Datasheet

FS91M68

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

Fortune SEMICONDUCTOR CORPORATION

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

Contents

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.

5. Pad Assignment

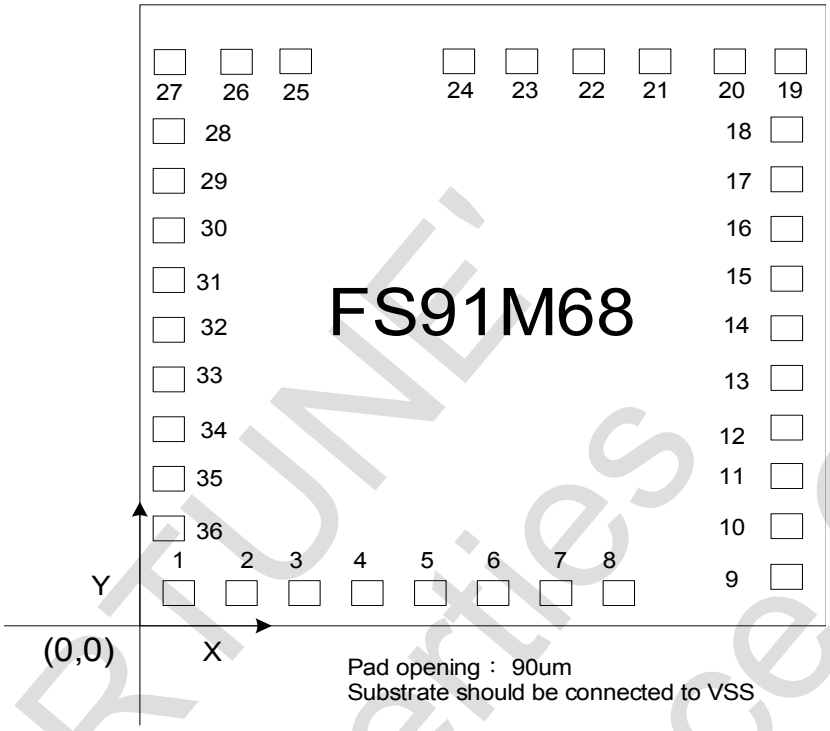


Figure 5-1 ： FS91M68 pad assignment

6. Pin Description

Name	In/Out	Pad NO.	Description
RF	I	1	Reference resistor connection
RS	I	2	Sensor resistor connection
VDD	I	3	Positive input of power supply (1.5V)
RST	I	4	CPU Reset Pin
VSS	I	5	Negative input of power supply
TST	I	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	O	9~11	LCD common driver
SEG1~13	O	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]	I	32	Input port
PT1[2]	I	33	Input port
PT1[1]	I	34	Input port
PT1[0]	I	35	Input port
SC	I	36	Comparator input

