Datasheet



8-bit MCU with 1k program ROM,64-byte RAM,1 R2F module and 3 × 13 LCD driver.



Fortune Semiconductor Corporation 富晶電子股份有限公司

23F., No.29-5,Sec. 2, Zhongzheng E. Rd Danshui Town, Taipei County 251, Taiwan Tel. : 886-2-28094742 Fax : 886-2-28094874 www.ic-fortune.com

This manual contains new product information. Fortune Semiconductor Corporation reserves the rights to modify the product specification without further notice. No liability is assumed by Fortune Semiconductor Corporation as a result of the use of this product. No rights under any patent accompany the sale of the product.

Contents

1.	GENE	RAL DESCRIPTION						
2.	FEATI	JRES4						
3.	APPLICATIONS							
4.	ORDERING INFORMATION							
5.	PAD ASSIGNMENT							
6.	PIN D	ESCRIPTION						
7.	FUNC	TIONAL BLOCK DIAGRAM						
8.	TYPIC	AL APPLICATION CIRCUIT						
	8.1	Digital Clinical Thermometer						
9.	ELEC	TRICAL CHARACTERISTICS						
	9.1	Absolute Maximum Ratings						
	9.2	D.C. Characteristics						
	9.3	A.C. Characteristics						
10.	FUNC	TION DESCRIPTION						
	10.1	CPU Core 9 10.1.1 Program Memory Organization 9 10.1.2 Data Memory Organization 9						
	10.2	Low Battery Detection						
	10.3	I/O Port						
	10.4	R2F Conversion Module						
	10.5	LCD Driver						
11.	INSTR	UCTION SET						
	11.1	Instruction Set Summary16						
12.	INSTR	UCTION DESCRIPTION						
13.	REVIS	SION HISTORY						

1. General Description

The FS91M68 is a 8-bit high performance and cost-efficient microcontroller with one R2F module and 3×13 LCD driver. The device is suited for use in low power LCD applications such as: thermometers etc.

2. Features

- 8-bit microcontroller.
- Embedded 1k-word ROM and 64-byte RAM.
- 1.5V battery operation, with about 40µA (Typ.) operation current, and 0.2µA (Typ.) sleep mode current.
- One R2F (Resistance to Frequency) conversion module for sensor and reference resistors.
- One high-speed comparator and one 16-bit counter with programmable gate time select.
- Build-in voltage doubler for 1/3 duty, 1/2 bias 3×13 LCD driver.
- Input port : 4-bit; In / Out port : 4-bit
- Two buzzer outputs.
- Build in low battery detector (LVD).
- Package : Dice form (36 pins), 44-pin LQFP.

3. Applications

Clinical thermometer.

• R/C Type Sensor Measurement.

4. Ordering Information

Product Number	Package Type
FS91M68-nnnV	Dice form of 36 pin
FS91M68-nnnV-PCD	44 pin QFP

Note1: Code number "nnnV" is assigned for customer; "nnn" = 001~999; "V" means Version = A~Z.

Rev. 1.3

5. Pad Assignment



Figure 5-1 : FS91M68 pad assignment

6. Pin Description

Name	In/Out	Pad NO.	Description
RF		1	Reference resistor connection
RS		2	Sensor resistor connection
VDD		3	Positive input of power supply (1.5V)
RST	I	4	CPU Reset Pin
VSS		5	Negative input of power supply
TST		6	Test pin for IC
RP	I/O	7	System oscillator external resistor connection (450k)
RN	I/O	8	System oscillator external resistor connection (450k)
COM1~3	0	9~11	LCD common driver
SEG1~13	0	12~24	LCD segment driver
C512	I/O	25	Voltage doubler capacitor negative terminal
CAP	I/O	26	Voltage doubler capacitor positive terminal
VEE	I/O	27	Voltage doubler output (+3.0V)
PT1[7]	I/O	28	I/O port shared with the positive buzzer output
PT1[6]	I/O	29	I/O port shared with the negative buzzer output
PT1[5]	I/O	30	I/O port
PT1[4]	I/O	31	I/O port
PT1[3]		32	Input port
PT1[2]		33	Input port
PT1[1]	I	34	Input port
PT1[0]		35	Input port
SC		36	Comparator input



7. Functional Block Diagram

8. Typical Application Circuit

Digital Clinical Thermometer





9. Electrical Characteristics

Absolute Maximum Ratings

Table 9-1 Absolute Maximum Ratings

Parameter	Rating	Unit
Supply Voltage to Ground Potential	-0.3 to 1.65	V
Applied Input/Output Voltage	-0.3 to VDD+0.15	V
Ambient Operating Temperature	0 to +70	°C
Storage Temperature	-55 to +150	°C
Soldering Temperature, Time	260°C, 10 Sec	

D.C. Characteristics

Table 9-2 D.C. Characteristics (VDD=1.5V, Ta=25℃)

Symbol	Parameter	Test Condition	Min	Тур	Max	Units
VDD	Recommended Operation Voltage		1.2	1.5	1.65	V
IDD	Supply Current	CPU, R2F On with Fsys=32KHz		40	60	μA
ISTB	Standby Current	CPU sleep, R2F and LCD off	0.1	0.2	1.0	μA

A.C. Characteristics

Table 9-3 A.C. Characteristics (VDD=1.5V, Ta=25°C)

Symbol	Parameter	Test Condition	Min	Тур	Max	Units
Fsys	System Clock	Rsys=450k, VDD=1.5V	25.6	32	38.4	KHz

Rev. 1.3

10. Function Description

CPU Core

10..1 Program Memory Organization

CPU has a 10-bit program counter capable of addressing 1K word program memory space. The reset vector is at 0000h and the interrupt vector is at 0004h.



