



SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Sub: Microcontroller Programming And Interfacing Subcode:23ECB202 Unit-II

INSTRUCTION SET/ Arithmetic Instructions

4/7/2025





Instruction Set in PIC16Cxx MC Family

- Complete set: 35 instructions.
- MC Architecture: RISC microcontroller.
- Instruction Types:
- 1. Data Processing Operations:
 - Copy data between registers.
 - Manipulate data in a single register.
 - Arithmetic operations.
 - Logic operations.

2. **Program Sequence Control Operations:**

- Unconditional Jump.
- Conditional Jump.
- Call.
- Control.





Word list

- f any memory location in a microcontroller
- W work register
- b bit position in 'f' register
- d destination bit
- *label* group of eight characters which marks the beginning of a part of the program
- **TOS** top of stack
- [] option
- > bit position inside register

Example:







Data transfer

Transfer of data in a MC is done between W register and an 'f' register.

Mnemoria	2	Description	Operation	Fleg	Cycle	Notes
		Data transfer	2 .			
MOVEW	k	Move constant to W	$\mathbf{k} \rightarrow \mathbf{W}$		1	
MOVWE	f	Move W to f	$W \rightarrow f$		1	10 1
MOVE	f, d	Move f	$\mathbf{f} \rightarrow \mathbf{d}$	Z	1	1,2
CERW	2 ⁸⁰	Clear W	0 → W	Z	1	
CLRF	f	Clear f	0 → f	Z	1	2
SVVAPF	f, d	Swap nibbles in f	$f(7:4), (3:0) \rightarrow f(3:0), (7:4)$	1	1 9	1,2

These instructions provide for:

- a constant being written in W register (MOVLW)
- data to be copied from W register onto RAM.
- data from RAM to be copied onto W register (or on the same RAM location, at which point only the status of Z flag changes).
- -Instruction CLRF writes constant 0 in 'f' register,
- Instruction CLRW writes constant 0 in register W.
- SWAPF instruction exchanges places of the 4-bit nibbles field inside a register.





Arithmetic and logic

PIC like most MCs supports only subtraction and addition.

Flags C, DC and Z are set depending on a result of addition or subtraction. Logic unit performs AND, OR, EX-OR, complement (COMF) and rotation (RLF & RRF).

ADDLW	k	Add constant and W	$W+1 \rightarrow W$	C,DC,Z	1	1
ADDWF	f, d	Add W and f	W+f→ d	C,DC,Z	11	1,2
SUBLW	k	Subtract W from constant	$W-k \rightarrow W$	C,DC,Z	1	
SUBWF	f, d	Subtract W from f	W-f→d	C,DC,Z	1	1,2
ANDLW	k	AND constant with W	W.AND.k→W	Z	1	100
ANDWF	f, d	AND W with f	W.AND.f→d	Z	1	1,2
IORLW	k	OR constant with W	$W.OR.k \rightarrow W$	Z	1	- 19
IORWF	f, d	OR W with f	W.OR.f→d	Z	1	1,2
XORLW	k	Exclusive OR constant with W	W.XOR.k→ W	Z	1	1,2
XORWF	f, d	Exclusive OR W with f	W.XOR.f→ d	Z	1	
INCF	f,d	Increment f	$f+1 \rightarrow f$	Z	1	1/2
DECF	f, d	Decrement f	f-l → f	Z	1	1,2
RLF	f, d	Rotate Left f trough carry		C	1	14
RRF	f,d	Rotate Right f trough carry		C	1	1,
COMF	f, d	Complement f	F→d	Z	1	1,





Bit operations

Instructions BCF and BSF do setting or cleaning of one bit anywhere in the memory.

The CPU first reads the whole byte, changes one bit in it and then writes in the entire byte at the same place.

Bit operations

BCF	f, b	Bit Clear f	0 → f(b)	1	1,2
BSF	f,b	Bit Set f	l → f(b)	- 34	1,2





Directing a program flow

- Instructions GOTO, CALL and RETURN are executed the same way as on all other microcontrollers, only stack is independent of internal RAM and limited to eight levels.
- 'RETLW k' instruction is identical with RETURN instruction, except that before coming back from a subprogram a constant defined by instruction operand is written in W register.

