Справочник по среднему семейству микроконтроллеров PICmicro™

Раздел 29. Система команд

Перевод основывается на технической документации DS33023A
компании Microchip Technology Incorporated, USA.

тел. (095) 737-7545
www.microchip.ru

PICmicro™ Mid-Range MCU Family Reference Manual

Содержание

29.1 Введение

Каждая команда состоит из одного 14 - разрядного слова, разделенного на код операции (OPCODE), определяющий тип команды и один или несколько операндов, определяющие операцию команды. Полный список команд смотрите в таблице 29-1.

Система команд аккумуляторного типа, ортогональна и разделена на три основных группы:

- Байт ориентированные команды;
- Бит ориентированные команды;
- Команды управления и операций с константами.

Описание полей кода операции смотрите в таблице 29-2.

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.

В бит ориентированных командах 'b' определяет номер бита участвующего в операции, а 'f' - указатель регистра, который содержит этот бит.

В командах управления или операциях с константами 'k' представляет восемь или одиннадцать бит константы или значения литералов.

Все команды выполняются за один машинный цикл, кроме команд условия, в которых получен истинный результат и инструкций изменяющих значение счетчика команд PC. В случае выполнения команды за два машинных цикла, во втором цикле выполняется инструкция NOP. Один машинный цикл состоит из четырех тактов генератора. Для тактового генератора с частотой 4 МГц все команды выполняются за 1мкс, если условие истинно или изменяется счетчик команд PC, команда выполняется за 2мкс.

Таблица 29-1 Список команд микроконтроллеров среднего семейства

Мнемоника команды		Описание	Циклов	14-разрядный код		Изм. флаги	Прим.
				Бит 13	Бит 0		
Байт ориентированные команды							
ADDWF	**f,d**	Сложение W и f	1	00 0111 dfff ffff		C,DC,Z	1,2
ANDWF	**f,d**	Побитное 'И' W и f	1	00 0101 dfff ffff		Z	1,2
CLRF	**f**	Очистить f	1	00 0001 1fff ffff		Z	2
CLRW	**-**	Очистить W	1	00 0001 0xxx xxxx		Z	
COMF	**f,d**	Инвертировать f	1	00 1001 dfff ffff		Z	1,2
DECF	**f,d**	Вычесть 1 из f	1	00 0011 dfff ffff		Z	1,2
DECFSZ	**f,d**	Вычесть 1 из f и пропустить если 0	1(2)	00 1011 dfff ffff			1,2,3
INCF	**f,d**	Прибавить 1 к f	1	00 1010 dfff ffff		Z	1,2
INCFSZ	**f,d**	Прибавить 1 к f и пропустить если 0	1(2)	00 1111 dfff ffff			1,2,3
IORWF	**f,d**	Побитное 'ИЛИ' W и f	1	00 0100 dfff ffff		Z	1,2
MOVF	**f,d**	Переслать f	1	00 1000 dfff ffff		Z	1,2
MOVWF	**f**	Переслать W в f	1	00 0000 1fff ffff			
NOP	**-**	Нет операции	1	00 0000 0xx0 0000			
RLF	**f,d**	Циклический сдвиг f влево через перенос	1	00 1101 dfff ffff		C	1,2
RRF	**f,d**	Циклический сдвиг f вправо через перенос	1	00 1100 dfff ffff		C	1,2
SUBWF	**f,d**	Вычесть W из f	1	00 0010 dfff ffff		C,DC,Z	1,2
SWAPF	**f,d**	Поменять местами полубайты в регистре f	1	00 1110 dfff ffff			1,2
XORWF	**f,d**	Побитное 'исключающее ИЛИ' W и f	1	00 0110 dfff ffff		Z	1,2
Бит ориентированные команды							
BCF	**f,b**	Очистить бит b в регистре f	1	01 00bb bfff ffff			1,2
BSF	**f,b**	Установить бит b в регистре f	1	01 01bb bfff ffff			1,2
BTFSC	**f,b**	Проверить бит b в регистре f, пропустить если 0	1(2)	01 10bb bfff ffff			3
BTFSS	**f,b**	Проверить бит b в регистре f, пропустить если 1	1(2)	01 11bb bfff ffff			3
Команды управления и операций с константами							
ADDLW	**k**	Сложить константу с W	1	11 111x kkkk kkkk		C,DC,Z	
ANDLW	**k**	Побитное 'И' константы и W	1	11 1001 kkkk kkkk		Z	
CALL	**k**	Вызов подпрограммы	2	10 0kkk kkkk kkkk			
CLRWDT	**-**	Очистить WDT	1	00 0000 0110 0100		-TO,-PD	
GOTO	**k**	Безусловный переход	2	10 1kkk kkkk kkkk			
IORLW	**k**	Побитное 'ИЛИ' константы и W	1	11 1000 kkkk kkkk		Z	
MOVLW	**k**	Переслать константу в W	1	11 00xx kkkk kkkk			
RETFIE	**-**	Возврат из подпрограммы с разрешением прерываний	2	00 0000 0000 1001			
RETLW	**k**	Возврат из подпрограммы с загрузкой константы в W	2	11 01xx kkkk kkkk			
RETURN	**-**	Возврат из подпрограммы	2	00 0000 0000 1000			
SLEEP	**-**	Перейти в режим SLEEP	1	00 0000 0110 0011		-TO,-PD	
SUBLW	**k**	Вычесть W из константы	1	11 110x kkkk kkkk		C,DC,Z	
XORLW	**k**	Побитное 'исключающее ИЛИ' константы и W	1	11 1010 kkkk kkkk		Z	

Примечания:

1. При выполнении операции "чтение - модификация - запись" с портом ввода/вывода (например MOVF PORTB,1) исходные значения считываются с выводов порта, а не из выходных защелок. Например, если в выходной защелке было записана '1', а на соответствующем выходе низкий уровень сигнала, то обратно будет записано значение '0'.
2. При выполнении записи в TMR0 (и d=1) предделитель TMR0 сбрасывается, если он подключен к модулю TMR0.
3. Если условие истинно или изменяется значение счетчика команд PC, то инструкция выполняется за два цикла. Во втором цикле выполняется команда NOP.