Figure 10-1 : Program memory map

Address		Content								
Address	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000H	IND0		Use contents of FSR0 to address data memory							
001H	IND1			Use content	s of FSR1 t	o address o	lata memor	у		
002H	FSR0	-		Indirect of	data memor	ry, address	point 00H;	7-bit only		
003H	FSR1			Indirect of	data memor	y, address	point 00H;	7-bit only		
004H	STATUS	-	-		-) -	-	С	Z	
005H	WORK	WORK register								
006H	INTF	-	-	$\overline{\mathbf{\cdot}}$	-	-	-	E0IF	TMIF	
007H	INTE	GIE		-	-	-	-	EOIE	TMIE	
008H	PT1	4			PT1[7:0]				
009H	PT1EN		PT1EN	N[7:4]		-	-	-	-	
00AH	PT1PU				PT1PL	J[7:0]				
00EH	PT1MR	BPE2	BPE1	CH_S	RF_EN	-	-	-	-	
010H	LCD0		LCDEN	:	SEG2[2 : 0]			SEG1[2:0]		
011H	LCD1	-	-	- SEG4[2 : 0]				SEG3[2:0]		
012H	LCD2	-	-		SEG6[2:0]			SEG5[2:0]		
013H	LCD3	-	- SEG8[2:0] SEG7[2:0]							

Address	Content									
Address	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
014H	LCD4	-	-	S	SEG10[2:()]	:	SEG9[2:0]		
015H	LCD5	-	-	SEG12[2:0]			S	SEG11[2 : 0]		
016H	LCD6	-	-	-	-	-	SEG13[2:0])]	
018H	CKCON	LCD_S	PMPEN	PCK_S	BP_S	TMRST	GT_S[2:0]			
019H	TMCNTH	TMCNT[15 : 8]								
01AH	TMCNTL				TMCN	T[7:0]				
01BH	RSCNTH				RSCNT	[15 : 8]				
01CH	RSCNTL	RSCNT[7 : 0]								
01DH	LowBatDct	IbEN	lowPwr	-	-	·	- BiasSEL1 BiasSEL0			
40H~7FH		General Data Memory								

• IND0: Indirect addressing mode address 0.

• IND1: Indirect addressing mode address 1.

• FSR0: Indirect addressing mode point 0.

• FSR1: Indirect addressing mode point 1.

• C: Carry flag.

• Z: Zero flag.

• E0IF, E0IE: PT1.0 external interrupt flag and enable.

• TMIF, TMIE: 8-bit Timer interrupt flag and enable.

• GIE: Global interrupt enable.



Low Battery Detection

Address	Nome				Con	itent			
Address	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
01DH	LowBatDct	IbEN	lowPwr	-	-	-	-	BiasSEL1	BiasSEL0

When resetting, IbEN=1, [BiasSEL1,BiasSEL0]=[0,0].

• "IbEN=1" enables low batery detection, "IbEN=0" disables low batery detection.

• When reading LowBatDct, lowPwr=1 is normal, lowPwr=0 indicates it is under preset low voltage.

Low Battery Detection option table :

BiasSEL1	BiasSEL0	Detect Voltage
0	0	1.329V
0	1	1.293V
1	0	1.260V
1	1	1.224V

***For various Low Voltage Detection option, the voltage detected might be different due to the slight variation of the IC process.

I/O Port

Address	Name	Content							
		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
008H	PT1	PT1[7:0]							
009H	PT1EN		PT1EN	N[7:4]			-	-	-
00AH	PT1PU				PT1PL	[0 : 7]J			
00EH	PT1MR	BPE2	BPE1	CH_S	RF_EN	-	-	-	-

• PT1[3 : 0] are input ports, with pull-up resistor enable control.

• PT1[7:4] are I/O ports, when PT1EN[7:4] =0, PT1[7:4] will be the input ports.

when PT1EN[7:4] = 1, PT1[7:4] will be the output ports

• When system reset or initial start-up, the default value of PT1EN[*] is 0.

PI1EN[*] PI1[*] setting	
0 Input port	/
1 Output port	

• When PT1PU[N]=0, PT1[N] has no pull-up resister ; When PT1PU[N]=1, PT1[N] has pull-up resister

PT1PU[*]	PT1[*] setting
0	Internal pull-up disable
1	Internal pull-up enable



Figure 10-2 : I/O ports

- PT1[7:0] have Schmitt-trigger inputs.
- When PT1[0] is set as an interrupt input, negative edge interrupt has absolute high priority, independent of GIE's control.
- BPE2 =1 & PT1EN[7]=1: PT1[7] is used as the positive input for a buzzer. BPE1=1 & PT1EN[6]=1: PT1[6] is used as negative input of the buzzer. When system reset or initial start-up, the default value of BPE[*] is 0.