				17 17	
BTFSC	f,b	Bit Test f, Skip if Clear	jump it f(b)=0	1 (2)	3
BTFSS	f,b	Bit Test f, Skip if Set	jump it f(b)=1	1 (2)	3
DECFSZ	f,d	Decrement f, Skip if 0	f-l → d, jump if Z=l	1(2)	1,2,3
INCFSZ	f,d	Increment f, Skip if 0	f+1 → d, jump ifZ=0	1(2)	1,2,3
GOTO	k	Go to address	$W.AND.k \rightarrow W$	2	
CALL	k	Call subroutine	W.AND.f→d	2	
RETURN	8	Return from Subroutine	$W.OR.k \rightarrow W$	2	
RETLW	k	Return with constant in W	W.OR.f→d	2	
RETFIE	i i	Return from interrupt	$W.XOR.k \rightarrow W$	2	

Directing a program flow



Look-up tables Design:

- This instruction 'RETLW k' enables us to design easily the Look-up tables (lists).
- We use them by determining data position on our table adding it to the address at which the table begins, and then we read data from that location (which is usually found in program memory).

•Table can be formed as a subprogram which consists of a series of 'RETLW k' instructions, where 'k' constants are members of the table.

•We write the position of a member of our table in W register, and using CALL instruction to call a subprogram which creates the table.

• The instruction ADDWF PCL, f adds the position of a W register member to the starting address of our table, found in PCL register, and so we get the real data address in program memory.

•When returning from a subprogram we will have in W register the contents of an addressed table member.

Other instructions

Main	molov 2
	call Lookup
	I
Lookup	addwf PCL, f
	retlw k
	retlw k1
	retlw k2
	:
	:
	retlw kn

					Y 1	
NOP	10	No Operation			1	ĺ
CLRWDT	19. 19.	Clear Watchdog Timer	0 → WDT,1→TO,1 → PD	TO, PD		
SLEEP	32	Go into standby mode	0 → WDT, HTO,0 → PD	TO,PD	1	





Instruction Execution Period

All instructions are executed in one cycle except for conditional branch instructions if condition was true, or if the contents of program counter was changed by some instruction. In that case, execution requires two instruction cycles, and the second cycle is executed as NOP (No Operation). Four oscillator clocks make up one instruction cycle. If we are using an oscillator with 4MHz frequency, the normal time for executing an instruction is 1 μ s, and in case of conditional branching, execution period is 2 μ s.





Operation: Operand: Flag: Number of words: Number of cycles: Example 1 MOVLV After instruction: Example 2 MOVLV	1 V O×5A W=O×5A V REGISTAR	Operand: Flag: Number of words: Number of cycles: Example 1 MOVW Before instruction: After instruction: Example 2 MOVW	1 F OPTION_REG OPTION_REG=0×20 W=0×40 OPTION_REG=0×40 W=0×40 F INDF W=0×17 FSR=0×C2
	W=0×10 and REGISTAR=0×40 W=0×40	After instruction:	FSR=0×C2 address contents 0×C2=0×00 W=0×17 FSR=0×C2 address contents 0×C2=0×17





Syntax: Description:	[<i>label</i>] MOVF f , d Contents of f register is stored in location determined by d operand. If d=0 , destination is W register. If d=1 , destination is f register itself. Option d=1 is used for testing the contents of f register because execution of this instruction affects Z flag in STATUS register.
Operation: Operand:	$ \begin{aligned} \mathbf{f} &\Rightarrow (\mathbf{d}) \\ 0 &\leq \mathbf{f} \leq 127 \\ \mathbf{d} &\in [0,1] \end{aligned} $
Flag: Number of words: Number of cycles:	Z 1
Example 1 MOVF	FSR, O
Before instruction: After instruction:	FSR=0×C2 W=0×00 W=0×C2 Z=0
Example 2 MOVF	INDF, O
Before instruction:	W=0×17 FSR=0×C2 address contents 0×C2=0×00
After instruction:	W=0×17 FSR=0×C2 address contents 0×C2=0×00 Z=1





