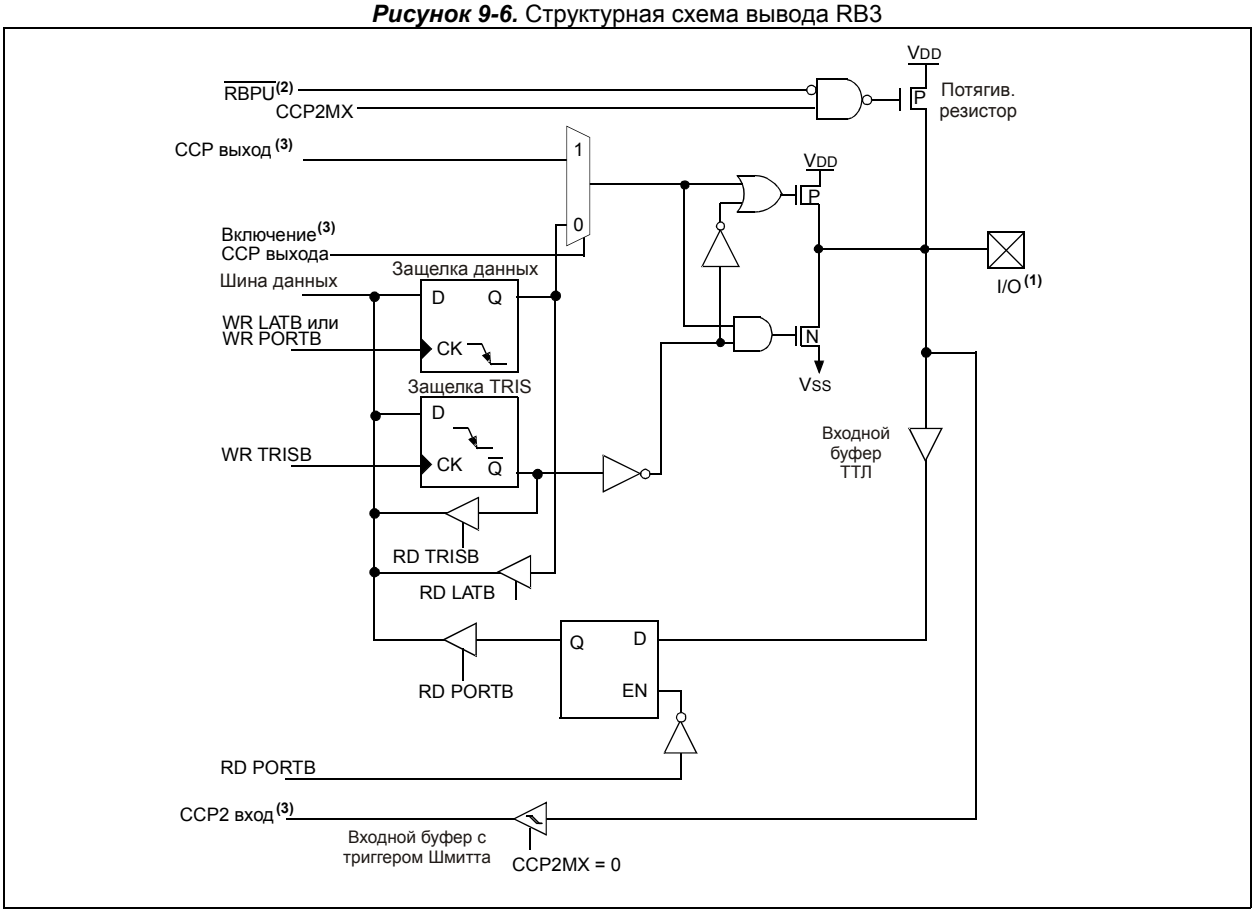
Рисунок 9-5. Структурная схема выводов RB2:RB0



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Для включения подтягивающих резисторов необходимо установить в '1' соответствующий бит TRISB и сбросить в '0' бит -RBPU (INTCON2<7>).

Рисунок 9-6. Структурная схема вывода RB3



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Для включения подтягивающих резисторов необходимо установить в '1' соответствующий бит TRISB и сбросить в '0' бит -RBPU (INTCON2<7>).
3. CCP2 мультиплексирован с RB3, если CCP2MX=0 (в регистре конфигурации).

Таблица 9-3. Функциональное назначение выводов PORTB

Обозначение	№ бита	Буфер	Описание
RB0/INT0	0	TTL/ST(1)	Вход/выход или внешнее прерывание INT0. Внутренние подтягивающие резисторы.
RB1/INT1	1	TTL/ST(1)	Вход/выход или внешнее прерывание INT1. Внутренние подтягивающие резисторы.
RB2/INT2	2	TTL/ST(1)	Вход/выход или внешнее прерывание INT2. Внутренние подтягивающие резисторы.
RB3/CCP2/INT3(3)	3	TTL/ST(4)	Вход/выход или внешнее прерывание INT3. Вход захвата 2, выход сравнения 2, выход ШИМ 2, если бит CCP2MX = 0 в битах конфигурации. Внутренние подтягивающие резисторы.
RB4	4	TTL	Вход/выход (прерывание по изменению сигнала на входе). Внутренние подтягивающие резисторы.
RB5/PGM	5	TTL/ST(2)	Вход/выход (прерывание по изменению сигнала на входе). Внутренние подтягивающие резисторы. Включение режима низковольтного программирования ICSP.
RB6/PGC	6	TTL/ST(2)	Вход/выход (прерывание по изменению сигнала на входе). Внутренние подтягивающие резисторы. Вход тактового сигнала для внутрисхемного программирования ICSP.
RB7/PGD	7	TTL/ST(2)	Вход/выход (прерывание по изменению сигнала на входе). Внутренние подтягивающие резисторы. Вывод данных для внутрисхемного программирования ICSP.

Обозначение: ST = вход с триггером Шмитта; TTL = входной буфер ТТЛ

Примечания:

1. Входной буфер с триггером Шмитта при использовании внешних прерываний.
2. Входной буфер с триггером Шмитта при работе в режиме последовательного программирования.
3. Режим работы вывода (RB3 или CCP2) определяется битом CCP2MX в регистре конфигурации.
4. Входной буфер с триггером Шмитта при использовании вывода как CCP2.

Таблица 9-4. Регистры и биты, связанные с работой PORTB

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F81h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx
F8Ah	LATB	Регистр выходных данных								xxxx xxxx
F93h	TRISB	Регистр направления данных								1111 1111
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FF1h	INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	-	TMR0IP	-	RBIP	1111 -1-1
FF0h	INTCON3	INT2IP	INT1IP	-	INT2IE	INT1IE	-	INT2IF	INT1IF	11-0 0-00

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

9.3 Регистры PORTC, TRISC, LATC

PORTC – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISC определяют направление каналов порта. Установка бита в ’1’ регистра TRISC переводит выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISC настраивает соответствующий канал как выход, содержимое защелки PORTC передается на вывод микроконтроллера.

Регистр защелки LATC отображается на память данных. Операция типа «чтение – модификация – запись» с регистром LATC будет выполнена с данными, записанными в порт ввода/вывода PORTC.

Выводы PORTC мультиплексированы с несколькими периферийными модулями (см. таблицу 9-5). На каналах PORTC присутствует входной буфер с триггером Шмитта.

При использовании периферийных модулей необходимо соответствующим образом настраивать биты регистра TRISC для каждого вывода PORTC. Некоторые периферийные модули отменяют действие битов TRISC, принудительно настраивая вывод на вход или выход. Требования к настройке битов TRISC смотрите в описании на соответствующий периферийный модуль.

Примечание. При сбросе POR каналы порта ввода/вывода PORTC настраиваются как цифровые входы.

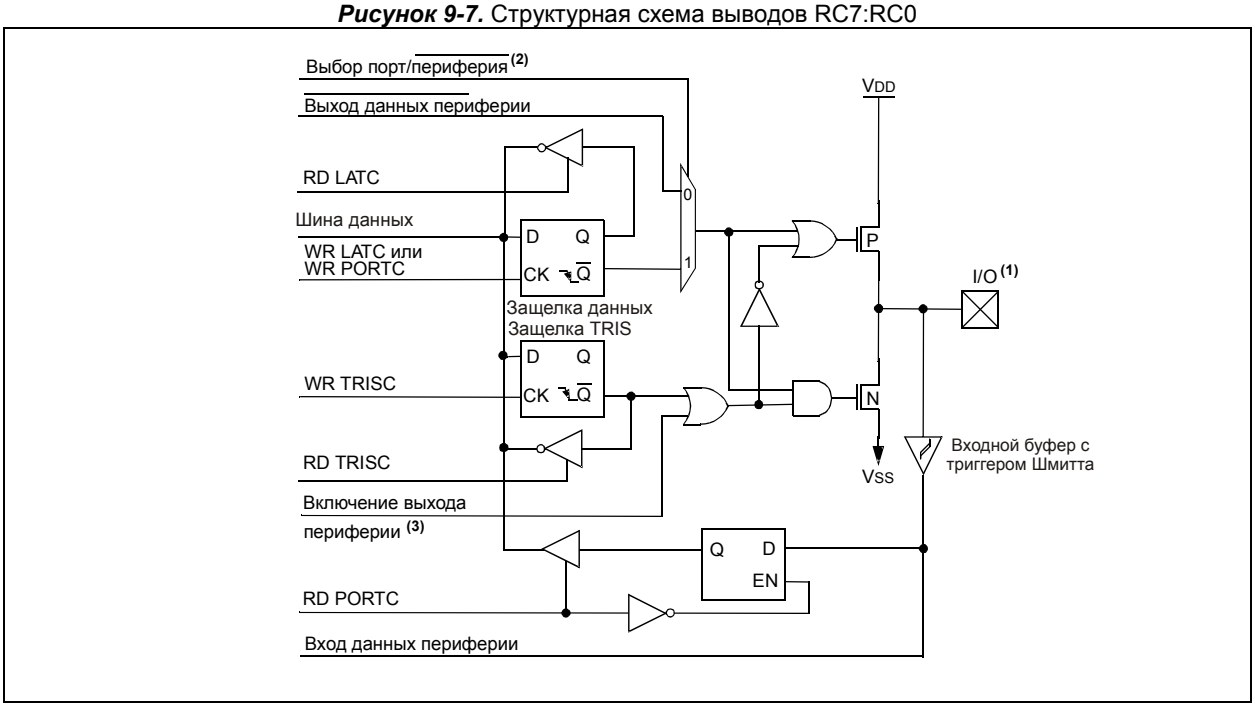
Реальное направление данных канала порта ввода/вывода не загружается в регистр TRISC, что позволяет использовать команды «чтение – модификация – запись» при обращении к регистру TRISC без ограничений.

Режим работы вывода RC1 управляется битом CCP2MX в регистрах конфигурации. По умолчанию вывод RC1 мультиплексирован с выводом периферийного модуля CCP2 (CCP2MX=1).

Пример 9-3. Инициализация PORTC

```
CLRF    PORTC          ; Инициализация PORTC
                       ; с очисткой выходной
                       ; защелки данных
CLRF    LATC           ; Альтернативный метод
                       ; очистки выходной
                       ; защелки данных
MOVLW   0xCF           ; Значение
                       ; инициализации
                       ; направления данных
MOVWF   TRISC          ; Установить RC<3:0> входами
                       ; RC<5:4> выходами
                       ; RC<7:6> входами
```

Рисунок 9-7. Структурная схема выводов RC7:RC0



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Режима канала – вывод используется периферийным модулем или цифровой порт ввода/вывода.
3. Сигнал разрешения (OE) от периферийного модуля, настраивать канал как выход.

Таблица 9-5. Функциональное назначение выводов PORTC

Обозначение	№ бита	Буфер	Описание
RC0/T1OSO/T1CKI	0	ST	Цифровой канал порта ввода/вывода. Выход для подключения кварцевого резонатора TMR1. Вход тактового сигнала для TMR1/TMR3.
RC1/T1OSI/CCP2	1	ST	Цифровой канал порта ввода/вывода. Вход для подключения кварцевого резонатора TMR1. Вход захвата 2, выход сравнения 2, выход ШИМ.
RC2/CCP1	2	ST	Цифровой канал порта ввода/вывода. Вход захвата 1, выход сравнения 1, выход ШИМ 1.
RC3/SCK/SCL	3	ST	Цифровой канал порта ввода/вывода. Вход/выход тактового сигнала в режиме SPI. Вход/выход тактового сигнала в режиме I^2C.
RC4/SDI/SDA	4	ST	Цифровой канал порта ввода/вывода. Вход данных в режиме SPI. Вход/выход данных в режиме I^2C.
RC5/SDO	5	ST	Цифровой канал порта ввода/вывода. Выход данных в режиме SPI.
RC6/TX/CK	6	ST	Цифровой канал порта ввода/вывода. Выход передатчика USART в асинхронном режиме. Вывод синхронизации в синхронном режиме USART.
RC7/RX/DT	7	ST	Цифровой канал порта ввода/вывода. Вход приемника USART в асинхронном режиме. Вывод данных USART в синхронном режиме.

Обозначение: ST = вход с триггером Шмитта

Таблица 9-6. Регистры и биты, связанные с работой PORTC

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F82h	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx
F8Bh	LATC	Регистр выходных данных								xxxx xxxx
F94h	TRISC	Регистр направления данных								1111 1111

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

9.4 Регистры PORTD, TRISD, LATD

Этот раздел предназначен только для микроконтроллеров PIC18F4X2.

PORTD – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISD определяют направление каналов порта. Установка бита в ’1’ регистра TRISD переводит выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISD настраивает соответствующий канал как выход, содержимое защелки PORTD передается на вывод микроконтроллера.

Регистр защелки LATD отображается на память данных. Операция типа «чтение – модификация – запись» с регистром LATD будет выполнена с данными, записанными в порт ввода/вывода PORTD.

На каналах PORTD присутствует входной буфер с триггером Шмитта. Каждый канал PORTD индивидуально настраивается на вход или выход.

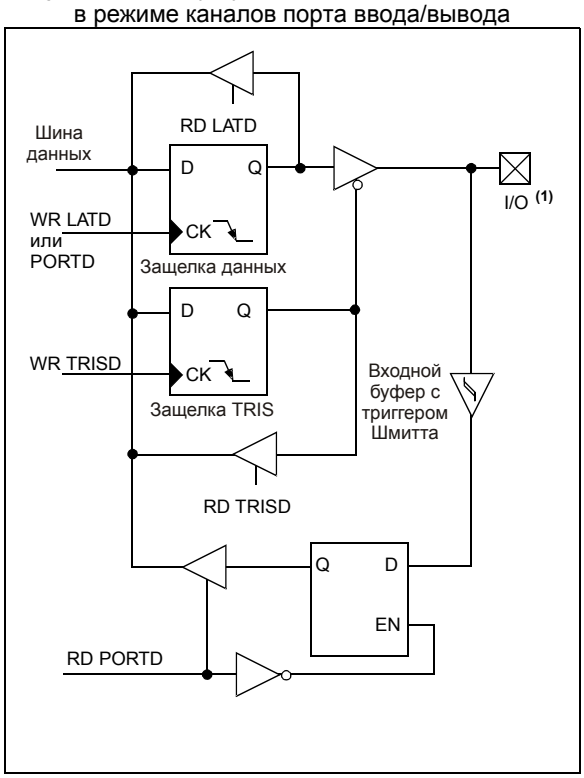
Примечание. При сбросе POR каналы порта ввода/вывода PORTD настраиваются как цифровые входы.

PORTD может работать как 8-разрядный микропроцессорный порт (ведомый параллельный порт), если бит PSPMODE (TRISE<4>) установлен в '1'. В режиме ведомого параллельного порта к входам подключены буферы ТТЛ. Подробное описание работы с ведомым параллельным портом смотрите в разделе 9.6.

Пример 9-4. Инициализация PORTD

```
CLRF    PORTD          ; Инициализация PORTD
                       ; с очисткой выходной
                       ; защелки данных
CLRF    LATD           ; Альтернативный метод
                       ; очистки выходной
                       ; защелки данных
MOVLW   0xCF           ; Значение
                       ; инициализации
                       ; направления данных
MOVWF   TRISD          ; Установить RD<3:0> входами
                       ; RD<5:4> выходами
                       ; RD<7:6> входами
```

Рисунок 9-8. Структурная схема выводов RD7:RD0 в режиме каналов порта ввода/вывода



Примечание 1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Таблица 9-7. Функциональное назначение выводов PORTD

Обозначение	№ бита	Буфер	Описание
RD0/PSP0	0	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 0.
RD1/PSP1	1	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 1.
RD2/PSP2	2	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 2.
RD3/PSP3	3	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 3.
RD4/PSP4	4	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 4.
RD5/PSP5	5	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 5.
RD6/PSP6	6	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 6.
RD7/PSP7	7	ST/TTL(1)	Вход/выход или параллельный ведомый порт бит 7.

Обозначение: ST = вход с триггером Шмитта; TTL = входной буфер ТТЛ

Примечание 1. Входной буфер с триггером Шмитта в режиме каналов порта ввода/вывода, входной буфер ТТЛ в режиме ведомого параллельного порта.

Таблица 9-8. Регистры и биты, связанные с работой PORTD

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F83h	PORD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx
F8Dh	LATD	Регистр выходных данных								xxxx xxxx
F95h	TRISD	Регистр направления данных								1111 1111
F96h	TRISE	IBF	OBF	PSPMODE	-	-	Регистр направления данных			0000 -111

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

9.5 Регистры PORTE, TRISE, LATE

Этот раздел предназначен только для микроконтроллеров PIC18F4X2.

PORTE – 3-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISE определяют направление каналов порта. Установка бита в ’1’ регистра TRISE переводит выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISE настраивает соответствующий канал как выход, содержимое защелки PORTE передается на вывод микроконтроллера.

Регистр защелки LATE отображается на память данных. Операция типа «чтение – модификация – запись» с регистром LATE будет выполнена с данными, записанными в порт ввода/вывода PORTE.

PORTE имеет три вывода (RE0/-RD/AN5, RE1/-WR/AN6, RE2/-CS/AN7), индивидуально настраиваемые на вход или выход. Выводы PORTE имеют входной буфер Шмитта.

В регистре TRISE размещаются биты управления ведомым параллельным портом.

Выводы PORTE мультиплексированы с аналоговыми входами. Когда каналы PORTE настроены как аналоговые входы, чтение PORTE чтение будет давать результат '0'.

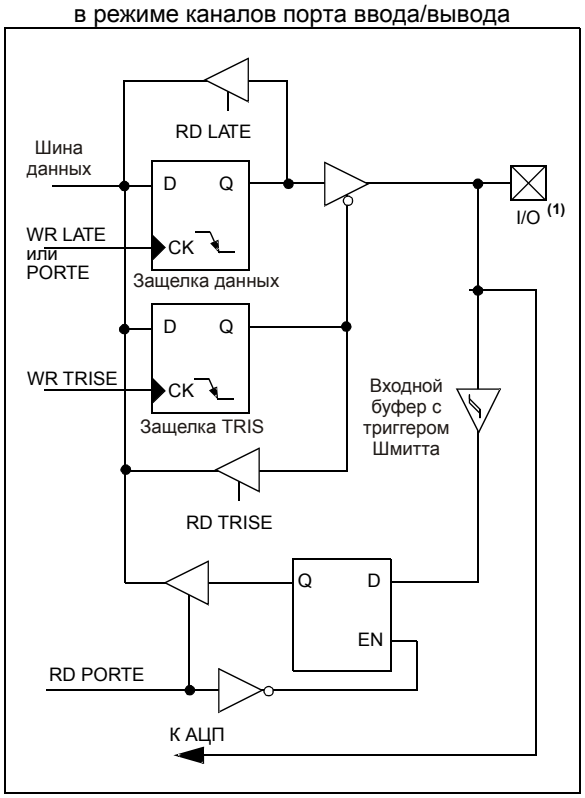
Биты регистра TRISE управляют направлением каналов PORTE, даже когда они используются как аналоговые входы. Пользователь должен удостовериться, что соответствующие каналы PORTE настроены на вход при использовании их в качестве аналоговых входов.

Примечание. После сброса по включению питания выводы PORTE настраиваются как аналоговые входы.

Пример 9-5. Инициализация PORTE

```
CLRF    PORTE       ; Инициализация PORTE
                    ; с очисткой выходной
                    ; защелки данных
CLRF    LATE        ; Альтернативный метод
                    ; очистки выходной
                    ; защелки данных
MOVLW   0x07        ; Настройка АЦП
MOVWF   ADCON1      ; как цифровых входов
MOVLW   0x05        ; Значение
                    ; инициализации
                    ; направления данных
MOVWF   TRISE       ; Установить RE<0> входом
                    ; RA<1> выходом
                    ; RE<2> входом
```

Рисунок 9-9. Структурная схема выводов RE2:RE0 в режиме каналов порта ввода/вывода



Примечание 1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Таблица 9-9. Функциональное назначение выводов PORTE

Обозначение	№ бита	Буфер	Описание
RE0/-RD/AN5	0	ST/TTL(1)	Вход/выход, вход управления чтением ведомого параллельного порта или аналоговый вход: -RD 1 = Ожидание 0 = Операция чтения. Защелка PORTD подключена к выводам PORTD (если -CS = 0)
RE1/-WR/AN6	1	ST/TTL(1)	Вход/выход, вход управления записью ведомого параллельного порта или аналоговый вход: -WR 1 = Ожидание 0 = Операция записи. Данные с выводов PORTD сохраняются во внутренней защелке PORTD (если -CS = 0)
RE2/-CS/AN7	2	ST/TTL(1)	Вход/выход, вход выбора микросхемы ведомого параллельного порта или аналоговый вход: -CS 1 = Микросхема не выбрана 0 = Микросхема выбрана

Обозначение: ST = вход с триггером Шмитта; TTL = входной буфер ТТЛ

Примечание 1. Входной буфер с триггером Шмитта в режиме каналов порта ввода/вывода, входной буфер ТТЛ в режиме ведомого параллельного порта.

Таблица 9-10. Регистры и биты, связанные с работой PORTE

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F84h	PORE	-	-	-	-	-	RE2	RE1	RE0	---- -000
F8Dh	LATE	-	-	-	-	-	Регистр выходных данных			---- -xxx
F96h	TRISE	IBF	OBF	PSPMODE	-	-	Регистр направления данных			0000 -111
FC1h	ADCON1	ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

Регистр 9-1. Регистр TRISE

R - 0	R - 0	R/W - 0	R/W - 0	U - 0	R/W - 1	R/W - 1	R/W - 1
IBF	OBF	IBOV	PSPMODE	-	TRISE2	TRISE1	TRISE0
Бит 7							Бит 0

Бит 7 **IBF:** Бит статуса приемного буфера ведомого параллельного порта
1 = принят байт данных
0 = байт данных не был получен

Бит 6 **OBF:** Бит статуса передающего буфера ведомого параллельного порта
1 = предварительно записанный байт данных еще не прочитан
0 = выходной буфер был прочитан

Бит 5 **IBOV:** Флаг переполнения приемного буфера ведомого параллельного порта
1 = произошла новая запись, а предыдущий байт не был прочитан (сбрасывается программно)
0 = переполнения не было

Бит 4 **PSPMODE:** Выбор режима ведомого параллельного порта
1 = режим ведомого параллельного порта
0 = выводы работают как каналы портов ввода/вывода

Бит 3 **Не используется:** Читается как ‘0’

Бит 2 **TRISE2:** Направление вывода RE2
1 = вход
0 = выход

Бит 1 **TRISE1:** Направление вывода RE1
1 = вход
0 = выход

Бит 0 **TRISE0:** Направление вывода RE0
1 = вход
0 = выход

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

9.6 Ведомый параллельный порт PSP

Этот раздел предназначен только для микроконтроллеров PIC18F4X2.

PORTD может работать как 8-разрядный параллельный порт (или порт микропроцессора), когда бит PSPMODE(TRISE<4>) установлен в '1'. В режиме ведомого параллельного порта данные асинхронно читаются или записываются внешними сигналами -RD (RE0/-RD) или -WR(RE1/-WR) соответственно.

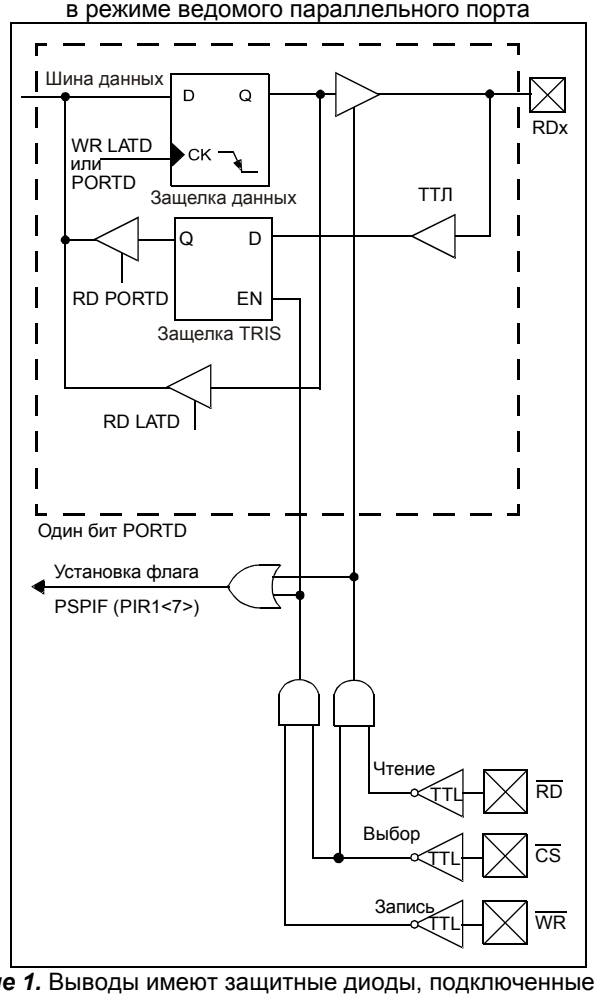
Внешний микропроцессор может читать/записывать данные в PORTD. Операции чтения/записи выполняются при низком логическом уровне сигналов на входах -RD, -WR и низком логическом уровне сигнала на входе выбора микросхемы -CS. Биты TRISE (TRISE<2:0>) должны быть установлены в '1' (выводы настроены на вход). В регистре ADCON1<3:0> выводы RE2:RE0 должны быть настроены как цифровые каналы ввода/вывода (биты PCFG3:PCFG0).

Фактически существуют два 8-разрядных регистра: один регистр для приема данных, другой - для передачи. Пользователь записывает 8-разрядные данные в выходную защелку PORTD, а читает данные со входной защелки (обратите внимание, выходная и входная защелка имеют один и тот же адрес). В этом режиме значение битов регистра TRISE игнорируется, т.к. направлением данных управляет внешнее устройство.

Запись в PSP происходит, если выводы -CS и -WR имеют низкий уровень сигнала. Чтение из PSP происходит, если выводы -CS и -RD имеют низкий уровень сигнала.

Для работы выводов PORTE в качестве управляющих входов ведомого параллельного порта нужно установить в ‘1’ бит PSPMODE(TRISE<4>). В этом режиме выводы должны работать как цифровые каналы ввода/вывода (регистр ADCON1) настроенные на вход (биты TRISE<2:0> должны быть установлены в ‘1’). Выводы ведомого параллельного порта имеют входные буферы ТТЛ.

Рисунок 9-10. Структурная схема выводов PORTD, PORTE в режиме ведомого параллельного порта



Примечание 1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Рисунок 9-11. Временная диаграмма записи в ведомый параллельный порт

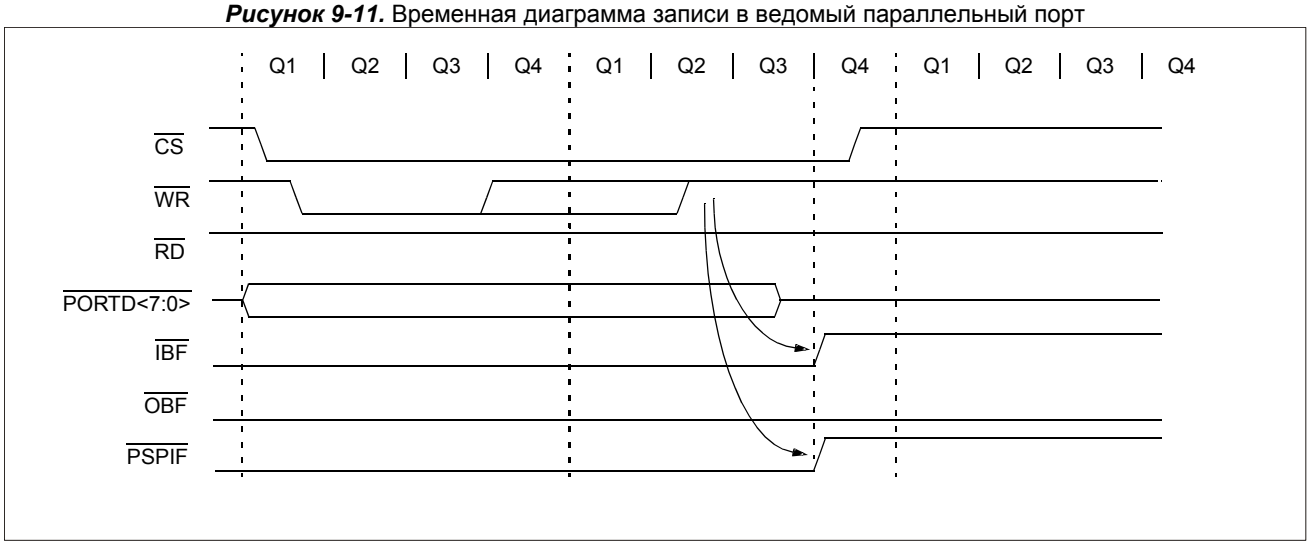


Рисунок 9-12. Временная диаграмма чтения ведомого параллельного порта

Таблица 9-11. Регистры и биты, связанные с работой ведомого параллельного порта

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
F83h	PORD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx
F8Dh	LATD	Регистр выходных данных								xxxx xxxx
F95h	TRISD	Регистр направления данных								1111 1111
F84h	PORE	-	-	-	-	-	RE2	RE1	RE0	---- -000
F8Dh	LATE	-	-	-	-	-	Регистр выходных данных			---- -xxx
F96h	TRISE	IBF	OBF	PSPMODE	-	-	Регистр направления данных			0000 -111
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FC1h	ADCON1	ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

10. Модуль таймера TMR0

Модуль таймера TMR0 имеет следующие особенности:

- Программный выбор режима работы – 8-разрядный или 16-разрядный таймер/счетчик
- Значение таймера доступно для записи и чтения
- Программируемый 8-разрядный предделитель
- Выбор источника тактового сигнала (внешний или внутренний)
- Генерация прерываний по переполнению от FFh к 00h в 8-разрядном режиме, от FFFFh к 0000h в 16-разрядном режиме
- Выбор активного фронта внешнего тактового сигнала

На рисунке 10-1 показана упрощенная структурная схема модуля TMR0 в 8-разрядном режиме, а на рисунке 10-2 – в 16-разрядном режиме.

В регистре T0CON расположены биты управления работой таймера TMR0. Регистр T0CON доступен для записи и чтения.

Регистр 10-1. Регистр управления таймером TMR0 T0CON

R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - 1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
Бит 7							Бит 0

Бит 7 **TMR0ON:** Бит разрешения работы TMR0
1 = таймер TMR0 включен
0 = таймер TMR0 выключен

Бит 6 **T08BIT:** Выбор режима работы таймера TMR0
1 = таймер TMR0 работает в режиме 8-разрядного таймера/счетчика
0 = таймер TMR0 работает в режиме 16-разрядного таймера/счетчика

Бит 5 **T0CS:** Выбор источника тактового сигнала для TMR0
1 = тактовый сигнал с вывода T0CKI
0 = внутренний тактовый сигнал (CLKOUT)

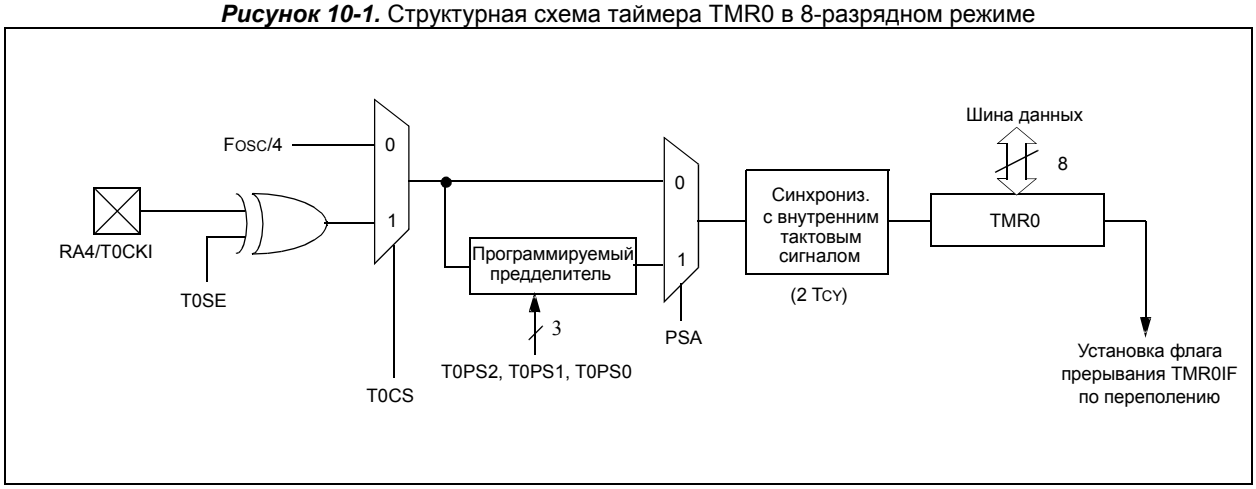
Бит 4 **T0SE:** Выбор активного фронта внешнего тактового сигнала
1 = приращения таймера TMR0 происходит по заднему фронту сигнала на выводе T0CKI
0 = приращения таймера TMR0 происходит по переднему фронту сигнала на выводе T0CKI

Бит 3 **PSA:** Подключение предделителя к таймеру TMR0
1 = таймер TMR0 работает без предделителя (используется тактовый сигнал с входа предделителя)
0 = таймер TMR0 работает с предделителем (используется тактовый сигнал с выхода предделителя)

Бит 2-1 **T0PS2:T0PS0:** Коэффициент деления предделителя TMR0
111 = 1:256
110 = 1:128
101 = 1:64
100 = 1:32
011 = 1:16
010 = 1:8
001 = 1:4
000 = 1:2

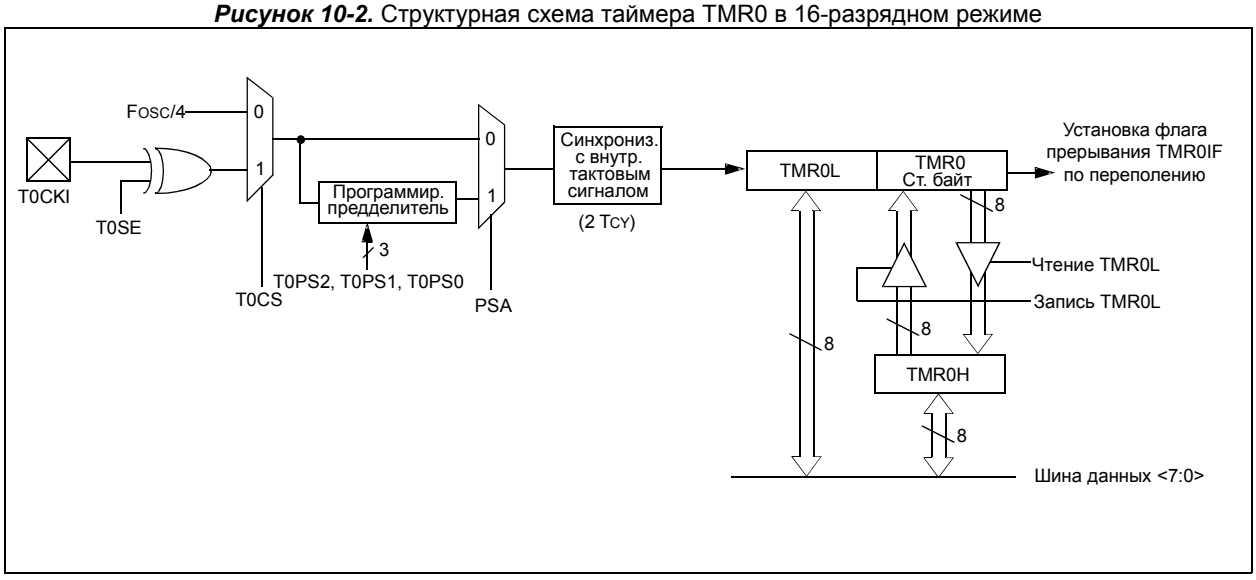
Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Рисунок 10-1. Структурная схема таймера TMR0 в 8-разрядном режиме



Примечание. После сброса микроконтроллера TMR0 работает в 8-разрядном режиме с внешним тактовым сигналом (вывод T0CKI) и максимальным коэффициентом деления предделителя.

Рисунок 10-2. Структурная схема таймера TMR0 в 16-разрядном режиме



Примечание. После сброса микроконтроллера TMR0 работает в 8-разрядном режиме с внешним тактовым сигналом (вывод T0CKI) и максимальным коэффициентом деления предделителя.

10.1 Работа таймера TMR0

Модуль TMR0 может работать в режиме таймера или счетчика.

Выбор режима таймера осуществляется сбросом бита T0CKI в ‘0’. В режиме таймера приращение TMR0 происходит на каждом машинном цикле микроконтроллера (если предделитель выключен). После записи в TMR0 приращение счетчика запрещено два следующих цикла. Пользователь должен скорректировать эту задержку перед записью нового значения в TMR0.

Если бит T0CS установлен в ‘1’, TMR0 работает в режиме счетчика с приращением от внешнего источника тактового сигнала на входе RA4/T0CKI. Активный фронт внешнего тактового сигнала выбирается битом T0SE Если T0SE=0, то активным является передний фронт сигнала). Основные требования к внешнему источнику тактового сигнала смотрите ниже по тексту.

При использовании внешнего тактового сигнала для TMR0 необходимо учитывать некоторые детали работы таймера. Активный фронт внешнего тактового сигнала синхронизируется с внутренней тактовой частотой микроконтроллера (F_{OSC}), из-за чего возникает задержка от получения активного фронта сигнала до приращения TMR0.

10.2 Предделитель

8-разрядный счетчик может работать как предделитель TMR0, он не доступен для записи и чтения.

Коэффициент деления предделителя определяется битами PSA и T0PS2:T0PS0.

Сброс бита PSA в ‘0’ отключает предделитель от таймера TMR0. Когда предделитель включен, то можно программно настроить его коэффициент деления от 1:2 до 1:256.

Если предделитель включен перед TMR0, любые команды записи в TMR0 (например, CLRF TMR0; MOVWF TMR0; BSF TMR0,x и т.д.) сбрасывают предделитель.

Примечание. Запись в TMR0 сбросит предделитель, если он включен, но коэффициент деления предделителя не изменится.

10.2.1 Переключение предделителя

Предделитель имеет программное управление (т.е. изменение коэффициента деления может быть произведено в течение выполнения программы).

10.3 Прерывание от TMR0

В 8-разрядном режиме таймера TMR0 при переполнении регистра TMR0 (переход от FFh к 00h) происходит установка флага прерываний TMR0IF. В 16-разрядном режиме флаг прерывания TMR0IF устанавливается в ‘1’, когда происходит переполнение сдвоенного регистра TMR0H:TMR0L (переход от FFFFh к 0000h). Прерывание может быть разрешено/запрещено битом TMR0IE. Бит TMR0IF должен быть программно сброшен в обработчике прерываний перед разрешением прерываний. Прерывание от TMR0 не может вывести микроконтроллер из режима SLEEP, т.к. модуль TMR0 в SLEEP режиме выключен.

10.4 Чтение и запись таймера в 16-разрядном режиме

Регистр TMR0H не является старшим байтом таймера/счетчика TMR0 в 16-разрядном режиме, он выполняет функции буфера (смотрите рисунок 10-2). Старший байт TMR0 не доступен для непосредственного чтения или записи. В TMR0H загружается старший байт TMR0 при чтении TMR0L. Это позволяет читать 16-разрядное значение полностью без необходимости проверки возможного переполнения младшего байта.

Запись старшего байта TMR0 должна выполняться через буферный регистр TMR0H. В старший байт TMR0 переписывается значение из TMR0H при записи в регистр TMR0L. Это позволяет сразу записывать 16-разрядное значение.

Таблица 10-1. Регистры и биты, связанные с работой модуля таймера TMR0

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FD7h	TMR0H	Регистр таймера 0 старший байт								0000 0000
FD6h	TMR0L	Регистр таймера 0 младший байт								xxxx xxxx
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FD5h	T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111
F92h	TRISA	-	Регистр направления данных							-111 1111

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

11. Модуль таймера TMR1

Модуль таймера TMR1 имеет следующие особенности:

- 16-разрядный таймер/счетчик (с двумя 8-разрядными регистрами TMR1H, TMR1L)
- Значение таймера доступно для записи и чтения (оба регистра)
- Выбор источника тактового сигнала (внешний или внутренний)
- Генерация прерываний по переполнению от FFFFh к 0000h
- Сброс таймера по сигналу триггера специального события модуля CCP

Структурная схема модуля таймера TMR1 показана на рисунке 11-1.

Управляющий регистр T1CON доступен для записи и чтения. Этот регистр содержит биты управления модулем таймера TMR1 и бит включения тактового генератора TMR1 таймера (T1OSCEN). Таймер TMR1 включается установкой в ‘1’ бита TMR1ON (T1CON<0>).

Регистр 11-1. Регистр управления таймером TMR1 T1CON

R/W - 0	U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
RD16	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON
Бит 7							Бит 0

Бит 7 **RD16:** Включение режима 16-разрядного чтения/записи
1 = чтение/запись регистров TMR1 выполняется за одну 16-разрядную операцию
0 = чтение/запись регистров TMR1 выполняется за две 8-разряднии операции

Бит 6 **Не используется:** Читается как ‘0’

Бит 5-4 **T1CKPS1:T1CKPS0:** Коэффициент деления предделителя TMR1
11 = 1:8
10 = 1:4
01 = 1:2
00 = 1:1

Бит 3 **T1OSCEN:** Включение тактового генератора TMR1
1 = генератор TMR1 включен
0 = генератор выключен
(инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления)

Бит 2 **-T1SYNC:** Синхронизация внешнего тактового сигнала
TMR1CS = 1
1 = не синхронизировать внешний тактовый сигнал
0 = синхронизировать внешний тактовый сигнал

TMR1CS = 0
Значение бита игнорируется. Используется внутренний тактовый сигнал.

Бит 1 **TMR1CS:** Выбор источника тактового сигнала
1 = внешний источник тактового сигнала с вывода RC0/T1OSO/T1CKI
(активным является передний фронт сигнала)
0 = внутренний тактовый сигнал Fosc/4

Бит 0 **TMR1ON:** Бит разрешения работы TMR1
1 = таймер TMR1 включен
0 = таймер TMR1 выключен

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

11.1 Работа таймера TMR1

Модуль таймера TMR1 может работать в одном из трех режимов:

- Таймер
- Синхронный счетчик
- Асинхронный счетчик

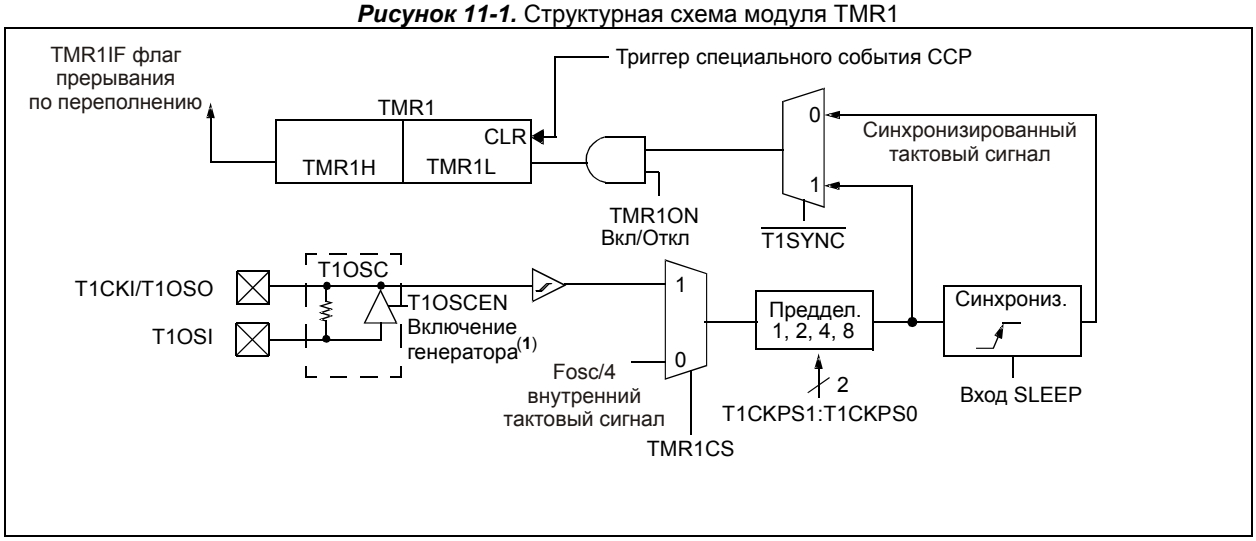
Режим работы определяется битом выбора источника тактового сигнала TMR1CS (T1CON<1>).

Если TMR1CS=0, то значение таймера TMR1 инкрементируется на каждом машинном цикле (если коэффициент предделителя 1:1). Когда TMR1CS=1, приращение происходит по каждому переднему фронту внешнего тактового сигнала или сигнала генератора TMR1 (если он включен).

Когда включен генератор тактовых импульсов (T1OSCEN=1), выводы RC1/T1OSI и RC0/T1OSO/T1CKI настроены как входы. Значение битов TRISC<1:0> игнорируется, а чтение данных с этих выводов дает результат '0'.

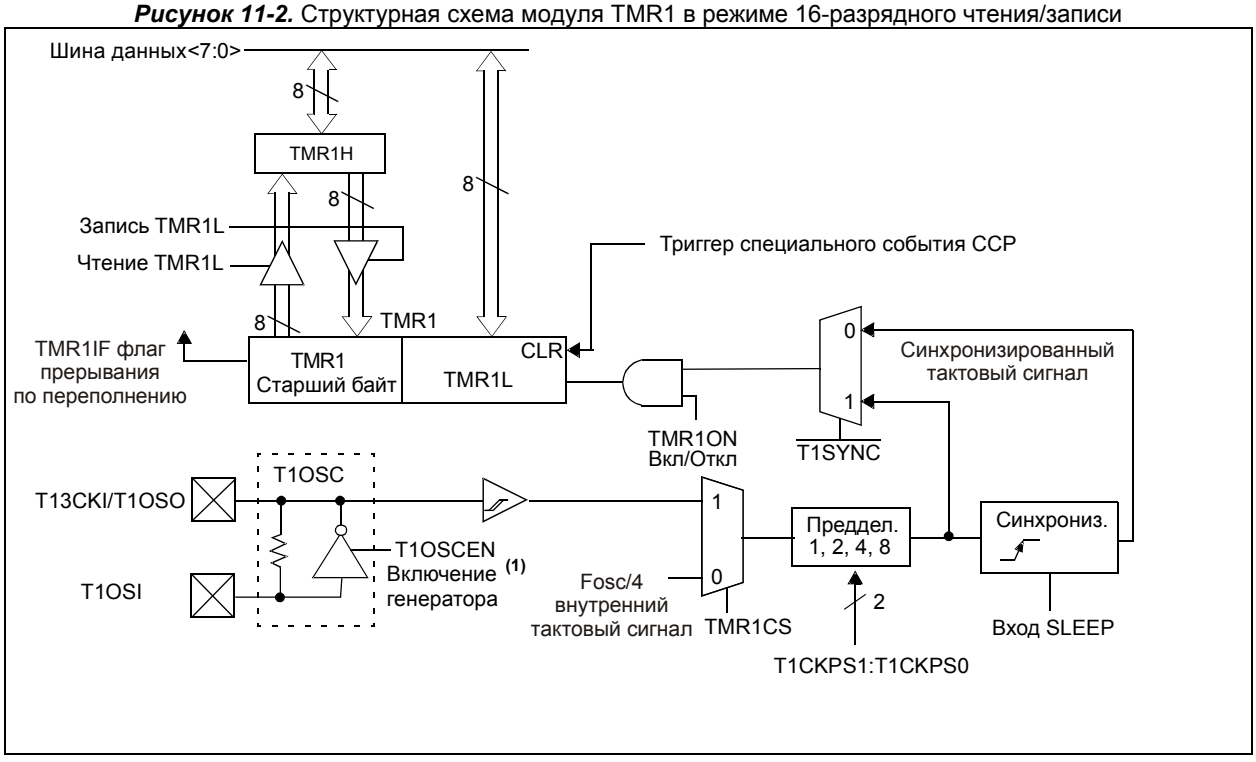
Модуль TMR1 имеет вход внутреннего сброса от CCP модуля (смотрите раздел 14).

Рисунок 11-1. Структурная схема модуля TMR1



Примечание. Если T1OSCEN=0, то инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления.

Рисунок 11-2. Структурная схема модуля TMR1 в режиме 16-разрядного чтения/записи



Примечание. Если T1OSCEN=0, то инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления.

11.2 Генератор TMR1

Кварцевый резонатор подключается к выводам T1OSI (вход) и T1OSO (выход усилителя). Включение генератора производится установкой бита T1OSEN (T1CON<3>). Максимальная частота резонатора 200КГц. Генератор позволяет работать TMR1 в SLEEP режиме микроконтроллера. Тактовый генератор TMR1 в основном предназначен для кварцевого резонатора 32кГц. В таблицы 11-1 указаны рекомендуемые значения конденсаторов для генератора TMR1.

Пользователь должен обеспечить программную задержку, чтобы гарантировать надлежащий запуск генератора.

Таблица 11-1. Выбор конденсаторов для генератора TMR1

Тип генератора	Частота	C1	C2
LP	32 кГц	TBD(1)	TBD(1)

Протестированные резонаторы:		
32.768кГц	Epson C-001 R32.768K-A	±20 PPM

Примечания:

1. При подборе емкости конденсаторов Microchip рекомендует конденсаторы 33пФ как отправную точку.
2. Большая емкость увеличивает стабильность генератора, но также увеличивает время запуска.
3. Каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного подбора внешних компонентов.
4. Указанная емкость конденсаторов является оценочной.

11.3 Прерывания от TMR1

Пара регистров TMR1 (TMR1H:TMR1L) инкрементируются от 0000h до FFFFh и переполняется к 0000h. Прерывание от TMR1, если разрешено, происходит при переполнении TMR1, устанавливая флаг TMR1IF (PIR1<0>). Прерывание от TMR1 можно разрешить/запретить установкой/сбросом бита TMR1IE(PIE1<0>).

11.4 Сброс TMR1 триггером модуля CCP

Если модуль CCP работает в режиме сравнения с триггером специального события (CCP1M3 : CCP1M0=1011), то сигнал триггера сбросит TMR1 и запустит преобразование АЦП (если АЦП включено).

Примечание. Сигнал с триггера специального события модуля CCP1 не будет устанавливать флаг прерывания TMR1IF (PIR1<0>) в ‘1’.

TMR1 должен работать в режиме синхронизированного внешнего тактового сигнала или внутреннего тактового сигнала. В асинхронном режиме функция сброса не работает.

Когда запись в TMR1 совпадает с сигналом сброса от триггера специальных событий, приоритет отдается записи в TMR1.

В этом режиме модуля CCP период сброса TMR1 сохраняется в регистрах CCPR1H:CCPR1L.

11.5 Чтение и запись таймера в 16-разрядном режиме

TMR1 может быть настроен для работы в режиме 16-разрядного чтения/записи (смотрите рисунок 11-2). Когда бит RD16(T1CON<7>) установлен в ‘1’, обращение по адресу TMR1H вызовет действие с буферным регистром. При чтении TMR1L значение старшего байта TMR1 будет загружено в буфер. Это позволяет читать 16-разрядное значение полностью без необходимости проверки возможного переполнения младшего байта.

Запись старшего байта TMR1 должна выполняться через буферный регистр TMR1H. В старший байт TMR1 переписывается значение из TMR1H при записи в регистр TMR1L. Это позволяет сразу записывать 16-разрядное значение.

В этом режиме старший байт TMR1 не доступен для непосредственного чтения или записи. Любая запись или чтение должно выполняться через буферный регистр TMR1. Запись в TMR1H не сбрасывает предделитель. Предделитель сбрасывается только при записи в TMR1L.

Таблица 11-2. Регистры и биты, связанные с работой модуля таймера TMR1

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FCFh	TMR1H	Регистр таймера 1 старший байт								xxxx xxxx
FCEh	TMR1L	Регистр таймера 1 младший байт								xxxx xxxx
FCDh	T1CON	RD16	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON	0-00 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

12. Модуль таймера TMR2

Модуль таймера TMR2 имеет следующие особенности:

- 8-разрядный таймер (регистр TMR2)
- 8-разрядный регистр периода (PR2)
- Регистры таймера доступны для записи и чтения
- Программируемый предделитель (1:1, 1:2, 1:16)
- Программируемый постделитель (1:1 – 1:16)
- Прерывания от TMR2 по достижению значения в PR2
- TMR2 может использоваться для генерации тактового сигнала модуля SSP

Биты управления таймером TMR2 находятся в регистре T2CON. TMR2 может быть выключен сбросом бита TMR2ON(T2CON<2>) для уменьшения энергопотребления. На рисунке 12-2 представлена структурная схема таймера TMR2. Значение коэффициентов предделителя и постделителя может быть выбрано в регистре T2CON.

12.1 Работа таймера TMR2

TMR2 может быть опорным таймером для CCP модуля в ШИМ режиме. Регистры TMR2 доступны для записи/чтения и очищаются при любом виде сброса. Входной тактовый сигнал ($F_{OSC}/4$) поступает через предделитель с программируемым коэффициентом деления (1:1, 1:4 или 1:16), определяемый битами T2CKPS1:T2CKPS0 (T2CON<1:0>). Сигнал переполнения TMR2 проходит через выходной 4-разрядный делитель с программируемым коэффициентом деления (от 1:1 до 1:16 включительно) для установки флага TMR2IF в регистре PIR1<1>.

Счетчик предделителя и выходного делителя сбрасываются в случае:

- Записи в регистр TMR2
- Записи в регистр T2CON
- Любого вида сброса микроконтроллера (POR, BOR, сброс WDT или активный сигнал -MCLR)

Регистр TMR2 не очищается при записи в T2CON.

Регистр 12-1. Регистр управления таймером TMR2 T2CON

U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
Бит 7							Бит 0

Бит 7 **Не используется:** Читается как ‘0’

Бит 6-3 **TOUTPS3: TOUTPS0:** Коэффициент деления выходного делителя TMR2
0000 = 1:1
0001 = 1:2
:
1111 = 1:16

Бит 2 **TMR2ON:** Бит разрешения работы TMR2
1 = таймер TMR2 включен
0 = таймер TMR2 выключен

Бит 1-0 **T2CKPS1:T2CKPS0:** Коэффициент деления предделителя TMR2
00 = 1:1
01 = 1:4
1x = 1:16

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

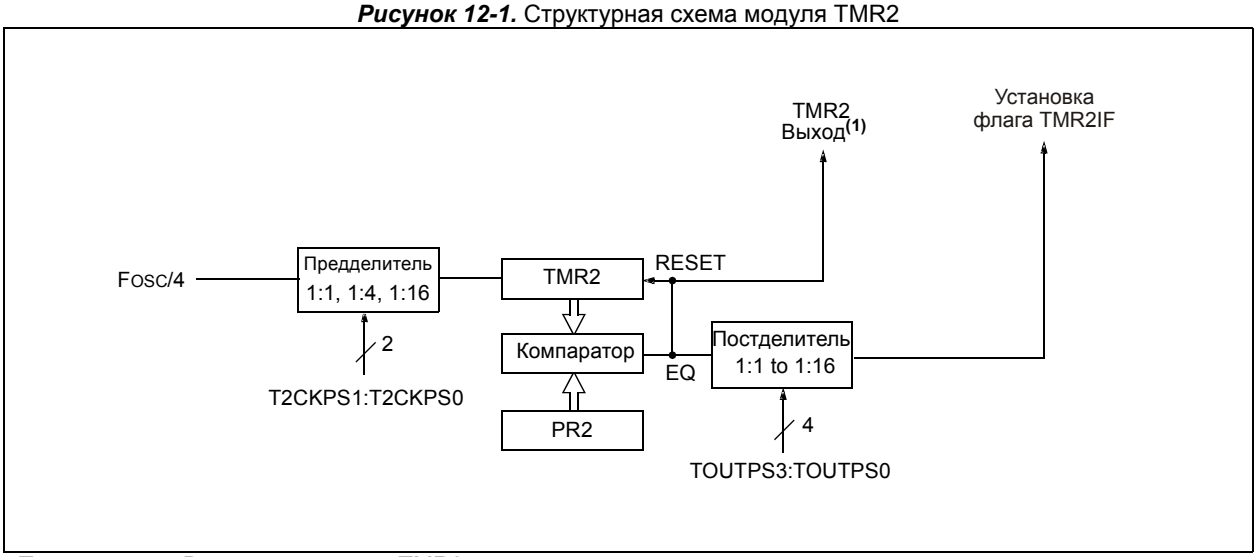
12.2 Прерывания от TMR2

TMR2 имеет 8-разрядный регистр периода PR2. TMR2 считает, инкрементируя от 00h до значения в регистре PR2, затем сбрасывается в 00h на следующем машинном цикле. Регистр PR2 доступен для записи и чтения. После сброса значение регистра PR2 равно FFh.

12.3 Выход TMR2

Сигнал переполнения TMR2 (до выходного предделителя) поступает в модуль MSSP для управления скоростью передачи данных.

Рисунок 12-1. Структурная схема модуля TMR2



Примечание. Выходной сигнал TMR2 может использоваться для программной настройки скорости передачи данных модуля MSSP.

Таблица 12-1. Регистры и биты, связанные с работой модуля таймера TMR2

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FCCh	TMR2	Регистр таймера 2								0000 0000
FCAh	T2CON	-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000
FCBh	PR2	Регистр периода таймера 2								1111 1111

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

13. Модуль таймера TMR3

Модуль таймера TMR3 имеет следующие особенности:

- 16-разрядный таймер/счетчик (с двумя 8-разрядными регистрами TMR3H, TMR3L)
- Значение таймера доступно для записи и чтения (оба регистра)
- Выбор источника тактового сигнала (внешний или внутренний)
- Генерация прерываний по переполнению от FFFFh к 0000h
- Сброс таймера по сигналу триггера специального события модуля CCP

Структурная схема модуля таймера TMR3 показана на рисунке 13-1.

Управляющий регистр T3CON доступен для записи и чтения. Этот регистр содержит биты управления таймера TMR3 и выбора тактового сигнала для модуля CCP.

Смотрите описание регистра T1CON. Этот регистр содержит биты управления TMR1 и бит включения тактового генератора TMR1 (T1OSCEN), который может использоваться для работы TMR3.

Регистр 13-1. Регистр управления таймером TMR3 T3CON

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	-T3SYNC	TMR3CS	TMR3ON
Бит 7							Бит 0

Бит 7 **RD16:** Включение режима 16-разрядного чтения/записи
1 = чтение/запись регистров TMR3 выполняется за одну 16-разрядную операцию
0 = чтение/запись регистров TMR3 выполняется за две 8-разряднии операции

Бит 6,3 **T3CCP2:T3CCP1:** Выбор источника тактового сигнала для работы модуля CCP
1x = TMR3 используется для работы CCP модулей в режиме сравнение/захват
01 = TMR3 используется для работы CCP2 модуля в режиме сравнение/захват
TMR1 используется для работы CCP1 модуля в режиме сравнение/захват
00 = TMR1 используется для работы CCP модулей в режиме сравнение/захват

Бит 5-4 **T3CKPS1:T3CKPS0:** Коэффициент деления предделителя TMR3
11 = 1:8
10 = 1:4
01 = 1:2
00 = 1:1

Бит 2 **-T3SYNC:** Синхронизация внешнего тактового сигнала
TMR3CS = 1
1 = не синхронизировать внешний тактовый сигнал
0 = синхронизировать внешний тактовый сигнал

TMR3CS = 0
Значение бита игнорируется. Используется внутренний тактовый сигнал.

Бит 1 **TMR3CS:** Выбор источника тактового сигнала
1 = внешний источник тактового сигнала с вывода RC0/T1OSO/T1CKI
(активным является передний фронт сигнала)
0 = внутренний тактовый сигнал Fosc/4

Бит 0 **TMR3ON:** Бит разрешения работы TMR3
1 = таймер TMR3 включен
0 = таймер TMR3 выключен

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

13.1 Работа таймера TMR3

Модуль таймера TMR3 может работать в одном из трех режимов:

- Таймер
- Синхронный счетчик
- Асинхронный счетчик

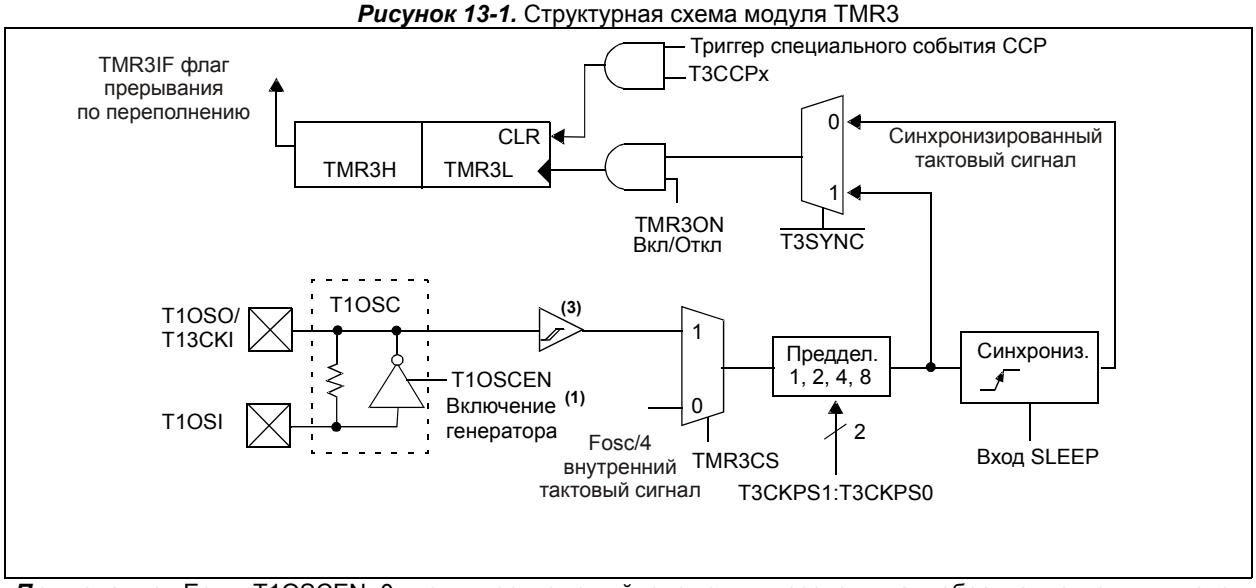
Режим работы определяется битом выбора источника тактового сигнала TMR3CS (T3CON<1>).

Если TMR3CS=0, то значение таймера TMR3 инкрементируется на каждом машинном цикле (если коэффициент предделителя 1:1). Когда TMR3CS=1, приращение происходит по каждому переднему фронту внешнего тактового сигнала или сигнала генератора TMR1 (если он включен).

Когда включен генератор тактовых импульсов (T1OSCEN=1), выводы RC1/T1OSI и RC0/T1OSO/T1CKI настроены как входы. Значение битов TRISC<1:0> игнорируется, а чтение данных с этих выводов дает результат '0'.

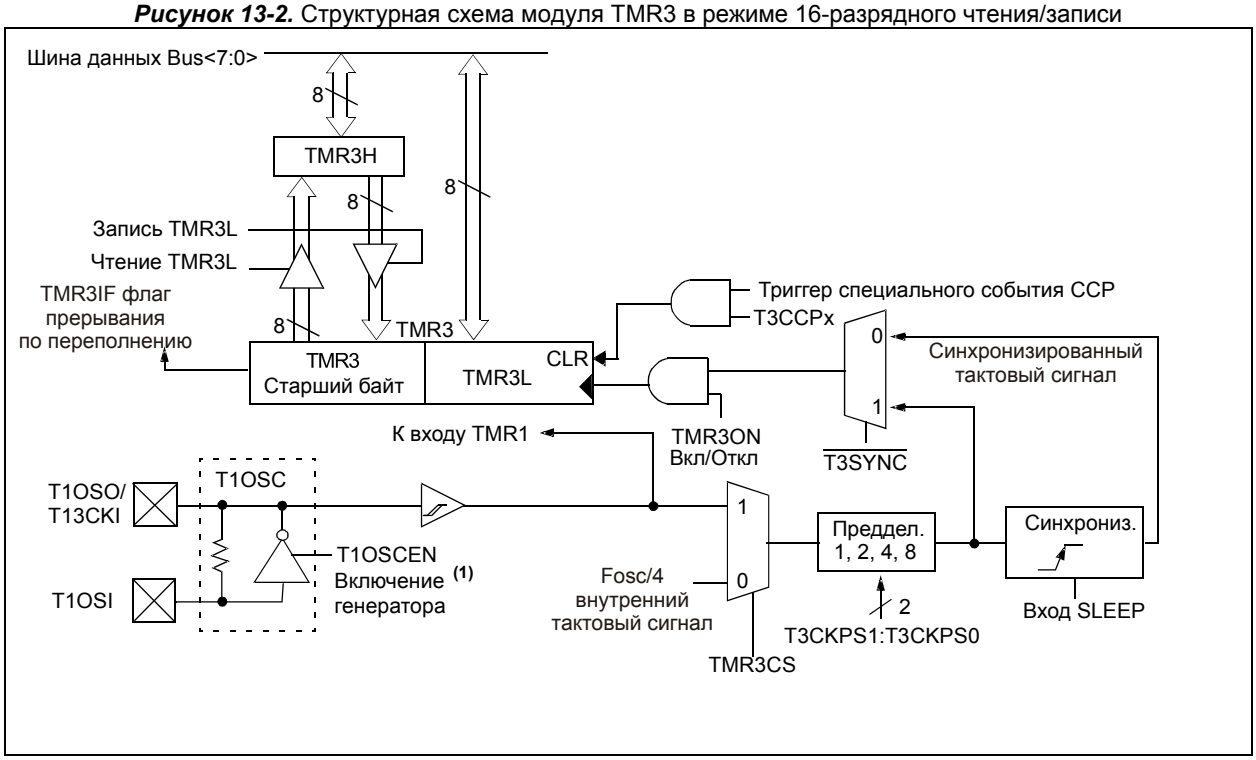
Модуль TMR3 имеет вход внутреннего сброса от CCP модуля (смотрите раздел 14).

Рисунок 13-1. Структурная схема модуля TMR3



Примечание. Если T1OSCEN=0, то инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления.

Рисунок 13-2. Структурная схема модуля TMR3 в режиме 16-разрядного чтения/записи



Примечание. Если T1OSCEN=0, то инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления.

13.2 Генератор TMR1

Кварцевый резонатор подключается к выводам T1OSI (вход) и T1OSO (выход усилителя). Включение генератора производится установкой бита T1OSEN (T1CON<3>). Максимальная частота резонатора 200КГц. Подробное описание смотрите в разделе 11.0.

13.3 Прерывания от TMR3

Пара регистров TMR3 (TMR3H:TMR3L) инкрементируются от 0000h до FFFFh и переполняется к 0000h. Прерывание от TMR3, если разрешено, происходит при переполнении TMR3, устанавливая флаг TMR3IF (PIR2<1>). Прерывание от TMR3 можно разрешить/запретить установкой/сбросом бита TMR3IE(PIE2<1>).

13.4 Сброс TMR3 триггером модуля CCP

Если модуль CCP работает в режиме сравнения с триггером специального события (CCP1M3 : CCP1M0=1011), то сигнал триггера сбросит TMR3.

Примечание. Сигнал с триггера специального события модуля CCP1 не будет устанавливать флаг прерывания TMR3IF (PIR2<1>) в ‘1’.

TMR3 должен работать в режиме синхронизированного внешнего тактового сигнала или внутреннего тактового сигнала. В асинхронном режиме функция сброса не работает. Когда запись в TMR3 совпадает с сигналом сброса от триггера специальных событий, приоритет отдается записи в TMR3. В этом режиме модуля CCP период сброса TMR1 сохраняется в регистрах CCPR1H:CCPR1L.

Таблица 13-1. Регистры и биты, связанные с работой модуля таймера TMR3

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FA2h	IRP2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
FA1h	PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
FA0h	PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000
FB3h	TMR3H	Регистр таймера 3 старший байт								xxxx xxxx
FB2h	TMR3L	Регистр таймера 3 младший байт								xxxx xxxx
FCDh	T1CON	RD16	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON	0-00 0000
FB1h	T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	-T3SYNC	TMR3CS	TMR3ON	0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

14. CCP модуль (Захват/Сравнение/ШИМ)

Каждый модуль CCP содержит 16-разрядный регистр, который может использоваться в качестве:
- 16-разрядного регистра захвата данных;
- 16-разрядного регистра сравнения;
- Двух 8-разрядных (ведущий и ведомый) регистров ШИМ.