BPE1	PT1EN[6]	PT1[6] setting
1	1	PT1[6] is used as the negative input of the buzzer
BPE2	PT1EN[7]	PT1[7] setting
1	1	PT1[7] is used as the positive input of the buzzer

• RF_EN enables R/F (Resistor to Frequency) switch module.

• CH_S=0 selects reference resistor (RF part) oscillator, CH_S=1 selects sensor resistor (RS part) oscillator.

Address	Norma				Con	tent			
Address	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
00EH	PT1MR	BPE2	BPE1	CH_S	RF_EN	-	-	-	-
018H	CKCON	LCD_S	PMPEN	PCK_S	BP_S	TMRST		GT_S[2:0]]
019H	TMCNTH				TMCNT	[15:8]			
01AH	TMCNTL				TMCN	T[7:0]			
01BH	RSCNTH				RSCNT	[15:8]			
01CH	RSCNTL				RSCN	T[7:0]			
01DH	LowBatDct	IbEN	lowPwr	-	-	-		BiasSEL1	BiasSEL0

R2F Conversion Module

• RF_EN enables R/F (Resistor to Frequency) switch module.

• CH_S=0 selects reference resistor (RF part) oscillator, CH_S=1 selects sensor resistor (RS part) oscillator.

• LCD_S : LCD clock setup.

• PMPEN : Clock enable in step-up circuit.

• PCK_S : Clock setup in step-up circuit.

• BP_S : Buzzer clock setup.

• TMCNT is the Timer for Gate time; RSCNT is The Counter for RF module.

• GT_S is for Gate time setting of RF module, please refer to the table below.

• TMRST : "0" clears all counter and stops counting;" 0→1 " starts to count until Gate time interrupt occurs.

When TMCNT re-counts, TMRST must be set as 0 first, so TMCNT and RSCNT will be cleared as 0. When TMRST is set as 1, TMCNT and RSCNT start counting until TMCNT[N]1 \rightarrow 0. Then TMCNT and RSCNT stop counting and store the value at the same time.



Figure 10-4 R2F Conversion Module

• R2F module Gate time setting :

GT_S[2:0]	TMCNT[N]	Gate time(Hz)
000	8	64
001	9	32
010	10	16
011	11	8
100	12	4
101	13	2
110	14	1
111	15	0.5

LCD Driver

Addroso	Nama				Con	tent			
Address	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
010H	LCD0	-	LCDEN		SEG2[2:0]		:	SEG1[2:0]	
011H	LCD1	-	-		SEG4[2:0]		:	SEG3[2:0]	
012H	LCD2	-	-		SEG6[2:0]		:	SEG5[2:0]	
013H	LCD3	-	-		SEG8[2:0]			SEG7[2:0]	
014H	LCD4	-	-	e.,	SEG10[2:0]	:	SEG9[2:0]	
015H	LCD5	-	-	5	SEG12[2:0]	S	SEG11[2:0)]
016H	LCD6	-		-	-	-	S	SEG13[2 : 0	0

• LCDEN option table :

LCDEN	LCD display
0	Disable
1	Enable

11. Instruction Set

The FS91M68 instruction set consists of 36 instructions. Each instruction is a 16-bit word with an OPCODE and one or more operands. The detail descriptions are below.

Note: FS91M68 does not have HALT instruction to avoid the system error occurrence when fast releasing and plugging-in the battery repeatedly.

Instruction Set Summary

Table 11-1: FS91M68 Instruction Set

Instruction	Operation	Cycle=Fsys/4	Flag
ADDLW k	$[W] \leftarrow [W] + k$	1	C, DC, Z
ADDPCW	[PC] ← [PC] + 1 + [W]	2	None
ADDWF f, d	$[Destination] \leftarrow [f] + [W]$	1	C, DC, Z
ADDWFC f, d	$[Destination] \leftarrow [f] + [W] + C$	1	C, DC, Z
ANDLW k	$[W] \leftarrow [W] AND k$	1	Z
ANDWF f, d	[Destination] \leftarrow [W] AND [f]	1	Z
BCF f, b	[f] ← 0	1	None
BSF f, b	[f] ← 1	1	None
BTFSC f, b	Skip if [f] = 0	1, 2	None
BTFSS f, b	Skip if [f] = 1	1, 2	None
CALL k	Push PC + 1 and GOTO k	2	None
CLRF f	[f] ← 0	1	Z
CLRWDT	Clear watch dog timer	1	None
COMF f, d	$[f] \leftarrow NOT([f])$	1	Z
DECF f, d	[Destination] \leftarrow [f] -1	1	Z
DECFSZ f, d	[Destination] \leftarrow [f] -1, skip if the result is zero	1, 2	None
GOTO k	PC ← k	2	None
INCF f, d	[Destination] ← [f] +1	1	Z
INCFSZ f, d	[Destination] \leftarrow [f] + 1, skip if the result is zero	1, 2	None
IORLW k	$[W] \leftarrow [W] \mid k$	1	Z
IORWF f, d	$[Destination] \leftarrow [W] [f]$	1	Z
MOVFW f	$[W] \leftarrow [f]$	1	None
MOVLW k	$[W] \leftarrow k$	1	None
MOVWF f	$[f] \leftarrow [W]$	1	None
NOP	No operation	1	None
RETFIE	Pop PC and GIE = 1	2	None
RETLW k	RETURN and W = k	2	None
RETURN	Pop PC	2	None
RLF f, d	$[Destination < n+1>] \leftarrow [f < n>]$	1	C,Z
RRF f, d	$[Destination < n-1>] \leftarrow [f < n>]$	1	C, Z
SLEEP	Stop OSC	1	PD
SUBLW k	$[W] \leftarrow k - [W]$	1	C, DC, Z
SUBWF f, d	$[Destination] \leftarrow [f] - [W]$	1	C, DC, Z
SUBWFC f, d	[Destination] \leftarrow [f] – [W] –C	1	C, DC, Z
XORLW k	$[W] \leftarrow [W] XOR k$	1	Z
XORWF f, d	[Destination] ← [W] XOR [f]	1	Z

