TABLE 9-2 PIC16FXX INSTRUCTION SET

Mnemonic,		Description	Cycles	14-Bit Opcode				Status	Notes	
Operano	ds			MSb			LSb	Affected		
BYTE-ORIENTED FILE REGISTER OPERATIONS										
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2	
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2	
CLRF	f	Clear f	1	00	0001	<u>l</u> fff	ffff	Z	24	
CLRW	-	Clear W	1	00	0001	0xxx	XXXX	Z		
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2	
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2	
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3	
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2	
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3	
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2	
MOVF	f, d	Move f	1	0.0	1000	dfff	ffff	Z	1,2	
MOVWF	f	Move W to f	1	0.0	0000	lfff	ffff		4	
NOP	-	No Operation	1	0.0	0000	0xx0	0000			
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	С	1,2	
RRF	f, d	Rotate Right f through Carry	1	0.0	1100	dfff	ffff	С	1,2	
SUBWF	f, d	Subtract W from f	1	0.0	0010	dfff	ffff	C,DC,Z	1,2	
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2	
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2	
	•	BIT-ORIENTED FILE REGIST	ER OPER	RATION	NS .					
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2	
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2	
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3	
		LITERAL AND CONTROL	OPERAT	IONS						
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z		
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z		
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk			
CLRWDT	-	Clear Watchdog Timer	1	0.0	0000	0110	0100	TO,PD		
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk			
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z		
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk			
RETFIE	-	Return from interrupt	2	0.0	0000	0000	1001			
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk			
RETURN	-	Return from Subroutine	2	0.0	0000	0000	1000			
SLEEP	-	Go into standby mode	1	0.0	0000	0110	0011	TO,PD		
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z		
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z		

Note 1: When an I/O register is modified as a function of itself (e.g., MOVF PORTB, 1), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

4: Die unterstrichenen Zeichen bei CLRF und MOVWF sind keine l (L) sondern 1 (eins)

^{2:} If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

^{3:} If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

9.1 <u>Instruction Descriptions</u>

Add Literal and W	ANDLW	AND Literal with W			
[labe] ADDLW k	Syntax:	[labe] ANDLW k			
0 ≤k ≤255	Operands:	0 ≤k ≤255			
$(W) + k \rightarrow (W)$	Operation:	(W) .AND. (k) \rightarrow (W)			
C, DC, Z	Status Affected:	Z			
11 111x kkkk kkkk	Encoding:	11 1001 kkkk kkkk			
The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.	Description:	The contents of W register are AND'ed with the eight bit literal 'k'. The result is placed in the W register.			
1	Words:	1			
1	Cycles:	1			
Q1 Q2 Q3 Q4	Q Cycle Activity:	Q1 Q2 Q3 Q4			
Decode Read Process Write to literal 'k' data W		Decode Read Process Write to data W			
ADDLW 0x15	Example	ANDLW 0x5F			
Before Instruction W = 0x10 After Instruction W = 0x25		Before Instruction W = 0xA3 After Instruction W = 0x03			
	$ [labe] \ ADDLW k \\ 0 \le k \le 255 \\ (W) + k $				

ADDWF	Add W and f	ANDWF	AND W with f			
Syntax:	[labe] ADDWF f,d	Syntax:	[labe] ANDWF f,d			
Operands:	$0 \le f \le 127$ $d \in [0,1]$	Operands:	$0 \le f \le 127$ $d \in [0,1]$			
Operation:	$(W) + (f) \rightarrow (destination)$	Operation:	(W) .AND. (f) \rightarrow (destination)			
Status Affected:	C, DC, Z	Status Affected:	Z			
Encoding:	00 0111 dfff ffff	Encoding:	00 0101 dfff ffff			
Description:	Add the contents of the W register with the contents of 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'.	Description:	AND the W register with contents of 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	Words:	1			
Cycles:	1	Cycles:	1			
Q Cycle Activity:	Q1 Q2 Q3 Q4	Q Cycle Activity:	Q1 Q2 Q3 Q4			
	Decode Read register data Write to destination		Decode Read Process Write to destination			
Example	ADDWF FSR, 0	Example	ANDWF FSR, 1			
	Before Instruction		Before Instruction			
	W = 0x17		W = 0x17			
	FSR = 0xC2		FSR = 0xC2			
	After Instruction W = 0xD9		After Instruction W = 0x17			
	FSR = 0xC2		FSR = 0x02			