Syntax: Description:	[<i>label</i>] MOVF f , d Contents of f register is stored in location determined by d operand. If d=0 , destination is W register. If d=1 , destination is f register itself. Option d=1 is used for testing the contents of f register because execution of this instruction affects Z flag in STATUS register.
Operation: Operand:	
Flag: Number of words: Number of cycles:	
Example 1 MOVF	FSR, O
Before instruction:	FSR=0×C2 W=0×00
After instruction:	W=0×00 W=0×C2 Z=0
Example 2 MOVF	INDF, O
Before instruction:	W=0×17 FSR=0×C2
After instruction:	address contents 0×C2=0×00 W=0×17 FSR=0×C2 address contents 0×C2=0×00 Z=1
4/7/2025	23ECB202/ Instruction Set/ Dr. Husna/ ECE/SNSCE





Syntax:	[<i>label</i>] CLRW	Syntax:	[label] CRLF f
Description:	Contents of W register evens out to zero, and Z flag in STATUS register is set to one.	Description:	Contents of 'f' register evens out to zero, and Z flag in status register is set to one.
Operation:	$0 \Rightarrow (\mathbf{W})$		
Operand:	<u>.</u>	Operation:	$0 \Rightarrow f$
Flag:	Z	Operand:	0 ≤ f ≤ 127
Number of words:	1	Flag:	Z
Number of cycles:	1	Number of words: Number of cycles:	(1353)
Example CLRW			94594 94594
		Example 1 CRLF	STATUS
Before instruction:	W=0×55	er w sa	
After instruction:	W=0×00	Before instruction:	STATUS=0×C2
	Z=1	After instruction:	STATUS=0×00
	Z=1		Z=1
Example 2 CLRF	INDF	Example 2 CLRF	INDF
Before instruction:	FSR=0×C2	Before instruction:	FSR=0×C2
	address contents 0×C2=0×33	2000 30 30 40	address contents 0×C2=0×33
After instruction:	FSR=0×C2	After instruction:	FSR=0×C2
	address contents 0×C2=0×00		address contents 0×C2=0×00
	Z=1		Z=1





Syntax: Description:	[<i>label</i>] SWAPF f , d Upper and lower half of f register exchange places. If d=0 , result is stored in W register. If d=1 , result is stored in f register.
Operation: Operand:	$f < 0: 3 > \Rightarrow d < 4: 7 >, f < 4: 7 > \Rightarrow d < 0: 3 >;$ $0 \le f \le 127$ $d \in [0, 1]$
Flag: Number of words: Number of cycles:	1
Example 1 SWAP	REG, O
Before instruction: After instruction:	
Example 2 SWAP	REG, 1
Before instruction: After instruction:	REG=0×F3 REG=0×3F





Syntax: Description:	[<i>label</i>] ADDLW k Contents of W register is added to 8-bit constant k and result is stored in W register.	Syntax: Description:	[<i>label</i>] ADDWF f , d Add contents of register W to register f . If d=0 , result is stored in W register. If d=1 , result is stored in f register.
Operation:	$(W) + k \Rightarrow W$	Operation:	$(W) + (f) \Rightarrow d$
101200000000000000000000000000000000000	0 ≤ k ≤ 255	i i i i i i i i i i i i i i i i i i i	$\mathbf{d} \in [0,1]$
		Operand:	0 ≤ f ≤ 127
Flag:	C, DC, Z	Flag:	C, DC, Z
Number of words	: 1	Number of words:	779.059
Number of cycles	: 1	Number of cycles:	1
2012/04/2012/02/05/2012/2012 21		Example 1 ADDW	/F FSR, 0
Example 1 ADDL	W 0×15	Before instruction:	27120276 36 251017
Before instruction:	W≐0×10	After instruction:	FSR=0×C2 W=0×D9
장애장 이 것 같아? 옷 것 같아? 것 같아?	W=0×25		FSR=0×C2
	ien stuette NV 201	Example 2 ADDLV	W INDF, 1
Example 2 ADDL	Warega	Before instruction:	W=0×17
5 D R D (05			FSR=0×C0
Before instruction:	W=U×1U	10115 ES 18 19	address contents 0×C2=0×20
	register contents REG=0×37	After instruction:	W=0×17
After instruction:	W=0×47		FSR=0×C2
ALCONDUCTOR			address contents 0×C2=0×37





