29.5 Instruction Descriptions

ADDLW — Add Literal and W

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → W

Status Affected: C, DC, Z

Encoding:

11	111x	kkkk	kkkk

Description: The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example1

```
ADDLW   0x15
```

Before Instruction
W = 0x10
After Instruction
W = 0x25

Example 2

```
ADDLW   MYREG
```

Before Instruction
W = 0x10
Address of MYREG † = 0x37
† MYREG is a symbol for a data memory location
After Instruction
W = 0x47

Example 3

```
ADDLW   HIGH (LU_TABLE)
```

Before Instruction
W = 0x10
Address of LU_TABLE † = 0x9375
† LU_TABLE is a label for an address in program memory
After Instruction
W = 0xA3

Example 4

```
ADDLW   MYREG
```

Before Instruction
W = 0x10
Address of PCL † = 0x02
† PCL is the symbol for the Program Counter low byte location
After Instruction
W = 0x12

© 1997 Microchip Technology Inc.

ADDWF — Add W and f

Syntax:	[*label*] ADDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) + (f) → destination
Status Affected:	C, DC, Z
Encoding:	00 | 0111 | dfff | ffff
Description:	Add the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
ADDWF   FSR, 0
```

Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0xD9
FSR = 0xC2

Example 2

```
ADDWF   INDF, 1
```

Before Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x20
After Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x37

Example 3

```
ADDWF   PCL
```

Case 1:
Before Instruction
W = 0x10
PCL = 0x37
C = x
After Instruction
PCL = 0x47
C = 0

Case 2:
Before Instruction
W = 0x10
PCL = 0xF7
PCH = 0x08
C = x
After Instruction
PCL = 0x07
PCH = 0x08
C = 1

ANDLW — And Literal with W

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W).AND. (k) → W

Status Affected: Z

Encoding:

11	1001	kkkk	kkkk

Description: The contents of W register are AND'ed with the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example 1

```
ANDLW   0x5F
```

```
Before Instruction                ; 0101 1111  (0x5F)
        W   =  0xA3               ; 1010 0011  (0xA3)
After Instruction                 ;----------  ------
        W   =  0x03               ; 0000 0011  (0x03)
```

Example 2

```
ANDLW   MYREG
```

Before Instruction
W = 0xA3
Address of MYREG † = 0x37
† MYREG is a symbol for a data memory location
After Instruction
W = 0x23

Example 3

```
ANDLW   HIGH (LU_TABLE)
```

Before Instruction
W = 0xA3
Address of LU_TABLE † = 0x9375
† LU_TABLE is a label for an address in program memory
After Instruction
W = 0x83

© 1997 Microchip Technology Inc.

ANDWF — AND W with f

Syntax:	[*label*] ANDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W).AND. (f) → destination
Status Affected:	Z
Encoding:	00 | 0101 | dfff | ffff
Description:	AND the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
ANDWF   FSR, 1
```

Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0x17
FSR = 0x02

```
; 0001 0111  (0x17)
; 1100 0010  (0xC2)
;----------  ------
; 0000 0010  (0x02)
```

Example 2

```
ANDWF   FSR, 0
```

Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0x02
FSR = 0xC2

```
; 0001 0111  (0x17)
; 1100 0010  (0xC2)
;----------  ------
; 0000 0010  (0x02)
```

Example 3

```
ANDWF   INDF, 1
```

Before Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x5A
After Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x15

BCF — Bit Clear f

Syntax: [*label*] BCF f,b

Operands: $0 \le f \le 127$
$0 \le b \le 7$

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

01	00bb	bfff	ffff

Description: Bit 'b' in register 'f' is cleared.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example 1

```
BCF     FLAG_REG, 7
```

Before Instruction

FLAG_REG = 0xC7 ; **1**100 0111

After Instruction

FLAG_REG = 0x47 ; **0**100 0111

Example 2

```
BCF     INDF, 3
```

Before Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x2F

After Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x27

© 1997 Microchip Technology Inc.