BCF	Bit Clea	r f			BTFSC	Bit Test,	Skip if C	lear		
Syntax:	[labe] Bo	CF f,b			Syntax:	[labe] BTFSC f,b				
Operands:	0 ≤f ≤127 0 ≤b ≤7				Operands:	0 ≤f ≤127 0 ≤b ≤7				
Operation:	0 → (f <b< td=""><td>>)</td><td></td><td></td><td>Operation:</td><td>skip if (f<</td><td>(b>) = 0</td><td></td><td></td></b<>	>)			Operation:	skip if (f<	(b>) = 0			
Status Affected:	None				Status Affected:	None				
Encoding:	01	00bb	bfff	ffff	Encoding:	01	10bb	bfff	ffff	
Description:	Bit 'b' in re	egister 'f' i	s cleared.	!	Description:	If bit 'b' in	register 'f' is	s '1' then th	ne next	
Words:	1						is executed		the nevt	
Cycles:	1					If bit 'b', in register 'f', is '0' then the next instruction is discarded, and a NOP is				
Q Cycle Activity:	Q1	Q2	Q3	Q4		executed instruction	instead, ma	king this a	2Tcy	
	Decode	Read register	Process data	Write register 'f'	Words:	1	•			
		'f'			Cycles:	1(2)				
Example	BCF	FLAG	REG, 7		Q Cycle Activity:	Q1	Q2	Q3	Q4	
·	Before Ir			,		Decode	Read register 'f'	Process data	No-Operat ion	
	After Ins	_	EG = 0xC7	(If Skip:	(2nd Cycle)				
		FLAG_RI	EG = 0x47	•		Q1	Q2	Q3	Q4	
						No-Operat ion	No-Operati on	No-Opera tion	No-Operat ion	
					Example	HERE FALSE TRUE	BTFSC GOTO •	FLAG,1 PROCESS	CODE	
BSF	Bit Set f					After Inst		address H	IERE	

Syntax:	[labe] BS	SF f,b					
Operands:	0 ≤f ≤127 0 ≤b ≤7						
Operation:	1 → (f)						
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	BSF	FLAG_F	REG, 7				
	Before In		n EG = 0x0A				

After Instruction

FLAG_REG = 0x8A

PC =

if FLAG<1>=1, PC = addr

address TRUE

 $address \ \mathtt{FALSE}$

BTFSS	Bit Test f, Skip if Set	CALL	Call Subroutine
Syntax:	[labe] BTFSS f,b	Syntax:	[label] CALL k
Operands:	0 ≤f ≤127	Operands:	0 ≤k ≤2047
	0 ≤ b < 7	Operation:	(PC)+ 1→ TOS,
Operation:	skip if $(f < b >) = 1$		k → PC<10:0>,
Status Affected:	None	Ctatus Affastad	(PCLATH<4:3>) → PC<12:11>
Encoding:	01 11bb bfff ffff	Status Affected:	None
Description:	If bit 'b' in register 'f' is '0' then the next instruction is executed.	Encoding:	10 Okkk kkkk kkkk
	If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2TcY instruction.	Description:	Call Subroutine. First, 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.CALL
Words:	1		is a two cycle instruction.
Cycles:	1(2)	Words:	1
Q Cycle Activity:	Q1 Q2 Q3 Q4	Cycles:	2
	Decode Read Process No-Operat register 'f' data ion	Q Cycle Activity:	Q1 Q2 Q3 Q4
If Skip:	(2nd Cycle)	1st Cycle	Decode Read Process Write to literal 'k', data PC
•	Q1 Q2 Q3 Q4		Push PC to Stack
	No-Operat ion No-Operat tion No-Operat	2nd Cycle	No-Opera tion No-Opera tion No-Opera ion No-Opera
Example	HERE BTFSC FLAG,1 FALSE GOTO PROCESS CODE	Example	HERE CALL THERE
	TRUE •		Before Instruction
	•		PC = Address HERE After Instruction
	Before Instruction		PC = Address THERE
	PC = address HERE		TOS = Address HERE+1
	After Instruction		
	if FLAG<1> = 0, PC = address FALSE		
	if FLAG<1> = 1,		
	PC = address TRUE		

