PIC18FXX2

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

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

Часть 2
(Организация памяти)

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

Содержание

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

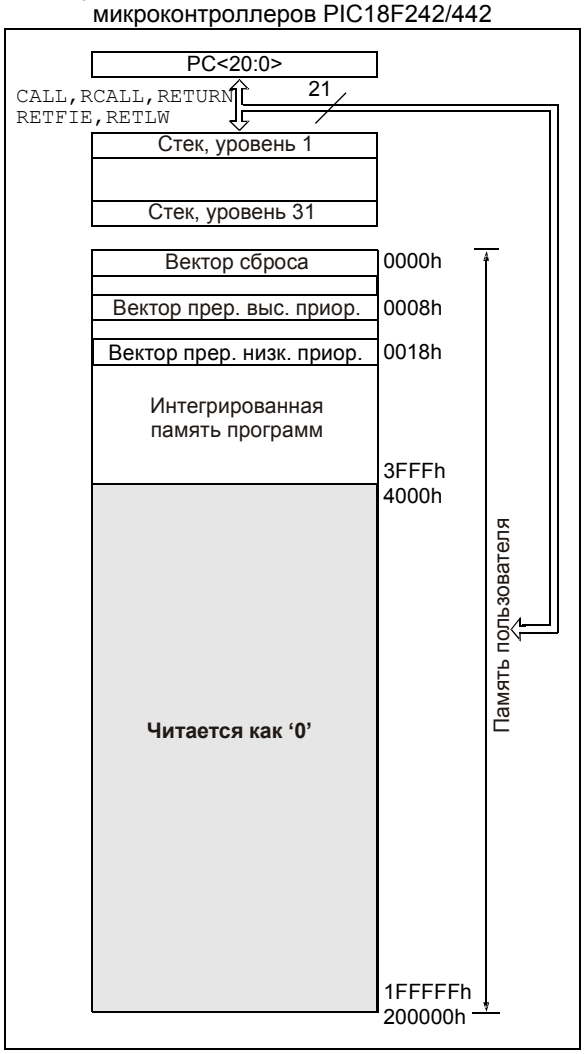


Рисунок 4-2. Карта памяти программ и стека микроконтроллеров PIC18F252/452

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

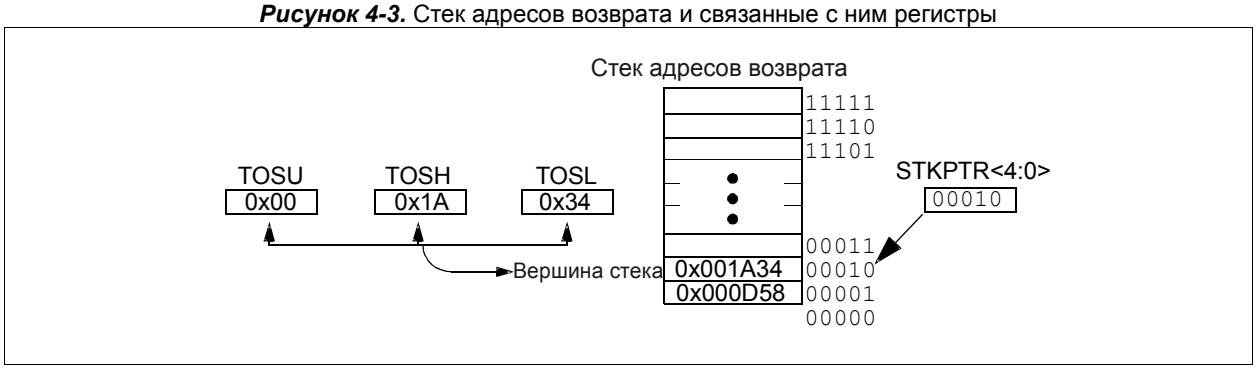
Бит 5 **Не используется:** Читается как ‘0’

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.