29.2 Формат команд

На рисунке 29-1 показано формат трех групп команд. Код команды может быть от 3 до 6 бит, что позволяет реализовать 35 команд.

Примечание 1. Любой не реализованный код операции сохранен для последующих разработок. Использование недокументированного кода операции может привести к непредсказуемым последствиям.

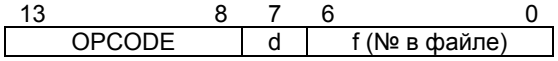
Примечание 2. Для совместимости программного обеспечения со следующими версиями микроконтроллеров PICmicro не используйте команды TRIS и OPTION.

Во всех примерах используется следующий формат шестнадцатеричных чисел:
0xhh, где h - шестнадцатеричная цифра.

Представление двоичного числа:
00000100b, где b - указатель двоичного числа.

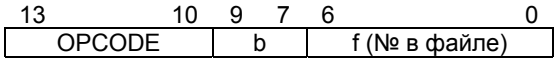
Рис. 29-1 Общий формат команд микроконтроллеров среднего семейства

Байт ориентированные операции с регистрами



d = 0 - результат сохраняется в w
d = 1 - результат сохраняется в f
f - 7-разрядный адрес регистра

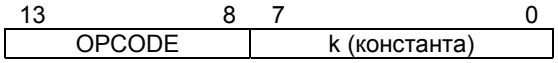
Бит ориентированные операции с регистрами



b - 3-разрядный номер бита в регистре
f - 7-разрядный адрес регистра

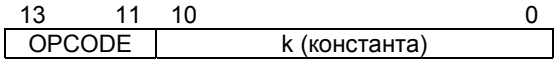
Команды управления и операций с константами

Общее



k - 8-разрядное значение

Только для инструкций CALL и GOTO



k - 11-разрядное значение

Таблица 29-2 Описание полей кода операции

Поле	Описание
f	Адрес регистра (от 0x00 до 0x7F)
w	Рабочий регистр (аккумулятор)
b	Номер бита в 8-разрядном регистре
k	Константа (данные или метка)
x	Не имеет значения (0 или 1). Ассемблер генерирует x=0 для совместимости программы микроконтроллера с инструментальными средствами
d	Указатель адресата результата операции: d = 0 - результат сохраняется в регистре w d = 1 - результат сохраняется в регистре f По умолчанию d = 1
label	Имя метки
TOS	Вершина стека
PC	Счетчик команд
PCLATH	Буфер старшего байта счетчика команд
GIE	Бит глобального разрешения прерываний
WDT	Сторожевой таймер
-TO	Флаг переполнения WDT
-PD	Флаг сброса по включению питания
dest	Приемник, регистр w или регистр памяти
[ ]	Дополнительные параметры
( )	Содержимое
→	Присвоение
< >	Битовое поле
∈	Из набора
Курсив	Термин, определяемый пользователем

29.3 Обращение к регистрам специального назначения

Система команд микроконтроллеров PICmicro среднего семейства позволяет напрямую обращаться к всем регистрам, включая регистры общего назначения. Существуют нюансы обращения к некоторым регистрам, которые должен учитывать разработчик программного обеспечения.

29.3.1 STATUS как регистр назначения при выполнении команды

Если обращение к регистру STATUS выполняется командой, которая воздействует на флаги Z, DC и C, то изменение этих трех битов командой заблокирована. Эти биты сбрасываются или устанавливаются согласно логике ядра микроконтроллера. Команды изменения регистра STATUS также не воздействуют на биты -TO и -PD. Поэтому результат выполнения команды с регистром STATUS может отличаться от ожидаемого. Например, команда CLRF STATUS сбросит три старших бита и установит бит Z (состояние регистра STATUS после выполнения команды 000uu1uu, где u - не изменяемый бит).

29.3.2 PCL как источник данных и регистр назначения при выполнении команды

Команды чтения, записи и "чтение - модификация - запись" регистра PCL могут иметь следующие результаты:

Чтение PCL:	PCL → dest; PCLATH не изменяется.
Запись PCL:	PCLATH → PCH; 8 - разрядное значение → PCL.
"Чтение - модификация - запись":	PCL → операнд АЛУ; PCLATH → PCH; 8 - разрядный результат → PCL.

Где PCH - старший байт счетчика команд (не адресуемый регистр), PCLATH - буфер старшего байта счетчика команд, dest - регистр назначения.

29.3.3 Битовые операции

При изменении любого бита регистра сначала регистр полностью читается из памяти данных, изменяется бит, обратно данные записываются в регистр ("чтение - модификация - запись"). Необходимо учитывать этот факт при обращении к некоторым регистрам специального назначения (например, регистры портов).

Примечание. Изменение состояния управляющих битов (включая флаги прерываний) выполняется в такте Q1, поэтому не возникает проблем при обращении командой "чтение - модификация - запись" к регистрам, содержащим эти биты.