CLRF	Clear f				
Syntax:	[labe] C	LRF f			
Operands:	$0 \le f \le 12$	27			
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	CLRF	FLAG	G_REG		
	Before In	struction	1		

FLAG_REG

FLAG_REG = Z =

After Instruction

0x5A

0x00 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 set.	is cleared	. Zero bit (Z) is		
Words:	1					
Cycles:	1					
Q Cycle Activity:	Q1	Q2	Q3	Q4		
	Decode	No-Opera tion	Process data	Write to W		
Example	CLRW					
	Before In					
			0x5A			
	After Inst		0x00			
			1			

CLRWDT	Cloor W	atobdog '	Timor				
Syntax:	[label] CLRWDT						
Operands:	None						
Operation:	00h → WDT 0 → WDT prescaler, 1 → \overline{TO} 1 → \overline{PD}						
Status Affected:	$\overline{TO}, \overline{PD}$						
Encoding:	00	0000	0110	0100			
Description:	CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits TO and PD are set.						
Words:	1						
Cycles:	1						
Q Cycle Activity:	Q1	Q2	Q3	Q4			
	Decode	No-Opera tion	Process data	Clear WDT Counter			
Example	CLRWDT						
	Before In	struction	nter =	?			
	After Inst		iter =	0x00 0			
		TO PD	= =	1			

COMF	Complement f	DECFSZ	Decrement f, Skip if 0				
Syntax:	[label] COMF f,d	Syntax:	[label] DECFSZ f,d				
Operands:	$0 \le f \le 127$ $d \in [0,1]$	Operands:	$0 \le f \le 127$ $d \in [0,1]$				
Operation:	$(\bar{f}) \rightarrow (destination)$	Operation:	(f) - 1 → (destination);				
Status Affected:	Z		skip if result = 0				
Encoding:	00 1001 dfff ffff	Status Affected:	None				
Description:	The contents of register 'f' are complemented. If 'd' is 0 the result is stored in	Encoding:	00 1011 dfff ffff				
	W. If 'd' is 1 the result is stored back in register 'f'.	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				
Words:	1	If the result is not 0	back in register 'f'. If the result is 1, the next instruction, is				
Cycles: Q Cycle Activity:	1 Q1 Q2 Q3 Q4		executed. If the result is 0, then a NOP is executed instead making it a 2Tcy instruction.				
	Decode Read Process Write to	Words:	1				
	register data destination	Cycles:	1(2)				
		Q Cycle Activity:	Q1 Q2 Q3 Q4				
Example	COMF REG1, 0		Decode Read Process Write to register 'f' data destination				
	Before Instruction REG1 = 0x13	If Claim.					
	After Instruction	If Skip:	(2nd Cycle) Q1 Q2 Q3 Q4				
	REG1 = 0x13 W = 0xEC		No-Opera No-Operati				
			No-Operat tion ion on				
DECF	Decrement f		1011				
Syntax:	Decrement f [labe] DECF f,d	Example	HERE DECFSZ CNT, 1				
		Example					
Syntax:	[labe] DECF f,d 0 ≤f ≤127	Example	HERE DECFSZ CNT, 1 GOTO LOOP				
Syntax: Operands:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction				
Syntax: Operands: Operation:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ (f) - 1 \Rightarrow (destination)	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • •				
Syntax: Operands: Operation: Status Affected:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ (f) - 1 → (destination) Z $00 0011 dfff ffff$ Decrement contents of register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0,				
Syntax: Operands: Operation: Status Affected: Encoding:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ (f) - 1 → (destination) Z $00 0011 dfff ffff$ Decrement contents of register 'f'. If 'd' is 0 the	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1				
Syntax: Operands: Operation: Status Affected: Encoding: Description:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ $(f) - 1 → (destination)$ Z $ 00 $	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ (f) - 1 → (destination) Z 00 0011 dfff ffff Decrement contents of 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'. 1	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words:	[labe] DECF f,d $0 \le f \le 127$ $d \in [0,1]$ (f) - 1 → (destination) Z $00 0011 dfff ffff$ Decrement contents of 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'. 1	Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles: Q Cycle Activity:		Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles:		Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles: Q Cycle Activity:		Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles: Q Cycle Activity:		Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				
Syntax: Operands: Operation: Status Affected: Encoding: Description: Words: Cycles: Q Cycle Activity:		Example	HERE DECFSZ CNT, 1 GOTO LOOP CONTINUE • • • Before Instruction PC = address HERE After Instruction CNT = CNT - 1 if CNT = 0, PC = address CONTINUE if CNT ≠ 0,				