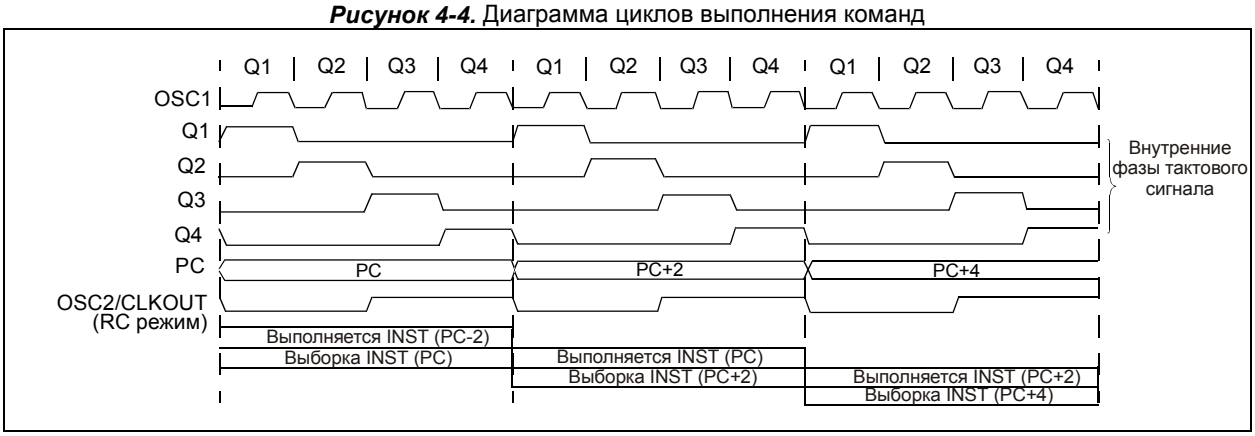
Команды CALL, RCALL, GOTO и команды возврата вызывают непосредственную запись в счетчик команд (значение регистров PCLATH, PCLATU не передается в счетчик команд).

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



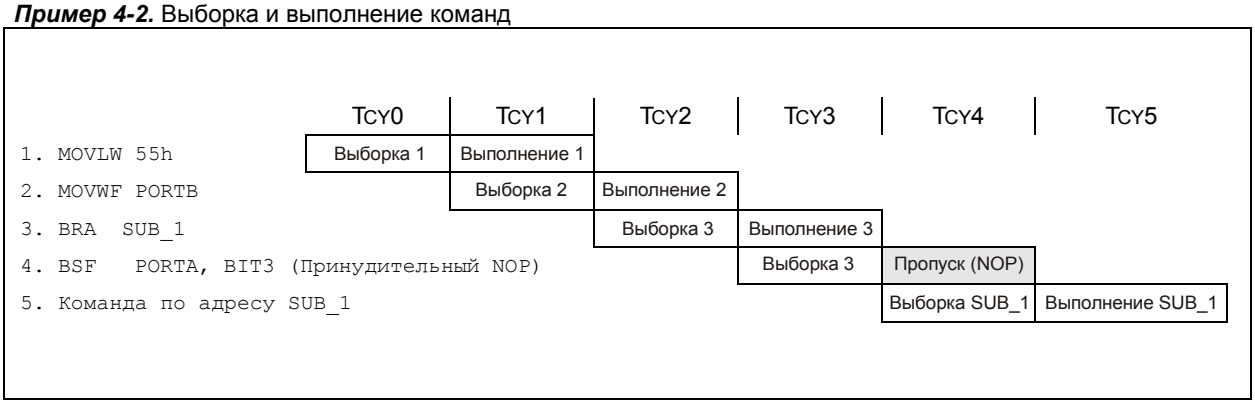
4.6 Конвейерная выборка и выполнение команд