Note :

- f : memory address (00h ~ 7Fh).
- w : work register.
- k : literal field, constant data or label.
- d : destination select: d=0 store result in W,
- d=1: store result in memory address f. • b : bit select $(0 \sim 7)$.
- [f] : the content of memory address f.

- PC : program counter.
- C : Carry flag
- DC : Digit carry flag
- Z : Zero flag
- TO : watchdog time out flag
- WDT : watchdog timer counter

12. Instruction Description

(By alphabetically)

ADDLW	Add Literal to W	ADDWF	Add W to f	
Syntax:	ADDLW k	Syntax:	ADDWF f, d	
Operation:	$0 \le k \le FFh$		$0 \le t \le F \ge h$	
Elan Affected	$[VV] \leftarrow [VV] + K$	Operation [.]	$u \in [0, 1]$ [Destination] \leftarrow [f] + [W]	
Description:	The content of Work register add	Flag Affected:	C. CD. Z	
2000.1010	literal "k" in Work register	Description:	Add the content of the W register	
Cycle:	1		and [f]. If d is 0, the result is	
Example:	Before instruction:		stored in the W register. If d is 1,	
ADDLW 08h	VV = 08h	Cycle:	the result is stored back in f.	
	W = 10h	Example 1	Before instruction	
	vv ron	ADDWF OPERAND. 0	OPERAND = C2h	
			W = 17h	
			After instruction:	
			OPERAND = C2h	
		Example 2	W = D9n Before instruction:	
		ADDWF OPERAND, 1	OPERAND = C2h	
			W = 17h	
			After instruction:	
			OPERAND = D9h	
			VV = 1/h	
ADDPCW	Add W to PC	ADDWFC	Add W, f and Carry	
Syntax:	ADDPCW	Syntax:	ADDWFC f, d	
Operation:	[PC] ← [PC] + 1 + [W], [W] < 79h		$0 \le f \le FFh$	
	$[PC] \leftarrow [PC] + 1 + ([W] - 100h),$		d ∈ [0,1]	
	otherwise	Operation:	$[\text{Destination}] \leftarrow [f] + [W] + C$	
Plag Affected:	None The relative address PC + 1 + W	Plag Allected.	C, DC, Z Add the content of the W	
Description.	are loaded into PC.	Description	register. [f] and Carry bit.	
Cycle:	2		If d is 0, the result is stored in the	
Example 1:	Before instruction:		W register.	
ADDPCW	W = 7Fh, PC = 0212h		If d is 1, the result is stored back	
	After instruction:	Cycle:	in t. 1	
Example 2 [.]	Before instruction:	Example:	I Before instruction:	
ADDPCW	W = 80h, PC = 0212h	ADDWFC OPERAND,1	C = 1	
	After instruction:		OPERAND = 02h	
	PC = 0193h		W = 4Dh	
Example 3:	Before instruction:		After instruction:	
ADDPCW	VV = FER, PC = 0212R		OPERAND = 50h	
	PC = 0211h		W = 4Dh	

ANDLW	AND literal with W	BSF	Bit Set f	
Syntax:	ANDLW k	Syntax:	BSF f, b	
Operation:	0 ≤ k ≤ FFn [W] ← [W] AND k		$0 \le t \le FFn$ $0 \le b \le 7$	
Flag Affected:	Z	Operation:	[f] ← 1	
Description:	AND the content of the W register with the eight-bit literal "k". The result is stored in the W register.	Flag Affected: Description: Cycle: Example: BSF FLAG, 2	None Bit b in [f] is set to 1. 1 Before instruction: FLAG = 89h	
Cycle:	1		After instruction:	
Example: ANDLW 5Fh	Before instruction: W = A3h		FLAG = 8Dh	
	W = 03h			

ANDWF	AND W and f	BTFSC	Bit Test skip if Clear
Syntax:	ANDWF f, d	Syntax:	BTFSC f, b
-	$0 \le f \le FFh$		$0 \le f \le FFh$
	d ∈ [0,1]		$0 \le b \le 7$
Operation:	[Destination] \leftarrow [W] AND [f]	Operation:	Skip if $[f < b >] = 0$
Flag Affected:	Ž	Flag Affected:	None
Description:	AND the content of the W	Description:	If bit 'b' in [f] is 0, the next fetched
	register with [f].		instruction is discarded and a
	If d is 0, the result is stored in the		NOP is executed instead making
	W register.		it a two-cycle instruction.
	If d is 1, the result is stored back	Cycle:	1, 2
	in f.	Example:	Before instruction:
Cycle:	1	Node BTFSC FLAG, 2	PC = address (Node)
Example 1:	Before instruction:	OP1 :	After instruction:
ANDWF OPERAND,0	W = 0Fh, OPERAND = 88h	OP2 :	If $FLAG<2> = 0$
	After instruction:		PC = address(OP2)
	W = 08h, OPERAND = 88h		If FLAG<2> = 1
Example 2:	Before instruction:		PC = address(OP1)
ANDWF OPERAND,1	W = 0Fh, OPERAND = 88h		
	After instruction:		
	W = 88h, OPERAND = 08h		
	< / ` C.		
5.05			