Syntax: Description:	[<i>label</i>] SUBLW k Contents of W reaister is	subtracted from	Syntax: Description:	[<i>label</i>] SUBWF f , d Contents of W register is sub	tracted from
bosonption	k constant, and result is		Description.	- 역상 관망 방송 것이 이렇게 집에 있다. 방법으로	
Operation:	$\mathbf{k} - (\mathbf{W}) \Rightarrow \mathbf{W}$			the contents of f register.	
Operand:	0 ≤ k ≤ 255			If d=0 , result is stored in W	
Flag:	C, DC, Z			If d=1 , result is stored in f re	egister.
Number of words:			Operation:	$(f) - (W) \Rightarrow d$	
Number of cycles:	1		Operand:	$0 \le f \le 127$	
<u> </u>				$\mathbf{d} \in [0,1]$	
Example 1 SUBLY	N 0×03		Flag:	C, DC, Z	
Before instruction:	W=0×01, C=×, Z=×		Number of words:	: 1	
After instruction:	W=0×02, C=1, Z=0	Result > 0	Number of cycles:	: 1	
Before instruction:	W=0×03, C=×, Z=×		Example 1 SUBW	/F REG, 1	
After instruction:	W=0×00, C=1, Z=1	Result = 0			
			Before instruction:	REG=3, W=2, C=×, Z=×	
Before instruction:	W=0×04, C=×, Z=×		After instruction:	REG=1, W=2, C=1, Z=0	Result > 0
After instruction:	W=0×FF, C=0, Z=0	Result < 0		19999 61119 53955 5665966	
2			Before instruction:	REG=2, W=2, C=×, Z=×	
Example 2 SUBLY	N REG		After instruction:	REG=0, W=2, C=1, Z=1	Result = 0
Before instruction:	W=0×10			Antonina Partina como en anterio 1970.	
	contents REG=0×37		Before instruction:	REG=1, W=2, C=×, Z=×	
After instruction:	W=0×27		After instuction:	REG=0×FF, W=2, C=0, Z=0	Result < 0
	C=1	Result > 0			





Syntax:	[label] AND	LW k		Syntax:	[<i>label</i>] ANDWF f , d		
Description:	contents of	peration logic W register an pred in W regi	id constant k .	Description:	Performs operation the contents of W a If d=0, result is sto	nd f registers red in W regis	ster.
Operation:	(W) .AND.	k ⇒ W	54.194.195.05	Operation:	If $d=1$, result is sto $(w) \rightarrow d$	reu in i regist	er,
Operand:	$0 \le \mathbf{k} \le 255$			Operand:	(W).AND. f ⇒ d 0≤f≤127		
Flaq:	Z			operanu.	$d \in [0,1]$		
Number of words:	1			Flaq:	7		
Number of cycles:	1			Number of words:	1		
				Number of cycles:			
Example 1 ANDLV	V O×5F				05		
				Example 1 ANDW	'F FSR, 1		
Before instruction:	W=0×A3	; 0101 1111	(0×5F)				1896 - 68532
After instruction:	W=0×03	; 1010 0011	(0×A3)	Ph. P. 44 365 251 252 35	V=0×17, FSR=0×C2		
				After instruction: V	V=0×17, FSR=02	; 1100 0010	(0×C2)
		; 0000 0011	(0×03)		8		
						; 0000 0010	(UXU2)
Example 2 ANDLV	V REG			Example 2 ANDW	F FSR, O		
Before instruction:	W-0×43	; 1010 0011	(0×A3)	Before instruction: V	V=0×17, FSR=0×C2	• 0001 0111	(0×17)
Defore instruction.	111 W 111W	; 0011 0111		- 전화 같은 것은 것은 것은 것은 것을 했다.	V=0×02, FSR=0×C2	요즘 물건이 이야지도 다 안 되어 있는 것이 같다.	(0×C2)
After instruction:	W=0×23		(,=0.02,100.000		
Anterninstruction	W-UAZ3	; 0010 0011	(0×23)			; 0000 0010	(0×02)