Работа модулей CCP1 и CCP2 идентична, за исключение функционирования триггера специального события. В таблице 14-1 и 14-2 указаны ресурсы, используемые модулем CCP. Далее будет описана работа модуля CCP1. Модуль CCP2 работает аналогично, отличия будут указаны отдельно.

Регистр 14-1. Регистры управления модулем CCP CCP1CON и CCP2CON

U - 0	U - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
-	-	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
Бит 7							Бит 0

Бит 7,6 **Не используется:** Читается как ‘0’

Бит 5,4 **DCxB1:DCxB0:** Биты 1 и 0 длительности импульса в ШИМ режиме
Режим захвата
Не используются

Режим сравнения
Не используются

Режим ШИМ
Младшие биты (1 и 0) 10-разрядной длительности импульса ШИМ. Старшие 8 битов (DCx9:DCx2) размещаются в регистре CCPRxL

Бит 3-0 **CCPxM3:CCPxM0:** Режим работы CCPx модуля
0000 = модуль CCPx выключен (сброс модуля CCPx)
0001 = резерв
0010 = сравнение, переключение уровня при совпадении (устанавливается флаг CCPxIF в ‘1’)
0011 = резерв
0100 = захвата по каждому заднему фронту сигнала
0101 = захват по каждому переднему фронту сигнала
0110 = захват по каждому 4-му переднему фронту сигнала
0111 = захват по каждому 16-му переднему фронту сигнала
1000 = сравнение, устанавливает выходной сигнал (устанавливается флаг CCPxIF в ‘1’)
1001 = сравнение, сбрасывает выходной сигнал (устанавливается флаг CCPxIF в ‘1’)
1010 = сравнение, на выходной сигнал не влияет (устанавливается флаг CCPxIF в ‘1’)
1011 = сравнение, триггер специальных функций (устанавливается флаг CCPxIF в ‘1’)
11xx = ШИМ режим

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

14.1 Модуль CCP1

Регистр CCPR1 модуля CCP состоит из двух 8-разрядных регистров: CCPR1L (младший байт), CCPR1H (старший байт). В регистре CCP1CON находятся управляющие биты модуля CCP1, доступные для записи и чтения.

14.2 Модуль CCP2

Регистр CCPR2 модуля CCP состоит из двух 8-разрядных регистров: CCPR2L (младший байт), CCPR2H (старший байт). В регистре CCP2CON находятся управляющие биты модуля CCP2, доступные для записи и чтения.

Таблица 14-1 Использование таймеров модулями CCP

Режим модуля CCP	Таймер
Захват	TMR1 или TMR3
Сравнение	TMR1 или TMR3
ШИМ	TMR2

Таблица 14-2 Взаимодействие двух модулей CCP

Режим CCPx	Режим CCPy	Взаимодействие
Захват	Захват	Базовый таймер TMR1 или TMR3. Каждому CCP модулю может быть назначен свой базовый таймер.
Захват	Сравнение	Модуль CCP, работающий в режиме сравнения, должен сбрасывать таймер TMR1 или TMR3 триггером специального события.
Сравнение	Сравнение	Модули CCP, работающие в режиме сравнения, должны сбрасывать таймер TMR1 или TMR3 триггером специального события.
ШИМ	ШИМ	Оба ШИМ имеют одинаковую частоту и фазу (базовый таймер TMR2)
ШИМ	Захват	Нет
ШИМ	Сравнение	Нет

14.3 Режим захвата

При возникновении события захвата 16-разрядное значение счетчика TMR1 или TMR3 переписывается в регистры CCPR1L:CCPR1H модуля CPP1. Событием захвата может быть:

- Каждый задний фронт сигнала на входе RC2/CCP1
- Каждый передний фронт сигнала на входе RC2/CCP1
- Каждый 4-й передний фронт сигнала на входе RC2/CCP1
- Каждый 16-й передний фронт сигнала на входе RC2/CCP1

Тип события захвата устанавливается битами CCP1M3:CCP1M0 в регистре CCP1CON. После выполнения захвата устанавливается флаг прерывания CCP1F (PIR1<2>) в ‘1’, который должен быть сброшен программно. Если происходит событие захвата до того как предыдущие данные были прочитаны, старое значение будет потеряно.

14.3.1 Настройка вывода модуля CCP

Порт ввода/вывода RC2/CCP1 должен быть настроен на вход установкой бита TRISC<2> в’1’.

Примечание. Если порт ввода/вывода RC2/CCP1 настроен на выход, то захват может происходить командой из программы.

14.3.2 Настройка таймера TMR1/TMR3

В случае использования внешнего тактового сигнала с вывода RC1/T1OSI/CCP2 таймер TMR1/TMR3 должен работать в синхронизированном режиме. В асинхронном режиме TMR1/TMR3 модуль CCP1 работать не будет. Выбор базового таймера для CCP выполняется в регистре T3CON.

14.3.3 Обработка прерываний

Когда изменяется режим работы модуля CCP, необходимо запрещать прерывания сбросом бита CCP1IE (PIE<2>) в ‘0’ для предотвращения ложных прерываний. После изменение режима работы модуля CCP1, перед разрешением прерываний, необходимо сбросить флаг CCP1IF (PIR1<2>) в ‘0’.

14.3.4 Предварительный счетчик событий модуля CCP

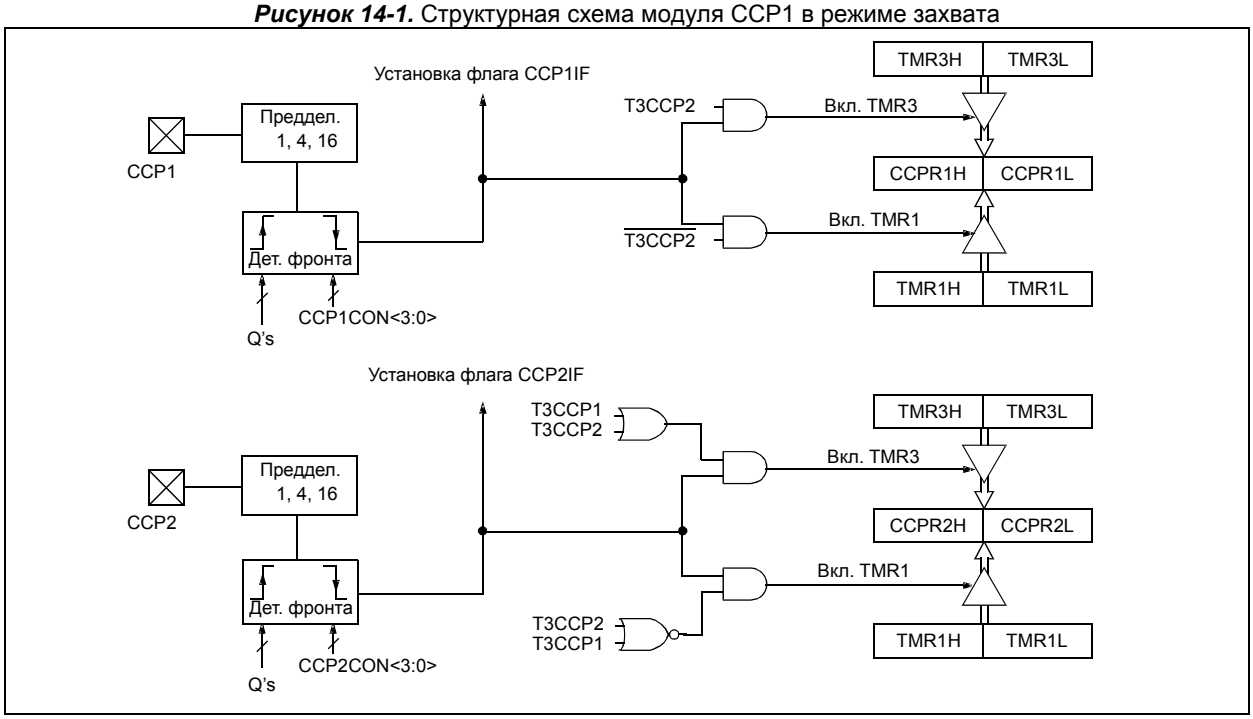
Существует четыре режима работы предварительного счетчика событий (определяется битами CCP1M3:CCP1M0). Включение режима захвата очищает предварительный счетчик событий. Переключение между типами событий не очищает счетчик событий, поэтому результат первого захвата после переключения может быть недостоверным. Любой сброс микроконтроллера очищает счетчик событий.

В примере 14-1 показано как нужно производить переключение типа события, чтобы не вызвать ложное прерывание.

Пример 14-1 Переключение типа события

```
CLRF    CCP1CON, F          ; Выключить CCP модуль
MOVLW   NEW_CAPT_PS         ; Записать W новый тип захвата и режим работы CCP
MOVWF   CCP1CON             ; Загрузить настройку в регистр CCP1CON
```

Рисунок 14-1. Структурная схема модуля CCP1 в режиме захвата



14.4 Режим сравнения

В этом режиме 16-разрядный регистр CCPR1 сравнивается со значением TMR1 или TMR3. Как только значения в регистрах становятся одинаковые, модуль CCP1 изменяет состояние вывода RC2/CCP1:

- Устанавливает высокий уровень сигнала
- Устанавливает низкий уровень сигнала
- На вывод не воздействует

Действие при совпадении может быть выбрано битами CCP1M3:CCP1M0 в регистре CCP1CON. В момент изменения состояния вывода устанавливается флаг прерывания CCP1IF в ‘1’.

14.4.1 Настройка вывода модуля CCP

Для изменения состояния вывода RC2/CCP1, он должен быть настроен на выход сбросом бита TRISC<2> в ‘0’.

Примечание. При очистке регистра CCP1CON на выводе RC2/CCP1 появится сигнал низкого уровня, что не является результатом сравнения или данными из выходной защелки PORTC.

14.4.2 Настройка таймера TMR1/TMR3

В случае использования внешнего тактового сигнала с вывода RC1/T1OSI/CCP2 таймер TMR1/TMR3 должен работать в синхронизированном режиме. В асинхронном режиме TMR1/TMR3 модуль CCP1 работать не будет. Выбор базового таймера для CCP выполняется в регистре T3CON.

14.4.3 Обработка прерываний

Программное изменение уровня сигнала на выходе CCP1 не вызовет генерацию прерывания. Прерывание генерируются только модулем CCP1.

14.4.4 Триггер специального события

В режиме сравнения модуля CCP1 может быть включен триггер специального события.

Триггер специального события CCP1 сбрасывает значения таймера TMR1 или TMR3 при каждом положительно выполненном сравнении. Регистр CCP1R является 16-разрядным программируемым регистром периода для TMR1.

Триггер специального события CCP2 сбрасывает значения таймера TMR1 или TMR3 и запускает преобразование АЦП (если модуль АЦП включен).

Примечание. Триггер специального события модулей CCP1 и CCP2 не устанавливает флаг прерывания TMR1IF (PIR1<0>) в '1'.

Рисунок 14-2. Структурная схема модуля CCP1 в режиме сравнения

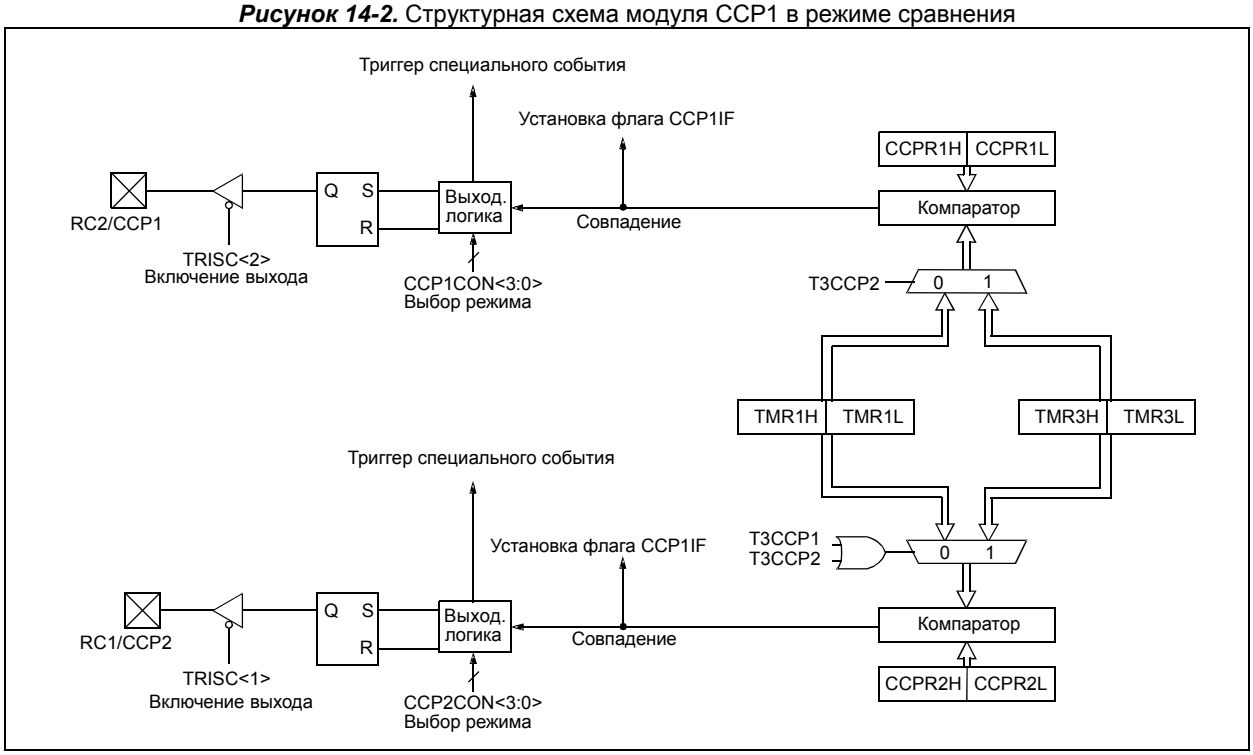


Таблица 14-3. Регистры и биты, связанные с работой модуля CCP в режиме захват/сравнение и TMR1, TMR3

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
F94h	TRISC	Регистр направления данных								1111 1111
FCFh	TMR1H	Регистр таймера 1 старший байт								xxxx xxxx
FCEh	TMR1L	Регистр таймера 1 младший байт								xxxx xxxx
FCDh	T1CON	RD16	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON	0-00 0000
FBFh	CCPR1H	Регистр 1 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
FBEh	CCPR1L	Регистр 1 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
FBDh	CCP1CON	-	-	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000
FBCh	CCPR2H	Регистр 2 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
FBBh	CCPR2L	Регистр 2 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
FBAh	CCP2CON	-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000
FA2h	IRP2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
FA1h	PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
FA0h	PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000
FB3h	TMR3H	Регистр таймера 3 старший байт								xxxx xxxx
FB2h	TMR3L	Регистр таймера 3 младший байт								xxxx xxxx
FB1h	T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	-T3SYNC	TMR3CS	TMR3ON	0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

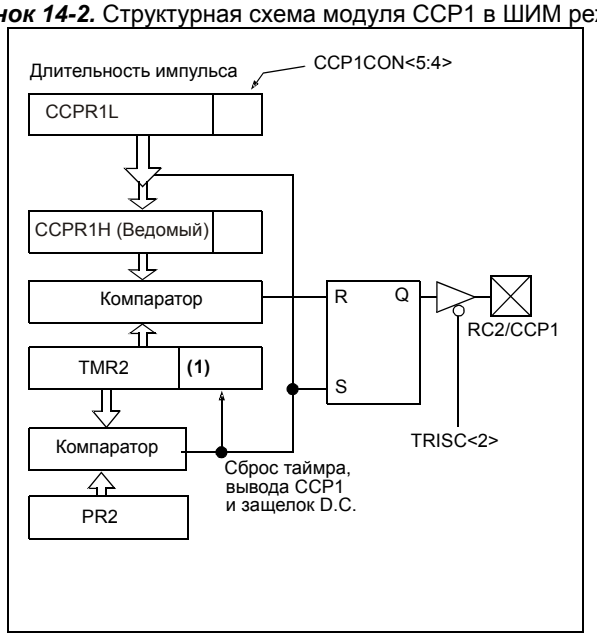
14.5 Режим ШИМ

В ШИМ режиме модуля CCP1 вывод CCP1 используется в качестве выхода 10-разрядного ШИМ. Т.к. вывод CCP1 мультиплицирован с цифровым каналом порта ввода/вывода, бит направления TRISC<2> должен быть сброшен в ‘0’ для настройки его как выход.

Примечание. Очистка регистра CCP1CON вынудит перевести вывод CCP1 в низкий логический уровень. Низкий логический уровень не является данными из защелки PORTC.

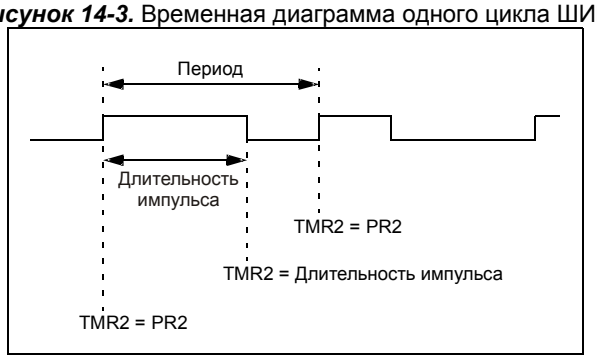
На рисунке 14-3 показана структурная схема модуля CCP1 в ШИМ режиме. Пошаговое описание настройки модуля CCP1 в ШИМ режиме смотрите в разделе 14.5.3.

Рисунок 14-2. Структурная схема модуля CCP1 в ШИМ режиме



На рисунке 14-4 показана временная диаграмма одного цикла ШИМ (период ШИМ и длительность высокого уровня сигнала). Частота ШИМ есть обратная величина периоду (1/период).

Рисунок 14-3. Временная диаграмма одного цикла ШИМ



14.5.1 Период ШИМ

Период ШИМ определяется значением в регистре PR2 и может быть вычислен по формуле:

Период ШИМ = [(PR2) + 1] x 4 x Tosc x (коэффициент предделителя TMR2)
Частота ШИМ = 1 / Период ШИМ

Когда значение TMR2 сравнивается с PR2, выполняются следующие действия:

- TMR2 сбрасывается в 00h
- Устанавливается высокий уровень сигнал на выводе CCP1 (Если скважность равна 0%, то сигнал в высокий уровень устанавливаться не будет)
- Модуль ШИМ начинает новый цикл, загружая значение из регистра CCPR1L в CCPR1H

Примечание. Выходной делитель TMR2 (см. раздел 12.0) не влияет на частоту ШИМ. Он может использоваться для отсчета времени, когда необходимо изменить скважность ШИМ.

14.5.2 Длительность импульса ШИМ

Длительность импульса ШИМ определяется битами в регистрах CCPR1L и CCP1CON<5:4>. Для 10-разрядного ШИМ старшие восемь бит сохраняются в регистре CCPR1L, а младшие два бита в регистре CCPCON<5:4> (CCPR1L:CCPCON<5:4>). Для вычисления длительности сигнала высокого уровня, воспользуйтесь следующей формулой:

Длительность импульса ШИМ = (CCPR1L:CCPCON<5:4>) x Tosc x (коэффициент предделителя TMR2)

Биты в регистре CCPR1L и CCP1CON<5:4> могут быть изменены в любое время, но значение в регистре CCPR1H не изменяется, пока не произойдет соответствие PR2 и TMR2. В ШИМ режиме регистр CCPR1H доступен только для чтения.

Регистр CCPR1H и внутренняя двух разрядная защелка образуют буфер ШИМ. Эффект буферизации необходим при записи нового значения длительности импульса ШИМ.

Когда значение CCPR1H и 2-разрядной внутренней защелки соответствует значению TMR2 и внутреннему 2-разрядному счетчику, в такте Q2 на выводе CCP1 будет установлен низкий уровень сигнала.

Расчет максимального разрядности ШИМ для данной частоты можно вычислить по формуле (бит):

$$= \frac{\log(\frac{Fosc}{Fpwm})}{\log(2)}$$

Примечание. Если длительность импульса ШИМ больше периода ШИМ, вывод CCP1 не будет иметь низкий уровень сигнала.

14.5.3 Последовательность настройки модуля CCP в ШИМ режиме

Рекомендованная последовательность включения модуля CCP в ШИМ режиме:

1. Установить период ШИМ в регистре PR2;
2. Установить длительность импульса в регистрах CCPR1L и CCP1CON <5:4>;
3. Настроить вывод CCP1 как выход, сбросив бит TRISC<2>;
4. Настроить предделитель и включить TMR2 в регистре T2CON;
5. Включить CCP1 в режиме ШИМ.

Таблица 14-4 Соответствие частоты ШИМ и разрядности ШИМ при тактовой частоте микроконтроллера 40МГц

Частота ШИМ	**2.44кГц**	**9.77кГц**	**36.09кГц**	**156.25кГц**	**312.50кГц**	**416.67кГц**
Коэффициент предделителя TMR2	16	4	1	1	1	1
Значение PR2	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Макс. разрядность ШИМ (бит)	14	12	10	8	7	6.58

Таблица 14-5. Регистры и биты, связанные с работой модуля CCP в режиме ШИМ и TMR2

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
F94h	TRISC	Регистр направления данных								1111 1111
FCCh	TMR2	Регистр таймера 2								0000 0000
FCAh	T2CON	-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000
FCBh	PR2	Регистр периода таймера 2								1111 1111
FBFh	CCPR1H	Регистр 1 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
FBEh	CCPR1L	Регистр 1 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
FBDh	CCP1CON	-	-	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000
FBCh	CCPR2H	Регистр 2 Захват/Сравнение/ШИМ старший байт								xxxx xxxx
FBBh	CCPR2L	Регистр 2 Захват/Сравнение/ШИМ младший байт								xxxx xxxx
FBAh	CCP2CON	-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

15. Модуль MSSP

15.1 Введение

Модуль ведущего синхронного последовательного порта (MSSP) может использоваться для связи с периферийными микросхемами или другими микроконтроллерами. Периферийными микросхемами могут быть: EEPROM память, сдвиговые регистры, драйверы ЖКИ, АЦП и др. Модуль MSSP может работать в одном из двух режимах:

- Последовательный периферийный интерфейс (SPI)
- Inter-Integrated Circuit (I^2C):

- ведущий режим
- ведомой режим (с поддержкой адреса общего вызова)

Для работы по интерфейсу I^2C аппаратно поддерживаются следующие режимы:

- Режим ведущего
- Режим ведущего с конкуренцией на шине
- Режим ведомого

15.2 Управляющие регистры

С модулем MSSP связаны три регистра: регистр статуса SSPSTAT и два регистра управление SSPCON1, SSPCO2. Работа с этими регистрами и отдельными битами регистров значительно отличается в зависимости от используемого интерфейса I^2C или SPI. Дополнительную информацию смотрите в описании соответствующего интерфейса.

15.3 Режим SPI

В SPI режиме возможен одновременный синхронный прием и передача 8-разрядных данных. Модуль SSP поддерживает четыре режима SPI с типовым использованием трех выводов микроконтроллера:

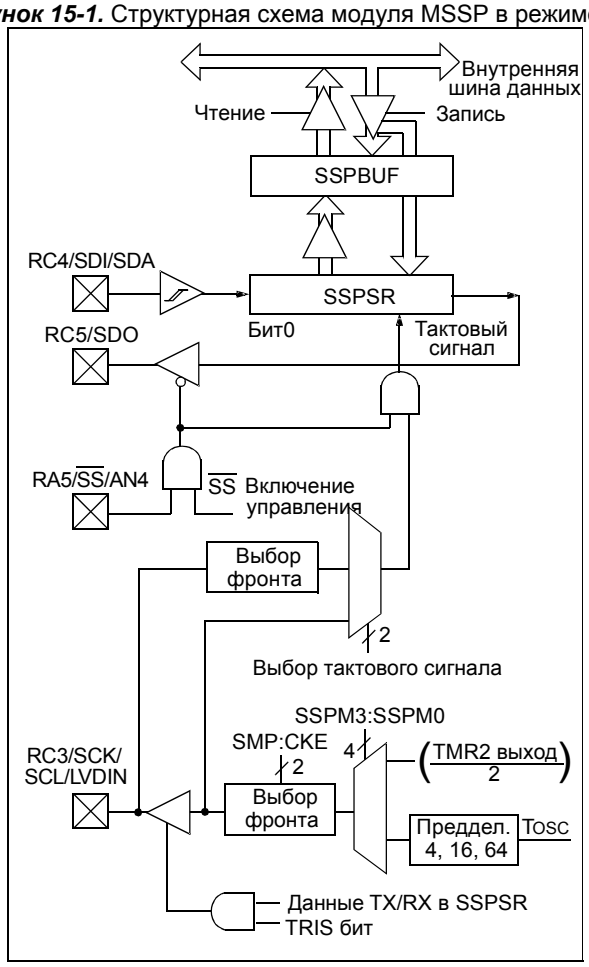
- Вход последовательных данных (SDI) –RC4/SDI/SDA
- Выход последовательных данных (SDO) – RC5/SDO
- Тактовый сигнал (SCK) – RC3/SCK/SCL/LVDIN

Дополнительно может быть задействован четвертый вывод для работы в режиме ведомого:

- Выбор ведомого (-SS) – RA5/-SS/AN4

На рисунке 15-1 показана структурная схема модуля MSSP в режиме SPI.

Рисунок 15-1. Структурная схема модуля MSSP в режиме SPI



15.3.1 Регистры

В режиме SPI модулем MSSP используется четыре регистра:

- Управляющий регистр 1 (SSPCON1)
- Регистр статуса (SSPSTAT)
- Буфер последовательно приемника/передатчика (SSPBUF)
- Сдвиговый регистр (SSPSR) – не адресуемый регистр

В регистрах SSPCON1 и SSPSTAT находятся биты управления и флаги состояния модуля MSSP в режиме SPI. Регистр SSPCON1 доступен для записи/чтения. Младшие 6 битов регистра SSPTAT доступны только для чтения. Старшие 2 бита регистра SSPSTAT доступны для записи и чтения.

Сдвиговый регистр SSPSR предназначен для приема и передачи данных. SSPBUF – буферный регистр. В/из него записываются/читаются данные.

При приеме данных регистры SSPSR и SSPBUF образуют двойной буфер. Когда в SSPSR байт данных загружается полностью, он переписывается в регистр SSPBUF, устанавливается флаг прерывания SSPIF.

При передаче данных регистр SSPBUF двойную буферизацию не имеет. Данные, записанные в SSPBUF, сразу переписываются в SSPSR.

Регистр 15-1. SSPSTAT: Регистр статуса модуля MSSP (режим SPI)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/-A	P	S	R/-W	UA	BF
Бит 7							Бит 0

бит 7 **SMP:** Фаза выборки бита

Ведущий режим SPI

1 = опрос входа в конце периода вывода данных

0 = опрос входа в середине периода вывода данных

Ведомый режим SPI

Для режима ведомого SPI этот бит всегда должен быть сброшен в '0'

бит 6 **CKE:** Выбор фронта тактового сигнала

SPI режим, CKP=0

1 = данные передаются по переднему фронту сигнала на выводе SCK

0 = данные передаются по заднему фронту сигнала на выводе SCK

SPI режим, CKP=1

1 = данные передаются по заднему фронту сигнала на выводе SCK

0 = данные передаются по переднему фронту сигнала на выводе SCK

бит 5 **D/-A:** Бит Данные/Адрес (только для режима I^2C)

бит 4 **P:** Бит STOP (только для режима I^2C)

Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.

бит 3 **S:** Бит START (только для режима I^2C)

Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.

бит 2 **R/-W:** Бит чтения/записи (только для режима I^2C)

бит 1 **UA:** Флаг обновления адреса устройства (только для режима 10-разрядного I^2C)

бит 0 **BF:** Бит статуса буфера

1 = прием завершен, буфер SSPBUF полон

0 = прием не завершен, буфер SSPBUF пуст

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 15-2. SSPCON1: Регистр управления 1 модуля MSSP (режим SPI)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
Бит 7							Бит 0

бит 7 **WCOL:** Бит конфликта записи (Только при передаче, сбрасывается программно)
1 = была предпринята попытка записи в SSPBUF во время передачи предыдущего байта
0 = конфликта не было

бит 6 **SSPOV:** Бит переполнения приемника
SPI режим
1 = принят новый байт, а SSPBUF содержит предыдущие данные(байт в SSPSR будет потерян). В ведомом режиме пользователь должен прочитать содержимое регистра SSPBUF даже, если только передает данные. В ведущем режиме бит в '1' не устанавливается, т.к. каждая операция инициализируется записью в SSPBUF. (сбрасывается в '0' программно)
0 = нет переполнения

Примечание. В режиме ведущего бит SSPOV не устанавливается, т.к. каждый прием данных инициируется записью в SSPBUF.

бит 5 **SSPEN:** Бит включения модуля MSSP
Когда модуль включен, соответствующие порты ввода/вывода настраиваются на выход или вход
SPI режим
1 = модуль MSSP включен, выводы SCK, SDO, SDI, -SS используются модулем MSSP
0 = модуль MSSP выключен, выводы работают как цифровые порты ввода/вывода

Примечание. При включении режима SPI выводы модуля MSSP должны быть соответствующим образом настроены.

бит 4 **CKP:** Бит выбора полярности тактового сигнала
1 = пассивный высокий уровень сигнала
0 = пассивный низкий уровень сигнала

бит 3-0 **SSPM3:SSPM0:** Режим работы модуля MSSP
0000 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/4$
0001 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/16$
0010 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/64$
0011 = ведущий режим SPI, тактовый сигнал = выход TMR2 / 2
0100 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS подключен к MSSP
0101 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS не подключен к MSSP

Примечание. Не указанные комбинации битов предназначены для настройки модуля MSSP в режим I^2C или зарезервированы.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

15.3.2 Работа модуля MSSP в режиме SPI

При инициализации SPI необходимо определить параметры работы модуля SPI битами SSPCON1<5:0>, SSPSTAT<7:6>. Управляющие биты определяют следующие параметры работы:

- Ведущий режим (SCK выход)
- Ведомый режим (SCK вход)
- Полярность тактового сигнала (пассивный уровень SCK)
- Фаза выборки входных данных
- Активный фронт тактового сигнала (передний, задний)
- Частота тактового сигнала (только в ведущем режиме)
- Режим выбора ведомого (только в режиме ведомого)

Модуль MSSP состоит из приемного/передающего регистра сдвига (SSPSR) и буферного регистра (SSBUF). В регистре SSPSR выполняется сдвиг данных из/в микроконтроллер старшим битом вперед. В регистре SSPBUF сохраняются записанные данные, пока не будут получены новые. Приняв 8 бит данных в регистр SSPSR они переписываются в SSPBUF, устанавливается в '1' флаг полного приемного буфера BF (SSPSTAT<0>) и флаг прерывания SSPIF. Двойная буферизация принимаемых данных позволяет принимать следующий байт до чтения предыдущего. Любая запись в регистр SSPBUF во время выполнения операции приема/передачи данных будет игнорирована, при этом устанавливается в '1' флаг WCOL (SSPCON<7>). Пользователь должен программно сбросить бит WCOL в '0', чтобы была возможность проверки выполнения записи в регистр SSPBUF. При приеме данных в режиме SPI регистр SSPBUF должен быть прочитан до момента окончания приема следующего байта. Бит статуса приемного буфера BF (SSPSTAT<0>) указывает на получение нового байта данных. Бит BF аппаратно сбрасывается в '0' при чтении регистра SSPBUF. Принятые данные могут быть недостоверными, если режим SPI используется только для передачи данных. Прерывания от модуля MSSP используются для определения завершения приема/передачи данных (в подпрограмме обработки прерываний необходимо прочитать/записать регистр SSPBUF). Если не планируется использовать прерывания от модуля MSSP, то необходимо предусмотреть программную проверку выполнения записи в регистр SSPBUF для передачи данных. В примере 15-1 показана загрузка данных в регистр SSPBUF (SSPSR) для передачи данных. Затененная команда требуется только, если принимаемые данные имеют какое-то значение (в некоторых приложениях модуль MSSP в режиме SPI используется только для передачи данных).

Пример 15-1. Загрузка данных в регистр SSPBUF(SSPSR)

```
LOOP    BTFSS  SSPSTAT, BF  ;Данные приняты?
        GOTO   LOOP         ;Нет
        MOVF   SSPBUF, W    ;Загрузить в W значение из SSPBUF
        MOVWF  RXDATA       ;Если необходимо, сохранить значение в памяти
        MOVF   TXDATA, W    ;Загрузить в W значение из TXDATA
        MOVWF  SSPBUF       ;Передать новые данные
```

Регистр SSPSR не доступен для непосредственного чтения или записи, все операции выполняются через регистр SSPBUF. В регистре SSPSTAT находятся биты, указывающие текущее состояние модуля MSSP.

15.3.3 Настройка выводов в режиме SPI

Для включения модуля MSSP необходимо установить бит SSPEN (SSPCON1<5>) в '1'. Для сброса или перенастройки режима SPI рекомендуется сбросить бит SSPEN в '0', выполнить изменения параметров работы, а затем вновь установить бит SSPEN в '1'. После включения MSSP в режиме SPI выводы SDI, SDO, SCK, -SS используются последовательным портом. Для корректной работы последовательного порта биты регистров TRIS должны быть настроены следующим образом:

- SDI, бит TRISC<4> должен быть установлен в '1'
- SDO, бит TRISC<5> должен быть сброшен в '0'
- SCK (ведущий режим), бит TRISC<3> должен быть сброшен в '0'
- SCK (ведомый режим), бит TRISC<3> должен быть установлен в '1'
- -SS, бит TRISA<5> должен быть установлен в '1'

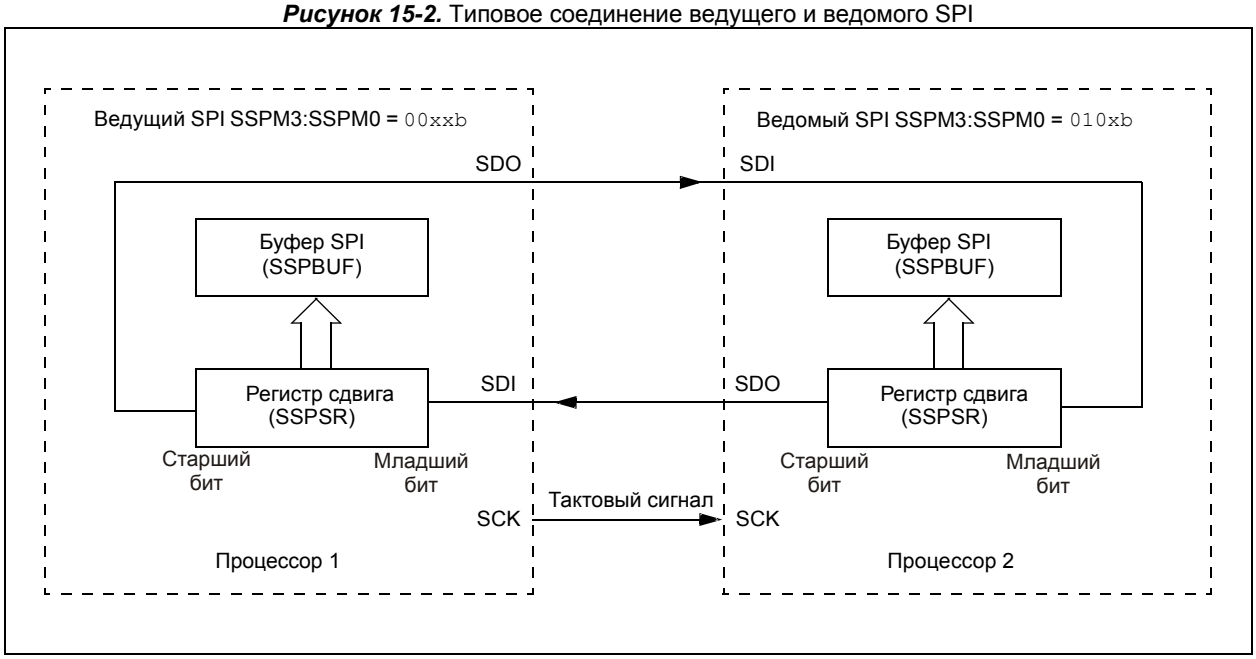
Любая нежелательная функция последовательного порта может быть выключена, настраивая соответствующие биты регистров направления данных TRIS. Например, если в режиме ведущего SPI выполняется только передача данных, то выводы SDI и -SS могут использоваться как цифровые выходы, сбросив соответствующие биты TRIS в '0'.

15.3.4 Типовое включение

На рисунке 15-2 показано типовое соединение двух микроконтроллеров. Главный микроконтроллер (процессор 1) инициализирует передачу, формируя тактовый сигнал SCK. Данные сдвигаются по установленному битом SMP фронту тактового сигнала. Для одновременного приема/передачи данных (фиктивных данных) оба микроконтроллера должны иметь одинаковую полярность тактового сигнала (бит CKP). Всего существует три сценария передачи данных:

- Ведущий передает данные - ведомый передает фиктивные данные
- Ведущий передает данные - ведомый передает данные
- Ведущий передает фиктивные данные - ведомый передает данные

Рисунок 15-2. Типовое соединение ведущего и ведомого SPI



15.3.5 Режим ведущего SPI

Ведущий шины может инициализировать передачу данных в любой момент, поскольку он генерирует тактовый сигнал, и определяет, когда ведомый (процессор 2) должен передать данные в соответствии с используемым протоколом.

В режиме ведущего данные передаются/приняты после их записи/чтения из регистра SSPBUF. Если в SPI режиме требуется только принимать данные, вывод SDO может быть заблокирован (настроен как вход). Данные с вывода SDI последовательно сдвигаются в регистр SSPSR с установленной скоростью. Каждый принятый байт загружается в регистр SSPBUF (как нормально полученный байт) с формированием прерываний и воздействием на соответствующие биты статуса. Эта функция может быть полезна при реализации "монитора шины".

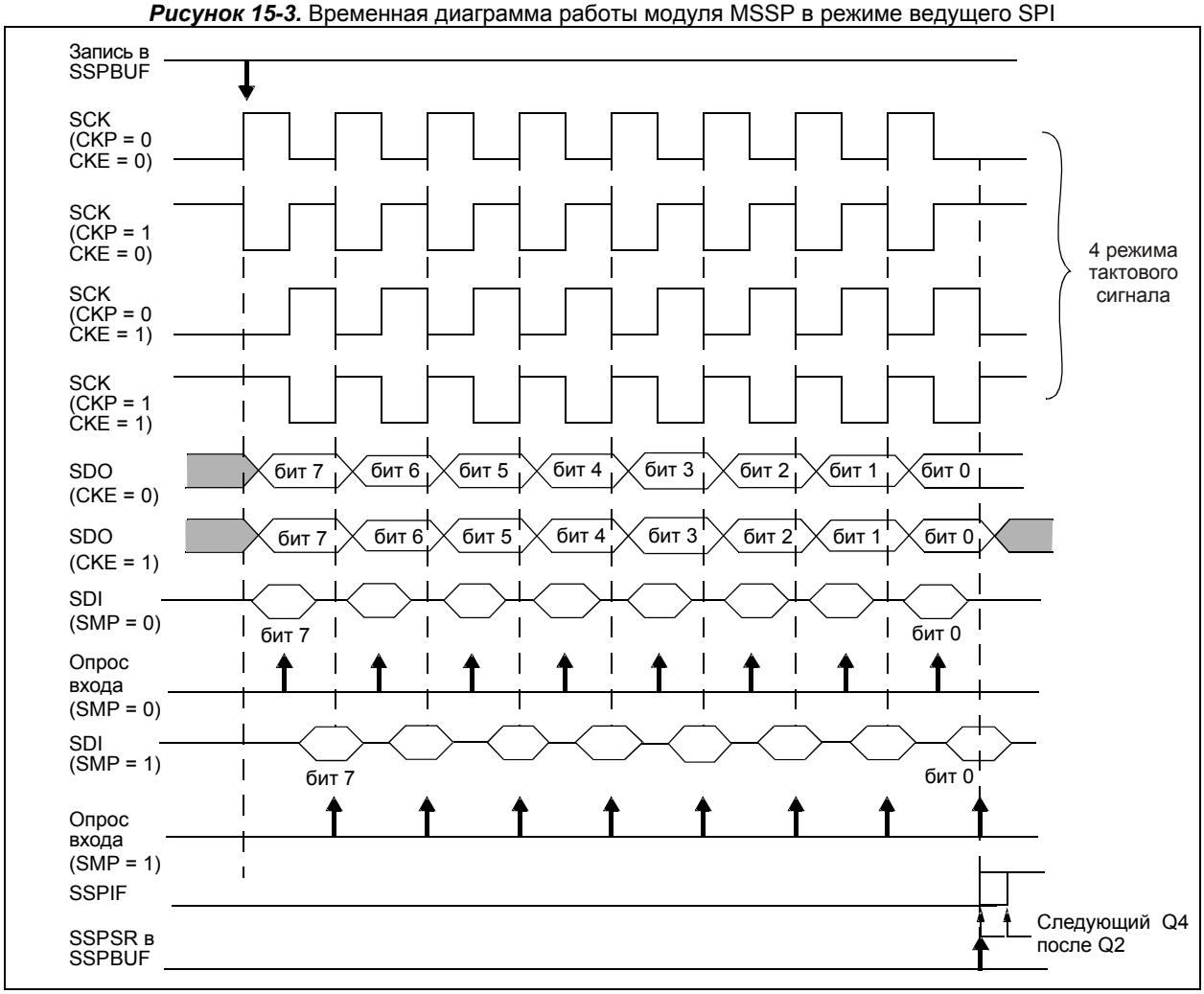
Полярность тактового сигнала устанавливается битом CKP (SSPCON1<4>), что позволяет получить различные методы передачи данных (см. рисунки 15-3, 15-5 и 15-6). Данные всегда передаются старшим битом вперед. В ведущем режиме частота тактового сигнала выбирается программно:

- FOSC/4 (или TCY)
- FOSC/16 (или 4 x TCY)
- FOSC/64 (или 16 x TCY)
- Выход таймера TMR2 / 2

Максимальная частота передачи данных 10МГц при тактовой частоте микроконтроллера 40МГц.

Временная диаграмма передачи данных в режиме ведущего SPI показана на рисунке 15-3. Бит CKE определяет по какому фронту тактового сигнала необходимо выполнять прием данных. Параметры выборки входных данных устанавливаются битом SMP. Поле загрузки принятых данных в регистр SSPBUF устанавливается флаг прерываний SSPIF в '1'.

Рисунок 15-3. Временная диаграмма работы модуля MSSP в режиме ведущего SPI



15.3.6 Режим ведомого SPI

В режиме ведомого данные передаются/принимаются по внешнему тактовому сигналу на выводе SCK. Когда принимается последний бит байта, устанавливается в '1' флаг прерываний SSPIF.

Полярность тактового сигнала выбирается битом CKP (SSPCON1<4>). Внешний тактовый сигнал должен удовлетворять требованиям длительности низкого и высокого логического уровня, описанным в разделе электрических характеристик.

В SLEEP режиме микроконтроллера ведомый может принимать/передавать данные. После приема данных микроконтроллер выходит из режима SLEEP, если разрешены прерывания от модуля MSSP.

15.3.7 Выбор ведомого в режиме SPI

В режиме SPI вывод -SS позволяет подключать несколько ведомых к одному ведущему. Модуль MSSP должен находиться в режиме ведомого SPI (SSPCON1<3:0> = 0100), бит TRIS для вывода -SS установлен в '1', чтобы позволить ведущему выбирать ведомого. Когда на выводе -SS присутствует низкий логический уровень, передача и прием данных разрешены, а вывод SDO управляется модулем SSP. Если на выводе -SS высокий уровень сигнала, то вывод SDO переходит в 3-е состояние. В зависимости от приложения может потребоваться внешний подтягивающий резистор на выводе SDO.

Примечания

1. В режиме ведомого SPI с поддержкой выбора ведомого по сигналу на выводе -SS (SSPCON1<3:0>=0100), SPI модуль сброшен, если на выводе -SS напряжение питания V_{DD}.
2. В режиме ведомого SPI и CKE = 1, необходимо разрешить управление с вывода -SS.

При сбросе модуля SSP в режиме SPI счетчик битов сдвигового регистра очищается. Сброс модуля в режиме SPI происходит при появлении высокого логического уровня на выводе -SS и сбросе в '0' бита SSPEN.

Для реализации двух проводного интерфейса вывод SDO может быть соединен с SDI. Когда SPI должен работать как приемник, вывод SDO настраивается на вход, что отключает передатчик от SDO. SDI всегда должен быть настроен как вход (функция SDI), т.к. это не создает конфликт шины.

Рисунок 15-4. Временная диаграмма синхронизации ведомого

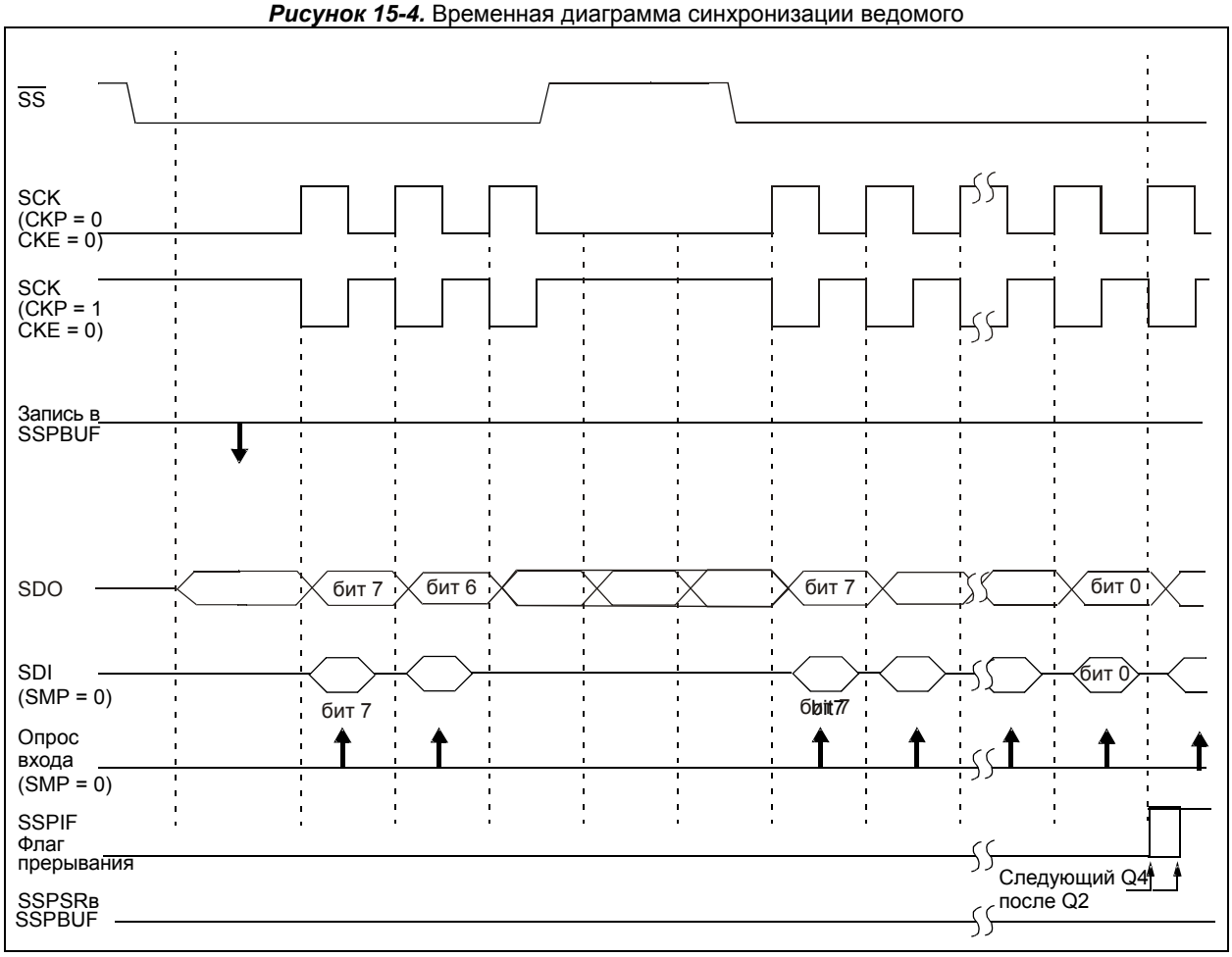


Рисунок 15-5. Временная диаграмма работы модуля MSSP в режиме ведомого SPI (CKE=0)

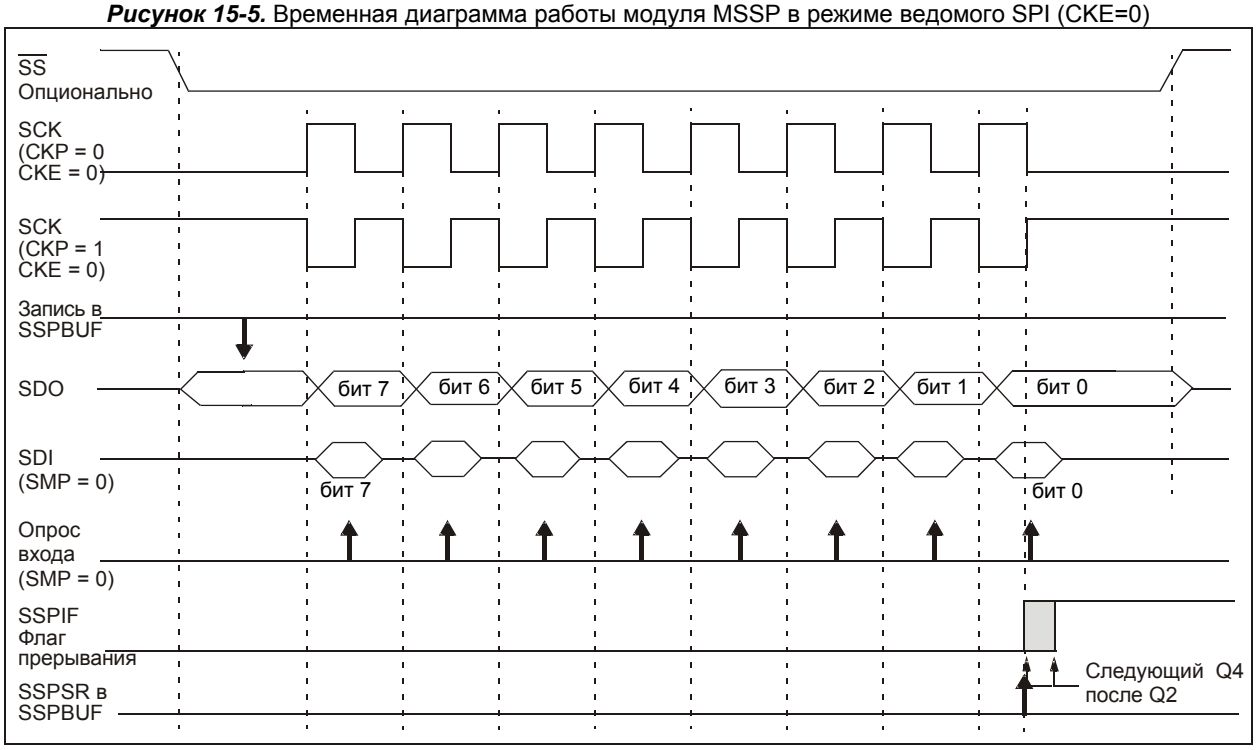
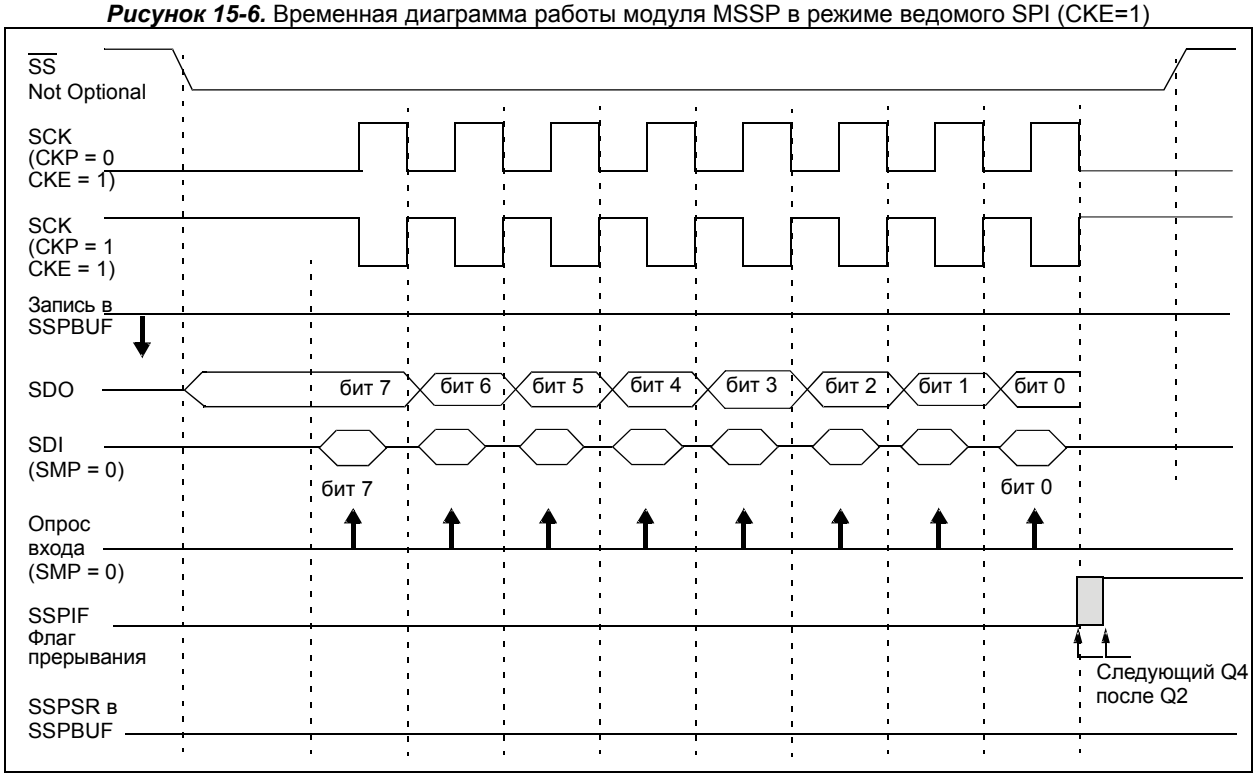


Рисунок 15-6. Временная диаграмма работы модуля MSSP в режиме ведомого SPI (CKE=1)



15.3.8 Работа в SLEEP режиме микроконтроллера

В режиме ведущего SPI тактовый сигнал модуля MSSP отсутствует, состояние приема/передачи данных не изменяется до выхода микроконтроллера из режима SLEEP. После выхода микроконтроллера из режима SLEEP модуль SSP продолжит передачу/прием данных.

В режиме ведомого SPI данные могут быть приняты/переданы, т.к. сдвиговый регистр работает асинхронно. Это позволяет в SLEEP режиме микроконтроллера принять/передать данные в/из сдвигового регистра. Как только будут приняты все 8 бит данных, устанавливается в '1' флаг прерывания от модуля MSSP, и если прерывания разрешены, микроконтроллер выйдет из SLEEP режима.

15.3.9 Эффект сброса

Любой сброс микроконтроллера выключает модуль MSSP, прием/передача данных прекращается.

15.3.10 Совместимость режимов шины

В таблице 15-1 показаны стандартные режимы шины SPI и соответствующая настройка битов CKP, CKE.

Таблица 15-1. Режимы шины SPI

Стандартные режимы SPI	Состояние управляющих битов	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

Таблица 15-2. Регистры и биты, связанные с работой модуля SSP в режиме SPI

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
F94h	TRISC	Регистр направления данных								1111 1111
FC9h	SSPBUF	Буфер приемника/передатчика модуля MSSP								xxxx xxxx
FC6h	SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000
F92h	TRISA	-	Регистр направления данных							-111 1111
FC7h	SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

15.4 Режим I^2C

Модуль MSSP полностью поддерживает все функции ведущих и ведомых устройств, включая поддержку общего вызова, аппаратные прерывания по детектированию битов START и STOP для определения занятости шины I^2C в режиме ведущего (при конкуренции на шине). В MSSP модуле реализована поддержка стандартного режима 7, 10-разрядной адресации.

Для работы с шиной I^2C используется два вывода SCL (сигнал синхронизации) и SDA (данные). Выводы SDA и SCL автоматически настраиваются при включении режима I^2C.

15.4.1 Регистры

Для управления модулем MSSP в режиме I^2C используется шесть регистров:

- SSPCON1, регистр управления 1 MSSP
- SSPCON2, регистр управления 2 MSSP
- SSPSTAT, регистр статуса MSSP
- SSPBUF, буфер приемника/передатчика
- SSPSR, сдвиговый регистр (пользователю не доступен)
- SSPADD, регистр адреса

В регистрах SSPCON1, SSPCON2 и SSPSTAT находятся биты управления и флаги состояния модуля MSSP в режиме SPI. Регистры SSPCON1, SSPCON2 доступны для записи/чтения. Младшие 6 битов регистра SSPTAT доступны только для чтения. Старшие 2 бита регистра SSPSTAT доступны для записи и чтения.

Сдвиговый регистр SSPSR предназначен для приема и передачи данных. SSPBUF – буферный регистр. В/из него записываются/читаются данные.

SSPADD предназначен для хранения адреса устройства, когда модуль MSSP настроен в режим ведомого I^2C. Если модуль MSSP настроен в режиме ведущего I^2C, то семь младших битов регистра SSPADD используются для указания значения перезагрузки генератора скорости передачи данных.

При приеме данных регистры SSPSR и SSPBUF образуют двойной буфер. Когда в SSPSR байт данных загружается полностью, он переписывается в регистр SSPBUF, устанавливается флаг прерывания SPSIF.

При передаче данных регистр SSPBUF двойную буферизацию не имеет. Данные, записанные в SSPBUF, сразу переписываются в SSPSR.

Регистр 15-3. SSPSTAT: Регистр статуса модуля MSSP (режим I^2C)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/-A	P	S	R/-W	UA	BF
Бит 7							Бит 0

бит 7 **SMP:** Управление длительностью фронта
Ведущий или ведомый режим I^2C
1 = управление длительностью фронта выключено в стандартном режиме (100кГц и 1МГц)
0 = управление длительностью фронта включено в скоростном режиме (400кГц)

бит 6 **CKE:** Выбор фронта тактового сигнала
Ведущий или ведомый режим I^2C
1= входные уровни соответствуют спецификации SMBus
0= входные уровни соответствуют спецификации I^2C

бит 5 **D/-A:** Бит Данные/Адрес (только для режима I^2C)
Ведущий режим I^2C
Зарезервировано

Ведомый режим I^2C
1 = последний принятый или переданный байт является информационным
0 = последний принятый или переданный байт является адресным

бит 4 **P:** Бит STOP
1 = указывает, что бит STOP был обнаружен последним (этот бит равен '0' после сброса)
0 = бит STOP не является последним

Примечание. Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.

бит 3 **S:** Бит START
1 = указывает, что бит START был обнаружен последним (этот бит равен '0' после сброса)
0 = бит START не является последним

Примечание. Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.

бит 2 **R/-W:** Бит чтения/записи
Ведомый режим I^2C
1 = чтение
0 = запись

Примечание. Значение бита действительно только после совпадения адреса и до приема бита START, STOP или -ACK.

Ведущий режим I^2C
1 = выполняется передача данных
0 = передачи данных не происходит

Примечание. Логическое ИЛИ этого бита с битами SEN, RSEN, PEN, RCEN или ACKEN укажет на неактивное состояние модуля MSSP.

бит 1 **UA:** Флаг обновления адреса устройства (только для режима 10-разрядного I^2C)
1 = необходимо обновить адрес в регистре SSPADD
0 = обновление адреса не требуется

бит 0 **BF:** Бит статуса буфера
Прием
1 = прием завершен, буфер SSPBUF полон
0 = прием не завершен, буфер SSPBUF пуст

Передача
1 = выполняется передача данных (исключая биты -ACK и STOP), буфер SSPBUF полон
0 = передача данных завершена (исключая биты -ACK и STOP), буфер SSPBUF пуст

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 15-4. SSPCON1: Регистр управления 1 модуля MSSP (режим I^2C)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
Бит 7							Бит 0

бит 7 **WCOL:** Бит конфликта записи

Ведущий режим, передача

1 = запись в SSPBUF была выполнена при не выполнении условий шины I^2C

0 = конфликта не было

Ведомый режим, передача

1 = была предпринята попытка записи в SSPBUF во время передачи предыдущего байта

0 = конфликта не было

Прием (Ведомый или ведущий режим)

Не имеет значения

бит 6 **SSPOV:** Бит переполнения приемника

Прием данных

1 = принят новый байт, а SSPBUF содержит предыдущие данные. (сбрасывается в '0' программно)

0 = нет переполнения

Передача данных

Не имеет значения

бит 5 **SSPEN:** Бит включения модуля MSSP

1 = модуль MSSP включен, выводы SDA, SCL используются модулем MSSP

0 = модуль MSSP выключен, выводы работают как цифровые порты ввода/вывода

бит 4 **CKP:** Управление удержанием линии SCL

Ведомый режим I^2C

1 = не управлять тактовым сигналом

0 = удерживать тактовый сигнал в низком логическом уровне (используется для подготовки данных)

Ведущий режим I^2C

Не имеет значения

бит 3-0 **SSPM3:SSPM0:** Режим работы модуля MSSP

0110 = ведомый режим I^2C, 7-разрядная адресация

0111 = ведомый режим I^2C, 10-разрядная адресация

1000 = ведущий режим I^2C, тактовый сигнал = $F_{OSC}/(4 * (SSPADD+1))$

1011 = программная поддержка ведущего режима I^2C (ведомый режим выключен)

1110 = программная поддержка ведущего режима I^2C, 7-разрядная адресация с разрешением прерываний по приему бит START и STOP

1111 = программная поддержка ведущего режима I^2C, 10- разрядная адресация с разрешением прерываний по приему бит START и STOP

Примечание. Не указанные комбинации битов предназначены для настройки модуля MSSP в режим SPI или зарезервированы.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 15-5. SSPCON2: Регистр управления 2 модуля MSSP (режим I^2C)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
Бит 7							Бит 0

бит 7 **GCEN:** Бит разрешения поддержки общего вызова (только для ведомого режима I^2C)
1 = разрешить прерывания при приеме в регистр SSPSR адреса общего вызова (0000h)
0 = поддержка общего вызова выключена

бит 6 **ACKSTAT:** Бит статуса подтверждения (только для ведущего режима I^2C)
Передача ведущего I^2C
1 = подтверждения не было получено от ведомого
0 = подтверждение от ведомого было получено

бит 5 **ACKDT:** Бит подтверждения (только для ведущего режима I^2C)
Прием ведущего I^2C
1 = нет подтверждение
0 = подтверждения

Примечание. Значение этого бита передается при разрешении формирование бита подтверждения.

бит 4 **ACKEN:** Сформировать бит подтверждения (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит ACKDT. Аппаратно сбрасывается в '0'
0 = подтверждение не формируется

бит 3 **RCEN:** Разрешить прием данных (только для ведущего режима I^2C)
1 = разрешить прием данных с шины I^2C
0 = приемник выключен

бит 2 **PEN:** Сформировать бит STOP (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит STOP. Аппаратно сбрасывается в '0'
0 = бит STOP не формируется

бит 1 **RSEN:** Сформировать бит повторный START (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит повторный START. Аппаратно сбрасывается в '0'
0 = бит повторный STAT не формируется

бит 0 **SEN:** Сформировать бит START/ Включение функции «растяжения» сигнала
Ведущий режим I^2C
1 = на выводах SCL, SDA формируется бит START. Аппаратно сбрасывается в '0'
0 = бит START не формируется

Ведомый режим I^2C
1 = функция «растяжения» тактового сигнала включена для приема и передачи данных ведомым
0 = функция «растяжения» выключена

Примечание. Для битов ACKEN, RCEN, PEN, RSEN, SEN. Если I^2C модуль не находится в пассивном состоянии, то ни один из битов не может быть установлен в '1' (поставлен в очередь), не может быть выполнена запись в регистр SSPBUF (или запись в регистр SSPBUF заблокирована).

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

15.4.2 Работа модуля MSSP в режиме I^2C

Включение модуля MSSP выполняется установкой бита SSPEN (SSPCON1<5>) в '1'.

В регистре SSPCON1 устанавливается требуемый режим I^2C. С помощью четырех битов (SSPCON1<3:0>) можно выбрать один из режимов I^2C:

- Ведомый режим I^2C, 7-разрядная адресация;
- Ведомый режим I^2C, 10-разрядная адресация;
- Ведущий режим I^2C, тактовый сигнал = $F_{OSC}/(4 * (SSPADD+1))$;
- Программная поддержка ведущего режима I^2C.

При выборе любого режима I^2C выводы SCL и SDA должны быть настроены на вход, установкой соответствующих битов регистра TRISC в '1'. После выбора режима I^2C и установки бита SSPEN в '1' выводы SDA (линия данных), SCL (линия синхронизации) подключаются к модулю MSSP.

15.4.3 Режим ведомого I2C

В режиме ведомого I^2C выводы SCL, SDA должны быть настроены на вход (TRISC<4:3> = 1). Модуль MSSP автоматически изменит направление вывода SDA при передаче данных ведомым.

В режиме ведомого автоматически генерируется прерывание при совпадении адреса. Дополнительно пользователь может выбрать режим, при котором будут генерироваться прерывания при обнаружении битов START, STOP.

При совпадении адреса или после приема байта данных (если предварительно совпал адрес) аппаратно генерируется бит подтверждения (-ACK), а затем данные из регистра SSPSR загружаются в SSPBUF.

Существует несколько условий, при которых бит -ACK не формируется (эти условия могут возникать одновременно):

a) Бит BF (SSPSTAT<0>) = 1 перед приемом данных
b) Бит переполнения SSPOV (SSPSTAT<6>) = 1 перед приемом данных

Если бит BF = 1, то значение из SSPSR не переписывается в регистр SSPBUF, а биты SSPIF и SSPOV устанавливаются в '1'. Бит BF аппаратно сбрасывается в '0' при чтении из регистра SSPBUF, а бит SSPOV необходимо сбрасывать в '0' программно.

Минимальная длительность логических уровней входного сигнала синхронизации SCL должна удовлетворять требованиям раздела электрических характеристик (см. параметры 100 и 101).

15.4.3.1 Адресация

После включения модуля MSSP ожидается формирование на шине бита START. Получив бит START, принимается 8 бит в сдвиговый регистр SSPSR. Выборка битов происходит по переднему фронту синхронизирующего сигнала на выводе SCL. По заднему фронту восьмого такта сигнала SCL значение в регистре SSPSR<7:1> сравнивается с содержимым регистра SSPADD. Если значение адреса совпадает, а биты BF и SSPOV равны нулю, то выполняются следующие действия:

a) Значение регистра SSPSR загружается SSPBUF по 8-му заднему фронту сигнала SCL
b) Устанавливается флаг BF в '1' (буфер полон) по 8-му заднему фронту сигнала SCL
c) Генерируется бит -ACK
d) Устанавливается флаг прерываний SSPIF в '1' (если разрешено, генерируется прерывание) по 9-му заднему фронту сигнала SCL.

В режиме ведомого при 10-разрядной адресации необходимо принять два байта адреса. Пять старших бит первого байта определяют: является ли полученный байт первым байтом 10-разрядного адреса. Бит R/-W(SSPSTAT<2>) должен быть настроен для приема второго байта адреса. Для 10-разрядной адресации первый байт адреса должен иметь формат '1111 0 A9 A8 0', где A9:A8 два старших бита адреса. Рекомендуемая последовательность действий при 10-разрядной адресации (шаги 7-9 для передачи ведомым):

1. Принять старший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1')
2. Записать младший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL)
3. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'
4. Принять младший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1')
5. Записать старший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL)
6. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'
7. Принять бит повторный START
8. Принять старший байт адреса (устанавливаются биты SSPIF и BF в '1')
9. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.

15.4.3.2 Прием данных ведомым

Если бит R/-W в адресном байте равен нулю, а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT сбрасывается в '0'. Принятый адрес загружается в регистр SSPBUF.

Если бит BF (буфер полон) или SSPOV (переполнение буфера) установлен в '1', то бит подтверждения -ACK не формируется. Эту ошибку необходимо обработать программно. Если было выполнено чтение из регистра SSPBUF но не был сброшен бит SSPOV в '0', то бит -ACK не формируется.

Прерывание от модуля MSSP генерируются при каждом принятом байте с шины I^2C, установкой флага SSPIF в '1' (сбрасывается программно). Регистр SSPSTAT используется для определения типа принятого байта.

Если бит SEN установлен (SSPCON2<0>=1), то линия синхронизации SCL будут удерживаться в низком уровне после каждого принятого байта. Тактовый сигнал отпускается установкой бита CKP (SSPCON1<4>) в ‘1’ (Смотрите раздел «Удержание тактового сигнала»).