GOTO	Uncondi	tional Br	anch		INCF	Increme	nt f				
Syntax:	[label]	GOTO	k		Syntax:	[label] INCF f,d					
Operands:	0 ≤ k ≤ 2	047			Operands:	0 ≤f ≤127					
Operation:	k → PC<10:0>						d ∈ [0,1]				
	PCLATH<4:3>→ PC<12:11>				Operation:	(f) + 1 → (destination)					
Status Affected:	None				Status Affected:	Z	Z				
Encoding:	10	1kkk	kkkk	kkkk	Encoding:	00	1010	dfff	ffff		
Description:	eleven bit into PC bit PC are loa	n uncondition uncondition uncondition in unconditio	value is lo The upper PCLATH<	aded bits of	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				Words:	1					
Cycles:	2				Cycles:	1					
Q Cycle Activity:	Q1	Q2	Q3	Q4	Q Cycle Activity:	Q1	Q2	Q3	Q4		
1st Cycle	Decode	Read literal 'k'	Process data	Write to PC		Decode	Read register 'f'	Process data	Write to destination		
2nd Cycle	No-Operat ion	No-Operat ion	No-Opera tion	No-Operat ion	Example	INCF	CNT,	1			
						Before Ir	struction	า			
Example	GOTO T	HERE					CNT 7	= 0xF	F		
	After Inst					After Ins	_	= 0			
		PC =	Address	THERE			CNT	= 0x0	0		
							Z	= 1			

INCFSZ	Increment f, Skip if 0	IORLW	Inclusive OR Literal with W
Syntax:	[label] INCFSZ f,d	Syntax:	[label] IORLW k
Operands:	0 ≤f ≤127	Operands:	0 ≤k ≤255
	$d \in [0,1]$	Operation:	(W) .OR. k → (W)
Operation:	(f) + 1 → (destination), skip if result = 0	Status Affected:	Z
Status Affected:	None	Encoding:	11 1000 kkkk kkkk
Encoding:	00 1111 dfff ffff	Description:	The contents of the W register is OR'ed with the eight bit literal 'k'. The
Description:	The contents of register 'f' are incre-		result is placed in the W register.
2 000	mented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is	Words:	1
If the result is not 0	placed back in register 'f'. If the result is 1, the next instruction is	Cycles:	1
	executed. If the result is 0, a NOP is executed instead making it a 2TCY instruction.	Q Cycle Activity:	Q1 Q2 Q3 Q4
Words:	1		Decode Read Process Write to literal 'k' data W
Cycles:	1(2)		
Q Cycle Activity:	Q1 Q2 Q3 Q4	Example	IORLW 0x35
	Decode Read Process Write to register 'f' data destination		Before Instruction W = 0x9A After Instruction
If Skip:	(2nd Cycle)		W = 0xBF
	Q1 Q2 Q3 Q4		Z = 1
	No-Operat tion No-Opera tion No-Operation		Z = 0 !!!!
Example	HERE INCFSZ CNT, 1 GOTO LOOP CONTINUE •		
	Before Instruction PC = address HERE After Instruction CNT = CNT + 1 if CNT= 0, PC = address CONTINUE if CNT ≠ 0, PC = address HERE +1		