Syntax:	[label] IORLW k	Syntax:	[label] IORWF f, d
Description:	Operation logic OR is performed over the contents of W register and over 8-bit constant k , and result is stored in W register.	Description:	Operation logic OR is performed over the contents of W and f registers. If d=0 , result is stored in W register If d=1 , result is stored in f register.
Operation:	(W) .OR. $(k) \Rightarrow W$	Operation:	(W) .OR. $(f) \Rightarrow d$
Operand:	$0 \le \mathbf{k} \le 255$	Operand:	0 ≤ f ≤ 127 d ∈ [0,1]
Flag: Number of words:	1	Flaq:	Z
Number of cycles:		Number of words: Number of cycles:	
Example 1 IORLV	V 0×35	Example 1 IORW	F REG, O
Before instruction: After instruction:	W=0×9A W=0×BF Z=0	A1007 360 301 302	REG=0×13, W=0×91 REG=0×13, W=0×93 Z=0
Example 2 IORLV	V REG	Example 2 IORW	F REG, 1
Before instruction:	W=0×9A contenst REG=0×37	(1943) 20 10 20 E	REG=0×13, W=0×91
After instruction:	W=0×9F Z=0	After instruction:	REG=0×93, W=0×91 Z=0





Syntax:	[label] XORI	LW k		Syntax:	[label] XORWF f, d		
Description:	Operation exclusive OR (XOR) is done over the contents of W		Description:	Operation exclusive OR is performed over the contents of W and f registers.		d over	
	register and	d constant k , bred in W regi	and		If d=0, result is store If d=1, result is store		
Operation:	(W) .XOR.	See State - and a second	SI BRIA	Operation:	$(W) \; . \text{XOR.} \; (f) \Rightarrow d$		
Operand:	0 ≤ k ≤ 255			Operand:	0 ≤ f ≤ 127		
Flag:	7			5	$\mathbf{d} \in [0,1]$		
Number of words:	1			Flag:	Z		
Number of cycles:	23370			Number of words:			
Number of cycles.	1			Number of cycles:	1		
Example 1 XORLV	V O×AF			Example 1 XORW	F REG, 1		
Before instruction:	W=0×B5	; 1010 1111	(O×AF)	Before instruction:	REG=0×AF, W=0×B5	; 1010 1111	(O×AF)
After instruction:	W=0×1A	; 1011 0101	(0×B5)	After instruction:	REG=0×1A, W=0×B5		CONSTRUCTION (CONSTRUCT)
		; 0001 1010	(0×1A)			; 0001 1010	(0×1A)
Example 2 XORLV	V REG			Example 2 XORW	F REG,O		
Before instruction:	W=N×AF	; 1010 1111	(0×A3)	Before instruction:	REG=0×AF, W=0×B5	; 1010 1111	(O×AF)
		; 0011 0111		(SAE EF 10 (A)	REG=0×AF, W=0×1A		2 - 10 - 10 - 10 - 10 - 10 - 10 - 10 - 1
After instruction:	W=0×18					; 0001 1010	





Syntax:	[/abe/] INCF f, d	Syntax:	[<i>label</i>] DECF f , d
Description:	Increments f register by one.	Description:	Decrements fregister by one.
	If d=0, result is stored in W reg	12/20/20/20/20/20/20/20/20/20/20/20/20/20	If d=0 , result is stored in W reg
	If d=1 , result is stored in f reg	65	If d=1 , result is stored in f reg
Operation:	$(\mathbf{f}) + 1 \Rightarrow \mathbf{d}$	Operation:	$(f) - 1 \Rightarrow d$
Operand:	0 ≤ f ≤ 127	Operand:	$0 \le \mathbf{f} \le 127$
ACCRETERS STORES	$\mathbf{d} \in [0,1]$	10.00 million (2000) 20	$\mathbf{d} \in [0,1]$
Flag:	Z	Flag:	Z
Number of words:	1	Number of words:	1
Number of cycles:	1	Number of cycles:	1
Example 1 INCF	REG, 1	Example 1 DECF	REG, 1
Before instruction:	REG=0×FF	Before instruction:	
	Z=0	NUS 32 20 20	Z=0
After instruction:	REG=0×00	After instruction:	
	Z=1		Z=1
Example 2 INCF	REG, O	Example 2 DECF	REG, O
Before instruction:	PEC-0X10	Before instruction:	REG=0×13
	W=X	s:	$W = \times$
	Z=0	26.53 46 47 63	Z=0
After instruction:	Z=0 REG=0×10	After instruction:	REG=0×13
Arterangeraction	W=0×11		W=0×12
	Z=0		Z=0