15.4.3.3 Передача данных ведомым

Если бит R/-W в адресном байте равен '1', а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT устанавливается в '1'. Принятый адрес загружается в регистр SSPBUF. Бит -ACK формируется девятым битом, после чего линия SCL удерживается в низком логическом уровне. Передаваемые данные должны быть записаны в регистр SSPBUF, после чего они автоматически переписываются в регистр SSPSR. После записи данных необходимо "отпустить" сигнал SCL установкой бита CKP(SSPCON1<4>) в '1'. Ведущий шины контролирует состояние линии SCL, ожидая смены уровня сигнала. Восемь бит загруженных данных последовательно сдвигаются по заднему фронту сигнала SCL, что гарантирует достоверное значение данных на линии SDA (см. рисунок 15-9).

Ведущее устройство формирует бит подтверждения -ACK на девятом такте сигнала SCL для каждого принятого байта. Если бит подтверждения -ACK не сформирован (высокий уровень сигнала SDA), передача данных завершена. Логика ведомого устройства настраивается на обнаружение бита START. Если бит подтверждения -ACK был получен (низкий уровень сигнала SDA), в регистр SSPBUF необходимо записать новый байт для передачи. Линию SCL также необходимо "отпустить", установкой бита CKP в '1'.

Модуль MSSP генерирует прерывание по каждому переданному байту, устанавливая бит SPPIF в '1' по заднему фронту девятого такта сигнала SCL. Флаг SSPIF должен быть сброшен программно. Регистр SSPSTAT используется для определения статуса передачи данных.

Рисунок 15-8. Временная диаграмма приема данных ведомым I^2C (7-разрядная адресация, SEN=0)

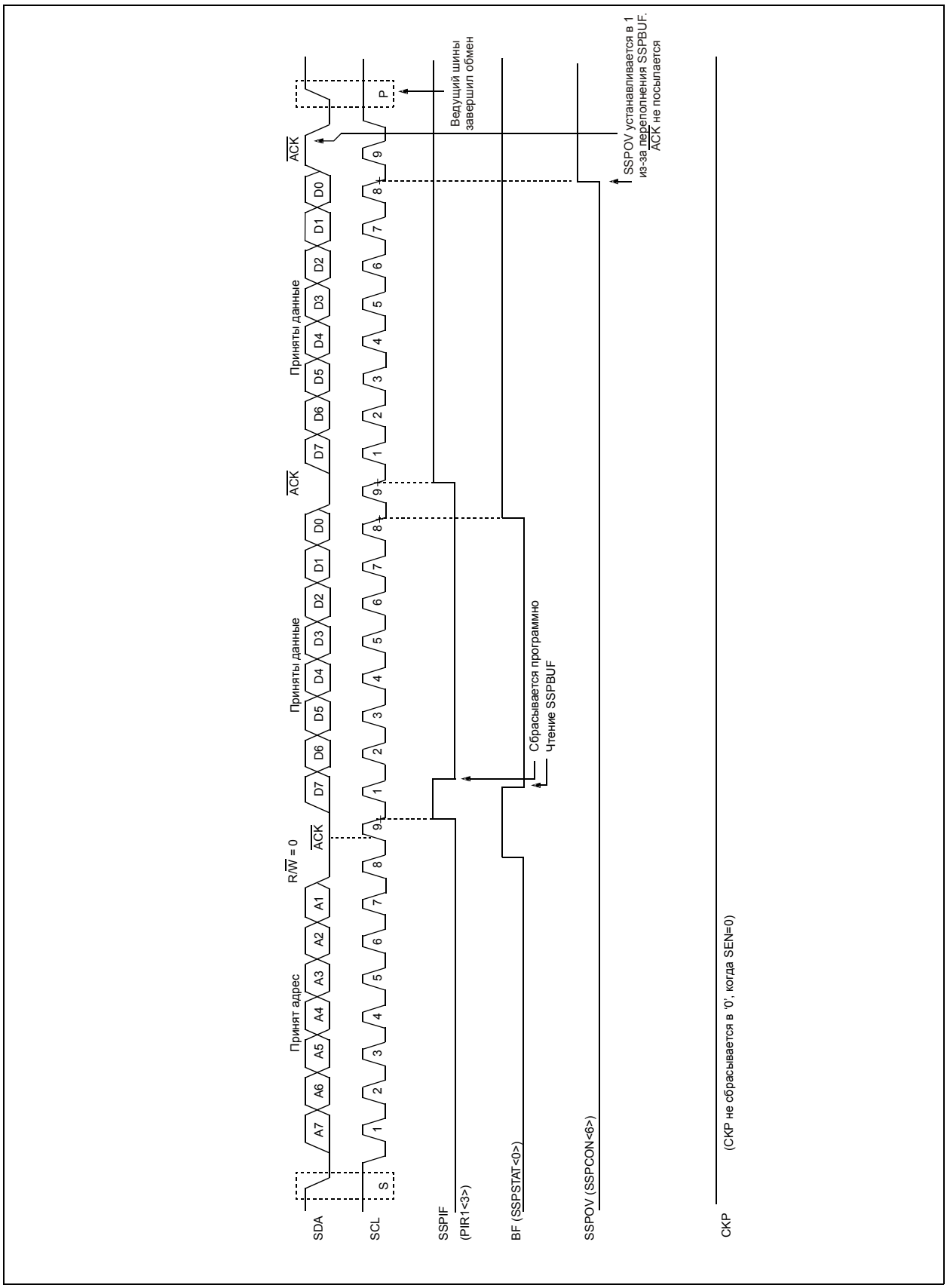


Рисунок 15-9. Временная диаграмма передачи данных ведомым I^2C (7-разрядная адресация, SEN=0)

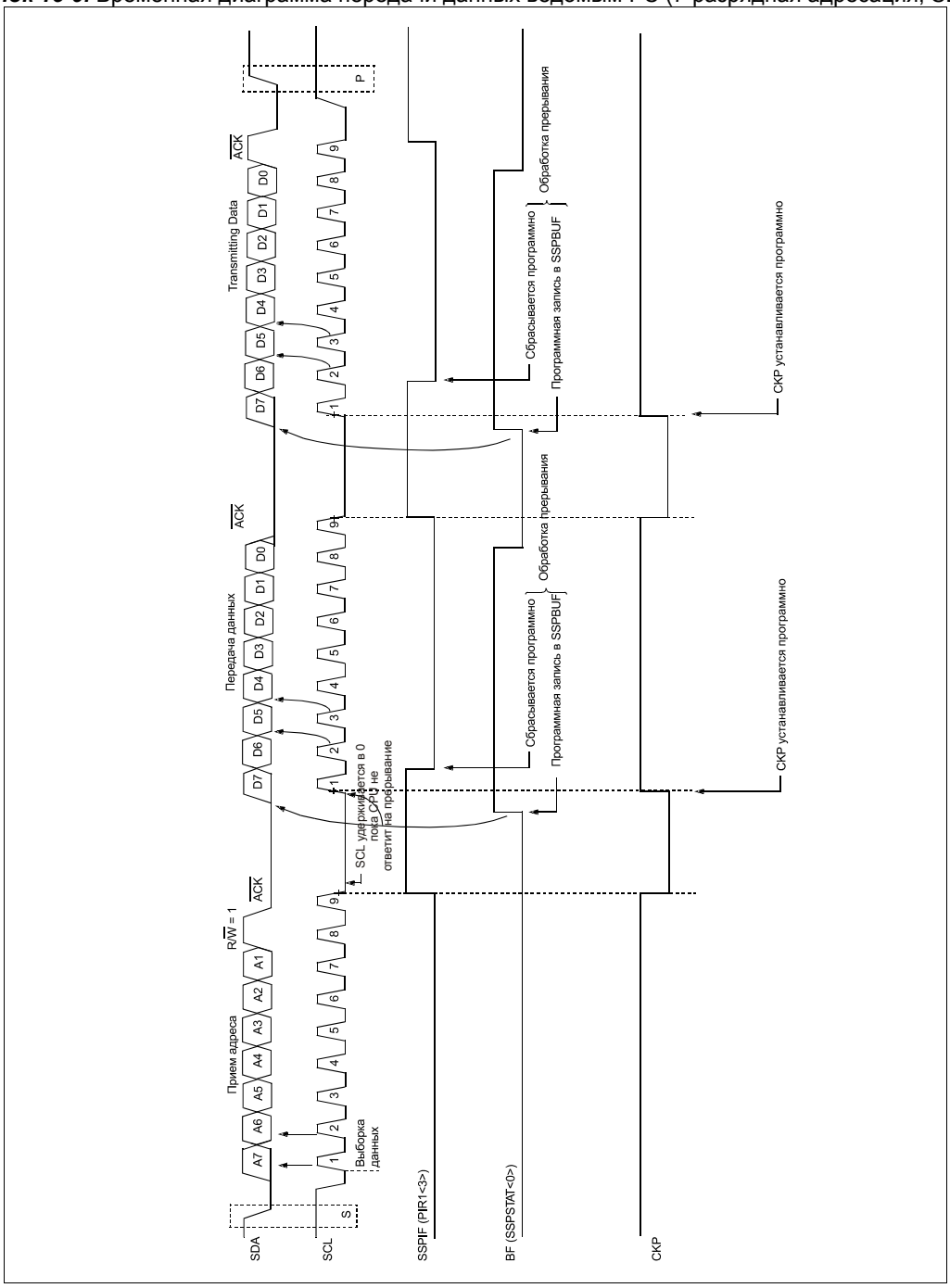


Рисунок 15-10. Временная диаграмма приема данных ведомым I^2C (10-разрядная адресация, SEN=0)

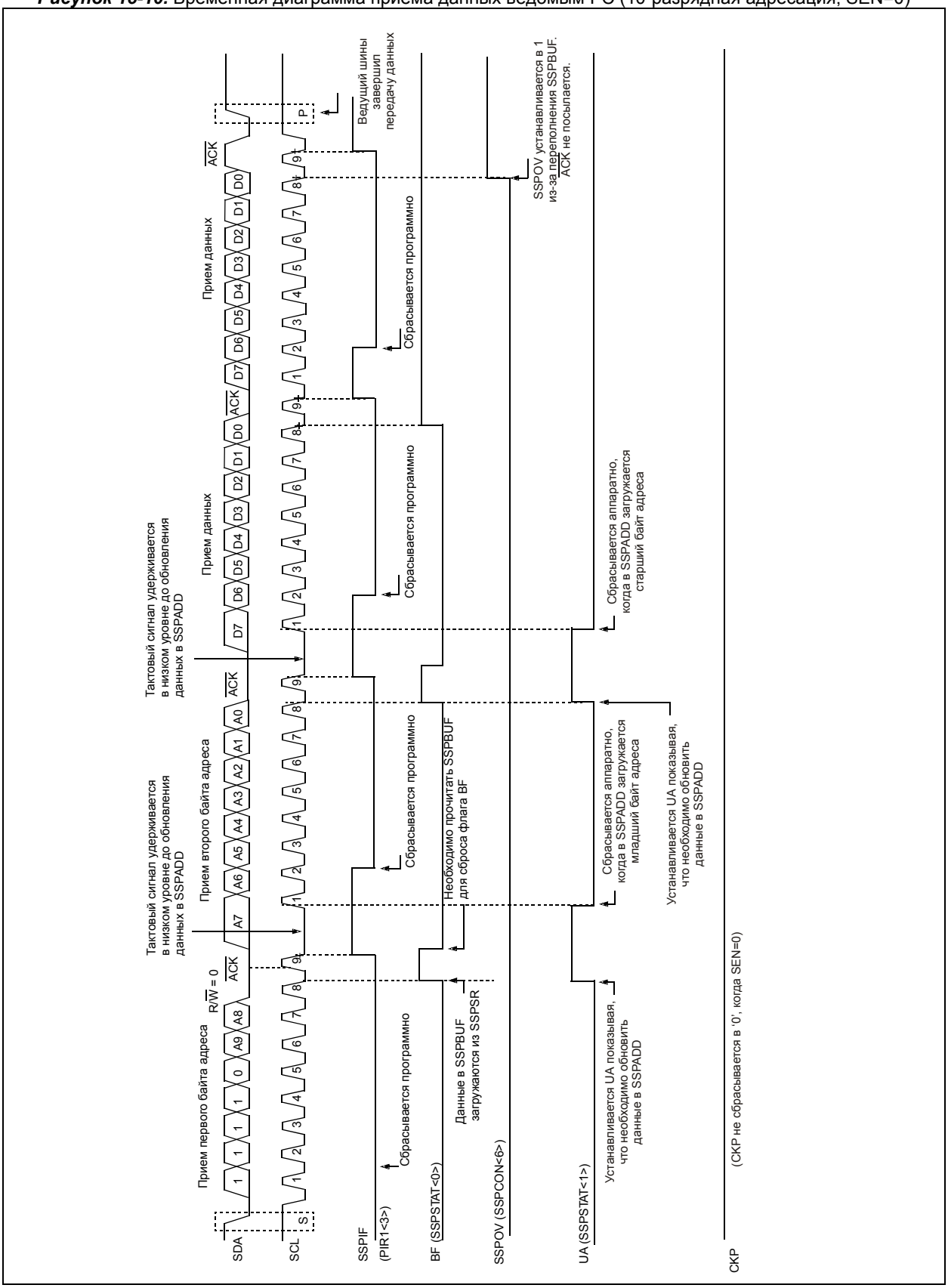
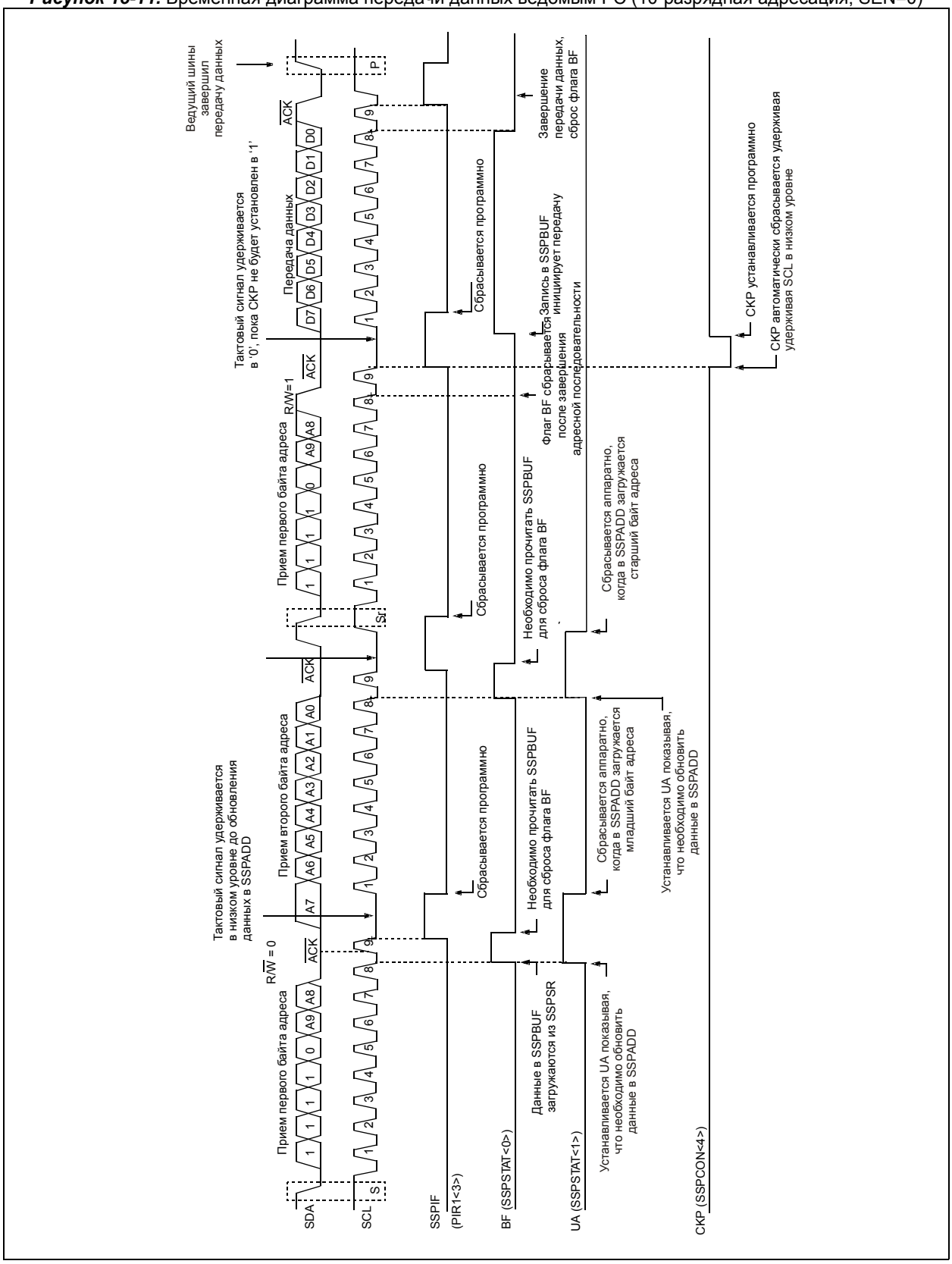


Рисунок 15-11. Временная диаграмма передачи данных ведомым I^2C (10-разрядная адресация, SEN=0)



15.4.4 Удержание тактового сигнала

В режиме 7 и 10-разрядной адресации возможно автоматическое удержание тактового сигнала при передаче данных.

Бит SEN (SSPCON2<0>) позволяет включить режим удержания тактового сигнала во время приема данных. Установленный в ‘1’ бит SEN заставляет удерживать вывод SCL в низком логическом уровне после приема каждого байта.

15.4.4.1 Удержание тактового сигнала в режиме ведомого с 7-разрядной адресацией при приеме данных (SEN=1)

В режиме ведомого с 7-разрядной адресацией по заднему фронту 9-го тактового импульса (в конце бита ACK), если бит BF установлен в ‘1’, то бит CKP автоматически сбросится в ‘0’ удерживая линию SCL в низком логическом уровне. Бит CKP должен быть установлен в ‘1’ программой пользователя прежде, чем прием будет продолжен. Удерживая линию SCL в низком логическом уровне у пользователя есть время, чтобы прочитать содержимое SSPBUF и выполнить необходимые действия перед приемом очередного байта. Эта функция позволяет предотвратить переполнение входного буфера (смотрите рисунок 15-13).

Примечания:

1. Если пользователь прочитает регистр SSPBUF перед задним фронтом 9-го тактового импульса (что сбросит бит BF в ‘0’), бит CKP не будет сброшен в '0' и тактовый сигнал удерживаться не будет.
2. Бит CKP устанавливается в ‘1’ программно, независимо от состояния бита BF. Необходима некоторая осторожность в сбросе бита BF перед новым приемом данных, чтобы предотвратить условие переполнения буфера.

15.4.4.2 Удержание тактового сигнала в режиме ведомого с 10-разрядной адресацией при приеме данных (SEN=1)

В режиме ведомого с 10-разрядной адресацией во время приема адреса также поддерживается функция удержания тактового сигнала, но бит CKP не сбрасывается. Если бит UA установлен после 9 тактов синхросигнала, то тактовый сигнал будет удерживаться. Бит UA устанавливается после приема старшего байта 10-разрядного адреса, а также после приема 2-й части 10-разрядного адреса с битом R/-W = 0. Прекращение удержания тактового сигнала происходит после обновления данных в регистре SSPADD. Тактовый сигнал будет удерживаться после приема каждого байта данных, как описано в режиме 7-разрядной адресации.

Примечание. Если пользователь проверяет состояние бита UA и сбрасывает его обновляя данные в SSPADD до 9-го тактового импульса (или произошел сброс BF чтением SSPBUF), то тактовый сигнал не будет удерживаться (бит CKP не будет установлен в ‘1’). Удержание тактового сигнала на основе состояния бита BF происходит только при приеме данных.

15.4.4.3 Удержание тактового сигнала в режиме ведомого с 7-разрядной адресацией при передаче данных

В режиме ведомого с 7-разрядной адресацией при передаче данных бит CKP автоматически сбрасывается после заднего фронта 9-го тактового импульса, если бит BF = 0. Удержание тактового сигнала происходит независимо от состояния бита SEN.

В программе пользователя необходимо установить бит CKP перед новой передачей данных. Удерживая линию SCL в низком логическом уровне у пользователя есть время выполнить необходимы действия и загрузить новые данные в SSPBUF прежде, чем ведущий шины начнет прием данных (смотрите рисунок 15-9).

Примечания:

1. Если пользователь очищает бит BF чтением содержимого SSPBUF до заднего фронта 9-го тактового импульса, то бит CKP не будет сброшен в ‘0’ и тактовый сигнал удерживаться не будет.
2. Бит CKP может быть установлен в ‘1’ вне зависимости от состояния бита BF.

15.4.4.4 Удержание тактового сигнала в режиме ведомого с 10-разрядной адресацией при передаче данных

В режиме ведомого с 10-разрядной адресацией при приеме адресной последовательности управление тактовым сигналом происходит аналогично, как при приеме данных по состоянию бита UA. Первые два адресных байта сопровождаются третьим адресным байтом, который содержит старшие биты 10-разрядного адреса и бит R/-W = 1. После приема 3-го адресного байта бит UA не устанавливается и модуль MSSP управляет тактовым сигналом в зависимости от состояния бита BF (смотрите рисунок 15-11).

15.4.4.5 Синхронизация тактового сигнала и бит CKP (SEN = 1)

Бит SEN также используется для синхронизации тактового сигнала при записи в бит CKP. Если пользователь сбросит бит CKP в ‘0’, то SCL будет удерживаться низком логическом уровне. Когда SEN=1, SCL не будет удерживаться в низком логическом уровне, пока на SCL не появится ‘0’. Линия SCL будет удерживаться в низком логическом уровне, пока CKP не будет установлен в ‘1’. Управление удержанием линии SCL не нарушает временные требования к шине I2C (смотрите рисунок 15-12).

Примечание. Если бит SEN=0 и произошел сброс бита CKP, то линия SCL будет немедленно переведена в низкий логический уровень независимо от текущего состояния.

Рисунок 15-12. Временная диаграмма синхронизации тактового сигнала

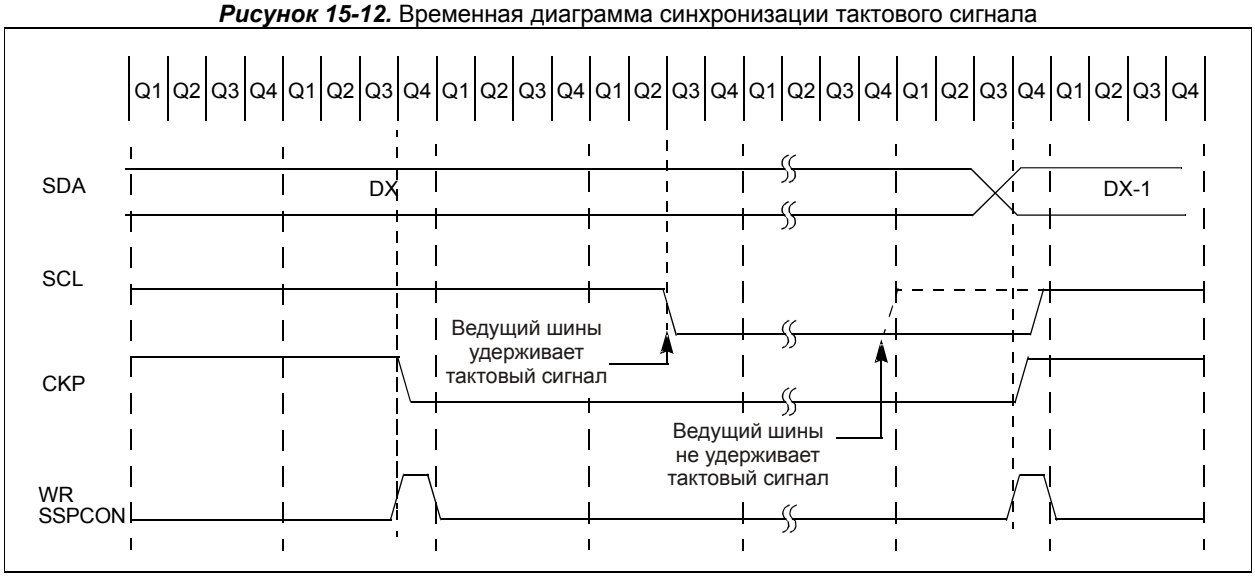


Рисунок 15-13. Временная диаграмма приема данных ведомым I^2C (7-разрядная адресация, SEN=1)

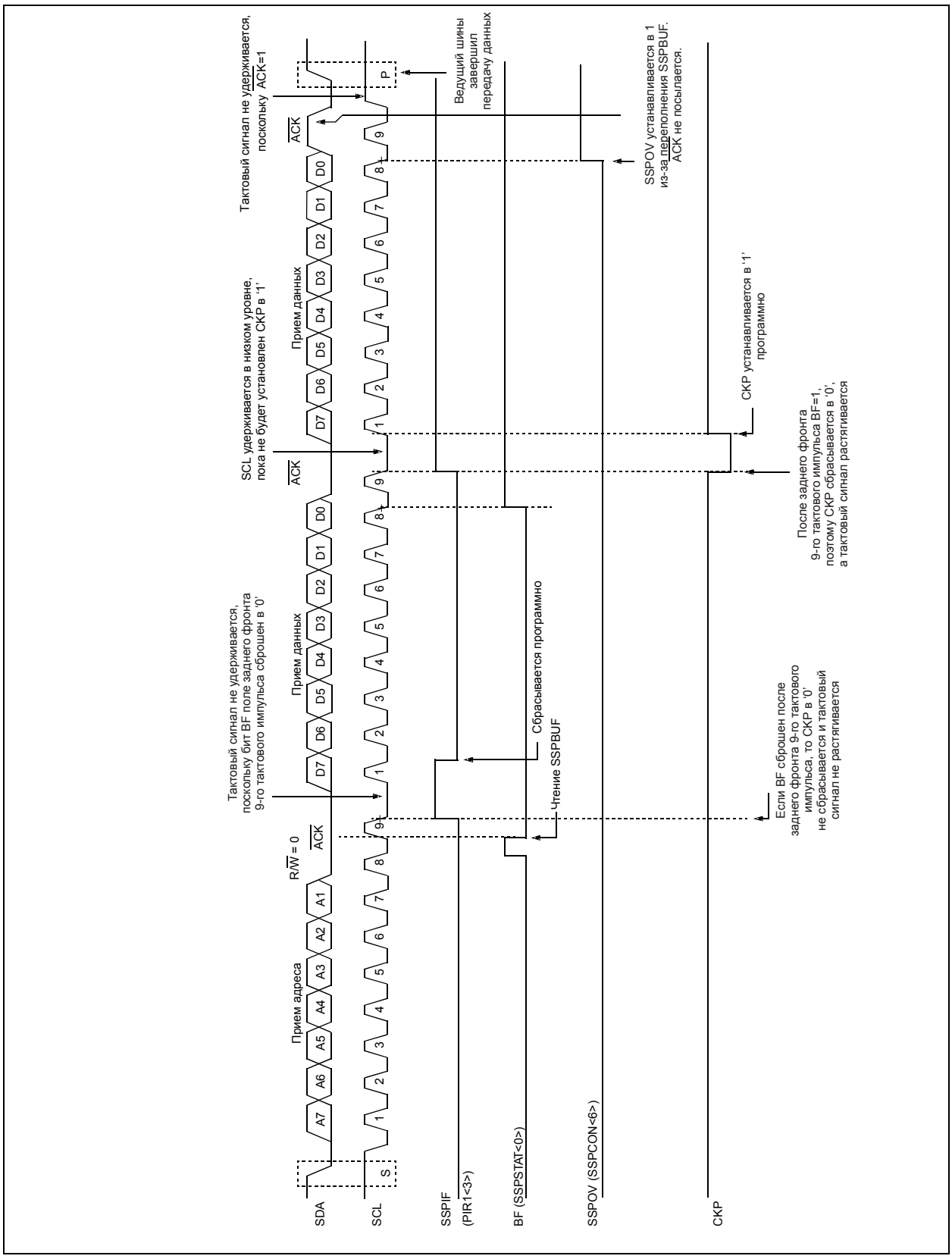
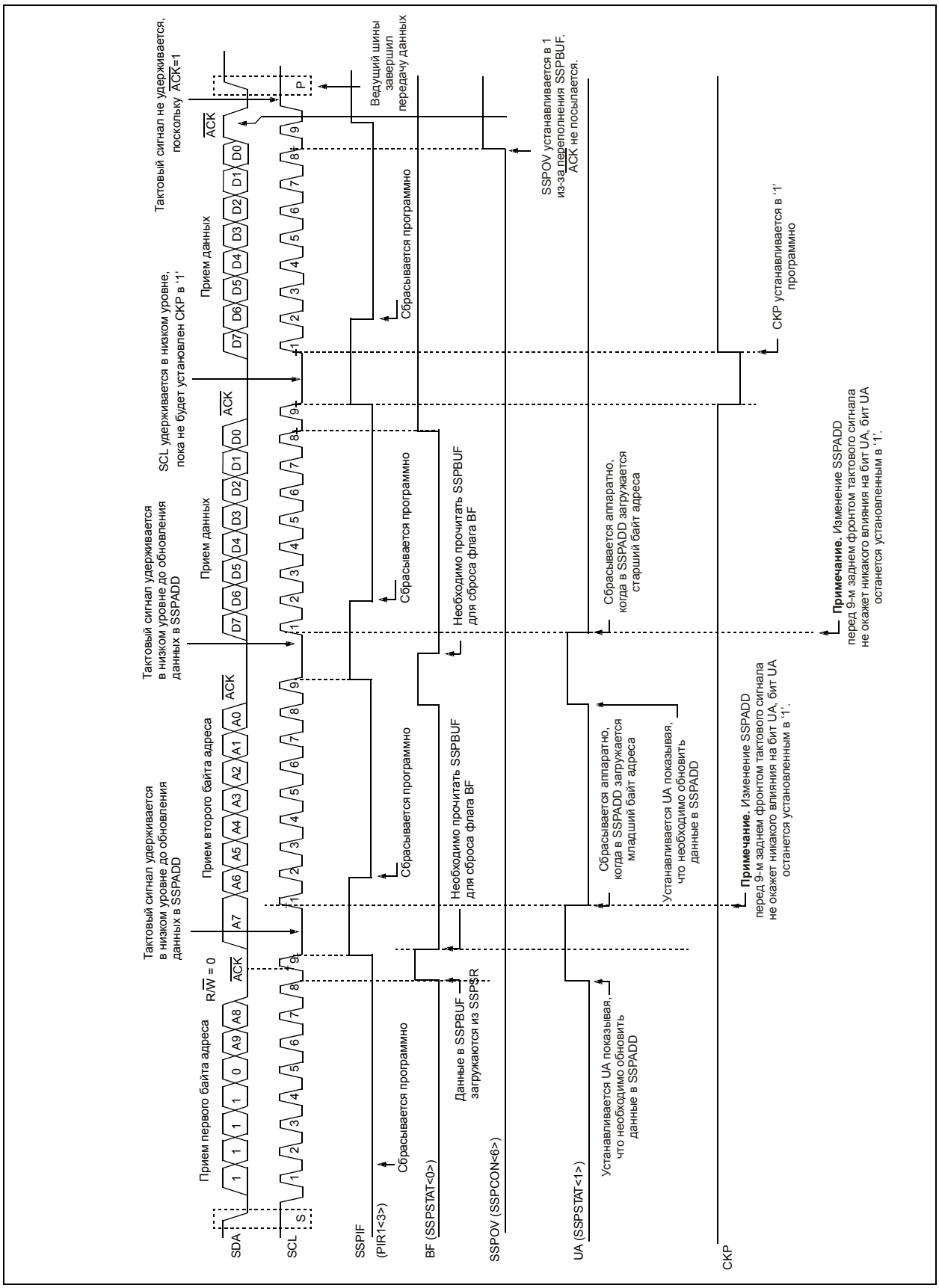


Рисунок 15-14. Временная диаграмма приема данных ведомым I^2C (10-разрядная адресация, SEN=1)



15.4.5 Поддержка общего вызова

Процедура адресации на шине I^2C такова, что первый после START байт определяет, к какому из ведомых устройств обращается ведущий шины. Исключением является адрес общего вызова, при использовании которого теоретически должны откликнуться все ведомые.

Адрес общего вызова – один из восьми зарезервированных адресов шины I^2C, все биты которого равны нулю (в том числе и бит R/-W).

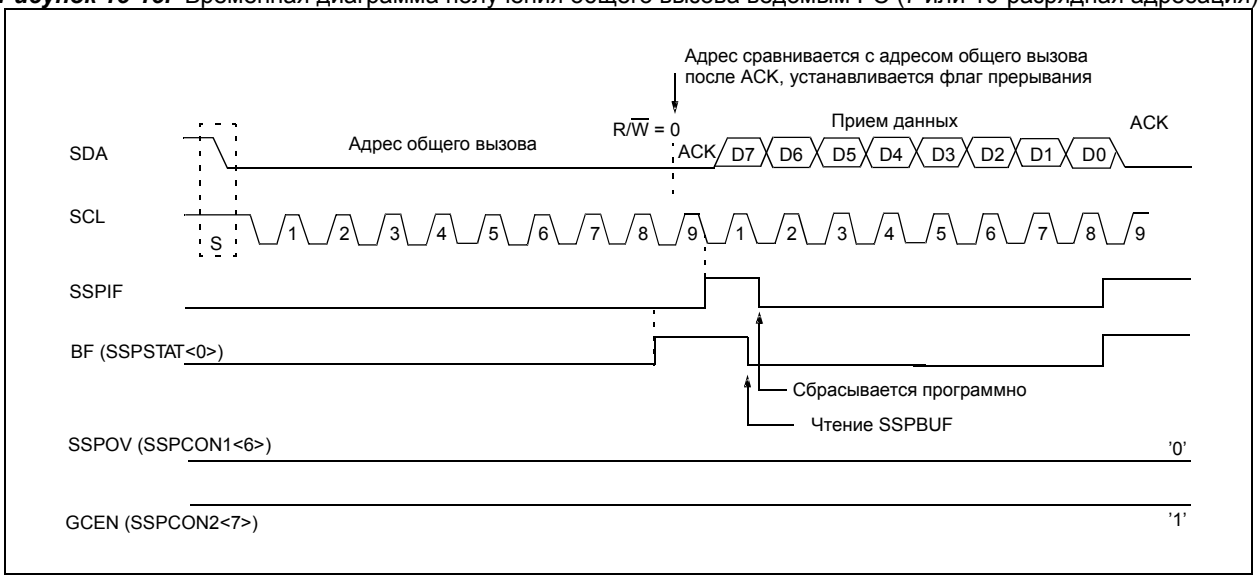
Распознавание адреса общего вызова включается установкой бита GCEN (SSPCON2<7>) в '1'. Следующий за START байт помещается в регистр SSPSR и сравнивается с содержимым SSPADD и фиксированным адресом общего вызова.

При получении адреса общего вызова, содержимое SSPSR переписывается в регистр SSPBUF (устанавливается бит BF в '1') по заднему фронту восьмого такта. На девятом такте формируется бит подтверждения (-ACK) и устанавливается флаг прерываний SSPIF в '1'.

Содержимое регистра SSPBUF позволяет определить получение общего вызова.

В 10-разрядном режиме требуется обновить содержимое регистра SSPADD для проверки соответствия младшего байта адреса после установки бита UA(SSPSTAT<1>) в '1'. Если получен адрес общего вызова в 10-разрядном режиме адресации при GCEN=1, то обновлять значение адреса не требуется. После формирование бита подтверждения ведущее устройство начнет принимать данные (см. рисунок 15-15).

Рисунок 15-15. Временная диаграмма получения общего вызова ведомым I^2C (7 или 10-разрядная адресация)



15.4.6 Режим ведущего I2C

Ведущий режим включается соответствующей настройкой битов SSPM в регистре SSPCON1 и установкой в '1' бита SSPEN. В режиме ведущего выводы SCL, SDA управляются аппаратно.

В режиме ведущего поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле MSSP. Шина находится в неактивном состоянии, если бит P=1 или оба бита S, P равны '0'.

Ведущий режим включается соответствующей настройкой битов SSPM в регистре SSPCON и установкой в '1' бита SSPEN. После включения ведущего режима аппаратно могут выполняться следующие функции:

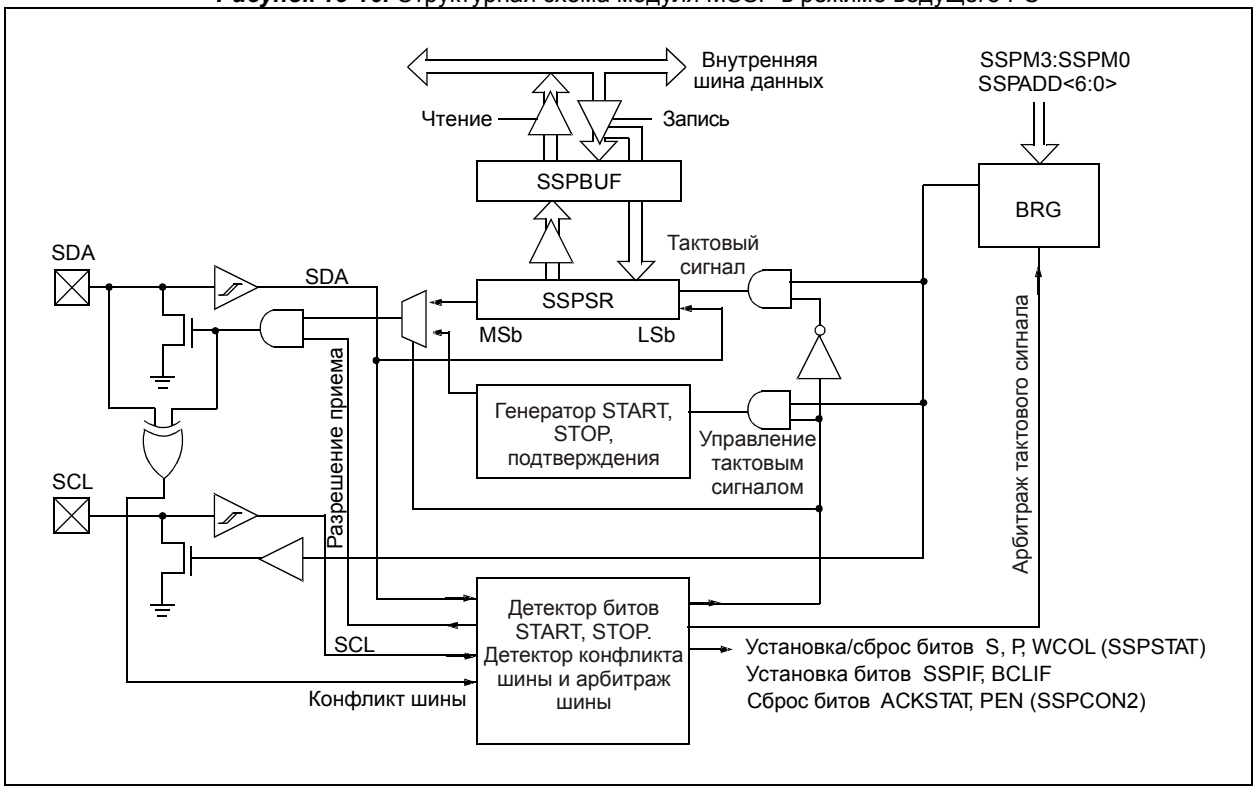
1. Формирование бита START на линии SCL и SDA
2. Формирование бита повторный START на линии SCL и SDA
3. Записью в регистр SSPBUF инициализируется передача байта данных/адреса
4. Формирование бита STOP на линии SCL и SDA
5. Настройка порта I2C на прием данных
6. Формирование бита подтверждения ACK после приема байта на линии SCL и SDA

Примечание. Модуль MSSP в ведущем режиме не имеет стека событий. Это означает, что пользователь не может к примеру инициировать передачу бита START и произвести запись в SSPBUF до того, как START будет завершен. При попытке осуществления подобной операции будет установлен бит WCOL в '1', указывая, что запись в регистр SSPBUF не произошла.

Следующие события на шине I^2C могут привести к установки флага прерываний SSPIF в '1':

- Выполнено условие START
- Выполнено условие STOP
- Передан/принят байт данных
- Передан бит подтверждения
- Выполнено условие повторный START

Рисунок 15-16. Структурная схема модуля MSSP в режиме ведущего I^2C



15.4.6.1 Работа в режиме ведущего I^2C

Ведущий формирует на шине I^2C тактовый сигнал и биты START, STOP. Текущий обмен данными завершается после формирования бита STOP или повторный START. Поскольку бит повторный START инициирует новый обмен данными, шина I^2C остается занятой.

Передатчик ведущего выдает данные на линию SDA, а тактовый сигнал на линию SCL. Первый передаваемый байт содержит 7-разрядный адрес приемника (при 7-разрядной адресации устройств) и бит направления данных R/-W=0. После каждого переданного 8-разрядного байта принимается бит подтверждения -ACK. Биты START и STOP формируется для указания начала и завершения передачи данных.

В режиме приема ведущем на шину I^2C сначала выдается байт, содержащий 7-разрядный адрес передатчика (при 7-разрядной адресации устройств) и бит направления данных R/-W = 1. Данные принимаются с линии SDA, а на линии SCL формирует тактовый сигнал. После каждого принятого байта формируется бит подтверждения. Биты START и STOP формируется для указания начала и завершения передачи данных.

Генератор скорости обмена BRG используется для установки требуемой частоты тактового сигнала на линии SCL – 100кГц, 400кГц или 1МГц. Значение для перезагрузки BRG берется из 7 младших бит регистра SSPADD. BRG начинает работу сразу после записи данных в регистр SSPBUF. Как только операция завершена (передан последний бит байта и принят бит подтверждения) генератор BRG останавливается, вывод SCL "отпускается".

Рекомендованная последовательность действий при передаче данных:

a) Инициировать START установкой бита SEN (SSPCON2<0>) в '1'
b) Ожидать прерывание (если оно разрешено) или установку бита SSPIF после завершения выполнения START
c) Записью в SSPBUF инициируется передача адреса
d) 7 бит адреса (при 7-разрядной адресации) и бит направления данных выдается на SDA
e) Принять подтверждение -ACK от приемника, результат записывается в бит ACKSTAT регистра SSPCON2
f) По заднему фронту девятого такта устанавливается бит SSPIF в '1'
g) Записью в SSPBUF инициируется передача данных
h) 8 бит данных выдаются на SDA
i) Принять подтверждение -ACK от приемника, результат записывается в бит ACKSTAT регистра SSPCON2
j) По заднему фронту девятого такта устанавливается бит SSPIF в '1'
k) Инициировать STOP установкой бита PEN (SSPCON2<6>) в '1'
l) Ожидать прерывание (если оно разрешено) или установку бита SSPIF после завершения выполнения STOP

15.4.7 Генератор скорости обмена

В ведущем режиме, значение для перезагрузки BRG берется из младших 7 бит регистра SSPADD (см. рисунок 15-17). После загрузки SSPADD в BRG, счетчик BRG считает, декрементируя до нуля (в тактах Q2 и Q4), и останавливается до следующей перезагрузки, которая не всегда производится автоматически. Если после окончания счета сигнал на линии SCL должен перейти в высокий уровень, перезагрузка производится только после этого перехода .

В таблице 15-3 представлены скорости обмена на шине I^2C в зависимости от тактовой частоты и значения перезагрузки BRG в регистре SSPADD.

Рисунок 15-17. Структурная схема генератора скорости обмена

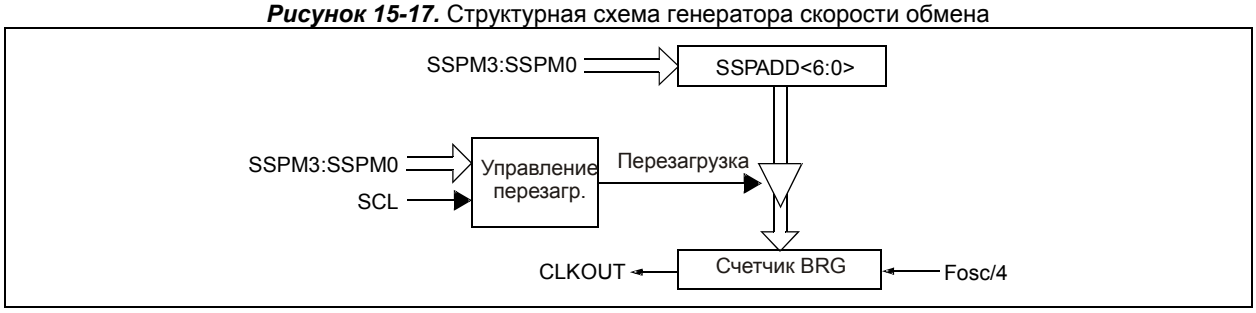


Таблица 15-3. Частота тактового сигнала шины I2C

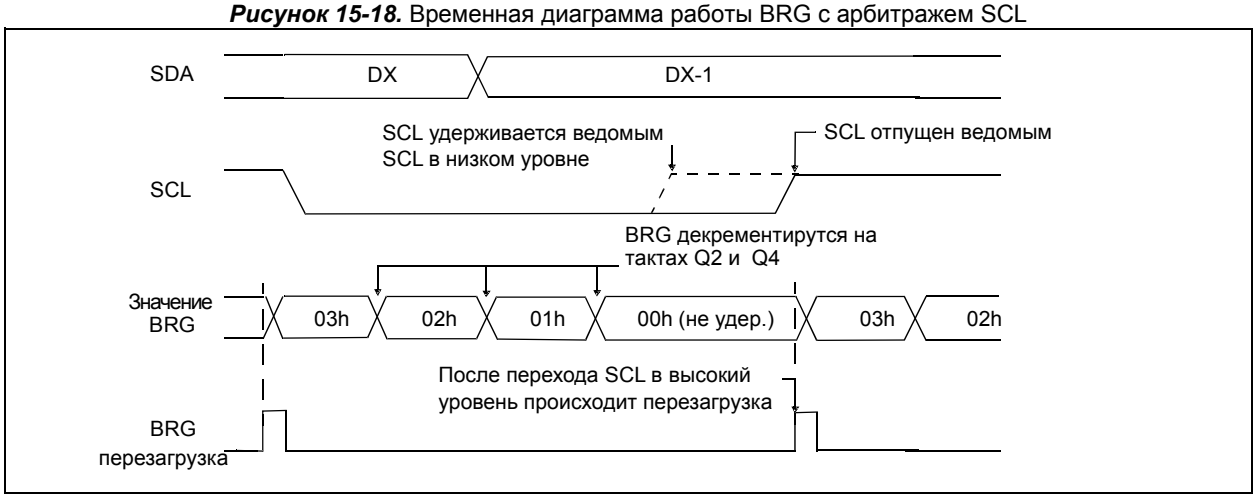
F_{CY} (МГц)	2 x F_{CY} (МГц)	Значение BRG	F_{SCL}
10	20	19h	400кГц
10	20	20h	312.5кГц
10	20	3Fh	100кГц
4	8	0Ah	400кГц
4	8	0Dh	308кГц
4	8	28h	100кГц
1	2	03h	333кГц
1	2	0Ah	100кГц
1	2	00h	1МГц

Примечание. Интерфейс I2C не приспособлен для работы на частоте 400кГц (более 100кГц), но если необходимо, то может работать на высокой скорости передачи данных.

15.4.7.1 Синхронизация тактового сигнала

Синхронизация тактового сигнала производится каждый раз во время приема/передачи данных, формирования бита START или STOP и т.д. При "отпускании" ведущем SCL (SCL должен перейти в высокий уровень). В этот момент BRG приостанавливается пока на SCL не появится высокий уровень сигнала. При появлении сигнала высокого уровня на SCL генератор BRG перегружается значением из SSPADD<6:0> и начинает счет. Если после окончания счета сигнал на линии SCL должен перейти в высокий уровень, перезагрузка производится только после этого перехода (смотрите рисунок 15-18).

Рисунок 15-18. Временная диаграмма работы BRG с арбитражем SCL



15.4.8 Формирование бита START в режиме ведущего I^2C

Чтобы инициировать формирование бита START на шине I^2C, необходимо установить бит SEN (SSPCON2<0>) в '1'. Если на линиях SCL и SDA высокий уровень сигнала, BRG загружается значением из регистра SSPADD <6:0> и начинает счет. Если по окончанию отсчета времени T_{BRG} сохраняется высокий уровень на SCL и SDA, сигнал SDA переводится в низкий логический уровень. Перевод SDA в низкий уровень, в то время когда на линии SCL высокий, и есть бит START на шине I^2C . После формирования бита START устанавливается бит S и флаг прерывания SSPIF в '1', BRG загружается новым значением и начинает счет. По окончании счета бит SEN (SSPCON2<0>) автоматически сбрасывается в '0', генератор останавливается, на SDA остается низкий уровень сигнала. Формирование бита START завершено.

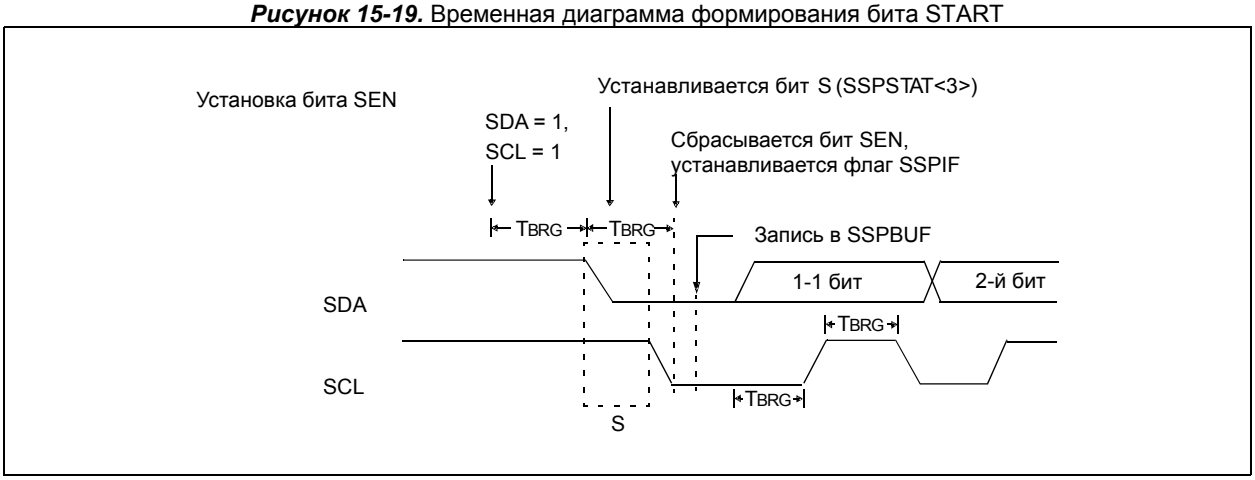
Примечание. Если в начале формирования бита START на SDA или SCL присутствует низкий уровень или во время выполнения START низкий уровень на SCL появляется раньше, чем на SDA, устанавливается флаг прерывания BCLIF (конфликт шины), выполнение START прекращается, MSSP переходит в состояние ожидания.

15.4.8.1 Флаг WCOL

Если во время формирования бита START производится попытка записи в SSPBUF, устанавливается бит WCOL, а запись не происходит.

Примечание. Поскольку MSSP не имеет стека событий, установка любого из младших 5 битов регистра SSPCON2 до завершения формирования бита START запрещено.

Рисунок 15-19. Временная диаграмма формирования бита START



15.4.9 Формирование бита повторный START в режиме ведущего I2C

Чтобы инициировать формирование бита повторный START, необходимо установить бит RSEN (SSPCON2<1>) в то время, когда модуль MSSP находится в режиме ожидания. При включении формирования бита повторный START линия SCL переводится в низкий логический уровень. Когда на SCL устанавливается низкий уровень сигнала, BRG перезагружается содержимым регистра SSPADD<6:0> и начинает отсчет, при этом SDA "отпускается" в высокий уровень. Если по окончанию счета BRG, на линии SDA сохраняется высокий уровень, SCL также "отпускается". BRG вновь перезагружается и начинает отсчет. Если по окончании отсчета времени T_{BRG} сохраняется высокий уровень на линиях SCL и SDA, сигнал SDA переводится в низкий уровень. Бит RSEN (SSPCON2<1>) аппаратно сбрасывается в '0'. После формирования бита повторный START на шине I^2C устанавливается бит S (SSPSTAT<3>) в '1'. Флаг SSPIF не будет установлен в '1' до тех пор, пока BRG не перезагрузится новым значением и начнет счет.

Примечания:

1. Если бит RSEN установлен в '1' во время выполнения какой либо операции на шине, то не будет выполнено никаких действий.
2. Если на SDA низкий уровень при переходе SCL из низкого уровня в высокий или низкий уровень на SCL появляется раньше, чем на SDA, устанавливается флаг прерывания BCLIF (конфликт шины), формирование бита повторный START прекращается, MSSP переходит в состояние ожидания.

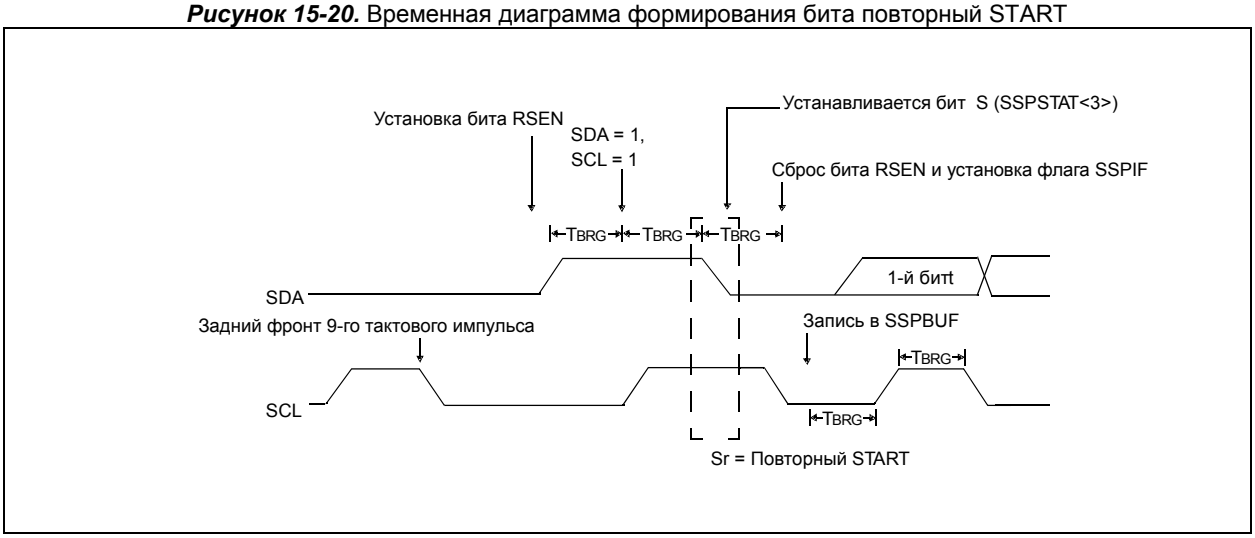
Сразу после установки бита SSPIF пользователь может загрузить регистр SSPBUF 7-разрядным адресом (для 7-разрядного режима адресации) или старшим байтом 10-разрядного адреса. По завершению передачи 8 бит и получения подтверждения ACK, можно передать байт данных или младший байт адреса.

15.4.9.1 Флаг WCOL

Если во время формирования бита повторный START производится попытка записи в SSPBUF, устанавливается бит WCOL, а запись не происходит.

Примечание. Поскольку MSSP не имеет стека событий, установка любого из младших 5 битов регистра SSPCON2 до завершения формирования бита повторный START запрещено.

Рисунок 15-20. Временная диаграмма формирования бита повторный START



15.4.10 Передача данных в режиме ведущего I^2C

Для инициализации передачи байта данных, 7-разрядного адреса или любой части 10-разрядного адреса нужно просто записать байт в регистр SSPBUF. В результате чего установится бит BF в '1', а BRG начнет формировать сигнал для передачи данных. Каждый передаваемый бит будет выдаваться на SDA по заднему фронту сигнала SCL. Низкий уровень на SCL удерживается в течение одно периода BRG. Данные должны поступать на SDA до прихода переднего фронта на SCL (см. раздел временных характеристик, параметр 106). После "отпускания" SCL в высокий уровень на время T_{BRG} данные должны удерживаться на SDA в том же состоянии. По окончании передачи 8-го бита сбрасывается флаг BF в '0', а ведущий "отпускает" SDA с тем, чтобы принять бит подтверждения. По заднему фронту 9-го такта значение ACK записывается в бит ACKSTAT регистра SSPCON2. В этот же момент устанавливается флаг SSPIF в '1', а BRG отключается до следующей операции на шине оставляя низкий уровень на SCL и отпуская SDA (смотрите рисунок 15-21).

15.4.10.1 Флаг BF

В режиме передачи данных бит BF (SSPSTAT<0>) аппаратно устанавливается в '1' после записи данных в регистр SSPBUF и аппаратно сбрасывается после передачи 8 бит данных.

15.4.10.2 Флаг WCOL

Если во время передачи данных производится попытка записи в регистр SSPBUF, устанавливается бит WCOL в '1', а запись не происходит. Бит WCOL сбрасывается программно.

15.4.10.3 Флаг ACKSTAT

В режиме передачи данных бит ACKSTAT(SSPCON2<6>) равен нулю, если ведомый сформировал подтверждение. Ведомый посылает подтверждение, если он распознал адрес (включая общий вызов) или корректно принял данные.

15.4.11 Прием данных в режиме ведущего I2C

Прием данных ведущем шины I^2C разрешается установкой бита RCEN(SSPCON2<3>) в '1'.

Примечание. При установке бита RCEN в '1' модуль MSSP должен находиться в режиме ожидания.

BRG начинает формировать тактовый сигнал SCL, для приема данных в сдвиговый регистр SSPSR. Каждый бит данных будет приниматься с SDA по заднему фронту SCL. По заднему фронту 8-го такта, значение из SSPSR переписывается в SPPBUF, устанавливается бит BF и SSPIF в '1', BGR останавливается, удерживая SCL в низком уровне, а модуль MSSP переходит в режим ожидания. После чтения регистра SSPBUF аппаратно сбрасывается бит BF в '0'. По окончанию приема, ведущий может сформировать бит подтверждения установкой бита ACKEN (SSPCON2<4>) в '1'.

15.4.11.1 Флаг BF

В режиме приема данных бит BF (SSPSTAT<0>) аппаратно устанавливается в '1' после загрузки данных в регистр SSPBUF и аппаратно сбрасывается после чтения регистра SSPBUF.

15.4.11.2 Флаг SSPOV

При приеме данных бит SSPOV устанавливается в '1', если в момент приема 8-го бита следующего байта бит BF=1 после приема предыдущего байта.

15.4.11.3 Флаг WCOL

Если во время приема данных производится попытка записи в регистр SSPBUF, устанавливается бит WCOL в '1', а запись не происходит. Бит WCOL сбрасывается программно.

Рисунок 15-21. Временная диаграмма передачи данных в режиме ведущего I^2C (7 или 10-разрядная адресация)

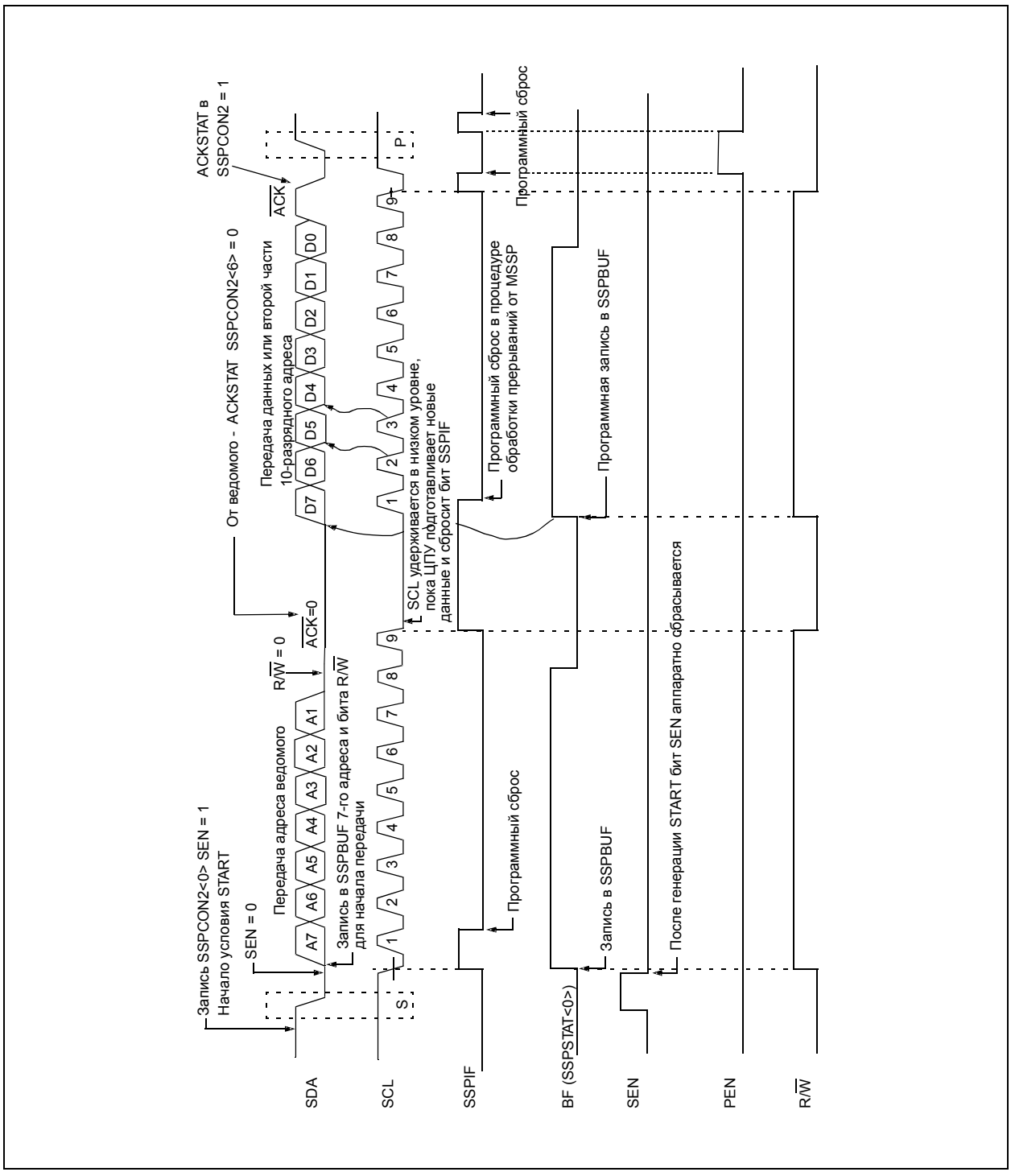
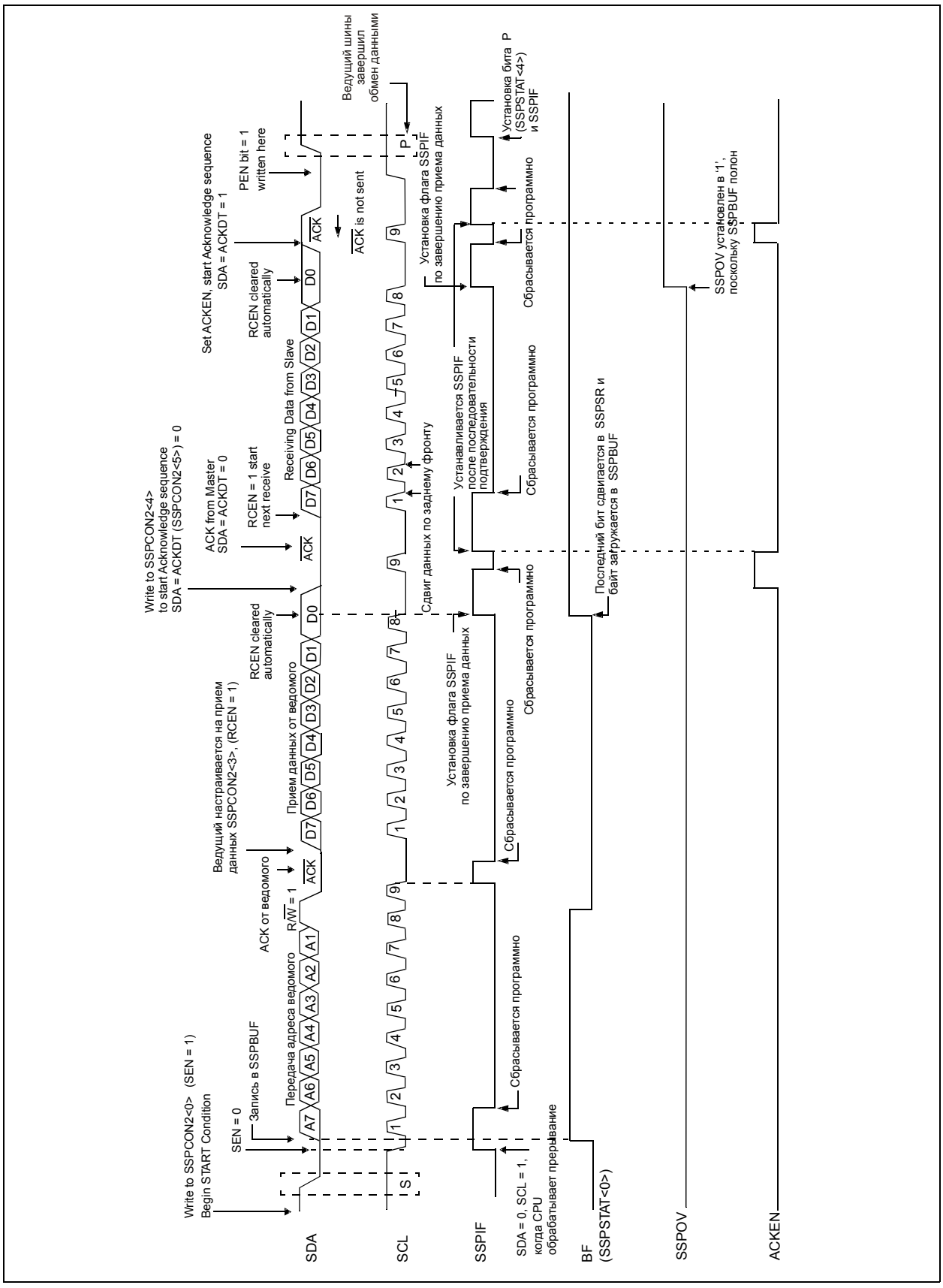


Рисунок 15-22. Временная диаграмма приема данных в режиме ведущего I^2C (7-разрядная адресация)



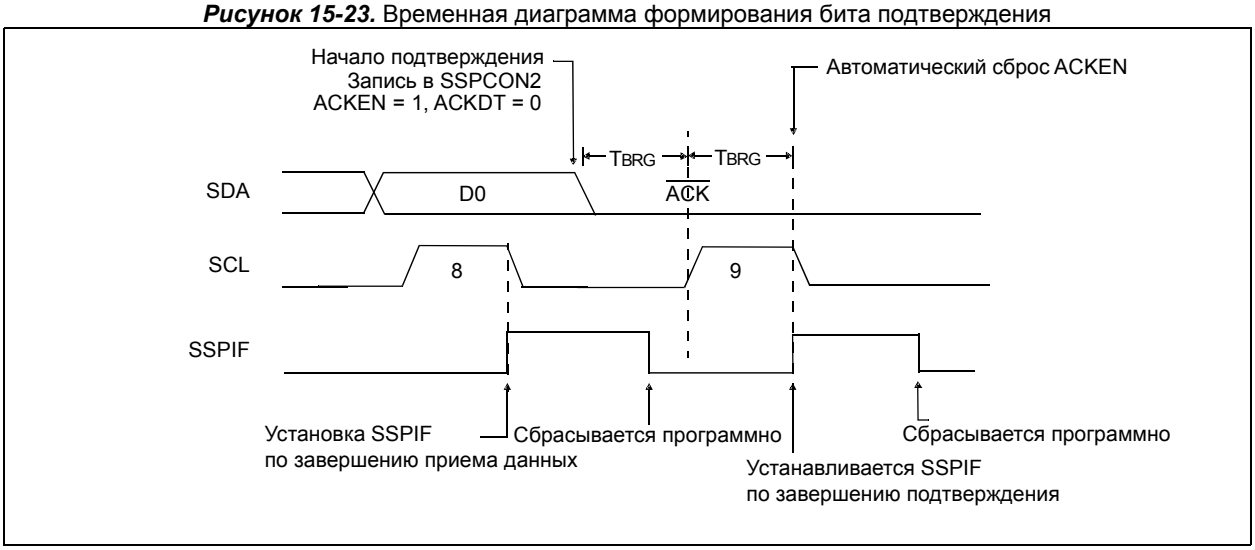
15.4.12 Формирование бита подтверждения в режиме ведущего I²C

Для инициализации формирования бита подтверждения на шине I²C необходимо установить бит ACKEN (SSPCON2<4>) в '1'. При установке этого бита на SCL выдается низкий уровень сигнала, а на SDA содержимое бита ACKDT. Если нужно подтвердить прием, бит ACKDT должен быть равен нулю. По окончанию счета BRG линия SCL "отпускается". Как только SCL перейдет из низкого уровня в высокий, BRG опять начнет счет. После окончания счета SCL переводится в низкий уровень, бит ACKEN автоматически сбрасывается в '0', устанавливается флаг прерывания SSPIF в '1', BGR останавливается, а модуль MSSP переходит в режим ожидания (см. рисунок 15-23).

15.4.12.1 Флаг WCOL

Если во время формирования бита подтверждения производится попытка записи в SSPBUF, устанавливается бит WCOL в '1', а запись не происходит.