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) \rightarrow (destination)					
Status Affected:	ℤ Ζ!!!					
Encoding:	00 0100 dfff ffff					
Description:	Inclusive OR the W register with contents of 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 Process Write to destination					
Example	IORWF RESULT, 0					
	Before Instruction RESULT = 0x13 W = 0x91					
	After Instruction					
	RESULT = 0x13 W = 0x93					
	VV = 0x93 $Z = 1$					
	Z = 0 !!!!					

MOVLW	Move Lit	teral to V	V			
Syntax:	[label]	MOVLW	/ k			
Operands:	0 ≤k ≤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		
Example	MOVLW	0x5A				
	After Instruction					
		W =	0x5A			

MOVF	Move f						
Syntax:	[label]	[label] MOVF f,d					
Operands:	$0 \le f \le 12$ $d \in [0,1]$	$0 \le f \le 127$ $d \in [0,1]$					
Operation:	(f) → (de:	stination)				
Status Affected:	Z						
Encoding:	00	1000	dfff	ffff			
Description:	The contents of register f is moved to a destination dependant 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	MOVF	FSR,	0				
	After Inst	ruction	e in FSR re	egister			

MOVWF	Move W to f					
Syntax:	[label]	[label] MOVWF f				
Operands:	0 ≤f ≤127					
Operation:	$(W) \rightarrow (f)$					
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		
(system y	Decode	Read register 'f'	Process data	Write register 'f'		
Example	MOVWF	OPTIC	ON_REG			
	Before Instruction OPTION = 0xFF W = 0x4F After Instruction OPTION = 0x4F W = 0x4F					

NOP	No Operation					
Syntax:	[label]	[label] NOP				
Operands:	None					
Operation:	No operation					
Status Affected:	None					
Encoding:	00	0000	0xx0	0000		
Description:	No operati	ion.				
Words:	1					
Cycles:	1					
Q Cycle Activity:	Q1	Q2	Q3	Q4		
	Decode	No-Opera tion	No-Opera tion	No-Operat ion		
Example	NOP					

OPTION	Load Option Register			
Syntax:	[label] OPTION			
Operands:	None			
Operation:	(W) → OPTION			
Status Affected:	None			
Encoding:	00 0000 0110 0010)		
Description: Words: Cycles: Example	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 addres it.	;		
Zianipio	To maintain upward compatibilit with future PIC16CXX products, do not use this instruction.	-		

RETFIE	Return from Interrupt						
Syntax:	[label]	RETFIE					
Operands:	None	None					
Operation:	TOS → F 1 → GIE	TOS → PC, 1 → GIE					
Status Affected:	None						
Encoding:	00 0000 0000 1001						
Description:	Return from Interrupt. Stack is POPed and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a two cycle instruction.						
Words:	1						
Cycles:	2						
Q Cycle Activity:	Q1	Q2	Q3	Q4			
1st Cycle	Decode	No-Opera tion	Set the GIE bit	Pop from the Stack			
2nd Cycle	No-Operat ion	No-Opera tion	No-Opera tion	No-Operat ion			