BSF — Bit Set f

Syntax:	[*label*] BSF f,b
Operands:	$0 \le f \le 127$ $0 \le b \le 7$
Operation:	$1 \rightarrow f<b>$
Status Affected:	None
Encoding:	01 | 01bb | bfff | ffff
Description:	Bit 'b' in register 'f' is set.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example 1

```
BSF     FLAG_REG, 7
```

Before Instruction

```
FLAG_REG =0x0A          ; 0000 1010
```

After Instruction

```
FLAG_REG =0x8A          ; 1000 1010
```

Example 2

```
BSF     INDF, 3
```

Before Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x20

After Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x28

BTFSC — Bit Test, Skip if Clear

Syntax: [*label*] BTFSC f,b

Operands: $0 \le f \le 127$
$0 \le b \le 7$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

01	10bb	bfff	ffff

Description: If bit 'b' in register 'f' is '0' then the next instruction is skipped.
If bit 'b' is '0' then the next instruction (fetched during the current instruction execution) is discarded, and a NOP is executed instead, making this a 2 cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	No operation

If skip (2nd cycle):

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example 1

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

Case 1: Before Instruction
PC = addressHERE
FLAG= xxx0 xxxx
After Instruction
Since FLAG<4>= 0,
PC = addressTRUE

Case 2: Before Instruction
PC = addressHERE
FLAG= xxx1 xxxx
After Instruction
Since FLAG<4>=1,
PC = addressFALSE

© 1997 Microchip Technology Inc.

BTFSS — Bit Test f, Skip if Set

Syntax:	[*label*] BTFSS f,b
Operands:	$0 \le f \le 127$ $0 \le b < 7$
Operation:	skip if (f<b>) = 1
Status Affected:	None
Encoding:	`01` `11bb` `bfff` `ffff`
Description:	If bit 'b' in register 'f' is '1' then the next instruction is skipped. If bit 'b' is '1', then the next instruction (fetched during the current instruction execution) is discarded and a NOP is executed instead, making this a 2 cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	No operation

If skip (2nd cycle):

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example 1

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

Case 1: Before Instruction
PC = addressHERE
FLAG= xxx0 xxxx
After Instruction
Since FLAG<4>= 0,
PC = addressFALSE

Case 2: Before Instruction
PC = addressHERE
FLAG= xxx1 xxxx
After Instruction
Since FLAG<4>=1,
PC = addressTRUE

CALL — Call Subroutine

Syntax:	[*label*] CALL k
Operands:	$0 \le k \le 2047$
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<4:3>) → PC<12:11>
Status Affected:	None
Encoding:	10 | 0kkk | kkkk | kkkk
Description:	Call Subroutine. First, the 13-bit return address (PC+1) is pushed onto the stack. The eleven bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH<4:3>. CALL is a two cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

1st cycle:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	No operation

2nd cycle:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example 1

```
HERE    CALL    THERE
```

Before Instruction
PC = AddressHERE
After Instruction
TOS = Address HERE+1
PC = Address THERE

© 1997 Microchip Technology Inc.

CLRF — Clear f

Syntax:	[*label*] CLRF f
Operands:	$0 \le f \le 127$
Operation:	00h → f 1 → Z
Status Affected:	Z
Encoding:	00 | 0001 | 1fff | ffff
Description:	The contents of register 'f' are cleared and the Z bit is set.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example 1

```
CLRF    FLAG_REG
```

Before Instruction
FLAG_REG=0x5A
After Instruction
FLAG_REG=0x00
Z = 1

Example 2

```
CLRF    INDF
```

Before Instruction
FSR = 0xC2
Contents of Address (FSR)=0xAA
After Instruction
FSR = 0xC2
Contents of Address (FSR)=0x00
Z = 1

CLRW — Clear W

Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → W 1 → Z
Status Affected:	Z
Encoding:	00 | 0001 | 0xxx | xxxx
Description:	W register is cleared. Zero bit (Z) is set.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'W'

Example 1

```
CLRW
```

Before Instruction
W = 0x5A
After Instruction
W = 0x00
Z = 1

© 1997 Microchip Technology Inc.