Рисунок 15-23. Временная диаграмма формирования бита подтверждения



Примечание. T_{BRG} = один период генератора скорости обмена данными.

15.4.13 Формирование бита STOP в режиме ведущего I2C

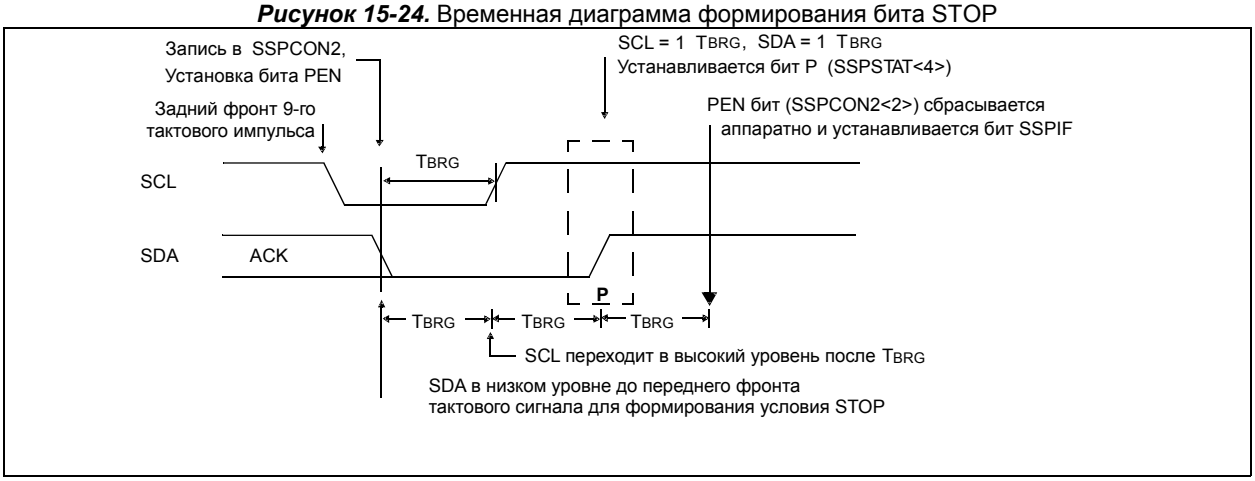
Чтобы инициировать формирование бита STOP, необходимо установить бит PEN (SSPCON2<2>) в '1'. По окончании приема/передачи данных, после прохождения заднего фронта тактового сигнала на SCL удерживается низкий уровень сигнала. При установки бита PEN ведущий выдает низкий уровень на линию SDA, перезагружает BRG и начинает счет до нуля. По окончании счета линия SCL "отпускается". Через время T_{BRG}, после установки высокого уровня на SCL, "отпускается" SDA. Когда на SDA появляется высокий уровень сигнала, устанавливаются биты P и SSPIF в '1', бит PEN автоматически сбрасывается в '0', а генератор BRG останавливается (см. рисунок 15-24).

Перед попыткой передать данные программное обеспечение должно проверить занятость шины (состояние битов S и P в регистре SSPSTAT). Если шина занята, то могут быть разрешены прерывания по обнаружению бита STOP.

15.4.13.1 Флаг WCOL

Если во время формирования бита STOP производится попытка записи в SSPBUF, устанавливается бит WCOL в '1', а запись не происходит.

Рисунок 15-24. Временная диаграмма формирования бита STOP



Примечание. T_{BRG} = один период генератора скорости обмена данными.

15.4.14 Работа в SLEEP режиме

Ведущий I^2C не может принимать адресные байты или байты данных в SLEEP режиме микроконтроллера.

15.4.15 Эффект сброса

При сбросе микроконтроллера модуль MSSP выключается, прекращается любой обмен данными.

15.4.16 Режим конкуренции

В режиме конкуренции, прерывания поле START и STOP позволяет определить, когда шина I^2C свободна. Биты S и P сбрасываются в '0' при сбросе микроконтроллера или при выключении модуля MSSP. Управление шиной может быть перехвачено, когда бит P=1 или шина простаивает (S=0 и P=0). Если шина занята, можно разрешить прерывания от MSSP для обнаружения бита STOP на шине.

При конкуренции линия SDA должна проверяться на соответствия уровня, при ожидаемом высоком уровне на выходе. Эта проверка производится автоматически, а результат помещается в бит BCLIF.

Арбитраж на шине I^2C может быть потерян во время:

- Передачи адреса
- Передачи данных
- Формирования бита START
- Формирования бита повторный START
- Формирования бита NACK

15.4.17 Режим конкуренции, арбитраж и конфликты шины

В режиме конкуренции необходимо поддерживать правила арбитража шины. Во время передачи адреса/данных на SDA ведущий может потерять арбитраж, если он формирует высокий уровень сигнала, а другой ведущий сформировал низкий уровень на SDA. При переходе SCL в высокий уровень, сигнал на SDA изменяться не может. Если на SDA ожидается высокий уровень, а в действительности низкий, значит возник конфликт шины. Обнаружив конфликт шины, ведущий устанавливает флаг прерывания BCLIF в '1', прекращает текущую операцию на шине и переводит порт I^2C в режим ожидания (см. рисунок 15-25).

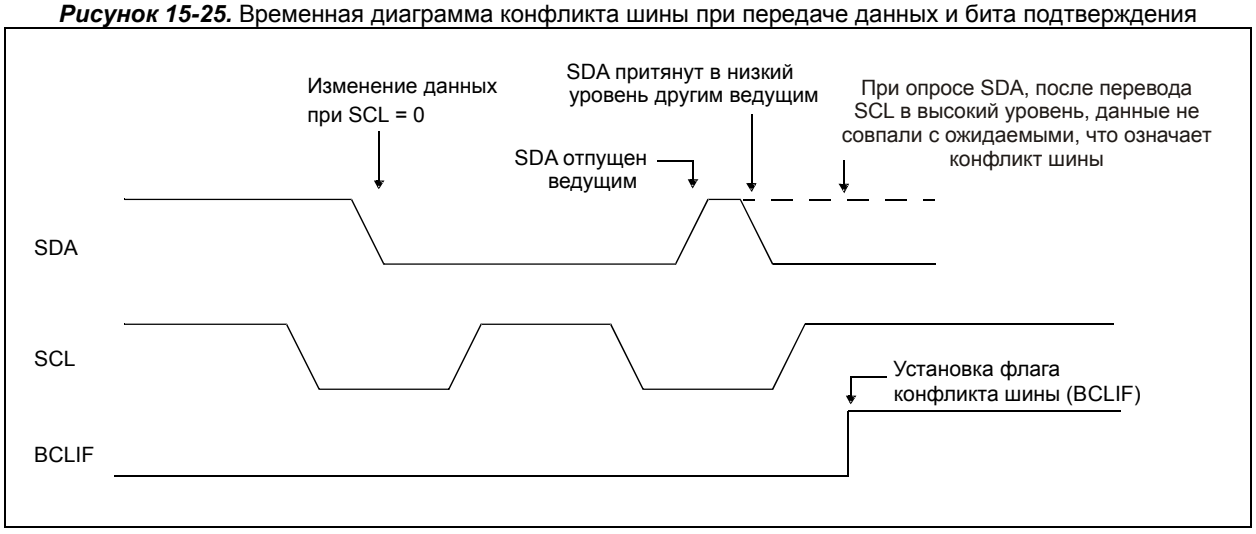
Если при возникновении конфликта шины выполнялась передача данных, она обрывается, устанавливается бит BF в '1', а линии SCL и SDA "отпускаются" в высокое состояние. В регистр SSPBUF может быть произведена запись, причем запись в SSPBUF инициирует передачу независимо от того, в какой момент передатчик отключился при возникновении конфликта шины. Если пользователь обрабатывает прерывания по конфликту шины, после освобождение шины он может продолжить обмен, сформировав бит START.

Если при возникновении конфликта выполнялось формирование бита START, повторный START, STOP или ACK, выполняемая операция обрывается, SCL и SDA "отпускаются", а соответствующий бит управления в SSPCON2 сбрасывается в '0'. Если пользователь обрабатывает прерывания по конфликту шины, после освобождение шины он может продолжить обмен, сформировав бит START.

Ведущий продолжает следить за состоянием шины, и при появлении бита STOP устанавливается флаг прерывания SSPIF в '1'.

В режиме конкуренции использование прерывания при обнаружении битов START и STOP позволяет определить занятость шины. Управление шиной может быть перехвачено при установленном бите P или сброшенных битах S и P.

Рисунок 15-25. Временная диаграмма конфликта шины при передаче данных и бита подтверждения



15.4.17.1 Конфликт шины при формировании бита START

Во время формирования бита START конфликт шины возникает если:

a) В начале START на SDA или SCL низкий уровень сигнала (см. рисунок 15-26)
b) На SCL низкий уровень появляется раньше чем на линии SDA (см. рисунок 15-27)

Во время формирования бита START сигналы SCL и SDA продолжают отслеживаться. Если SCL или SDA имеют низкий уровень сигнала, то формирование бита START прекращается, устанавливается флаг BCLIF в '1', а модуль MSSP переходит в режим ожидания (см. рисунок 15-26).

Бит START начинается при наличии высокого уровня сигнала на линиях SCL и SDA. Если на SCL появляется низкий уровень раньше, чем на SDA, возникает конфликт шины, поскольку это подразумевает, что другой ведущий пытается в это время передать данные.

Если во время счета BRG на SDA появляется низкий уровень сигнала, BRG сбрасывается, а на SDA формируется низкий уровень раньше времени (см. рисунок 15-28). Если же на SDA высокий уровень, низкий уровень формируется в конце счета BRG. Генератор BRG перезагружается и считает до нуля. Если в это время на SCL появится низкий уровень, конфликт шины не возникает. В конце счета BRG SCL переводится в низкий уровень.

Примечание. Конфликт шины во время START не возникает, потому что два или более ведущих могут сформировать START одновременно, но при этом один из них первым переведет SDA в низкий уровень. Конфликт шины не возникает, поскольку ведущие могут продолжить арбитраж во время передачи адреса, данных, формировании бита повторный START и STOP.

Рисунок 15-26. Временная диаграмма конфликта шины во время формирования бита START (только SDA)

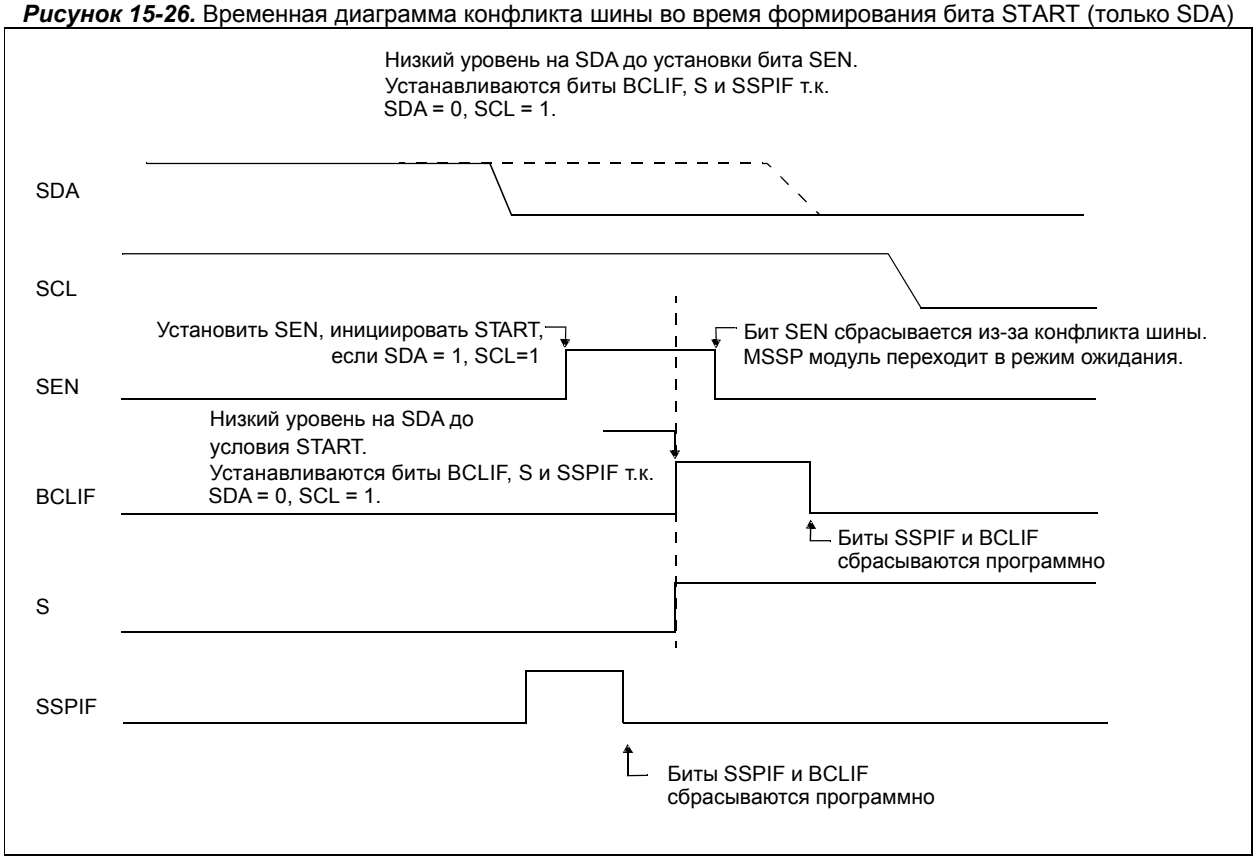


Рисунок 15-27. Временная диаграмма конфликта шины во время формирования бита START (SCL=0)

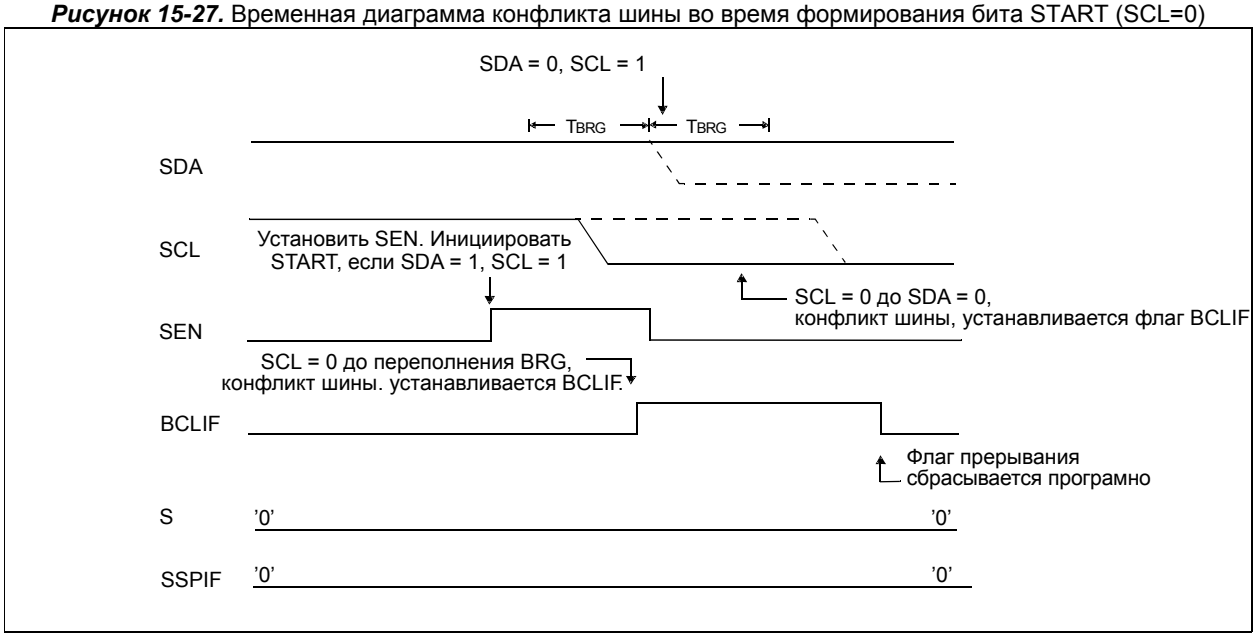
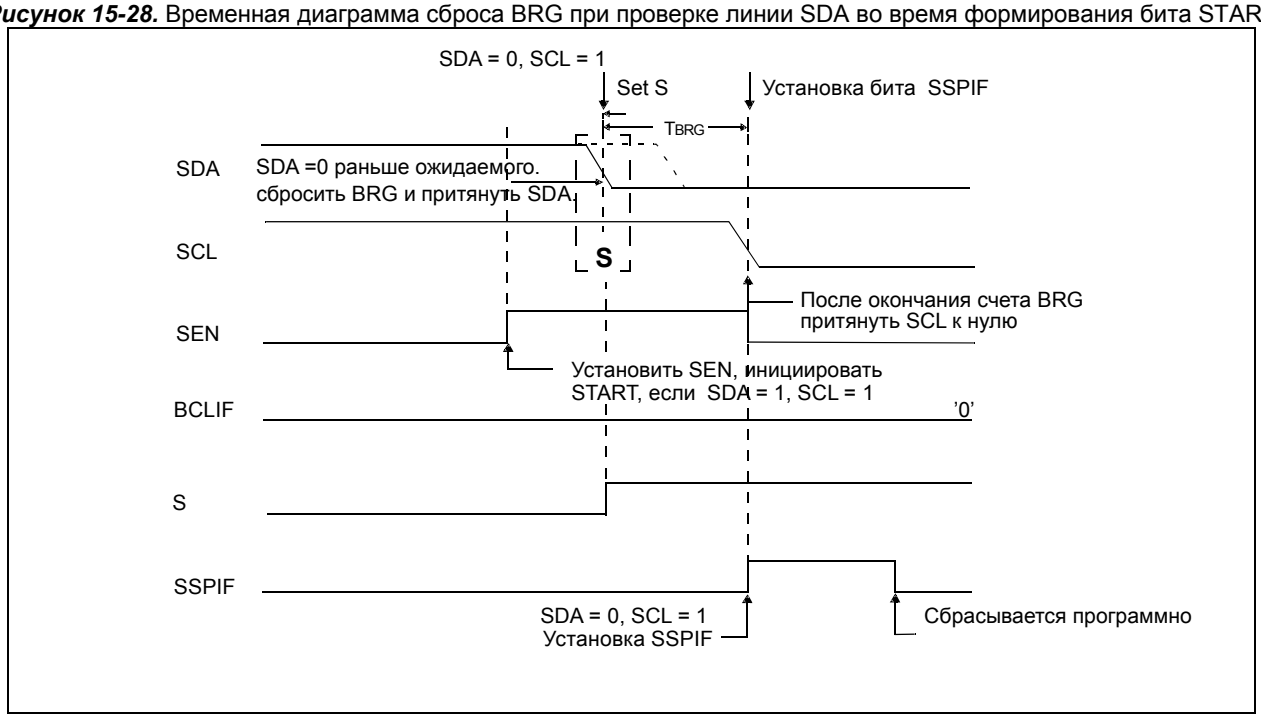


Рисунок 15-28. Временная диаграмма сброса BRG при проверке линии SDA во время формирования бита START



15.4.17.2 Конфликт шины при формировании бита повторный START

Во время формирования бита повторный START конфликт шины возникает если:

a) На SDA низкий уровень при переходе SCL из низкого уровня в высокий
b) SCL переходит в низкий уровень раньше SDA, что указывает на то, что другой ведущий пытается передать данные

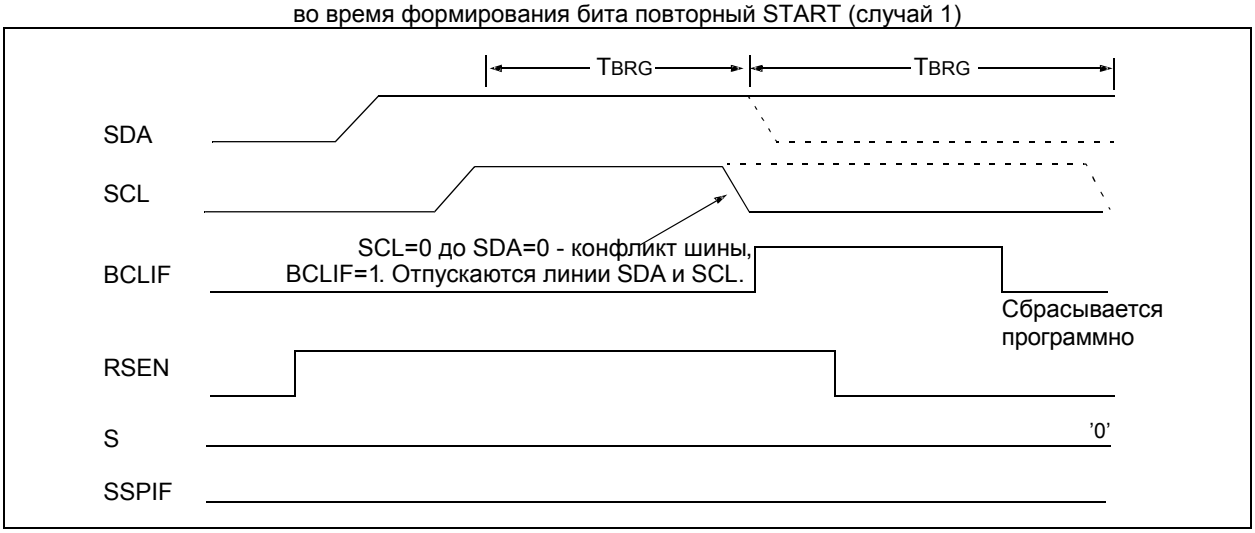
После "отпускания" линии SDA сигнал на выводе должен перейти в высокий уровень, после чего BRG перезагружается и начинает счет. Затем "отпускается" SCL и при появлении на нем высокого уровня опрашивается SDA. Если на SDA низкий уровень сигнала, значит произошел конфликт шины, т.е. другой ведущий пытается передать данные. Если на SDA высокий уровень, то BRG снова перезагружается и начинается счет. Если SDA переходит в низкий уровень до окончания счета, конфликт шины не происходит, поскольку два или более ведущих могут пытаться получить доступ к шине одновременно.

Если на линии SCL сигнал переходит в низкий уровень до окончания счета, а на SDA сохраняется высокий уровень, значит, произошел конфликт шины, т.е. другой ведущий пытается передать данные.

Если по окончании счета BGR на SCL и SDA высокий уровень, то SDA переводится в низкий уровень, а BRG перезагружается и начинает счет. По окончании счета, независимо от уровня сигнала на SCL он переводится в низкий уровень (см. рисунок 15-30).

Рисунок 15-29. Временная диаграмма конфликта шины во время формирования бита повторный START (случай 1)

Рисунок 15-30. Временная диаграмма конфликта шины во время формирования бита повторный START (случай 1)



15.4.17.3 Конфликт шины при формировании бита STOP

Во время формирования бита STOP конфликт шины возникает если:

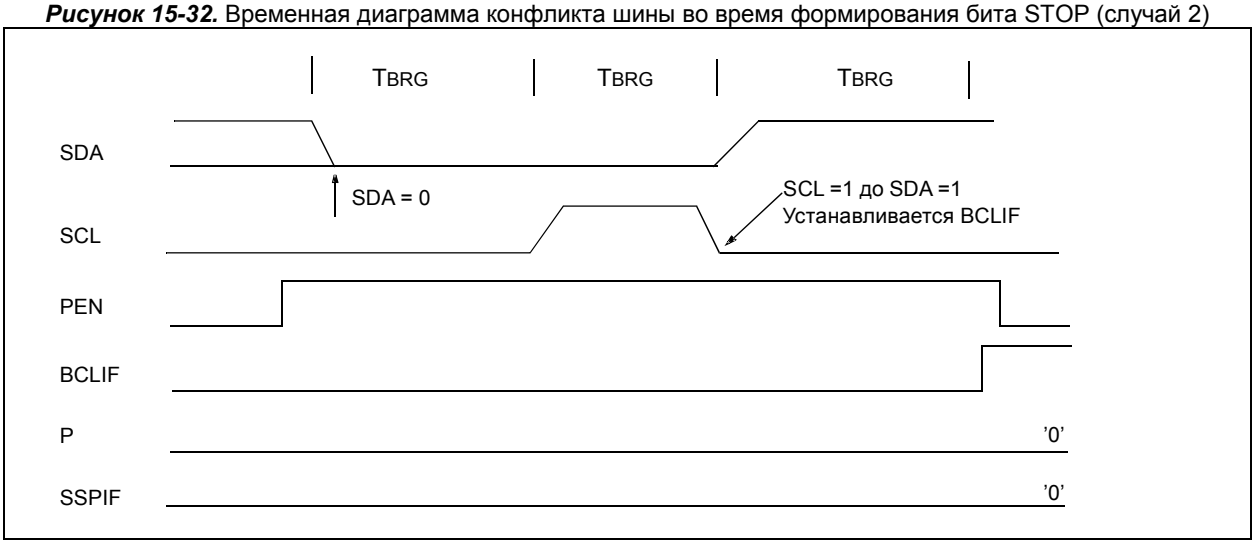
a) После "отпускания" линии SDA и окончания счета BRG на SDA по-прежнему низкий уровень сигнала (см. рисунок 15-31)

b) После "отпускания" линии SDA сигнал на SCL переходит в низкий уровень до того, как на SDA перейти в высокий уровень (см. рисунок 15-32)

Формирование бита STOP начинается с перевода линии SDA в низкий уровень, затем SCL "отпускается". После появления на SCL высокого уровня BRG перезагружается и начинает счет. По окончании счета SDA "отпускается", BRG перезагружается и снова начинает счет и опрашивает SDA. Если на нем низкий уровень или на SCL появился низкий уровень до перехода SDA в высокий, значит, произошел конфликт шины, т.е. другой ведущий пытается передать данные.

Рисунок 15-31. Временная диаграмма конфликта шины во время формирования бита STOP (случай 1)

Рисунок 15-32. Временная диаграмма конфликта шины во время формирования бита STOP (случай 2)



16. Адресуемый универсальный синхронно-асинхронный приемопередатчик (USART)

USART – это один из модулей последовательного порта ввода/вывода (имеет существенные отличия от модуля MSSP), который может работать в полнодуплексном асинхронном режиме для связи с терминалами, персональными компьютерами или синхронном полудуплексном режиме для связи с микросхемами ЦАП, АЦП, последовательными EEPROM и т.д.

USART может работать в одном из трех режимов:

- Асинхронный, полный дуплекс
- Ведущий синхронный, полудуплекс
- Ведомый синхронный, полудуплекс

Биты SPEN (RCSTA<7>) и TRISC<7:6> должны быть установлены в ‘1’ для использования выводов RC6/TX/CK и RC7/RX/DT в качестве портов универсального синхронно-асинхронного приемопередатчика. Модуль USART поддерживает режим детектирования 9-разрядного адреса для работы в сетевом режиме.

Для настройки модуля USART предусмотренные два регистра: TXSTA – регистр управления и статуса передатчика USART; RCSTA – регистр управления и статуса приемника USART.

Регистр 16-1. TXSTA: Регистр управления и статуса передатчика

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D
Бит 7							Бит 0

бит 7 **CSRC:** Выбор источника тактового сигнала
Синхронный режим
1 = ведущий, внутренний тактовый сигнал от BRG
0 = ведомый, внешний тактовый сигнал с входа CK

Асинхронный режим
Не имеет значения

бит 6 **TX9:** Разрешение 9-разрядной передачи
1 = 9-разрядная передача
0 = 8-разрядная передача

бит 5 **TXEN:** Разрешение передачи
1 = разрешена
0 = запрещена
Примечание. В синхронном режиме биты SREN/CREN отменяют действие бита TXEN.

бит 4 **SYNC:** Режим работы USART
1 = синхронный
0 = асинхронный

бит 3 **Не используется:** читается как ‘0’

бит 2 **BRGH:** Выбор высокоскоростного режима
Синхронный режим
Не имеет значения

Асинхронный режим
1 = высокоскоростной режим
0 = низкоскоростной режим

бит 1 **TRMT:** Флаг очистки сдвигового регистра передатчика TSR
1 = TSR пуст
0 = TSR полон

бит 0 **TX9D:** 9-й бит передаваемых данных (может использоваться для программной проверки четности)

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 16-2. RCSTA: Регистр управления и статуса приемника

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
Бит 7							Бит 0

бит 7 **SPEN:** Разрешение работы последовательного порта
1 = модуль USART включен (выводы RX/DT, TX/CK подключены к USART)
0 = модуль USART выключен

бит 6 **RX9:** Разрешение 9-разрядного приема
1 = 9-разрядный прием
0 = 8-разрядный прием

бит 5 **SREN:** Разрешение одиночного приема
Синхронный режим
1 = разрешен одиночный прием
0 = запрещен одиночный прием
Сбрасывается в ‘0’ по завершению приема.
Примечание. В режиме ведомого не имеет значения

Асинхронный режим
Не имеет значения

бит 4 **CREN:** Разрешение приема
Синхронный режим
1 = прием разрешен (при установке бита CREN автоматически сбрасывается бит SREN)
0 = прием запрещен

Асинхронный режим
1 = прием разрешен
0 = прием запрещен

бит 3 **ADDEN:** Разрешение детектирования адреса
Асинхронный 9-разрядный прием (RX9=1)
1 = детектирование адреса разрешено. Если бит RSR<8>=1, то генерируется прерывание и загружается приемный буфер.
0 = детектирование адреса запрещено. Принимаются все байты, девятый бит может использоваться для проверки четности.

бит 2 **FERR:** Ошибка кадра, сбрасывается при чтении регистра RCREG
1 = произошла ошибка кадра
0 = ошибки кадра не было

бит 1 **OERR:** Ошибка переполнения внутреннего буфера, устанавливается в ‘0’ при сбросе бита CREN
1 = произошла ошибка переполнения
0 = ошибки переполнения не было

бит 0 **RX9D:** 9-й бит принятых данных (может использоваться для программной проверки четности)

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

16.1 Генератор скорости обмена USART BRG

BRG используется для работы USART в синхронном ведущем и асинхронном режимах. BRG представляет собой отдельный 8-разрядный генератор скорости обмена в бодах, период которого определяется значением в регистре SPBRG. В асинхронном режиме бит BRGH (TXSTA<2>) тоже влияет на скорость обмена (в синхронном режиме бит BRGH игнорируется). В таблице 16-1 указаны формулы для вычисления скорости обмена в бодах при различных режимах работы модуля USART (относительно внутреннего тактового сигнала микроконтроллера).

Учитывая требуемую скорость и F_{OSC}, выбирается самое близкое целое значение для записи в регистр SPBRG (от 0 до 255), рассчитанное по формулам приведенным в таблице 16-1. Затем рассчитывается ошибка скорости обмена.

В примере 16-1 показан расчет значения для регистра SPBRG и погрешность скорости обмена для следующих условий:

F_{OSC} = 16 МГц
Скорость приема/передачи данных = 9600 бит/с
BRGH = 0
SYNC = 0

В некоторых случаях может быть выгодно использовать высокоскоростной режим работы USART (BRGH=1), поскольку уравнение FOSC / (16 (X + 1)) позволяет уменьшить погрешность скорости.

Запись нового значения в регистр SPBRG сбрасывает таймер BRG, гарантируя немедленный переход на новую скорость.

16.1.1 Выборка

Сигнал с входа RX/DT опрашивается цепью мажоритарного детектора три раза за такт передачи, чтобы определить, высокого или низкого уровня сигнал присутствует на входе.

Пример 16-1. Расчет значения для регистра SPBRG и погрешность скорости обмена

Желаемое значение скорости = F_{OSC} / (64 (X + 1))

X = (F_{OSC} / Желаемое значение скорости) / 64) –1

X = (16 000 000 / 9600) / 64) –1

X = [25.042] = 25

Вычисленное значение скорости = 16 000 000 / (64 (25 + 1)) = 9615

Ошибка = 100 x (Вычисленное – Желаемое) / Желаемое значение скорости

Ошибка = 100 x (9615 – 9600) / 9600 = 0.16%

Таблица 16-1. Формулы расчета скорости обмена данными

SYNC	BRGH = 0	BRGH = 1
0	(Асинхронный) Скорость обмена = F_{OSC} / (64 (X + 1))	(Асинхронный) Скорость обмена = F_{OSC} / (16 (X + 1))
1	(Синхронный) Скорость обмена = F_{OSC} / (4 (X + 1))	(Синхронный) Скорость обмена = F_{OSC} / (4 (X + 1))

X = значение регистра SPBRG (от 0 до 255)

Таблица 16-2. Регистры и биты, связанные с работой генератора BRG

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FAFh	SPBRG	Регистр скорости обмена USART								0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

Таблица 16-3 Скорость обмена в асинхронном режиме (BRGH=0)

Скорость обмена (К)	F_{OSC} = 40 МГц			F_{OSC} = 20 МГц			F_{OSC} = 16 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	Нет	-	-
1,2	Нет	-	-	1,221	+1,73	255	1,202	+0,16	207
2,4	2.441	-1.70	255	2,404	+0,16	129	2,404	+0,16	103
9,6	9.615	-0.16	64	9,469	-1,36	32	9,615	+0,16	25
19,2	18.94	+1.38	32	19,53	+1,73	15	19,23	+0,16	12
76,8	78.13	-1.70	7	78,13	+1,73	3	83,33	+8,51	2
96	89.29	+7.52	6	104,2	+8,51	2	Нет	-	-
300	312.5	-4.00	1	312,5	+4,17	0	Нет	-	-
500	625.0	-20.0	0	Нет	-	-	Нет	-	-
Максим.	625,0	-	0	312,5	-	0	250	-	0
Миним.	2,441	-	255	1,221	-	255	0,977	-	255

Скорость обмена (К)	F_{OSC} = 10 МГц			F_{OSC} = 7,15909 МГц			F_{OSC} = 5,0688 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	0,31	+3,13	255
1,2	1,202	+0,16	129	1,203	+0,23	92	1,2	0	65
2,4	2,404	+0,16	64	2,380	-0,83	46	2,4	0	32
9,6	9,615	+1,73	15	9,322	-2,90	11	9,9	+3,13	7
19,2	19,53	+1,73	7	18,64	-2,90	5	19,8	+3,13	3
76,8	78,13	+1,73	1	Нет	-	-	79,2	+3,13	0
96	Нет	-	-	Нет	-	-	Нет	-	-
300	Нет	-	-	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	156,3	-	0	111,9	-	0	79,2	-	0
Миним.	0,6104	-	255	0,437	-	255	0,3094	-	255

Скорость обмена (К)	F_{OSC} = 4 МГц			F_{OSC} = 3,579545 МГц			F_{OSC} = 1 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	0,3005	-0,17	207	0,301	+0,23	185	0,300	+0,16	51
1,2	1,202	+1,67	51	1,190	-,083	46	1,202	+0,16	12
2,4	2,404	+1,67	25	2,432	+1,32	22	2,232	-6,99	6
9,6	Нет	-	-	9,322	-2,90	5	Нет	-	-
19,2	Нет	-	-	18,64	-2,90	2	Нет	-	-
76,8	Нет	-	-	Нет	-	-	Нет	-	-
96	Нет	-	-	Нет	-	-	Нет	-	-
300	Нет	-	-	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	62,500	-	0	55,93	-	0	15,63	-	0
Миним.	3,906	-	255	0,2185	-	255	0,0610	-	255

Скорость обмена (К)	F_{OSC} = 32,768 кГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	0,256	-14,67	1
1,2	Нет	-	-
2,4	Нет	-	-
9,6	Нет	-	-
19,2	Нет	-	-
76,8	Нет	-	-
96	Нет	-	-
300	Нет	-	-
500	Нет	-	-
Максим.	0,512	-	0
Миним.	0,0020	-	255

Таблица 16-4. Скорость обмена в асинхронном режиме (BRGH=1)

Скорость обмена (К)	F_{OSC} = 40 МГц			F_{OSC} = 20 МГц			F_{OSC} = 16 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	9.766	-1.70	255	9,615	+0,16	129	9,615	+0,16	103
19,2	19.231	-0.16	129	19,230	+0,16	64	19,230	+0,16	51
38,4	38.461	-0.16	64	37,878	-1,36	32	38,461	+0,16	25
57,6	58.139	-0.93	42	56,818	-1,36	21	58,823	+2,12	16
115,2	113.64	1.38	21	113,636	-1,36	10	111,111	-3,55	8
250	250	0	9	250	0	4	250	0	3
625	625	0	3	625	0	1	Нет	-	-
1250	1250	0	1	1250	0	0	Нет	-	-

Скорость обмена (К)	F_{OSC} = 10 МГц			F_{OSC} = 7,16 МГц			F_{OSC} = 5,068 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	9,615	+0,16	64	9,520	-0,83	46	9,6	0	32
19,2	18,939	-1,36	32	19,454	+1,32	22	18,645	-2,94	16
38,4	39,062	+1,7	15	37,286	-2,90	11	39,6	+3,12	7
57,6	56,818	-1,36	10	55,930	-2,90	7	52,8	-8,33	5
115,2	125	+8,51	4	111,860	-2,90	3	105,6	-8,33	2
250	Нет	-	-	Нет	-	-	Нет	-	-
625	625	0	0	Нет	-	-	Нет	-	-
1250	Нет	-	-	Нет	-	-	Нет	-	-

Скорость обмена (К)	F_{OSC} = 4 МГц			F_{OSC} = 3,579 МГц			F_{OSC} = 1 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	Нет	-	-	9,727	+1,32	22	8,928	-6,99	6
19,2	1,202	+0,17	207	18,643	-2,90	11	20,833	+8,51	2
38,4	2,403	+0,13	103	37,286	-2,90	5	31,25	-18,61	1
57,6	9,615	+0,16	25	55,930	-2,90	3	62,5	+8,51	0
115,2	19,231	+0,16	12	111,860	-2,90	1	Нет	-	-
250	Нет	-	-	223,721	-10,51	0	Нет	-	-
625	Нет	-	-	Нет	-	-	Нет	-	-
1250	Нет	-	-	Нет	-	-	Нет	-	-

Скорость обмена (К)	F_{OSC} = 32,768 кГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	Нет	-	-
19,2	Нет	-	-
38,4	Нет	-	-
57,6	Нет	-	-
115,2	Нет	-	-
250	Нет	-	-
625	Нет	-	-
1250	Нет	-	-

16.2 Асинхронный режим USART

В этом режиме USART использует стандартный формат NRZ: один стартовый бит, восемь или девять битов данных и один стоповый бит. Наиболее часто встречается 8-разрядный формат передачи данных. Интегрированный 8-разрядный генератор BRG позволяет получить стандартные скорости передачи данных. Генератор скорости обмена может работать в одном из двух режимов: высокоскоростной (x16 BRGH=1 TXSTA<2>), низкоскоростной (x64 BRGH=0 TXSTA<2>). Приемник и передатчик последовательного порта работают независимо друг от друга, но используют один и тот же формат данных и одинаковую скорость обмена. Бит четности аппаратно не поддерживается, но может быть реализован программно, применяя 9-разрядный формат данных. Все данные передаются младшим битом вперед. В SLEEP режиме микроконтроллера модуль USART(асинхронный режим) выключен.

Выбор асинхронного режима USART выполняется сбросом бита SYNC в ‘0’ (TXSTA<4>).

Модуль USART в асинхронном режиме состоит из следующих элементов:

- Генератор скорости обмена
- Цепь опроса
- Асинхронный передатчик
- Асинхронный приемник

16.2.1 Асинхронный передатчик USART

Структурная схема асинхронного передатчика USART показана на рисунке 16-1. Главным в передатчике является сдвиговый регистр TSR, который получает данные из буфера передатчика TXREG. Данные в регистр TXREG загружаются программно. После передачи стопового бита предыдущего байта, в последнем машинном такте цикла BRG, TSR загружается новым значением из TXREG (если оно присутствует), после чего устанавливается флаг прерывания TXIF. Прерывание может быть разрешено или запрещено битом TXIE. Флаг TXIF устанавливается независимо от состояния бита TXIE и не может быть сброшен в ‘0’ программно. Очистка флага TXIF происходит только после загрузки новых данных в регистр TXREG. Аналогичным образом бит TRMT (TXSTA<1>) отображает состояние регистра TSR. Бит TRMT доступен только на чтение и не может вызвать генерацию прерывания.

Примечания:
1. Регистр TSR не отображается на память и не доступен для чтения.
2. Флаг TXIF устанавливается в ‘1’ только, когда бит TXEN=1 и сбрасывается автоматически в '0' после загрузки новых данных в регистр TXREG.

Рекомендованная последовательность действий для передачи данных в асинхронном режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 16.1)
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’
3. Если необходимо, разрешить прерывания установкой бита TXIE в ‘1’
4. Если передача 9-разрядная, установить бит TX9 в ‘1’
5. Разрешить передачу установкой бита TXEN в ‘1’, автоматически устанавливается флаг TXIF
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D
7. Записать данные в регистр TXREG (начало передачи данных)
8. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в ‘1’

Рисунок 16-1. Структурная схема асинхронного передатчика USART

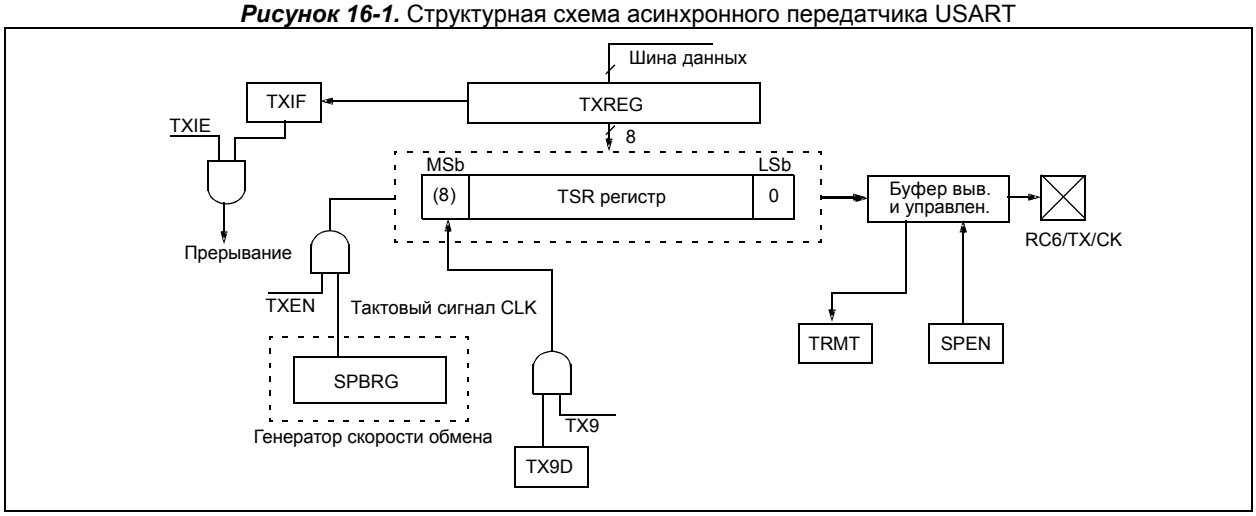


Рисунок 16-2. Временная диаграмма асинхронной передачи данных

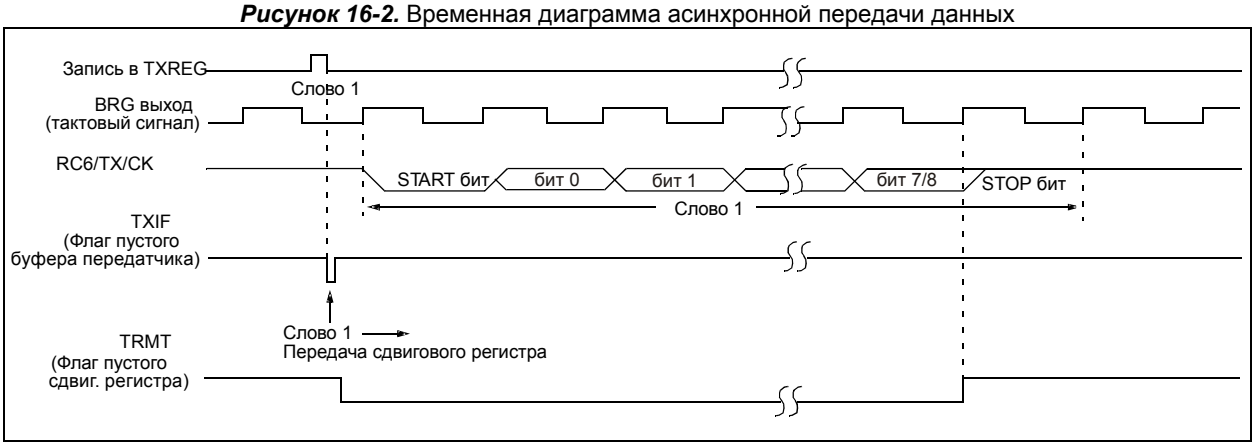


Рисунок 16-3. Временная диаграмма слитной асинхронной передачи (последовательная передача двух байт)

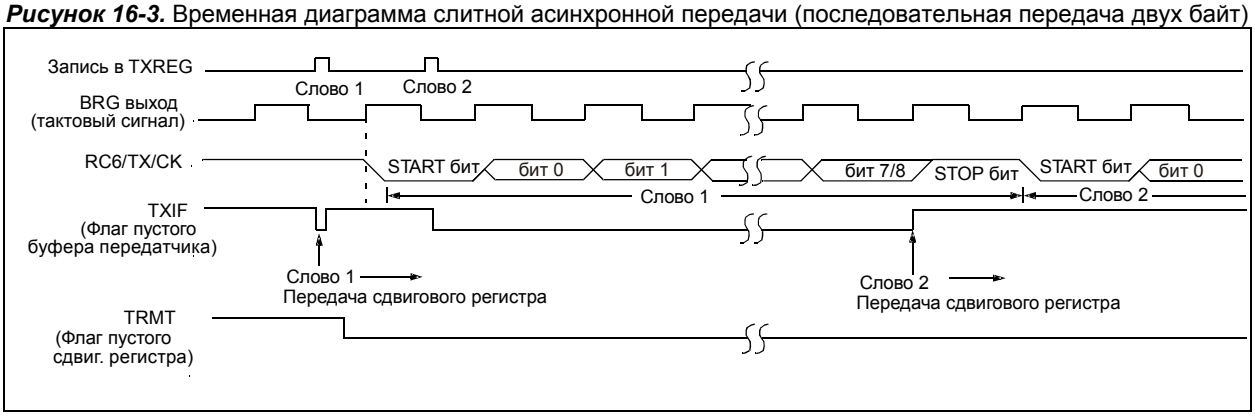


Таблица 16-5. Регистры и биты, связанные с работой передатчика USART в асинхронном режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FADh	TXREG	Регистр передатчика USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 -010
FAFh	SPBRG	Регистр скорости обмена USART								0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

16.2.2 Асинхронный приемник USART

Структурная схема асинхронного приемника USART показана на рисунке 16-4. Данные подаются на вход RC7/RX/DT в блок восстановления данных, представляющий собой скоростной сдвиговый регистр, работающий на частоте в 16 раз превышающей скорость передачи или F_{OSC}. Этот режим обычно применяется для работы по интерфейсу RS-232.

Рекомендованные действия при приеме данных в асинхронном режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 16.1)
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’
3. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’
4. Если прием 9-разрядный, установить бит RX9 в ‘1’
5. Разрешить прием установкой бита CREN в ‘1’
6. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE
7. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки
8. Считать 8 бит данных из регистра RCREG
9. При возникновении ошибки переполнения сбросить бит CREN в ‘0’
10. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в ‘1’

16.2.3 Настройка 9-разрядного асинхронного приема с детектированием адреса

Этот режим обычно применяется для работы по интерфейсу RS-485.

Рекомендованная последовательность действия при использовании детектора адреса:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 16.1)
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’
3. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’
4. Установить бит RX9 в ‘1’ для включения 9-разрядного приема
5. Установить бит ADDEN в ‘1’ для разрешения детектирования адреса
6. Разрешить прием установкой бита CREN в ‘1’
7. Ожидать установку бита RCIF или прерывание, если оно разрешено битами RCIE, GIE
8. Считать 9-й бит данных из регистра RCSTA и проверить возникновение ошибки
9. Считать 8 бит данных из регистра RCREG для проверки адресации устройства
10. При возникновении ошибки переполнения сбросить бит CREN в ‘0’
11. Если принятый адрес соответствует адресу устройства, сбросить биты ADDEN и RCIF в ‘0’ для начала приема данных

Рисунок 16-4. Структурная схема асинхронного приемника USART

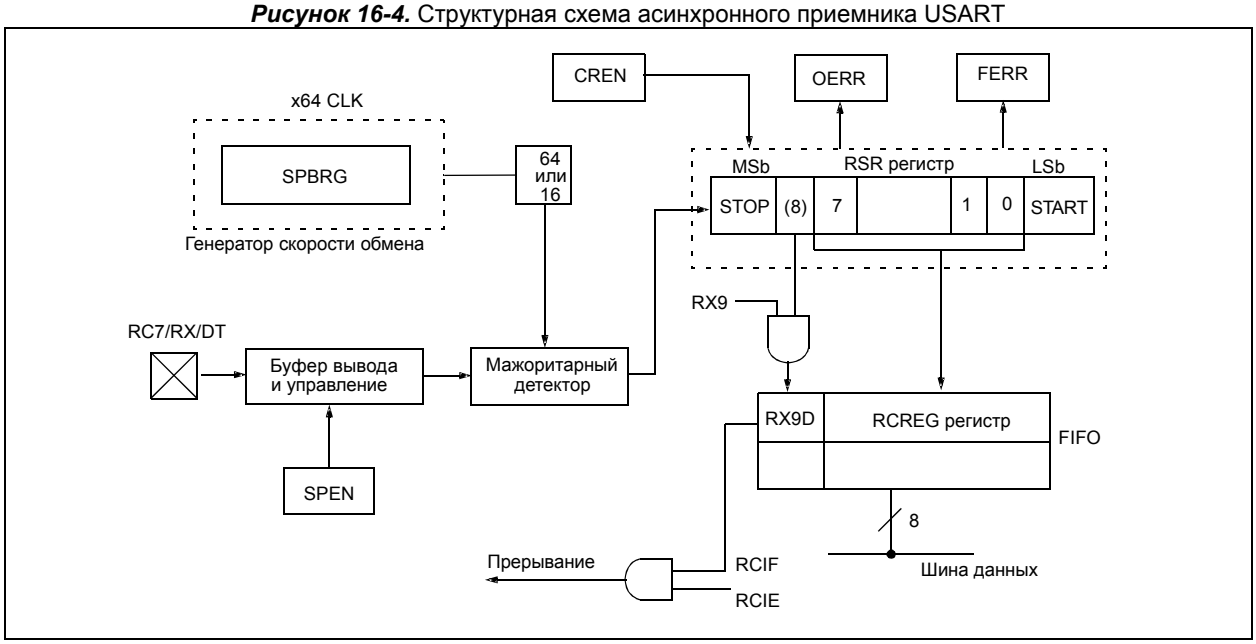
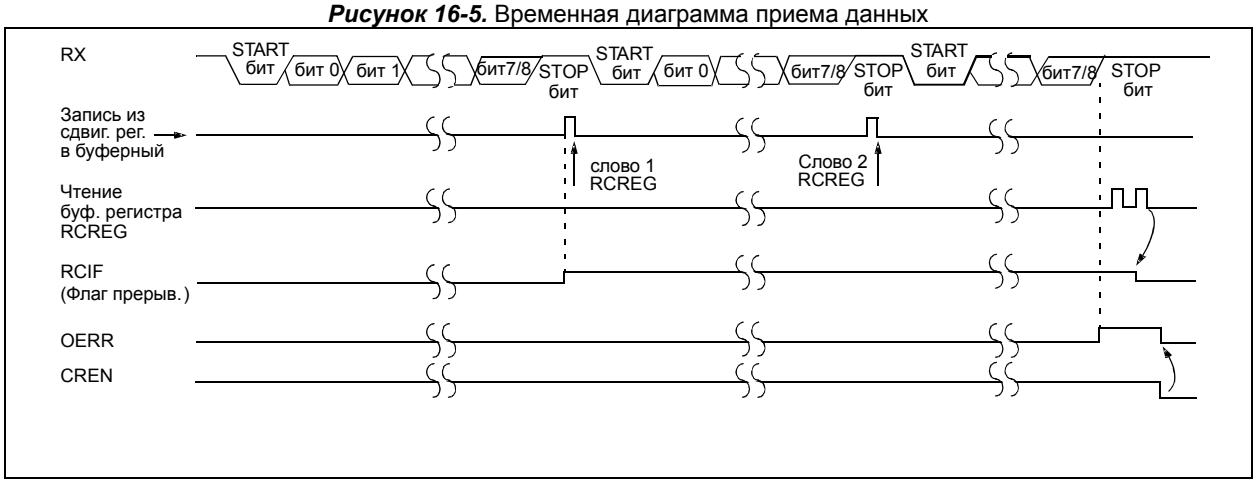


Рисунок 16-5. Временная диаграмма приема данных



Примечание. На временной диаграмме показан последовательный прием трех байт. Регистр RCREG (приемный буфер) читается после приема трех байт, поэтому устанавливается бит OERR в '1'.

Таблица 16-6. Регистры и биты, связанные с работой приемника USART в асинхронном режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FAEh	RCREG	Регистр приемника USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010
FAFh	SPBRG	Регистр скорости обмена USART								0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

16.3 Синхронный ведущий режим USART

В ведущем синхронном режиме данные передаются полудуплексом, т.е. прием и передача не происходит одновременно. При передаче запрещен прием и наоборот. Синхронный режим включается установкой бита SYNC (TXSTA<4>) в ‘1’. Также необходимо включить модуль USART, установкой бита SPEN (RCSTA<7>) в ‘1’, для настройки портов ввода/вывода RC6/TX/CK и RC7/RX/DT в качестве тактового сигнала CK и линии данных DT соответственно. В режиме ведущего модуль USART формирует тактовый сигнал CK. Выбор режима ведущего производится установкой бита CSRC (TXSTA<7>) в ‘1’.

16.3.1 Передача синхронного ведущего

Структурная схема передатчика USART показана на рисунке 16-1. Главным в передатчике является сдвиговый регистр TSR. Сдвиговый регистр получает данные из буфера передатчика TXREG. В регистр TSR не загружаются новые данные, пока не будет передан последний бит предыдущего байта. После передачи последнего бита предыдущего байта TSR загружается новым значением из TXREG (если оно присутствует), и устанавливается флаг прерывания TXIF (PIR1<4>). Это прерывание может быть разрешено/запрещено битом TXIE (PIE1<4>). Флаг TXIF устанавливается вне зависимости от состояния бита TXIE и может быть сброшен только загрузкой новых данных в регистр TXREG. Также, как TXIF отображает состояние TXREG, бит TRMT (TXSTA<1>) показывает состояние регистра TSR. Этот бит не вызывает генерацию прерывания, доступен только на чтение и устанавливается в ‘1’, когда регистр TSR пуст. Регистр TSR не отображается на память и не доступен пользователю.

Рекомендованная последовательность действий для передачи данных в синхронном ведущем режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 16.1)
2. Выбрать синхронный ведущий режим установкой битов SYNC, SPEN и CSRC в ‘1’
3. Если необходимо, разрешить прерывания установкой бита TXIE в ‘1’
4. Если передача 9-разрядная, установить бит TX9 в ‘1’
5. Разрешить передачу установкой бита TXEN в ‘1’
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D
7. Записать данные в регистр TXREG
8. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в ‘1’

Таблица 16-7. Регистры и биты, связанные с работой передатчика USART в синхронном ведущем режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FADh	TXREG	Регистр передатчика USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 −010
FAFh	SPBRG	Регистр скорости обмена USART								0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

Рисунок 16-6. Временная диаграмма синхронной передачи двух 8-разрядных слов (SPBRG = 0)

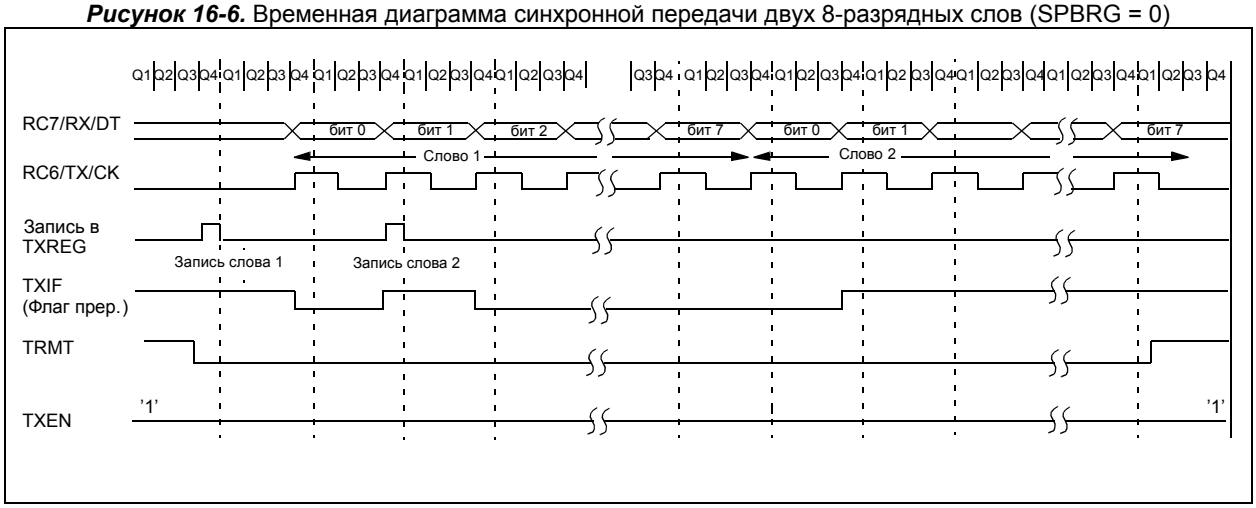
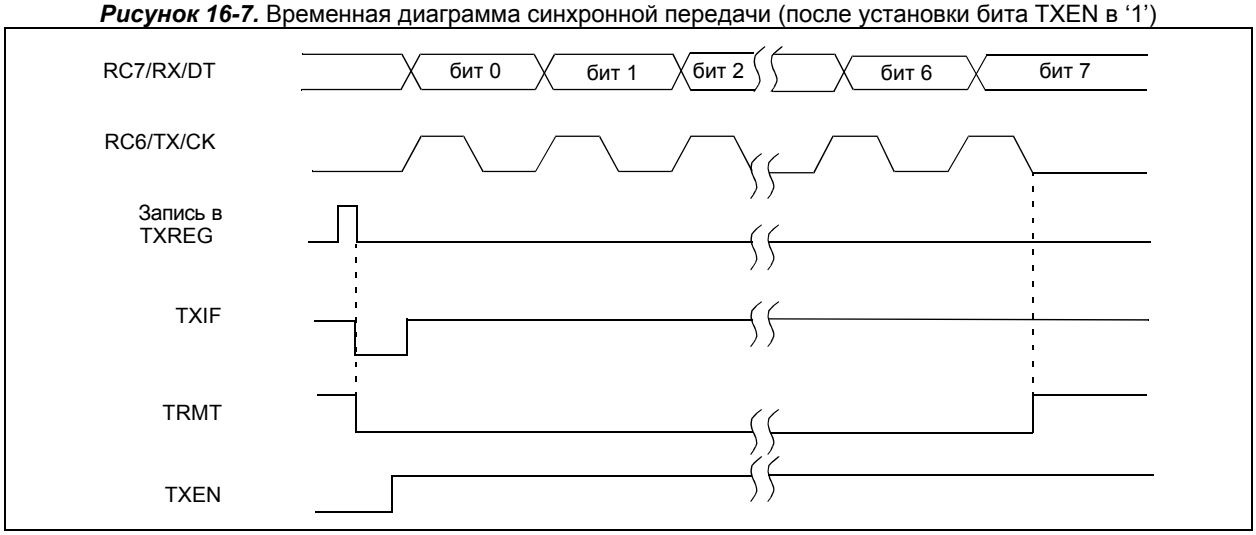


Рисунок 16-7. Временная диаграмма синхронной передачи (после установки бита TXEN в ‘1’)



16.3.2 Прием синхронного ведущего

В синхронном режиме прием разрешается установкой битов CREN (RCSTA<4>) или SREN (RCSTA<5>) в ‘1’. Линия данных RC7/RX/DT опрашивается по заднему фронту тактового сигнала. Если бит SREN установлен в ‘1’, то принимается одиночное слово. Если бит CREN установлен в ‘1’, то в не зависимости от состояния бита SREN будет производиться поточный прием данных, пока CREN не будет равен нулю. Если оба бита (CREN, SREN) установлены в ‘1’, то приоритет отдается биту CREN.

Рекомендованные действия при приеме данных в синхронном ведущем режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 16.1)
2. Выбрать синхронный ведущий режим установкой битов SYNC, SPEN и CSRC в ‘1’
3. Сбросить биты SREN и CREN в ‘0’
4. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’
5. Если прием 9-разрядный, установить бит RX9 в ‘1’
6. Если необходимо выполнить одиночный прием, установите бит SREN в ‘1’. Для поточного приема установите бит CREN в ‘1’
7. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE
8. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки
9. Считать 8 бит данных из регистра RCREG
10. При возникновении ошибки переполнения сбросить бит CREN в ‘0’
11. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в ‘1’

Таблица 16-8. Регистры и биты, связанные с работой приемника USART в синхронном ведущем режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FAEh	RCREG	Регистр приемника USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 -010
FAFh	SPBRG	Регистр скорости обмена USART								0000 0000

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

Рисунок 16-8. Временная диаграмма синхронного приема в режиме ведущего (SREN = 1, SPBRG = 0)

16.4 Синхронный ведомый режим USART

Режим ведомого отличается от ведущего тем, что микроконтроллер использует тактовый сигнал с входа RC6/TX/CK, а не формирует его самостоятельно. Это позволяет устройству принимать и передавать данные в SLEEP режиме. Выбрать режим ведомого можно сбросом бита CSRC (TXSTA<7>) в ‘0’.

16.4.1 Передача синхронного ведомого

Работа передатчика в обоих синхронных режимах одинакова, за исключением работы ведомого в SLEEP режиме микроконтроллера.

Если в TXREG были записаны два слова подряд и исполнена команда SLEEP, выполняются следующие действия:

a) Первое слово сразу записывается в TSR и передается по мере прихода тактового сигнала
b) Второе слово остается в TXREG
c) Флаг TXIF не устанавливается в ‘1’
d) После передачи первого слова, второе слово передается из TXREG в TSR, и устанавливается флаг TXIF в ‘1’
e) Если установлен бит TXIE в ‘1’, микроконтроллер выходит из режима SLEEP, происходит переход по вектору прерывания, если прерывания разрешены

Рекомендованная последовательность действий для передачи данных в синхронном ведомом режиме:

1. Выбрать синхронный ведомый режим установкой битов SYNC, SPEN в ‘1’ и сбросом CSRC в ‘0’
2. Сбросить биты SREN и CREN в ‘0’
3. Если необходимо, разрешить прерывания установкой бита TXIE в ‘1’
4. Если передача 9-разрядная, установить бит TX9 в ‘1’
5. Разрешить передачу установкой бита TXEN в ‘1’
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D
7. Для начала передачи записать данные в регистр TXREG
8. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в '1'

Таблица 16-9. Регистры и биты, связанные с работой передатчика USART в синхронном ведомом режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FADh	TXREG	Регистр передатчика USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 −010

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

16.4.2 Прием синхронного ведомого

Работа приемника в обоих синхронных режимах одинакова, кроме работы в режиме SLEEP. В синхронном ведомом режиме не учитывается состояние бита SREN.

Если перед выполнением команды SLEEP был разрешен прием (бит CREN = 1), то модуль USART может принять слово в SLEEP режиме микроконтроллера. По окончанию приема данные передаются из регистра RSR в RCREG. Если бит RCIE = 1, микроконтроллер выйдет из режима SLEEP. Если прерывания разрешены, произойдет переход по адресу вектора прерываний.

Рекомендованные действия при приеме данных в синхронном ведомом режиме:

1. Выбрать синхронный ведомый режим установкой битов SYNC, SPEN в ‘1’ и сбросом CSRC в ‘0’
2. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’
3. Если прием 9-разрядный, установить бит RX9 в ‘1’
4. Установите бит CREN в ‘1’ для разрешения приема
5. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE
6. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки
7. Считать 8 бит данных из регистра RCREG
8. При возникновении ошибки переполнения сбросить бит CREN в ‘0’
9. Если используются прерывания, то биты GIE и PEIE в регистре INTCON<7:6> должны быть установлены в '1'

Таблица 16-10. Регистры и биты, связанные с работой приемника USART в синхронном ведомом режиме

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FABh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FAEh	RCREG	Регистр приемника USART								0000 0000
FACh	TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.

Затененные ячейки на работу не влияют.

Примечание 1. Эти биты в микроконтроллерах PIC18F2X2 не реализованы, они должны поддерживаться сброшенными в ‘0’.

17. Модуль АЦП

Модуль аналого-цифрового преобразования (АЦП) имеет пять каналов у PIC18F2x2 и восемь каналов у PIC18F4x2. Модуль АЦП имеет управляющие регистры ADCON0 и ADCON1, совместимые с модулями АЦП микроконтроллеров среднего семейства.

Входной аналоговый сигнал через коммутатор каналов заряжает внутренний конденсатор АЦП C_{HOLD}. Модуль АЦП преобразует напряжение, удерживаемое на конденсаторе C_{HOLD} в соответствующий 10-разрядный цифровой код методом последовательного приближения.

Для управления АЦП в микроконтроллере используется 4 регистра:

- Регистр результата ADRESH (старший байт)
- Регистр результата ADRESL (младший байт)
- Регистр управления ADCON0
- Регистр управления ADCON1

Регистр ADCON0 используется для настройки работы модуля АЦП, а с помощью регистра ADCON1 устанавливается, какие входы микроконтроллера будут использоваться модулем АЦП и в каком режиме (аналоговый вход или цифровой порт ввода/вывода).

Регистр 17-1. ADCON0: Управляющий регистр модуля АЦП

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/-DONE	-	ADON
Бит 7							Бит 0

бит 7-6 **ADCS1:ADCS0:** Выбор источника тактового сигнала (биты ADCON0 выделены **полужирным** шрифтом)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Тактовый сигнал АЦП
0	**00**	$F_{OSC}/2$
0	**01**	$F_{OSC}/8$
0	**10**	$F_{OSC}/32$
0	**11**	F_{RC} (внутренний RC генератор модуля АЦП)
1	**00**	$F_{OSC}/4$
1	**01**	$F_{OSC}/16$
1	**10**	$F_{OSC}/64$
1	**11**	F_{RC} (внутренний RC генератор модуля АЦП)

бит 5-3 **CHS2:CHS0:** Выбор 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)

Примечание. В микроконтроллерах PIC18F2x2 все 8 каналов АЦП не реализованы. Номера не реализованных каналов АЦП зарезервированы. Не рекомендуется выбирать номер не реализованного канала АЦП.

бит 2 **GO/-DONE:** Бит статуса модуля АЦП
Если ADON=1
1 = модуль АЦП выполняет преобразование (установка бита вызывает начало преобразования)
0 = состояние ожидания (аппаратно сбрасывается по завершению преобразования)

бит 1 **Не используется:** читается как ‘0’

бит 0 **ADON:** Бит включения модуля АЦП
1 = модуль АЦП включен
0 = модуль АЦП выключен и не потребляет тока

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Регистр 17-2. ADCON1: Управляющий регистр модуля АЦП

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0
Бит 7							Бит 0

бит 7 **ADFM:** Формат сохранения 10-разрядного результата
1 = правое выравнивание, 6 старших бит ADRESH читаются как ‘0’
0 = левое выравнивание, 6 младших бит ADRESL читаются как ‘0’

бит 6 **ADCS2:** Выбор источника тактового сигнала (биты ADCON1 выделены **полужирным** шрифтом)

ADCON1 <ADCS2>	**ADCON0 <ADCS1:ADCS0>**	**Тактовый сигнал АЦП**
0	00	$F_{OSC}/2$
0	01	$F_{OSC}/8$
0	10	$F_{OSC}/32$
0	11	F_{RC} (внутренний RC генератор модуля АЦП)
1	00	$F_{OSC}/4$
1	01	$F_{OSC}/16$
1	10	$F_{OSC}/64$
1	11	F_{RC} (внутренний RC генератор модуля АЦП)

бит 5-4 **Не используются:** читаются как ‘0’

бит 3-0 **PCFG3:PCFG0:** Управляющие биты настройки каналов АЦП

PCFG <3:0>	**AN7**	**AN6**	**AN5**	**AN4**	**AN3**	**AN2**	**AN1**	**AN0**	**V_{REF}+**	**V_{REF}-**	**Кан./ V_{REF}(2)**
0000	A	A	A	A	A	A	A	A	V_{DD}	V_{SS}	8/0
0001	A	A	A	A	V_{REF}+	A	A	A	AN3	V_{SS}	7/1
0010	D	D	D	A	A	A	A	A	V_{DD}	V_{SS}	5/0
0011	D	D	D	A	V_{REF}+	A	A	A	AN3	V_{SS}	4/1
0100	D	D	D	D	A	D	A	A	V_{DD}	V_{SS}	3/0
0101	D	D	D	D	V_{REF}+	D	A	A	AN3	V_{SS}	2/1
011x	D	D	D	D	D	D	D	D	-	-	0/0
1000	A	A	A	A	V_{REF}+	V_{REF}-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	V_{DD}	V_{SS}	6/0
1010	D	D	A	A	V_{REF}+	A	A	A	AN3	V_{SS}	5/1
1011	D	D	A	A	V_{REF}+	V_{REF}-	A	A	AN3	AN2	4/2
1100	D	D	D	A	V_{REF}+	V_{REF}-	A	A	AN3	AN2	3/2
1101	D	D	D	D	V_{REF}+	V_{REF}-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	V_{DD}	V_{SS}	1/0
1111	D	D	D	D	V_{REF}+	V_{REF}-	D	A	AN3	AN2	1/2

A = аналоговый вход D = цифровой канал ввода/вывода

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

Источник верхнего и нижнего опорного напряжения может быть программно выбран с выводов V_{DD}, V_{SS}, RA3/AN3/V_{REF+} и RA2/AN2/V_{REF-}.