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

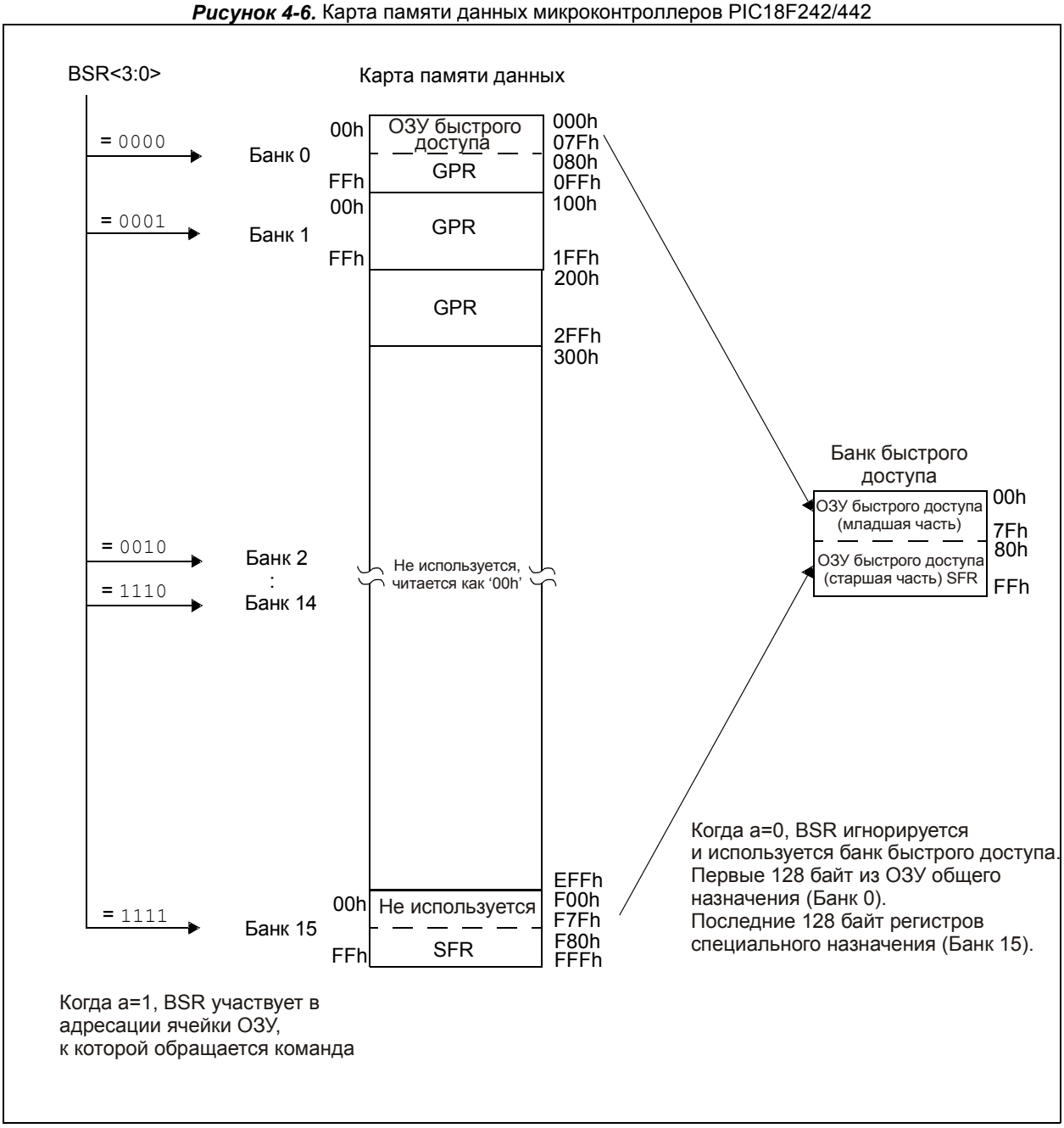


Рисунок 4-7 Карта памяти данных микроконтроллеров PIC18F252/452

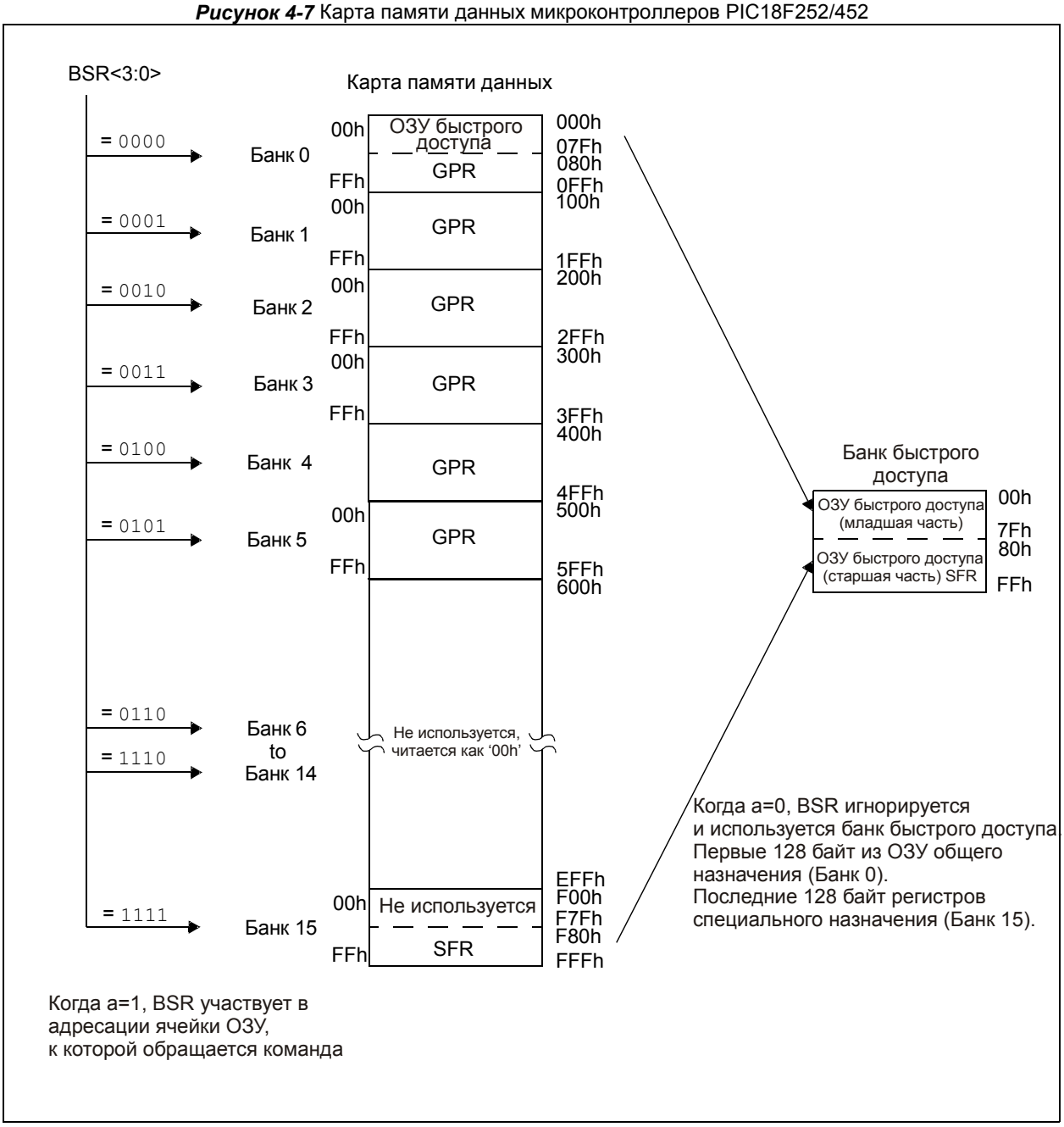


Таблица 4-1. Карта памяти регистров специального назначения

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
TSU	-	-	-	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 FSR0 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 FSR0 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.

