Datasheet

FS98O02

8-bit MCU with 2k program EPROM, 128-byte RAM,
14-bit ADC, 12-bit GIO, 4 × 12 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 Dist., New Taipei City 251, Taiwan
Tel.：886-2-28094742
Fax：886-2-28094874
www.ic-fortune.com

Contents

1. General Description

The FS98O02 is a high performance, low cost 8-bit MCU with 2k program EPROM, 128-byte data RAM, one 14-bit ADC, 12-bit GIO, and 4 × 12 LCD driver. With a few external passive components such as resistors and capacitors, the FS98O02 can be easily implemented to form a simple portable tire gauge, voltage panel meter, or manual range DMM, etc. Especially, FS98O02's EPROM could be written by program instruction, so users can write table or calibration data in the unused EPROM address more than one time.

2. Features

- 8-bit RISC CPU core with 39 single word instructions.
- Embedded 2k x 16 program ROM（07ffH isn't usable by user）, 128-byte data RAM.
- Instruction Programmer Function.
- Operating voltage is from 2.4V to 3.6V.
- Operating current is about 1.5mA; sleep current is about 2μA.
- Embedded internal 1MHz / 4MHz oscillator.
- 4-level hardware stacks.
- 3 Interrupt sources（external: input port 1<0>, internal: timer, ADC）.
- One 14-bit noise free ADC.
- Embedded Voltage Regulator（2.5V / 3.6V regulated output for **FS98O02A** and **FS98O02C**）.
- Embedded Voltage Regulator（2.5V / $VDD_{(max.\ 3.6V)}$ regulated output for **FS98O02B**）.
- Embedded Voltage Doubler（3V / 2xVDDP pumped output）.
- 4-bit Input Port and 8-bit bi-directional I/O port including 1-bit for buzzer output.
- Embedded Low Voltage Reset（LVR）and Low Battery Detector（LBD）.
- 4 x 12 LCD driver（3V peak-to-peak）.
- Watchdog timer.

3. Applications

- Simple portable tire gauge.
- Voltage panel meter.
- Manual range DMM.
- Scale.

4. Ordering Information

Product Number	Description	Package Type
FS98O02A-D	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	49-pin Dice form
FS98O02A-nnnV-D	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	49-pin Dice form
FS98O02A-PCE	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	64-pin LQFP
FS98O02A-nnnV-PCE	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	64-pin LQFP
FS98O02B-D	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	49-pin Dice form
FS98O02B-nnnV-D	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	49-pin Dice form
FS98O02B-PCE	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	64-pin LQFP
FS98O02B-nnnV-PCE	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	64-pin LQFP
FS98O02C-D	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	49-pin Dice form
FS98O02C-nnnV-D	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	49-pin Dice form
FS98O02C-PCE	MCU with OTP ROM；The customer has to program the compiled hex code into OTP ROM.	64-pin LQFP
FS98O02C-nnnV-PCE	MCU with program type；FSC programs the customer's compiled hex code into OTP ROM at factory before shipping.	64-pin LQFP

Note1：Code number（nnnV）is assigned for customer.
Note2：Code number（nnn = 001~999）; Version（V = A~Z）
Note3：PCE means package of Pb-free and LQFP 64 pin.

5. Pin Configuration

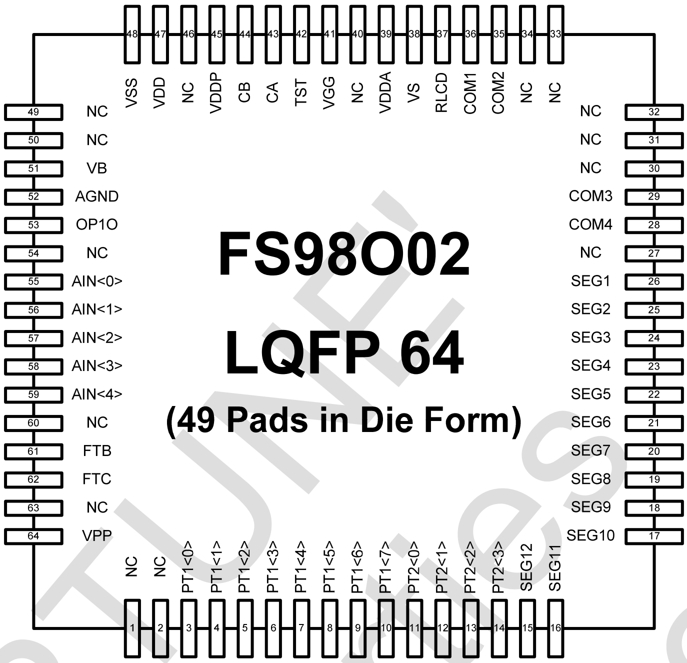


Figure 5-1: LQFP64

6. Pin Description

Name	In/Out	Pin No	Description
P1<0>/INT	I	3	Input port 1.0 or interrupt input
P1<1>/PROEN	I	4	Input port1.1 , SPI or Instruction Programmer select
P1<2>~P1<3>	I	5~6	Input port 1.2~1.3
P1<4>~P1<5>	I/O	7~8	I/O port 1.4~1.5
P1<6>/ Analog Input	I/O	9	I/O port 1.6 or Analog channel input
PT1<7>/BZ	I/O	10	I/O port 1.7 or buzzer output
PT2<0>~PT2<3>	I/O	11~14	I/O port 2.0~2.3
SEG12~SEG1	O	15~26	LCD segment driver output
COM4~COM3	O	28~29	LCD common driver output
COM2~COM1	O	35~36	LCD common driver output
RLCD	I	37	LCD Voltage Input（usually connect 10~100kΩ to VDDA or 3V VGG）
VS	O	38	Voltage source from VDDA
VDDA	O	39	Voltage regulator power output（2.5V / 3.6V for **FS98O02A**、**FS98O02C** or 2.5V / $VDD_{(max.\ 3.6V)}$ for **FS98O02B**）
VGG	O	41	Voltage doubler output（3V or 2xVDDP）
TST	I	42	Test Mode control pin（low active）
CA	I/O	43	Voltage doubler capacitor positive connection

Name	In/Out	Pin No	Description
CB	I/O	44	Voltage doubler capacitor negative connection
VDDP	I	45	Analog power supply
VDD	I	47	Positive power supply
VSS	I	48	Negative power supply（ground）
VB	I	51	Analog circuit bias current input
AGND	I/O	52	Analog ground
OP1O	O	53	OPAMP output
AIN0(AD0)~ AIN1(AD1)	I	55~56	Analog signal input channel
AIN2(AD2)	I	57	Analog signal input channel (usually for ADC VIL input)
AIN3(AD3)	I	58	Analog signal input channel (usually for ADC VRH input)
AIN4(AD4)	I	59	Analog signal input channel (usually for ADC VRL input)
FTB, FTC	I/O	61, 62	ADC pre-filter capacitor connection
VPP/RST	I	64	Program Input Voltage or Reset