Example RETFIE

After Interrupt PC = GIE =

PC = TOS GIE = 1

RETLW	Return v	vith Liter	al in W		_	RETURN	Return f	rom Sub	routine	
Syntax:	[label]	RETLW	k		ı.	Syntax:	[label]	RETURI	N	
Operands:	0 ≤ k ≤ 2!	55				Operands:	None			
Operation:	k → (W);					Operation:	TOS → F	C		
	TOS → F	PC .				Status Affected:	None			
Status Affected:	None				,	Encoding:	0 0	0000	0000	1000
Encoding:	11	01xx	kkkk	kkkk		Description:	Return fro	m subrout	ine. The st	ack is
Description:		΄. The pro	gram coun	ter is			is loaded i	nto the pro	•	` ,
	loaded fro return add			•		Words:	is a two cy	cie iristruc	LIOH.	
	instruction	,		,			1			
Words:	1					Cycles:	2	00	00	0.4
Cycles:	2					Q Cycle Activity:	Q1	Q2	Q3	Q4
Q Cycle Activity:	Q1	Q2	Q3	Q4		1st Cycle	Decode	No-Opera tion	No-Opera tion	Pop from the Stack
1st Cycle	Decode	Read literal 'k'	No-Opera tion	Write to W, Pop from the Stack		2nd Cycle	No-Operat ion	No-Opera tion	No-Opera tion	No-Opera tion
2nd Cycle		No-Opera	No-Opera	No-Operat		Example	RETURN			
•	No-Operat ion	tion	tion	ion		Ехатріс	After Inte	rrunt		
	L			I	I			-	TOS	
Example	CALL TABL	;offse	ntains tab et value ow has tab							
TABLE	ADDWF PC RETLW k1 RETLW k2	;W = oi;Begin;								
	RETLW kn		of table							
	Before In									
	After Inst		0x07							
		W =	value of k	3						

RLF	Rotate L	eft f thre	ough Ca	rry	RRF	Rotate F	Right f th	rough C	arry
Syntax:	[label]		RLF f	,d	Syntax:	[label]	RRF f	,d	
Operands:	$0 \le f \le 12$ $d \in [0,1]$				Operands:	$0 \le f \le 12$ $d \in [0,1]$			
Operation:	See desc	cription b	elow		Operation:	See des	cription b	elow	
Status Affected:	С				Status Affected:	С			
Encoding:	0.0	1101	dfff	ffff	Encoding:	0.0	1100	dfff	ffff
Description:	The conte one bit to Flag. If 'd' W register back in re	the left thing is 0 the refunction. If 'd' is 1 gister 'f'.	rough the esult is pla	Carry aced in the t is stored	Description:	one bit to Flag. If 'd'	the right t is 0 the re r. If 'd' is 1	ister 'f' are hrough the esult is pla . the result Register f	e Carry ced in the is placed
Words:	1				Words:	1			
Cycles:	1				Cycles:	1			
Q Cycle Activity:	Q1	Q2	Q3	Q4	Q Cycle Activity:	Q1	Q2	Q3	Q4
	Decode	Read register 'f'	Process data	Write to destination		Decode	Read register 'f'	Process data	Write to destination
Example	RLF	REC	G1,0		Example	RRF		REG1,0	
	Before Ir		า			Before Ir	nstruction	า	
		REG1 C	= 111 = 0	.0 0110			REG1		0 0110
	After Inst	_	= 0			After Ins	C	= 0	
		REG1	= 111	.0 0110		, (1101 1113	REG1	= 111	0 0110
		W		00 1100			W		1 0011
		С	= 1				С	= 0	

SLEEP

Syntax: [label] SLEEP

Operands: None

Operation: $00h \rightarrow WDT$,

0 → WDT prescaler,

 $1 \to \overline{TO}, \\ 0 \to \overline{PD}$

Status Affected: TO, PD

Encoding: 00

Description: The power-down status bit, \overline{PD} is

cleared. Time-out status bit, TO is set. Watchdog Timer and its pres-

0000

caler are cleared.

The processor is put into SLEEP mode with the oscillator stopped. See

0110

0011

Section 14.8 for more details.