29.4 Такты выполнения команд

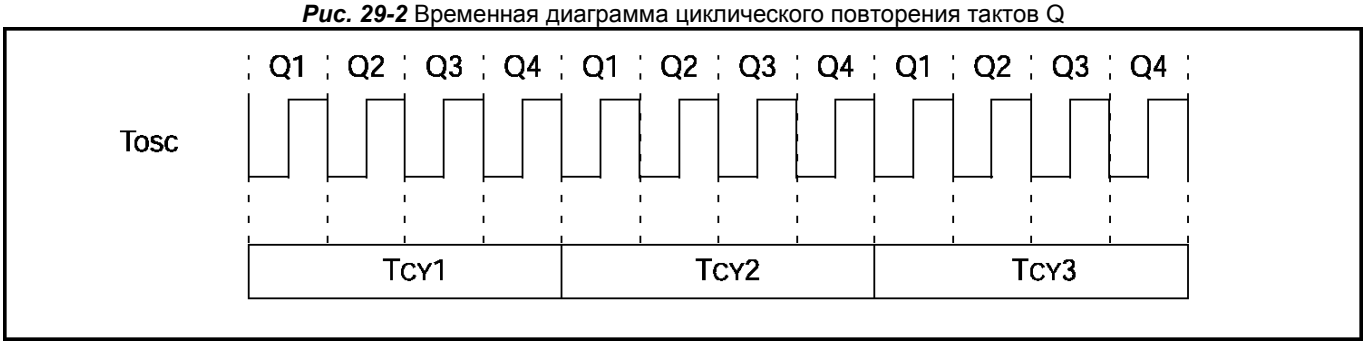
Каждый цикл команды (T_{CY}) состоит из четырех тактов (Q1-Q4). Такт Q равен по длительности периоду тактового генератора (T_{OSC}). Такты Q обеспечивают жесткую синхронизацию декодирования, чтения данных, обработки данных, записи результата для каждого цикла команды. На диаграмме показано соотношение тактов Q к циклу команды.

Цикл команды (T_{CY}), состоящий из 4-х тактов, обобщенно выглядит следующим образом:

Q1: Детектирование команды или принудительной пустой операции (NOP)
Q2: Операция чтения данных или отсутствие операции
Q3: Обработка данных
Q4: Операция записи данных или отсутствие операции

Детальный состав операций команды по тактам Q смотрите в описании команды.

Рис. 29-2 Временная диаграмма циклического повторения тактов Q



29.5 Описание команд

ADDLW Сложить константу с W

Синтаксис: [*label*] ADDLW k

Операнды: $0 \le k \le 255$

Операция: $(W) + k \rightarrow (W)$

Измен. флаги: C, DC, Z

Код:

11	111x	kkkk	kkkk

Описание: Содержимое регистра W складывается с 8 - разрядной константой 'k'. Результат 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 1:

```
ADDLW   0x15
```

До выполнения команды

W = 0x10

После выполнения команды

W = 0x25

Пример 2:

```
ADDLW   MYREG
```

До выполнения команды

W = 0x10
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x47

Пример 3:

```
ADDLW   HIGH (LU_TABLE)
```

До выполнения команды

W = 0x10
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0xA3

ADDWF Сложение W и f

Синтаксис: [*label*] ADDWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) + (f) → (dest)

Измен. флаги: C, DC, Z

Код:

00	0111	dfff	ffff

Описание: Сложить содержимое регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
ADDWF   FSR,0
До выполнения команды
        W = 0x17
        FSR = 0xC2
После выполнения команды
        W = 0xD9
        FSR = 0xC2
```

Пример 2:
```
ADDWF   INDF,1
До выполнения команды
        W = 0x17
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x20
После выполнения команды
        W = 0x17
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x37
```

Пример 3:
```
ADDWF   PCL,0
Случай 1
До выполнения команды
        W = 0x10
        PCL = 0x37
        C = x
После выполнения команды
        PCL = 0x47
        C = 0
Случай 2
До выполнения команды
        W = 0x10
        PCL = 0xF7
        PCH = 0x08
        C = x
После выполнения команды
        PCL = 0x07
        PCH = 0x08
        C = 1
```

ANDLW Побитное 'И' константы и W

Синтаксис: [*label*] ANDLW k

Операнды: $0 \le k \le 255$

Операция: (W) .AND. k → (W)

Измен. флаги: Z

Код:

11	1001	kkkk	kkkk

Описание: Выполняется побитное 'И' содержимого регистра W и 8 - разрядной константы 'k'. Результат 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 1: ANDLW 0x5F (0101 1111)

До выполнения команды

W = 0xA3 (1010 0011)

После выполнения команды

W = 0x03 (0000 0011)

Пример 2: ANDLW MYREG

До выполнения команды

W = 0xA3
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x23

Пример 3: ANDLW HIGH (LU_TABLE)

До выполнения команды

W = 0xA3
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0x83

ANDWF Побитное 'И' W и f

Синтаксис: [*label*] ANDWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .AND. (f) → (dest)

Измен. флаги: Z

Код:

00	0101	dfff	ffff

Описание: Выполняется побитное 'И' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:

```
ANDWF   FSR,1
```

До выполнения команды
W = 0x17 (0001 0111)
FSR = 0xC2 (1100 0010)
После выполнения команды
W = 0x17
FSR = 0x02 (0000 0010)

Пример 2:

```
ANDWF   FSR,0
```

До выполнения команды
W = 0x17 (0001 0111)
FSR = 0xC2 (1100 0010)
После выполнения команды
W = 0x02 (0000 0010)
FSR = 0xC2

Пример 3:

```
ANDWF   INDF,1
```

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x5A
После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x15

BCF Очистить бит b в регистре f

Синтаксис: [*label*] BCF f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: 0 → (f<b>)

Измен. флаги: Нет

Код:

01	00bb	bfff	ffff

Описание: Очистить бит 'b' в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
BCF     FLAG_REG,7
```