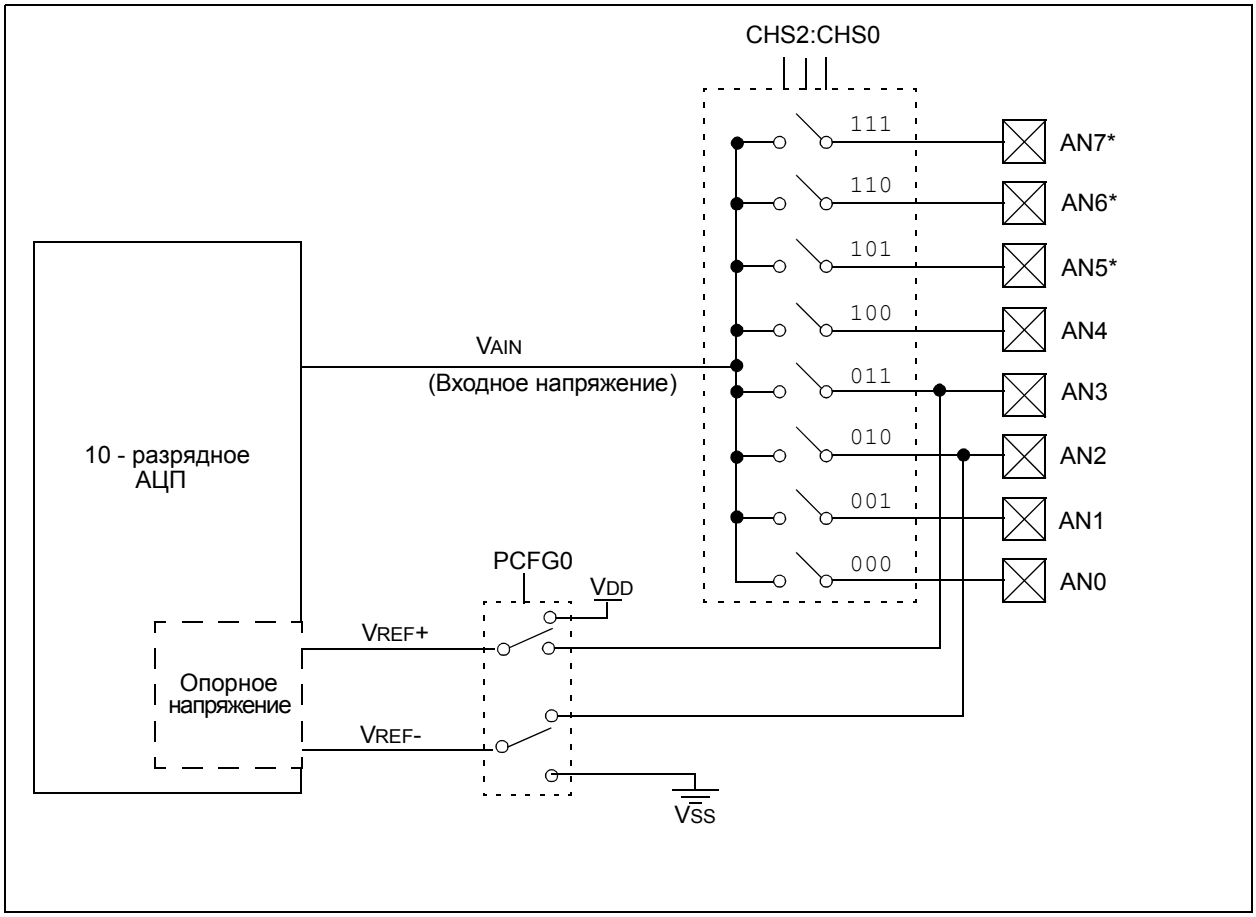
Допускается работа модуля АЦП в SLEEP режиме микроконтроллера, при этом в качестве источника тактовых импульсов для АЦП должен быть выбран RC генератор АЦП.

При сбросе микроконтроллера значения всех его регистров устанавливаются по умолчанию. Сброс выключает модуль АЦП, а также останавливает процесс преобразования, если он был начат.

Каждый канал порта, связанный с модулем АЦП, может быть настроен как аналоговый вход (RA3 и RA2 как входы опорного напряжения) или цифрового входа/выхода.

В регистре ADRESH:ADRESL сохраняется 10-разрядный результат аналого-цифрового преобразования. Когда преобразование завершено, результат преобразования записывается в регистр ADRESH:ADRESL, после чего сбрасывается флаг GO/-DONE (ADCON0<2>) и устанавливается флаг прерывания ADIF. Структурная схема модуля АЦП показана на рисунке 17-1.

Рисунок 17-1. Структурная схема модуля АЦП



* Эти каналы не реализованы в микроконтроллерах PIC18F2x2.

Регистры ADRESH, ADRESL после сброса POR будут содержать неизвестное значение, а после остальных видов сброса не изменят своего значения.

После включения и конфигурации АЦП выбирается рабочий аналоговый канал. Соответствующие биты TRIS аналоговых каналов должны настраивать порт ввода/вывода на вход. Перед началом преобразования необходимо выдержать временную паузу, расчет которой приведен в разделе 17.1.

Рекомендованная последовательность действий для работы с АЦП:

1. Настроить модуль АЦП:
   - Настроить выводы как аналоговые входы, входы V_{REF} или цифровые каналы ввода/вывода (ADCON1)
   - Выбрать входной канал АЦП (ADCON0)
   - Выбрать источник тактовых импульсов для АЦП (ADCON0)
   - Включить модуль АЦП (ADCON0)
2. Настроить прерывание от модуля АЦП (если необходимо):
   - Сбросить бит ADIF в '0'
   - Установить бит ADIE в '1'
   - Установить бит PEIE в '1;
   - Установить бит GIE в '1'
3. Выдержать паузу, необходимую для зарядки конденсатора C_{HOLD}
4. Начать аналого-цифровое преобразование:
   - Установить бит GO/-DONE в '1' (ADCON0)
5. Ожидать окончания преобразования:
   - Ожидать пока бит GO/-DONE не будет сброшен в '0' ИЛИ
   - Ожидать прерывание по окончанию преобразования
6. Считать результат преобразования из регистров ADRESH:ADRESL, сбросить бит ADIF в '0', если это необходимо.
7. Для следующего преобразования необходимо выполнить шаги начиная с пункта 1 или 2. Время преобразования одного бита определяется как время T_{AD}. Минимальное время ожидания перед следующим преобразованием должно составлять не менее $2T_{AD}$.

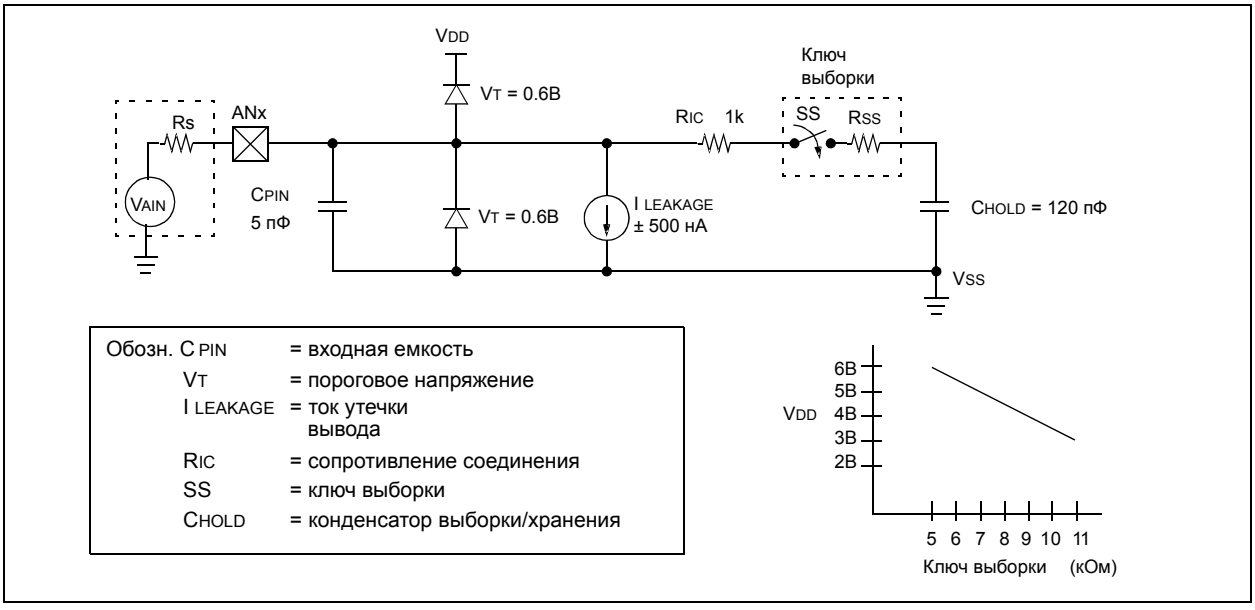
17.1 Временные требования к подключению канала АЦП

Для обеспечения необходимой точности преобразования, конденсатор C_{HOLD} должен успевать полностью заряжаться до уровня входного напряжения. Схема аналогового входа АЦП показана на рисунке 17-2. Сопротивления R_S и R_{SS} непосредственно влияют на время зарядки конденсатора C_{HOLD}. Величина сопротивления ключа выборки (R_{SS}) зависит от напряжения питания V_{DD}. **Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 2.5кОм.** При меньших значениях сопротивления источника сигнала - меньше суммарное время преобразования.

Примечание. Когда инициализировано преобразование АЦП, конденсатор C_{HOLD} отключен от аналогового входа.

После того, как будет выбран один из нескольких аналоговых входных каналов, но прежде, чем будет производиться преобразование, должно пройти определенное время. Для нахождения данного времени воспользуетесь уравнением 17-1. Это уравнение дает результат с ошибкой в ½ LSb (1024 шагов АЦП). Ошибка в ½ LSb, это максимальная погрешность, позволяющая функционировать модулю АЦП с необходимой точностью.

Рисунок 17-2. Схема аналогового входа АЦП



Уравнение 17-1. Вычисление временной задержки

T_{ACQ} = Время задержки усилителя + Время заряда конденсатора C_{HOLD} + Температурный коэффициент
= $T_{AMP} + T_C + T_{COFF}$

Уравнение 17-2 Минимальное время заряда конденсатора C_{HOLD}

$$V_{HOLD} = (V_{REF} - (V_{REF}/512))\bullet(1 - e^{(-Tc/Chold(Ric + Rss + Rs))})$$

T_C = - 120пФ (1кОм + R_{SS} + R_S) Ln(1/2047)

В примере 17-1 показано вычисление минимального значения времени T_{ACQ}. Вычисления основываются на следующих исходных данных:

C_{HOLD}	= 120пФ
Rs	= 2.5кОм
Ошибка преобразования	≤ 1/2 Lsb
V_{DD}	= 5В → Rss = 7 кОм
Температура	= 50°C (максимально возможная)
V_{HOLD}	= 0В @ t = 0

Пример 17-1 Вычисление минимального значения времени T_{ACQ}

T_{ACQ} = $T_{AMP} + T_C + T_{COFF}$
Температурный коэффициент необходимо использовать только при рабочей температуре более 25°C.

T_{ACQ} = 2мкс + T_C + [(Температура - 25°C)(0.05мкс/°C)]

T_C = - C_{HOLD} ($R_{IC} + R_{SS} + R_S$) Ln(1/2047)
= - 120пФ (1кОм + 7кОм + 2.5кОм) Ln(0.0004885)
= - 120пФ (10.5кОм) Ln(0. 0004885)
= -1.26мкс (-7.6241)
= 9.61мкс

T_{ACQ} = 2мкс + 9.61мкс + [(50°C - 25°C)(0.05мкс/°C)]
= 11.61мкс + 1.25мкс
= 12.86мкс

17.2 Выбор источника тактовых импульсов для АЦП

Время получения одного бита результата определяется параметром T_{AD}. Для 10-разрядного результата требуется как минимум $12T_{AD}$. Параметры тактового сигнала для АЦП определяются программно, T_{AD} может принимать следующие значения:

- $2T_{OSC}$
- $4T_{OSC}$
- $8T_{OSC}$
- $16T_{OSC}$
- $32T_{OSC}$
- $64T_{OSC}$
- Внутренний RC генератор модуля АЦП (2-6мкс).

Для получения корректного результата преобразования необходимо выбрать источник тактового сигнала АЦП, обеспечивающий время T_{AD} не менее 1.6 мкс.

В таблице 17-1 указано максимальное значение тактовой частоты микроконтроллера для каждого режима синхронизирующего сигнала АЦП.

Таблица 17-1 Максимальное значение F_{OSC}, удовлетворяющее требованию к T_{AD}

Выбор T_{AD}		**Максимальная F_{OSC}**	
Режим	**ADCS2:ADCS0**	**PIC18Fxx2**	**PIC18LFxx2**
$2T_{OSC}$	000	1.25 МГц	666 кГц
$4T_{OSC}$	100	2.50 МГц	1.33 МГц
$8T_{OSC}$	001	5.00 МГц	2.67 МГц
$16T_{OSC}$	101	10.00 МГц	5.33 МГц
$32T_{OSC}$	010	20.00 МГц	10.67 МГц
$64T_{OSC}$	110	40.00 МГц	21.33 МГц
RC	011	-	-

Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс для PIC18Fxx2 и 6мкс для PIC18LFxx2.
2. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

17.3 Настройка аналоговых входов

Регистры ADCON1, TRISA и TRISE отвечают за настройку выводов АЦП. Если выводы микросхемы настраиваются как аналоговые входы, то при этом должны быть установлены соответствующие биты в регистре TRIS. Если соответствующий бит сброшен в '0', вывод микросхемы настроен как цифровой выход со значениями выходных напряжений V_{OH} или V_{OL}.

Модуль АЦП работает независимо от состояния битов CHS2:CHS0 и битов TRIS.

Примечания:

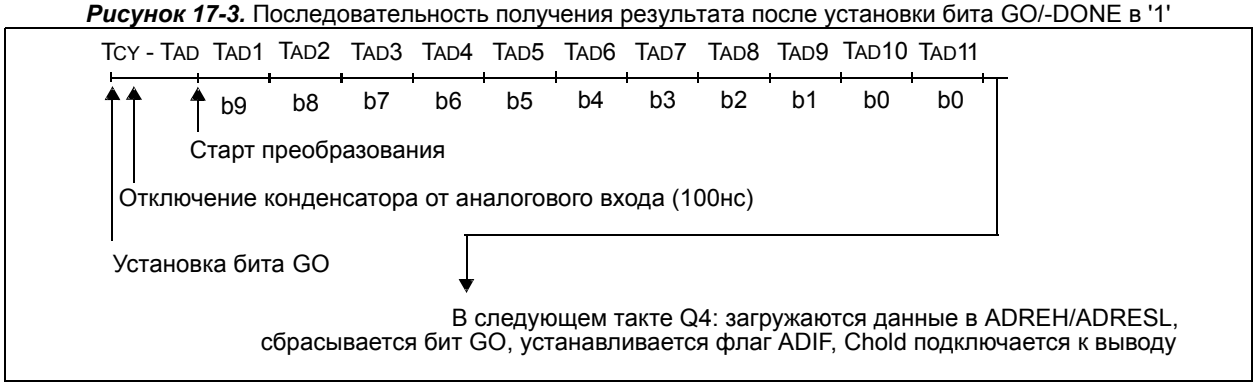
1. При чтении содержимого регистра порта нули будут установлены в тех разрядах, которые были настроены как аналоговые входы. Настроенные на цифровой вход каналы будут преобразовывать входные аналоговые уровни в цифровые, что не окажет влияния на точность преобразования.
2. Значения напряжений, подаваемых на выводы, настроены как аналоговые входы, включая выводы (AN7:AN0), могут влиять на ток потребления входного буфера, который может выйти за пределы значений, оговоренных в технической спецификации.

17.4 Аналого-цифровое преобразование

На рисунке 17-3 показана последовательность получения результата после установки бита GO/-DONE в '1'. Сброс бита GO/-DONE в '0' во время преобразования приведет к его прекращению. При этом регистры результата (ADRESH:ADRESL) не изменят своего содержимого. После досрочного завершения преобразования необходимо обеспечить временную задержку $2T_{AD}$. Выдержав требуемую паузу, можно начать новое преобразования установкой бита GO/-DONE в '1'.

Примечание. Бит GO/-DONE и бит включения АЦП должны устанавливаться разными командами.

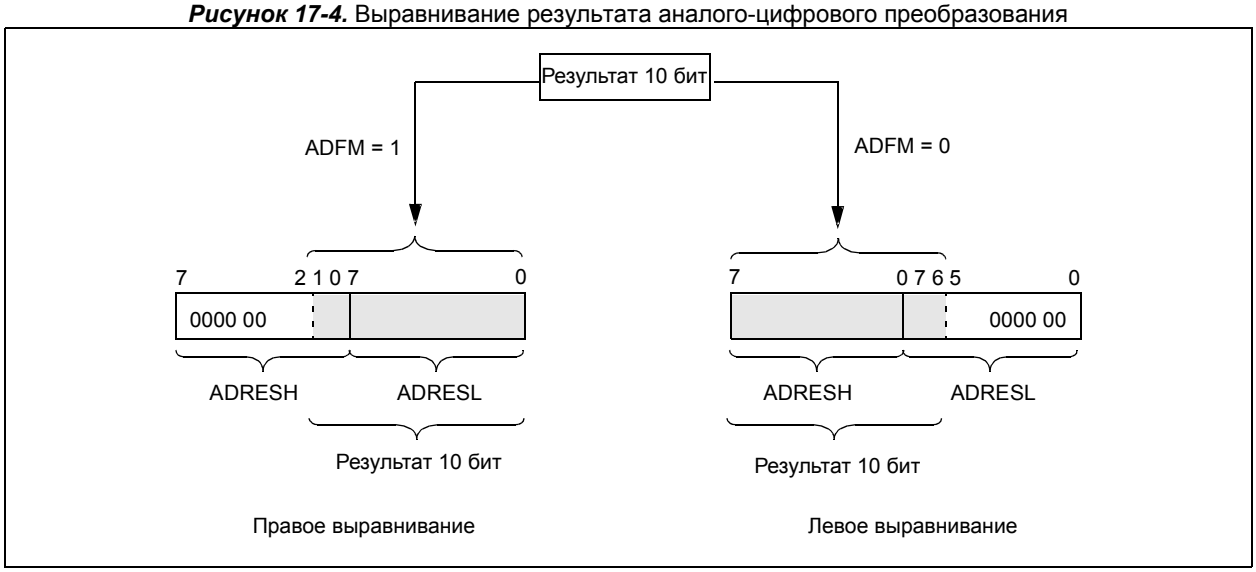
Рисунок 17-3. Последовательность получения результата после установки бита GO/-DONE в '1'



17.5 Выравнивание результата преобразования

10-разрядный результат преобразования сохраняется в спаренном 16-разрядном регистре ADRESH:ADRESL. Запись результата преобразования может выполняться с правым или левым выравниванием, в зависимости от значения бита ADFM (см. рисунок 17-4). Не задействованные биты регистра ADRESH:ADRESL читаются как ‘0’. Если модуль АЦП выключен, то 8-разрядные регистры ADRESH и ADREL могут использоваться как регистры общего назначения.

Рисунок 17-4. Выравнивание результата аналого-цифрового преобразования



17.6 Использование триггера CCP2

Аналого-цифровое преобразование может быть запущено при помощи “триггера специального события” модуля CCP2. Для этого необходимо, чтобы биты CCP2M3:CCP2M0 (CCP2CON<3:0>) были запрограммированы как 1011 и был включен модуль АЦП (бит ADON должен быть установлен в '1'). При срабатывании триггера бит GO/-DONE будет установлен в '1', тем самым, запуская преобразование, а содержимое таймера TMR1 (или TMR3) будет обнулено. Таймер сбрасывается и автоматически повторяет запуск преобразования через определенные промежутки времени. Пользователю необходимо будет только вовремя считывать содержимое регистров ADRESH:ADRESL. До начала преобразования необходимо выбрать соответствующий аналоговый канал, прежде чем “триггер специального события“ вызовет установку бита GO/-DONE.

При выключенном модуле АЦП (бит ADON сброшен в '0') сигнал “триггера специального события” игнорируется, но таймер TMR1 (или TMR3) продолжает работать и сбрасываться.

Таблица 17-2. Регистры и биты, связанные с работой модуля АЦП

Адрес	Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Значение после POR, BOR
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
F9Fh	IRP1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000
F9Eh	PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000
F98h	PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000
FA2h	IRP2	-	-	-	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111
FA1h	PIR2	-	-	-	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000
FA0h	PIE2	-	-	-	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000
FC4h	ADRESH	Старший байт результата преобразования АЦП								xxxx xxxx
FC3h	ADRESL	Младший байт результата преобразования АЦП								xxxx xxxx
FC2h	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/-DONE	-	ADON	0000 00-0
FC1h	ADCON1	ADFM	ADCS2	-	-	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000
F80h	PORTA	-	RA6	RA5	RA4	RA3	RA2	RA1	RA0	-x0x 0000
F89h	LATA	-	Регистр выходных данных							-xxx xxxx
F92h	TRISA	-	Регистр направления данных							-111 1111
F84h	PORE	-	-	-	-	-	RE2	RE1	RE0	---- -000
F8Dh	LATE	-	-	-	-	-	Регистр выходных данных			---- -xxx
F96h	TRISE	IBF	OBF	³SPMODE	-	-	Регистр направления данных			0000 -111

Обозначения: x = неизвестно; u = не изменяется; r = резерв; - = не реализован, читается как ‘0’.
Затененные ячейки на работу не влияют.

18. Детектор пониженного напряжения LVD

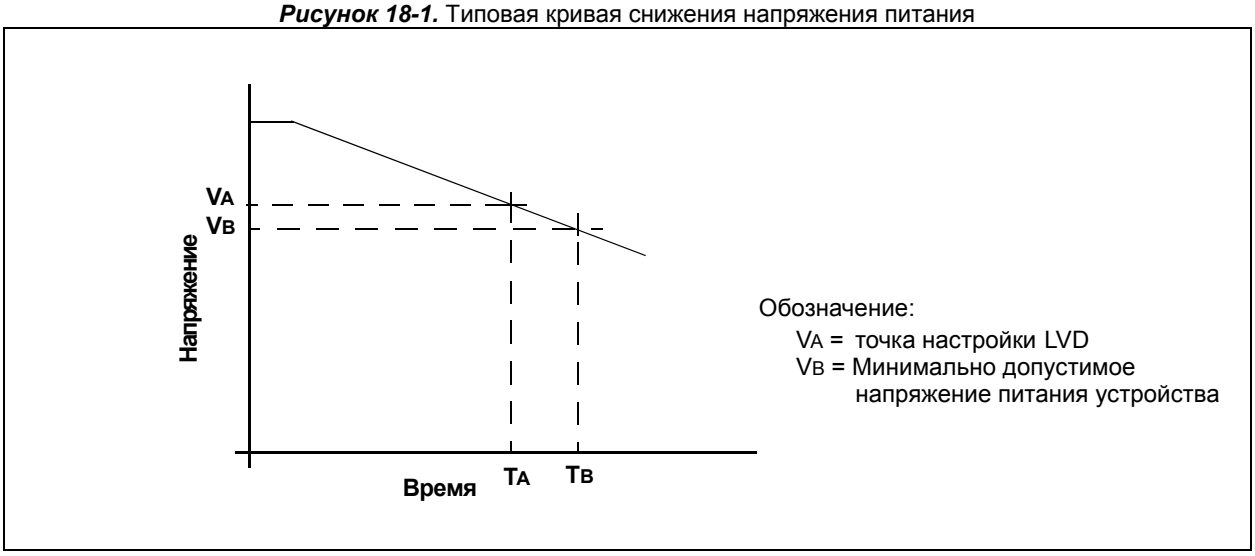
Во многих приложениях желательной функцией является возможность определения снижения напряжения питания V_{DD} ниже установленного уровня. Эта функция может быть полезна, когда необходимо выполнить определенные программные процедуры прежде, чем напряжения питания устройства станет ниже рабочего диапазона. Это можно сделать применяя модуль LVD – детектор пониженного напряжения.

В модуле LVD предусмотрена программируемая схема выбора контрольного уровня напряжения. Когда напряжение питания устройства становится ниже контрольного, устанавливается флаг прерывания. Если прерывания от модуля LVD разрешены, то произойдет переход по вектору прерывания и программное обеспечение может обработать событие снижения напряжения.

Детектор пониженного напряжения имеет программное управление. Это позволяет программе пользователя выключить модуль LVD для снижения потребляемого тока.

На рисунке 18-1 представлена типовая кривая снижения напряжения питания для устройств с батарейным питанием. Когда напряжение питание равно V_A, модуль LVD генерирует прерывание в момент T_A. Программа пользователя имеет некоторое время для завершения работы. Уровень напряжения V_B – минимальное напряжение питания устройства, определенное спецификацией. Разница T_B – T_A – интервал времени для завершения работы устройства.

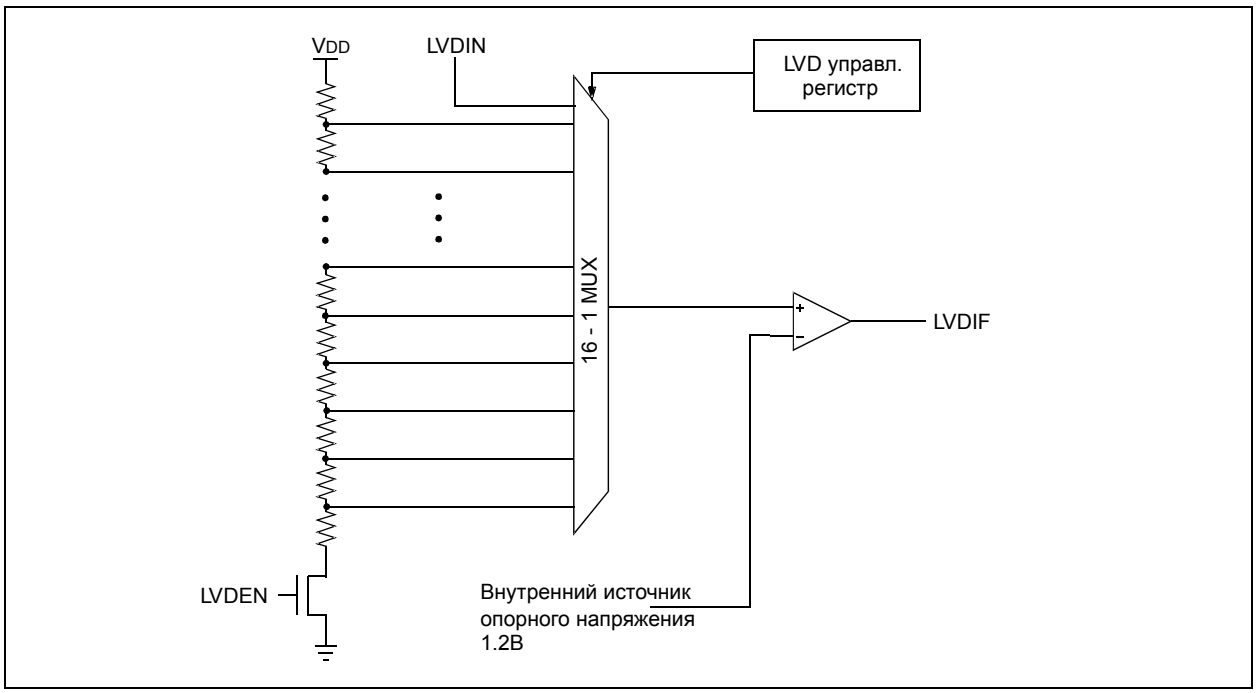
Рисунок 18-1. Типовая кривая снижения напряжения питания



Блок схема модуля LVD представлена на рисунке 18-2. Компаратор использует внутренний источник опорного напряжения как отправную точку. Когда напряжение питания становится ниже контрольной точки, устанавливается в ‘1’ флаг LVDIF.

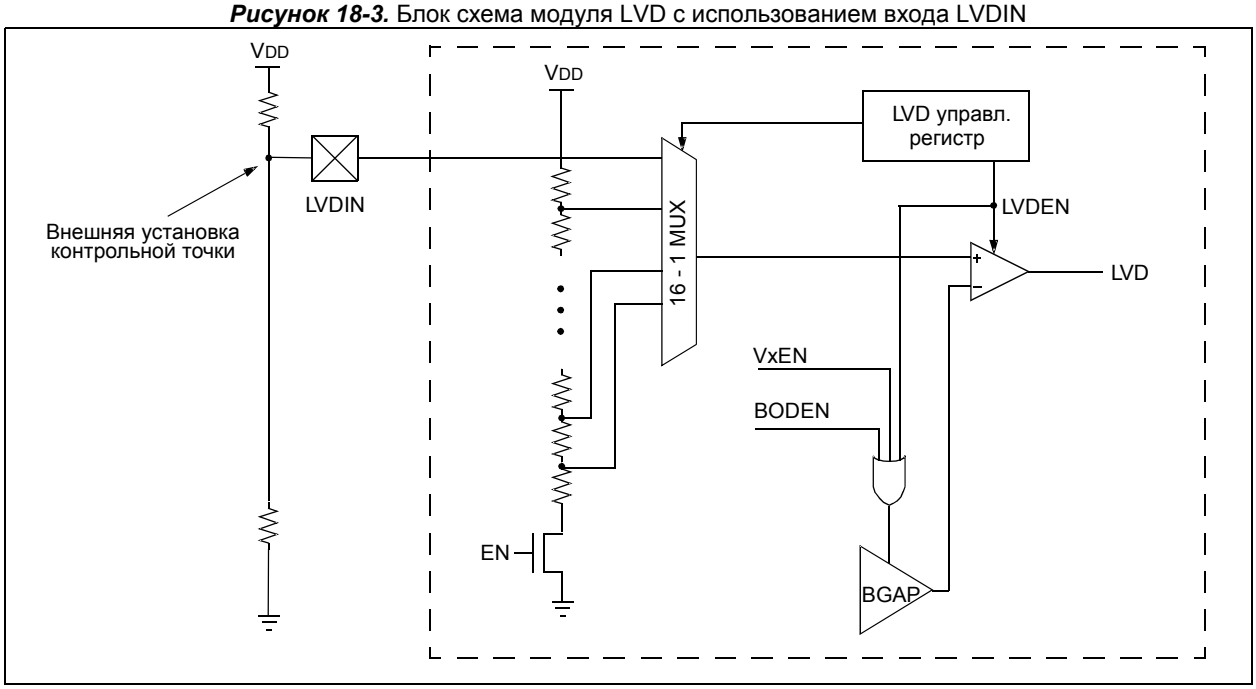
Каждый узел в последовательно включенных резисторах представляет собой конкретный уровень напряжения контрольной точки. Напряжение контрольной точки – минимальное напряжение питания, при котором может работать устройство прежде, чем будет сформировано прерывание от модуля LVD. Когда напряжение питания равно контрольному уровню, напряжение на выбранной точке резистивного делителя равно напряжению внутреннего источника опорного напряжения 1.2В модуля LVD. Компаратор переключается, устанавливая флаг прерывания LVDIF. Один из 16-ти уровней контрольного напряжения может быть выбран программно битами LVDL3:LVDL0 (LVDCON<3:0>) (смотрите рисунок 18-2).

Рисунок 18-2. Блок схема модуля LVD



LVD модуль имеет дополнительную особенность, которая позволяет пользователю устанавливать напряжение контрольной точки внешней схемой (LVDL3:LVDL0 = 1111). В этом режиме вход компаратора подключен к выводу LVDIN (смотрите рисунок 18-3). Установка напряжение контрольной точки внешней схемой предоставляет разработчику дополнительную гибкость в обнаружении снижения напряжения питания в рабочем диапазоне напряжений питания.

Рисунок 18-3. Блок схема модуля LVD с использованием входа LVDIN



18.1 Регистр управления

Биты управления модулем LVD расположены в регистре LVDCON.

Регистр 18-1. LVDCON: Регистр управления модуля LVD

U-0	U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0
Бит 7							Бит 0

бит 7-6 **Не используется:** читается как ‘0’

бит 5 **IRVST:** Флаг стабилизации источника опорного напряжения модуля LVD
1 = источник опорного напряжения стабилизировался
0 = источник опорного напряжения стабилизировался или модуль LVD выключен

бит 4 **LVDEN:** Включение модуля LVD
1 = модуль LVD включен
0 = модуль LVD выключен

бит 3-0 **LVDL3:LVDL0:** Выбор напряжения контрольной точки модуля LVD
1111 = внешний аналоговый сигнал с вывода LVDIN
1110 = 4.5В – 4.77В
1101 = 4.2В – 4.45В
1100 = 4.0В – 4.24В
1011 = 3.8В – 4.03В
1010 = 3.6В – 3.82В
1001 = 3.5В – 3.71В
1000 = 3.3В – 3.50В
0111 = 3.0В – 3.18В
0110 = 2.8В – 2.97В
0101 = 2.7В – 2.86В
0100 = 2.5В – 2.65В
0011 = 2.4В – 2.54В
0010 = 2.2В – 2.33В
0001 = 2.0В – 2.12В
0000 = резерв

Примечание. Режим работы, установленный битами LVDL3:LVDL0 ниже рабочего диапазона напряжения питания, не тестировался.

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

18.2 Работа модуля LVD

В зависимости от источника напряжения питания и тока потребления устройства, напряжение питания обычно уменьшается относительно медленно. Это означает, что модель LVD может постоянно не работать. Чтобы уменьшить ток потребления устройством схему LVD модуля можно кратковременно включать для проверки напряжения питания. После выполнения проверки модуль LVD может быть выключен.

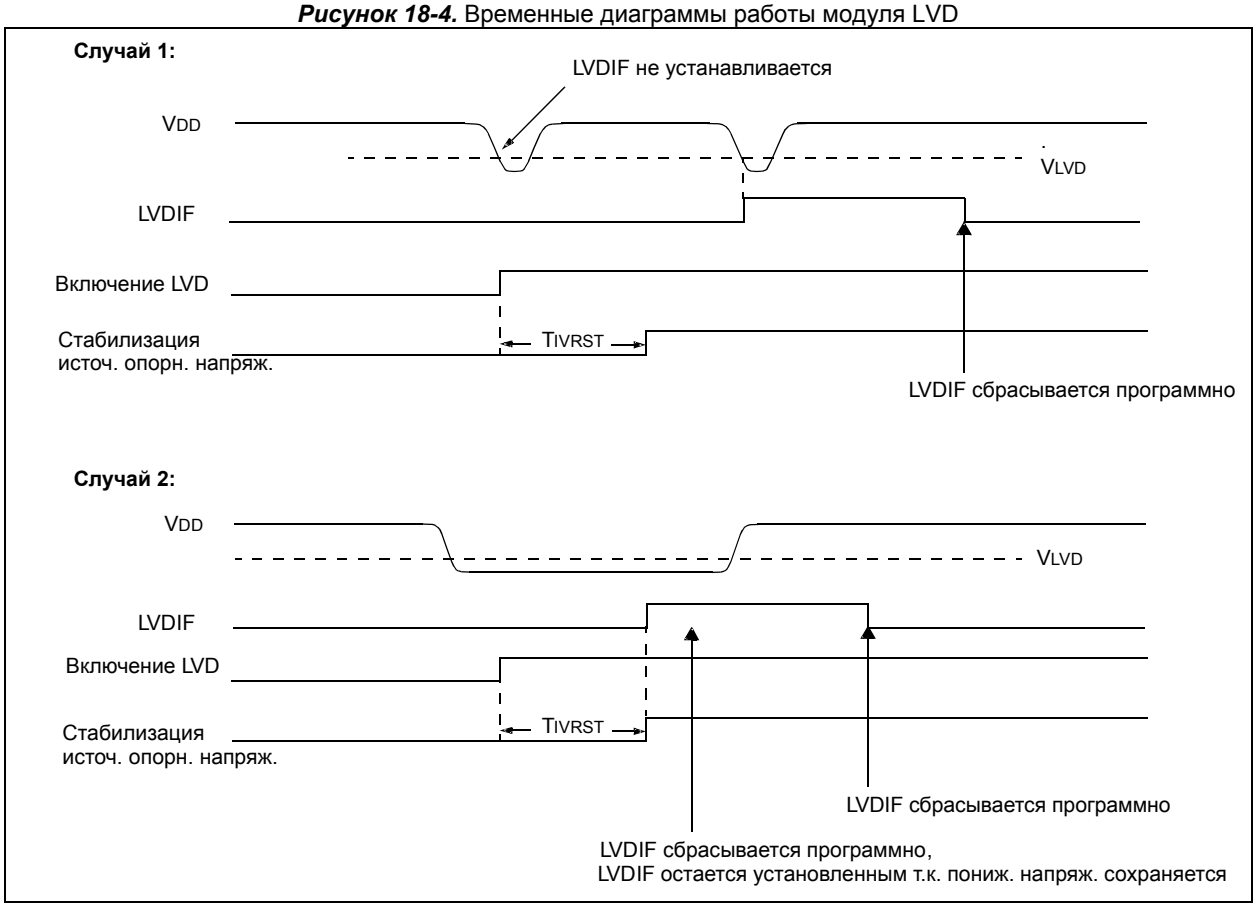
После каждого включения модуля LVD требуется некоторое время для стабилизации работы его схемы. Как только работа схемы стабилизировалась, флаги состояния модуля LVD могут быть сброшены и выполнена проверка напряжения питания.

Рекомендуется следующая последовательность действий для настройки модуля LVD:

1. Установить значение битов LVDL3:LVDL0 для выбора напряжения контрольной точки
2. Гарантировать, что прерывания от модуля LVD запрещены (бит LVDIE=0 или GIE=0)
3. Включить модуль LVD (LVDEN=1 LVDCON<4>)
4. Ожидать стабилизацию схемы модуля LVD (бит IRVST =1)
5. Сбросить флаг LVDIF, который мог установиться в ‘1’ пока стабилизировалась работы схемы модуля LVD
6. Разрешить прерывания от модуля LVD (LVDIE=1, GIE=1)

На рисунке 18-4 представленные типовые временные диаграммы работы модуля LVD.

Рисунок 18-4. Временные диаграммы работы модуля LVD



18.2.1 Внутренний источник опорного напряжения

Внутренний источник опорного напряжения LVD модуля может использоваться другой внутренней схемой микроконтроллера (программируемый сброс по снижению напряжения питания BOR). Если эти схемы выключены (для снижения энергопотребления), то источник опорного напряжения требует некоторого времени для стабилизации работы прежде, чем будет надежно выполняться обнаружение снижения напряжения питания. Время стабилизации не зависит от тактового сигнала микроконтроллера (смотрите параметр №36 в электрических спецификациях). Прерывания от модуля LVD не должны разрешаться, пока не стабилизируется источник опорного напряжения модуля LVD (смотрите временные диаграммы на рисунке 18-4).

18.2.2 Ток потребления

Когда модуль LVD включен, компаратор и резистивная цепочка потребляют статический ток. Полный ток потребления, когда модуль LVD включен, указан в электрических спецификациях (параметр *D022B).

18.3 Работа модуля LVD в SLEEP режиме

Если модуль LVD включен, то он продолжает работать в SLEEP режиме микроконтроллера. Если напряжение питания пересекает контрольную точку, то будет установлен флаг LVDIF и микроконтроллер выйдет из режима SLEEP. Выполнение программы перейдет по вектору прерывания, если глобально разрешены прерывания.

18.4 Эффект сброса

При сбросе микроконтроллера регистр LVDCON инициализируется в первоначальное состояние. Это означает, что после сброса микроконтроллера модуль LVD выключен.

19. Особенности микроконтроллеров PIC18FXX2

PIC18FXX2 имеют много усовершенствований повышающие надежность системы, снижающие стоимость устройства и число внешних компонентов, а также предусмотрены режимы энергосбережения и возможность защиты кода программы.

Основные достоинства:

- Выбор тактового генератора
- Сброс
  - сброс по включению питания (POR)
  - таймер включения питания (PWRT)
  - таймер запуска генератора (OSC)
  - сброс по снижению напряжения питания (BOR)
- Прерывания
- Сторожевой таймер (WDT)
- Режим энергосбережения (SLEEP)
- Защита кода программы
- Область памяти для идентификатора
- Внутрисхемное программирование по последовательному порту (ICSP)

В микроконтроллеры PIC18FXX2 встроен сторожевой таймер WDT, который может быть выключен в битах конфигурации микроконтроллера или программно. Для повышения надежности сторожевой таймер WDT имеет собственный RC генератор. Дополнительных два таймера выполняют задержку старта работы микроконтроллера. Первый, таймер запуска генератора (OST), удерживает микроконтроллер в состоянии сброса, пока не стабилизируется частота тактового генератора. Второй, таймер включения питания (PWRT), инициализируется после включения питания и удерживает микроконтроллер в состоянии сброса, пока не стабилизируется напряжение питания. В большинстве приложений эти функции микроконтроллера позволяют исключить внешние схемы сброса.

Режим SLEEP предназначен для обеспечения сверхнизкого энергопотребления. Микроконтроллер может выйти из режима SLEEP по сигналу внешнего сброса, по переполнению сторожевого таймера или при возникновении прерываний. Выбор режима работы тактового генератора позволяет использовать микроконтроллеры в различных приложениях. Режим тактового генератора RC позволяет уменьшить стоимость устройства, а режим LP снизить энергопотребление. Биты конфигурации микроконтроллера используются для указания режима его работы.

19.1 Биты конфигурации

Биты конфигурации могут быть запрограммированы (читаются как ‘0’) или оставлены не запрограммированными (читаются как ‘1’) для указания режима работы микроконтроллера. Биты конфигурации расположены в памяти программ начиная с адреса 300000h.

Заметьте, что адрес 300000h расположен за пределами пользовательской памяти программ. Фактически, к конфигурационному регистру (область памяти 300000h – 3FFFFFh) можно обратиться только командами таблично чтения/записи.

Программирование битов конфигурации выполняется аналогично программированию Flash памяти программ. По установке бита WR в регистре EECON1 инициализируется запись в регистр конфигурации (длительность записи управляется аппаратно). В нормальном режиме в указатель TBLPTR загружен адрес регистра конфигурации, по команде TBLWT регистрируются данные для регистра конфигурации. Установкой бита WR инициализируется длинная запись в регистр конфигурации. Запись в регистры конфигурации выполняется побайтно. Чтобы записать или стереть ячейку конфигурации нужно записать по команде TBLWT ‘0’ или ‘1’.

Таблица 19-1. Биты конфигурации и идентификации микроконтроллера

Имя		Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Не запрог. значение
300001h	CONFIG1H	-	-	-OSCEN	-	-	FOSC2	FOSC1	FOSC0	--1- -111
300002h	CONFIG2L	-	-	-	-	BORV1	BORV0	BODEN	-PWRTEN	---- 1111
300003h	CONFIG2H	-	-	-	-	WDTPS2	WDTPS1	WDTPS0	WDTEN	---- 1111
300005h	CONFIG3H	-	-	-	-	-	-	-	CCP2MX	---- ---1
300006h	CONFIG4L	-DEBUG	-	-	-	-	LVP	-	STVREN	1--- -1-1
300008h	CONFIG5L	-	-	-	-	CP3	CP2	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	-	-	-	-	-	-	11-- ----
30000Ah	CONFIG6L	-	-	-	-	WRT3	WRT2	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	-	-	-	-	-	111- ----
30000Ch	CONFIG7L	-	-	-	-	EBTR3	EBTR2	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	-	EBTRB	-	-	-	-	-	-	-1-- ----
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	**(1)**
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DE6	DEV5	DEV4	DEV3	0000 0100

Затененные ячейки на работу не влияют.

Примечание 1. Значение битов смотрите в описании регистра DEVID1.

Регистр 19-1. Регистр конфигурации GONFIG1H (адрес 300001h)

U-0	U-0	R/P-1	U-0	U-0	R/P-1	R/P-1	R/P-1
-	-	-OSCEN	-	-	FOSC2	FOSC1	FOSC0
Бит 7							Бит 0

бит 7-6 **Не используется:** читается как ‘0’

бит 5 **-OSCEN:** Разрешение переключения источника тактового сигнала
1 = функция переключения источника тактового сигнала заблокирована (только основной генератор)
0 = разрешено переключения источника тактового сигнала

бит 4-3 **Не используется:** читается как ‘0’

бит 2-0 **FOSC2:FOSC0:** Режим тактового генератора
111 = RC генератор, вывод OSC2 работает как RA6
110 = HS генератора с включенной функцией PLL. Тактовая частота = 4 x F_{OSC}
101 = EC режим генератора, вывод OSC2 работает как RA6
100 = EC режим генератора, на выводе OSC2 тактовая сигнал $F_{OSC}/4$
011 = RC режим генератора
010 = HS режим генератора
001 = XT режим генератора
000 = LP режим генератора

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-2. Регистр конфигурации GONFIG2L (адрес 300002h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
-	-	-	-	BORV1	BORV0	BODEN	-PWRTEN
Бит 7							Бит 0

бит 7-4 **Не используется:** читается как ‘0’

бит 3-2 **BORV1:BORV0:** Напряжение сброса BOR
11 = V_{BOR} = 2.0B
10 = V_{BOR} = 2.7B
01 = V_{BOR} = 4.2B
00 = V_{BOR} = 4.5B

бит 1 **BOREN:** Разрешение сброса по снижению напряжение питания
1 = сброс BOR разрешен
0 = сброс BOR запрещен

Примечание. При разрешении сброса BOR автоматически разрешается работа таймера PWRT независимо от состояния бита –PWRTEN. Необходимо гарантировать, что при разрешении сброса BOR включен таймер PWRT.

бит 0 **-PWRTEN:** Разрешение работы таймера включения питания
1 = PWRT включен
0 = PWRT выключен

Примечание. При разрешении сброса BOR автоматически разрешается работа таймера PWRT независимо от состояния бита –PWRTEN. Необходимо гарантировать, что при разрешении сброса BOR включен таймер PWRT.

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-3. Регистр конфигурации GONFIG2H (адрес 300003h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
-	-	-	-	WDTPS2	WDTPS1	WDTPS0	WDTEN
Бит 7							Бит 0

бит 7-4 **Не используется:** читается как ‘0’

бит 3-1 **WDTPS2:WDTPS0:** Коэффициент постделителя WDT
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

бит 0 **WDTEN:** Включение WDT
1 = WDT включен
0 = WDT выключен (управление битом SWDTEN)

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-4. Регистр конфигурации GONFIG3H (адрес 300005h)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/P-1
-	-	-	-	-	-	-	CCP2MX
Бит 7							Бит 0

бит 7-1 **Не используется:** читается как ‘0’

бит 0 **CCP2MX:** Управление мультиплексором CCP2
1 = вход/выход CCP2 подключен к выводу RC1
0 = вход/выход CCP2 подключен к выводу RB3

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-5. Регистр конфигурации GONFIG4L (адрес 300006h)

R/P-1	U-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
-DEBUG	-	-	-	-	LVP	-	STVREN
Бит 7							Бит 0

бит 7 **-DEBUG:** Включение внутрисхемной отладки
1 = внутрисхемная отладка выключена, выводы RB6 и RB7 работают как каналы вводы/вывода
0 = внутрисхемная отладка включена, выводы RB6 и RB7 используются отладчиком

бит 6-3 **Не используется:** читается как ‘0’

бит 2 **LVP:** Разрешения низковольтного программирования
1 = низковольтное программирование разрешено
0 = низковольтное программирование запрещено

бит 1 **Не используется:** читается как ‘0’

бит 0 **STVREN:** Сброс при переполнении/исчерпании стека
1 = при переполнении/исчерпании стека происходит сброс микроконтроллера
0 = при переполнении/исчерпании стека сброс микроконтроллера не выполняется

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-6. Регистр конфигурации GONFIG5L (адрес 300008h)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
-	-	-	-	CP3(1)	CP2(1)	CP1(1)	CP0(1)
Бит 7							Бит 0

бит 7-4 **Не используется:** читается как ‘0’

бит 3 **CP3:** Защита памяти программ(1)
1 = блок 3 (006000 – 007FFFh) защита выключена
0 = блок 3 (006000 – 007FFFh) защита включена

бит 2 **CP2:** Защита памяти программ(1)
1 = блок 2 (004000 – 005FFFh) защита выключена
0 = блок 2 (004000 – 005FFFh) защита включена

бит 1 **CP1:** Защита памяти программ
1 = блок 1 (002000 – 003FFFh) защита выключена
0 = блок 1 (002000 – 003FFFh) защита включена

бит 0 **CP0:** Защита памяти программ
1 = блок 0 (000200 – 001FFFh) защита выключена
0 = блок 0 (000200 – 001FFFh) защита включена

Примечание 1. Не реализованы в PIC18FX42, должны быть оставлены не запрограммированными.

Обозначения		
R = чтение бита	C = очистка бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-7. Регистр конфигурации GONFIG5H (адрес 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	-	-	-	-	-	-
Бит 7							Бит 0

бит 7 **CPD:** Защита EEPROM памяти данных
1 = защита EEPROM памяти данных выключена
0 = защита EEPROM памяти данных включена

бит 6 **CPB:** Защита памяти программ
1 = загрузочный блок (000000 – 0001FFh) защита выключена
0 = загрузочный блок (000000 – 0001FFh) защита включена

бит 5-0 **Не используется:** читается как ‘0’

Обозначения		
R = чтение бита	C = очистка бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-8. Регистр конфигурации GONFIG6L (адрес 30000Ah)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
-	-	-	-	WRT3(1)	WRT2(1)	WRT1(1)	WRT0(1)
Бит 7							Бит 0

бит 7-4 **Не используется:** читается как ‘0’

бит 3 **WRT3:** Защита записи в память программ(1)
1 = блок 3 (006000 – 007FFFh) защита выключена
0 = блок 3 (006000 – 007FFFh) защита включена

бит 2 **WRT2:** Защита записи в память программ (1)
1 = блок 2 (004000 – 005FFFh) защита выключена
0 = блок 2 (004000 – 005FFFh) защита включена

бит 1 **WRT1:** Защита записи в память программ
1 = блок 1 (002000 – 003FFFh) защита выключена
0 = блок 1 (002000 – 003FFFh) защита включена

бит 0 **WRT0:** Защита записи в память программ
1 = блок 0 (000200 – 001FFFh) защита выключена
0 = блок 0 (000200 – 001FFFh) защита включена

Примечание 1. Не реализованы в PIC18FX42, должны быть оставлены не запрограммированными.

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-9. Регистр конфигурации GONFIG6H (адрес 30000Bh)

R/P-1	R/P-1	R/P-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC	-	-	-	-	-
Бит 7							Бит 0

бит 7 **WRTD:** Защита записи в EEPROM память данных
1 = запись в EEPROM память данных разрешена
0 = запись в EEPROM память данных запрещена

бит 6 **WRTB:** Защита записи в память программ
1 = загрузочный блок (000000 – 0001FFh) защита выключена
0 = загрузочный блок (000000 – 0001FFh) защита включена

бит 5 **WRTC:** Защита записи в регистры конфигурации
1 = регистры конфигурации (300000 – 3000FFh) защита выключена
0 = регистры конфигурации (300000 – 3000FFh) защита включена

бит 4-0 **Не используется:** читается как ‘0’

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-10. Регистр конфигурации GONFIG7L (адрес 30000Ch)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
-	-	-	-	EBTR3(1)	EBTR2(1)	EBTR1(1)	EBTR0(1)
Бит 7							Бит 0

бит 7-4 **Не используется:** читается как ‘0’

бит 3 **EBTR3:** Защита памяти программ от табличного чтения(1)
1 = блок 3 (006000 – 007FFFh) защита выключена
0 = блок 3 (006000 – 007FFFh) защита включена

бит 2 **EBTR2:** Защита памяти программ от табличного чтения (1)
1 = блок 2 (004000 – 005FFFh) защита выключена
0 = блок 2 (004000 – 005FFFh) защита включена

бит 1 **EBTR1:** Защита памяти программ от табличного чтения
1 = блок 1 (002000 – 003FFFh) защита выключена
0 = блок 1 (002000 – 003FFFh) защита включена

бит 0 **EBTR0:** Защита памяти программ от табличного чтения
1 = блок 0 (000200 – 001FFFh) защита выключена
0 = блок 0 (000200 – 001FFFh) защита включена

Примечание 1. Не реализованы в PIC18FX42, должны быть оставлены не запрограммированными.

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-11. Регистр конфигурации GONFIG7H (адрес 30000Dh)

U-0	R/P-1	U-0	U-0	U-0	U-0	U-0	U-0
-	EBTRB	-	-	-	-	-	-
Бит 7							Бит 0

бит 7 **Не используется:** читается как ‘0’

бит 6 **EBTRB:** Защита памяти программ от табличного чтения
1 = загрузочный блок (000000 – 0001FFh) защита выключена
0 = загрузочный блок (000000 – 0001FFh) защита включена

бит 5-0 **Не используется:** читается как ‘0’

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-12. 1-й ID регистр для микроконтроллеров PIC18FXX2

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
Бит 7							Бит 0

бит 7-5 **DEV2:DEV0:** ID микроконтроллера
000 = PIC18F252
001 = PIC18F452
100 = PIC18F242
001 = PIC18F442

бит 4-0 **REV4:REV0:** ID ревизии кристалла микроконтроллера

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

Регистр 19-13. 2-й ID регистр для микроконтроллеров PIC18FXX2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
Бит 7							Бит 0

бит 7-0 **DEV10:DEV2:** ID микроконтроллера
Эти биты используются совместно с битами DEV2:DEV0 в 1-м ID регистре.

Обозначения		
R = чтение бита	P = программирование бита	U = не используется, читается как ‘0’
- n = не запрограммированное значение		

19.2 Сторожевой таймер WDT

Встроенный сторожевой таймер WDT работает от отдельного RC генератора, не требующего внешних компонентов. Это позволяет работать сторожевому таймеру WDT при выключенном тактовом генераторе (выводы OSC1/CLKI, OSC2/CLKO) в SLEEP режиме микроконтроллера.

В нормальном режиме работы при переполнении WDT происходит сброс микроконтроллера. Если микроконтроллер находится в SLEEP режиме, переполнение WDT выводит его из режима SLEEP с продолжением нормальной работы. Бит –TO в регистре RCON сбрасывается в ‘0’, если произошло переполнение WDT.

Сторожевой таймер может быть включен/выключен в битах конфигурации. Если сторожевой таймер включен, то программное управление WDT заблокировано. Когда бит WDTEN = 0 в регистрах конфигурации, битом SWDTEN можно программно включить/выключить WDT.

Период переполнения WDT сотрите в электрических спецификациях (параметр #31). Коэффициент выходного делителя WDT устанавливается в битах конфигурации.

Примечание. Команды CLRWDT и SLEEP сбрасывают сторожевой таймер и постделитель, если он подключен к WDT, откладывая сброс устройства.

Примечание. Команда CLRWDT сбрасывают сторожевой таймер и постделитель, если он подключен к WDT, но не изменяет коэффициент деления постделителя.

19.2.1 Регистр управления

Бит программного управления WDT расположен в регистре WDTCON. Регистр доступен для записи и чтения. Программное управление работой WDT возможно только, если WDT выключен в битах конфигурации микроконтроллера.

Регистр 19-14. WDTCON: Регистр программного управления WDT

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
-	-	-	-	-	-	-	SWDTEN
Бит 7							Бит 0

бит 7-1 **Не используются:** читаются как ‘0’

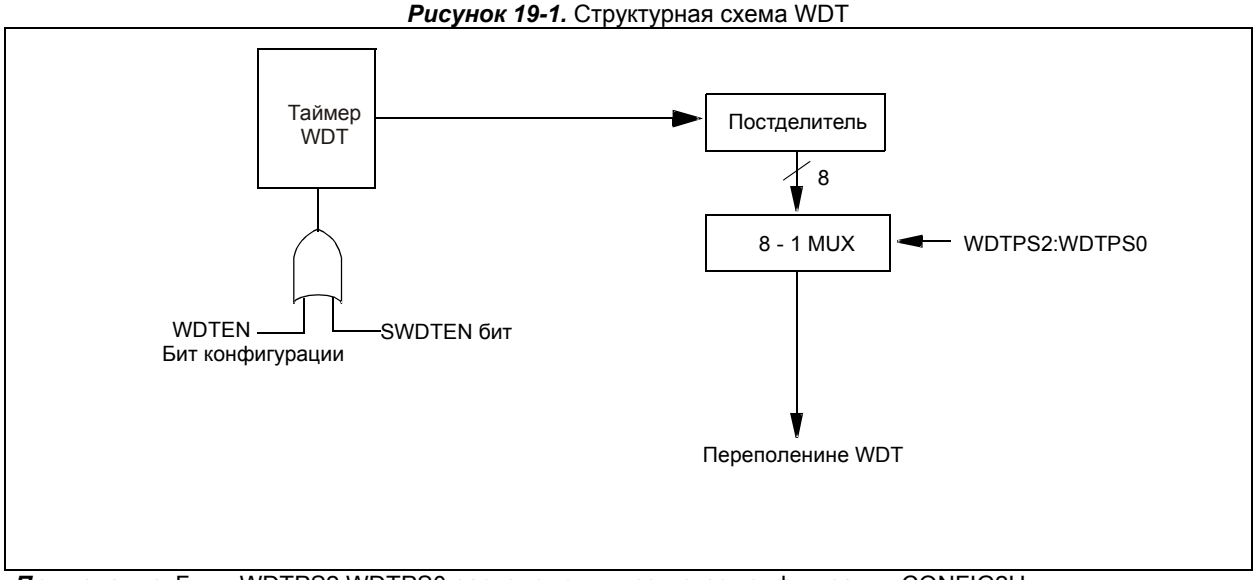
бит 0 **SWDTEN:** Программное включение WDT
1 = WDT включен
0 = WDT выключен, если бит WDTEN в регистрах конфигурации равен ‘0’

Обозначения			
R = чтение бита	W = запись бита	U = не используется, читается как ‘0’	
- n = значение после POR	‘1’ = бит установлен	‘0’ = бит сброшен	X = неизвестное сост.

19.2.2 Постделитель WDT

WDT имеет постделитель, который может увеличить период переполнения WDT. Коэффициент постделителя устанавливается в юитах конфигурации в регистре CONFIG2H.

Рисунок 19-1. Структурная схема WDT



Примечание. Биты WDTPS2:WDTPS0 расположены в регистре конфигурации CONFIG2H.

Таблица 19-2. Регистры и биты, связанные с работой WDT

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0
CONFIG2H	-	-	-	-	WDTPS2	WDTPS1	WDTPS0	WDTEN
RCON	IPEN	-	-	-RI	-TO	-PD	-POR	-BOR
WDTCON	-	-	-	-	-	-	-	SWDTEN

Затененные ячейки на работу не влияют.

19.3 Режим энергосбережения SLEEP

Переход в режим энергосбережения происходит по команде SLEEP.

При переходе в режим SLEEP сторожевой таймер WDT сбрасывается, но продолжает работать. В регистре RCON<3> бит -PD сбрасывается в '0', бит -TO RCON<4> устанавливается в '1', тактовый генератор микроконтроллера выключен. Порты ввода/вывода остаются в том же состоянии, что и до выполнения команды SLEEP (высокий уровень, низкий уровень, третье состояние).

Для снижения энергопотребления в SLEEP режиме все каналы ввода/вывода должны быть подключены к V_{DD} или V_{SS} при отсутствии токов из внешней схемы через выводы портов. Выводы, находящиеся в третьем состоянии, должны иметь высокий или низкий уровень сигнала, чтобы избежать токов переключения входных буферов. Вход T0CKI должен быть подключен к V_{DD} или V_{SS} для снижения энергопотребления. Должны учитываться внутренние подтягивающие резисторы, включенные на входах PORTB. На входе -MCLR должен быть высокий уровень сигнала (V_{IHMC}).

19.3.1 Выход из режима SLEEP

Микроконтроллер выйдет из режима SLEEP по одному из следующих событий:

1. Внешний сброс по сигналу на входе -MCLR
2. Переполнение сторожевого таймера WDT (если он включен)
3. Периферийное прерывание (INT, изменение уровня сигнала на входах RB7:RB4 и др.)

Список прерываний от периферийных модулей, которые могут вывести микроконтроллер из режима SLEEP:

1. Чтение/запись PSP
2. Переполнение TMR1 в режиме асинхронного счетчика
3. Переполнение TMR3 в режиме асинхронного счетчика
4. Прерывание от модуля CCP
5. Триггер специального события (TMR1 должен работать в режиме асинхронного счетчика)
6. Обнаружение START/STOP на шине I^2C модулем MSSP
7. Прием/передача байта в режиме ведомого SPI/I^2C
8. Прием/передача USART в ведомом синхронном режиме
9. Завершение преобразования АЦП (когда используется внутренний RC генератор для АЦП)
10. Завершение записи в EEPROM
11. Прерывание от модуля LVD

Другие прерывания от периферийных модулей не могут вывести микроконтроллер из режима SLEEP.

Внешний сигнал -MCLR вызывает сброс микроконтроллера. Другие события вызывают продолжение выполнения программы. Биты -TO и -PD в регистре RCON могут использоваться для определения причины сброса микроконтроллера. Бит -PD сбрасывается в '0' при переходе в режим SLEEP. Бит -TO сбрасывается в '0' если произошло переполнение WDT.

При выполнении команды SLEEP происходит предвыборка следующей инструкции (PC+2). Если прерывание должно выводить микроконтроллер из режима SLEEP, соответствующий бит разрешения прерывания устанавливается в '1'. Микроконтроллер выходит из режима SLEEP независимо от состояния бита GIE. Если GIE=0, выполняется следующая инструкция после SLEEP без перехода по вектору прерываний. Если GIE=1, исполняется следующая инструкция после SLEEP и происходит переход на подпрограмму обработки прерываний. Когда выполнение какой-либо команды при выходе из режима SLEEP нежелательно, необходимо поле команды SLEEP использовать инструкцию NOP.

19.3.2 Выход из режима SLEEP по прерыванию

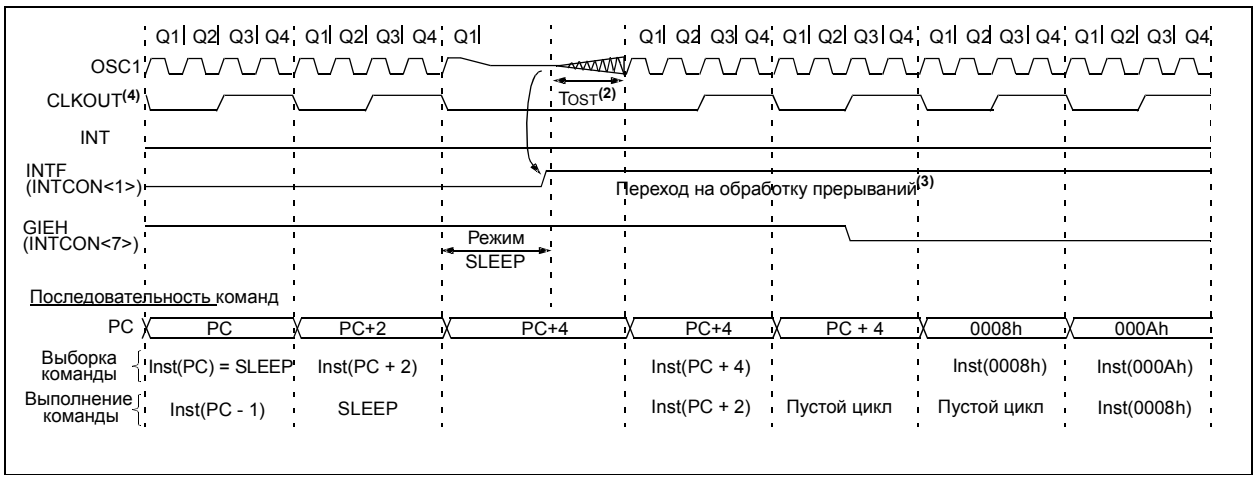
Когда бит глобального разрешения прерываний GIE сброшен в '0', а бит разрешения периферийных прерываний и соответствующий флаг прерывания установлен в '1', то возникнет одно из следующих событий:

- Если прерывание возникает перед выполнением команды SLEEP, то вместо инструкции SLEEP будет выполнен пустой цикл NOP, WDT и постделитель WDT не будут сброшены, бит -TO не будет установлен в '1', а бит -PD не будет сброшен в '0'.
- Если прерывание возникает в течение или после выполнения инструкции SLEEP, то микроконтроллер немедленно выйдет из режима SLEEP, а команда SLEEP выполняется полностью. WDT и постделитель WDT сброшены, бит -TO установлен в '1', бит -PD сброшен в '0'.

Даже если флаги прерываний были проверены перед выполнением команды SLEEP, они могут быть установлены в течение выполнения инструкции SLEEP. Для контроля полного выполнения команды SLEEP проверьте состояние бита -PD. Если -PD = 1, то вместо инструкции SLEEP был выполнен пустой цикл NOP.

Для гарантированного сброса WDT перед инструкцией SLEEP рекомендуется использовать команду CLRWDT.

Рисунок 19-2. Временная диаграмма выхода микроконтроллера из режима SLEEP по прерыванию с входа INT



Примечания:

1. Режим генератора XT, HS или LP.
2. Предполагается, что GIE=1. После выхода из режима SLEEP произойдет переход по вектору прерывания.
3. T_{OST} = 1024 T_{OSC} (не масштабный рисунок). Для RC режима генератора задержка отсутствует.
4. CLKOUT не доступен для этих режимов генератора, но показан для пояснения диаграммы.

19.4 Верификация и защита кода программы

Структура защиты Flash памяти микроконтроллеров PIC18 существенно отличается от других PICmicro.

Пользовательская память программ разделена на блоки. Один из блоков – бок загрузчика 512 байт. Остальная память разделена на 4 блока с бинарными границами.

Каждый из пяти блоков имеет три управляющих бита защиты:

- Бит защиты (CPn)
- Бит защиты записи (WRTn)
- Бит защиты внешнего табличного чтения (EBTRn)

На рисунке 19-3 представлена организация памяти программ с 16- и 32-килобайтными памяти и биты защиты, связанные с каждым блоком. Фактическое расположение битов смотрите в таблице 19-3.

Рисунок 19-3. Защита памяти программ в микроконтроллерах PIC18FXX2

Размер памяти/микроконтроллер			
16 кбайт (PIC18FX42)	**32 кбайт (PIC18FX52)**	**Диапазон адресов**	**Биты защиты блока памяти программ:**
Загрузочный блок	Загрузочный блок	000000h 0001FFh	CPB, WRTB, EBTRB
Блок 0	Блок 0	000200h 001FFFh	CP0, WRT0, EBTR0
Блок 1	Блок 1	002000h 003FFFh	CP1, WRT1, EBTR1
Не реализовано читается как ‘0’	Блок 2	004000h 005FFFh	CP2, WRT2, EBTR2
Не реализовано читается как ‘0’	Блок 3	006000h 007FFFh	CP3, WRT3, EBTR3
Не реализовано читается как ‘0’	Не реализовано читается как ‘0’	008000h 1FFFFFh	(Не реализованная область памяти)

Таблица 19-3. Регистры и биты, связанные с защитой кода

Имя		Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Не запрог. значение
300008h	CONFIG5L	-	-	-	-	CP3	CP2	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	-	-	-	-	-	-	11-- ----
30000Ah	CONFIG6L	-	-	-	-	WRT3	WRT2	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	-	-	-	-	-	111- ----
30000Ch	CONFIG7L	-	-	-	-	EBTR3	EBTR2	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	-	EBTRB	-	-	-	-	-	-	-1-- ----

Затененные ячейки на работу не влияют.