Words: 1
Cycles: 1

Q Cycle Activity:

 Q1
 Q2
 Q3
 Q4

 Decode
 No-Opera tion
 No-Opera tion
 Go to Sleep

Example: SLEEP

SUBLW Subtract W from Literal

Syntax: [label] SUBLW k

Operands: $0 \le k \le 255$ Operation: $k - (W) \rightarrow (W)$

Status Affected: C, DC, Z

Encoding: 11 110x kkkk kkkk

Description: The contents of W register is subtracted (2's comple-

ment 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 data

Example 1: SUBLW 0x02

Before Instruction

W = 1 C = ? Z = ?

After Instruction

W = 1

C = 1; result is positive

Z = 0

Example 2: Before Instruction

W = 2 C = ? Z = ?

After Instruction

W = 0

C = 1; result is zero

1

7 =

Example 3: Before Instruction

W = 3 C = ?

Z = ?

After Instruction

W = 0xFF

C = 0; result is negative

Z = 0

SUBWF	Subtract	W from f	•				
Syntax:	[label]	SUBWF	f,d				
Operands:	$0 \le f \le 12$ $d \in [0,1]$	$0 \le f \le 127$ $d \in [0,1]$					
Operation:	(f) - (W) -	(destina	ition)				
Status Affected:	C, DC, Z						
Encoding:	00	0010	dfff	ffff			
Description:	Subtract (2's complement method) contents of 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				
	Before In:						
	REG1 W C Z	C = ?					
	After Insti	ruction					
	REG1		1				
	W C	=	2	no citivo			
	Z	=	1; result is 0	positive			
Example 2:	Before In:	struction					
	REG1		2				
	W C	= =	2 ?				
	Z	=	?				
	After Insti	ruction					
	REG1 W	. =	0				
	C Z	= =	1; result is	zero			
Example 3:	Before In:	struction					
	REG1		1				
	W C	=	2				
	Z	=	?				
	After Insti	ruction					
	REG1 W		0xFF				
	VV C	=	2 0; result is	negative			
	Z	=	0				

SWAPF	Swap Nibbles in f						
Syntax:	[label]	[label] SWAPF f,d					
Operands:	$0 \le f \le 127$ $d \in [0,1]$						
Operation:	(f<3:0>) · (f<7:4>) ·	→ (destin → (destin		,			
Status Affected:	None						
Encoding:	0.0	1110	dfff	ffff			
Description:	The upper and lower nibbles of contents 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	SWAPF	REG,	0				
	Before In	struction					
		REG1	= 0:	xA5			
	After Inst	ruction					
		REG1 W	-	xA5 x5A			

TRIS	Load TRIS Register			
Syntax:	[labe] TRIS f			
Operands:	5 ≤f ≤7			
Operation:	(W) → TRIS register f;			
Status Affected:	None			
Encoding:	00 0000 0110 Offf			
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.			

XORLW	Exclusive OR Literal with W	XORWF	Exclusive OR W with f
Syntax:	[labe] XORLW k	Syntax:	[labe] XORWF f,d
Operands:	0 ≤k ≤255	Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation: Status Affected: Encoding: Description:	(W) .XOR. k → (W) Z 11 1010 kkkk kkkk The contents of the W register are XOR'ed with the eight bit literal 'k'. The result is placed in the W regis-	Operation: Status Affected: Encoding: Description:	(W) .XOR. (f) → (destination) Z
Words: Cycles: Q Cycle Activity:	ter. 1 Q1 Q2 Q3 Q4 Decode Read Process Write to data W	Words: Cycles: Q Cycle Activity:	1 the result is stored back in register 'f'. 1 Q1 Q2 Q3 Q4 Decode Read Process Write to data destination
Example:	XORLW 0xAF Before Instruction W = 0xB5 After Instruction W = 0x1A	Example	XORWF REG 1 Before Instruction REG = 0xAF W = 0xB5 After Instruction REG = 0x1A