До выполнения команды

FLAG_REG = 0xC7 (**1**100 0111)

После выполнения команды

FLAG_REG = 0x47 (**0**100 0111)

Пример 2:

```
BCF     INDF,3
```

До выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x2F

После выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x27

BSF Установить бит b в регистре f

Синтаксис: [*label*] BSF f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: 1 → (f<b>)

Измен. флаги: Нет

Код:

01	01bb	bfff	ffff

Описание: Установить бит 'b' в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
BSF     FLAG_REG,7
```

До выполнения команды

FLAG_REG = 0x0A (**0**000 1010)

После выполнения команды

FLAG_REG = 0x8A (**1**000 1010)

Пример 2:

```
BSF     INDF,3
```

До выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x20

После выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x28

BTFSC Проверить бит b в регистре f, пропустить если 0

Синтаксис: [*label*] BTFSC f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: пропустить если (f<b>) = 0

Измен. флаги: Нет

Код:

01	10bb	bfff	ffff

Описание: Если бит 'b' в регистре 'f' равен '1' , то исполняется следующая инструкция.
Если бит 'b' в регистре 'f' равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Нет операции

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE    BTFSC   FLAG,4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
```

Случай 1
До выполнения команды
PC = адрес HERE
FLAG = xxx0 xxxx
После выполнения команды
Т.к. FLAG<4> = 0,
PC = адрес TRUE

Случай 2
До выполнения команды
PC = адрес HERE
FLAG = xxx1 xxxx
После выполнения команды
Т.к. FLAG<4> = 1,
PC = адрес FALSE

BTFSS Проверить бит b в регистре f, пропустить если 1

Синтаксис: [*label*] BTFSS f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: пропустить если (f<b>) = 1

Измен. флаги: Нет

Код:

01	11bb	bfff	ffff

Описание: Если бит 'b' в регистре 'f' равен '0' , то исполняется следующая инструкция.
Если бит 'b' в регистре 'f' равен '1' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Нет операции

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE    BTFSS   FLAG,4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
```

Случай 1 До выполнения команды
PC = адрес HERE
FLAG = xxx0 xxxx
После выполнения команды
T.к. FLAG<4> = 0,
PC = адрес FALSE

Случай 2 До выполнения команды
PC = адрес HERE
FLAG = xxx1 xxxx
После выполнения команды
T.к. FLAG<4> = 1,
PC = адрес TRUE

CALL Вызов подпрограммы

Синтаксис: [*label*] CALL k

Операнды: $0 \le k \le 2047$

Операция: (PC) + 1 → TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Измен. флаги: Нет

Код:

10	0kkk	kkkk	Kkkk

Описание: Вызов подпрограммы. Адрес следующей инструкции (PC+1) помещается в вершину стека. Одиннадцать бит адреса загружаются из кода команды в счетчик команд PC<10:0>. Два старших бита загружаются в счетчик команд PC<12:11> из регистра PCLATH. Команда CALL выполняется за два цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: HERE CALL THERE

До выполнения команды
PC = адрес HERE

После выполнения команды
PC = адрес THERE
TOS = адрес HERE + 1

CLRF Очистить f

Синтаксис: [*label*] CLRF f

Операнды: $0 \le f \le 127$

Операция: $00h \rightarrow (f)$
$1 \rightarrow Z$

Измен. флаги: Z

Код:

00	0001	1fff	ffff

Описание: Очистить содержимое регистра 'f' и установить флаг Z

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Запись в регистр 'f'

Пример 1:

```
CLRF    FLAG_REG
До выполнения команды
        FLAG_REG = 0x5A
После выполнения команды
        FLAG_REG = 0x00
        Z = 1
```

Пример 2:

```
CLRF    INDF
До выполнения команды
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0xAA
После выполнения команды
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x00
        Z = 1
```

CLRW Очистить W

Синтаксис: [*label*] CLRW

Операнды: Нет

Операция: 00h → (W)
1 → Z

Измен. флаги: Z

Код:

00	0001	0xxx	xxxx

Описание: Очистить содержимое регистра W и установить флаг Z

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Запись в регистр W

Пример 1: CLRW

До выполнения команды
W = 0x5A

После выполнения команды
W = 0x00
Z = 1

CLRWDT Очистить WDT

Синтаксис: [*label*] CLRWDT

Операнды: Нет

Операция: 00h → WDT,
00h → предделитель WDT,
1 → -TO
1 → -PD

Измен. флаги: -TO, -PD

Код:

00	0000	0110	0100

Описание: Инструкция CLRWDT сбрасывает WDT и предделитель, если он подключен к WDT. В регистре STATUS устанавливает биты -TO и -PD.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Сбросить WDT

Пример 1: CLRWDT

До выполнения команды
Счетчик WDT = ?
Делитель WDT = 1:128

После выполнения команды
Счетчик WDT = 0
Счетчик делителя WDT = 0
-TO = 1
- PD = 1
Делитель WDT = 1:128

Примечание. Команда CLRWDT не влияет на коэффициент делителя WDT.

COMF Инвертировать f

Синтаксис: [*label*] COMF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (-f) → (dest)

Измен. флаги: Z

Код:

00	1101	dfff	ffff

Описание: Инвертировать все биты в регистре 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
COMF    REG1,0
```
До выполнения команды
```
        REG1 = 0x13
```
После выполнения команды
```
        REG1 = 0x13
        W = 0xEC
```

Пример 2:
```
COMF    INDF,1
```
До выполнения команды
```
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0xAA
```
После выполнения команды
```
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x55
```

Пример 3:
```
COMF    REG1,1
```
До выполнения команды
```
        REG1 = 0xFF
```
После выполнения команды
```
        REG1 = 0x00
        Z = 1
```