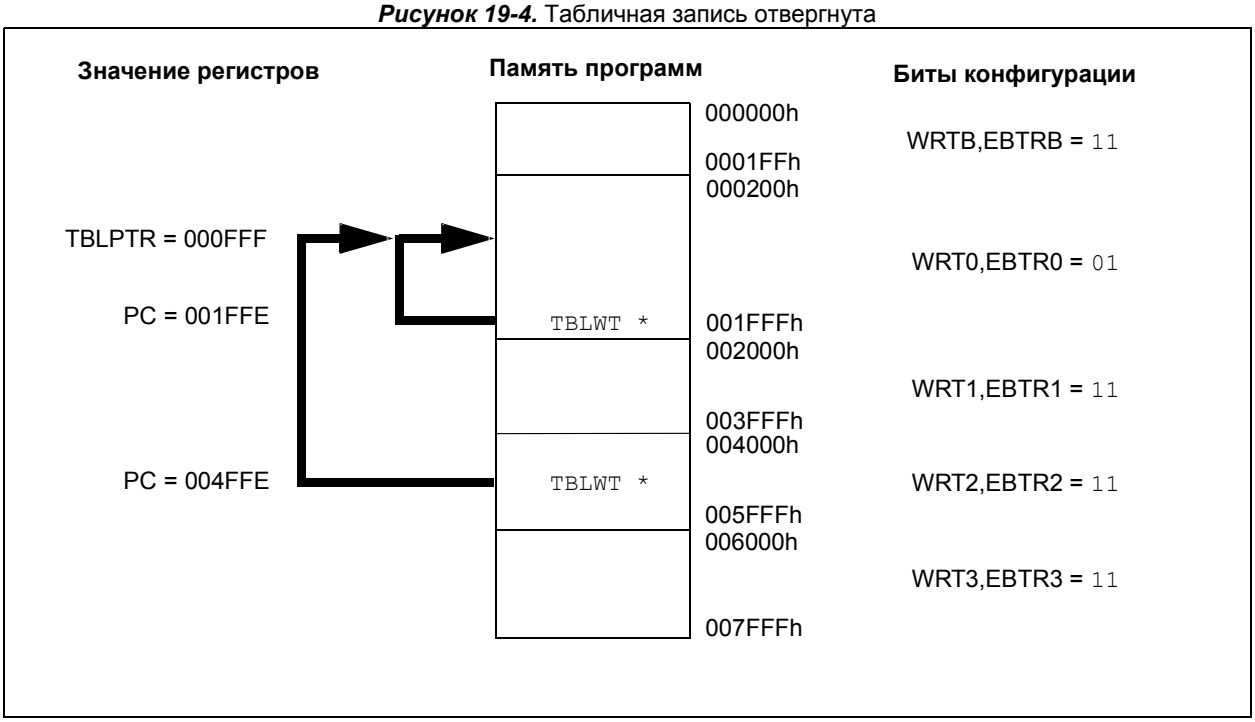
19.4.1 Защита памяти программ

Пользовательская память программ может бить прочитана или записана командами табличного чтения/записи. ID микроконтроллера может быть прочитано командой табличного чтения. Биты конфигурации могут быть изменены табличным чтением/записью.

Для работы программы пользователя биты CPn не оказывают прямого эффекта. Биты CPn запрещают внешнее чтение/запись. Блок памяти программ может быть защищен от табличной записи, если бит WRTn равен ‘0’. Биты EBTRn управляют табличным чтением. Если бит EBTRn запрограммирован в ‘0’, то табличное чтение блока возможно только командами чтения, выполняемыми в этом блоке. Чтение командами из других блоков заблокировано (чтение будет давать результат ‘0’). На рисунках 19-4, 19-5 и 19-6 смотрите пояснения операций табличного чтения, записи и чтения с включенной защитой.

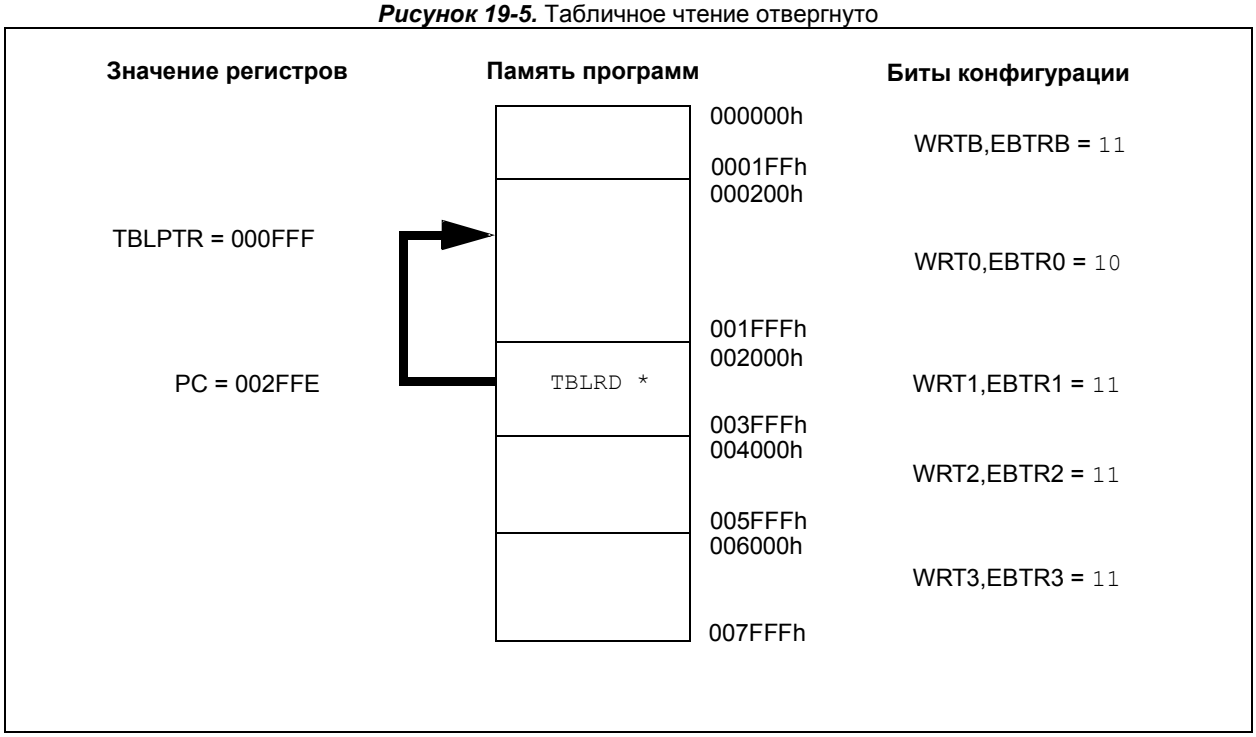
Примечание. Биты защиты от чтения EBTRn могут быть только запрограммированы в ‘0’ из ‘1’. Не возможно записать в биты EBTRn ‘1’, если они в ‘0’. Биты EBTRn устанавливаются в ‘1’ только при полном стирании памяти микроконтроллера. Команды стирания блока и полного стирания памяти могут быть выполнены только по интерфейсу ICSP внешним программатором.

Рисунок 19-4. Табличная запись отвергнута



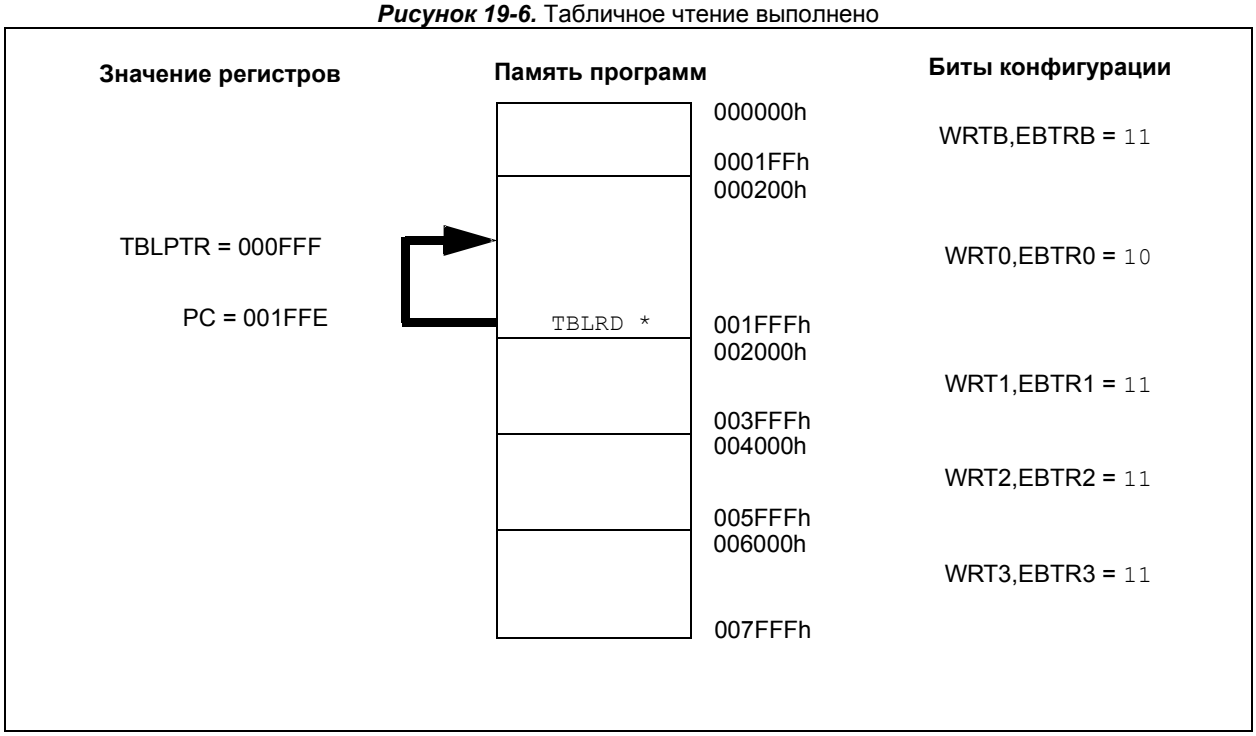
Результат. Любая табличная запись игнорируется, когда WRTn=0.

Рисунок 19-5. Табличное чтение отвергнуто



Результат. Любое табличное чтение между блоками памяти программ запрещено, когда EBTRn=0. В TABLAT помещается значение ‘0’.

Рисунок 19-6. Табличное чтение выполнено



Результат. Табличное чтение разрешено внутри блока памяти, когда EBTRn=0. В TABLAT помещается значение по указателю TBLPTR.

19.4.2 Защита EEPROM памяти данных

EEPROM память данных может быть защищена от внешнего чтения/записи битами конфигурации CPD, WRTD. Если запрограммирован бит CPD в ‘0’, то запрещено внешнее чтение/запись EEPROM памяти данных. Если запрограммирован только бит WRTD, то запрещена внешняя запись в EEPROM память данных. Защита EEPROM памяти данных не оказывает влияние на операции с памятью командами микроконтроллера.

19.4.3 Защита регистров конфигурации

Регистры конфигурации могут быть защищены от записи. Бит WRTC управляет защитой регистров конфигурации. Командами микроконтроллера бит WRTC доступен только для чтения. Изменить бит WRTC можно только по протоколу программирования ICSP.

19.5 Размещение идентификатора ID

Восемь ячеек памяти программ (200000h-200007h) предназначены для размещения идентификатора, которые могут использоваться для сохранения контрольной суммы или другой информации. Доступ к ID регистрам возможен в нормальном режиме работы командами TBLRD и TBLWT. Регистры ID могут быть прочитаны при включенной защите кода.

19.6 Внутрисхемное программирование ICSP

Микроконтроллеры PIC18FXX2 могут быть запрограммированы по последовательному интерфейсу в готовом изделии. Программирование выполняется по двум линиям последовательно интерфейса (данные, синхронизация) и трем дополнительным линиям: напряжение питания, общий провод, напряжение программирования. Это позволяет изготавливать платы с не запрограммированными микроконтроллерами, а затем загружать в них программу перед поставкой изделия. Данная функция также позволяет обновлять программное обеспечение микроконтроллеров.

19.7 Внутрисхемный отладчик ICD

Если бит DEBUG в регистре конфигурации CONFIG4Lравен нулю, то разрешен режим внутрисхемной отладки. Эта функция предоставляет простые функции отладки кода программы при использовании MPLAB IDE. Для работы отладчика используется часть ресурсов микроконтроллера, показанных в таблице 19-4.

Таблица 19-4. Ресурсы, используемые для режима внутрисхемной отладки

Порты ввода/вывода	RB6, RB7
Стек	2 уровня
Память программ	512 байт
Память данных	10 байт

Для использования режима внутрисхемной отладки схема устройства должна предусматривать возможность подключения к выводам -MCLR/V_{PP}, V_{DD}, GND, RB6 и RB7, аналогично режиму внутрисхемного программирования ICSP.

19.8 Режим низковольтного программирования

Бит LVP в регистре конфигурации CONFIG4L используется для разрешения режима низковольтного программирования. Этот режим позволяет запрограммировать микроконтроллер по интерфейсу ICSP при одном источнике питания (не требуется подавать напряжение V_{IHH} на вывод -MCLR). По умолчанию LVP=1, разрешая низковольтное программирование. При этом вывод RB5/PGH используется для низковольтного программирования и перестает быть цифровым портом ввода/вывода. Микроконтроллер переходит в режим программирования, когда на выводе RB5/PGM высокий уровень сигнала. Режим стандартного программирования по прежнему доступен (когда на выводе -MCLR напряжение V_{IHH}).

Примечания:

1. Режим стандартного программирования всегда доступен, независимо от состояния бита LVP.
2. В режиме низковольтного программирования вывод RB5/PGM не может использоваться как цифровой порт ввода/вывода.
3. В режиме низковольтного программирования бит TRISB<5> должен быть сброшен в '0' для отключения подтягивающего резистора на входе.

Если режим низковольтного программирования не используется, бит LVP должен быть сброшен в '0', вывод RB5/PGM становится цифровым портом ввода/вывода. Бит LVP может быть изменен только в стандартном режиме программирования (когда на выводе -MCLR напряжение V_{IHH}). Когда бит LVP=0, возможен только стандартный режим программирования/проверки микроконтроллера.

В режиме программирования ICSP при выполнении операции стирания всей памяти (включая снятие защиты) напряжение питания должно быть от 4.5В до 5.5В. Все остальные операции программирования могут быть выполнены во всем диапазоне напряжений питания.

20. Описание системы команд

Набор команд PIC18FXXX несколько расширен по сравнению с предыдущими версиями PICmicro, но обеспечивает легкость переноса программы, написанной для микроконтроллеров PICmicro среднего семейства.

Большинство команд занимают одно слово в памяти программ (16 бит), но есть 4 команды, для которых необходимо два слова в памяти программ.

Каждая команда состоит из одного 16-разрядного слова, разделенного на код операции (OPCODE), определяющий тип команды и один или несколько операндов, определяющие операцию команды.

Система команд ортогональна, все команды разделена на четыре основных группы:

- **Байт-ориентированные** команды
- **Бит-ориентированные** команды
- Операции с **константами**
- **Управляющие** команды

Сводный список команд PIC18FXXX смотрите в таблице 20-2. Описание полей команды и дескрипторов смотрите в таблице 20-1.

Байт-ориентированные команды имеют следующие операнды:

- Регистр (‘f’), к которому выполняется обращение
- Размещение результата команды (‘d’)
- Доступ к памяти (‘a’)

Для байт-ориентированных команд 'f' является указателем регистра, а 'd' указателем адресата результата. Указатель регистра определяет, какой регистр должен использоваться в команде. Указатель адресата определяет, где будет сохранен результат. Если 'd'=0, результат сохраняется в регистре W. Если 'd'=1, результат сохраняется в регистре, который используется в команде.

Бит-ориентированные команды имеют следующие операнды:

- Регистр (‘f’), к которому выполняется обращение
- Бит в регистре (‘b’)
- Доступ к памяти (‘a’)

В бит-ориентированных командах 'b' определяет номер бита участвующего в операции, а 'f' - указатель регистра, который содержит этот бит.

Команды операций с **константами** могут иметь операнды:

- Константа, которая будет загружена в регистр (‘k’)
- Регистр FSR, чтобы загрузить значение в указываемый регистр
- Нет операндов (‘-‘)

Команды **управления** могут содержать следующие операнды:

- Адрес в памяти программ (‘n’)
- Режим выполнения команд вызова (Call) или возврата (Return) (‘s’)
- Режим выполнения команд табличного чтения/записи (‘m’)
- Нет операндов (‘-‘)

Практически все команды занимают одно слово в памяти программ, кроме четырех команд, занимающих два слова в памяти программ. Эти команды занимают два слова из-за большого числа операндов (команда 32 бита). Во втором слове 4 старших бита всегда равны ‘1’. Если второе слово команды будет выполнено как отдельная команда, то вместо каких-либо операций выполняется пустой цикл NOP.

Все однословные команды выполняются за один машинный цикл, кроме команд, условие которых истинно или в результате исполнения команды изменяется счетчик команд PC. В этом случае команда исполняется за два цикла, дополнительно выполняя пустой цикл NOP.

Двухсловные команды выполняются за два цикла.

Один машинный цикл состоит из 4-х периодов тактового сигнала. Таким образом, при тактовой частоте 4МГц нормальная длительность выполнения команды 1мкс. Если условие команды истинно или изменяется значение счетчика команд, то команда выполняется за два машинных цикла 2мкс.

На рисунке 20-1 представлены общие форматы команд.

Во всех примерах применяется формат чисел “nnh’, чтобы представить шестнадцатеричное число, где h - шестнадцатеричная цифра.

Мнемоника команд, поддерживаемая ассемблером MPASM, показана в таблице 20-2.

Подробное описание каждой команды смотрите в разделе 20.1.

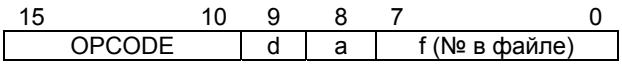
Таблица 20-1. Описание полей команды и дескрипторов

Обозначение	Описание
a	Бит доступа к памяти: a = 0 : Обращение к ОЗУ быстрого доступа (значение регистра BSR игнорируется) a = 1 : Обращение к ОЗУ с учетом значения регистра BSR
bbb	Номер бита в 8-разрядном регистре (от 0 до 7)
BSR	Регистр выбора банка памяти. Используется для выбора текущего банка памяти.
d	Бит размещения результата команды: d = 0 : Результата помещается в регистр WREG d = 1 : Результат сохраняется в регистре ‘f’
dest	Адресат результата: регистр WREG или регистр ‘f’ в ОЗУ
f	8-разрядный адрес регистра (от 0x00 до 0xFF)
fs	12-разрядный адрес регистр источника (от 0x000 до 0xFFF)
fd	12-разрядный адрес регистр приемника (от 0x000 до 0xFFF)
k	Константа или метка (8-, 12- или 20-разрядное значение)
label	Имя метки
mm * *+ *- +*	Режим работы регистр TBLPTR при операциях табличного чтения/записи (может использоваться только в операциях табличного чтения/записи). Значение регистра TBLPTR не изменяется Значение регистра TBLPTR пост - инкрементируется Значение регистра TBLPTR пост - декрементируется Значение регистра TBLPTR пред - инкрементируется
n	Относительный адрес (знаковый) для команд относительного перехода или абсолютный адрес для команд вызова подпрограмм, перехода и возврата
PRODH	Старший байт результата умножения
PRODL	Младший байт результата умножения
s	Бит быстрого вызова подпрограммы/возврата s = 0 : значение дополнительных регистров не используется s = 1 : значение некоторых регистров сохраняется/восстанавливается в дополнительных регистрах (быстрый режим)
u	Не используется или не реализовано
WREG	Рабочий регистр (аккумулятор)
TBLPTR	21-разядный табличный указатель (указатель в памяти программ)
TABLAT	8-разрядная защелка табличного чтения/записи
TOS	Вершина стека
PC	Счетчик команд
PCL	Младший байт счетчика команд
PCH	Старший байт счетчика команд
PCLATH	Защелка старшего байта счетчика команд
PCLATU	Защелка верхнего байта счетчика команд
GIE	Бит глобального разрешения прерываний
WDT	Сторожевой таймер
-TO	Бит переполнения WDT
-PD	Бит включения питания
C, DC, Z, OV, N	Флаги АЛУ: перенос, десятичный перенос, нуль, переполнение, отрицательный результат
[ ]	Опционально
( )	Контекст
$\rightarrow$	Присвоение
< >	Битовое поле
$\in$	Из набора

Рисунок 20-1. Общий формат команд

Байт ориентированные команды с регистрами

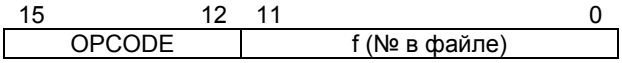
Пример

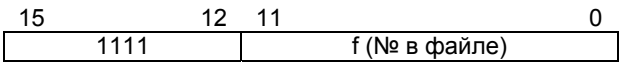


ADDWF MYREG, W, B

d = 0 - результата помещается в регистр WREG
d = 1 - результат сохраняется в регистре ‘f’
a = 0 : Обращение к ОЗУ быстрого доступа
a = 1 : Обращение к ОЗУ с учетом значения регистра BSR
f = 8-разрядный адрес регистра

Команды перемещения **Байт-Байт**

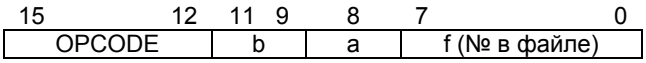




MOVFF MYREG1, MYREG2

f = 12-разрядный адрес регистра

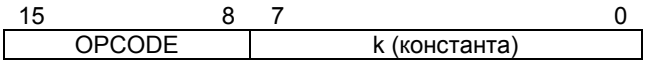
Бит ориентированные операции с регистрами



BSF MYREG, bit, B

b = номер бита в 8-разрядном регистре
a = 0 : Обращение к ОЗУ быстрого доступа
a = 1 : Обращение к ОЗУ с учетом значения регистра BSR
f = 8-разрядный адрес регистра

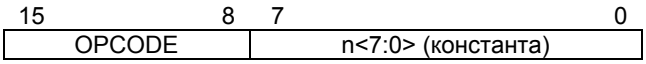
Операции с **константами**

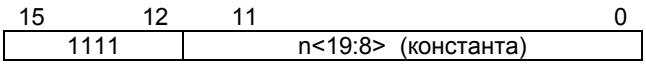


MOVLW 0x7F

k = 8-разрядная константа

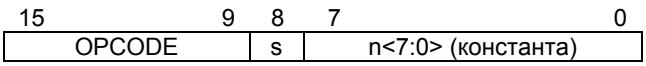
Команды управления

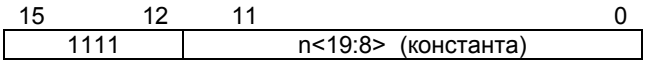




GOTO Label

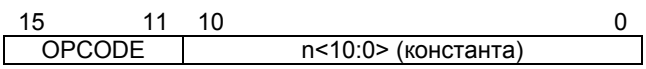
n = 20-разрядная константа



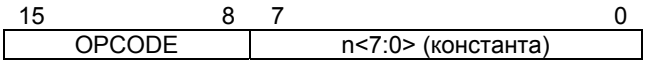


CALL MYFUNC

s = бит быстрого вызова подпрограммы/возврата



BRA MYFUNC



BC MYFUNC

Таблица 20-2. Список команд PIC18FXXX

Мнемоника и операнды		Описание	Циклов	Слово команды (16 бит) Ст.			Мл.	Воздействие на флаги АЛУ	Прим.
Байт ориентированные команды с регистрами									
ADDWF	f, d, a	Сложение WREG и f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Сложение WREG, f и бита C	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	Логическое И WREG и f	1	0001	01da	ffff	ffff	Z, N	1, 2
CLRF	f, a	Очистка f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Инверсия f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Сравнить WREG и f, проп. если =	1(2 или 3)	0110	001a	ffff	ffff	-	4
CPFSGT	f, a	Сравнить WREG и f, проп. если >	1(2 или 3)	0110	010a	ffff	ffff	-	4
CPFSLT	f, a	Сравнить WREG и f, проп. если <	1(2 или 3)	0110	000a	ffff	ffff	-	4
DECF	f, d, a	Декремент f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1,2,3,4
DECFSZ	f, d, a	Декремент f, пропустить если 0	1(2 или 3)	0010	11da	ffff	ffff	-	1,2,3,4
DCFSNZ	f, d, a	Декремент f, пропустить если не 0	1(2 или 3)	0100	11da	ffff	ffff	-	1,2,3,4
INCF	f, d, a	Инкремент f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1,2,3,4
INCFSZ	f, d, a	Инкремент f, пропустить если 0	1(2 или 3)	0011	11da	ffff	ffff	-	1,2,3,4
INFSNZ	f, d, a	Инкремент f, пропустить если не 0	1(2 или 3)	0100	10da	ffff	ffff	-	1,2,3,4
IORWF	f, d, a	Логическое ИЛИ WREG и f	1	0001	01da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Переместить f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	fs, fd	Переместить fs (источник) 1-е слово в fd (приемник) 2-е слово	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	-	
MOVWF	f, a	Переместить WREG в f	1	0110	111a	ffff	ffff	-	
MULWF	f, a	Умножение WREG и f	1	0000	001a	ffff	ffff	-	
NEGF	f, a	Негативное значение f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	1, 2
RLCF	f, d, a	Сдвиг влево через перенос	1	0011	01da	ffff	ffff	C, Z, N	
RLNCF	f, d, a	Сдвиг влево без переноса	1	0100	01da	ffff	ffff	Z, N	1, 2
RRCF	f, d, a	Сдвиг вправо через перенос	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Сдвиг вправо без переноса	1	0100	00da	ffff	ffff	Z, N	1, 2
SETF	f	Установить все биты f	1	0110	100a	ffff	ffff	-	
SUBFWB	f, d, a	Вычитание f из WREG с заемом	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWF	f, d, a	Вычитание WREG из f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Вычитание WREG из f с заемом	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	1, 2
SWAPF	f, d, a	Поменять местами полубайты в f	1	0011	10da	ffff	ffff	-	4
TSTFSZ	f, a	Тест f, пропустить если 0	1(2 или 3)	0110	011a	ffff	ffff	-	1, 2
XORWF	f, d, a	Лог. исключающее ИЛИ WREG и f	1	0001	10da	ffff	ffff	Z, N	1, 2
Бит ориентированные операции с регистрами									
BCF	f, b, a	Сброс бита в f	1	1001	bbba	ffff	ffff	-	1, 2
BSF	f, b, a	Установка бита в f	1	1000	bbba	ffff	ffff	-	1, 2
BTFSC	f, b, a	Тест бита, пропустить если ‘0’	1(2 или 3)	1011	bbba	ffff	ffff	-	3, 4
BTFSS	f, b, a	Тест бита, пропустить если ‘1’	1(2 или 3)	1010	bbba	ffff	ffff	-	3, 4
BTG	f, b, a	Инверсия бита в f	1	0111	bbba	ffff	ffff	-	1, 2

Примечания:

1. При выполнении операции «чтение-модификация-запись» с портом ввода вывода (например, MOVF PORTB, 1, 0) исходное значение считывается с выводов порта, а не из выходных защелок. Например, в защелке данных записано ‘1’, а вывод настроен как вход и на этом входе сигнал с уровнем ‘0’, обратно в защелку будет записано значение ‘0’.
2. При записи в TMR0 (и в команде бит d = 1) предделитель TMR0 сбрасывается, если он подключен к TMR0.
3. Если условие истинно, или изменяется счетчик команд PC, то команда выполняется за два цикла. Во втором цикле выполняется NOP.
4. Некоторые команды состоят из двух 16-разрядных слов. Если по каким-то причинам счетчик команд попадет на 2-е слово команды, то оно будет выполнено как NOP.
5. Если производится запись во внутреннюю память, то следующая команда не начнет выполняться до тех пор, пока не закончится цикл записи.

Таблица 20-2. Список команд PIC18FXXX (продолжение)

Мнемоника и операнды		Описание	Циклов	Слово команды (16 бит) Ст.			Мл.	Воздействие на флаги АЛУ	Прим.
Команды управления									
BC	n	Переход, если перенос (C = 1)	1(2)	1110	0010	nnnn	nnnn	-	
BN	n	Переход, если нег. результ. (N = 1)	1(2)	1110	0110	nnnn	nnnn	-	
BNC	n	Переход, если нет переноса (C = 0)	1(2)	1110	0011	nnnn	nnnn	-	
BNN	n	Переход, если пол. результ. (N = 0)	1(2)	1110	0111	nnnn	nnnn	-	
BNOV	n	Переход, если нет переполн. (OV = 0)	1(2)	1110	0101	nnnn	nnnn	-	
BNZ	n	Переход, если не нуль (Z = 0)	2	1110	0001	nnnn	nnnn	-	
BOV	n	Переход, если переполнение (OV = 1)	1(2)	1110	0100	nnnn	nnnn	-	
BRA	n	Безусловный переход	1(2)	1101	0nnn	nnnn	nnnn	-	
BZ	n	Переход, если нуль (Z = 1)	1(2)	1110	0000	nnnn	nnnn	-	
CALL	n, s	Переход на подпрограмму. 1-е слово	2	1110	110s	kkkk	kkkk	-	
		2-е слово		1111	kkkk	kkkk	kkkk		
CLRWDT	-	Сбросить сторожевой таймер	1	0000	0000	0000	0100	-TO, -PD	
DAW	-	Десятичная коррекция WREG	1	0000	0000	0000	0111	C	
GOTO	n	Переход по адресу, 1-е слово	2	1110	1111	kkkk	kkkk	-	
		2-е слово		1111	kkkk	kkkk	kkkk		
NOP	-	Нет операции	1	0000	0000	0000	0000	-	
NOP	-	Нет операции	1	1111	xxxx	xxxx	xxxx	-	4
POP	-	Чтение вершины стека возврата TOS	1	0000	0000	0000	0110	-	
PUSH	-	Запись в вершину стека возврата TOS	1	0000	0000	0000	0101	-	
RCALL	n	Короткий переход на подпрограмму	2	1101	1nnn	nnnn	nnnn	-	
RESET	-	Программный сброс	1	0000	0000	1111	1111	все	
RETFIE	s	Возврат из пп с разреш. прерываний	2	0000	0000	0001	000s	GIEH/GIEL	
RETLW	k	Возврат из пп с загрузкой WREG	2	0000	1100	kkkk	kkkk	-	
RETURN	s	Возврат из подпрограммы	2	0000	0000	0001	001s	-	
SLEEP	-	Переход в SLEEP режим	1	0000	0000	0000	0011	-TO, -PD	
Операции с константами									
ADDLW	k	Прибавить константу к WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	Логическое И константы и WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	Логическое ИЛИ константы и WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	Поместить константу (12 бит) в	2	1110	1110	00ff	kkkk	-	
		FSR (2 слова)		1111	0000	kkkk	kkkk		
MOVLB	k	Поместить константу в BSR<3:0>	1	0000	0001	0000	kkkk	-	
MOVLW	k	Поместить константу в WREG	1	0000	1110	kkkk	kkkk	-	
MULLW	k	Умножение константы на WREG	1	0000	1101	kkkk	kkkk	-	
RETLW	k	Возврат из пп с загрузкой WREG	2	0000	1100	kkkk	kkkk	-	
SUBLW	k	Вычитание WREG из константы	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	Лог. исключ. ИЛИ константы и WREG	1	0000	1010	kkkk	kkkk	Z, N	
Операции память данных ↔ память программ									
TBLRD*		Табличное чтение	2	0000	0000	0000	1000	-	
TBLRD*+		Табличное чтение с пост-инкрементом	2	0000	0000	0000	1001	-	
TBLRD*-		Табличное чтение с пост-декрементом	2	0000	0000	0000	1010	-	
TBLRD+*		Табличное чтение с пред-инкрементом	2	0000	0000	0000	1011	-	
TBLWT*		Табличная запись	2	0000	0000	0000	1100	-	5
TBLWT*+		Табличная запись с пост-инкрементом	2	0000	0000	0000	1101	-	5
TBLWT*-		Табличная запись с пост-декрементом	2	0000	0000	0000	1110	-	5
TBLWT+*		Табличная запись с пред-инкрементом	2	0000	0000	0000	1111	-	5

Примечания:

1. При выполнении операции «чтение-модификация-запись» с портом ввода вывода (например, MOVF PORTB, 1, 0) исходное значение считывается с выводов порта, а не из выходных защелок. Например, в защелке данных записано ‘1’, а вывод настроен как вход и на этом входе сигнал с уровнем ‘0’, обратно в защелку будет записано значение ‘0’.
2. При записи в TMR0 (и в команде бит d = 1) предделитель TMR0 сбрасывается, если он подключен к TMR0.
3. Если условие истинно, или изменяется счетчик команд PC, то команда выполняется за два цикла. Во втором цикле выполняется NOP.
4. Некоторые команды состоят из двух 16-разрядных слов. Если по каким-то причинам счетчик команд попадет на 2-е слово команды, то оно будет выполнено как NOP.
5. Если производится запись во внутреннюю память, то следующая команда не начнет выполняться до тех пор, пока не закончится цикл записи.

20.1 Подробное описание команд

ADDLW Прибавить константу к WREG

Синтаксис: [*label*] ADDLW k

Операнды: $0 \le k \le 255$

Операция: $(W) + k \rightarrow W$

Измен. флаги: N, OV, C, DC, Z

Код:

0000	1111	kkkk	kkkk

Описание: Содержимое регистра W складывается с 8-разрядной константой ‘k’, результат сохраняется в регистр W

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр W

Пример:

```
ADDLW   0x15
```

До выполнения команды

W = 0x10

После выполнения команды

W = 0x25

ADDWF Сложение WREG и f

Синтаксис: [*label*] ADDWF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: $(W) + (f) \rightarrow dest$

Измен. флаги: N, OV, C, DC, Z

Код:

0010	01da	ffff	ffff

Описание: Сложение содержимого регистров W и ‘f’. Если d=0, результат сохраняется в регистре W, если d=1, то в регистре’ f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
ADDWF   REG, 0, 0
```

До выполнения команды

W = 0x17

REG = 0xC2

После выполнения команды

W = 0xD9

REG = 0xC2

ADDWFC Сложение WREG, f и бита C

Синтаксис: [*label*] ADDWFC f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: (W) + (f) + (C) → dest

Измен. флаги: N, OV, C, DC, Z

Код:

0010	00da	ffff	ffff

Описание: Сложение содержимого регистров W и ‘f’ и бита C. Если d=0, результат сохраняется в регистре W, если d=1, то в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример: ADDWFC REG, 0, 1

До выполнения команды
C = 1
REG = 0x4D
W = 0x02

После выполнения команды
C = 0
REG = 0x02
W = 0x50

ANDLW Логическое И константы и WREG

Синтаксис: [*label*] ANDLW k

Операнды: $0 \le k \le 255$

Операция: (W) . AND . k → W

Измен. флаги: N, Z

Код:

0000	1011	kkkk	kkkk

Описание: Операция И с содержимым регистра W и 8-разрядной константой ‘k’. Результат операции сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр W

Пример: ANDLW 0x5F

До выполнения команды
W = 0xA3

После выполнения команды
W = 0x03

ANDWF Логическое И WREG и f

Синтаксис: [*label*] ANDWF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: (W) . AND . (f) → dest

Измен. флаги: N, Z

Код:

0001	01da	ffff	ffff

Описание: Логическая операция поразрядного И регистров W и ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример: ANDWF REG, 0, 0

До выполнения команды
W = 0x17
REG = 0xC2

После выполнения команды
W = 0x02
REG = 0xC2

BC Переход, если перенос (C = 1)

Синтаксис: [*label*] BC n

Операнды: $-128 \le n \le 127$

Операция: Если C = 1
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0010	nnnn	nnnn

Описание: Если C=1, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BC 5

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если C = 1 PC = адрес (HERE + 12)
Если C = 0 PC = адрес (HERE + 2)

BCF Сброс бита в f

Синтаксис: [*label*] BCF f,b[,a]

Операнды: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Операция: $0 \rightarrow f<b>$

Измен. флаги: Нет

Код:

1001	bbba	ffff	ffff

Описание: Сброс бита ‘b’ в регистре ‘f’. Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример:

```
BCF     FLAG_REG, 7, 0
```

До выполнения команды
FLAG_REG = 0xC7
После выполнения команды
FLAG_REG = 0x47

BN Переход, если нег. резулт. (N = 1)

Синтаксис: [*label*] BN n

Операнды: $-128 \le n \le 127$

Операция: Если N = 1
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0110	nnnn	nnnn

Описание: Если N=1, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример:

```
HERE    BN      Jump
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
Если N = 1 PC = адрес (Jump)
Если N = 0 PC = адрес (HERE + 2)

BNC Переход, если нет переноса (C = 0)

Синтаксис: [*label*] BNC n

Операнды: $-128 \le n \le 127$

Операция: Если C = 0
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0011	nnnn	nnnn

Описание: Если C=0, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BNC Jump

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если C = 0 PC = адрес (Jump)
Если C = 1 PC = адрес (HERE + 2)

BNN Переход, если пол. резулт. (N = 0)

Синтаксис: [*label*] BNN n

Операнды: $-128 \le n \le 127$

Операция: Если N = 0
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0111	nnnn	nnnn

Описание: Если N=0, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BNN Jump

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если N = 0 PC = адрес (Jump)
Если N = 1 PC = адрес (HERE + 2)

BNOV Переход, если нет переполн. (OV = 0)

Синтаксис: [*label*] BNOV n

Операнды: $-128 \le n \le 127$

Операция: Если OV = 0
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0101	nnnn	nnnn

Описание: Если OV=0, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BNOV Jump

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если OV=0 PC = адрес (Jump)
Если OV=1 PC = адрес (HERE + 2)

BNZ Переход, если не нуль (Z = 0)

Синтаксис: [*label*] BNZ n

Операнды: $-128 \le n \le 127$

Операция: Если Z = 0
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0001	nnnn	nnnn

Описание: Если Z=0, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BNZ Jump

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если Z = 0 PC = адрес (Jump)
Если Z = 1 PC = адрес (HERE + 2)

BRA Безусловный переход

Синтаксис: [*label*] BRA n

Операнды: $-1024 \le n \le 1023$

Операция: $(PC) + 2 + 2n \rightarrow PC$

Измен. флаги: Нет

Код:

1101	0nnn	nnnn	nnnn

Описание: Командой выполняется безусловный переход. Значение PC будет равно PC + 2 + 2n. Команда исполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    BRA     Jump
```

До выполнения команды

PC = адрес (HERE)

После выполнения команды

PC = адрес (Jump)

BSF Установка бита в f

Синтаксис: [*label*] BSF f,b[,a]

Операнды: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Операция: $1 \rightarrow f<b>$

Измен. флаги: Нет

Код:

1000	bbba	ffff	ffff

Описание: Установка бита ‘b’ в регистре ‘f’. Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример:

```
BSF     FLAG_REG, 7, 1
```

До выполнения команды

FLAG_REG = 0x0A

После выполнения команды

FLAG_REG = 0x8A

BTFSC Тест бита, пропустить если ‘0’

Синтаксис: [*label*] BTFSC f,b[,a]

Операнды: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Операция: Пропустить, если (f<b>) = 0

Измен. флаги: Нет

Код:

1011	bbba	ffff	ffff

Описание: Если бит ‘b’ в регистре ‘f’ = 0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    BTFSC FLAG, 1, 0
FALSE   :
TRUE    :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
FLAG<1>=0 PC = адрес (TRUE)
FLAG<1>=1 PC = адрес (FALSE)

BTFSS Тест бита, пропустить если ‘1’

Синтаксис: [*label*] BTFSS f,b[,a]

Операнды: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Операция: Пропустить, если (f<b>) = 1

Измен. флаги: Нет

Код:

1010	bbba	ffff	ffff

Описание: Если бит ‘b’ в регистре ‘f’ = 1, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    BTFSS FLAG, 1, 0
FALSE   :
TRUE    :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
FLAG<1>=0 PC = адрес (FALSE)
FLAG<1>=1 PC = адрес (TRUE)

BTG Инверсия бита в f

Синтаксис: [*label*] BTG f,b[,a]

Операнды: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Операция: -(f<b>) → f<b>

Измен. флаги: Нет

Код:

0111	bbba	ffff	ffff

Описание: Инверсия бита ‘b’ в регистре ‘f’. Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример:

```
BTG     PORTC, 4, 0
```

До выполнения команды
PORTC = 0x75
После выполнения команды
PORTC = 0x65

BOV Переход, если переполнение (OV = 1)

Синтаксис: [*label*] BOV n

Операнды: $-128 \le n \le 127$

Операция: Если OV = 1
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0100	nnnn	nnnn

Описание: Если OV=1, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример:

```
HERE    BOV     Jump
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
Если OV=1 PC = адрес (Jump)
Если OV=0 PC = адрес (HERE + 2)

BZ Переход, если нуль (Z = 1)

Синтаксис: [*label*] BZ n

Операнды: $-128 \le n \le 127$

Операция: Если Z = 1
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1110	0000	nnnn	nnnn

Описание: Если Z=1, то происходит переход по адресу PC+2+2n (это действие выполняется за два такта). Если условие ложно, то выполняется следующая команда.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Если переход

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Если нет перехода

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’	Выполнение	Нет операции

Пример: HERE BZ Jump

До выполнения команды
PC = адрес (HERE)

После выполнения команды
Если Z = 1 PC = адрес (Jump)
Если Z = 0 PC = адрес (HERE + 2)

CALL Переход на подпрограмму

Синтаксис: [*label*] CALL k[,s]

Операнды: $0 \le k \le 1048575$
$s \in [0,1]$

Операция: (PC) + 4 → TOS,
k → PC <20:1>,
Если s = 1,
(W) → WS,
(STATUS) → STATUSS
(BSR) → BSRS

Измен. флаги: Нет

Код:1-е слово k<7:0>	1110	110s	kkkk	kkkk
2-е слово k<19:8>	1111	kkkk	kkkk	kkkk

Описание: Вызов подпрограммы во всем диапазоне адресуемой памяти (2 мегабайта). В начале, адрес возврата из подпрограммы (PC+4) сохраняется в стеке. Если s=1, то содержимое регистров W, STATUS, BSR сохраняются в спец. регистрах WS, STATUSS, BSRS. При s=0 обновление регистров не производится (по умолчанию). 20-разрядная константа загружается в PC<20:1>. Команда выполняется за 2 цикла.

Слов: 2

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы k<7:0>	Запись PC в стек	Чтение k<19:8>, Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    CALL    THERE, 1
```

До выполнения команды

PC = адрес (HERE)

После выполнения команды

PC = адрес (THERE)
TOS = адрес (HERE)
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF Очистка f

Синтаксис: [*label*] CLRF f[,a]

Операнды: $0 \le f \le 255$
$a \in [0,1]$

Операция: $000h \rightarrow f$
$1 \rightarrow Z$

Измен. флаги: Z

Код:

0110	101a	ffff	ffff

Описание: Очистка содержимого регистра ‘f’. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример:

```
CLRF    FLAG_REG, 1
До выполнения команды
        FLAG_REG = 0x5A
После выполнения команды
        FLAG_REG = 0x00
```

CLRWDT Сбросить сторожевой таймер

Синтаксис: [*label*] CLRWDT

Операнды: Нет

Операция: $000h \rightarrow$ WDT
$000h \rightarrow$ предделитель WDT
$1 \rightarrow$ -TO
$1 \rightarrow$ -PD

Измен. флаги: -TO, -PD

Код:

0000	0000	0000	0100

Описание: Сброс сторожевого таймера WDT и предделителя WDT. – TO = 1, -PD = 1.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Нет операции

Пример:

```
        CLRWDT
До выполнения команды
        Счетчик WDT = ?
После выполнения команды
        Счетчик WDT = 0x00
        Предделитель WDT = 0
        -TO = 1
        -PD = 1
```

COMF Инверсия f

Синтаксис: [*label*] COMF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: $(-f) \rightarrow dest$

Измен. флаги: N, Z

Код:

0001	11da	ffff	ffff

Описание: Инвертирование битов регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
COMF    REG, 0, 0
```

До выполнения команды

REG = 0x13

После выполнения команды

W = 0x13
REG = 0xEC

CPFSEQ Сравнить WREG и f, проп. если f = W

Синтаксис: [*label*] CPFSEQ f[,a]

Операнды: $0 \le f \le 255$
$a \in [0,1]$

Операция: (f) – (W)
Пропустить, если (f) = (W)
(незнаковое сравнение)

Измен. флаги: Нет

Код:

0110	001a	ffff	ffff

Описание: Сравнивается значение регистра ‘f’ с содержимым регистра W. Если ‘f’=W, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
REG = W PC = адрес (EQUAL)
REG ≠ W PC = адрес (NEQUAL)

CPFSGT Сравнить WREG и f, проп. если f > W

Синтаксис: [*label*] CPFSGT f[,a]

Операнды: $0 \le f \le 255$
$a \in [0,1]$

Операция: (f) – (W)
Пропустить, если (f) > (W)
(незнаковое сравнение)

Измен. флаги: Нет

Код:

0110	010a	ffff	ffff

Описание: Сравнивается значение регистра ‘f’ с содержимым регистра W. Если ‘f’ > W, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
REG > W PC = адрес (GREATER)
REG ≤ W PC = адрес (NGREATER)

CPFSLT Сравнить WREG и f, проп. если f < W

Синтаксис: [*label*] CPFSLT f[,a]

Операнды: $0 \le f \le 255$
$a \in [0,1]$

Операция: (f) – (W)
Пропустить, если (f) < (W)
(незнаковое сравнение)

Измен. флаги: Нет

Код:

0110	000a	ffff	ffff

Описание: Сравнивается значение регистра ‘f’ с содержимым регистра W. Если ‘f’ < W, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
REG < W PC = адрес (LESS)
REG ≥ W PC = адрес (NLESS)

DAW Десятичная коррекция WREG

Синтаксис: [*label*] DAW

Операнды: Нет

Операция: Если [W<3:0> > 9] или [DC = 1],
то (W<3:0>) + 6 → (W<3:0>);
иначе (W<3:0>) → (W<3:0>);

Если [W<7:4> > 9] или [C = 1],
то (W<7:4>) + 6 → (W<7:4>);
иначе (W<7:4>) → (W<7:4>);

Измен. флаги: C

Код:

0000	0000	0000	0111

Описание: Десятичная коррекция 8 бит регистра W, результата сложения двух переменных (в формате BCD) в корректный BCD формат.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра W	Выполнение	Запись в регистр W

Пример 1:

```
DAW
```

До выполнения команды

```
W = 0xA5
C = 0
DC = 0
```

После выполнения команды

```
W = 05
C = 1
DC = 0
```

Пример 2:

До выполнения команды

```
W = 0xCE
C = 0
DC = 0
```

После выполнения команды

```
W = 0x34
C = 1
DC = 0
```

DECF Декремент f

Синтаксис: [*label*] DECF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: (f) – 1 → dest

Измен. флаги: C, DC, N, OV, Z

Код:

0000	01da	ffff	ffff

Описание: Декремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
DECF    CNT, 1, 0
```

До выполнения команды

CNT = 0x01
Z = 0

После выполнения команды

CNT = 0x00
Z = 1

DECFSZ Декремент f, пропустить если 0

Синтаксис: [*label*] DECFSZ f[,d[,a]]

Операнды: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Операция: (f) – 1 → dest
Пропустить, если результат 0

Измен. флаги: Нет

Код:

0010	11da	ffff	ffff

Описание: Декремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если ‘f’ = 0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE        DECFSZ CNT, 1, 1
            GOTO LOOP
CONTINUE
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
CNT = CNT - 1
CNT = 0 PC = адрес (CONTINUE)
CNT ≠ 0 PC = адрес (HERE + 2)

DCFSNZ Декремент f, пропустить если не 0

Синтаксис: [*label*] DECFSNZ f[,d[,a]]

Операнды: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Операция: (f) – 1 → dest
Пропустить, если результат не 0

Измен. флаги: Нет

Код:

0100	11da	ffff	ffff

Описание: Декремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если ‘f’ ≠ 0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    DCFSNZ TEMP, 1, 0
ZERO    :
NZERO   :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
TEMP = TEMP - 1
TEMP = 0 PC = адрес (ZERO)
TEMP ≠ 0 PC = адрес (NZERO)

GOTO Переход по адресу

Синтаксис: [*label*] GOTO k

Операнды: $0 \le k \le 1048575$

Операция: $k \rightarrow PC <20:1>$

Измен. флаги: Нет

Код:1-е слово k<7:0>	1110	1111	kkkk	kkkk
2-е слово k<19:8>	1111	kkkk	kkkk	kkkk

Описание: Безусловный переход по любому адресу в пределах 2-х мегабайтного адресного пространства. 20-разрядное значение ‘k’ загружается в PC<20:1>. Команда выполняется за два цикла.

Слов: 2

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы k<7:0>	Нет операции	Чтение k<19:8>, Запись в PC
Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
GOTO    THERE
```

После выполнения команды

PC = адрес (THERE)

INCF Инкремент f

Синтаксис: [*label*] INCF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: $(f) + 1 \rightarrow dest$

Измен. флаги: C, DC, N, OV, Z

Код:	0010	10da	ffff	ffff

Описание: Инкремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
INCF    CNT, 1, 0
```

До выполнения команды

CNT = 0xFF
Z = 0
C = ?
DC = ?

После выполнения команды

CNT = 0x00
Z = 1
C = 1
DC = 1

INCFSZ Инкремент f, пропустить если 0

Синтаксис: [*label*] INCFSZ f[,d[,a]]

Операнды: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Операция: (f) + 1 → dest
Пропустить, если результат 0

Измен. флаги: Нет

Код:

0011	11da	ffff	ffff

Описание: Инкремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если ‘f’ = 0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    INCFSZ CNT, 1, 0
NZERO   :
ZERO    :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
CNT = CNT + 1
CNT = 0 PC = адрес (ZERO)
CNT ≠ 0 PC = адрес (NZERO)

INCFSNZ Инкремент f, пропустить если не 0

Синтаксис: [*label*] INCFSNZ f[,d[,a]]

Операнды: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Операция: (f) + 1 → dest
Пропустить, если результат не 0

Измен. флаги: Нет

Код:

0100	10da	ffff	ffff

Описание: Инкремент значения регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если ‘f’ ≠ 0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    INCFSZ REG, 1, 0
ZERO    :
NZERO   :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
REG = REG + 1
REG = 0 PC = адрес (ZERO)
REG ≠ 0 PC = адрес (NZERO)

IORLW Логическое ИЛИ константы и WREG

Синтаксис: [*label*] IORLW k

Операнды: $0 \le k \le 255$

Операция: (W) . OR . k $\rightarrow$ W

Измен. флаги: N, Z

Код:

0000	1011	kkkk	kkkk

Описание: Операция ИЛИ с содержимым регистра W и 8-разрядной константой ‘k’. Результат операции сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр W

Пример:

```
IORLW   0x35
```

До выполнения команды

W = 0x9A

После выполнения команды

W = 0xBF

IORWF Логическое ИЛИ WREG и f

Синтаксис: [*label*] IORWF f[,d[,a]]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: (W) . OR . (f) $\rightarrow$ dest

Измен. флаги: N, Z

Код:

0001	00da	ffff	ffff

Описание: Логическая операция поразрядного ИЛИ регистров W и ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
IORWF   RESULT, 0, 1
```

До выполнения команды

W = 0x91
RESULT = 0x13

После выполнения команды

W = 0x93
RESULT = 0x13

LFSR Поместить константу (12 бит) в FSR

Синтаксис: [*label*] FSR f, k

Операнды: $0 \le f \le 2$
$0 \le k \le 4095$

Операция: $k \rightarrow FSRf$

Измен. флаги: Нет

Код: слово 1 k<11:8>	1110	1110	00ff	kkkk
2-е слово k<7:0>	1111	0000	kkkk	kkkk

Описание: 12-разрядная константа ‘k’ загружается в регистр FSR (указатель косвенной адресации)

Слов: 2

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы k<11:8>	Нет операции	Запись k<11:8> в FSRfH <11:8>
Нет операции	Чтение константы k<7:0>	Нет операции	Запись k<7:0> в FSRfL <7:0>

Пример:

```
LFSR 2, 0x3AB
```

После выполнения команды

```
LFSRH = 0x03
LFSRL = 0xAB
```

MOVF Переместить f

Синтаксис: [*label*] MOVF f[,d[,a]]

Операнды: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Операция: $(f) \rightarrow dest$

Измен. флаги: N, Z

Код:	0101	00da	ffff	ffff

Описание: Содержимое регистра ‘f’ пересылается в зависимости от состояния бита d. Если d=0, то значение сохраняется в регистре W, если d=1, то значение сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
MOVF REG, 0, 0
```

До выполнения команды

```
W = 0xFF
REG = 0x22
```

После выполнения команды

```
W = 0x22
REG = 0x22
```

MOVFF Переместить из f в f

Синтаксис: [*label*] MOVFF fs, fd

Операнды: $0 \le fs \le 4095$
$0 \le fs \le 4095$

Операция: (fs) → fd

Измен. флаги: Нет

Код: источник fs	1100	ffff	ffff	ffff
приемник fd	1111	ffff	ffff	ffff

Описание: Содержимое регистра fs пересылается в регистр fd. Регистры fs и fd могут находиться в любом месте адресного пространства размером в 4096 байт (000h-FFFh). В качестве fs и fd может использоваться W. Команда MOVFF может применяться для пересылки данных в периферийные устройства, такие, как буфер передатчика, порт ввода/вывода и др. В качестве fs в команде MOVFF нельзя использовать регистры: PCL, TOSU, TOSH и TOSL.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘fs’	Выполнение	Нет операции
Нет операции	Нет операции	Нет операции	Запись в регистр ‘fd’

Пример:

```
MOVFF   REG1, REG2
До выполнения команды
        REG1= 0x33
        REG2 = 0x11
После выполнения команды
        REG1= 0x33
        REG2 = 0x33
```

MOVLB Поместить константу в BSR<3:0>

Синтаксис: [*label*] MOVLB k

Операнды: $0 \le k \le 255$

Операция: k → BSR

Измен. флаги: Нет

Код:	0000	0001	kkkk	kkkk

Описание: 8-разрядная константа ‘k’ загружается в регистр BSR (регистр выбора банка памяти).

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр BSR

Пример:

```
MOVLB   5
До выполнения команды
        BSR = 0x02
После выполнения команды
        BSR = 0x05
```

MOVLW Поместить константу в WREG

Синтаксис: [*label*] MOVLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow W$

Измен. флаги: Нет

Код:

0000	1110	kkkk	kkkk

Описание: 8-разрядная константа ‘k’ загружается в регистр W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр W

Пример: MOVLW 0x5A

После выполнения команды

W = 0x5A

MOVWF Переместить WREG в f

Синтаксис: [*label*] MOVWF f[,a]

Операнды: $0 \le f \le 255$

$a \in [0,1]$

Операция: $(W) \rightarrow f$

Измен. флаги: Нет

Код:

0110	111a	ffff	ffff

Описание: Пересылка содержимого регистра W в регистр ‘f’. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра W	Выполнение	Запись в регистр ‘f’

Пример: MOVWF REG, 0

До выполнения команды

W = 0x4F

REG = 0xFF

После выполнения команды

W = 0x4F

REG = 0x4F

MULLW Умножение константы на WREG

Синтаксис: [*label*] MULLW k

Операнды: $0 \le k \le 255$

Операция: (W) x k $\rightarrow$ PRODH:PRODL

Измен. флаги: Нет

Код:

0000	1101	kkkk	kkkk

Описание: Умножение содержимого регистра W и 8-разрядной константы. 16-разрядный результат помещается в регистровую пару PRODH:PRODL (регистр PRODH содержит старший байт). Выполнение команды не изменяет содержимого регистра W и не влияет на флаги АЛУ. Нулевой результат возможен, но он не детектируется.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в PRODH: PRODL

Пример:

```
MULLW   0xC4
До выполнения команды
        W = 0xE2
        PRODH = ?
        PRODL = ?
После выполнения команды
        W = 0xE2
        PRODH = 0xAD
        PRODL = 0x08
```

MULWF Умножение WREG и f

Синтаксис: [*label*] MULWF f[,a]

Операнды: $0 \le f \le 255$

$a \in [0,1]$

Операция: (W) x (f) $\rightarrow$ PRODH:PRODL

Измен. флаги: Нет

Код:

0000	001a	ffff	ffff

Описание: Умножение содержимого регистров W и ‘f’. 16-разрядный результат помещается в регистровую пару PRODH:PRODL (регистр PRODH содержит старший байт). Выполнение команды не изменяет содержимого регистров W, ‘f’ и не влияет на флаги АЛУ. Нулевой результат возможен, но он не детектируется. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в PRODH: PRODL

Пример:

```
MULWF   REG, 1
```

До выполнения команды

```
W = 0xC4
REG = 0xB5
PRODH = ?
PRODL = ?
```

После выполнения команды

```
W = 0xC4
REG = 0xB5
PRODH = 0x8A
PRODL = 0x94
```

NEGF Негативное значение f

Синтаксис: [*label*] NEGF f[,a]

Операнды: $0 \le f \le 255$

$a \in [0,1]$

Операция: $(-f) + 1 \rightarrow f$

Измен. флаги: N, OV, C, DC, Z

Код:

0110	110a	ffff	ffff

Описание: Негативное значение ‘f’ в формате дополнения до 2. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример: NEGF REG, 1

До выполнения команды

REG = 0x3A

После выполнения команды

REG = 0xC6

NOP Нет операции

Синтаксис: [*label*] NOP

Операнды: Нет

Операция: Нет операции

Измен. флаги: Нет

Код:

0000	0000	0000	0000
1111	xxxx	xxxx	xxxx

Описание: Нет операции.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Нет операции

Пример: Нет

POP Чтение вершины стека возврата TOS

Синтаксис: [*label*] POP

Операнды: Нет

Операция: (TOS) →

Измен. флаги: Нет

Код:

0000	0000	0000	0110

Описание: Перемещение всего содержимого стека на один уровень вверх. Предыдущее значение, находящееся на вершине стека, утрачивается. Данная команда может использоваться для программного управления стеком.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	POP TOS	Нет операции

Пример:

```
POP
GOTO    NEW
```

До выполнения команды

TOS = 0031A2h
Стек (на 1 уровень ниже) = 014332h

После выполнения команды

TOS = 014332h
PC = NEW

PUSH Запись в вершину стека возврата TOS

Синтаксис: [*label*] PUSH

Операнды: Нет

Операция: (PC+2) → TOS

Измен. флаги: Нет

Код:

0000	0000	0000	0101

Описание: Данная команда помещает в вершину стека (TOS) значение PC+2. Предыдущее значение, находящееся на вершине стека перемещается на один уровень вниз.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	PUSH PC+2	Нет операции	Нет операции

Пример:

```
PUSH
```

До выполнения команды

TOS = 00345Ah
PC = 000124h

После выполнения команды

PC = 000126h
TOS = 000126h
Стек (на 1 уровень ниже) = 00345Ah

RCALL Короткий переход на подпрограмму

Синтаксис: [*label*] RCALL n

Операнды: $-1024 \le n \le 1023$

Операция: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Измен. флаги: Нет

Код:

1101	1nnn	nnnn	nnnn

Описание: Производится вызов подпрограммы, находящийся в пределах 1кб от текущей позиции. В стек помещается адрес возврата (PC+2). После этого в программный счетчик загружается новый адрес PC+2+2n. Команда выполняется за два цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘n’ PUSH PC	Выполнение	Запись в регистр PC
Нет операции	Нет операции	Нет операции	Нет операции

Пример:
```
HERE    RCALL   Jump
```
До выполнения команды
PC = адрес (HERE)
После выполнения команды
PC = адрес (Jump)
TOS = адрес (HERE-2)

RESET Программный сброс

Синтаксис: [*label*] RESET

Операнды: Нет

Операция: Сброс всех регистров и флагов аналогично MCLR

Измен. флаги: Все

Код:

0000	0000	1111	1111

Описание: Команда делает возможным сброс микроконтроллера, аналогичный аппаратному сбросу MCLR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Старт программного сброса	Нет операции	Нет операции

Пример:
```
RESET
```
После выполнения команды
Регистры = значение после сброса
ОЗУ = значение после сброса

RETFIE Возврат из пп с разреш. прерываний

Синтаксис: [*label*] RETFIE [s]

Операнды: s ∈ [0, 1]

Операция: (TOS) → PC,
1 → GIE/GIEH или PEIE/GIEL,
Если s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH не изменяются

Измен. флаги: GIE/GIEH, PEIE/GIEL

Код:

0000	0000	0001	000s

Описание: Возврат из прерывания. Значение, находящееся на вершине стека (TOS), загружается в PC. Прерывания включаются установкой бита глобального разрешения прерываний высокого/низкого приоритета. Если s=1, содержимое скрытых регистров WS, STATUSS и BSRS загружаются в соответствующие регистры W, STATUS и BSR. Если s=0, то загрузки не производится (по умолчанию).

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	POP PC GIEH =1 или GIEL = 1
Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
RETFIE  1
```

После выполнения команды

PC = TOS
W = WS
BSR = BSRS
STATUS = STATUSS
GIE/GIEH, PEIE/GIEL = 1

RETLW Возврат из пп с загрузкой WREG

Синтаксис: [*label*] RETLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow W$,
(TOS) $\rightarrow$ PC,
PCLATU, PCLATH are unchanged

Измен. флаги: Нет

Код:

0000	1100	kkkk	kkkk

Описание: В регистр W записывается 8-разрядная константа ‘k’. Программный счетчик загружается значением, взятым с вершины стека (адресом возврата). Защелка старшего байта адреса (PCLATH) при этом не меняет своего содержимого.