7. Functional Block Diagram

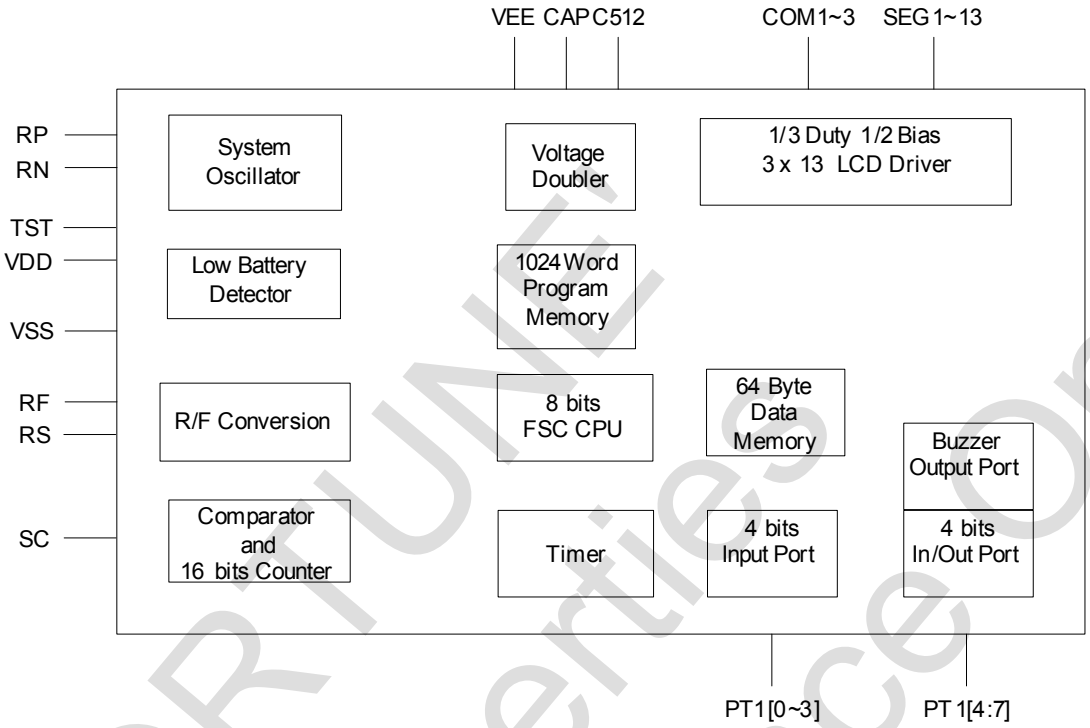


Figure 7-1 ： FS91M68 function block

8. Typical Application Circuit

Digital Clinical Thermometer

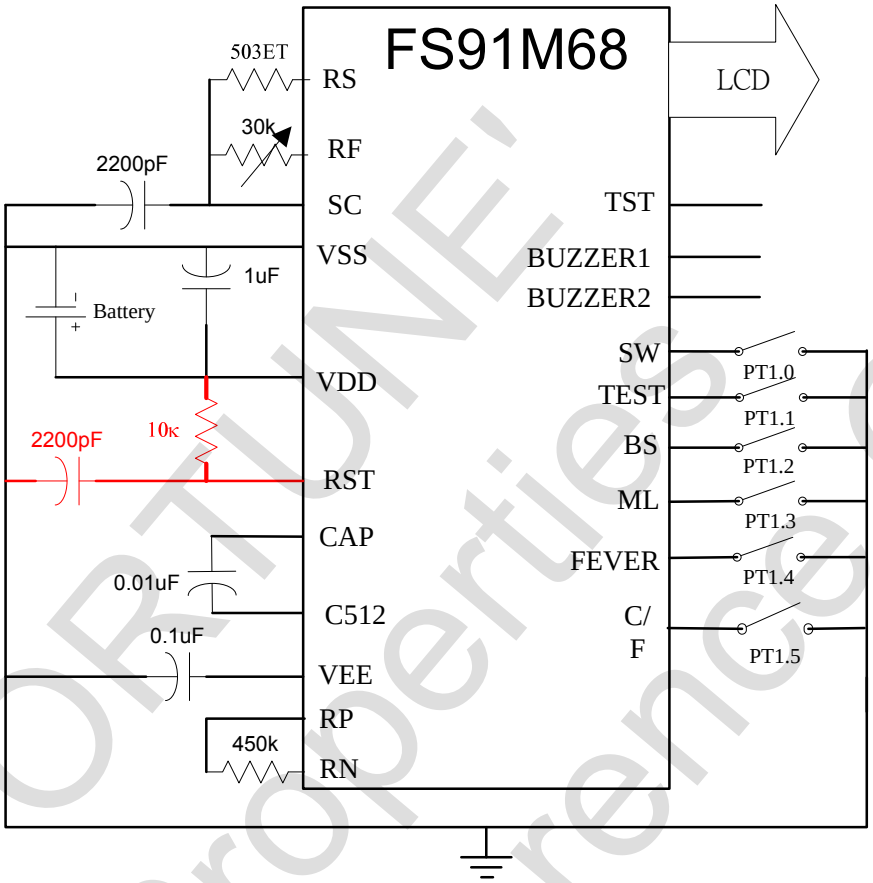


Figure 8-1 ： FS91M68 application circuit

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 (V_{DD}=1.5V, Ta=25℃)

Symbol	Parameter	Test Condition	Min	Typ	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 (V_{DD}=1.5V, Ta=25℃)

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

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.