BCF	Bit Clear f	BTFSS	Bit Test skip if Set
Syntax:	$\begin{array}{ll} BCF & f, b \\ 0 \leq f \leq FFh \\ 0 \leq b \leq 7 \end{array}$	Syntax:	BTFSS f, b $0 \le f \le FFh$ $0 \le b \le 7$
Operation: Flag Affected: Description: Cycle: Example: BCF FLAG. 2	[f] ← 0 None Bit b in [f] is reset to 0. 1 Before instruction: FLAG = 8Dh	Operation: Flag Affected: Description:	Skip if [f] = 1 None If bit 'b' in [f] is 1, the next fetched instruction is discarded and a NOP is executed instead making it a two-cycle instruction.
	After instruction: FLAG = 89h	Cycle: Example: Node BTFSS FLAG, 2 OP1 : OP2 :	1, 2 Before instruction: PC = address (Node) After instruction: If FLAG<2> = 0 PC = address(OP1) If FLAG<2> = 1 PC = address(OP2)

CALL		Subroutine CALL	COMF	Complement f
Syntax:		CALL k	Syntax:	COMF f, d
		$0 \le k \le 1FFFh$		$0 \le f \le 255$
Operation:		Push Stack	Operation	$d \in [0,1]$
		$[10p \text{ Stack}] \leftarrow PC + 1$ PC $\leftarrow k$	Operation: Flag Affected	[ĭ] ← NU I ([ĭ]) 7
Flag Affected		None	Description	If f is complemented if f is 0 the
Description:		Subroutine Call. First, return	2000.000	result is stored in the W register.
		address PC + 1 is pushed onto		If d is 1, the result is stored back
		the stack. The immediate		in [f]
Cycle:			Cycle: Example 1:	1 Before instruction:
Example:		Before instruction:	COMF OPERAND.0	W = 88h. OPERAND = 23h
HERE CALL	THERE	PC = address (HERE)		After instruction:
		After instruction:		W = DCh, OPERAND = 23h
		PC = address (THERE)	Example 2:	Before instruction:
		10S = address (HERE + 1)	COMF OPERAND,1	W = 880, $OPERAND = 230$
				W = 88h. OPERAND = DCh
			DEAL	
CLRF			DECF	Decrement f
Syntax:		ULKF I 0 < f < 255	Syntax:	DECET, a 0 < f < 255
Operation.		$0 \le 1 \le 200$ If $l \leftarrow 0$		d ∈ [0 1]
Flag Affected:		None	Operation:	[Destination] \leftarrow [f] -1
Description:		Reset the content of memory	Flag Affected:	ż
.		address f	Description:	[f] is decremented. If d is 0, the
Cycle:		1 Refere instruction:		result is stored in the W register.
		WORK = 5Ah		in ffl
		After instruction:	Cvcle:	1
		WORK = 00h	Example 1:	Before instruction:
			DECF OPERAND,0	W = 88h, OPERAND = 23h
				After instruction:
			Example 2:	W = 22n, OPERAND = 23n Before instruction:
			DECF OPERAND,1	W = 88h, OPERAND = 23h
				After instruction:
				W = 88h, OPERAND = 22h
CLRWDT		Clear watch dog timer	DECFSZ	Decrement f, skip if zero
Syntax:		CLRWDT	Syntax:	DECFSZ f, d
Operation:		Watch dog timer counter will be		$0 \leq f \leq FFh$
Flag Affected:		reset		d ∈ [0,1]
Description		CLRWDT instruction will reset	Operation:	[Destination] \leftarrow [f] -1, skip if the
		watch dog timer counter.	Flag Affected	None
Cycle:		1	Description:	[f] is decremented. If d is 0, the
Example:		After instruction:		result is stored in the W register.
CLRWDI		VVDT = U		If d is 1, the result is stored back
				In [I].
				fetched instruction is discarded
				and a NOP is executed instead
				making it a two-cycle instruction.
			Cycle:	1, 2
			Example:	Before instruction:
			NODE DECESZ FLAG,	After instruction:
4			OP1 :	[FLAG] = [FLAG] - 1
			OP2 :	If [FLAG] = 0
				PC = address(OP1)
				If $[FLAG] \neq 0$
				PC = address(OP2)