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	POP PC, запись в W
Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
CALL    TABLE   ; W = смещение таблицы
                ; после выполнения
                ; W = значение из таблицы
:
TABLE
        ADDWF   PCL
        RETLW k0
        RETLW k1
        :
        :
        RETLW kn
```

До выполнения команды

W = 0x07

После выполнения команды

W = значение kn

RETURN Возврат из подпрограммы

Синтаксис: [*label*] RETURN [s]

Операнды: s ∈ [0, 1]

Операция: (TOS) → PC,
Если s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH не изменяются

Измен. флаги: Нет

Код:

0000	0000	0001	001s

Описание: Возврат из подпрограммы. Значение, находящееся на вершине стека (TOS), загружается в PC. Если s=1, содержимое скрытых регистров WS, STATUSS и BSRS загружаются в соответствующие регистры W, STATUS и BSR. Если s=0, то загрузки не производится (по умолчанию).

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	POP PC
Нет операции	Нет операции	Нет операции	Нет операции

Пример: RETURN 1

После выполнения команды
PC = TOS

RLCF Сдвиг влево через перенос

Синтаксис: [*label*] RLCF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f<n>) → dest<n+1>,

(f<7>) → C,

(C) → dest <0>

Измен. флаги: C, N, Z

Код:

0011	01da	ffff	ffff

Описание: Содержимое регистра ‘f’ циклически сдвигается на один бит влево через флаг переноса. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись значения

Пример:

```
RLCF    REG, 0, 0
До выполнения команды
        REG = 1110 0110
        C = 0
После выполнения команды
        REG = 1110 0110
        W = 1100 1100
        C = 1
```

RLNCF Сдвиг влево без переноса

Синтаксис: [*label*] RLNCF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f<n>) → dest<n+1>,

(f<7>) → dest <0>

Измен. флаги: N, Z

Код:

0100	01da	ffff	ffff

Описание: Содержимое регистра ‘f’ циклически сдвигается на один бит влево. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись значения

Пример: RLNCF REG, 1, 0

До выполнения команды

REG = 1010 1011

После выполнения команды

REG = 0101 0111

RRCF Сдвиг вправо через перенос

Синтаксис: [*label*] RRCF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f<n>) → dest<n-1>,

(f<0>) → C,

(C) → dest <7>

Измен. флаги: C, N, Z

Код:

0011	00da	ffff	ffff

Описание: Содержимое регистра ‘f’ циклически сдвигается на один бит вправо через флаг переноса. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись значения

Пример:

```
RRCF    REG, 0, 0
До выполнения команды
        REG = 1110 0110
        C = 0
После выполнения команды
        REG = 1110 0110
        W = 0111 0011
        C = 0
```

RRNCF Сдвиг вправо без переноса

Синтаксис: [*label*] RRNCF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f<n>) → dest<n-1>,

(f<0>) → dest <7>

Измен. флаги: N, Z

Код:

0100	00da	ffff	ffff

Описание: Содержимое регистра ‘f’ циклически сдвигается на один бит вправо. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись значения

Пример 1: RRNCF REG, 1, 0

До выполнения команды

REG = 1101 0111

После выполнения команды

REG = 1110 1101

Пример 2: RRNCF REG, 0, 0

До выполнения команды

W = ?

REG = 1101 0111

После выполнения команды

W = 1110 1011

REG = 1101 0111

SETF Установить все биты f

Синтаксис: [*label*] SETF f [,a]

Операнды: $0 \le f \le 255$

$a \in [0, 1]$

Операция: FFh $\rightarrow$ f

Измен. флаги: Нет

Код:

0110	100a	ffff	ffff

Описание: Записать в регистр значение 0xFF. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись в регистр ‘f’

Пример:

```
SETF    REG,    1,
```

До выполнения команды

REG = 0x5A

После выполнения команды

REG = 0xFF

SLEEP Переход в SLEEP режим

Синтаксис: [*label*] SEEP

Операнды: Нет

Операция: 00h $\rightarrow$ WDT,

0 $\rightarrow$ WDT,

1 $\rightarrow$ - TO,

0 $\rightarrow$ - PD

Измен. флаги: - TO, - PD

Код:

0000	0000	0000	0011

Описание: Биты -PD=0, -TO=1. Сброс сторожевого таймера WDT. Переход в режим SLEEP с остановкой тактового генератора.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	

Пример:

```
SLEEP
```

До выполнения команды

- TO = ?

- PD = ?

После выполнения команды

- TO = 1 *

- PD = 0

SUBFWB Вычитание f из WREG с заемом

Синтаксис: [*label*] SUBFWB f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (W) – (f) – (-C) → dest

Измен. флаги: N, OV, C, DC, Z

Код:

0101	01da	ffff	ffff

Описание: Из регистра W вычитается значение регистра ‘f’ вместе с флагом переноса. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример 1:

```
SUBFWB  REG, 1, 0
До выполнения команды
        REG = 3
        W = 2
        C = 1
После выполнения команды
        REG = FF
        W = 2
        C = 0
        Z = 0
        N = 1
```

Пример 2:

```
SUBFWB  REG, 0, 0
До выполнения команды
        REG = 2
        W = 5
        C = 1
После выполнения команды
        REG = 2
        W = 3
        C = 1
        Z = 0
        N = 0
```

Пример 3:

```
SUBFWB  REG, 1, 0
До выполнения команды
        REG = 1
        W = 2
        C = 0
После выполнения команды
        REG = 0
        W = 2
        C = 1
        Z = 1
        N = 0
```

SUBLW Вычитание WREG из константы

Синтаксис: [*label*] SUBLW k

Операнды: $0 \le k \le 255$

Операция: $k - (W) \rightarrow W$

Измен. флаги: N, OV, C, DC, Z

Код:

0000	1000	kkkk	kkkk

Описание: Содержимое W вычитается из 8-разрядной константы ‘k’. Результат помещается в регистр W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в W

Пример 1:
SUBLW 0x02
До выполнения команды
W = 1
C = ?
После выполнения команды
W = 1
C = 1
Z = 0
N = 0

Пример 2:
SUBLW 0x02
До выполнения команды
W = 2
C = ?
После выполнения команды
W = 0
C = 1
Z = 1
N = 0

Пример 3:
SUBLW 0x02
До выполнения команды
W = 3
C = ?
После выполнения команды
W = FF
C = 0
Z = 0
N = 1

SUBWF Вычитание WREG из f

Синтаксис: [*label*] SUBWF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f) - (W) → dest,

Измен. флаги: N, OV, C, DC, Z

Код:

0101	11da	ffff	ffff

Описание: Содержимое регистра W вычитается из регистра ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример 1: SUBWF REG, 1, 0

До выполнения команды

REG = 3

W = 2

C = ?

После выполнения команды

REG = 1

W = 2

C = 1

Z = 0

N = 0

Пример 2: SUBWF REG, 0, 0

До выполнения команды

REG = 2

W = 2

C = ?

После выполнения команды

REG = 2

W = 0

C = 1

Z = 1

N = 0

Пример 3: SUBWF REG, 1, 0

До выполнения команды

REG = 1

W = 2

C = ?

После выполнения команды

REG = FFh

W = 2

C = 0

Z = 0

N = 1

SUBWFB Вычитание WREG из f с заемом

Синтаксис: [*label*] SUBWFB f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f) - (W) – (-C) → dest,

Измен. флаги: N, OV, C, DC, Z

Код:

0101	10da	ffff	ffff

Описание: Из регистра ‘f’ вычитается значение регистра W вместе с флагом переноса. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример 1: SUBWFB REG, 1, 0

До выполнения команды

REG = 0x19 (0001 1001)

W = 0x0D (0000 1101)

C = 1

После выполнения команды

REG = 0x0C (0000 1011)

W = 0x0D (0000 1101)

C = 1

Z = 0

N = 0

Пример 2: SUBWFB REG, 0, 0

До выполнения команды

REG = 0x1B (0001 1011)

W = 0x1A (0001 1010)

C = 0

После выполнения команды

REG = 0x1B (0001 1011)

W = 0x00

C = 1

Z = 1

N = 0

Пример 3: SUBWFB REG, 1, 0

До выполнения команды

REG = 0x03 (0000 0011)

W = 0x0E (0000 1101)

C = 1

После выполнения команды

REG = 0xF5

W = 0x0E

C = 0

Z = 0

N = 1

SWAPF Поменять местами полубайты в f

Синтаксис: [*label*] SWAPF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0, 1]$

$a \in [0, 1]$

Операция: (f<3:0>) → dest<7:4>,

(f<7:4>) → dest <3:0>

Измен. флаги: Нет

Код:

0011	10da	ffff	ffff

Описание: Старший и младший полубайт значения регистра ‘f’ меняются местами. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример: SWAPF REG, 1, 0

До выполнения команды

REG = 0x53

После выполнения команды

REG = 0x35

TBLRD Табличное чтение

Синтаксис: [*label*] TBLRD (*; *+; *-; +*)

Операнды: Нет

Операция: Если TBLRD*,
(Память программ (TBLPTR)) → TABLAT;
TBLPTR – не изменяется;
Если TBLRD*+,
(Память программ (TBLPTR)) → TABLAT;
TBLPTR +1 → TBLPTR;
Если TBLRD*-,
(Память программ (TBLPTR)) → TABLAT;
TBLPTR –1 → TBLPTR;
Если TBLRD+*,
(TBLPTR) +1 → TBLPTR;
(Память программ (TBLPTR)) → TABLAT;

Измен. флаги: Нет

Код:

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

Описание: Инструкция используется для чтения содержимого памяти программ. Для доступа к памяти используется указатель TBLPTR. Данный (21-разрядный) указатель позволяет адресовать любой байт адресного пространства размером 2Мбайта.
TBLPTR[0]=0: Младший байт слова, находящегося в памяти программы.
TBLPTR[0]=1: Старший байт слова, находящегося в памяти программы.
Команда TBLRD может изменять значение указателя TBLPTR следующим образом:
- не изменять
- увеличение на 1 после выполнения команды
- уменьшение на 1 после выполнения команды
- увеличение на 1 перед выполнением команды

Слов: 1

Циклов: 2

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Нет операции
Нет операции	Нет операции (Чтение программы памяти)	Нет операции	Нет операции (запись TABLAT)

Пример 1: TBLRD *+ ;

До выполнения команды
TABLAT = 0x55
TBLPTR = 0x00A356
MEMORY(0x00A356) = 0x34

После выполнения команды
TABLAT = 0x34
TBLPTR = 0x00A357

Пример 2: TBLRD +* ;

До выполнения команды
TABLAT = 0xAA
TBLPTR = 0x01A357
MEMORY(0x00A357) = 0x12
MEMORY(0x00A358) = 0x34

После выполнения команды
TABLAT = 0x34
TBLPTR = 0x01A358

TBLWT Табличная запись

Синтаксис: [*label*] TBLWT (*; *+; *-; +*)

Операнды: Нет

Операция: Если TBLWT *,
(TABLAT)→ Память программ (TBLPTR) или рег. защелки;
Если TBLWT*+,
(TABLAT)→ Память программ (TBLPTR) или рег. защелки;
TBLPTR +1 → TBLPTR;
Если TBLWT*-,
(TABLAT)→ Память программ (TBLPTR) или рег. защелки;
TBLPTR –1 → TBLPTR;
Если TBLWT+*,
(TBLPTR) +1 → TBLPTR;
(TABLAT)→ Память программ (TBLPTR) или рег. защелки;

Измен. флаги: Нет

Код:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Описание: Инструкция используется для записи в память программ. Для доступа к памяти используется указатель TBLPTR. Данный (21-разрядный) указатель позволяет адресовать любой байт адресного пространства размером 2Мбайта.
TBLPTR[0]=0: Младший байт слова, находящегося в памяти программы.
TBLPTR[0]=1: Старший байт слова, находящегося в памяти программы.
Команда TBLRD может изменять значение указателя TBLPTR следующим образом:
- не изменять
- увеличение на 1 после выполнения команды
- уменьшение на 1 после выполнения команды
- увеличение на 1 перед выполнением команды

Слов: 1

Циклов: 2 (больше, если длинная запись в память)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Нет операции
Нет операции	Нет операции (Чтение TABLAT)	Нет операции	Нет операции (зап. в рег.защ. или память)

Пример 1: TBLWT *+ ;

До выполнения команды
TABLAT = 0x55
TBLPTR = 0x00A356
MEMORY или рег.защ.(0x00A356) = 0xFF

После выполнения команды
TABLAT = 0x55
TBLPTR = 0x00A357
MEMORY или рег.защ.(0x00A356) = 0x55

Пример 2: TBLWT +* ;

До выполнения команды
TABLAT = 0x34
TBLPTR = 0x01389A
MEMORY или рег.защ.(0x01389A) = 0xFF
MEMORY или рег.защ.(0x01389B) = 0xFF

После выполнения команды
TABLAT = 0x34
TBLPTR = 0x01389B
MEMORY или рег.защ.(0x01389A) = 0xFF
MEMORY или рег.защ.(0x01389B) = 0x34

TSTFSZ Тест f, пропустить если 0

Синтаксис: [*label*] TSTFSZ f [,a]

Операнды: $0 \le f \le 255$
$a \in [0,1]$

Операция: Пропустить, если f = 0

Измен. флаги: Нет

Код:

0110	011a	ffff	ffff

Описание: Если ‘f’=0, вместо следующей команды выполняется пустая операция (NOP), растягивая выполнение команды на 2 цикла. Если a = 0, выбран банк быстрого доступа (значение BSR игнорируется). Если a = 1, используется регистр BSR для выбора банка памяти данных.

Слов: 1

Циклов: 1(2) 3 цикла, если пропуск двухсловной команды

Выполнение команды по тактам

	Q1	Q2	Q3	Q4
	Декодирование команды	Чтение регистра ‘f’	Выполнение	Нет операции
Если пропуск	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
Если пропуск 2-х словной команды	Q1	Q2	Q3	Q4
	Нет операции	Нет операции	Нет операции	Нет операции
	Нет операции	Нет операции	Нет операции	Нет операции

Пример:

```
HERE    TSTFSZ CNT, 1
NZERO   :
ZERO    :
```

До выполнения команды
PC = адрес (HERE)
После выполнения команды
Если CNT = 0x00, PC = адрес (NZERO)
Если CNT ≠ 0x00, PC = адрес (ZERO)

XORLW Лог. исключ. ИЛИ константы и WREG

Синтаксис: [*label*] XORLW k

Операнды: $0 \le k \le 255$

Операция: (W) . XOR . k → W

Измен. флаги: N, Z

Код:

0000	1010	kkkk	kkkk

Описание: Операция исключающего ИЛИ с содержимым регистра W и 8-разрядной константой ‘k’. Результат операции сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы ‘k’	Выполнение	Запись в регистр W

Пример:

```
XORLW   0xAF
```

До выполнения команды

W = 0xB5

После выполнения команды

W = 0x1A

XORWF Логическое исключающее ИЛИ WREG и f

Синтаксис: [*label*] XORWF f [,d [,a]

Операнды: $0 \le f \le 255$

$d \in [0,1]$

$a \in [0,1]$

Операция: (W) . XOR . (f) → dest

Измен. флаги: N, Z

Код:

0001	10da	ffff	ffff

Описание: Логическая операция поразрядного исключающего ИЛИ регистров W и ‘f’. Если d=0, то результат сохраняется в регистре W, если d=1, то результат сохраняется в регистре ‘f’ (по умолчанию). Если a = 0, выбран банк быстрого доступа. Если a = 1, используется BSR.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра ‘f’	Выполнение	Запись результата

Пример:

```
XORWF   REG, 0, 1
```

До выполнения команды

REG = 0xAF
W = 0xB5

После выполнения команды

REG = 0x1A
W = 0xB5

21. Поддержка разработчиков

Микроконтроллеры PICmicro обеспечены большим спектром аппаратных и программных инструментальных средств проектирования:

- Интегрированная среда проектирования:
  - Программное обеспечение MPLAB-IDE
- Ассемблер/Компилятор/Линкер:
  - Ассемблер MPASM
  - Компиляторы MLAB-C17 и MPLAB-C18
  - Линкер MPLINK/ Организатор библиотек MPLIB
- Симулятор:
  - Программный симулятор MLAB-SIM
- Эмуляторы:
  - Внутрисхемный эмулятор реального времени MPLAB-ICE2000
  - ICEPIC
- Внутрисхемный отладчик:
  - MLAB-ICD
- Программаторы:
  - Универсальный программатор PRO MATE II
  - Недорогой программатор PICSTART для начала работы с PICmicro
- Недорогие демонстрационные платы:
  - PICDEM-1
  - PICDEM-2
  - PICDEM-3;
  - PICDEM-17
  - KeeLoq

21.1 Интегрированная среда проектирования MPLAB-IDE

Программное обеспечение MPLAB-IDE предназначено для разработки программного обеспечения 8-разрядных микроконтроллеров PICmicro, работающее под управлением операционной системы Windows.

Основные характеристики MPLAB-IDE:

- Многофункциональные возможности:
  - Редактор
  - Симулятор
  - Программатор (приобретается отдельно)
  - Эмулятор (приобретается отдельно)
- Полнофункциональный редактор
- Организатор проекта
- Настройка панелей инструментов и параметров отображения
- Строка состояния
- Интерактивная помощь

MPLAB-IDE позволяет Вам:

- Редактировать исходные файлы написанные на языке ассемблера или C
- Быстро выполнять трансляцию и компиляцию проекта автоматически загружая параметры используемого микроконтроллера PICmicro
- Выполнять отладку программы с использованием:
  - Исходных файлов
  - Листинга программы
  - Объектного кода

Однотипная работа инструментальных модулей интегрированной среды проектирования MPLAB-IDE позволяет легко перейти от программного симулятора MPLAB-SIM к использованию полнофункционального эмулятора.

21.2 Ассемблер MPASM

MPASM - полнофункциональный универсальный макроассемблер для всех семейств микроконтроллеров PICmicro. Ассемблер может генерировать шестнадцатиразрядный файл пригодный для записи в микроконтроллер или формировать перемещаемые объектные файлы для линкера MPLINK.

MPASM имеет интерфейс командной строки и оконный интерфейс, работает под управлением операционной системы Windows 3.X и выше. Может работать как автономное приложение. MPASM генерирует объектные файлы, шестнадцатеричные HEX файлы в стандарте Intel, файл карты памяти (для детализации использования памяти микроконтроллера), файл листинга программы (текст программы совмещен с кодами микроконтроллера) и файл отладки для MPLAB-IDE.

Особенности MPASM:

- MPASM и MPLINK интегрированы в MPLAB-IDE
- MPASM поддерживает систему макрокоманд, упрощающих написание текста программы
- Позволяет выполнять компиляцию условных блоков текста программы

Директивы MPASM дают возможность управлять компиляцией исходного текста программы.

21.3 С компиляторы MPLAB-C17 и MPLAB-C18

MPLAB-C17 и MPLAB-C18 - полнофункциональные ANSI 'C' компиляторы с интегрированной средой разработки для микроконтроллеров семейств PIC17CXXX и PIC18CXXX соответственно. Для упрощения отладки текста программы компиляторы обеспечивают интеграцию в средства проектирования с передачей информации об используемых переменных в формате совместимом с MPLAB-IDE.

21.4 Линкер MPLINK, организатор библиотек MPLIB

MPLINK - линкер перемещаемых объектных файлов, сгенерированных программами MPASM, MPLAB-C17 и MPLAB-C18. Линкер выполняет связь объектных файлов с предварительно компилированными файлами библиотек и файлами сценария.

MPLIB - организатор библиотек предварительно откомпилированных исходных файлов, которые нужно использовать с MPLINK. Когда подпрограмма библиотечного файла вызывается из исходного файла, в приложение будет включена только необходимый модуль. Это позволяет эффективно использовать большие библиотеки в различных приложениях. MPLIB управляет созданием и изменением библиотечных файлов.

Особенности MPLINK:

- MPLINK работает совместно с MPASM, MPLAB-C17 и MPLAB-C18
- MPLINK позволяет разбивать память микроконтроллера на разделы

Особенности MPLIB:

- MPLIB упрощает подключение дополнительных файлов потому, что позволяет подключить одну библиотеку вместо множества мелких файлов
- MPLIB группирует связанные модули
- MPLIB позволяет добавлять, изменять, удалять и заменять модули в библиотечных файлах.

21.5 Программный симулятор MPLAB-SIM

Симулятор MPLAB-SIM позволяет проследить выполнение программы микроконтроллеров PICmicro на уровне команд по шагам или в режиме анимации. На любой команде выполнение программы может быть остановлено для проверки и изменения памяти. Функции стимула позволяют моделировать сигнал с логическими уровнями на входах микроконтроллера. MPLAB-SIM полностью поддерживает символьную отладку, используя MPLAB-C17, MPLAB-C18 и MPASM. MPLAB-SIM является доступным и удобным средством отладки программ для микроконтроллеров PICmicro.

21.6 Универсальный эмулятор MPLAB-ICE

Универсальный эмулятор MPLAB-CE обеспечивает разработчиков полным набором инструментальных средств проектирования устройств с применением микроконтроллеров PICmicro. Управление работой эмулятора выполняется из интегрированной среды проектирования MPLAB-IDE с возможностью редактирования, компиляции, загрузки и выполнения программы.

Заменяемые поды позволяют быстро перенастроить эмулятор для работы с другим типом микроконтроллеров. Универсальная архитектура MPLAB-ICE дает возможность поддерживать новые типы микроконтроллеров PICmicro.

Эмулятор MPLAB-ICE был разработан как система эмуляции (анимации) в реальном масштабе времени с дополнительными возможностями, присутствующих в дорогих инструментальных средствах. Эмулятор работает под управлением распространенной операционной системы Microsoft Windows 3.x/95/98.

MPLAB-ICE 2000 - полнофункциональная система эмуляции с усовершенствованными функциями трассировки, триггеров и управляющих особенностей. Оба эмулятора используют одинаковые поды и работают во всех допустимых режимах микроконтроллеров PICmicro.

21.7 Внутрисхемный эмулятор ICEPIC

ICEPIC - недорогой эмулятор, предназначенный для однократно программируемых (OTP) 8-разрядных микроконтроллеров семейств PIC16C5X, PIC16C6X, PIC16C7X и PIC16CXXX. Модульная структура позволяет поддерживать все типы микроконтроллеров семейства PIC16C5X и PIC16CXXX за счет сменных подов.

21.8 Внутрисхемный отладчик MPLAB-ICD

Внутрисхемный отладчик MPLAB-ICD является мощным недорогим инструментом отладки программы. Работа MPLAB-ICD основана на функции внутрисхемной отладки Flash микроконтроллеров семейства PIC16F87X. Эта особенность, совместно с функцией внутрисхемного последовательного программирования, позволяет запрограммировать микроконтроллер непосредственно из среды проектирования MPLAB-IDE. MPLAB-ICD дает возможность быстро выполнить отладку программы, выполняя ее по шагам или в режиме реального времени.

21.9 Универсальный программатор PRO MATE II

Универсальный программатор PRO MATE II может работать автономно и под управлением PC совместимого компьютера. Для максимальной надежности программирования в программаторе PRO MATE II можно указать напряжения V_{DD} и V_{PP}. В программатор встроен ЖКИ дисплей для вывода сообщений об ошибках и клавиатура для ввода команд. Модульная колодка позволяет программировать микросхемы в различных корпусах. В автономном режиме программатор PRO MATE II может проверять микроконтроллер и устанавливать биты защиты.

21.10 Программатор PICSTART Plus

Недорогой программатор PICSTART Plus предназначен для начала работы с микроконтроллерами PICmicro, подключается к PC совместимому компьютеру через COM (RS-232) порт и работает под управлением интегрированной среды проектирования MPLAB-IDE. PICSTART Plus поддерживает все микроконтроллеры PICmicro в корпусах до 40 выводов. Микроконтроллеры с большим числом выводов (PIC16C92X, PIC17C76X) поддерживаются при использовании адаптеров.

21.11 Демонстрационная плата PICDEM-1

Демонстрационная плата PICDEM-1 предназначена для микроконтроллеров PIC16C5X (PIC26C54, PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 и PIC17C44. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART Plus. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, потенциометр для моделирования аналогового входа, выключатели и восемь светодиодов подключенных к PORTB.

21.12 Демонстрационная плата PICDEM-2

Демонстрационная плата PICDEM-2 предназначена для микроконтроллеров PIC16C62, PIC16C64, PIC16C65, PIC16C73 и PIC16C74. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART Plus. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, потенциометр для моделирования аналогового входа, последовательная EEPROM память для демонстрации работы шины I^2C, выводы для подключения ЖКИ и дополнительной клавиатуры.

21.13 Демонстрационная плата PICDEM-3

Демонстрационная плата PICDEM-3 предназначена для микроконтроллеров PIC16C923 и PIC16C924 выполненных в 44-выводном PLCC корпусе с интегрированным ЖКИ модулем. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART Plus. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, выключатели; потенциометр для моделирования аналогового входа; термистор; выводы для подключения ЖКИ и дополнительной клавиатуры; 12-разрядный ЖКИ для отображения времени, даты и температуры; дополнительный интерфейс RS-232; программное обеспечение работающее под управлением операционной системы Windows 3.x для передачи данных на PC совместимый компьютер.

21.14 Демонстрационная плата PICDEM-17

Демонстрационная плата PICDEM-17 предназначена для микроконтроллеров PIC17C752, PIC17C756, PIC17C762 и PIC17C766. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART Plus. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя.

21.15 KeeLoq (с функциями программатора)

Оценочная система KeeLoq предназначена для микросхем HCS фирмы Microchip. В состав комплекта входит: ЖКИ дисплей для отображения изменяющихся кодов, декодер, интерфейс программирования.

Таблица 21-1 Инструментальные средства проектирования от компании Microchip

		PIC12CXXX	PIC14000	PIC16C5X	PIC16C6X	PIC16CXXX	PIC16F62X	PIC16C7X	PIC16C7XX	PIC16C8X	PIC16F8XX	PIC16C9XX	PIC17C4X	PIC17C7XX	PIC18CXX2	24CXX/ 25CXX/ 93CXX	HCSXXX	MCRFXXX	MCP2510
Программное обеспечение	MPLAB-IDE	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√			
	MPLAB-C17												√	√					
	MPLAB-C18														√				
	MPASM/MPLINK	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√		
Эмуляторы	MPLAB-ICE	√	√	√	√	√	√**	√	√	√	√	√	√	√	√	√		√	
	PICMASTER	√	√	√	√	√		√	√	√		√	√	√				√	
	ICEPIC	√		√	√	√		√	√	√		√						√	
Отладчик	MPLAB-ICD				√*			√*			√							√	
Программа-торы	PICSTART	√	√	√	√	√	√**	√	√	√	√	√	√	√	√	√		√	
	PRO MATE II	√	√	√	√	√	√**	√	√	√	√	√	√	√	√	√		√	
Демонстрационные и отладочные платы	SIMICE	√		√															
	PICDEM-1			√		√		√+		√			√						
	PICDEM-2				√+			√+							√				
	PICDEM-3											√							
	PICDEM-14A		√																
	PICDEM-17													√					
	KeeLoq															√			
	KeeLoq транспондеры																√		
	Программатор microID																	√	
	DEMO microID 125кГц																	√	
	Проверка коллизий microID 125кГц																	√	
	Проверка коллизий microID 13.56МГц																	√	
	DEMO MCP2510 CAN																		√

* Обратитесь в представительства компании Microchip для получения дополнительной информации по использованию MPLAB-ICD с PIC16C62, 63, 64, 65, 72, 73, 74, 76, 77

** Обратитесь в представительства компании Microchip для уточнения времени доступности к заказу.

\+ Инструментальное средство доступно для некоторых устройств.

22. Электрические характеристики

Максимально допустимые значения (*)

Предельная рабочая температура ..... от -55°C до +125°C
Температура хранения ..... от -65°C до +150°C
Напряжение V_{DD} относительно V_{SS} .....от -0.3В до +6.5В
Напряжение -MCLR относительно V_{SS} (2).....от 0В до +13.25В
Напряжение RA4 относительно V_{SS} .....от 0В до +8.5В
Напряжение на остальных выводах относительно V_{SS} ..... от -0.3В до (V_{DD}+0.3В)
Рассеиваемая мощность (1) 1Вт
Максимальный ток вывода V_{SS}.....300мА
Максимальный ток вывода V_{DD} 250мА
Входной запирающий ток I_{IK} (V_I < 0 или V_I > V_{DD}).....±20мА
Выходной запирающий ток I_{OK} (V_O < 0 или V_O > V_{DD}).....±20мА
Максимальный выходной ток стока канала ввода/вывода.....25мА
Максимальный выходной ток истока канала ввода/вывода.....25мА
Максимальный общий выходной ток стока портов ввода/вывода PORTA, PORTB, PORTE(3).....200мА
Максимальный общий выходной ток истока портов ввода/вывода PORTA, PORTB, PORTE(3).....200мА
Максимальный общий выходной ток стока портов ввода/вывода PORTC, PORTD(3).....200мА
Максимальный общий выходной ток истока портов ввода/вывода PORTC, PORTD(3).....200мА

Примечание 1. Потребляемая мощность рассчитывается по формуле:

$$P = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma\{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OL} \times I_{OL})$$

Примечание 2. Броски напряжения на выводе -MCLR ниже V_{SS} приводят к появлению больших токов (около 80мА), что может привести к срабатыванию защелки. Поэтому рекомендуется последовательно включать резистор сопротивлением от 50ОМ до 100Ом для подачи низкого уровня на этот вывод вместо непосредственного подключения к V_{SS}.

Примечание 3. PORTD, PORTE в микроконтроллерах PIC18F2X2 не реализованы.

Примечание *. Выход за указанные значения может привести к необратимым повреждениям микроконтроллера. Не предусмотрена работа микроконтроллера в предельном режиме в течение длительного времени. Длительная эксплуатация микроконтроллера в недопустимых условиях может повлиять на его надежность.

Рисунок 22-1. График рекомендованных комбинаций значений напряжения питания и тактовой частоты для PIC18FXX2 (Промышленный температурный диапазон)

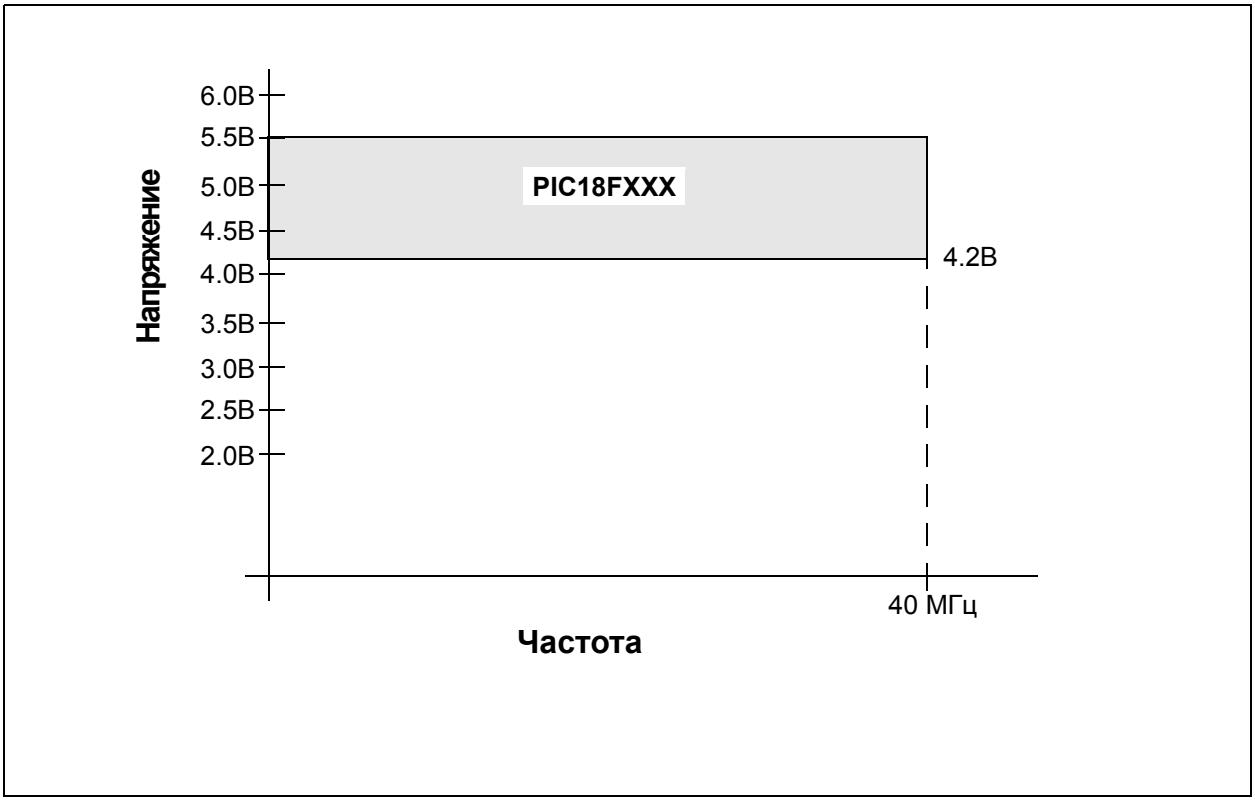
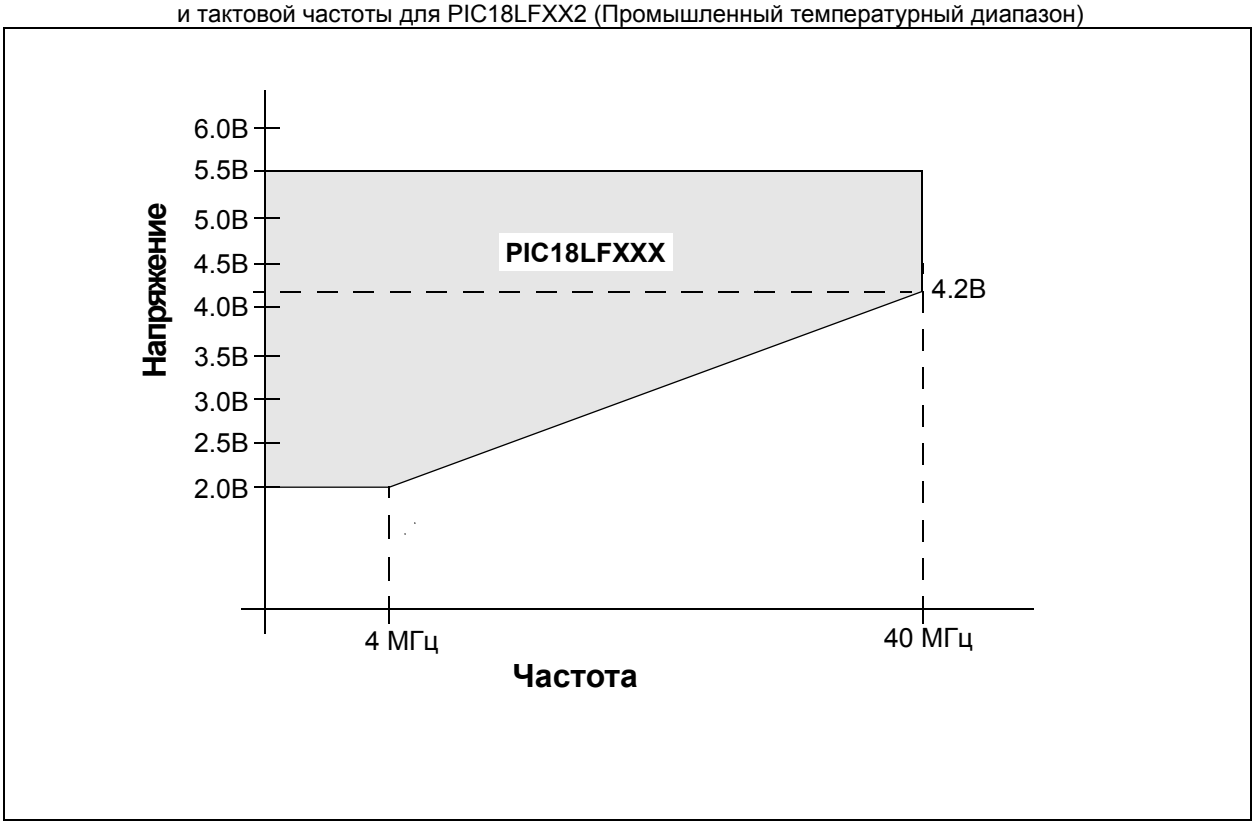


Рисунок 22-2. График рекомендованных комбинаций значений напряжения питания и тактовой частоты для PIC18LFXX2 (Промышленный температурный диапазон)



F_{MAX} = (20.0МГц/В) ($V_{DDAPPMIN}$ – 2.0В) + 4МГц.
Примечание. $V_{DDAPPMIN}$ – минимальное напряжение питания PIC18FXX2 в устройстве.

22.1 Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I

PIC18LFXX2-I			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C				
PIC18LFXX2-I **PIC18LFXX2-E**			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	V_{DD}	Напряжение питания					HS, XT, RC и LP режим
D001		PIC18LFXX2	2.0	-	5.5	В	
D001		PIC18FXX2	4.2	-	5.5	В	
D002	V_{DR}	Напряжение сохранения данных в ОЗУ (1)	1.5	-	-	В	
D003	V_{POR}	Стартовое напряжение V_{DD} для формирования POR	-	-	0.7	В	Смотрите «сброс POR»
D004	S_{VDD}	Скорость нарастания V_{DD} для формирования POR	0.05	-	-	В/мс	Смотрите «сброс POR»
	V_{BOR}	Напряжение сброса BOR					
D005		PIC18LFXX2					
		BOR1 : BOR0 = 11	2.0	-	2.16	В	
		BOR1 : BOR0 = 10	2.7	-	2.86	В	
		BOR1 : BOR0 = 01	4.2	-	4.46	В	
		BOR1 : BOR0 = 00	4.5	-	4.78	В	
D005		PIC18FXX2					
		BOR1 : BOR0 = 1x	-	-	-	В	Не в рабочем диапазоне V_{DD}
		BOR1 : BOR0 = 01	4.2	-	4.46	В	
		BOR1 : BOR0 = 00	4.5	-	4.78	В	

Примечания:

1. Предел, до которого может быть понижено напряжение питания V_{DD} без потери данных в ОЗУ.
2. Ток потребления в основном зависит от напряжения питания и тактовой частоты. Другие факторы, влияющие на ток потребления: выходная нагрузка и частота переключения каналов ввода/вывода; тип тактового генератора; температура и выполняемая программа. Измерения I_{DD} проводилось в следующих условиях: внешний тактовый сигнал (меандр); каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}; -MCLR = V_{DD}; WDT выключен.
3. Потребляемый ток в SLEEP режиме не зависит от типа тактового генератора. При измерении тока все каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}.
4. В RC режиме генератора ток через внешний резистор не учитывается. Ток, протекающий через внешний резистор, может быть рассчитан по формуле Ir = $V_{DD}/2R_{EXT}$ (мА), где R_{EXT} в кОм.

Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I (продолжение)

PIC18LFXX2-I			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C				
PIC18LFXX2-I **PIC18LFXX2-E**			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	I_{DD}	Ток потребления (2,4)					
D010		PIC18LFXX2	-	0.68	2.0	мА	XT, RC, RCIO F_{OSC} = 4МГц, V_{DD}= 2.0В
D010		PIC18FXX2	-	0.4	4	мА	XT, RC, RCIO F_{OSC} = 4МГц, V_{DD}= 4.2В
D010A		PIC18LFXX2	-	28	55	мкА	LP F_{OSC} = 32кГц, V_{DD}= 2.0В
D010A		PIC18FXX2	-	88	250	мкА	LP F_{OSC} = 32кГц, V_{DD}= 4.2В
D010C		PIC18LFXX2	-	-	38	мА	EC, ECIO F_{OSC} = 40МГц, V_{DD}= 5.5В
D010C		PIC18FXX2	-	-	38	мА	EC, ECIO F_{OSC} = 40МГц, V_{DD}= 5.5В
D013		PIC18LFXX2					HS
			-	1.32	3.5	мА	F_{OSC} = 6МГц, V_{DD}= 2.0В
			-	13.46	25	мА	F_{OSC} = 25МГц, V_{DD}= 5.5В
							HS + PLL
			-	19.1	38	мА	F_{OSC} = 10МГц, V_{DD}= 5.5В
D013		PIC18FXX2					HS
			-	13.46	25	мА	F_{OSC} = 25МГц, V_{DD}= 5.5В
							HS + PLL
			-	19.1	38	мА	F_{OSC} = 10МГц, V_{DD}= 5.5В
D014		PIC18LFXX2	-	29.6	55	мкА	Генератор TMR1 F_{OSC} = 32кГц, V_{DD}= 2.0В
D014		PIC18FXX2					OSCB
			-	-	200	мкА	F_{OSC} = 32кГц, V_{DD}= 4.2В, -40°C ≤ T_A ≤ +85°C
			-	-	250	мкА	F_{OSC} = 32кГц, V_{DD}= 4.2В, -40°C ≤ T_A ≤ +125°C

Примечания:

1. Предел, до которого может быть понижено напряжение питания V_{DD} без потери данных в ОЗУ.
2. Ток потребления в основном зависит от напряжения питания и тактовой частоты. Другие факторы, влияющие на ток потребления: выходная нагрузка и частота переключения каналов ввода/вывода; тип тактового генератора; температура и выполняемая программа. Измерения I_{DD} проводилось в следующих условиях: внешний тактовый сигнал (меандр); каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}; -MCLR = V_{DD}; WDT выключен.
3. Потребляемый ток в SLEEP режиме не зависит от типа тактового генератора. При измерении тока все каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}.
4. В RC режиме генератора ток через внешний резистор не учитывается. Ток, протекающий через внешний резистор, может быть рассчитан по формуле Ir = $V_{DD}/2R_{EXT}$ (мА), где R_{EXT} в кОм.

Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I (продолжение)

PIC18LFXX2-I			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C				
PIC18LFXX2-I **PIC18LFXX2-E**			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	I_{PD}	Ток потребления в SLEEP режиме(3)					
D020		PIC18LFXX2	-	0.09	2	мкА	V_{DD}= 2.0В, -40°C ≤ T_A ≤ +85°C
			-	0.11	4	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +85°C
D020		PIC18FXX2		0.10	3	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +85°C
			-	0.11	4	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +85°C
D021B			-	0.10	15	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +125°C
			-	0.11	20	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +125°C
		Ток потребления отдельных модулей					
D022	ΔI_{WDT}	Сторожевой таймер WDT	-	-	1	мкА	V_{DD}= 2.0В
		PIC18LFXX2	-	-	15	мкА	V_{DD}= 5.5В
D022		Сторожевой таймер WDT		-	15	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +85°C
		PIC18FXX2	-	-	20	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +125°C
D022A	ΔI_{BOR}	Сброс BOR	-	-	445	мкА	V_{DD}= 5.5В
		PIC18LFXX2					
D022A		Сброс BOR		-	50	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +85°C
		PIC18FXX2	-	-	50	мкА	V_{DD}= 5.5В, -40°C ≤ T_A ≤ +125°C
D022B	ΔI_{LWD}	Модуль LVD	-	-	45	мкА	V_{DD}= 2.0В
		PIC18LFXX2					
D022B		Модуль LVD		-	40	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +85°C
		PIC18FXX2	-	-	50	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +125°C
D025	ΔI_{OSCB}	Генератор TMR1	-	-	15	мкА	V_{DD}= 2.0В
		PIC18LFXX2					
D025		Генератор TMR1		-	100	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +85°C
		PIC18FXX2	-	-	120	мкА	V_{DD}= 4.2В, -40°C ≤ T_A ≤ +125°C
D025	ΔI_{AD}	АЦП	-	-	15	мкА	V_{DD}= 2.0В, АЦП включено,
		PIC18LFXX2					нет преобразования

Примечания:

1. Предел, до которого может быть понижено напряжение питания V_{DD} без потери данных в ОЗУ.
2. Ток потребления в основном зависит от напряжения питания и тактовой частоты. Другие факторы, влияющие на ток потребления: выходная нагрузка и частота переключения каналов ввода/вывода; тип тактового генератора; температура и выполняемая программа. Измерения I_{DD} проводилось в следующих условиях: внешний тактовый сигнал (меандр); каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}; -MCLR = V_{DD}; WDT выключен.
3. Потребляемый ток в SLEEP режиме не зависит от типа тактового генератора. При измерении тока все каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}.
4. В RC режиме генератора ток через внешний резистор не учитывается. Ток, протекающий через внешний резистор, может быть рассчитан по формуле Ir = $V_{DD}/2R_{EXT}$ (мА), где R_{EXT} в кОм.

22.2 Электрические характеристики PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I

			Стандартные рабочие условия (если не указано иное) Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	V_{IL}	Входное напряжение низкого уровня					
		Канал порта ввода/вывода					
D030		ТТЛ буфер	V_{SS}	-	0.8	В	V_{DD} = от 4.5В до 5.5В
D030A			V_{SS}	-	$0.15V_{DD}$	В	иначе
D031		Триггер Шмитта	V_{SS}	-	$0.2V_{DD}$	В	
D032		-MCLR, OSC1 (RC, EC)(1)	V_{SS}	-	$0.2V_{DD}$	В	
D033		OSC1 (XT, HS, LP)	V_{SS}	-	$0.3V_{DD}$	В	
	V_{IH}	Входное напряжение высокого уровня					
		Канал порта ввода/вывода					
D040		ТТЛ буфер	2.0	-	V_{DD}	В	V_{DD} = от 4.5В до 5.5В
D040A			$0.25V_{DD}$+0.8	-	V_{DD}	В	иначе
D041		Триггер Шмитта	$0.8V_{DD}$	-	V_{DD}	В	
D042		-MCLR	$0.8V_{DD}$	-	V_{DD}	В	
D042A		OSC1 (XT, HS, LP)	$0.7V_{DD}$	-	V_{DD}	В	
D043		OSC1 (RC, EC)(1)	$0.9V_{DD}$	-	V_{DD}	В	
D070	I_{PURB}	Ток через подтягивающие резисторы PORTB	50	-	400	мкА	V_{DD} = 5.0В, V_{PIN} = V_{SS}
	I_{IL}	Входной ток утечки (2,3)					
D060		Порт ввода/вывода	-	-	±1	мкА	V_{SS} ≤ V_{PIN} ≤ V_{DD}, 3-е сост.
D061		-MCLR, RA4/T0CKI	-	-	±5	мкА	V_{SS} ≤ V_{PIN} ≤ V_{DD}
D063		OSC1	-	-	±5	мкА	V_{SS} ≤ V_{PIN} ≤ V_{DD}, XT, HS, LP
	V_{OL}	Выходное напряжение низкого уровня					V_{DD} = 4.5В
D080		Канал ввода/вывода	-	-	0.6	В	I_{OL}=8.5 мА, -40°C до +85°C
D080A			-	-	0.6	В	I_{OL}=7.0 мА, -40°C до +125°C
D083		OSC2/CLKOUT (RC)	-	-	0.6	В	I_{OL}=1.6 мА, -40°C до +85°C
D083A			-	-	0.6	В	I_{OL}=1.2 мА, -40°C до +125°C
	V_{OH}	Выходное напряжение высокого уровня					V_{DD}=4.5В
D090		Канал ввода/вывода (3)	V_{DD} - 0.7	-	-	В	I_{OH}=-3.0 мА,-40°C до +85°C
D090A			V_{DD} - 0.7	-	-	В	I_{OH}=-2.5 мА,-40°C до +125°C
D092		OSC2/CLKOUT (RC)	V_{DD} - 0.7	-	-	В	I_{OH}=-1.3 мА,-40°C до +85°C
D092A			V_{DD} - 0.7	-	-	В	I_{OH}=-1.0 мА,-40°C до +125°C
D150*	V_{OD}	Напряжение на выходе с открытым стоком	-	-	8.5	В	RA4
		Емкостная нагрузка на выходах					
D100	C_{OSC2}	Вывод OSC2	-	-	15	пФ	XT, HS, LP
D101	C_{IO}	Все каналы ввода/вывода и OSC2 в RC режиме	-	-	50	пФ	
D102	C_B	SCL, SDA в режиме I^2C	-	-	400	пФ	Режим I^2C

Примечания:

1. В RC режиме генератора на входе OSC1 включен триггер Шмитта. Не рекомендуется использовать внешний тактовый сигнал для PICmicro в RC режиме тактового генератора.
2. Ток утечки на выводе -MCLR зависит от приложенного напряжения. Параметры указаны для нормального режима работы. В других режимах может возникнуть больший ток утечки.
3. Отрицательный ток показывает, что он вытекает из вывода.

Рисунок 22-3. Характеристика детектора пониженного напряжения

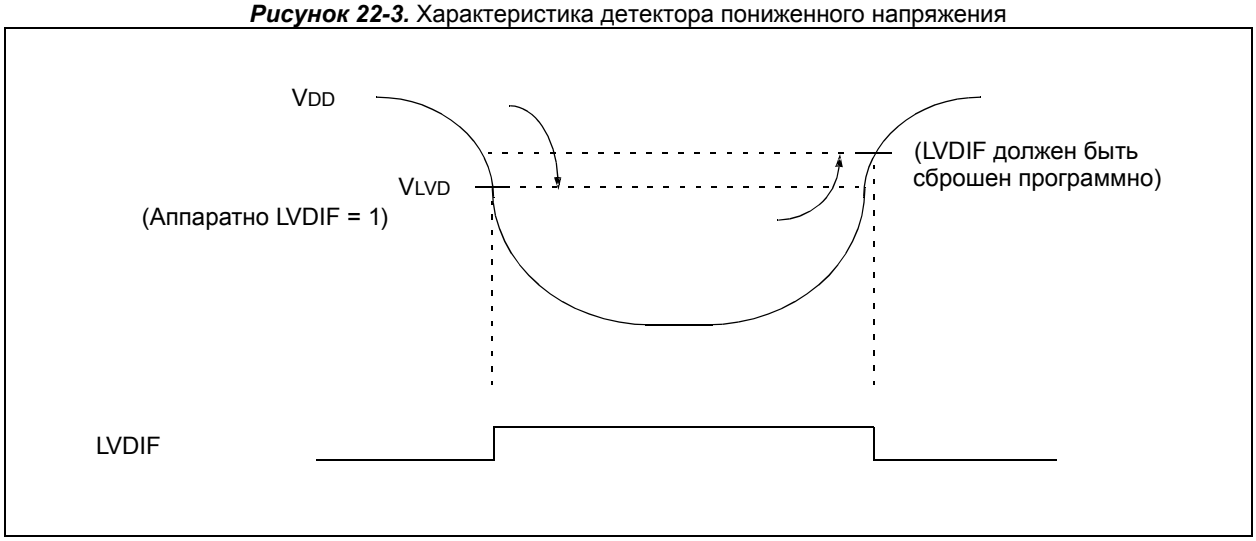


Таблица 22-1. Характеристика детектора пониженного напряжения

			Стандартные рабочие условия (если не указано иное) Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
D420	V_{LVD}	Порог детектора пониженного напряжения питания (V_{DD} = 5.0В)					
		LVV = 0000	1.8	1.86	1.91	В	
		LVV = 0001	2.0	2.06	2.12	В	
		LVV = 0010	2.2	2.27	2.34	В	
		LVV = 0011	2.4	2.47	2.55	В	
		LVV = 0100	2.5	2.58	2.66	В	
		LVV = 0101	2.6	2.78	2.86	В	
		LVV = 0110	2.8	2.89	2.98	В	
		LVV = 0111	3.0	3.10	3.20	В	
		LVV = 1000	3.3	3.41	3.52	В	
		LVV = 1001	3.5	3.61	3.72	В	
		LVV = 1010	3.6	3.72	3.84	В	
		LVV = 1011	3.8	3.92	4.04	В	
		LVV = 1100	4.0	4.13	4.26	В	
		LVV = 1101	4.2	4.33	4.46	В	
		LVV = 1110	4.5	4.64	4.78	В	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Таблица 22-2. Программирование памяти программ и памяти данных

			Стандартные рабочие условия (если не указано иное) Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
		Внутрисхемное программирование (1)					
D110	V_{PP}	Напряжение на –MCLR/V_{PP}	9.00	-	13.25	В	(2)
D112	I_{PP}	Ток –MCLR/V_{PP}	-	-	5	мА	
D113	I_{DDP}	Ток потребления во время программирования	-	-	10	мА	
		EEPROM память данных					
D120	E_D	Число циклов стирание/запись	100K	1M	-	C/3	-40°C ≤ T_A ≤ +85°C
D120A	E_D	Число циклов стирание/запись	10K	100K	-	C/3	-40°C ≤ T_A ≤ +125°C
D121	V_{DRW}	Напряжение питания для записи/чтения	V_{MIN}	-	5.5	В	V_{MIN}- минимальное напряжение питания
D122	T_{DEW}	Время цикла стирание/запись	-	4	-	мс	
		FLASH память программ					
D130	E_P	Число циклов стирание/запись	10K	100K	-	C/3	-40°C ≤ T_A ≤ +85°C
D130A	E_P	Число циклов стирание/запись	1000	10K	-	C/3	-40°C ≤ T_A ≤ +125°C
D131	V_{PR}	Напряжение питания для чтения	V_{MIN}	-	5.5		V_{MIN}- минимальное напряжение питания
D132A	V_{IE}	Напряжение питания для блочного стирания	4.5В	-	5.5		Порт ICSP
D132B	V_{IW}	Напряжение питания для полного внешнего стирания или записи	4.5В	-	5.5		Порт ICSP
D132A	V_{PEW}	Напряжение питания для самопрограммирования	V_{MIN}	-	5.5		V_{MIN}- минимальное напряжение питания
D133	T_{IE}	Длительность цикла блочного стирания (ICSP)	-	5	-	мс	V_{DD} > 4.5В
D133A	T_{IW}	Длительность цикла полного стирания или записи (ICSP)	1	-	-	мс	V_{DD} > 4.5В
D133A	T_{IW}	Время цикла самопрограммирования	-	2	-	мс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Параметры для программирования внутренней памяти программ командами табличной записи.
2. Указанное напряжение на выводе может присутствовать и не в режиме программирования, но это не рекомендуется.
3. Дополнительно смотрите разделы 6.5.1 и 5.5.2.

22.3 Временные диаграммы и спецификации

22.3.1 Символьное обозначение временных параметров

Символьное обозначение временных параметров имеет один из следующих форматов:

1. TppS2ppS
2. TppS
3. $T_{CC:ST}$ (только спецификация I^2C)
4. T_S (только спецификация I^2C)

T			
F	Частота	T	Время

Строчные символы (pp) и их значение

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	-RD
cs	-CS	rw	-RD или -WR
di	SDI	sc	SCK
do	SDO	ss	-SS
dt	Входные данные	t0	T0CKI
io	Канал ввода/вывода	t1	T1CKI
mc	-MCLR	wr	-WR

Прописные символы и их значение

S			
F	Задний фронт	P	Период
H	Высокий уровень	R	Передний фронт
I	Неверный (3-е состояние)	V	Верный
L	Низкий уровень	Z	3-е состояние
Только I^2C			
AA	Доступ вывода	High	Высокий уровень
BUF	Шина свободна	Low	Низкий уровень

$T_{CC:ST}$ (только спецификация I^2C)

CC			
HD	Удержание	SU	Установка
ST			
DAT	Сохранение данных на входе	STO	Условие STOP
STA	Условие START	Low	Низкий уровень

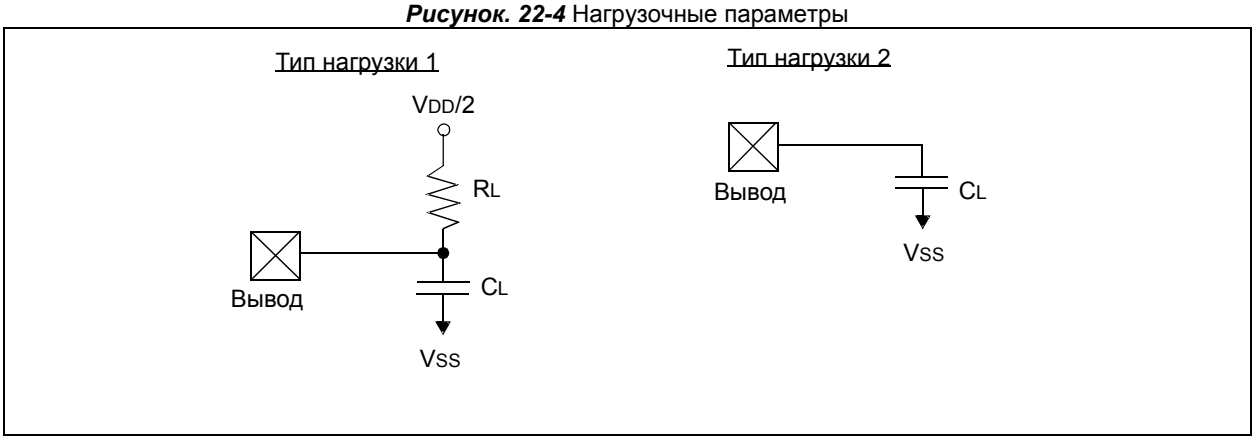
22.3.2 Условия временных диаграмм и параметров

Температурный диапазон и напряжение питания указаны в таблице 22-3. На рисунке 22-4 представлены условия емкостной нагрузки выводов.

Таблица 22-3. Значения температуры и напряжения питания для временных диаграмм

Рабочий диапазон напряжения питания V_{DD} смотрите в разделе 22.1 и 22.2.	**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C

Рисунок. 22-4 Нагрузочные параметры



R_L = 464Ом
C_L = 50пФ (для всех выводов, кроме OSC2, включая PORTD и PORTE в режиме портов ввода/вывода)

22.3.3 Временные диаграммы и параметры

Рисунок 22-5. Временная диаграмма внешнего тактового сигнала

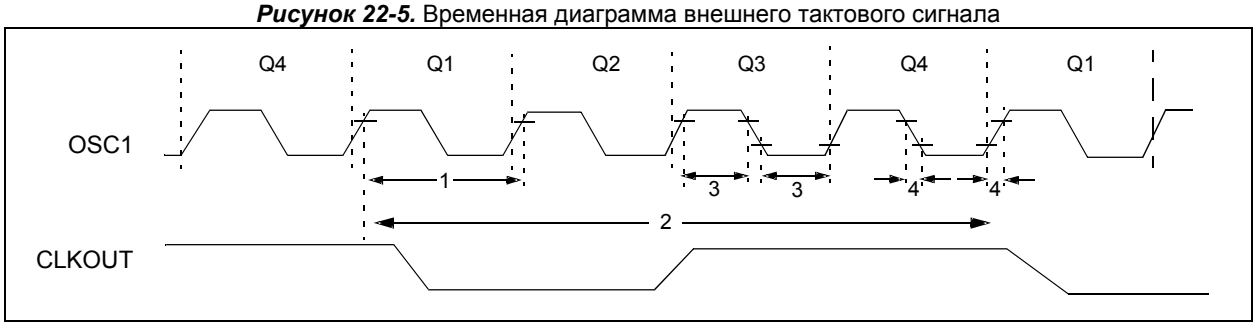


Таблица 22-4. Параметры внешнего тактового сигнала

№ пар.	Обоз.	Описание	Мин.	Макс.	Ед.	Примечание
1A	F_{OSC}	Частота внешнего тактового сигнала(1) Частота генератора(1)	DC DC 0.1 4 4 5	40 4 4 25 10 200	МГц МГц МГц МГц МГц кГц	EC, ECIO RC XT HS HS + PLL LP
1	T_{OSC}	Период внешнего тактового сигнала(1) Период генератора(1)	25 250 250 25 100 25	- - 10000 250 250 -	нс нс нс нс нс мкс	EC, ECIO RC XT HS HS + PLL LP
2	T_{CY}	Время выполнения инструкции(1)	100.0	DC	нс	$T_{CY} = 4/F_{OSC}$
3	$T_{OS}L$, $T_{OS}H$	Длительность высокого/низкого уровня CLKIN (OSC1)	30 2.5 10	- - -	нс мкс нс	XT LP HS
4	TosR, TosF	Длительность переднего/заднего фронта внешнего тактового сигнала (OSC1)	- - -	25 50 7.5	нс нс нс	XT LP HS режим

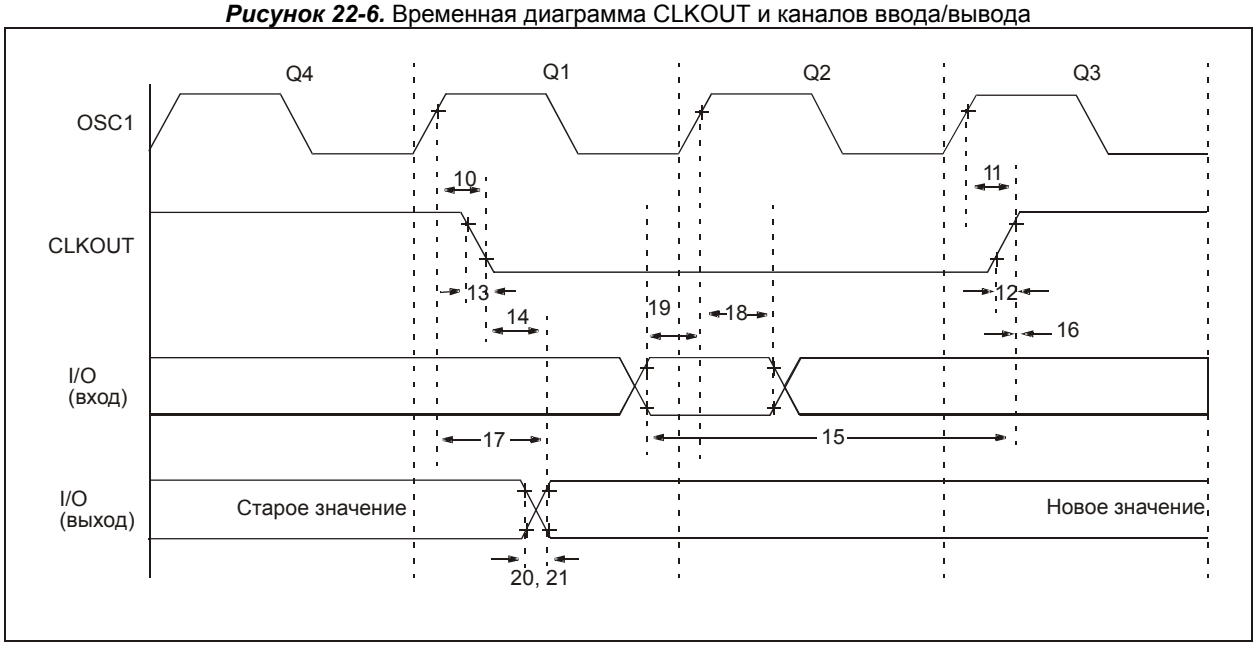
Примечание 1. Машинный цикл микроконтроллера равняется 4 периодам тактового сигнала. Все приведенные значения основываются на характеристиках конкретного типа генератора в стандартных условиях при выполнении программы. Выход за указанные пределы может привести к нестабильной работе генератора и/или к большему потребляемому току. Все микроконтроллеры проверены в режиме "Мин." при внешнем тактовом сигнале на выводе OSC1/CLKIN.

Таблица 22-5. Параметры тактового сигнала в режиме PLL

№ пар.	Обоз.	Описание	Мин.	Тип.**	Макс.	Ед.	Примечание
-	F_{OSC}	Частота генератора	4	-	10	МГц	Только HS режим
-	F_{SYS}	Внутренняя тактовая частота	16	-	40	МГц	Только HS режим
-	t_{rc}	Время запуска схемы PLL	-	-	2	мс	
-	Δ_{CLK}	Стабильность сигнала CLKOUT	-2	-	+2	%	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Рисунок 22-6. Временная диаграмма CLKOUT и каналов ввода/вывода



Примечание. Условие нагрузки показано на рисунке 22-4.

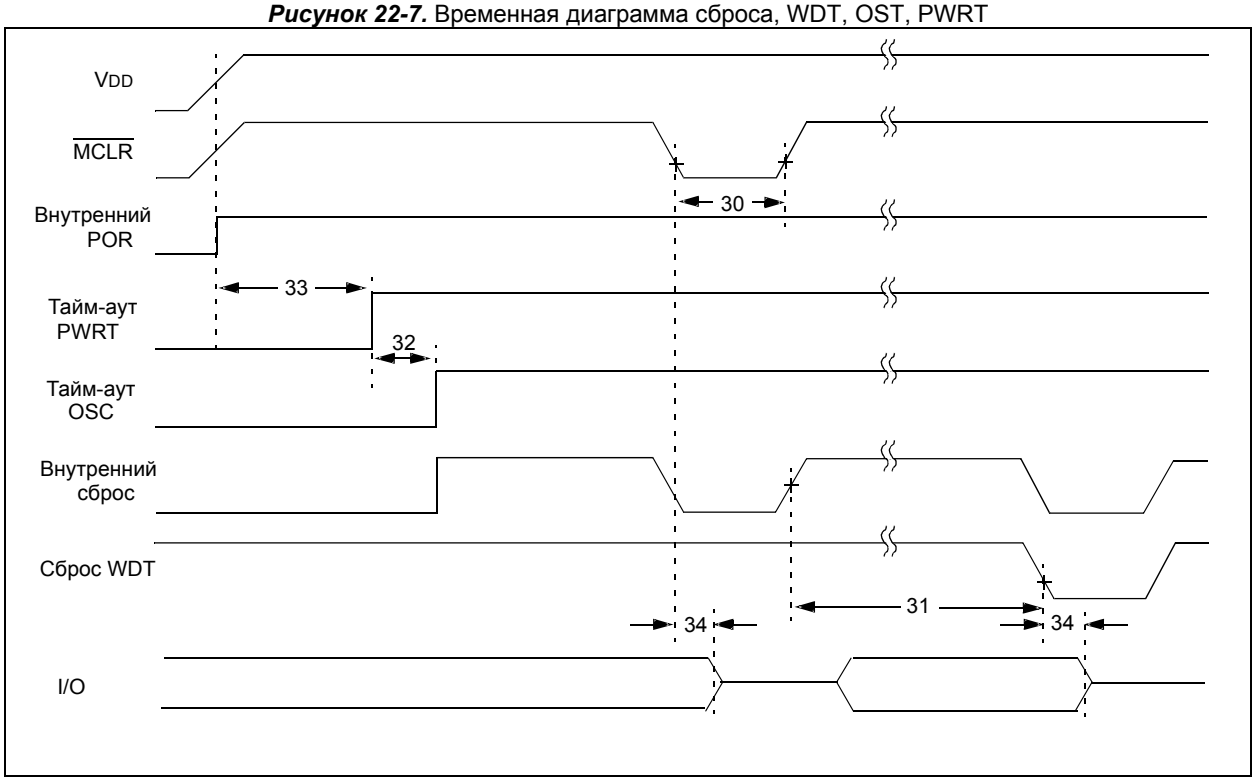
Таблица 22-6 Параметры CLKOUT и каналов ввода/вывода

№ пар.	Обоз.	Описание		Мин.	Тип.	Макс.	Ед.	Примечание
10*	TosH2ckL	От OSC1 ↑ до CLKOUT ↓		-	75	200	нс	(1)
11*	TosH2ckH	От OSC1 ↑ до CLKOUT ↑		-	75	200	нс	(1)
12*	TckR	CLKOUT длит. переднего фронта		-	35	100	нс	(1)
13*	TckF	CLKOUT длит. заднего фронта		-	35	100	нс	(1)
14*	TckL2ioV	От CLKOUT ↓ до установл. выхода		-	-	$0.5T_{CY}+20$	нс	(1)
15*	TioV2ckH	От установл. входа до CLKOUT ↑		$0.55T_{CY}+25$	-	-	нс	(1)
16*	TckH2ioI	Удержание входа после CLKOUT ↑		0	-	-	нс	(1)
17*	TosH2ioV	От OSC1 ↑ до установл. выхода		-	50	150	нс	
18*	TosH2ioI	Удержание входа после OSC1 ↑	**F**	100	-	-	нс	
			LF	200	-	-	нс	
19	TioV2osH	Переход в режим входа относ. OSC1↑		0	-	-	нс	
20	TioR	Длительность переднего фронта на выходе порта ввода/вывода	**F**	-	10	25	нс	
			LF	-	-	60	нс	
21	TioF	Длительность заднего фронта на выходе порта ввода/вывода	**F**	-	10	25	нс	
			LF	-	-	60	нс	
22***	Tinp	Длит. высокого/низкого уровня INT		T_{CY}	-	-	нс	
23***	Trbp	Длит. высокого/низкого уровня RB7:RB4		T_{CY}	-	-	нс	
24***	Trcp	Длит. высокого/низкого уровня RC7:RC4		T_{CY}	-	-	нс	

*** - Асинхронные события, не связанные с внутренним тактовым сигналом.

Примечание 1. Измерения проведены в RC режиме генератора, где CLKOUT = 4 x T_{OSC}.

Рисунок 22-7. Временная диаграмма сброса, WDT, OST, PWRT



Примечание. Условие нагрузки показано на рисунке 22-4.

Рисунок 22-8. Временная диаграмма работы сброса BOR

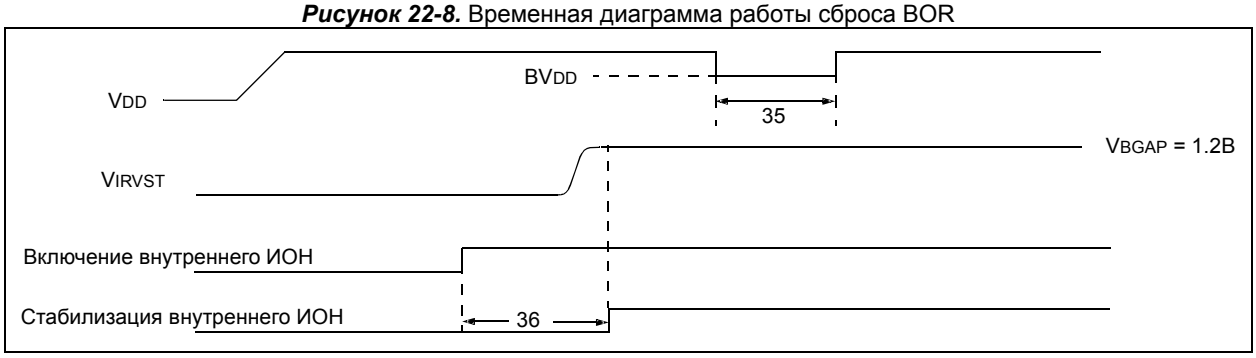
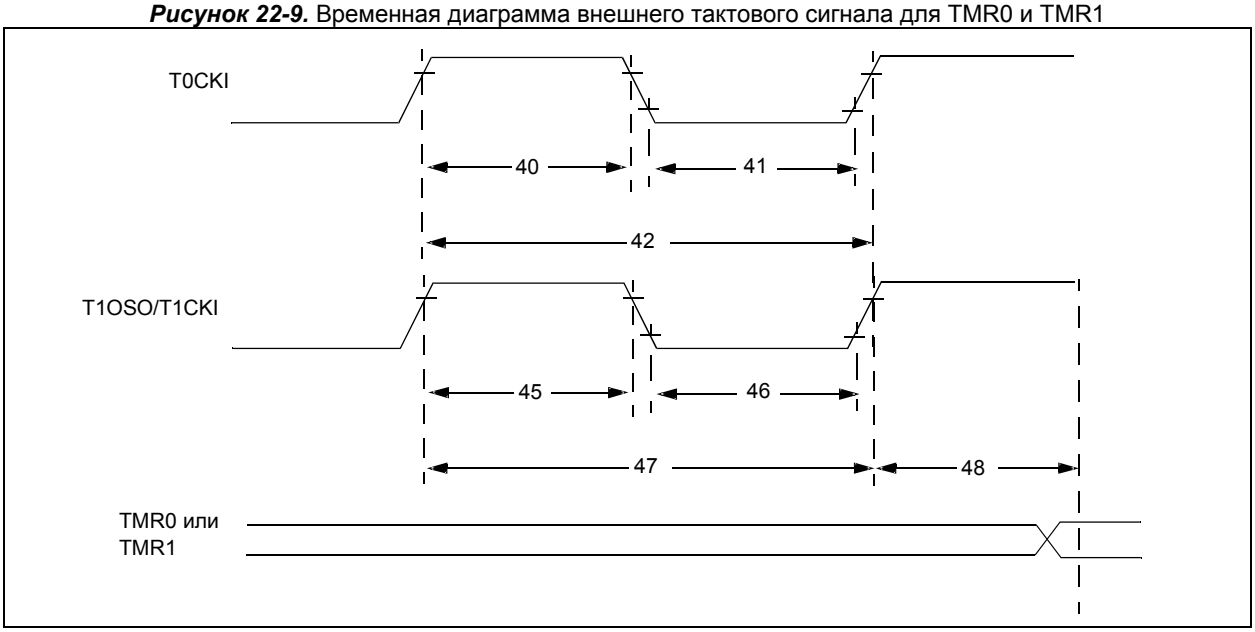


Таблица 22-7 Параметры сброса, WDT, OST, PWRT, BOR

№ пар.	Обоз.	Описание	Мин.	Тип.	Макс.	Ед.	Примечание
30	TmcL	Длительность импульса -MCLR	2	-	-	мкс	
31*	Twdt	Период переполнения WDT (без предделителя)	7	18	33	мс	
32	Tost	Период OST	-	$1024T_{OSC}$	-	-	T_{OSC} = период OSC1
33*	Tpwrt	Период PWRT	28	72	132	мс	
34	T_{IOZ}	От сброса -MCLR или WDT до перевода каналов ввода/вывода 3-е состояние	-	2	-	мкс	
35	T_{BOR}	Длительность импульса BOR	200	-	-	мкс	$V_{DD} \leq V_{BOR}$ (D005)
36	T_{IVRST}	Время стабилизации внутреннего ИОН	-	20	50	мкс	

Рисунок 22-9. Временная диаграмма внешнего тактового сигнала для TMR0 и TMR1

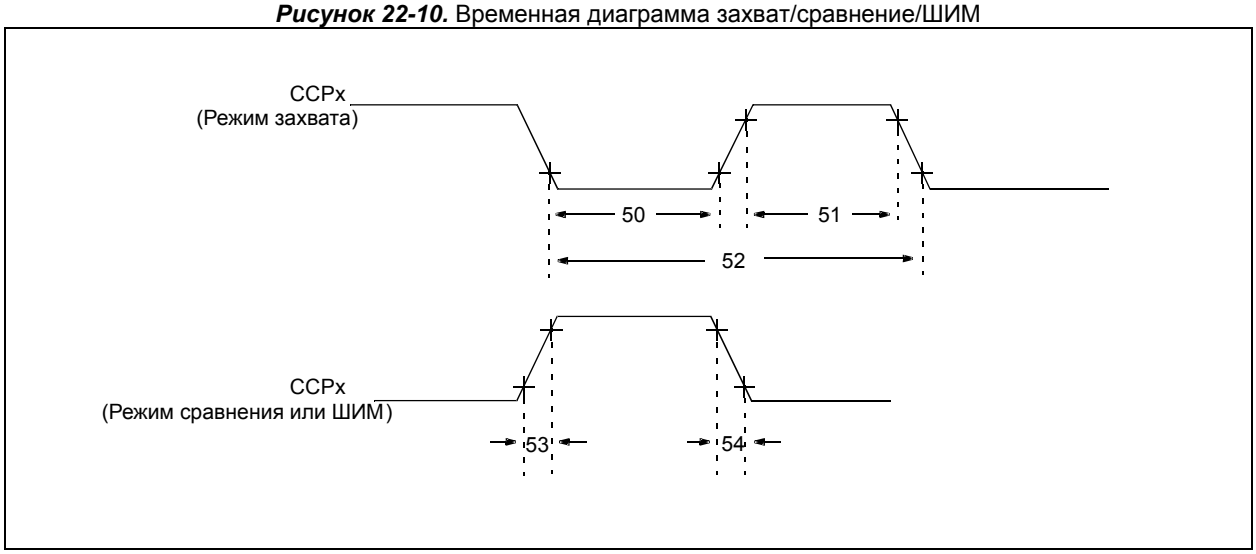


Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-8. Параметры внешнего тактового сигнала для TMR0 и TMR1

№ пар.	Обоз.	Описание			Мин.	Макс.	Ед.	Примечание
40	Tt0H	Длительность высокого уровня T0CKI	Без предделителя		$0.5T_{CY}$+20	-	нс	Также должен выполняться параметр 42
			С предделителем		10	-	нс	
41	Tt0L	Длительность низкого уровня T0CKI	Без предделителя		$0.5T_{CY}$+20	-	нс	
			С предделителем		10	-	нс	
42	Tt0P	Период T0CKI	Без предделителя		T_{CY}+40	-	нс	
			С предделителем		20 или $(T_{CY}+40)/N$	-	нс	N = коэфф.предд.
45	Tt1H	Длительность высокого уровня T1CKI	Синхр.реж. без преддел.		$0.5T_{CY}$+20	-	нс	Также должен выполняться параметр 47
			Синхр. режим с преддел.	**F**	10	-	нс	
				LF	25	-	нс	
			Асинхронный режим	**F**	30	-	нс	
				LF	50	-	нс	
46	Tt1L	Длительность низкого уровня T1CKI	Синхр.реж. без преддел.		$0.5T_{CY}$+20	-	нс	Также должен выполняться параметр 47
			Синхр. режим с преддел.	**F**	10	-	нс	
				LF	25	-	нс	
			Асинхронный режим	**F**	30	-	нс	
				LF	TBD	-	нс	
47	Tt1P	Период T1CKI	Синхронный режим		20 или $(T_{CY}+40)/N$	-	нс	N = коэфф.предд.
			Асинхронный режим		60	-	нс	
	Ft1	Частота резонатора для TMR1 (T1OSCEN=1)			DC	50	кГц	
48	TCKE1	Задержка от активного фронта тактового сигнала до приращения TMR1			$2T_{OSC}$	$7T_{OSC}$	-	

Рисунок 22-10. Временная диаграмма захват/сравнение/ШИМ

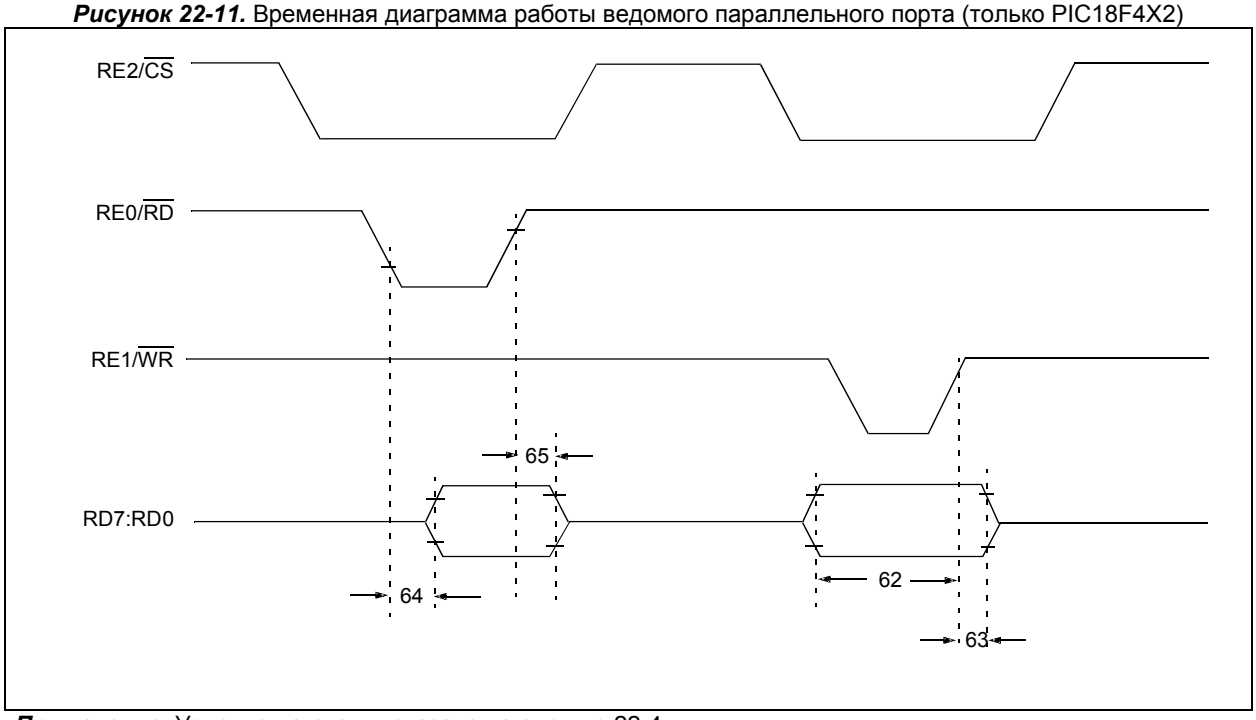


Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-9. Параметры захват/сравнение/ШИМ (CCP1 и CCP2)

№ пар.	Обоз.	Описание			Мин.	Макс.	Ед.	Примечание
50	TccL	Сигнал низкого уровня CCP1 и CCP2	Без предделителя		$0.5T_{CY}$+20	-	нс	
			С предделителем	**F**	10	-	нс	
				LF	20	-	нс	
51	TccL	Сигнал высокого уровня CCP1 и CCP2	Без предделителя		$0.5T_{CY}$+20	-	нс	
			С предделителем	**F**	10	-	нс	
				LF	20	-	нс	
52	TccP	Период входного сигнала CCP			$(3T_{CY}+40)/N$	-	нс	N = коэфф.предд.
53	TccR	Время установление высокого уровня сигн. на вых. CCP1 и CCP2		**F**	-	25	нс	
				LF	-	45	нс	
54	TccF	Время установление низкого уровня сигн. на вых. CCP1 и CCP2		**F**	-	25	нс	
				LF	-	45	нс	

Рисунок 22-11. Временная диаграмма работы ведомого параллельного порта (только PIC18F4X2)

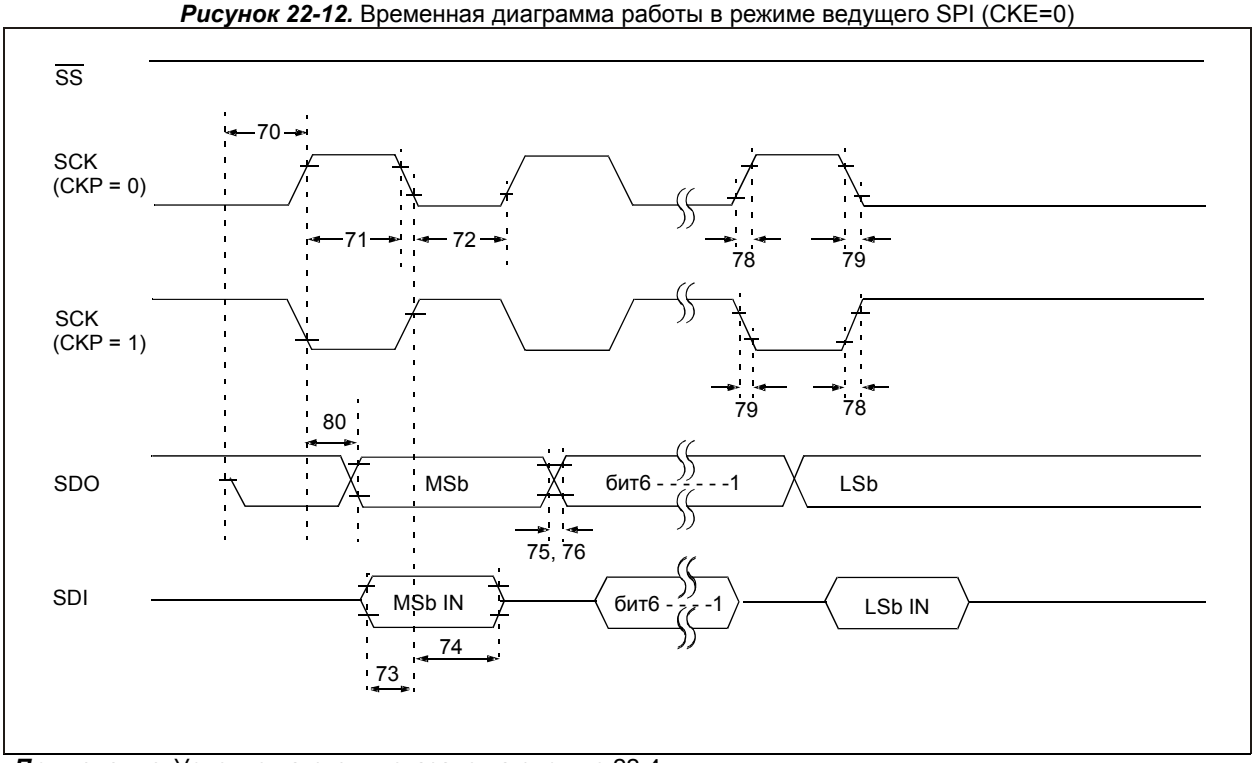


Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-10. Параметры работы ведомого параллельного порта (только PIC18F4X2)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
62	TdtV2H	Установка данных перед -WR↑ или -CS↑		20 25	- -	нс нс	 Только для расшир. диап.
63	TwrH2dtl	Удержание данных после -WR↑ или -CS↑	**F**	20	-	нс	
			LF	35	-	нс	
64	TrdL2dtV	Формирование данных после -RD↓ и -CS↓		- -	80 90	нс нс	 Только для расшир. диап.
65	TrdH2dtl	Неправильные данные после -RD↑ или -CS↑		10	30	нс	
66	TibfINH	Запрет сброса бита IBF от -WR↑ или -CS↑		-	$3T_{CY}$		

Рисунок 22-12. Временная диаграмма работы в режиме ведущего SPI (CKE=0)



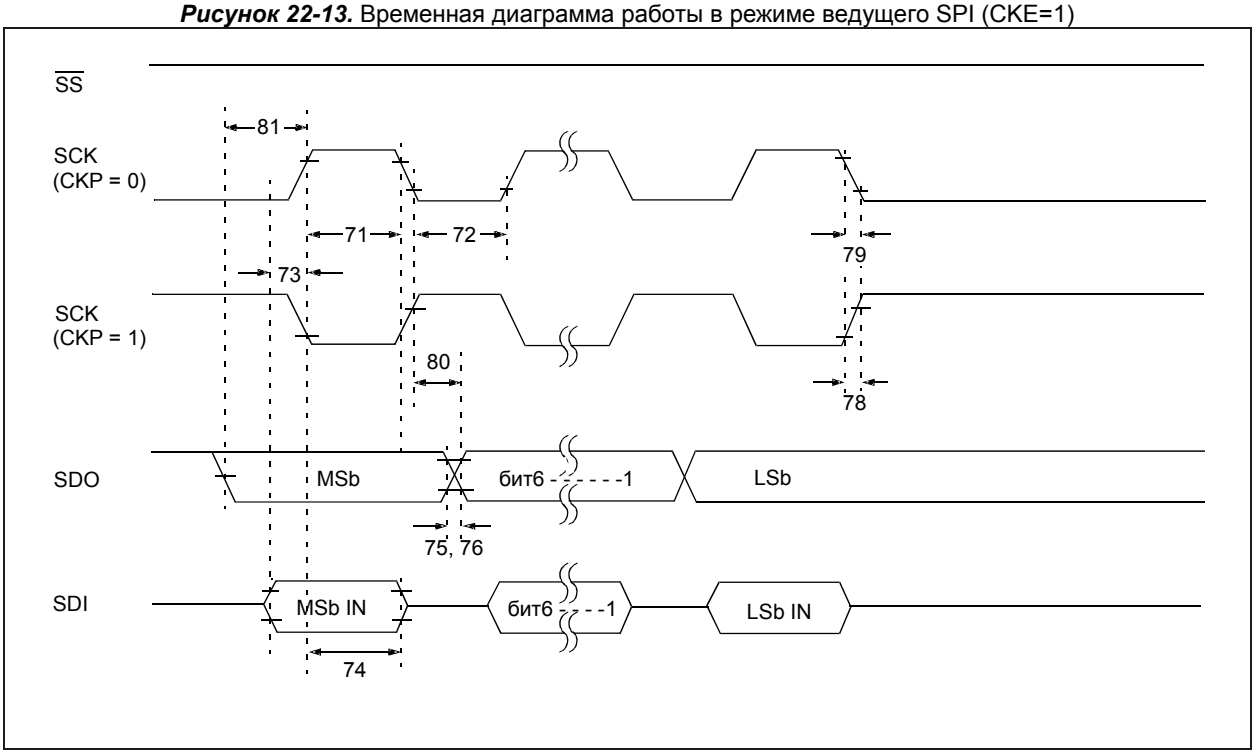
Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-11. Параметры работы в режиме ведущего SPI (CKE=0)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	нс	
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
71A			Одиночный	40	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
72A			Одиночный	40	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	нс	
73A	$T_{B}2_{B}$	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**F**	-	25	нс	
			LF	-	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	25	нс	
78	TscR	Длительность переднего фронта на SCK	**F**	-	25	нс	
			LF	-	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**F**	-	50	нс	
			LF	-	100	нс	

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рисунок 22-13. Временная диаграмма работы в режиме ведущего SPI (CKE=1)



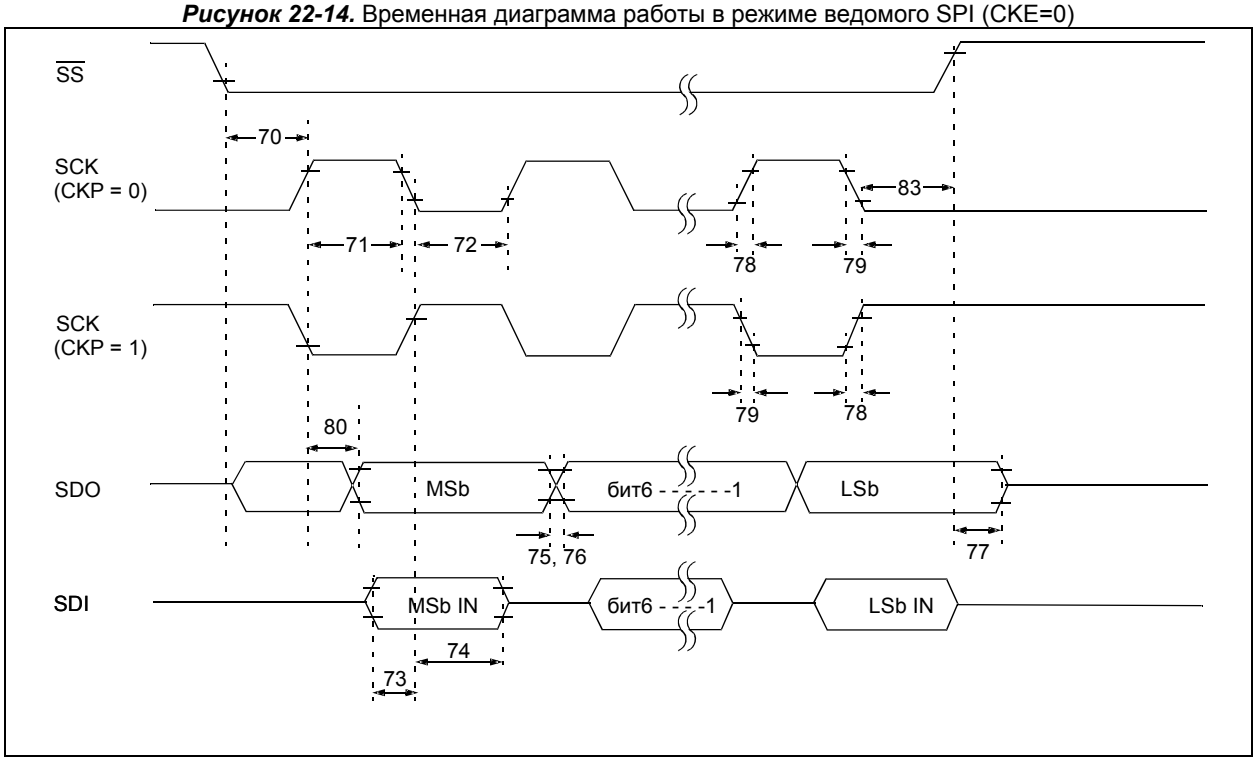
Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-12. Параметры работы в режиме ведущего SPI (CKE=1)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
71A			Одиночный	40	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
72A			Одиночный	40	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	нс	
73A	T_B2_B	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**F**	-	25	нс	
			LF	-	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	25	нс	
78	TscR	Длительность переднего фронта на SCK	**F**	-	25	нс	
			LF	-	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**F**	-	50	нс	
			LF	-	100	нс	
81	TdoV2scH, TdoV2scL	Установка данных на выходе SDO после фронта SCK		T_{CY}	-	нс	

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рисунок 22-14. Временная диаграмма работы в режиме ведомого SPI (CKE=0)



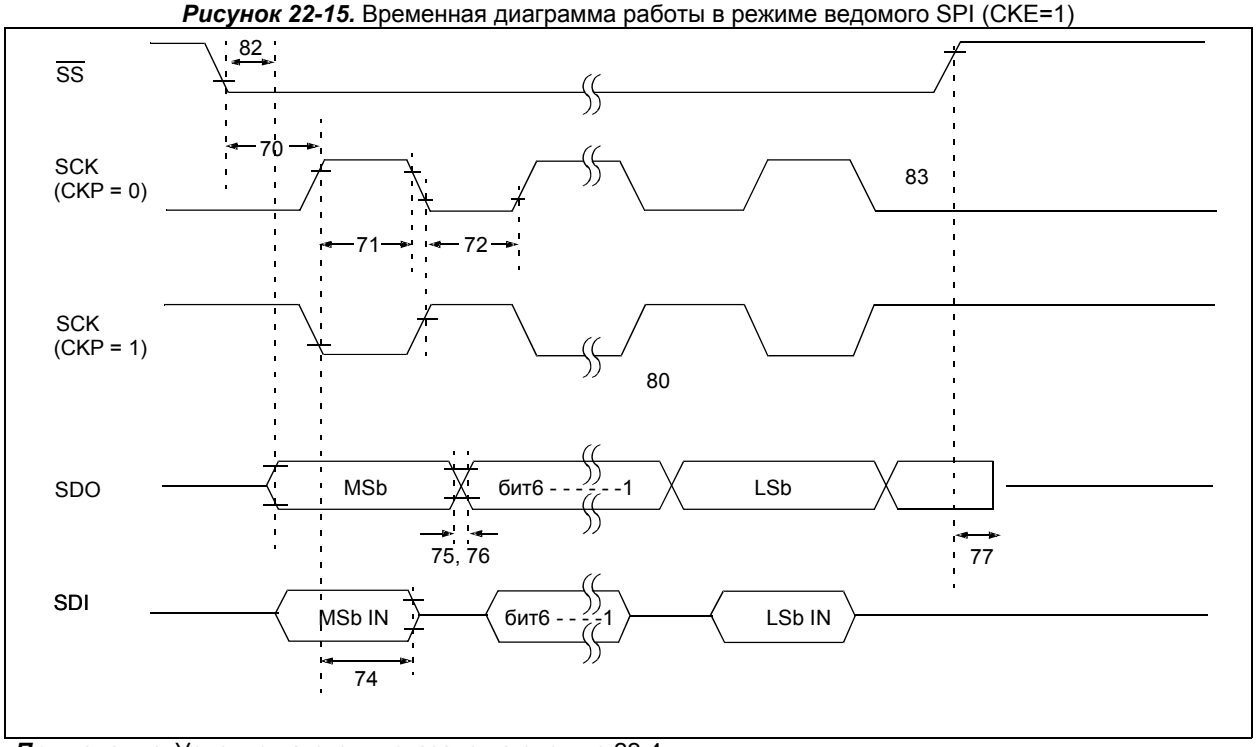
Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-13. Параметры работы в режиме ведомого SPI (CKE=0)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	нс	
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
71A			Одиночный	40	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	нс	
72A			Одиночный	40	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	нс	
73A	$T_{B}2_{B}$	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**F**	-	25	нс	
			LF	-	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	25	нс	
77	TssH2doZ	Перевод SDO в 3-е состояние после SS↑		10	50	нс	
78	TscR	Длительность переднего фронта на SCK	**F**	-	25	нс	
			LF	-	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**F**	-	50	нс	
			LF	-	100	нс	
83	TscH2ssH, TscL2ssH	SS↑ после фронта SCK		$1.5T_{CY}$ + 40	-	нс	

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рисунок 22-15. Временная диаграмма работы в режиме ведомого SPI (CKE=1)



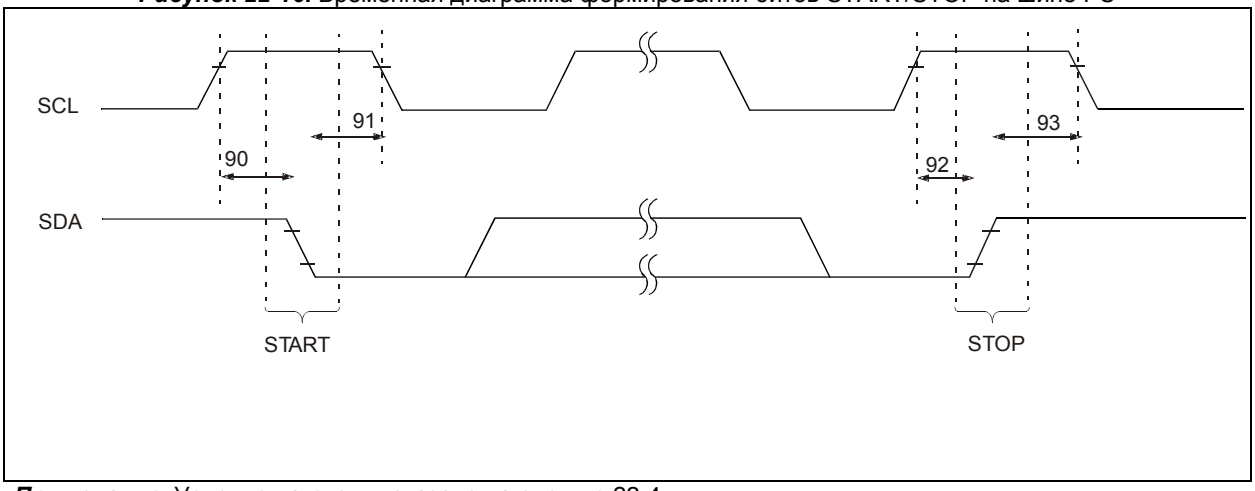
Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-14. Параметры работы в режиме ведомого SPI (CKE=1)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	нс	
71 71A	TscH	Высокий уров. сигн. SCK	Непрерыван. Одиночный	$1.25T_{CY}$ + 30 40	- -	нс нс	 (1)
72 72A	TscL	Низкий уров. сигн. SCK	Непрерыван. Одиночный	$1.25T_{CY}$ + 30 40	- -	нс нс	 (1)
73A	T_B2_B	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**F** **LF**	- -	25 45	нс нс	
76	TdoF	Длительность заднего фронта на SDO		-	25	нс	
77	TssH2doZ	Перевод SDO в 3-е состояние после SS↑		10	50	нс	
78	TscR	Длительность переднего фронта на SCK	**F** **LF**	- -	25 45	нс нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**F** **LF**	- -	50 100	нс нс	
82	TssL2doV	Достов. данные на вых. SDO после SS↓	**F** **LF**	- -	50 100	нс нс	
83	TscH2ssH, TscL2ssH	SS↑ после фронта SCK		$1.5T_{CY}$ + 40	-	нс	

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рисунок 22-16. Временная диаграмма формирования битов START/STOP на шине I^2C

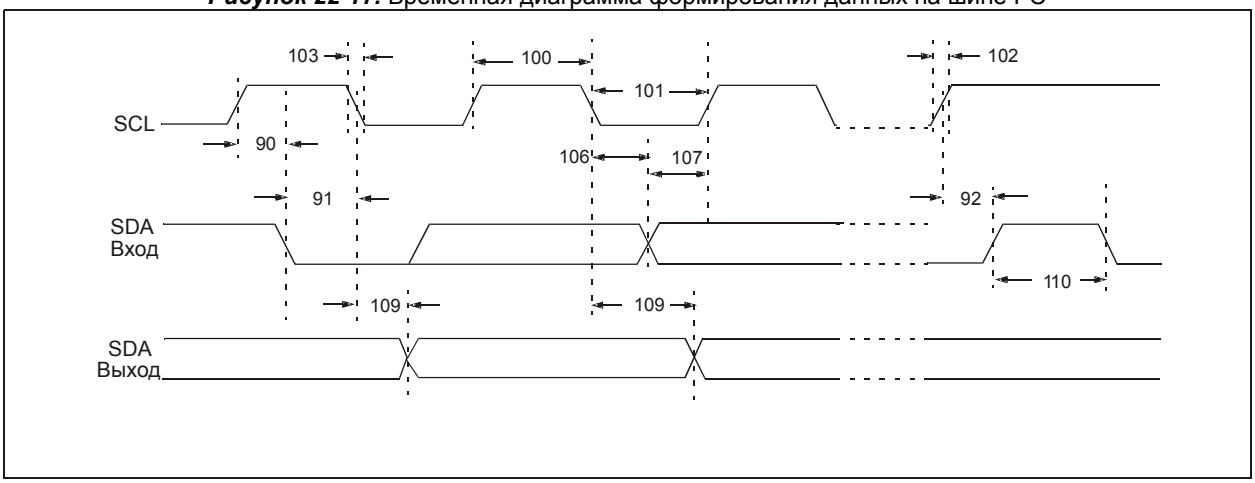


Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-15. Параметры формирования битов START/STOP на шине I^2C (режим ведомого)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
90	Tsu:sta	Установка условия START	Режим 100 кГц	4700	-	нс	Только при формировании бита повторный START
			Режим 400 кГц	600	-		
91	Thd:sta	Удержание условия START	Режим 100 кГц	4000	-	нс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	600	-		
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4700	-	нс	
			Режим 400 кГц	600			
93	Thd:sto	Удержание условия STOP	Режим 100 кГц	4000	-	нс	
			Режим 400 кГц	600	-		

Рисунок 22-17. Временная диаграмма формирования данных на шине I^2C



Примечание. Условие нагрузки показано на рисунке 22-4.

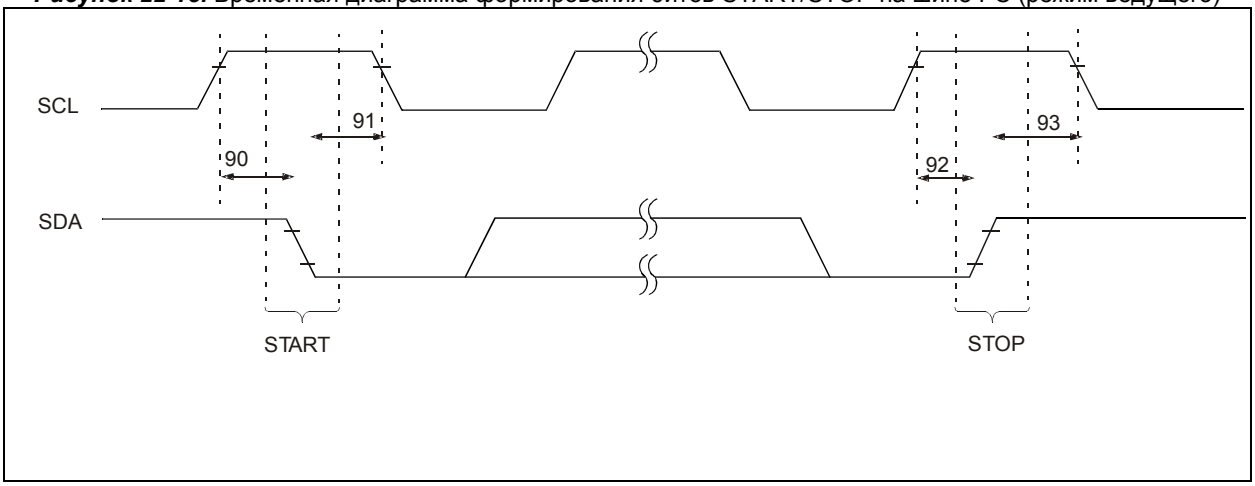
Таблица 22-16. Параметры формирования бита данных на шине I^2C (режим ведомого)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
100	Thigh	Длительность высокого уровня тактового сигнала	Режим 100 кГц	4.0	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	0.6	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
101	Tlow	Длительность низкого уровня тактового сигнала	Режим 100 кГц	4.7	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	1.3	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
102	Tr	Долит. переднего фронта на SDA и SCL	Режим 100 кГц	-	1000	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
103	Tf	Долит. заднего фронта на SDA и SCL	Режим 100 кГц	-	300	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
90	Tsu:sta	Установка условия START	Режим 100 кГц	4.7	-	мкс	Только при формировании бита повторный START
			Режим 400 кГц	0.6	-	мкс	
91	Thd:sta	Удержание условия START	Режим 100 кГц	4.0	-	мкс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	0.6	-	мкс	
106	Thd:dat	Удержание данных на входе	Режим 100 кГц	0	-	нс	
			Режим 400 кГц	0	0.9	мкс	
107	Tsu:dat	Установка данных на входе	Режим 100 кГц	250	-	нс	Примечание 2
			Режим 400 кГц	100	-	нс	
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4.7	-	мкс	
			Режим 400 кГц	0.6	-	мкс	
109	Taa	Достоверность сигнала на выходе	Режим 100 кГц	-	3500	нс	Примечание 1
			Режим 400 кГц	-	-	нс	
110	Tbuf	Время не занятости шины	Режим 100 кГц	4.7	-	мкс	Задержка перед новой передачей
			Режим 400 кГц	1.3	-	мкс	
	Cb	Емкостная нагрузка линии		-	400	пФ	

Примечания:

1. Необходимо выдерживать эту минимальную задержку относительно заднего фронта SCL, чтобы избежать ложное формирование битов START и STOP.
2. Устройства с высокоскоростным режимом обмена (400кГц) могут использоваться в стандартном режиме (100кГц), но требование Tsu:dat ≥ 250нс необходимо выполнять. Это условие автоматически будет выполняться, если не возникает удержания линии SCL в низком логическом уровне. Если возникает удержание линии SCL в низком логическом уровне, то необходимо сформировать бит данных на SDA Tr.max + Tsu:dat = 1000 + 250 = 1250 нс (согласно спецификации I^2C) прежде, чем SCL будет "отпущена".

Рисунок 22-18. Временная диаграмма формирования битов START/STOP на шине I^2C (режим ведущего)



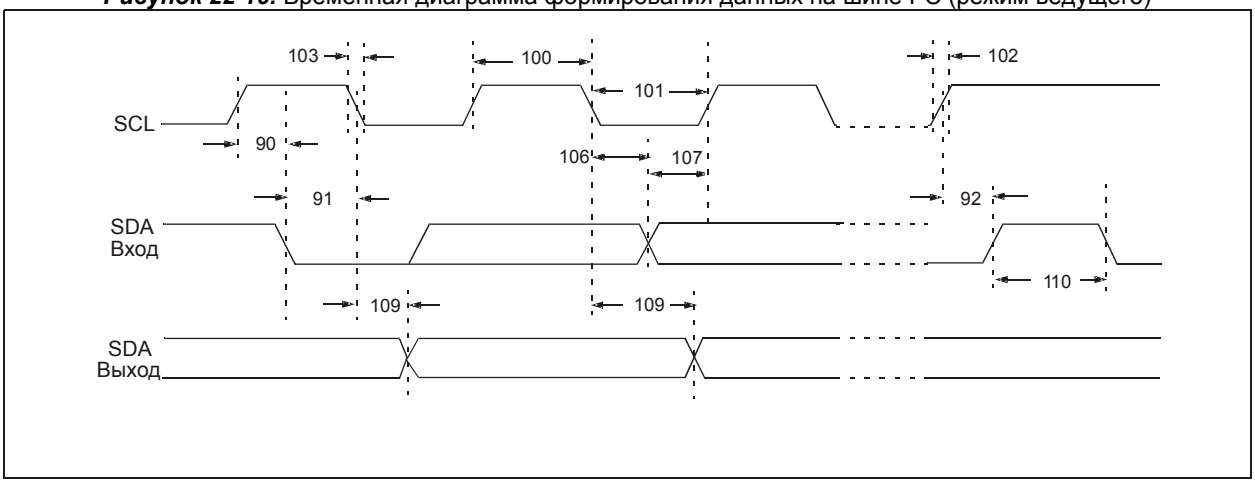
Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-17. Параметры формирования битов START/STOP на шине I^2C (режим ведущего)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
90	Tsu:sta	Установка условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	нс	Только при формировании бита повторный START
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-		
91	Thd:sta	Удержание условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	нс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-		
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	нс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$			
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-		
93	Thd:sto	Удержание условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	нс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-		

Примечание 1. Максимальная емкость вывода 10пФ (для всех выводов I^2C).

Рисунок 22-19. Временная диаграмма формирования данных на шине I^2C (режим ведущего)



Примечание. Условие нагрузки показано на рисунке 22-4.

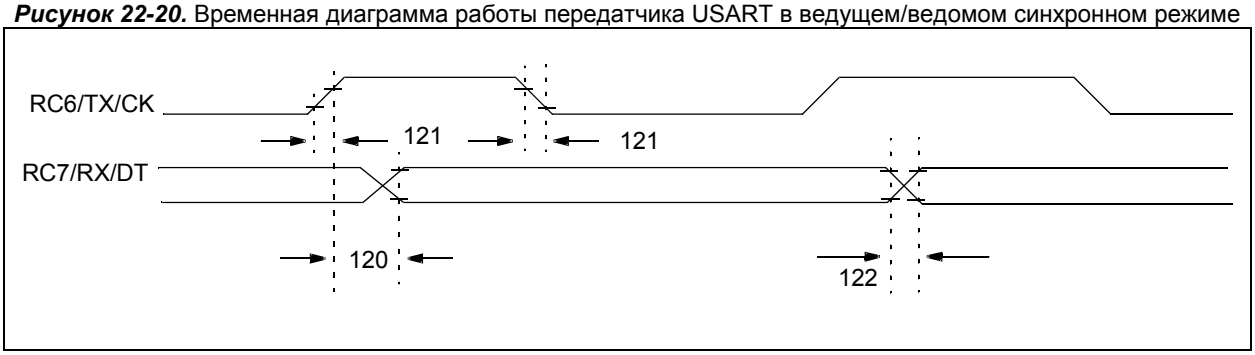
Таблица 22-18. Параметры формирования бита данных на шине I^2C (режим ведущего)

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
100	Thigh	Длительность высокого уровня тактового сигнала	Режим 100 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)$	-	мкс	
101	Tlow	Длительность низкого уровня тактового сигнала	Режим 100 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)$	-	мкс	
102	Tr	Долит. переднего фронта на SDA и SCL	Режим 100 кГц	-	1000	нс	10пФ ≤ Cb ≤ 400пФ
			Режим 400 кГц	20 + 0.1 Cb	300	нс	
			Режим 1 МГц[1]		300	нс	
103	Tf	Долит. заднего фронта на SDA и SCL	Режим 100 кГц	-	300	нс	10пФ ≤ Cb ≤ 400пФ
			Режим 400 кГц	20 + 0.1 Cb	300	нс	
			Режим 1 МГц[1]		100	нс	
90	Tsu:sta	Установка условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	Только при формировании бита повторный START
			Режим 400 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)$	-	мкс	
91	Thd:sta	Удержание условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)$	-	мкс	
106	Thd:dat	Удержание данных на входе	Режим 100 кГц	0	-	нс	
			Режим 400 кГц	0	0.9	мкс	
			Режим 1 МГц[1]	TBD	-	нс	
107	Tsu:dat	Установка данных на входе	Режим 100 кГц	250	-	нс	Примечание 2
			Режим 400 кГц	100	-	нс	
			Режим 1 МГц[1]	TBD	-	нс	
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)$	-	мкс	
109	Taa	Достоверность сигнала на выходе	Режим 100 кГц	-	3500	нс	
			Режим 400 кГц	-	1000	нс	
			Режим 1 МГц[1]	-	-	нс	
110	Tbuf	Время не занятости шины	Режим 100 кГц	4.7	-	мкс	Задержка перед новой передачей
			Режим 400 кГц	1.3	-	мкс	
			Режим 1 МГц[1]	TBD	-	мкс	
	Cb	Емкостная нагрузка линии		-	400	пФ	

Примечания:

1. Максимальная емкость вывода 10пФ (для всех выводов I^2C).
2. Устройства с высокоскоростным режимом обмена (400кГц) могут использоваться в стандартном режиме (100кГц), но требование Tsu:dat ≥ 250нс необходимо выполнять. Это условие автоматически будет выполняться, если не возникает удержания линии SCL в низком логическом уровне. Если возникает удержание линии SCL в низком логическом уровне, то необходимо сформировать бит данных на SDA Tr.max + Tsu:dat = 1000 + 250 = 1250 нс (согласно спецификации I2C) прежде, чем SCL будет "отпущена".

Рисунок 22-20. Временная диаграмма работы передатчика USART в ведущем/ведомом синхронном режиме

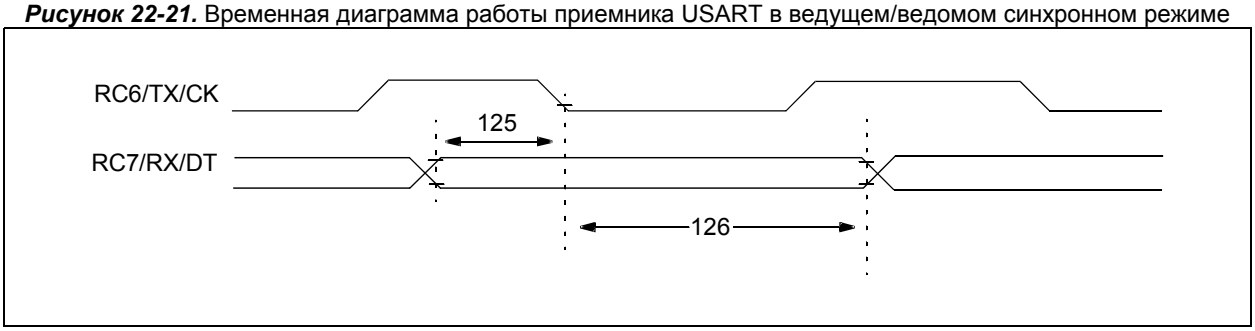


Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-19. Параметры работы передатчика USART в ведущем/ведомом синхронном режиме

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
120	TckH2dtV	Действ. данные после перехода такт. сигнала в высокий уровень	**F**	-	40	нс	
			LF	-	100	нс	
121	Tckrf	Длительность заднего/переднего фронта такт. сигн. (ведущий)	**F**	-	20	нс	
			LF	-	50	нс	
122	Tdtrf	Длительность переднего/заднего фронта данных	**F**	-	20	нс	
			LF	-	50	нс	

Рисунок 22-21. Временная диаграмма работы приемника USART в ведущем/ведомом синхронном режиме



Примечание. Условие нагрузки показано на рисунке 22-4.

Таблица 22-20. Параметры работы приемника USART в ведущем/ведомом синхронном режиме

№ пар.	Обоз.	Описание	Мин.	Макс.	Ед.	Примечание
125	TdtV2ckL	Установка данных после CK↓	10	-	нс	
126	TckL2dl	Удержание данных после CK↓	15	-	нс	

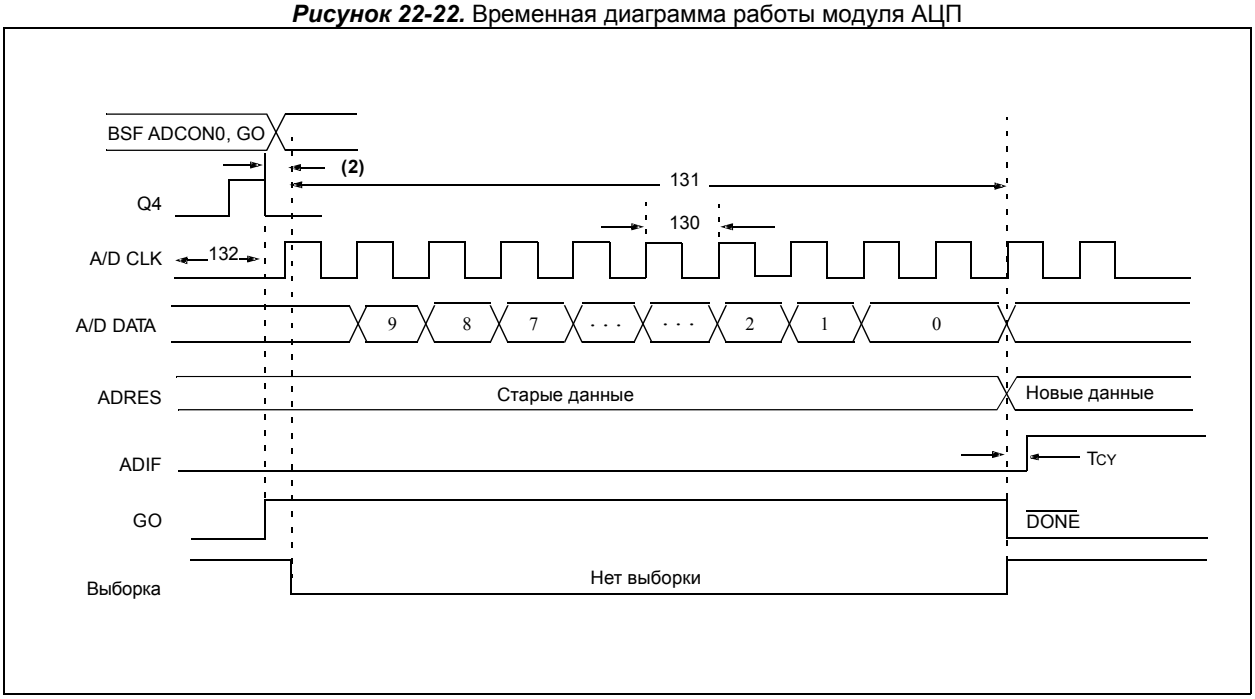
Таблица 22-21. Характеристика АЦП (PIC18FXX2-I, PIC18FXX2-E, PIC18LFXX2-I)

№ пар.	Обоз.	Описание		Мин.	Тип**	Макс.	Ед.	Примечание
A01	N_R	Разрядность		- -	- -	10 TBD	бит бит	$V_{REF} = V_{DD} \geq 3.0В$ $V_{REF} = V_{DD} < 3.0В$
A03	E_{IL}	Интегральная погрешность		- -	- -	$< \pm 1$ TBD	LSb LSb	$V_{REF} = V_{DD} \geq 3.0В$ $V_{REF} = V_{DD} < 3.0В$
A04	E_{DL}	Дифференциальная погрешность		- -	- -	$< \pm 1$ TBD	LSb LSb	$V_{REF} = V_{DD} \geq 3.0В$ $V_{REF} = V_{DD} < 3.0В$
A06	E_{OFF}	Ошибка смещения		- -	- -	$< \pm 2$ TBD	LSb LSb	$V_{REF} = V_{DD} \geq 3.0В$ $V_{REF} = V_{DD} < 3.0В$
A07	E_{GN}	Ошибка усиления		- -	- -	$< \pm 1$ TBD	LSb LSb	$V_{REF} = V_{DD} \geq 3.0В$ $V_{REF} = V_{DD} < 3.0В$
A10	-	Монотонность(3)		Гарантируется			-	$V_{SS} \leq V_{AIN} \leq V_{REF}$
A20 A20A	V_{REF}	Опорное напряжение (V_{REF+}-V_{REF-})		0 3	- -	- -	В В	 Минимальное значение для 10-разрядного АЦП
A21	V_{REF+}	Положительное опорное напр.		AV_{SS}		AV_{DD} + 0.3	В	
A22	V_{REF-}	Отрицательное опорное напр.		AV_{SS} – 0.3		AV_{DD}	В	
A25	V_{AIN}	Аналоговый вход		AV_{SS} – 0.3	-	V_{REF} + 0.3	В	
A30	Z_{AIN}	Сопротивление источника сигн.		-	-	10.0	кОм	
A40	I_{AD}	Потребляемый ток АЦП	**F**	-	180	-	мкА	Среднее потребление при включенном АЦП(1)
			LF	-	90	-	мкА	
A50	I_{REF}	Потребляемый ток от источника опорного напряжения(2)		10 -	- -	1000 10	мкА мкА	Во время выборки V_{AIN}. Основано на дифференц. значении заряда C_{HOLD} до V_{AIN}. Во время преобразования.

Примечания:

1. Выключенный модуль АЦП не потребляет тока, кроме токов утечки.
2. $V_{SS} \leq V_{AIN} \leq V_{REF}$.
3. Результат АЦП никогда не уменьшается с увеличением напряжения на входе и не имеет кодов отсутствия напряжения.

Рисунок 22-22. Временная диаграмма работы модуля АЦП



Примечание 1. Если используется внутренний RC генератор для АЦП, то добавляется время T_{CY} перед запуском АЦП, позволяющее выполнить команду SLEEP.

Примечание 2. Минимальная задержка RC цепочки (номинальное значение 100нс) включая отсоединение внутреннего конденсатора C_{HOLD} от аналогового входа.

Таблица 22-22. Параметры работы модуля АЦП

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
130	T_{AD}	Период тактового сигнала АЦП	**F**	1.6	20	мкс	Основа T_{OSC}, $V_{REF} \geq 3.0$ В
			LF	3.0	20	мкс	Основа T_{OSC}, $V_{REF} \geq 2.0$ В
			F	2.0	6.0	мкс	RC генератор АЦП
			LF	3.0	9.0	мкс	RC генератор АЦП
131	T_{CNV}	Время преобразования (1)		11	12*	T_{AD}	
132	T_{ACQ}	Время выборки(3)		15 10	- -	мкс мкс	от -40°C до +125°C от 0°C до +125°C
135	T_{SWC}	Время переключения от преобразования к выборке		-	-		Примечание 4
136	T_{AMP}	Время реакции усилителя		1	-	мкс	Примечание 5

Примечания:

1. Регистр ADRES может быть прочитан в следующем цикле.
2. Смотрите раздел "10 - разрядное АЦП" для выбора минимального значения.
3. Время заряда конденсатора C_{HOLD} до входного напряжения, когда изменение напряжения соответствует полной шкале (переход от AV_{DD} к AV_{SS} или от AV_{SS} к AV_{DD}).
4. В следующем цикле на такте Q4.
5. Минимальное время - задержка усилителя. Может использоваться, если напряжение на входе изменилось не более, чем на 1 LSb (т.е. 20мВ @ 5.12В) от последнего измерения.

23. Характеристика микроконтроллеров

Представленные графики и таблицы предназначены для оценки проекта и не проверяются (не гарантируются).

Графики в этом разделе не проверены и предназначены только для оценки при разработке устройств. В некоторых графиках представлены данные вне рабочего диапазона (в частности для напряжения питания V_{DD}). Это только информационные данные.

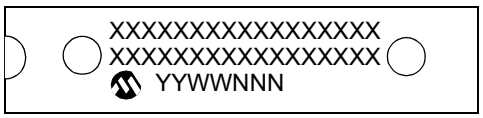
Данные, представленные в этом разделе, являются среднестатистическим результатом испытаний большого числа микроконтроллеров в течение длительного времени. Типовое значение подразумевает среднее (при температуре +25°C), а минимальное и максимальное - соответственно (среднее - 3σ) и (среднее + 3σ), где σ - стандартный разброс.

На момент подготовки перевода информация отсутствовала в оригинальной документации.

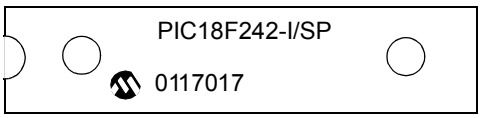
24. Корпуса микроконтроллеров

24.1 Описание обозначений на корпусах микроконтроллеров

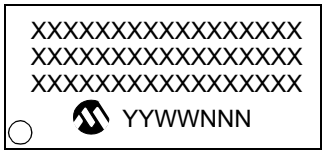
28 - выводный PDIP (тонкий DIP)



Пример



28 -выводный SOIC



Пример

Обозначения:

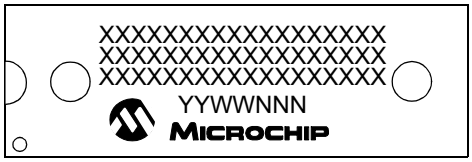
XX..X	Тип микроконтроллера*
YY	Две цифры года изготовления
WW	Две цифры номера недели изготовления считая с 1 января.
NNN	Алфавитно-цифровой код

Примечание. Если тип микроконтроллера не помещается в одну строку, то он будет перемещен на другую строку, ограничивая число доступных символов для информации заказчика.

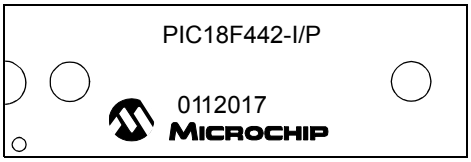
* Стандартная маркировка OTP микросхем состоит из: типа микроконтроллера, код года, код недели, код завода изготовителя, код упаковщика кристалла в корпус. Изменение маркировки микросхемы выполняется за отдельную плату. Для QTP микроконтроллеров стоимость маркировки входит в цену микросхем QTP.

Описание обозначений на корпусах микроконтроллеров (продолжение)

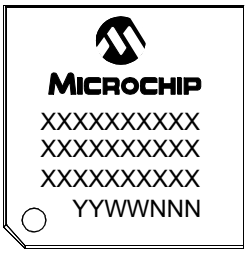
40-выводный PDIP



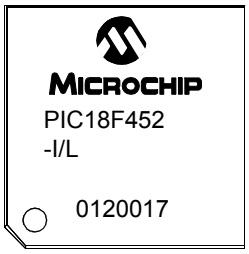
Пример



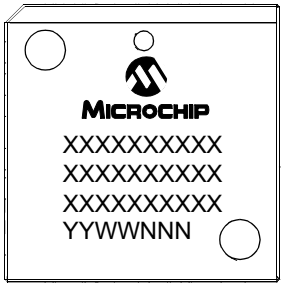
44-выводный TQFP



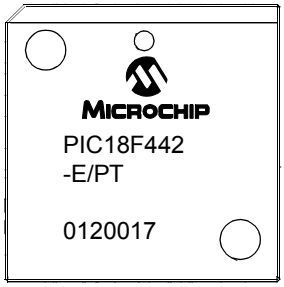
Пример



44-выводный PLCC

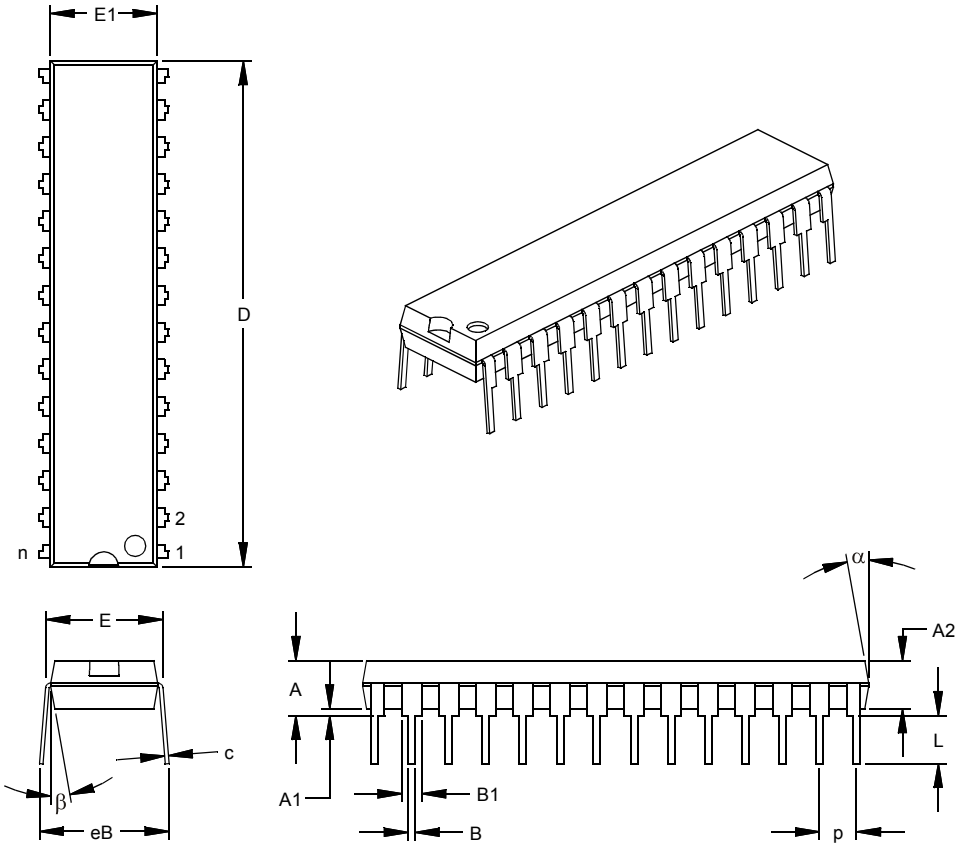


Пример



24.2 Чертежи корпусов

24.2.1 Тип корпуса: 28-выводный PDIP - 300mil



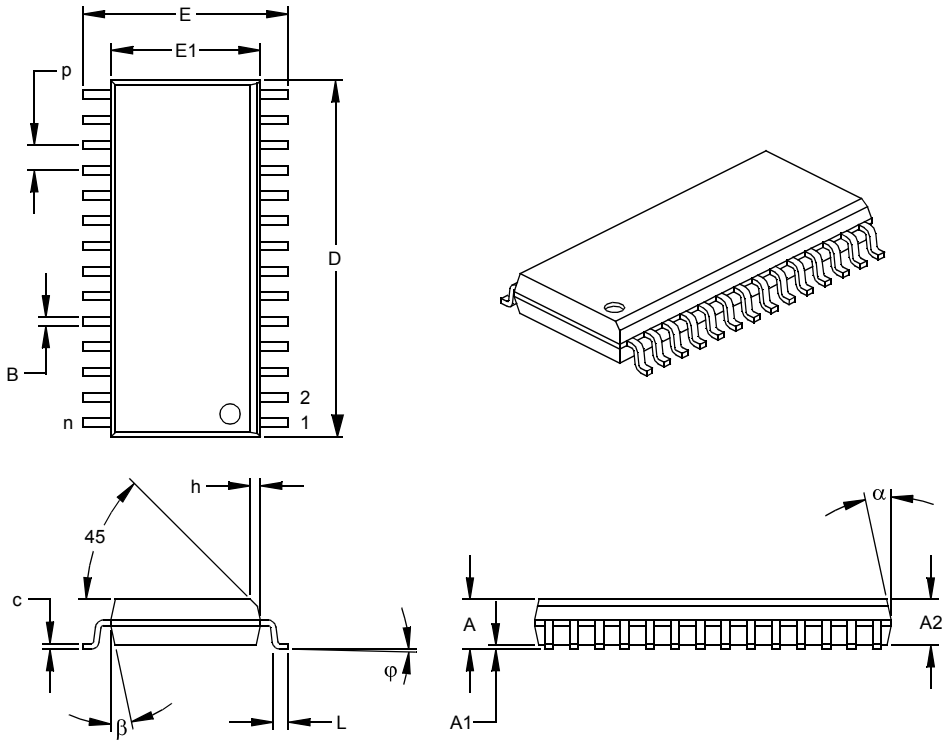
Единицы измерения		Дюймы*			Миллиметры		
Пределы размеров		Мин.	Ном.	Макс.	Мин.	Ном.	Макс.
Число выводов	n		28			28	
Расстояние между выводами	p		0.100			2.54	
Высота корпуса	A	0.140	0.150	0.160	3.56	3.81	4.06
Толщина корпуса	A2	0.125	0.130	0.135	3.18	3.30	3.43
Расстояние между корпусом и платой	A1	0.015			0.38		
Ширина корпуса с выводами	E	0.300	0.310	0.352	7.62	7.87	8.26
Ширина корпуса	E1	0.275	0.285	0.295	6.99	7.24	7.49
Длина корпуса	D	1.345	1.365	1.385	34.16	34.67	35.18
Длина нижней части вывода	L	0.125	0.130	0.135	3.18	3.30	3.43
Толщина вывода	c	0.008	0.012	0.015	0.20	0.29	0.38
Ширина верхней части вывода	B1	0.040	0.053	0.065	1.02	1.33	1.65
Ширина нижней части вывода	B	0.016	0.019	0.022	0.41	0.48	0.56
Полная ширина корпуса с выводами	eB	0.320	0.350	0.430	8.13	8.89	10.92
Угол фаски верхней части корпуса	α	5	10	15	5	10	15
Угол фаски нижней части корпуса	β	5	10	15	5	10	15

* Основные размеры.

Эквивалент JEDEC: MO-095

Примечание. Параметры D и E1 не включают выступы. Выступы в сторону не должны превышать 0.010"(0.254мм).

24.2.2 Тип корпуса: 28-выводный SOIC - 300mil



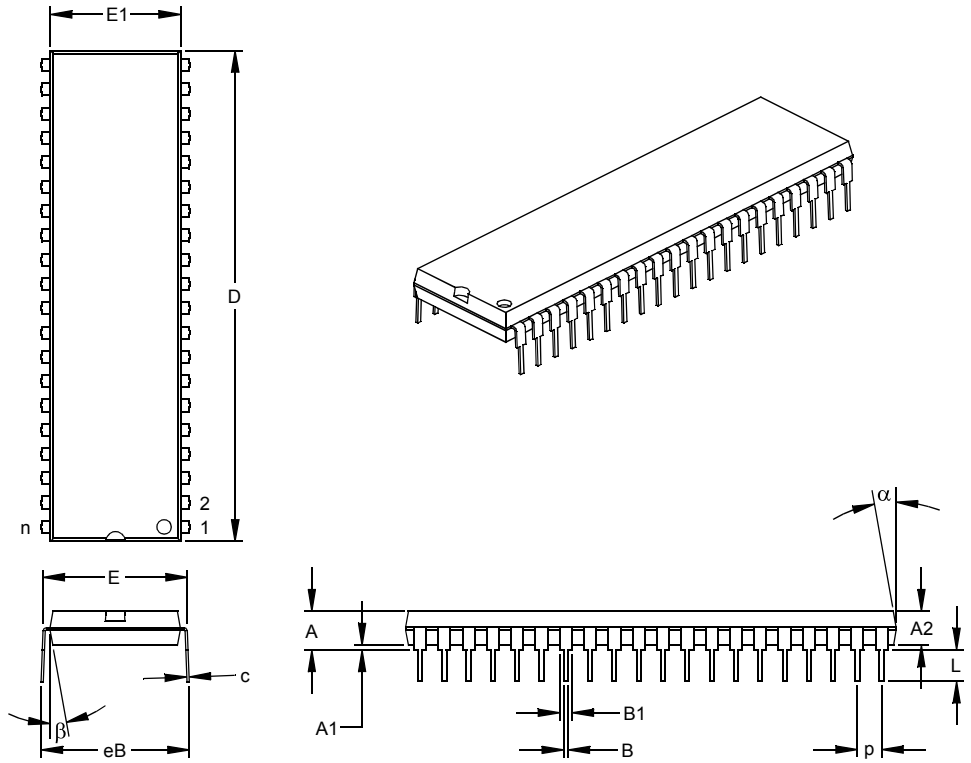
Единицы измерения		Дюймы*			Миллиметры		
Пределы размеров		Мин.	Ном.	Макс.	Мин.	Ном.	Макс.
Число выводов	n		28			28	
Расстояние между выводами	p		0.050			1.27	
Толщина корпуса с выводами	A	0.093	0.099	0.104	2.36	2.50	2.64
Толщина корпуса	A2	0.088	0.091	0.094	2.24	2.31	2.39
Расстояние между корпусом и платой	A1	0.004	0.008	0.012	0.10	0.20	0.30
Ширина корпуса с выводами	E	0.394	0.407	0.420	10.01	10.34	10.67
Ширина корпуса	E1	0.288	0.295	0.299	7.32	7.49	7.59
Длина корпуса	D	0.695	0.704	0.712	17.65	17.87	18.08
Размер ориентирующей фаски	h	0.010	0.020	0.029	0.25	0.50	0.74
Длина нижней части вывода	L	0.016	0.033	0.050	0.41	0.84	1.27
Угол наклона нижней части вывода	φ	0	4	8	0	4	8
Толщина вывода	c	0.009	0.011	0.013	0.23	0.28	0.33
Ширина вывода	B	0.014	0.017	0.20	0.36	0.42	0.51
Угол фаски верхней части корпуса	α	0	12	15	0	12	15
Угол фаски нижней части корпуса	β	0	12	15	0	12	15

* Основные размеры.

Эквивалент JEDEC: MS-013

Примечание. Параметры D и E1 не включают выступы. Выступы в сторону не должны превышать 0.010"(0.254мм).

24.2.3 Тип корпуса: 40-выводный PDIP - 600mil



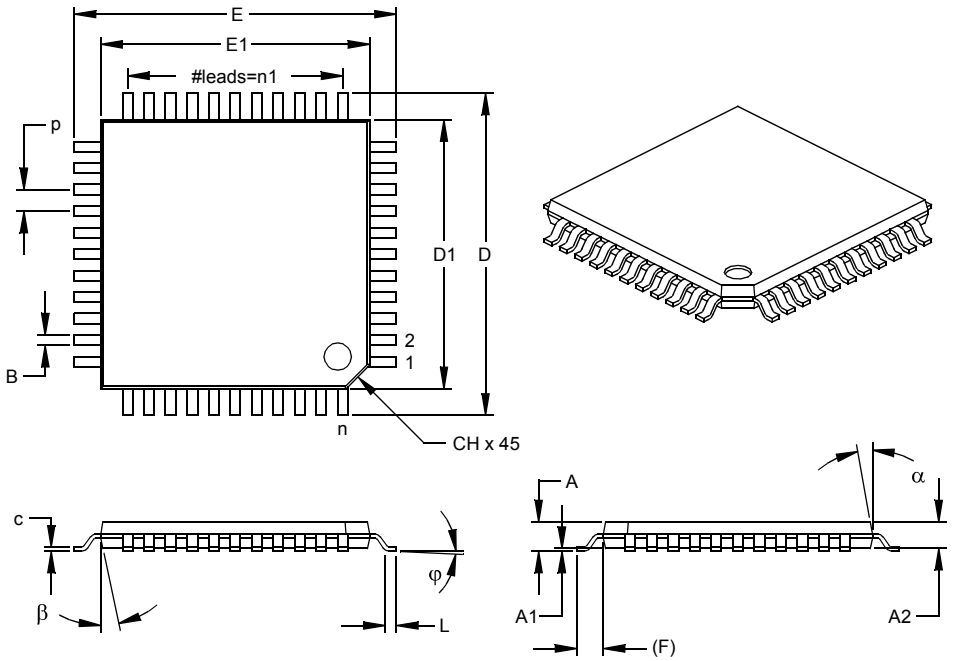
Единицы измерения		Дюймы*			Миллиметры		
Пределы размеров		Мин.	Ном.	Макс.	Мин.	Ном.	Макс.
Число выводов	n		40			40	
Расстояние между выводами	p		0.100			2.54	
Высота корпуса	A	0.160	0.175	0.190	4.06	4.45	4.83
Толщина корпуса	A2	0.140	0.150	0.160	3.56	3.81	4.06
Расстояние между корпусом и платой	A1	0.015			0.38		
Ширина корпуса с выводами	E	0.595	0.600	0.625	15.11	15.24	15.88
Ширина корпуса	E1	0.530	0.545	0.560	13.46	13.84	14.22
Длина корпуса	D	2.045	2.058	2.065	51.94	52.26	52.45
Длина нижней части вывода	L	0.120	0.130	0.135	3.05	3.30	3.43
Толщина вывода	c	0.008	0.012	0.015	0.20	0.29	0.38
Ширина верхней части вывода	B1	0.030	0.050	0.070	0.76	1.27	1.78
Ширина нижней части вывода	B	0.014	0.018	0.022	0.36	0.46	0.56
Полная ширина корпуса с выводами	eB	0.620	0.650	0.680	15.75	16.51	17.27
Угол фаски верхней части корпуса	α	5	10	15	5	10	15
Угол фаски нижней части корпуса	β	5	10	15	5	10	15

* Основные размеры.

Эквивалент JEDEC: MO-011

Примечание. Параметры D и E1 не включают выступы. Выступы в сторону не должны превышать 0.010"(0.254мм).

24.2.4 Тип корпуса: 44-выводный TQFP



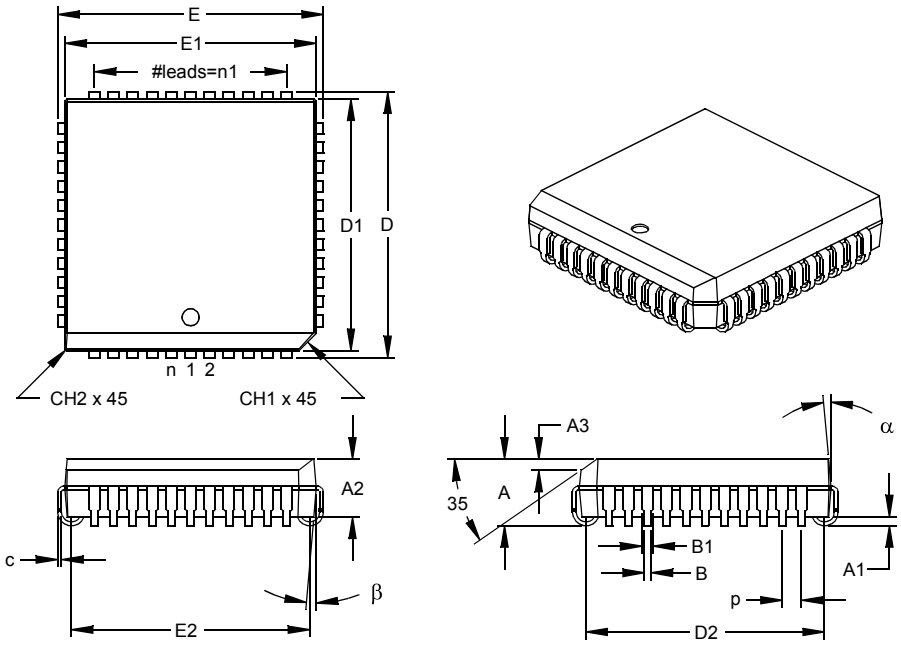
Единицы измерения		Дюймы			Миллиметры*		
Пределы размеров		Мин.	Ном.	Макс.	Мин.	Ном.	Макс.
Число выводов	n		44			44	
Расстояние между выводами	p		0.031			0.80	
Число выводов с одной стороны	n1		11			11	
Толщина корпуса с выводами	A	0.039	0.043	0.047	1.00	1.10	1.20
Толщина корпуса	A2	0.037	0.039	0.041	0.95	1.00	1.05
Расстояние между корпусом и платой	A1	0.002	0.004	0.006	0.05	0.10	0.15
Длина нижней части вывода	L	0.018	0.024	0.030	0.45	0.60	0.75
Длина вывода	(F)		0.039		1.00		
Угол наклона нижней части вывода	φ	0	3.5	7	0	3.5	7
Ширина корпуса с выводами	E	0.463	0.472	0.482	11.75	12.00	12.25
Длина корпуса с выводами	D	0.463	0.472	0.482	11.75	12.00	12.25
Ширина корпуса	E1	0.390	0.394	0.398	9.90	10.00	10.10
Длина корпуса	D1	0.390	0.394	0.398	9.90	10.00	10.10
Толщина вывода	c	0.004	0.006	0.008	0.09	0.15	0.20
Ширина вывода	B	0.012	0.015	0.017	0.30	0.38	0.44
Размер ориентирующей фаски	CH	0.025	0.035	0.045	0.64	0.89	1.14
Угол фаски верхней части корпуса	α	5	10	15	5	10	15
Угол фаски нижней части корпуса	β	5	10	15	5	10	15

* Основные размеры.

Эквивалент JEDEC: MS-026

Примечание. Параметры D и E1 не включают выступы. Выступы в сторону не должны превышать 0.010"(0.254мм).

24.2.5 Тип корпуса: 44-выводный PLCC



Единицы измерения		Дюймы*			Миллиметры		
Пределы размеров		Мин.	Ном.	Макс.	Мин.	Ном.	Макс.
Число выводов	n		44			44	
Расстояние между выводами	p		0.050			1.27	
Число выводов с одной стороны	n1		11			11	
Толщина корпуса с выводами	A	0.165	0.173	0.180	4.19	4.39	4.57
Толщина корпуса	A2	0.145	0.153	0.160	3.68	3.87	4.06
Расстояние между корпусом и платой	A1	0.020	0.028	0.035	0.51	0.71	0.89
Высота ориентирующей фаски	A3	0.024	0.029	0.34	0.61	0.74	0.86
Размер ориентирующей фаски	CH1	0.040	0.045	0.050	1.02	1.14	1.27
Размер ориентирующей фаски	CH2	0.000	0.005	0.010	0.00	0.13	0.25
Ширина корпуса с выводами	E	0.685	0.690	0.695	17.40	17.53	17.65
Длина корпуса с выводами	D	0.685	0.690	0.695	17.40	17.53	17.65
Ширина корпуса	E1	0.650	0.653	0.656	16.51	16.59	16.66
Длина корпуса	D1	0.650	0.653	0.656	16.51	16.59	16.66
Расстояние между выводами	E2	0.590	0.620	0.630	14.99	15.75	16.00
Расстояние между выводами	D2	0.590	0.620	0.630	14.99	15.75	16.00
Толщина вывода	c	0.008	0.011	0.013	0.20	0.27	0.33
Ширина верхней части вывода	B1	0.026	0.029	0.032	0.66	0.74	0.81
Ширина нижней части вывода	B	0.013	0.020	0.021	0.33	0.51	0.53
Угол фаски верхней части корпуса	α	0	5	10	0	5	10
Угол фаски нижней части корпуса	β	0	5	10	0	5	10

* Основные размеры.

Эквивалент JEDEC: MO-047

Примечание. Параметры D и E1 не включают выступы. Выступы в сторону не должны превышать 0.010"(0.254мм).

24.3 Правила идентификации типа микроконтроллеров PIC18FXX2

Чтобы определить параметры микроконтроллеров воспользуйтесь ниже описанным правилом.

PART№	**X**	**/XX**	**XXX**
Микроконтроллер	**Температурный диапазон**	**Корпус**	**Образец**

Микроконтроллер

PIC18FXX2[(1)], PIC18FXX2T[(2)], 4.2В ≤ V_{DD} ≤ 5.5В
PIC18LFXX2[(1)], PIC18LFXX2T[(2)], 2.5В ≤ V_{DD} ≤ 5.5В

Температурный диапазон

I = от -40°C до +85°C
E = от -40°C до +125°C

Корпус

PT = TQFP
SO = SOIC
SP = тонкий PDIP
P = PDIP
L = PLCC

Пример

1. **PIC18LF452-I/P 301** = промышленный температурный диапазон, корпус PDIP, расширенный диапазон напряжения питания, код QTP 301.
2. **PIC18LF242-I/SO** = промышленный температурный диапазон, корпус SOIC, расширенный диапазон напряжения питания.
3. **PIC18F442-E/P** = расширенный температурный диапазон, корпус PDIP, нормальный диапазон напряжения питания.

Примечания:

1. F = CMOS FLASH; LF = CMOS FLASH с расширенным напряжением питания.
2. T = для работы в условиях вибрации, только корпуса SOIC, PLCC, MQFP, TQFP.

Уважаемые господа!

ООО «Микро-Чип» поставляет полную номенклатуру комплектующих фирмы **Microchip Technology Inc** и осуществляет качественную техническую поддержку на русском языке.

С техническими вопросами Вы можете обращаться по адресу support@microchip.ru

По вопросам поставок комплектующих Вы можете обращаться к нам по телефонам:
(095) 963-9601
(095) 737-7545
и адресу sales@microchip.ru

На сайте
www.microchip.ru
Вы можете узнать последние новости нашей фирмы, найти техническую документацию и информацию по наличию комплектующих на складе.