7. Functional Block Diagram

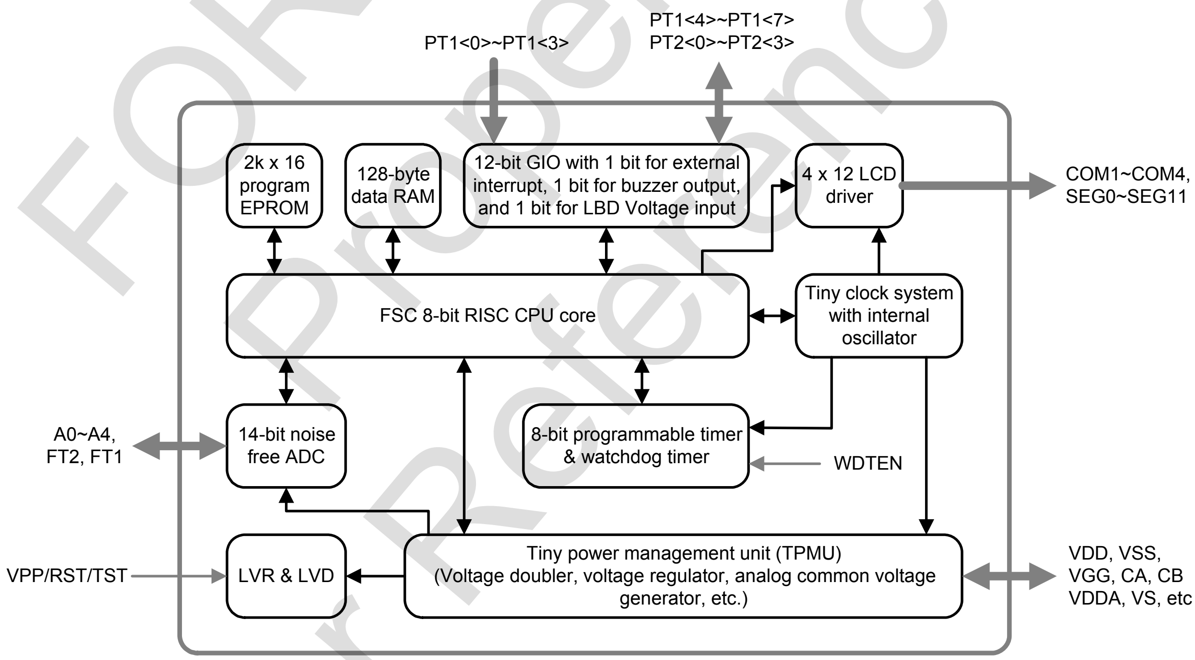


Figure 7-1: Functional Block Diagram

8. Typical Application Circuit

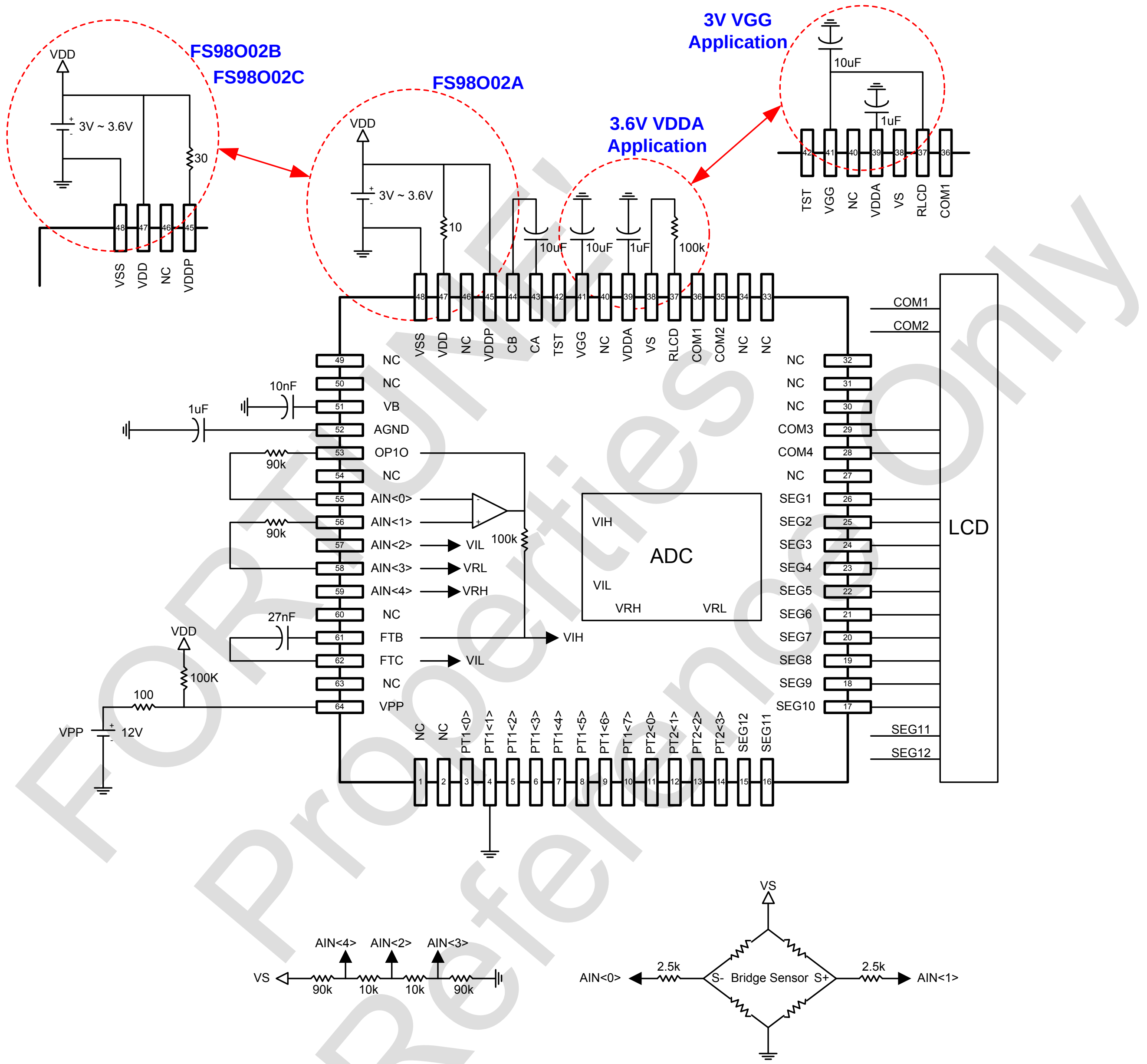
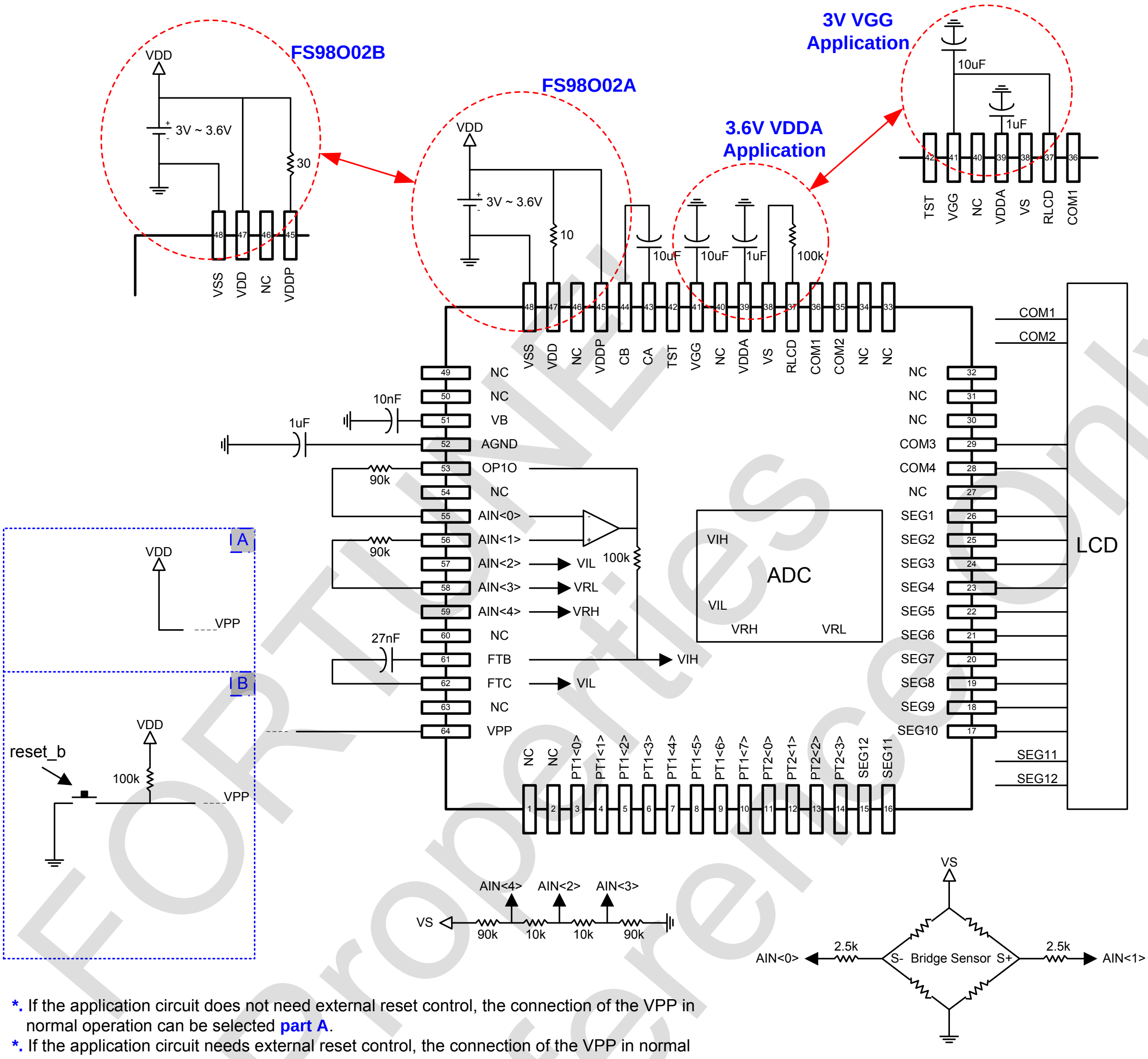


Figure 8-1: Scale, Instruction Programmer Calibration Data to EPROM

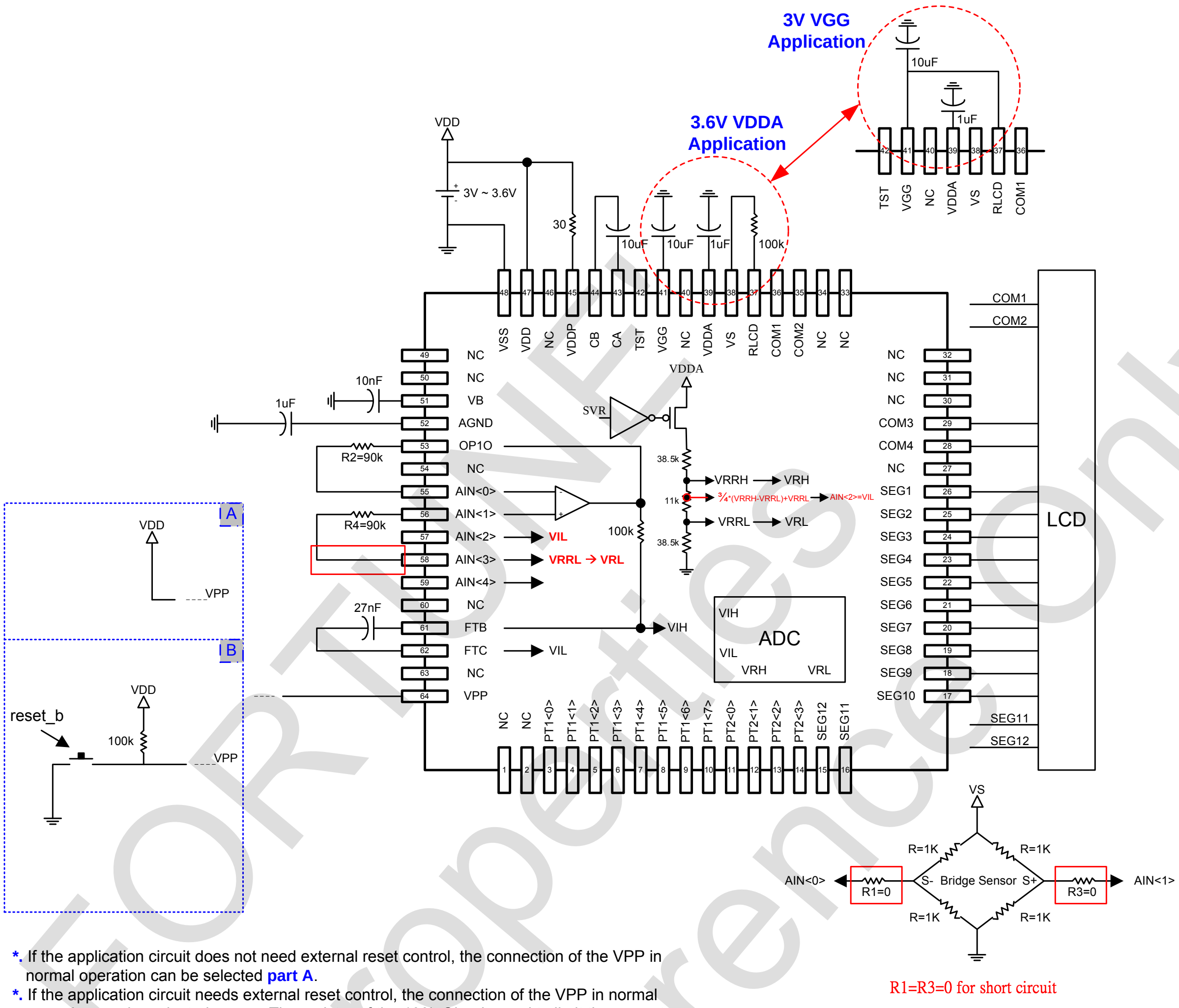
Note. In the instruction program mode, VPP must be connected to 100Ω Resistor. Please keep VGG between 5.4 and 6.2V when executing instruction programming. Please turn off VDDA or VS before executing instruction programming if the loading of VDDA or VS is over 8 mA. And turn off the LCD bias circuit in the instruction program mode if the application circuit is VGG pin connect to RLCD pin.