GOTO	Unconditional Branch	IORLW	Inclusive OR literal with W
Syntax:	GOTOk	Syntax:	IORLW k
,	$0 \le k \le 1FFFh$,	$0 \le k \le FFh$
Operation:	PC ← k	Operation:	$[W] \leftarrow [W] \mid k$
Flag Affected:	None	Flag Affected:	Z
Description:	The immediate address is	Description:	Inclusive OR the content of the
	loaded into PC.		W register and the eight-bit
Cycle:	2		literal "k". The result is stored in
Example:	After instruction:		the W register.
GOTO THERE	PC = address (THERE)	Cycle:	1 Defere instruction
			Before Instruction:
		IORLW 65H	After instruction:
			W = FDh
INCE	Increment f		Inclusive OP W with f
Suptax:	INCE f d	Syntax:	
Syntax.		Syntax.	
Operation:	$u \in [0, 1]$	Operation:	$u \in [0, 1]$
Elag Affected:	$[\text{Destination}] \leftarrow [f] + 1$	Elog Afforted:	$[Destination] \leftarrow [vv] [I]$
Description:	L If is incremented. If d is 0, the	Play Allected.	Inclusive OR the content of the
Description.	result is stored in the W register	Description.	W register and [f] If d is 0 the
	If d is 1 the result is stored back		result is stored in the W register
	in [f].		If d is 1, the result is stored back
Cvcle:	1		in [f].
Example 1:	Before instruction:	Cycle:	1
INCF OPERAND,0	W = 88h, OPERAND = 23h	Example:	Before instruction:
	After instruction:	IORWF OPERAND,1	W = 88h, OPERAND = 23h
	W = 24h, OPERAND = 23h		After instruction:
Example 2:	Before instruction:		W = 88h, OPERAND = ABh
INCF OPERAND,1	W = 88h, OPERAND = 23h		
	After instruction:		
	W = 88h, OPERAND = 24h		*
INCFSZ	Increment f, skip if zero		
Syntax:	INCFSZ f. d		
	0 < f < FFh		
	d ∈ [0,1]		
Operation:	[Destination] \leftarrow [f] + 1, skip if the		
	result is zero		
Flag Affected:	None		
Description:	[f] is incremented. If d is 0, the		
	result is stored in the W register.		
	If d is 1, the result is stored back		
	in [f].		
	If the result is 0, then the next		
	fetched instruction is discarded		
	and a NOP is executed instead		
	making it a two-cycle instruction.		
Cycle:	1, 2		
Example:	Before instruction:		
Node INCFSZ FLAG, 1	PC = address (Node)		
OP1 :	After instruction:		
0P2 :	[FLAG] = [FLAG] + 1		
	IT $[FLAG] = 0$		
	PC = address(OP2)		
	IT [FLAG] $\neq 0$		
	PC = address(OP1)		

MOVFW	Move f to W	RETFIE	Return from Interrupt
Syntax:	MOVFW f	Syntax:	RETFIE
	$0 \le f \le FFh$	Operation:	[Top Stack] => PC
Operation:	[W] ← [f]		Pop Stack
Flag Affected:	None		1 => GIE
Description:	Move data from [f] to the W	Flag Affected:	None
	register.	Description:	The program counter is loaded
Cycle:	1		from the top stack, then pop
Example:	Before instruction:		stack. Setting the GIE bit
MOVFW OPERAND	W = 88h, OPERAND = 23h		enables interrupts.
	After instruction:	Cycle:	2
	W = 23h, OPERAND = 23h	Example:	After instruction:
		RETFIE	PC = [Top Stack]
			GIE = 1
MOVLW	Move literal to W	RETLW	Return and move literal to W
Syntax:	MOVLW k	Syntax:	RETLW k
	$0 \le k \le FFh$		$0 \le k \le FFh$
Operation:	$[W] \leftarrow k$	Operation:	[W] ← k
Flag Affected:	None		[Top Stack] => PC
Description:	Move the eight-bit literal "k" to		Pop Stack
	the content of the W register.	Flag Affected:	None
Cycle:	1	Description:	Move the eight-bit literal "k" to
Example:	Before instruction:		the content of the W register.
MOVLW 23H	W = 88h		The program counter is loaded
	After instruction:		from the top stack, then pop
	W = 23h		stack.
		Cycle:	2
		Example:	Before instruction:
MOVWF	Move W to f	CALL TABLE	WREG = 0x07
Syntax:	MOVWF f	:	After instruction:
	$0 \le f \le FFh$	TABLE	WREG = value of k7
Operation:	$[f] \leftarrow [W]$	ADDWF PC	
Flag Affected:	None	RETLW KO	
Description:	Move data from the W register	RETLW k1	
	to [f].	· · · · · · · · · · · · · · · · · · ·	
Cycle:	1	RETLW kn	
	Before instruction:		
Example:	M = 99h ODEDAND = 22h		
Example: MOVWF OPERAND	VV = 0011, UPERAIND = 2011		
Example: MOVWF OPERAND	After instruction:		
Example: MOVWF OPERAND	After instruction: W = 88h, OPERAND = 88h		
Example: MOVWF OPERAND	After instruction: W = 88h, OPERAND = 88h	0	
Example: MOVWF OPERAND	After instruction: W = 88h, OPERAND = 88h		

NOP	No Operation	Return	Return from Subroutine
Syntax:	NOP	Syntax:	RETURN
Operation:	No Operation	Operation:	[Top Stack] => PC
Flag Affected	None		Pop Stack
Description:	No operation NOP is used for	Flag Affected:	None
2000.10.00	one instruction cycle delay.	Description:	The program counter is loaded
Cycle:	1		from the top stack, then pop
		Quala	Stack.
		Cycle.	Z After instruction:
		Example.	PC = [Top Stack]
		Return	