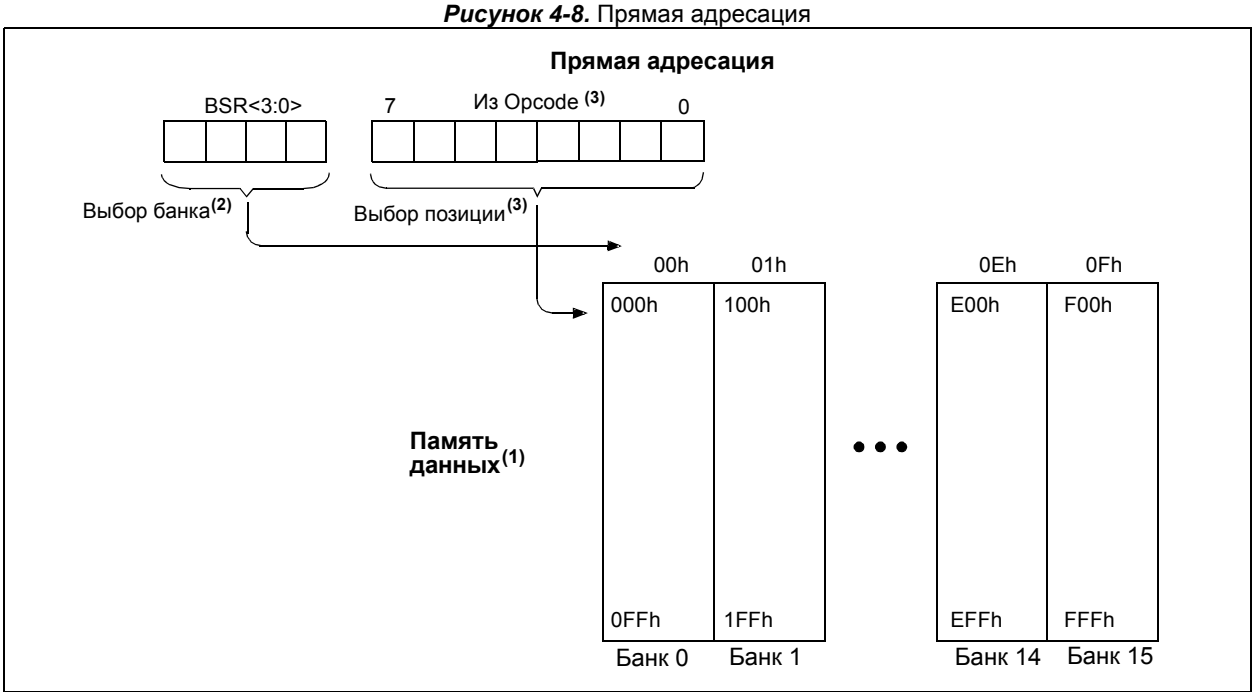
Если выбранный банк не реализован, то чтение будет давать результат ‘0’, а любая запись игнорируется. Биты регистра STATUS будут изменяться в соответствии с выполняемой командой.