CLRWDT — Clear Watchdog Timer

Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT 0 → WDT prescaler count, 1 → $\overline{TO}$ 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Encoding:	00 | 0000 | 0110 | 0100
Description:	CLRWDT instruction clears the Watchdog Timer. It also clears the prescaler count of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process data	Clear WDT Counter

Example 1 `CLRWDT`

Before Instruction
- WDT counter= x
- WDT prescaler =1:128

After Instruction
- WDT counter=0x00
- WDT prescaler count=0
- $\overline{TO}$ = 1
- $\overline{PD}$ = 1
- WDT prescaler =1:128

Note: The CLRWDT instruction does not affect the assignment of the WDT prescaler.

COMF — Complement f

Syntax: [*label*] COMF f,d

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

Operation: $(\bar{f}) \rightarrow$ destination

Status Affected: Z

Encoding:

00	1001	dfff	ffff

Description: The contents of register 'f' are 1's complemented. If 'd' is 0 the result is stored in W. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
COMF    REG1, 0
```

Before Instruction
REG1= 0x13
After Instruction
REG1= 0x13
W = 0xEC

Example 2

```
COMF    INDF, 1
```

Before Instruction
FSR = 0xC2
Contents of Address (FSR)=0xAA
After Instruction
FSR = 0xC2
Contents of Address (FSR)=0x55

Example 3

```
COMF    REG1, 1
```

Before Instruction
REG1= 0xFF
After Instruction
REG1= 0x00
Z = 1

© 1997 Microchip Technology Inc.

DECF — Decrement f

Syntax:	[*label*] DECF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → destination
Status Affected:	Z
Encoding:	00 | 0011 | dfff | ffff
Description:	Decrement register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
DECF    CNT, 1
```

Before Instruction
CNT = 0x01
Z = 0
After Instruction
CNT = 0x00
Z = 1

Example 2

```
DECF    INDF, 1
```

Before Instruction
FSR = 0xC2
Contents of Address (FSR) = 0x01
Z = 0
After Instruction
FSR = 0xC2
Contents of Address (FSR) = 0x00
Z = 1

Example 3

```
DECF    CNT, 0
```

Before Instruction
CNT = 0x10
W = x
Z = 0
After Instruction
CNT = 0x10
W = 0x0F
Z = 0

DECFSZ — Decrement f, Skip if 0

Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → destination; skip if result = 0
Status Affected:	None
Encoding:	00 | 1011 | dfff | ffff
Description:	The contents of register 'f' are decremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'. If the result is 0, then the next instruction (fetched during the current instruction execution) is discarded and a NOP is executed instead, making this a 2 cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

If skip (2nd cycle):

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example

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

Case 1: Before Instruction
- PC = address HERE
- CNT = 0x01

After Instruction
- CNT = 0x00
- PC = address CONTINUE

Case 2: Before Instruction
- PC = address HERE
- CNT = 0x02

After Instruction
- CNT = 0x01
- PC = address HERE + 1

© 1997 Microchip Technology Inc.

GOTO — Unconditional Branch

Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 2047$
Operation:	k → PC<10:0> PCLATH<4:3> → PC<12:11>
Status Affected:	None
Encoding:	`10` `1kkk` `kkkk` `kkkk`
Description:	`GOTO` is an unconditional branch. The eleven bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. `GOTO` is a two cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

1st cycle:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>	Process data	No operation

2nd cycle:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example `GOTO THERE`

After Instruction

PC =Address`THERE`

INCF — Increment f

Syntax: [*label*] INCF f,d

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

Operation: (f) + 1 → destination

Status Affected: Z

Encoding:

00	1010	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
INCF    CNT, 1
```

Before Instruction
CNT = 0xFF
Z = 0
After Instruction
CNT = 0x00
Z = 1

Example 2