RLF	Rotate left [f] through Carry	SUBLW	Subtract W from literal
Syntax:	RLF f, d	Syntax:	SUBLW k
	$0 \le f \le FFh$		$0 \le k \le FFh$
Onenstien	d ∈ [0,1]	Operation:	$[W] \leftarrow k - [W]$
Operation:	[Destination< $n+1>$] \leftarrow [f< $n>$]	Flag Affected:	C, DC, Z
	$[\text{Destination} < 0 >] \leftarrow C$	Description:	Subtract the content of the W
Flag Affected:	$C \leftarrow [1 < 7 >]$		register from the eight-bit literal
Description:	If is rotated one bit to the left		R . The result is stored in the w
	through the Carry bit If d is 0	Cycle:	1
	the result is stored in the W	Example 1:	Before instruction:
C Baristar f	register. If d is 1, the result is	SUBLW 02H	W = 01h
C Register I	stored back in [f].		After instruction:
Cvcle:	1		W = 01h
Example:	Before instruction:		C = 1
RLF OPERAND, 1	C = 0	-	Z = 0
	W = 88h, OPERAND = E6h	Example 2:	Before instruction:
	After instruction:	SUBLW 02H	VV = 02h
	C = 1		W = 00h
	W = 88h, OPERAND =		C = 1
	CCh		Z = 1
		Example 3:	Before instruction:
RRF	Rotate right [f] through Carry	SUBLW 02H	W = 03h
Syntax	RRF f d		After instruction:
Oyntax.	0 < f < FFh		W = FFh
	$d \in [0, 1]$		C = 0
Operation:	[Destination $<$ n-1>] \leftarrow [f $<$ n>]		$\Sigma = 0$
.,	[Destination $<7>1 \leftarrow C$		
	C ← [f<7>]	SUBWF	Subtract W from f
Flag Affected:	C ← [f<7>] C	SUBWF Syntax:	Subtract W from f
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C [f] is rotated one bit to the right	SUBWF Syntax:	Subtract W from fSUBWFf, d $0 \le f \le FFh$
Flag Affected: Description:	$C \leftarrow [f<7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0,	SUBWF Syntax:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$
Flag Affected: Description:	$C \leftarrow [f<7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is	SUBWF Syntax: Operation:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ d $\in [0,1]$ [Destination] $\leftarrow [f] - [W]$
Flag Affected: Description:	$C \leftarrow [f < 7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]	SUBWF Syntax: Operation: Flag Affected:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z
Flag Affected: Description:	$C \leftarrow [f < 7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f].	SUBWF Syntax: Operation: Flag Affected: Description:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] \leftarrow [f] – [W]C, DC, ZSubtract the content of the W
Flag Affected: Description:	$C \leftarrow [f < 7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1	SUBWF Syntax: Operation: Flag Affected: Description:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] \leftarrow [f] – [W]C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, the
Flag Affected: Description: C c c c c c c c c c c c c c c c c c c c	$C \leftarrow [f < 7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction:	SUBWF Syntax: Operation: Flag Affected: Description:	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register.
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0	$C \leftarrow [f < 7>]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95b	SUBWF Syntax: Operation: Flag Affected: Description:	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0	$C \leftarrow [f < 7 >]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction:	SUBWF Syntax: Operation: Flag Affected: Description:	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C (f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1:	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction:
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ $C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C (f) is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction:
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1	Subtract W from fSUBWFf, d $0 \le f \le FFh$ d $d \in [0,1]$ [Destination] \leftarrow [f] – [W]C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32h
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. I Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1
Flag Affected: Description: $f(c) \in Register f$ Cycle: Example: RRF OPERAND, 0 SLEEP	$C \leftarrow [f < 7 >]$ C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h Oscillator stop	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Defension instruction:
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C (f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP SLEEP OPULATION:	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWE OPERAND 4	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Before instruction: OPERAND = 01h W = 01h
Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator stop PD$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Before instruction: OPERAND = 01h, W = 01h After instruction:
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 Sutter Syntax: Operation: Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C (f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU l oscillator is stopped CPU	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Before instruction: OPERAND = 01h, W = 01h After instruction: OPERAND = 00h
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SUEEP Syntax: Operation: Flag Affected: Description:	C ← [f<7>] C [f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1	Subtract W from fSUBWFf, d $0 \le f \le FFh$ d $d \in [0,1]$ [Destination] \leftarrow [f] – [W]C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32hC = 1Z = 0Before instruction:OPERAND = 01h, W = 01hAfter instruction:OPERAND = 00hC = 1
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description:	$C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32hC = 1Z = 0Before instruction:OPERAND = 01h, W = 01hAfter instruction:OPERAND = 00hC = 1Z = 1
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description: Cycle	$C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources. 1$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1 Example 3:	Subtract W from fSUBWFf, d $0 \le f \le FFh$ d $d \in [0,1]$ [Destination] \leftarrow [f] - [W]C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32hC = 1Z = 0Before instruction:OPERAND = 00hC = 1Z = 1Before instruction:
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description: Cycle: Example: Cycle	$C \leftarrow [f < 7 >]$ C (f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources. 1 After instruction:	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1 Example 3: SUBWF OPERAND, 1	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] \leftarrow [f] – [W]C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32hC = 1Z = 0Before instruction:OPERAND = 00hC = 1Z = 1Before instruction:OPERAND = 00hC = 1Z = 1Before instruction:OPERAND = 04h, W = 05h
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description: Cycle: Example: SLEEP	$C \leftarrow [f < 7 >]$ $C \leftarrow [f < 7 >]$ C $F[f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. I Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources. I After instruction: PD = 1$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1 Example 3: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Before instruction: OPERAND = 01h, W = 01h After instruction: OPERAND = 00h C = 1 Z = 1 Before instruction: OPERAND = 04h, W = 05h After instruction: OPERAND = 04h, W = 05h
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description: Flag Affected: Description: Cycle: Example: SLEEP	$C \leftarrow [f < 7 >]$ C $(f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources. 1 After instruction: PD = 1 TO = 0$	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1 Example 3: SUBWF OPERAND, 1	Subtract W from fSUBWFf, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] $\leftarrow [f] - [W]$ C, DC, ZSubtract the content of the Wregister from [f]. If d is 0, theresult is stored in the W register.If d is 1, the result is stored backin [f],1Before instruction:OPERAND = 33h, W = 01hAfter instruction:OPERAND = 32hC = 1Z = 0Before instruction:OPERAND = 00hC = 1Z = 1Before instruction:OPERAND = 00hC = 1Z = 1Before instruction:OPERAND = 04h, W = 05hAfter instruction:OPERAND = FFhCCOPERAND = FFh
Flag Affected: Description: Cycle: Example: RRF OPERAND, 0 SLEEP Syntax: Operation: Flag Affected: Description: Flag Affected: Description: Cycle: Example: SLEEP	$C \leftarrow [f < 7 >]$ C (f] is rotated one bit to the right through the Carry bit. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]. 1 Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h SLEEP CPU oscillator stop SLEEP CPU oscillator is stopped PD CPU oscillator is stopped. CPU can be waked up by external interrupt sources. 1 After instruction: PD = 1 TO = 0 If WDT causes wake up, TO	SUBWF Syntax: Operation: Flag Affected: Description: Cycle: Example 1: SUBWF OPERAND, 1 Example 2: SUBWF OPERAND, 1 Example 3: SUBWF OPERAND, 1	Subtract W from f SUBWF f, d $0 \le f \le FFh$ $d \in [0,1]$ [Destination] \leftarrow [f] - [W] C, DC, Z Subtract the content of the W register from [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f], 1 Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0 Before instruction: OPERAND = 01h, W = 01h After instruction: OPERAND = 00h C = 1 Z = 1 Before instruction: OPERAND = 04h, W = 05h After instruction: OPERAND = FFh C = 0 Z = 0