[label] RRF f, d [label] RLF f, d Syntax: Syntax: Description: Contents of fregister is rotated by one Description: Contents of **f** register is rotated by one space to the right through C space to the left through C flag. If d=0, result is stored in W register. If d=0, result is stored in W register. If d=1, result is stored in f register. If d=1, result is stored in f register. Operation: $(f < n >) \Rightarrow d < n - 1 >, f < 0 > \Rightarrow C, C \Rightarrow d < 7 >$ $(f < n >) \Rightarrow d < n + 1 >, f < 7 > \Rightarrow C, C \Rightarrow d < 0 >;$ **Operation:** Operand: $0 \leq \mathbf{f} \leq 127$ Operand: $0 \leq \mathbf{f} \leq 127$ $\mathbf{d} \in [0,1]$ $d \in [0,1]$ Flag: С Flag: C registar f C + Number of words: 1 registar f Number of words: 1 Number of cycles: 1 Number of cycles: 1 Example 1 RRF REG, 0 Example 1 RLF REG, 0 Before instruction: REG=1110 0110 Before instruction: REG=1110 0110 W=X C=0C=0 After instruction: REG=1110 0110 After instruction: REG=1110 0110 W=1100 1100 W=0111 0011 C=1 C=0Example 2 RLF REG, 1 Example 2 RRF REG, 1 Before instruction: REG=1110 0110 Before instruction: REG=1110 0110 C=0C=0After instruction: REG=1100 1100 After instruction: REG=0111 0011 C=1 C=0





Syntax:	[/abe/] COMF f, d	Syntax:	[<i>label</i>] BCF f, b
Description:	Contents of f register is complemented	Description:	Reset bit b in f register.
	If d=0 , result is stored in W register.	Operation:	$(0) \Rightarrow f < b >$
	If d=1, result is stored in f register.	Operand:	0 ≤ f ≤ 127
Operation:	$\overline{(\mathbf{f})} \Rightarrow \mathbf{d}$		$0 \le \mathbf{b} \le 7$
Operand:	0 ≤ f ≤ 127	Flaq:	
	$\mathbf{d} \in [0,1]$	Number of words:	
Flag:	Z	Number of cycles:	8388
Number of words:	1	Number of cycles.	- -
Number of cycles:	1	Example 1 BCF F	REG, 7
Example 1 COMF	REG, O	Before instruction:	REG=0×C7; 1100 0111 (0×C7)
Before instruction: After instruction:	REG=0×13 ; 0001 0011 (0×13) REG=0×13 ; complement	After instruction:	REG=0×47; 0100 0111 (0×47)
Arter instruction.	W=0×EC	Example 2 BCF I	NDF, 3
	; 1110 1100 (O×EC)	Before instruction:	₩-0×17
Example 2 COMF	INDE 1	Defote anstruction.	FSR=0×C2
Example 2 COMP	INDE, I		
Before instruction:	ECD-0×C2	After instruction:	address contents (FSR)=0×2F
before instruction.	address contents (FSR)=0×AA	Anteranstructiona	W=0×17
After instruction:	FSR=0×C2		FSR=0×C2
Anterninstruction#	address contents (FSR)=0×55		address contents (FSR)=0×27





Syntax:	[label] BSF f, b	Syntax:	[<i>label</i>] BTFSC f, b	
Description:	Set bit b in f register.	Description:	김부가 물을 못 한다. 한다. 태가 문가 가지 않는 것이 좋아?	er equals zero, then we skip the next instruction.
Operation:	$1 \Rightarrow f < b >$			o, during execution of the current instruction,
Operand:	0 ≤ f ≤ 127			next one is disabled, and NOP instruction executes ing the current one a two-cycle instruction.
Constraints of Statement of Statements	0 ≤ b ≤ 7	Operation:	Skip next instruct	그 아이들 이 것은 이 것은 것은 것은 것은 것을 것 같아. 것 같아요. 편의 것은 것은 것을 것 같아. 것 ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
Flag:		100000000000000000000000000000000000000	0 ≤ f ≤ 127	
Number of words:	1	2012 (CALES ALCONS) (CALES	0 ≤ b ≤ 7	
Number of cycles:	1	Flag:	1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999	
8		Number of v		
Example 1 BSF F	REG, 7	Number of c	ycles: 1 or 2 dep	ending on a b bit
	REG=0×07;00000111 (0×07) REG=0×17;10000111 (0×17)	Example		va avuer and opping
		2000 No. 01 (200)	BTFSC REG,1	;Test bit no.1 in REG
Example 2 BCF I	NDF, 3	20.02275.0.0	a mana a	;Skip this line if =0
		LAB_03	5 113 IA I	;Skip here if =1
Before instruction:	W=0×17 FSR=0×C2	Before instruc	ction, program cou	nter was at address LAB_01.
After instruction:	address contents (FSR)=0×20 W=0×17	After instruct address LAB_	02210101	n REG register was zero, program counter points to
	FSR=0×C2 address contents (FSR)=0×28	If the first bit	in REG register wa	as one, program counter points to address LAB_02.