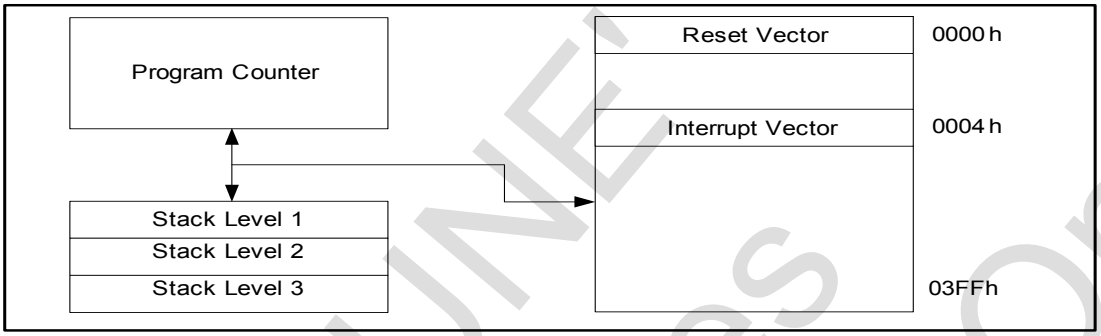


Figure 10-1： Program memory map

10..2 Data Memory Organization

Address	Name	Content							
		bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
000H	IND0	Use contents of FSR0 to address data memory							
001H	IND1	Use contents of FSR1 to address data memory							
002H	FSR0	-	Indirect data memory, address point 00H; 7-bit only						
003H	FSR1	-	Indirect data memory, address point 00H; 7-bit only						
004H	STATUS	-	-	-	-	-	-	C	Z
005H	WORK	WORK register							
006H	INTF	-	-	-	-	-	-	E0IF	TMIF
007H	INTE	GIE	-	-	-	-	-	EOIE	TMIE
008H	PT1	PT1[7：0]							
009H	PT1EN	PT1EN[7：4]				-	-	-	-
00AH	PT1PU	PT1PU[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	Name	Content							
		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
014H	LCD4	-	-	SEG10[2：0]			SEG9[2：0]		
015H	LCD5	-	-	SEG12[2：0]			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	TMCNT[7：0]							
01BH	RSCNTH	RSCNT[15：8]							
01CH	RSCNTL	RSCNT[7：0]							
01DH	LowBatDct	lbEN	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	Name	Content							
		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
01DH	LowBatDct	lbEN	lowPwr	-	-	-	-	BiasSEL1	BiasSEL0

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

- "lbEN=1" enables low batery detection, "lbEN=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[7：4]				-	-	-	-
00AH	PT1PU	PT1PU[7：0]							
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.

PT1EN[*]	PT1[*] 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

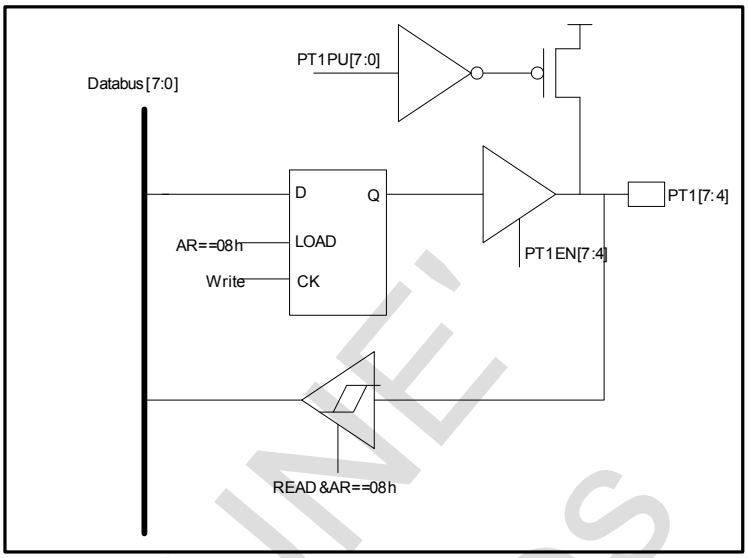


Figure 10-2： I/O ports

- 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.