DECF Вычесть 1 из f

Синтаксис: [*label*] DECF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - 1 → (dest)

Измен. флаги: Z

Код:

00	0011	dfff	ffff

Описание: Декрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
DECF    CNT,1
```
До выполнения команды
CNT = 0x01
Z = 0
После выполнения команды
CNT = 0x00
Z = 1

Пример 2:
```
DECF    INDF,1
```
До выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x01
Z = 0
После выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3:
```
DECF    CNT,0
```
До выполнения команды
CNT = 0x10
W = x
Z = 0
После выполнения команды
CNT = 0x10
W = 0x0F
Z = 0

DECFSZ Вычесть 1 из f и пропустить если 0

Синтаксис: [*label*] DECFSZ f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - 1 → (dest); пропустить если результат равен 0

Измен. флаги: Нет

Код:

00	1011	dfff	ffff

Описание: Декрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.
Если результат не равен '0' , то исполняется следующая инструкция. Если результат равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE      DECFSZ   CNT,1
          GOTO     LOOP
CONTINUE  •
          •
```

Случай 1
До выполнения команды
PC = адрес HERE
CNT = 0x01
После выполнения команды
CNT = 0x00
PC = адрес CONTINUE

Случай 2
До выполнения команды
PC = адрес HERE
CNT = 0x02
После выполнения команды
CNT = 0x01
PC = адрес HERE + 1

GOTO Безусловный переход

Синтаксис: [*label*] GOTO k

Операнды: $0 \le k \le 2047$

Операция: $k \rightarrow PC<10:0>$,
$(PCLATH<4:3>) \rightarrow PC<12:11>$

Измен. флаги: Нет

Код:

10	1kkk	kkkk	kkkk

Описание: Выполнить безусловный переход. Одиннадцать бит адреса загружаются из кода команды в счетчик команд PC<10:0>. Два старших бита загружаются в счетчик команд PC<12:11> из регистра PCLATH. Команда GOTO выполняется за два цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: GOTO THERE

После выполнения команды

PC = адрес THERE

INCF Прибавить 1 к f

Синтаксис: [*label*] INCF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) + 1 → (dest)

Измен. флаги: Z

Код:

00	1010	dfff	ffff

Описание: Инкрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: INCF CNT,1

До выполнения команды
CNT = 0xFF
Z = 0

После выполнения команды
CNT = 0x00
Z = 1

Пример 2: INCF INDF,1

До выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0xFF
Z = 0

После выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3: INCF CNT,0

До выполнения команды
CNT = 0x10
W = x
Z = 0

После выполнения команды
CNT = 0x10
W = 0x11
Z = 0

INCFSZ Прибавить 1 к f и пропустить если 0

Синтаксис: [*label*] INCFSZ f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) + 1 → (dest); пропустить если результат равен 0

Измен. флаги: Нет

Код:

00	1111	dfff	ffff

Описание: Инкрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'. Если результат не равен '0' , то исполняется следующая инструкция. Если результат равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE      INCFSZ   CNT,1
          GOTO     LOOP
CONTINUE  •
          •
```

Случай 1
До выполнения команды
PC = адрес HERE
CNT = 0xFF
После выполнения команды
CNT = 0x00
PC = адрес CONTINUE

Случай 2
До выполнения команды
PC = адрес HERE
CNT = 0x00
После выполнения команды
CNT = 0x01
PC = адрес HERE + 1

IORLW Побитное 'ИЛИ' константы и W

Синтаксис: [*label*] IORLW k

Операнды: $0 \le k \le 255$

Операция: (W) .OR. k → (W)

Измен. флаги: Z

Код:

11	1000	kkkk	kkkk

Описание: Выполняется побитное 'ИЛИ' содержимого регистра W и 8-разрядной константы 'k'. Результат 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 1:

```
IORLW   0x35
```

До выполнения команды
W = 0x9A

После выполнения команды
W = 0xBF
Z = 0

Пример 2:

```
IORLW   MYREG
```

До выполнения команды
W = 0x9A
MYREG = 0x37 (адрес регистра)

После выполнения команды
W = 0x9F
Z = 0

Пример 3:

```
IORLW   HIGH (LU_TABLE)
```

До выполнения команды
W = 0x9A
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды
W = 0x9B

Пример 4:

```
IORLW   0x00
```

До выполнения команды
W = 0x00

После выполнения команды
W = 0x00
Z = 1

IORWF Побитное 'ИЛИ' W и f

Синтаксис: [*label*] IORWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .OR. (f) → (dest)

Измен. флаги: Z

Код:

00	0100	dfff	ffff

Описание: Выполняется побитное 'ИЛИ' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: IORWF RESULT,0

До выполнения команды
RESULT = 0x13
W = 0x91

После выполнения команды
RESULT = 0x13
W = 0x93
Z = 0

Пример 2: IORWF INDF,1

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x30

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x37
Z = 0

Пример 3: IORWF RESULT,1

Случай 1 До выполнения команды
RESULT = 0x13
W = 0x91

После выполнения команды
RESULT = 0x93
W = 0x91
Z = 0

Случай 2 До выполнения команды
RESULT = 0x00
W = 0x00

После выполнения команды
RESULT = 0x00
W = 0x00
Z = 1

MOVF Переслать f

Синтаксис: [*label*] MOVF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) → (dest)

Измен. флаги: Z

Код:

00	1000	dfff	ffff

Описание: Содержимое регистра 'f' пересылается в регистр адресата. Если d=0, значение сохраняется в регистре W. Если d=1, значение сохраняется в регистре 'f'. d=1 используется для проверки содержимого регистра 'f' на ноль.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: MOVF FSR,0

До выполнения команды
W = 0x00
FSR = 0xC2

После выполнения команды
W = 0xC2
FSR = 0xC2
Z = 0

Пример 2: MOVF INDF,1

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3: MOVF FSR,1

Случай 1 До выполнения команды
FSR = 0x43

После выполнения команды
FSR = 0x43
Z = 0

Случай 2 До выполнения команды
FSR = 0x00

После выполнения команды
FSR = 0x00
Z = 1

MOVLW Переслать константу в W

Синтаксис: [*label*] MOVLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow (W)$

Измен. флаги: Нет

Код:

11	00xx	kkkk	kkkk

Описание: Переслать константу 'k' в регистр W. В неиспользуемых битах ассемблер устанавливает '0'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1:

```
MOVLW   0x5A
```

После выполнения команды

W = 0x5A

Пример 2:

```
MOVLW   MYREG
```

До выполнения команды

W = 0x10
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x37
Z = 0

Пример 3:

```
MOVLW   HIGH (LU_TABLE)
```

До выполнения команды

W = 0x10
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0x93

MOVWF Переслать W в f

Синтаксис: [*label*] MOVWF f

Операнды: $0 \le f \le 127$

Операция: $(W) \rightarrow (f)$

Измен. флаги: Нет

Код:

00	0000	1fff	ffff

Описание: Переслать содержимое регистра W в регистр 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
MOVWF   OPTION_REG
```

До выполнения команды
OPTION = 0xFF
W = 0x4F

После выполнения команды
OPTION = 0x4F
W = 0x4F

Пример 2:

```
MOVWF   INDF
```

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x17

NOP Нет операции

Синтаксис: [*label*] NOP

Операнды: Нет

Операция: Нет операции

Измен. флаги: Нет

Код:

00	0000	0xx0	0000

Описание: Нет операции

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Нет операции

Пример 1: HERE NOP

До выполнения команды

PC = адрес HERE

После выполнения команды

PC = адрес HERE + 1

OPTION Загрузить регистр OPTION

Синтаксис:	[*label*] OPTION
Операнды:	Нет
Операция:	(W) → OPTION
Измен. флаги:	Нет

Код:

00	0000	0110	0010

Описание: Переслать содержимое регистра W в регистр OPTION. Инструкция поддерживается для совместимости программы с семейством микроконтроллеров PIC16C5X. Запись/чтение регистра OPTION можно выполнить прямой или косвенной адресацией.

Слов:	1
Циклов:	1
Пример:	

Для совместимости программного обеспечения с последующими выпускаемыми микроконтроллерами семейства PIC16CXX не рекомендуется использовать эту инструкцию.

RETFIE Возврат из подпрограммы с разрешением прерываний

Синтаксис: [*label*] RETFIE

Операнды: Нет

Операция: TOS → PC
1 → GIE

Измен. флаги: Нет

Код:

00	0000	0000	1001

Описание: Возврат из подпрограммы обработки прерываний. Вершина стека TOS загружается в счетчик команд PC. Устанавливается в '1' флаг глобального разрешения прерываний GIE(INTCON<7>). Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: RETFIE

После выполнения команды
PC = TOS
GIE = 1

RETLW Возврат из подпрограммы с загрузкой константы в W

Синтаксис: [*label*] RETLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow (W)$
$TOS \rightarrow PC$

Измен. флаги: Нет

Код:

11	01xx	kkkk	kkkk

Описание: В регистр W загружается 8-разрядная константа. Вершина стека TOS загружается в счетчик команд PC. Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
        CALL    TABLE
        •
        •
TABLE   ADDWF   PCL, f
        RETLW   k1
        RETLW   k2
        •
        •
        RETLW   kn
```

До выполнения команды
W = 0x07

После выполнения команды
W = значение k8
PC = TOS = адрес HERE +1

RETURN Возврат из подпрограммы

Синтаксис: [*label*] RETURN

Операнды: Нет

Операция: TOS → PC

Измен. флаги: Нет

Код:

00	0000	0000	1000

Описание: Возврат из подпрограммы. Вершина стека TOS загружается в счетчик команд PC. Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: RETURN

После выполнения команды

PC = TOS

RLF Циклический сдвиг f влево через перенос

Синтаксис: [*label*] RLF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

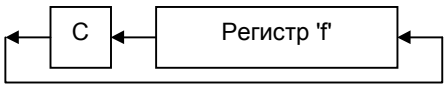
Операция: См. описание

Измен. флаги: C

Код:

00	1101	dfff	ffff

Описание: Выполняется циклический сдвиг влево содержимого регистра 'f' через бит C регистра STATUS. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.



Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: RLF REG1,0

До выполнения команды
REG1 = 1110 0110
C = 0

После выполнения команды
REG1 = 1110 0110
W = 1100 1100
C = 1

Пример 2: RLF INDF,1

Случай 1 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x3A (0011 1010)
C = 1

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x75 (0111 0101)
C = 0

Случай 2 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0xB9 (1011 1001)
C = 0

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x72 (0111 0010)
C = 1

RRF Циклический сдвиг f вправо через перенос

Синтаксис: [*label*] RRF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

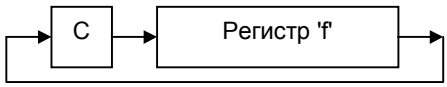
Операция: См. описание

Измен. флаги: C

Код:

00	1100	dfff	ffff

Описание: Выполняется циклический сдвиг вправо содержимого регистра 'f' через бит C регистра STATUS. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.



Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: RRF REG1,0

До выполнения команды
REG1 = 1110 0110
W = xxxx xxxx
C = 0

После выполнения команды
REG1 = 1110 0110
W = 0111 0011
C = 0

Пример 2: RRF INDF,1

Случай 1 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x3A (0011 1010)
C = 1

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x9D (1001 1101)
C = 0

Случай 2 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x39 (0011 1001)
C = 0

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x1C (0001 1100)
C = 1

SLEEP Перейти в режим SLEEP

Синтаксис: [*label*] SLEEP

Операнды: Нет

Операция: 00h → WDT
00h → предделитель WDT
1 → - TO
0 → - PD

Измен. флаги: -TO, -PD

Код:

00	0000	0110	0011

Описание: Сбросить флаг включения питания -PD в '0'. Установить флаг переполнения WDT -TO в '1'. Очистить таймер WDT и его предделитель. Перевести микроконтроллер в режим SLEEP и выключить тактовый генератор.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Переход в SLEEP режим

Пример 1: SLEEP

Примечание. Команда SLEEP не влияет на коэффициент делителя WDT.

SUBLW Вычесть W из константы

Синтаксис: [*label*] SUBLW k

Операнды: $0 \le k \le 255$

Операция: k - (W) → (W)

Измен. флаги: C, DC, Z

Код:

11	110x	kkkk	kkkk

Описание: Вычесть содержимое регистра W из 8-разрядной константы 'k'. Результат 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 1: SUBLW 0x02

Случай 1 До выполнения команды
W = 0x01
C = ?
Z = ?
После выполнения команды
W = 0x01
C = 1 ; результат положительный
Z = 0

Случай 2 До выполнения команды
W = 0x02
C = ?
Z = ?
После выполнения команды
W = 0x00
C = 1 ; результат нулевой
Z = 1

Случай 3 До выполнения команды
W = 0x03
C = ?
Z = ?
После выполнения команды
W = 0xFF
C = 0 ; результат отрицательный
Z = 0

Пример 2: SUBLW MYREG
До выполнения команды
W = 0x10
MYREG = 0x37 (адрес регистра)
После выполнения команды
W = 0x27
C = 1

SUBWF Вычесть W из f

Синтаксис: [*label*] SUBWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - (W) → (dest)

Измен. флаги: C, DC, Z

Код:

00	0010	dfff	ffff

Описание: Вычесть содержимое регистра W из регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: SUBWF REG1,1

Случай 1 До выполнения команды
REG1 = 0x03
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0x01
W = 0x02
C =1 ; результат положительный
Z = 0

Случай 2 До выполнения команды
REG1 = 0x02
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0x00
W = 0x02
C =1 ; результат нулевой
Z = 1

Случай 3 До выполнения команды
REG1 = 0x01
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0xFF
W = 0x02
C =0 ; результат отрицательный
Z = 0

SWAPF Поменять местами полубайты в регистре f

Синтаксис: [*label*] SWAPF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f<3:0>) → (dest<7:4>)
(f<7:4>) → (dest<3:0>)

Измен. флаги: Нет

Код:

00	1110	dfff	ffff

Описание: Поменять местами старший и младший полубайты регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: SWAPF REG,0

До выполнения команды
REG = 0xA5

После выполнения команды
REG = 0xA5
W = 0x5A

Пример 2: SWAPF INDF,1

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x20

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x02

Пример 3: SWAPF REG,1

До выполнения команды
REG = 0xA5

После выполнения команды
REG = 0x5A

TRIS Загрузить регистр TRIS

Синтаксис:	[*label*] TRIS f
Операнды:	$5 \le f \le 7$
Операция:	(W) → TRIS регистр f
Измен. флаги:	Нет

Код:

00	0000	0110	0fff

Описание: Переслать содержимое W в регистр TRIS. Инструкция поддерживается для совместимости программы с семейством микроконтроллеров PIC16C5X. Запись/чтение регистра OPTION можно выполнить прямой или косвенной адресацией.

Слов:

Циклов:

Пример:

Для совместимости программного обеспечения с последующими выпускаемыми микроконтроллерами семейства PIC16CXX не рекомендуется использовать эту инструкцию.

XORLW Побитное 'исключающее ИЛИ' константы и W

Синтаксис: [*label*] IORLW k

Операнды: $0 \le k \le 255$

Операция: (W) .XOR. k → (W)

Измен. флаги: Z

Код:

11	1010	kkkk	kkkk

Описание: Выполняется побитное 'исключающее ИЛИ' содержимого регистра W и 8-разрядной константы 'k'. Результат 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 1: XORLW 0xAF (1010 1111)

До выполнения команды
W = 0xB5 (1011 0101)

После выполнения команды
W = 0x1A (0001 1010)
Z = 0

Пример 2: XORLW MYREG

До выполнения команды
W = 0xAF
MYREG = 0x37 (адрес регистра)

После выполнения команды
W = 0x18
Z = 0

Пример 3: XORLW HIGH (LU_TABLE)

До выполнения команды