```
INCF    INDF, 1
```

Before Instruction
FSR = 0xC2
Contents of Address (FSR) = 0xFF
Z = 0
After Instruction
FSR = 0xC2
Contents of Address (FSR) = 0x00
Z = 1

Example 3

```
INCF    CNT, 0
```

Before Instruction
CNT = 0x10
W = x
Z = 0
After Instruction
CNT = 0x10
W = 0x11
Z = 0

© 1997 Microchip Technology Inc.

INCFSZ — Increment f, Skip if 0

Syntax:	[*label*] INCFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → destination, skip if result = 0
Status Affected:	None
Encoding:	`00` `1111` `dfff` `ffff`
Description:	The contents of register 'f' are incremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'. If the result is 0, then the next instruction (fetched during the current instruction execution) is discarded and a NOP is executed instead, making this a 2 cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

If skip (2nd cycle):

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example

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

Case 1: Before Instruction
- PC = address `HERE`
- CNT = 0xFF

After Instruction
- CNT = 0x00
- PC = address `CONTINUE`

Case 2: Before Instruction
- PC = address `HERE`
- CNT = 0x00

After Instruction
- CNT = 0x01
- PC = address `HERE` + 1

IORLW — Inclusive OR Literal with W

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W).OR. k → W

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

Description: The contents of the W register is OR'ed with the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example 1

```
IORLW   0x35
```

Before Instruction
W = 0x9A
After Instruction
W = 0xBF
Z = 0

Example 2

```
IORLW   MYREG
```

Before Instruction
W = 0x9A
Address of MYREG † = 0x37
† MYREG is a symbol for a data memory location
After Instruction
W = 0x9F
Z = 0

Example 3

```
IORLW   HIGH (LU_TABLE)
```

Before Instruction
W = 0x9A
Address of LU_TABLE † = 0x9375
† LU_TABLE is a label for an address in program memory
After Instruction
W = 0x9B
Z = 0

Example 4

```
IORLW   0x00
```

Before Instruction
W = 0x00
After Instruction
W = 0x00
Z = 1

© 1997 Microchip Technology Inc.

IORWF — Inclusive OR W with f

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W).OR. (f) → destination
Status Affected:	$\overline{Z}$
Encoding:	00 | 0100 | dfff | ffff
Description:	Inclusive OR the W register with register 'f'. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
IORWF   RESULT, 0
```

Before Instruction
RESULT=0x13
W = 0x91
After Instruction
RESULT=0x13
W = 0x93
Z = 0

Example 2

```
IORWF   INDF, 1
```

Before Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x30
After Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x37
Z = 0

Example 3

```
IORWF   RESULT, 1
```

Case 1: Before Instruction
RESULT=0x13
W = 0x91
After Instruction
RESULT=0x93
W = 0x91
Z = 0

Case 2: Before Instruction
RESULT=0x00
W = 0x00
After Instruction
RESULT=0x00
W = 0x00
Z = 1

MOVLW — Move Literal to W

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

11	00xx	kkkk	kkkk

Description: The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example 1

```
MOVLW   0x5A
```

After Instruction
W = 0x5A

Example 2

```
MOVLW   MYREG
```

Before Instruction
W = 0x10
Address of MYREG † = 0x37
† MYREG is a symbol for a data memory location
After Instruction
W = 0x37

Example 3

```
MOVLW   HIGH (LU_TABLE)
```

Before Instruction
W = 0x10
Address of LU_TABLE † = 0x9375
† LU_TABLE is a label for an address in program memory
After Instruction
W = 0x93

© 1997 Microchip Technology Inc.