*. If the application circuit does not need external reset control, the connection of the VPP in normal operation can be selected **part A**.

*. If the application circuit needs external reset control, the connection of the VPP in normal operation can be selected **part B**. The resistor of the 100k Ohm is used to limit the current.

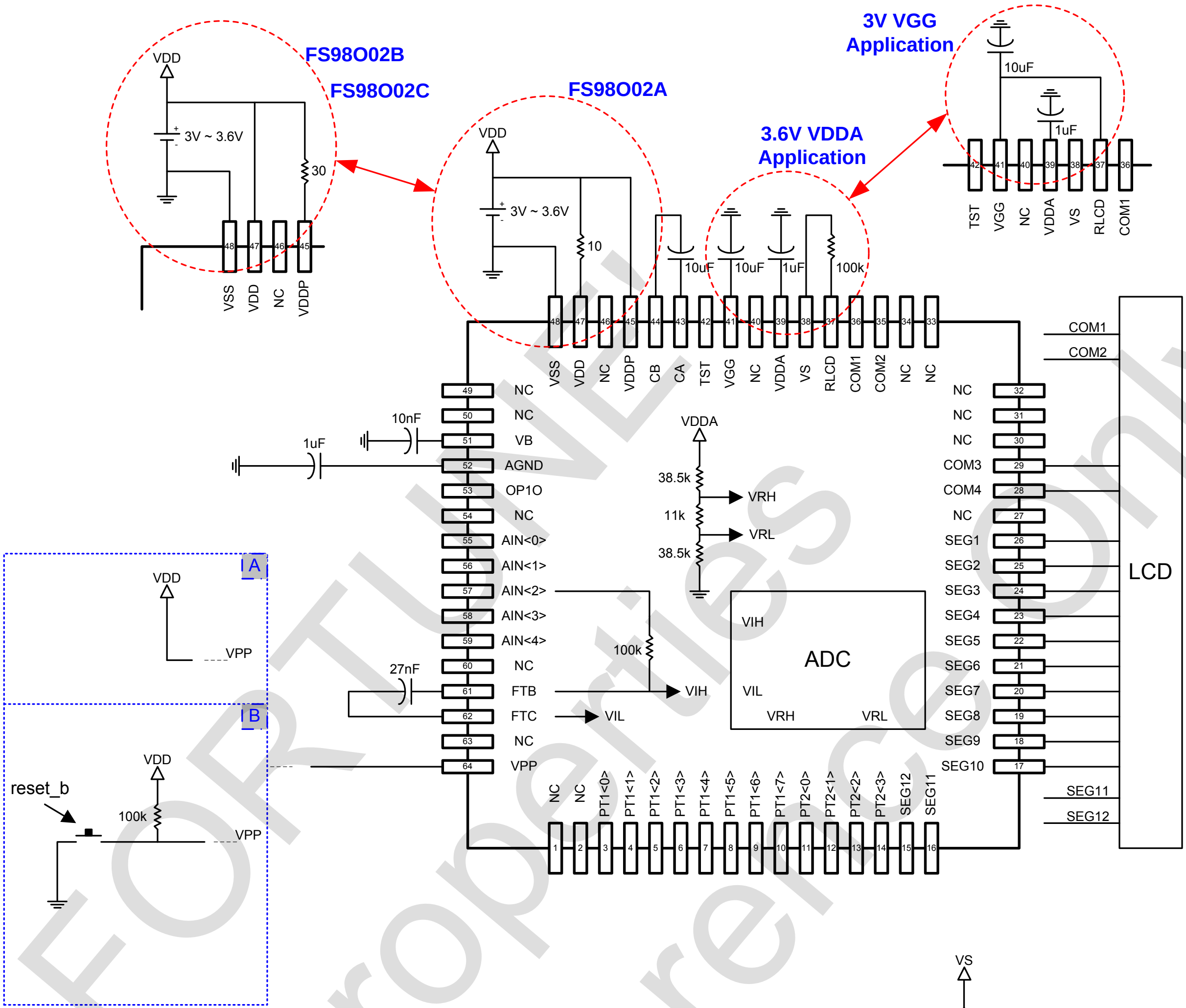
*Figure 8-2: Scale, Normal mode for **FS98O02A** and **FS98O02B***

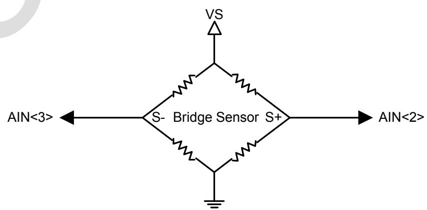


*. If the application circuit does not need external reset control, the connection of the VPP in normal operation can be selected **part A**.

*. If the application circuit needs external reset control, the connection of the VPP in normal operation can be selected **part B**. The resistor of the 100k Ohm is used to limit the current.