R2F Conversion Module

Address	Name	Content							
		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	TMCNT[7：0]							
01BH	RSCNTH	RSCNT[15：8]							
01CH	RSCNTL	RSCNT[7：0]							
01DH	LowBatDct	lbEN	lowPwr	-	-	-	-	BiasSEL1	BiasSEL0

- 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 → 0. Then TMCNT and RSCNT stop counting and store the value at the same time.

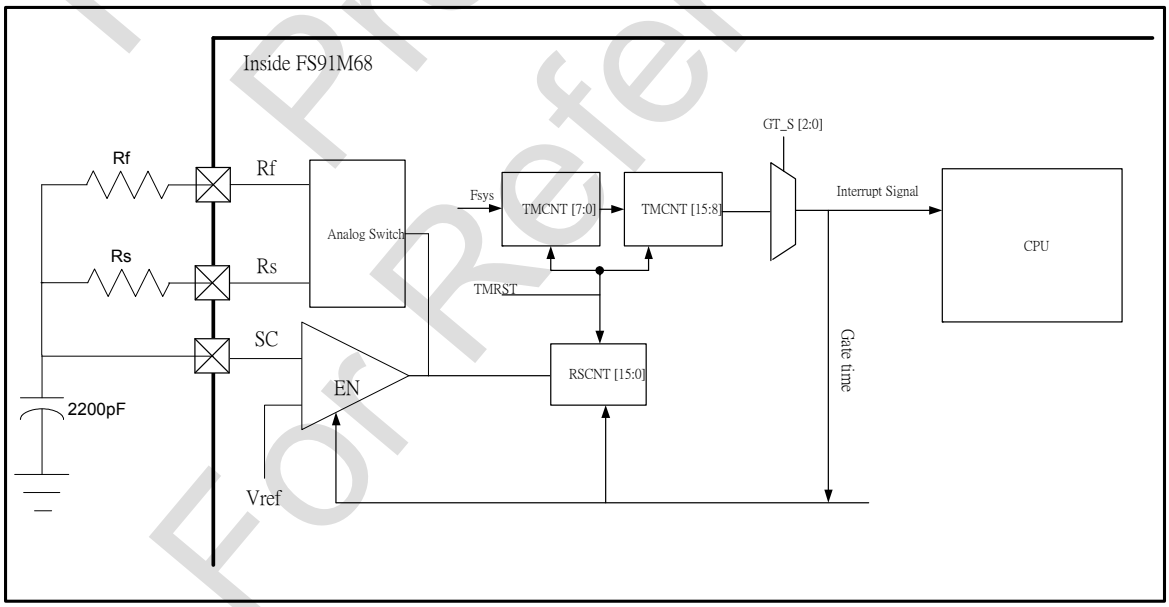


Figure 10-4 R2F Conversion Module

- 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

Address	Name	Content							
		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	-	-	SEG10[2：0]			SEG9[2：0]		
015H	LCD5	-	-	SEG12[2：0]			SEG11[2：0]		
016H	LCD6	-	-	-	-	-	SEG13[2：0]		