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

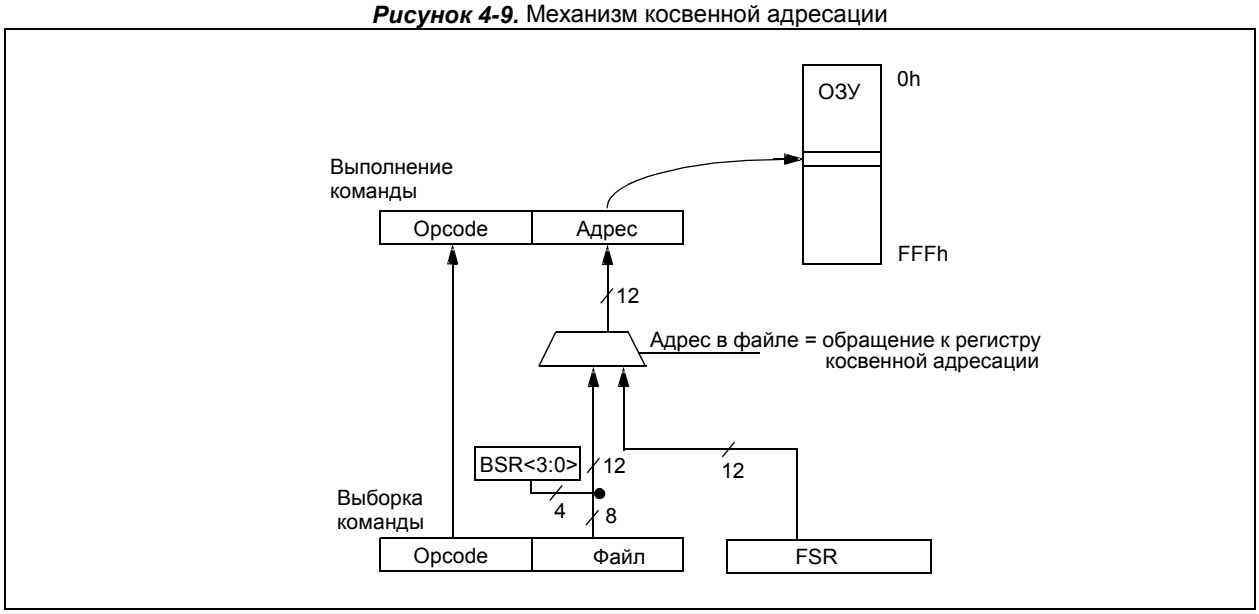
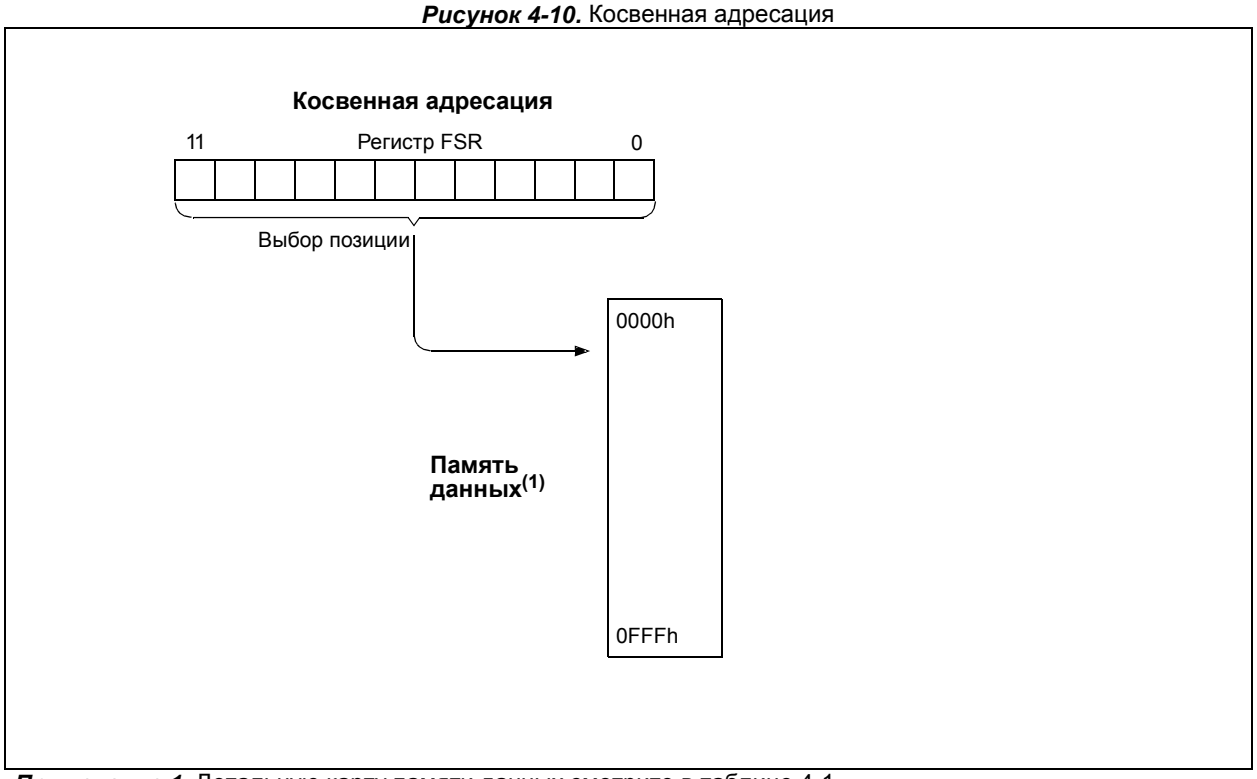


Рисунок 4-10. Косвенная адресация



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.

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.