*Figure 8-3: Scale, Normal mode for **FS98O02C***





*. If the application circuit does not need external reset control, the connection of the VPP in normal operation can be selected **part A**.

*. If the application circuit needs external reset control, the connection of the VPP in normal operation can be selected **part B**. The resistor of the 100k Ohm is used to limit the current.

Figure 8-4: Tire Gauge

9. Electrical Characteristics

9.1 Absolute Maximum Ratings

Parameter	Rating	Unit
Supply voltage to ground	-0.3 to 5.5	V
Input/output voltage to ground	-0.3 to VDD+0.3	V
Operating temperature	-40 to +85	°C
Storage temperature	-55 to +150	°C
Soldering temperature/Time	260°C/10 Sec	
ESD immunity, Human Body Model/Machine Model	≤1.5kV/200V	
Latch-up immunity	≤100mA	

9.2 DC Characteristics

(VDD = 3V, T_A = 25℃, unless otherwise noted)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
VDD	Supply voltage	-40 to +85 °C	2.4		3.6	V
IDD	Supply current	In operating mode		1.5		mA
IPD	Sleep mode supply current	In sleep mode, LVR enable		2		μA
VIH	Digital Input high voltage	PT1, RST_	0.7			VDD
VIL	Digital Input low voltage	PT1, RST_			0.3	VDD
VIHSH	Input Hys. High Voltage	Schmitt-trigger port		0.45		VDD
VIHSL	Input Hys. Low Voltage	Schmitt-trigger port		0.20		VDD
IOH	High Level Output Current	VOH=VDD-0.3 V		3		mA
IOL	Low Level Output Current	VOL=0.3 V		5		mA
VSR	Voltage source switch resistor			10		Ω
VDDA	Analog Power for 3.6V Output（no load）for **FS98O02A** and **FS98O02C**	VGG ＞ 4V	3.45	3.6	3.85	V
	Analog Power for $VDD_{(max.\ 3.6V)}$ Output（no load）for **FS98O02B**	VDD ≤ 4V			3.6	V
	Analog Power for 2.5V Output（no load）	VGG ＞ 2.7V	2.37	2.5	2.67	V
KTCREF	VDDA temperature coefficient	T_A = 0 ~ 50℃		100		ppm/℃
VLBAT	Low battery detection voltage	VDD = 2.5V, [ADOH,ADOM] is about 2710h	2.2	2.4	2.6	V
VLVR	Low voltage reset voltage			1.65		V
VLCD	LCD driver peak to peak voltage		2.6	2.8	3.0	V
FCK	Internal RC oscillator frequency for 1MHz		0.8	1.0	1.2	MHz
	Internal RC oscillator frequency for 4MHz		3.0	4.0	4.8	MHz
FWDT	Internal WDT Clock			1.0		kHz
VPP	Instruction Programmer input Voltage	PROEN = 0	11	12	13	V

9.3 ADC Characteristics

(VDD = 3V, T_A = 25℃, unless otherwise noted)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
VDIN	ADC differential input voltage range	To VSS	1		2.2	V
VRIN	ADC reference input voltage range	(VRH, VRL), ADC Gain = 1	0.25		0.5	V
	Resolution			±15625		Counts
	ADC linearity error	VRIN = 0.44V for FRC=1MHz	-0.1	0	+0.1	mV

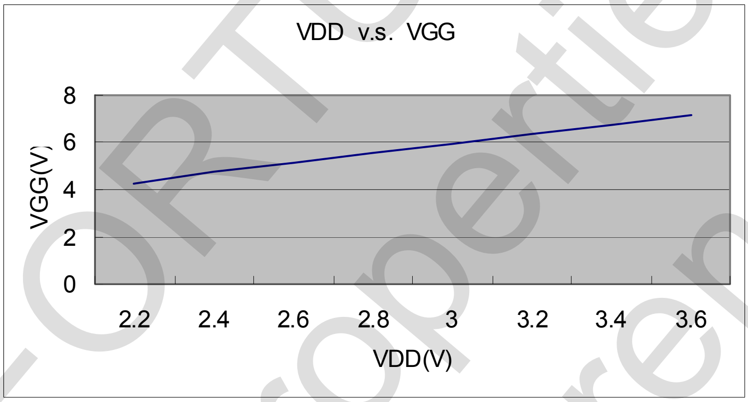
9.4 OPAMP Characteristics

(VDD=3V, T_A=25℃, unless otherwise noted)

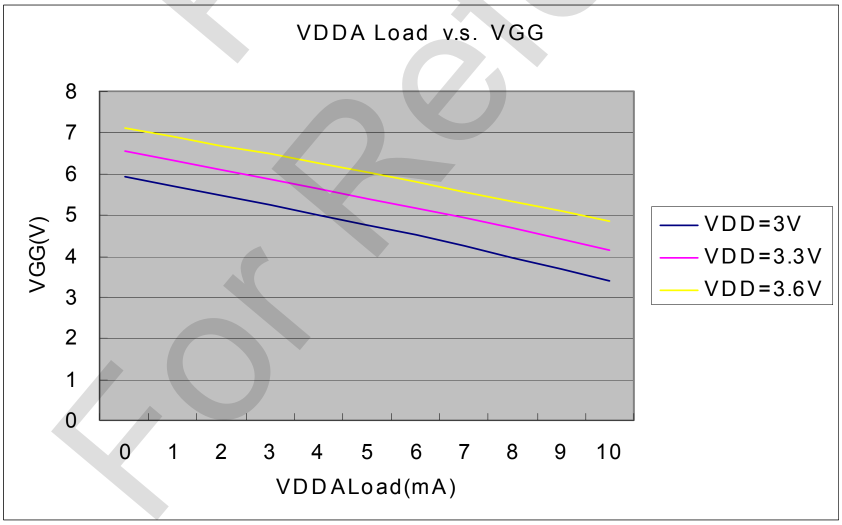
Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
	Input Offset			1.5		mV
	Input Offset Voltage with Chopper	Rs<100Ω		20		μV
	Input Reference Noise	Rs=100Ω, 0.1Hz~1Hz		1.0		μVpp
	Input Reference Noise with Chopper	Rs=100Ω, 0.1Hz~1Hz		0.5		μVpp
	Input Bias Current			10	30	pA
	Input Bias Current with Chopper			100	300	pA
	Input Common Mode Range		0.5		2.4	V
	Output Voltage Range		0.5		2.4	V
	Chopper Clock Frequency	S_CHCK[1:0]=11		1k		Hz
	Capacitor Load			50	100	pF

10. Typical Performance Characteristics

1. VDD(V) vs. VGG(V)
   Temp= 25℃, VGG Capacitor= 10μF, Charge pump Capacitor CA-CB= 10μF, VDDA and VS(no loading)



2. VDDA Load(mA) vs. VGG(V)
   Temp = 25℃, VGG Capacitor= 10μF, Charge pump Capacitor CA-CB= 10μF



11. CPU Core

Figure 11-1 shows the CPU core block diagram used in FS98O02.