• Please make sure all interrupt flags are cleared before running SLEEP; "NOP" command must follow HALT and SLEEP commands.

SUBWFC		Subtract W and Carry from f
Syntax:		SUBWFC f, d
		$0 \le f \le FFh$
		d ∈ [0,1]
Operation:		$[Destination] \leftarrow [f] - [W] - \dot{C}$
Flag Affected	1:	C, DC, Z
Description:		Subtract the content of the W
		register from [f]. If d is 0, the
		result is stored in the W register.
		in G IS 1, the result is stored back
Cycle:		1
Example 1:		Before instruction:
SUBWFC	OPERAND,	OPERAND = 33h, W = 01h
1		C = 1
		After instruction:
		OPERAND = $32n, C = 1, Z = 0$
Example 2:		Before instruction:
SUBWFC	OPERAND.	OPERAND = 02h, W = 01h
1	,	C = 0
		After instruction:
		OPERAND = 00h, C = 1, Z =
E		1 Defensionalization
Example 3:		Before instruction: OPEPAND = 0.4b, $W = 0.5b$
1	OFERAND,	C = 0
•		After instruction:
		OPERAND = FEh, C = 0, Z = 0

XORLW		Exclusive OR literal with W
Syntax:		XORLW k
		$0 \le k \le FFh$
Operation:		$[W] \leftarrow [W] XOR k$
Flag Affecte	d:	Z
Description:		Exclusive OR the content of the
		W register and the eight-bit
		literal "k". The result is stored in
		the W register.
Cycle:		1
Example:		Before instruction:
XORLW	5Fh	W = ACh
		After instruction:

After instruction: W = F3h

XORWF		Exclusive OR W and f
Syntax:		XORWF f, d
		$0 \le f \le FFh$
		d ∈ [0,1]
Operation:		[Destination] \leftarrow [W] XOR [f]
Flag Affected	:	Z
Description:		Exclusive OR the content of the
		W register and [f]. If d is 0, the
		result is stored in the W register.
		in fill
Cycle [.]		11 [1]. 1
Example:		Before instruction
XORWF	OPERAND,	OPERAND = 5Fh, W = ACh
1	· · · ·	After instruction:
		OPERAND = F3h

13. Revision History

Ver.	Date	Page	Description
1.0	2006/8/17	All	Initial release
1.1	2006/11/9	5	Pin description redefine
		8	Table 9-2, Units of IDD and ISTB change to be "µA".
1.2	2006/12/21	7	Figure 8-1, Rst pin must be pull high, connect resistance($10k\Omega$) and capacitor($2200pF$).
1.3	2014/5/22	2	Revised company address