- 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] ← [W] + k	1	C, DC, Z
ADDPCW	[PC] ← [PC] + 1 + [W]	2	None
ADDWF f, d	[Destination] ← [f] + [W]	1	C, DC, Z
ADDWFC f, d	[Destination] ← [f] + [W] + C	1	C, DC, Z
ANDLW k	[W] ← [W] AND k	1	Z
ANDWF f, d	[Destination] ← [W] AND [f]	1	Z
BCF f, b	[f<b>] ← 0	1	None
BSF f, b	[f<b>] ← 1	1	None
BTFSC f, b	Skip if [f<b>] = 0	1, 2	None
BTFSS f, b	Skip if [f<b>] = 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] ← NOT([f])	1	Z
DECF f, d	[Destination] ← [f] -1	1	Z
DECFSZ f, d	[Destination] ← [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] ← [f] + 1, skip if the result is zero	1, 2	None
IORLW k	[W] ← [W] | k	1	Z
IORWF f, d	[Destination] ← [W] | [f]	1	Z
MOVFW f	[W] ← [f]	1	None
MOVLW k	[W] ← k	1	None
MOVWF f	[f] ← [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>] ← [f<n>]	1	C,Z
RRF f, d	[Destination<n-1>] ← [f<n>]	1	C, Z
SLEEP	Stop OSC	1	PD
SUBLW k	[W] ← k – [W]	1	C, DC, Z
SUBWF f, d	[Destination] ← [f] – [W]	1	C, DC, Z
SUBWFC f, d	[Destination] ← [f] – [W] –C	1	C, DC, Z
XORLW k	[W] ← [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 (0~7).
- [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**
Syntax:	ADDLW k
	0 ≤ k ≤ FFh
Operation:	[W] ← [W] + k
Flag Affected:	C, DC, Z
Description:	The content of Work register add literal “k” in Work register
Cycle:	1
Example:	Before instruction:
ADDLW 08h	W = 08h
	After instruction:
	W = 10h

ADDWF	**Add W to f**
Syntax:	ADDWF f, d
	0 ≤ f ≤ FFh
	d ∈ [0,1]
Operation:	[Destination] ← [f] + [W]
Flag Affected:	C, CD, Z
Description:	Add the content of the W register and [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in f.
Cycle:	1
Example 1:	Before instruction:
ADDWF OPERAND, 0	OPERAND = C2h
	W = 17h
	After instruction:
	OPERAND = C2h
	W = D9h
Example 2:	Before instruction:
ADDWF OPERAND, 1	OPERAND = C2h
	W = 17h
	After instruction:
	OPERAND = D9h
	W = 17h

ADDPCW	**Add W to PC**
Syntax:	ADDPCW
Operation:	[PC] ← [PC] + 1 + [W], [W] < 79h
	[PC] ← [PC] + 1 + ([W] – 100h), otherwise
Flag Affected:	None
Description:	The relative address PC + 1 + W are loaded into PC.
Cycle:	2
Example 1:	Before instruction:
ADDPCW	W = 7Fh, PC = 0212h
	After instruction:
	PC = 0292h
Example 2:	Before instruction:
ADDPCW	W = 80h, PC = 0212h
	After instruction:
	PC = 0193h
Example 3:	Before instruction:
ADDPCW	W = FEh, PC = 0212h
	After instruction:
	PC = 0211h

ADDWFC	**Add W, f and Carry**
Syntax:	ADDWFC f, d
	0 ≤ f ≤ FFh
	d ∈ [0,1]
Operation:	[Destination] ← [f] + [W] + C
Flag Affected:	C, DC, Z
Description:	Add the content of the W register, [f] and 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.
Cycle:	1
Example:	Before instruction:
ADDWFC OPERAND,1	C = 1
	OPERAND = 02h
	W = 4Dh
	After instruction:
	C = 0
	OPERAND = 50h
	W = 4Dh

ANDLW	AND literal with W
Syntax:	ANDLW k
	0 ≤ k ≤ FFh
Operation:	[W] ← [W] AND k
Flag Affected:	Z
Description:	AND the content of the W register with the eight-bit literal "k". The result is stored in the W register.
Cycle:	1
Example:	Before instruction:
ANDLW 5Fh	W = A3h
	After instruction:
	W = 03h

ANDWF	AND W and f
Syntax:	ANDWF f, d
	0 ≤ f ≤ FFh
	d ∈ [0,1]
Operation:	[Destination] ← [W] AND [f]
Flag Affected:	Z
Description:	AND the content of the W register with [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in f.
Cycle:	1
Example 1:	Before instruction:
ANDWF OPERAND,0	W = 0Fh, OPERAND = 88h
	After instruction:
	W = 08h, OPERAND = 88h
Example 2:	Before instruction:
ANDWF OPERAND,1	W = 0Fh, OPERAND = 88h
	After instruction:
	W = 88h, OPERAND = 08h