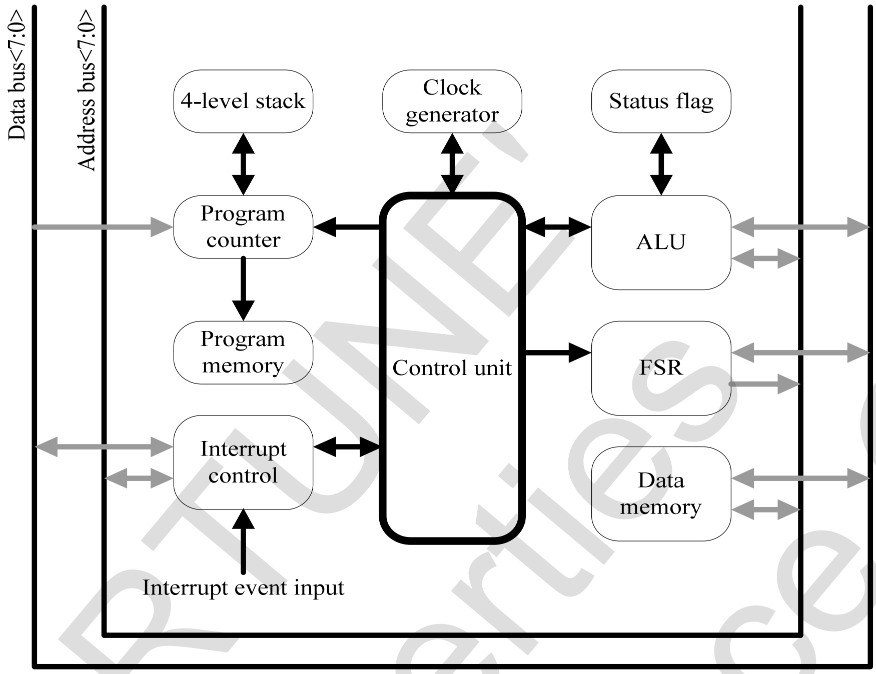


Figure 11-1: CPU Core Block Diagram

11.1 Program Memory Organization

The CPU has a 10-bit program counter capable of address up to 2k x 16 program memory space. The reset vector is at 0000h and the interrupt vector is at 0004h.

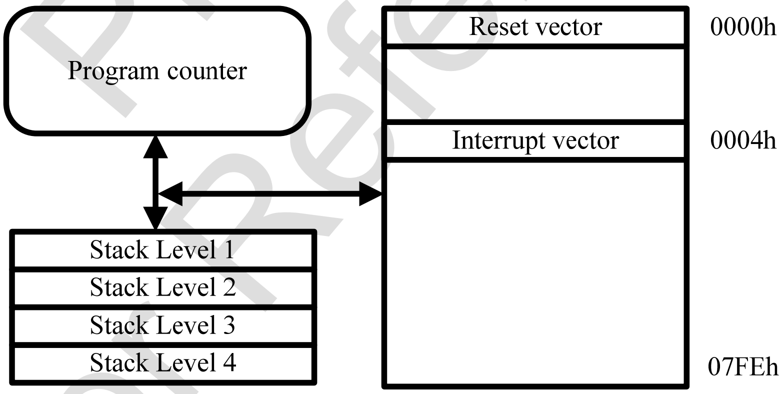


Figure 11-2: Program Memory Origination

11.2 Data Memory Organization

The data memory is partitioned into three parts. The address 00h~07h areas are system special registers, like indirect address, indirect address pointer, status register, working register, interrupt flag, interrupt control register. The address 08h~7Fh areas are peripheral special registers, like I/O ports, timer, ADC, signal conditional network control register, LCD driver. The address 80h~FFh areas are general data memory.

Table 11-1: Data Memory Organization

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State	Details on page:
00h	IND0	Use contents of FSR0 to address data memory								uuuu uuuu	uuuu uuuu	16
02h	FSR0	Indirect data memory, address point 0								uuuu uuuu	uuuu uuuu	16
04h	STATUS				PD	TO	DC	C	Z	---0 0uuu	---u 1uuu	16
05h	WORK	WORK register								uuuu uuuu	uuuu uuuu	16
06h	INTF				TMIF			ADIF	E0IF	---0 --00	---0 –00	25, 27, 28
07h	INTE	GIE			TMIE			ADIE	E0IE	0--0 --00	0--0 –00	25, 27, 28
08h~7Fh		Peripheral special registers										18
80h~FFh		General data memory (128-byte SRAM)								uuuu uuuu	uuuu uuuu	

Note 1: "u" means unknown or unchanged. "-" means unimplemented, read as "0".
Note 2: The "Reset State" indicates the registers state after external reset and low voltage reset.
Note 3: The "WDT Reset State" indicates the registers state after watchdog time out reset.

11.3 System Special Registers

System special registers are used by the CPU to control the operation of the device. Some registers are used for data memory access, some for logic judgment, and some for arithmetic, etc.

Table 11-2: System Special Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State	Details on page:
00h	IND0	Use contents of FSR0 to address data memory								uuuu uuuu	uuuu uuuu	16
02h	FSR0	Indirect data memory, address point 0								uuuu uuuu	uuuu uuuu	16
04h	STATUS				PD	TO	DC	C	Z	---0 0uuu	---u 1uuu	16
05h	WORK	WORK register								uuuu uuuu	uuuu uuuu	16
06h	INTF				TMIF			ADIF	E0IF	---0 --00	---0 --00	25, 27, 28
07h	INTE	GIE			TMIE			ADIE	E0IE	0--0 --00	0--0 --00	25, 27, 28

Note 1: "u" means unknown or unchanged. "-" means unimplemented, read as "0".
Note 2: The "Reset State" indicates the registers state after external reset and low voltage reset.
Note 3: The "WDT Reset State" indicates the registers state after watchdog time out reset.

- **IND0 : Address 00h**

The IND0 registers at data memory address are not physical registers. Any instruction using the IND0 register actually access the data pointed by the FSR0 register.

A simple program to clear data memory 80h-0BFh using indirect addressing is shown as Example 11-1.

```
        MOVLW   080h
        MOVWF   FSR0
NEXT:   CLRF    IND0        ; Clear the content of memory address pointed by FSR0
        INCFSZ  FSR0, 1     ; FSR0 = FSR0 + 1, and judge if FSR0 = 0
        GOTO    NEXT
```

Example 11-1: Using Indirect Addressing

- **FSR0 : Address 02h**

Indirect addressing pointers FSR0 correspond to IND0 respectively.