MOVF — Move f

Syntax:	[*label*] MOVF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) → destination
Status Affected:	Z
Encoding:	00 | 1000 | dfff | ffff
Description:	The contents of register 'f' is moved to a destination dependent upon the status of 'd'. If 'd' = 0, destination is W register. If 'd' = 1, the destination is file register 'f' itself. 'd' = 1 is useful to test a file register since status flag Z is affected.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
MOVF    FSR, 0
```

Before Instruction
W = 0x00
FSR = 0xC2
After Instruction
W = 0xC2
Z = 0

Example 2

```
MOVF    INDF, 0
```

Before Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x00
After Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x00
Z = 1

Example 3

```
MOVF    FSR, 1
```

Case 1: Before Instruction
FSR = 0x43
After Instruction
FSR = 0x43
Z = 0

Case 2: Before Instruction
FSR = 0x00
After Instruction
FSR = 0x00
Z = 1

MOVWF — Move W to f

Syntax: [*label*] MOVWF f

Operands: $0 \le f \le 127$

Operation: (W) → f

Status Affected: None

Encoding:

00	0000	1fff	ffff

Description: Move data from W register to register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example 1

```
MOVWF   OPTION_REG
```

Before Instruction
OPTION_REG=0xFF
W = 0x4F

After Instruction
OPTION_REG=0x4F
W = 0x4F

Example 2

```
MOVWF   INDF
```

Before Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x00

After Instruction
W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x17

© 1997 Microchip Technology Inc.

NOP — No Operation

Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Encoding:	00 | 0000 | 0xx0 | 0000
Description:	No operation.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example : `HERE NOP`

Before Instruction
PC = address HERE
After Instruction
PC = address HERE + 1

OPTION — Load Option Register

Syntax:	[*label*] OPTION
Operands:	None
Operation:	(W) → OPTION
Status Affected:	None
Encoding:	00 | 0000 | 0110 | 0010
Description:	The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it.
Words:	1
Cycles:	1

To maintain upward compatibility with future PIC16CXX products, do not use this instruction.

© 1997 Microchip Technology Inc.

RETFIE — Return from Interrupt

Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None
Encoding:	00 | 0000 | 0000 | 1001
Description:	Return from Interrupt. The 13-bit address at the Top of Stack (TOS) is loaded in the PC. The Global Interrupt Enable bit, GIE (INTCON<7>), is automatically set, enabling Interrupts. This is a two cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

1st cycle:

Q1	Q2	Q3	Q4
Decode	No operation	Process data	No operation

2nd cycle:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example

```
RETFIE
```

After Instruction

PC = TOS
GIE = 1

RETLW — Return with Literal in W

Syntax: [*label*] RETLW k

Operands: $0 \le k \le 255$

Operation: k → W;
TOS → PC

Status Affected: None

Encoding:

11	01xx	kkkk	kkkk

Description: The W register is loaded with the eight bit literal 'k'. The program counter is loaded 13-bit address at the Top of Stack (the return address). This is a two cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

1st cycle:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

2nd cycle:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example

```
HERE    CALL TABLE   ; W contains table
                     ; offset value
        •            ; W now has table value
        •
        •
TABLE   ADDWF PC     ;W = offset
        RETLW k1     ;Begin table
        RETLW k2     ;
        •
        •
        •
        RETLW kn     ; End of table
```

Before Instruction
W = 0x07

After Instruction
W = value of k8
PC = TOS = Address `Here + 1`

© 1997 Microchip Technology Inc.

RETURN — Return from Subroutine

Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Encoding:	00 | 0000 | 0000 | 1000
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

1st cycle:

Q1	Q2	Q3	Q4
Decode	No operation	Process data	No operation

2nd cycle:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

Example