BCF	Bit Clear f
Syntax:	BCF f, b
	0 ≤ f ≤ FFh
	0 ≤ b ≤ 7
Operation:	[f<b>] ← 0
Flag Affected:	None
Description:	Bit b in [f] is reset to 0.
Cycle:	1
Example:	Before instruction:
BCF FLAG, 2	FLAG = 8Dh
	After instruction:
	FLAG = 89h

BSF	Bit Set f
Syntax:	BSF f, b
	0 ≤ f ≤ FFh
	0 ≤ b ≤ 7
Operation:	[f<b>] ← 1
Flag Affected:	None
Description:	Bit b in [f] is set to 1.
Cycle:	1
Example:	Before instruction:
BSF FLAG, 2	FLAG = 89h
	After instruction:
	FLAG = 8Dh

BTFSC	Bit Test skip if Clear
Syntax:	BTFSC f, b
	0 ≤ f ≤ FFh
	0 ≤ b ≤ 7
Operation:	Skip if [f<b>] = 0
Flag Affected:	None
Description:	If bit 'b' in [f] is 0, 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 BTFSC FLAG, 2	PC = address (Node)
OP1 :	After instruction:
OP2 :	If FLAG<2> = 0
	PC = address(OP2)
	If FLAG<2> = 1
	PC = address(OP1)

BTFSS	Bit Test skip if Set
Syntax:	BTFSS f, b
	0 ≤ f ≤ FFh
	0 ≤ b ≤ 7
Operation:	Skip if [f<b>] = 1
Flag Affected:	None
Description:	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.
Cycle:	1, 2
Example:	Before instruction:
Node BTFSS FLAG, 2	PC = address (Node)
OP1 :	After instruction:
OP2 :	If FLAG<2> = 0
	PC = address(OP1)
	If FLAG<2> = 1
	PC = address(OP2)

CALL	Subroutine CALL
Syntax:	CALL k 0 ≤ k ≤ 1FFFh
Operation:	Push Stack [Top Stack] ← PC + 1 PC ← k
Flag Affected:	None
Description:	Subroutine Call. First, return address PC + 1 is pushed onto the stack. The immediate address is loaded into PC.
Cycle:	2
Example: **HERE CALL THERE**	Before instruction: PC = address (HERE) After instruction: PC = address (THERE) TOS = address (HERE + 1)

CLRF	Clear f
Syntax:	CLRF f 0 ≤ f ≤ 255
Operation:	[f] ← 0
Flag Affected:	None
Description:	Reset the content of memory address f
Cycle:	1
Example: **CLRF WORK**	Before instruction: WORK = 5Ah After instruction: WORK = 00h

CLRWDT	Clear watch dog timer
Syntax:	CLRWDT
Operation:	Watch dog timer counter will be reset
Flag Affected:	None
Description:	CLRWDT instruction will reset watch dog timer counter.
Cycle:	1
Example: **CLRWDT**	After instruction: WDT = 0

COMF	Complement f
Syntax:	COMF f, d 0 ≤ f ≤ 255 d ∈ [0,1]
Operation:	[f] ← NOT([f])
Flag Affected:	Z
Description:	[f] is complemented. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f]
Cycle:	1
Example 1: **COMF OPERAND,0**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = DCh, OPERAND = 23h
Example 2: **COMF OPERAND,1**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = 88h, OPERAND = DCh

DECF	Decrement f
Syntax:	DECF f, d 0 ≤ f ≤ 255 d ∈ [0,1]
Operation:	[Destination] ← [f] -1
Flag Affected:	Z
Description:	[f] is decremented. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f].
Cycle:	1
Example 1: **DECF OPERAND,0**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = 22h, OPERAND = 23h
Example 2: **DECF OPERAND,1**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = 88h, OPERAND = 22h

DECFSZ	Decrement f, skip if zero
Syntax:	DECFSZ f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [f] -1, skip if the result is zero
Flag Affected:	None
Description:	[f] is decremented. 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: **Node DECFSZ FLAG, 1** OP1 : OP2 :	Before instruction: PC = address (Node) After instruction: [FLAG] = [FLAG] - 1 If [FLAG] = 0 PC = address(OP1) If [FLAG] ≠ 0 PC = address(OP2)