- **STATUS : Address 04h**

The STATUS register contains the arithmetic status of the ALU. The function of each bit in STATUS register is described in Table 11-3.

Table 11-3: Status Register

Bit	Symbol	Description
7~5	-	No use.
4	PD	Power down flag. 1: After power on reset or cleared by writing 0 (which shuts off oscillator clock, thus neither of the MCU clock or operation will be in conduct). 0: By execution of the SLEEP instruction, but not the HALT instruction (which only turns off the MCU clock).
3	-	No use.
2	DC	Digit carry flag (ADDWF, SUBWF instructions) 1: A carry-out from the 4th low order bit of the result occurred. 0: No carry-out from the 4th low order bit of the result.
1	C	Carry flag (~Borrow)
0	Z	Zero flag 1: The result of an arithmetic or logic operation is zero. 0: The result of an arithmetic or logic operation is not zero.

- **WORK : Address 05h**

WORK register is used to store temporary data for arithmetic, data moving, etc.

- **INTF, INTE: Address 06h, 07h**

The interrupt enable register (INTE) records individual interrupt request. When some interrupt event occurs and related interrupt enable bit = 1, the related interrupt flag in interrupt flag register (INTF) will be set. The global interrupt enable bit (GIE) will enable CPU interrupt procedure. When GIE = 1 and any interrupt flag is set, CPU interrupt procedure would be executed. CPU interrupt procedure executes GIE reset and CALL 0004h.

When interrupt signal happened within instruction duty cycle, the CPU must wait and till instruction duty cycle end of this program then produce an “Interrupt Flag” before go into next step.

This Example 11-2 program is specially mentioned here for halt and sleep mode.

```
MAIN:
          HALT
          NOP
          GOTO      MAIN
MAIN_SLEEP:
          CLRF      INTF
          SLEEP
          NOP
          GOTO      SYSINI
```

Example 11-2: Halt and Sleep Mode Example

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

The INTF register contains the status of every interrupt event. The function of each bit in INTF register is described in Table 11-4.

Table 11-4: INTF Register

Bit	Symbol	Description
7~5	-	No use.
4	TMIF	8-bit timer Interrupt flag.
3~2	-	No use.
1	ADIF	Analog to digital converter Interrupt flag.
0	E0IF	PT1<0> external interrupt flag.

The INTE register defines if the CPU will accept related interrupt event. The function of each bit in INTE register is described in Table 11-5.

Table 11-5: INTE Register

Bit	Symbol	Description
7	GIE	Global interrupt enable bit.
6~5	-	No use.
4	TMIE	8-bit timer Interrupt enable bit.
3~2	-	No use.
1	ADIE	Analog to digital converter Interrupt enable bit.
0	E0IE	PT1<0> external interrupt enable bit.

11.4 Peripheral Special Registers

The peripheral special registers are used to control I/O ports, timer, ADC, signal conditional network, LCD driver, and others.

Table 11-6: Peripheral Special Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State	Details on page:
09h	PCK					S_CHCK[1:0]		S_BEEP	EN_OSC4M	---- 0000	---- 0000	24, 25
0Ah	EADRH	PAR[15:8]								uuuu uuuu	uuuu uuuu	35
0Bh	EADRL	PAR[7:0]								uuuu uuuu	uuuu uuuu	35
0Ch	EDAH	EDATA[15:8]								uuuu uuuu	uuuu uuuu	35
0Dh	TMOUT	TMOUT [7:0]								0000 0000	0000 0000	25, 27
0Eh	TMCON	TMRST	WDTEN	WTS [1:0]		TMEN	INS [2:0]			1000 0000	1u00 0000	25, 27
0Fh	TMMOD	TMMOD [7:0]								0000 0000	0000 0000	27
10h	ADOH	ADO [23:16]								0000 0000	0000 0000	29
11h	ADOM	ADO [15:8]								0000 0000	0000 0000	29

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State	Details on page:
12h	ADOL	ADO [7:0]								0000 0000	0000 0000	29
13h	ADCON					ADRST	ADM [2:0]			---- 0000	---- 0000	29
1Fh	SVD								SVD	---- ---1	---- ---1	19, 22
20h	PT1	PT1 [7:0]								uuuu uuuu	uuuu uuuu	25
21h	PT1EN	PT1EN[7:4]								0000 ----	0000 ----	25
22h	PT1PU	PT1PU [7:0]								0000 0000	0000 0000	25
23h	PT1MR	BPE	EN_LBDAIN					E0M [1:0]		00-- --00	00-- --00	25
24h	PT2					PT2 [3:0]				---- uuuu	---- uuuu	26
25h	PT2EN					PT2EN [3:0]				---- 0000	---- 0000	26
26h	PT2PU					PT2PU [3:0]				---- 0000	---- 0000	26
2Ah	NETB		SINL[1:0]		SINH [1:0]		SFT [2]		SFT [0]	-000 00-0	-000 00-0	30
2Ch	NETD	EPMAT	SVRH[0]	SVRL[1:0]		ERV	EPBLK	SLVD	SVR	u000 uu00	u000 uu00	30, 35
2Eh	NETF	EN_PUMP	EN_VS	EN_LCDB	LCDEN	EN_VGG3	EN_VDDAPU	EN_VDDA25	EN_VDDA	0000 0100	0000 0100	19, 36
2Fh	NETG					ADG [1:0]		ADEN	AZ	---- 0000	---- 0000	29
33h	NETK	SILB[2:0]			EN_LB	OP1EN	SOP1P[1:0]		SOP1N	0000 0000	0000 0000	22, 34
40h	LCD1	SEG1 [3:0]				SEG0 [3:0]				uuuu uuuu	uuuu uuuu	36
41h	LCD2	SEG3 [3:0]				SEG2 [3:0]				uuuu uuuu	uuuu uuuu	36
42h	LCD3	SEG5 [3:0]				SEG4 [3:0]				uuuu uuuu	uuuu uuuu	36
43h	LCD4	SEG7 [3:0]				SEG6 [3:0]				uuuu uuuu	uuuu uuuu	36
44h	LCD5	SEG9 [3:0]				SEG8 [3:0]				uuuu uuuu	uuuu uuuu	36
45h	LCD6	SEG11 [3:0]				SEG10 [3:0]				uuuu uuuu	uuuu uuuu	36

Note 1: “u” means unknown or unchanged. “-“ means unimplemented, read as “0”.
Note 2: The “Reset State” indicates the registers state after external reset and low voltage reset.
Note 3: The “WDT Reset State” indicates the registers state after watchdog time out reset.