Syntax:	[<i>label</i>] BTFSS f , b		Syntax:	[/abe/] INCFSZ f, d
Descriptio	n:If bit b in f registe	er equals one, then skip over the next instruction.	Description	1: Contents of f register is incremented by one.
2010/06/02/10/07/07	If bit b equals one	, during execution of the current instruction, the		If d=0, result is stored in W register.
		ed, and NOP instruction is executed instead, thus		If $d=1$, result is stored in f reqister.
		t one a two-cycle instruction.		If result =0, the next instruction is executed as NOP making
Operation	: Skip next instructi	가 가지 않는 것은 것은 것 같은 것이 있는 것은 것은 것은 것은 것은 것은 것을 하는 것을 알려요. 것은 것은 것을 받았는 것을 다 있다. 것은 것은 것은 것을 받았는 것을 알려요. 것은 것은 것을 알려요. 것은 것을 같은 것을 알려요. 것은 것을 같이		the current one a two-cycle instruction.
	0 ≤ f ≤ 127		Operation:	$(f) + 1 \Rightarrow d$
70 . 0.000000000000000000000000000000000	0 ≤ b ≤ 7		Operand:	$0 \le f \le 127$
Flaq:			7. 6 /19/07/17/31	$\mathbf{d} \in [0,1]$
	f words: 1		Flag:	gent total
Number o	f cycles: 1 or 2 dep	ending on a b bit	Number of	words: 1
			Number of	cycles: 1 or 2 depending on a result
Example			1992 1993 1993	· · · · · · · · · · · · · · · · · · ·
BARAR SA			Example	
LAB_01	BTFSS REG,1	;Test bit no.1 in REG	lan an a	
LAB_02		;Skip this line if =1	LAB_01	INCFSZ_REG, 1_; Increase the contents REG by one.
LAB_03		;Skip here if =0	LAB_02	, , , , , , ; Skip this line if =0
1672			LAB_03	; Skip here if =0
Before inst	ruction, program cou	nter was at address LAB_01	1998. 1997 - 28 - 26	
		-	The content:	s of program counter before instruction, PC=address LAB_01
After instru	ction, if the first bit i	n REG register was one, program counter points to	3893 Z8 95	
address LA		2 21 2 1	The content:	s of REG after executing an instruction REG=REG+1, if REG=0,
			program cou	inter points to label address LAB_03. Otherwise, pc
If the first	bit in REG register w	as zero, program counter points to address LAB_02.	NO 0000 00 00	Idress of the next instruction or to LAB_02.





Syntax: Description:	[<i>label</i>] DECFSZ f , d Contents of f register is decr by one.	
busunption	If d=0 , result is stored in W register. If d=1 , result is stored in f register. If result = 0, next instruction is executed as NOP, thus making the current one, a two-cycle instruction.	Syntax:[label] GOTO kDescription:Unconditional jump to address k.Operation: $k \Rightarrow PC < 10:0>$, (PCLATH <4:3>) $\Rightarrow PC < 12:11>$ Operand: $0 \le k \le 2048$
Operation:	$(f) - 1 \Rightarrow d$	Flag: _
Operand:	$0 \le f \le 127$	Number of words: 1
	$\mathbf{d} \in [0,1]$	Number of cycles: 2
Flag:	· •	
Number of wor	이번 사람이 있는 것이 같아요.	Example
Number of cycl	es: 1 or 2 depending on a result	8
Example		LAB_00
· 전의 한 관련 등 이 이 한	Z CNT, 1 ; Decr the contents REG by one ; Skip this line if = 0 ; Skip here if = 1	LAB_01
The contents of ; PC=address L/	program counter before instruction, xB_01	Before instruction: PC=address LAB_00 After instruction: PC=address LAB_01
The contents of CNT=CNT-1, if	CNT register after executing an instruction CNT=0,	
Otherwise, pro	points to address of label LAB_03. gram counter points to	
address of the fo	Ilowing instruction, or to LAB_02.	