```
HERE    RETURN
```

After Instruction

PC = TOS

RLF — Rotate Left f through Carry

Syntax: [*label*] RLF f,d

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

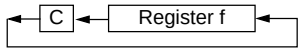
Operation: See description below

Status Affected: C

Encoding:

00	1101	dfff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1 `RLF REG1,0`

Before Instruction
```
REG1=  1110 0110
C   =   0
```
After Instruction
```
REG1=1110 0110
W   =1100 1100
C   =1
```

Example 2 `RLF INDF, 1`

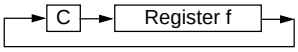
Case 1: Before Instruction
```
W   =  xxxx xxxx
FSR =  0xC2
Contents of Address (FSR) = 0011 1010
C   =  1
```
After Instruction
```
W   =  0x17
FSR =  0xC2
Contents of Address (FSR) = 0111 0101
C   =  0
```

Case 2: Before Instruction
```
W   =  xxxx xxxx
FSR =  0xC2
Contents of Address (FSR) = 1011 1001
C   =  0
```
After Instruction
```
W   =  0x17
FSR =  0xC2
Contents of Address (FSR) = 0111 0010
C   =  1
```


© 1997 Microchip Technology Inc.

RRF — Rotate Right f through Carry

Syntax:	[*label*] RRF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Encoding:	00 | 1100 | dfff | ffff
Description:	The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.



Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
RRF          REG1,0
```

Before Instruction

```
REG1= 1110 0110
W   =  xxxx xxxx
C   =  0
```

After Instruction

```
REG1= 1110 0110
W   =  0111 0011
C   =  0
```

Example 2

```
RRF     INDF, 1
```

Case 1:

Before Instruction

```
W   =  xxxx xxxx
FSR =  0xC2
Contents of Address (FSR) = 0011 1010
C   =  1
```

After Instruction

```
W   =  0x17
FSR =  0xC2
Contents of Address (FSR) = 1001 1101
C   =  0
```

Case 2:

Before Instruction

```
W   =  xxxx xxxx
FSR =  0xC2
Contents of Address (FSR) = 0011 1001
C   =  0
```

After Instruction

```
W   =  0x17
FSR =  0xC2
Contents of Address (FSR) = 0001 1100
C   =  1
```

SLEEP

Syntax: [*label*] SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT prescaler count,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

00	0000	0110	0011

Description: The power-down status bit, $\overline{PD}$ is cleared. Time-out status bit, $\overline{TO}$ is set. Watchdog Timer and its prescaler count are cleared.
The processor is put into SLEEP mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	Go to sleep

Example: SLEEP

Note: The SLEEP instruction does not affect the assignment of the WDT prescaler

© 1997 Microchip Technology Inc.

SUBLW — Subtract W from Literal

Syntax:	[*label*] SUBLW k
Operands:	$0 \le k \le 255$
Operation:	k - (W) → W
Status Affected:	C, DC, Z
Encoding:	11 | 110x | kkkk | kkkk
Description:	The W register is subtracted (2's complement method) from the eight bit literal 'k'. The result is placed in the W register.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example 1: `SUBLW 0x02`

Case 1: Before Instruction

```
W  =  0x01
C  =  x
Z  =  x
```

After Instruction

```
W  =  0x01
C  =  1          ; result is positive
Z  =  0
```

Case 2: Before Instruction

```
W  =  0x02
C  =  x
Z  =  x
```

After Instruction

```
W  =  0x00
C  =  1          ; result is zero
Z  =  1
```

Case 3: Before Instruction

```
W  =  0x03
C  =  x
Z  =  x
```

After Instruction

```
W  =  0xFF
C  =  0          ; result is negative
Z  =  0
```

Example 2 `SUBLW MYREG`

Before Instruction

W = 0x10

Address of `MYREG` † = 0x37

† `MYREG` is a symbol for a data memory location

After Instruction

```
W  =  0x27
C  =  1      ; result is positive
```

29

Instruction Set

SUBWF — Subtract W from f

Syntax:	[*label*] SUBWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - (W) → destination
Status Affected:	C, DC, Z
Encoding:	`00` `0010` `dfff` `ffff`
Description:	Subtract (2's complement method) W register from register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1: `SUBWF REG1,1`

Case 1: Before Instruction

```
REG1= 3
W   = 2
C   = x
Z   = x
```

After Instruction

```
REG1= 1
W   = 2
C   = 1        ; result is positive
Z   = 0
```

Case 2: Before Instruction

```
REG1= 2
W   = 2
C   = x
Z   = x
```

After Instruction

```
REG1= 0
W   = 2
C   = 1        ; result is zero
Z   = 1
```

Case 3: Before Instruction

```
REG1= 1
W   = 2
C   = x
Z   = x
```

After Instruction

```
REG1= 0xFF
W   = 2
C   = 0        ; result is negative
Z   = 0
```


© 1997 Microchip Technology Inc.