GOTO	Unconditional Branch
Syntax:	GOTOk 0 ≤ k ≤ 1FFFh
Operation:	PC ← k
Flag Affected:	None
Description:	The immediate address is loaded into PC.
Cycle:	2
Example:	After instruction:
GOTO THERE	PC = address (THERE)

INCF	Increment f
Syntax:	INCF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [f] +1
Flag Affected:	Z
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].
Cycle:	1
Example 1:	Before instruction:
INCF OPERAND,0	W = 88h, OPERAND = 23h After instruction: W = 24h, OPERAND = 23h
Example 2:	Before instruction:
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] ← [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:
OP2 :	[FLAG] = [FLAG] + 1 If [FLAG] = 0 PC = address(OP2) If [FLAG] ≠ 0 PC = address(OP1)

IORLW	Inclusive OR literal with W
Syntax:	IORLW k 0 ≤ k ≤ FFh
Operation:	[W] ← [W] | k
Flag Affected:	Z
Description:	Inclusive 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:
IORLW 85H	W = 69h After instruction: W = EDh

IORWF	Inclusive OR W with f
Syntax:	IORWF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [W] | [f]
Flag Affected:	Z
Description:	Inclusive OR the content of the W register and [f]. If d is 0, the result is stored in the W register. If d is 1, the result is stored back in [f].
Cycle:	1
Example:	Before instruction:
IORWF OPERAND,1	W = 88h, OPERAND = 23h After instruction: W = 88h, OPERAND = ABh

MOVFW	Move f to W
Syntax:	MOVFW f
	0 ≤ f ≤ FFh
Operation:	[W] ← [f]
Flag Affected:	None
Description:	Move data from [f] to the W register.
Cycle:	1
Example:	Before instruction:
MOVFW OPERAND	W = 88h, OPERAND = 23h
	After instruction:
	W = 23h, OPERAND = 23h

MOVLW	Move literal to W
Syntax:	MOVLW k
	0 ≤ k ≤ FFh
Operation:	[W] ← k
Flag Affected:	None
Description:	Move the eight-bit literal "k" to the content of the W register.
Cycle:	1
Example:	Before instruction:
MOVLW 23H	W = 88h
	After instruction:
	W = 23h

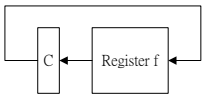
MOVWF	Move W to f
Syntax:	MOVWF f
	0 ≤ f ≤ FFh
Operation:	[f] ← [W]
Flag Affected:	None
Description:	Move data from the W register to [f].
Cycle:	1
Example:	Before instruction:
MOVWF OPERAND	W = 88h, OPERAND = 23h
	After instruction:
	W = 88h, OPERAND = 88h

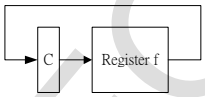
NOP	No Operation
Syntax:	NOP
Operation:	No Operation
Flag Affected:	None
Description:	No operation. NOP is used for one instruction cycle delay.
Cycle:	1

RETFIE	Return from Interrupt
Syntax:	RETFIE
Operation:	[Top Stack] => PC
	Pop Stack
	1 => GIE
Flag Affected:	None
Description:	The program counter is loaded from the top stack, then pop stack. Setting the GIE bit enables interrupts.
Cycle:	2
Example:	After instruction:
RETFIE	PC = [Top Stack]
	GIE = 1

RETLW	Return and move literal to W
Syntax:	RETLW k
	0 ≤ k ≤ FFh
Operation:	[W] ← k
	[Top Stack] => PC
	Pop Stack
Flag Affected:	None
Description:	Move the eight-bit literal "k" to the content of the W register. The program counter is loaded from the top stack, then pop stack.
Cycle:	2
Example:	Before instruction:
CALL TABLE	WREG = 0x07
:	After instruction:
TABLE	WREG = value of k7
ADDWF PC	
RETLW k0	
RETLW k1	
:	
RETLW kn	

Return	Return from Subroutine
Syntax:	RETURN
Operation:	[Top Stack] => PC
	Pop Stack
Flag Affected:	None
Description:	The program counter is loaded from the top stack, then pop stack.
Cycle:	2
Example:	After instruction:
Return	PC = [Top Stack]

RLF	Rotate left [f] through Carry
Syntax:	RLF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination<n+1>] ← [f<n>] [Destination<0>] ← C C ← [f<7>]
Flag Affected:	C, Z
Description:	[f] is rotated one bit to the left 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].
	