W = 0xAF
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды
W = 0x3C
Z = 0

XORWF Побитное 'исключающее ИЛИ' W и f

Синтаксис: [*label*] XORWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .XOR. (f) → (dest)

Измен. флаги: Z

Код:

00	0100	dfff	ffff

Описание: Выполняется побитное 'исключающее ИЛИ' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: XORWF REG,1

До выполнения команды
REG = 0xAF
W = 0xB5

После выполнения команды
REG = 0x1A
W = 0xB5

Пример 2: XORWF REG,0

До выполнения команды
REG = 0xAF
W = 0xB5

После выполнения команды
REG = 0xAF
W = 0x1A

Пример 3: XORWF INDF,1

До выполнения команды
W = 0xB5
FSR = 0xC2
Значение регистра с адресом в FSR = 0xAF

После выполнения команды
W = 0xB5
FSR = 0xC2
Значение регистра с адресом в FSR = 0x1A

29.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Как можно непосредственно изменить значение регистра W? Требуется декрементировать значение в регистре W.

Ответ 1:

Существует несколько способов:

1. В микроконтроллерах PICmicro среднего семейства предусмотрены команды, работающие с регистром W. Например, если необходимо выполнить декремент содержимого регистра W, то можно это выполнить командой ADDLW 0xFF (0x - префикс, обозначающий шестнадцатеричное число в ассемблере MPASM).
2. Заметьте, что большинство команд могут изменять содержимое регистра памяти данных с сохранением результата в W. Это означает, что можно выполнить декремент значения при загрузке регистра W. Адресатом выполнения команды должен быть W (DECF адрес регистра, W).

Вопрос 2: Существует ли какая-нибудь опасность использования команды TRIS для микроконтроллеров PIC16CXXX? В документации на микроконтроллер сказано, что не рекомендуется использовать эти команды.

Ответ 2:

Для совместимости программного обеспечения с последующими версиями микроконтроллеров не рекомендуется использовать команду TRIS. Необходимо учитывать, что с помощью команды TRIS можно настроить только порты A, B и C. Новые микроконтроллеры могут не поддерживать эту команду.

Вопрос 3: Нужно переключать банк памяти данных (выбрать банк 1) при использовании команды TRIS? Требуется настроить направление каналов ввода/вывода PORTA.

Ответ 3:

При использовании команды TRIS переключать банк памяти данных не требуется. Для совместимости программного обеспечения с последующими версиями микроконтроллеров не рекомендуется использовать команду TRIS.

Вопрос 4: В документации указано, что требуется осторожность при использовании команд "чтение - модификация - запись". Объясните пожалуйста почему.

Ответ 4:

Простой пример команды со структурой "чтение - модификация - запись" - бит ориентированная команда BCF. Можно предполагать, что выполняется просто сброс бита, управляющего выводом порта. Фактически происходит чтение состояния всего порта ввода/вывода, затем сбрасывается требуемый бит и новое значение записывается в порт ввода/вывода (или регистр). Любая команда, зависящая от текущего значение регистра является командой со структурой "чтение - модификация - запись" (ADDWF, SUBWF, BCF, BSF, INCF, XORWF и др.). Команды, независящие от текущего значения регистра не являются командами "чтение - модификация - запись" (MOVWF, CLRF и др.).

Рассмотрим одну ситуацию выполнения команд "чтение - модификация - запись" для порта ввода/вывода. Например, все биты регистра TRISB настраивают PORTB на выход, и на всех выводах PORTB установлен высокий логический уровень сигнала. Теперь Вы настраиваете RB3 как вход, на котором присутствует низкий логический уровень. Выполняете команду BCF PORTB,6, чтобы на RB6 установить низкий логический уровень. Если Вы опять настроите вывод RB3 как выход, то на нем будет формироваться низкий логический уровень, хотя ранее Вы устанавливали высокий логический уровень. При выполнении команды BCF для другого вывода порта (RB6) происходит чтение состояния всего порта (включая 0 на RB3). Бит 6 изменяется к требуемому значению, но т.к. на RB3 был прочитан '0', он будет записан в защелку порта. Когда вывод будет настроен на выход, новое значение будет передано на вывод.

Вопрос 5: При использовании команды BCF другие выводы порта принимают низкий логический уровень. Почему?

Ответ 5:

1. Случай, когда команда "чтение - модификация - запись" изменяет состояние других выводов порта, можно продемонстрировать следующим образом. Предположим, что все каналы PORTC настроены на выход и имеют низкий логический уровень. К каждому выводу подключен светодиод, который светится при формировании высокого логического уровня на выводах порта. Параллельно каждому светодиоду подключен конденсатор с емкостью 100мкФ. Программа микроконтроллера выполняется очень быстро, тактовая частота 20МГц. Теперь последовательно формируем команды, включающие светодиоды: BSF PORTC,0; BSF PORTC,1; BSF PORTC,2 и т.д. Вы можете видеть, что только на последнем выводе высокий уровень сигнала и только последний светодиод светится. Это произошло потому , что конденсаторы требуют некоторого времени для зарядки до напряжения высокого логического уровня. Поскольку каждый вывод устанавливался в '1' прежде, чем зарядится конденсатор предыдущего вывода, чтение давало результат '0'. Это '0' записывается в выходную защелку, восстанавливая низкий логический уровень на выводе (команда "чтение - модификация - запись"). Учитывать этот эффект необходимо только при высокой тактовой частоте микроконтроллера и выполнении последовательных операций с портами ввода/вывода.
2. Такая ситуация возможна в микроконтроллерах PIC16C7XX, если Вы не настроили каналы порта ввода/вывода должным образом в регистре ADCON1. Если вывод настроен как аналоговый вход, то чтение будет давать результат '0' независимо от уровня напряжения на выводе. Это исключение к правилу, что всегда выполняется чтение состояние вывода. Вы можете настраивать аналоговый вывод как цифровой выход в регистре TRISA, и управлять логическим уровнем на выходе, но чтение всегда будет давать результат '0'. Поэтому, если вы обращаетесь к порту командой "чтение - модификация - запись", все аналоговые выводы читаются как '0', командой изменяется прочитанное значение и записывается назад в защелку порта как '0'. На аналоговых входах могут присутствовать нелогические уровни, поэтому входные цифровые буферы выключены для предотвращения возможного повышенного энергопотребления.

29.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с системой команд микроконтроллеров PICmicro MCU:

Документ	Номер
В настоящее время документы не подготовлены	

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.