Syntax: Description	[<i>label</i>] CALL k : Instruction calls a subprogram. First, return		
o o o o o o o o o o o o o o o o o o o	address (PC+1) is stored on stack, then 11-bit direct operand k , which contains the subprogram	Syntax:	[label] RETURN
	address, is stored in program counter.	Description:	Contents from the top of a stack
Operation:	(PC) + 1 \Rightarrow Top Of Stack (TOS)		is stored in program counter.
	$\mathbf{k} \Rightarrow PC < 10:0>, (PCLATH < 4:3>) \Rightarrow PC < 12:11>$		
	Fig. 32	Operation:	TOS ⇒ program counter PC
Charles Control (Control (Contro) (Contro) (Control (Contro) (Contro) (Contro) (C	0 ≤ k ≤ 2048	Operand:	9898
Flag: Number of	- worder 1	Flaq:	9 5 9
Number of	· · · · · · · · · · · · · · · · · · ·	Number of wor	ids: 1
		Number of cycl	
Example		Maniber of eye	
LAB_01	CALL LAB_02 ; Call subrutine LAB_02 :	Example RE	TURN
		Before instructio	n: PC=X
LAB_02			TOS=X
Before instru	iction: PC=address LAB_01	After instruction:	: PC=TOS
10	TOS=X		TOS=TOS-1
After instruc	tion: PC=address LAB_02 TOS=LAB_01		





Syntax:	[/abe/] RETLW k		
	8-bit constant k is stored in W register. Value off the top of a stack is stored in pc		[<i>label</i>] RETFIE Return from a subprogram. Value
Operation: Operand:	(k) ⇒ W ; TOS ⇒ PC 0 ≤ k ≤ 255	bescription.	from TOS is stored in PC. Interrupts are enabled by setting a GIE bit.
Flag: Number of w	- Inrds: 1		TOS \Rightarrow PC; 1 \Rightarrow GIE
Number of c		Operand: Flag: Number of w	- -
Example	RETLW 0×43	Number of cy	
Before instruc	8/2017936 9/2019	Example	RETFIE
2012 22 21 25	PC=X TOS=X	Before instruc	tion: PC=× GIE=0
After instructi	on: W=0×43 PC=TOS	After instructio	on: PC=TOS
	TOS=TOS-1		





Syntax:	[<i>label</i>] NOP	Syntax: [lab	e/] CLRWDT
Description:	Does not execute any operation or affect any flag.	Description: Wa	tchdoq timer is reset. Prescaler he Watchdog timer is also reset,
Operation:		and	status bits TO and PD are set also
Operand:			
Flag:	8 .	Operation:	$0 \Rightarrow WDT$
Number of w	ords: 1		$0 \Rightarrow$ WDT prescaler
Number of cy	cles: 1		$1 \Rightarrow \overline{\text{TO}} \\ 1 \Rightarrow \overline{\text{PD}}$
Example	NOP	Operand: Flag:	- TO, PD
Before instruct	tion: PC=×	Number of word	
After instructio	on: PC=×+1	Number of cycle	s: 1
		Example CLR	WDT
		Before instruction:	: WDT counter=× WDT prescaler=1:128
		After instruction:	WDT counter=0×00 WDT prescaler counter=0 TO=1 PD=1 WDT prescaler=1: 128





Syntax:	[/abe/] SLEEP
Description:	Processor goes into low consumption mode. Oscis stopped. PD (Power Down) status bit is reset. TO bit is set. WDT (Watchdog) timer and its prescaler are reset.
Operation:	° ⇒ WDT ′′
	$0 \Rightarrow WDT \text{ prescaler}$
	$1 \Rightarrow \overline{\text{TO}}$
	$0 \Rightarrow \overline{PD}$
Operand:	
Flag:	TO, PD
Number of words:	1
Number of cycles:	1
Example SLEEP	
	WDT counter=×
	WDT prescaler=×
After instruction:	WDT counter=0×00
	WDT prescaler=0
	TO=1
	PD=0