Cycle:	1
Example: **RLF OPERAND, 1**	Before instruction: C = 0 W = 88h, OPERAND = E6h After instruction: C = 1 W = 88h, OPERAND = CCh

RRF	Rotate right [f] through Carry
Syntax:	RRF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination<n-1>] ← [f<n>] [Destination<7>] ← C C ← [f<7>]
Flag Affected:	C
Description:	[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].
	
Cycle:	1
Example: **RRF OPERAND, 0**	Before instruction: C = 0 OPERAND = 95h After instruction: C = 1 W = 4Ah, OPERAND = 95h

SLEEP	Oscillator stop
Syntax:	SLEEP
Operation:	CPU oscillator is stopped
Flag Affected:	PD
Description:	CPU oscillator is stopped. CPU can be waked up by external interrupt sources.
Cycle:	1
Example: **SLEEP**	After instruction: PD = 1 TO = 0 If WDT causes wake up, TO = 1

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

SUBLW	Subtract W from literal
Syntax:	SUBLW k 0 ≤ k ≤ FFh
Operation:	[W] ← k – [W]
Flag Affected:	C, DC, Z
Description:	Subtract the content of the W register from the eight-bit literal "k". The result is stored in the W register.
Cycle:	1
Example 1: **SUBLW 02H**	Before instruction: W = 01h After instruction: W = 01h C = 1 Z = 0
Example 2: **SUBLW 02H**	Before instruction: W = 02h After instruction: W = 00h C = 1 Z = 1
Example 3: **SUBLW 02H**	Before instruction: W = 03h After instruction: W = FFh C = 0 Z = 0

SUBWF	Subtract W from f
Syntax:	SUBWF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [f] – [W]
Flag Affected:	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. If d is 1, the result is stored back in [f],
Cycle:	1
Example 1: **SUBWF OPERAND, 1**	Before instruction: OPERAND = 33h, W = 01h After instruction: OPERAND = 32h C = 1 Z = 0
Example 2: **SUBWF OPERAND, 1**	Before instruction: OPERAND = 01h, W = 01h After instruction: OPERAND = 00h C = 1 Z = 1
Example 3: **SUBWF OPERAND, 1**	Before instruction: OPERAND = 04h, W = 05h After instruction: OPERAND = FFh C = 0 Z = 0

SUBWFC	Subtract W and Carry from f
Syntax:	SUBWFC f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [f] – [W] –Ċ
Flag Affected:	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. If d is 1, the result is stored back in [f].
Cycle:	1
Example 1: **SUBWFC OPERAND, 1**	Before instruction: OPERAND = 33h, W = 01h C = 1 After instruction: OPERAND = 32h, C = 1, Z = 0
Example 2: **SUBWFC OPERAND, 1**	Before instruction: OPERAND = 02h, W = 01h C = 0 After instruction: OPERAND = 00h, C = 1, Z = 1
Example 3: **SUBWFC OPERAND, 1**	Before instruction: OPERAND = 04h, W = 05h C = 0 After instruction: OPERAND = FEh, C = 0, Z = 0

XORLW	Exclusive OR literal with W
Syntax:	XORLW k 0 ≤ k ≤ FFh
Operation:	[W] ← [W] XOR k
Flag Affected:	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: **XORLW 5Fh**	Before instruction: W = ACh After instruction: W = F3h

XORWF	Exclusive OR W and f
Syntax:	XORWF f, d 0 ≤ f ≤ FFh d ∈ [0,1]
Operation:	[Destination] ← [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. If d is 1, the result is stored back in [f].
Cycle:	1
Example: **XORWF OPERAND, 1**	Before instruction: OPERAND = 5Fh, W = ACh 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(10kΩ) and capacitor(2200pF).
1.3	2014/5/22	2	Revised company address