SWAPF — Swap Nibbles in f

Syntax:	[*label*] SWAPF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f<3:0>) → destination<7:4>, (f<7:4>) → destination<3:0>
Status Affected:	None
Encoding:	00 | 1110 | dfff | ffff
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0 the result is placed in W register. If 'd' is 1 the result is placed in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
SWAPF   REG, 0
```

Before Instruction

REG1= 0xA5

After Instruction

REG1= 0xA5
W = 0x5A

Example 2

```
SWAPF   INDF, 1
```

Before Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x20

After Instruction

W = 0x17
FSR = 0xC2
Contents of Address (FSR) = 0x02

Example 3

```
SWAPF   REG, 1
```

Before Instruction

REG1= 0xA5

After Instruction

REG1= 0x5A

TRIS — Load TRIS Register

Syntax:	[*label*] TRIS f
Operands:	$5 \le f \le 7$
Operation:	(W) $\rightarrow$ TRIS register f;
Status Affected:	None
Encoding:	00 | 0000 | 0110 | 0fff
Description:	The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.
Words:	1
Cycles:	1

Example

To maintain upward compatibility with future PIC16CXX products, do not use this instruction.

© 1997 Microchip Technology Inc.

XORLW — Exclusive OR Literal with W

Syntax:	[*label*] XORLW k
Operands:	$0 \le k \le 255$
Operation:	(W).XOR. k → W
Status Affected:	Z
Encoding:	11 | 1010 | kkkk | kkkk
Description:	The contents of the W register are XOR'ed with the eight bit literal 'k'. The result is placed in the W register.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W register

Example 1

```
XORLW   0xAF                ; 1010 1111   (0xAF)
Before Instruction          ; 1011 0101   (0xB5)
        W   =   0xB5        ; ---------   ------
After Instruction           ; 0001 1010   (0x1A)
        W   =   0x1A
        Z   =   0
```

Example 2

```
XORLW   MYREG
```

Before Instruction
W = 0xAF
Address of MYREG † = 0x37
† MYREG is a symbol for a data memory location

After Instruction
W = 0x18
Z = 0

Example 3

```
XORLW   HIGH (LU_TABLE)
```

Before Instruction
W = 0xAF
Address of LU_TABLE † = 0x9375
† LU_TABLE is a label for an address in program memory

After Instruction
W = 0x3C
Z = 0

XORWF — Exclusive OR W with f

Syntax: [*label*] XORWF f,d

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

Operation: (W).XOR. (f) → destination

Status Affected: Z

Encoding:

00	0110	dfff	ffff

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1

```
XORWF    REG, 1                ; 1010 1111   (0xAF)
```

Before Instruction ; 1011 0101 (0xB5)

```
         REG=  0xAF            ; ---------   ------
         W   =  0xB5           ; 0001 1010   (0x1A)
```

After Instruction

```
         REG=  0x1A
         W   =  0xB5
```

Example 2

```
XORWF    REG, 0                ; 1010 1111   (0xAF)
```

Before Instruction ; 1011 0101 (0xB5)

```
         REG=  0xAF            ; ---------   ------
         W   =  0xB5           ; 0001 1010   (0x1A)
```

After Instruction

```
         REG=  0xAF
         W   =  0x1A
```

Example 3

```
XORWF   INDF, 1
```

Before Instruction

```
         W   =  0xB5
         FSR =  0xC2
         Contents of Address (FSR) = 0xAF
```

After Instruction

```
         W   =  0xB5
         FSR =  0xC2
         Contents of Address (FSR) = 0x1A
```


© 1997 Microchip Technology Inc.