12. Power System

There’re some important power management blocks in FS98O02, such as voltage doublers, voltage regulator, analog bias circuit, analog common voltage generator, etc. The power system related registers are in Table 12-1.

Table 12-1: Power System Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
1Fh	SVD								SVD	---- ---1	---- ---1
2Eh	NETF	EN_PUMP	EN_VS	EN_LCDB	LCDEN	EN_VGG3	EN_VDDAPU	EN_VDDA25	EN_VDDA	0000 0100	0000 0100
33h	NETK	SILB[2:0]			EN_LB					0000 0000	0000 0000

12.1 Voltage Doubler

The voltage doubler is used to generate double voltage of VDDP or 3V VGG output. The doubled voltage of VDDP is used for regulator power supply and the 3V VGG output can be used for LCD power supply. When voltage doubler is turned off（EN_PUMP = 0）, the default VGG is shorted to VDDP and this VGG output can be used for LCD power supply. During the LCD application, if the VDDP voltage is reduced, the brightness of the LCD may be darkened. User can turn on voltage doubler by EN_PUMP after setting the EN_VGG3 to pump the VGG voltage to 3V to supply LCD circuit. It can lighten LCD and hold the LCD brightness.

Note. Before turn on the 3V VGG application, please turn on the Regulator（EN_VDDA=1）first. On the other hand, to turn on the double voltage of VDDP application, the Regulator can be turn off.

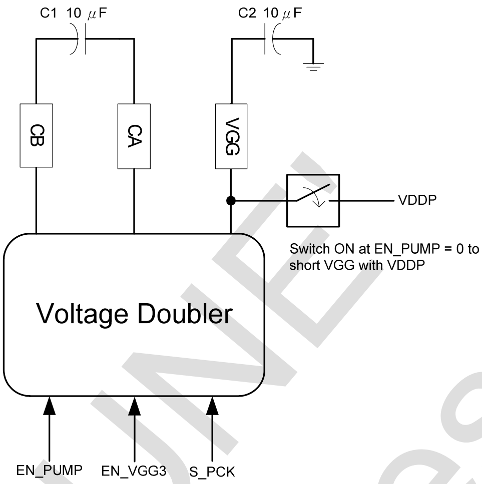


*Figure 12-1: **Voltage** Doubler*

When EN_PUMP = 1, voltage doubler is enabled. The VGG voltage is about two times of VDDP when EN_VGG3 = 0（default） or about 3V when EN_VGG3 = 1 and EN_VDDA = 1. When EN_PUMP = 0, you can input a voltage as voltage regulator power supply.

Voltage doubler operation frequency is selected by S_PCK. The details are described in “Voltage Doubler Operation Frequency” on page 28.

Typical value for C1 or C2 is 1μF~10μF. For large load current, larger capacitors should be used to reduce the output voltage ripple. If a polarity capacitor is used for C1, the CB pin should be connected to the negative terminal of the capacitor, and the CA pin to the positive terminal.

12.2 The **FS98O02A** and **FS98O02C** Voltage Regulator

The voltage regulator is used to regulate the doubled voltage of VDDP from VGG to analog power supply, VDDA. VDDA is the power supply voltage for analog circuit and LCD driver.

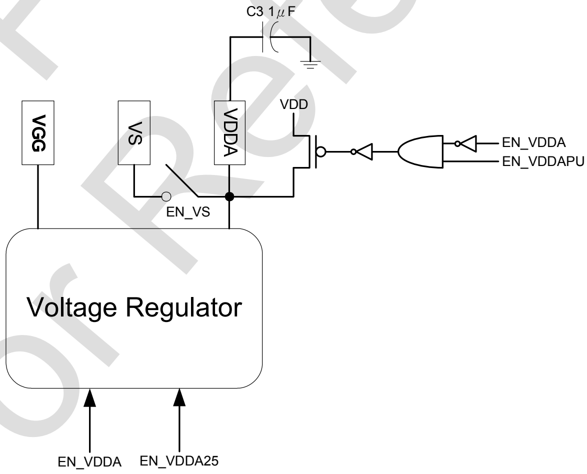


*Figure 12-2: The **FS98O02A** and **FS98O02C** Voltage Regulator*

When EN_VDDA = 1, the voltage regulator will be enabled and the VDDA is about 3.6V when EN_VDDA25 = 0（default）or about 2.5V when EN_VDDA25 = 1. Otherwise VDDA can be used as external regulated power supply input after turn off EN_VDDA and EN_VDDAPU. The default of the EN_VDDAPU is turned on to enable the pull-high circuit to pull high the VDDA to VDD when regulator is turned off. After turning on the regulator, the pull-high circuit will be turned off automatically until the regulator being turned off. If turn off the EN_VDDAPU, the VDDA pin will be floating when regulator is turned off.

The typical capacitance for C3 is 1μF~10μF. For large load current, large capacitor should be used to increase the output voltage stability.

Note. Except the VDDA pin is used for input, the EN_VDDAPU should be turn on（default）.

12.3 The **FS98O02B** Voltage Regulator

The voltage regulator is used to regulate the voltage of the VDD to analog power supply, VDDA. VDDA is the power supply voltage for analog circuit and LCD driver.

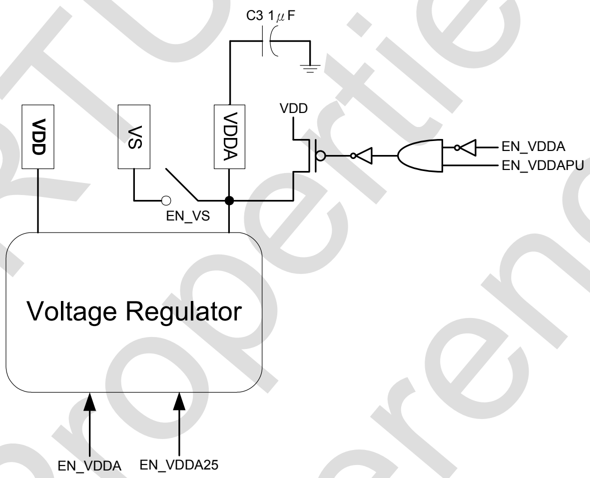


*Figure 12-3: The **FS98O02B** Voltage Regulator*

When EN_VDDA = 1, the voltage regulator will be enabled and the VDDA is about $VDD_{(max.\ 3.6V)}$ when EN_VDDA25 = 0（default）or about 2.5V when EN_VDDA25 = 1. Otherwise VDDA can be used as external regulated power supply input after turn off EN_VDDA and EN_VDDAPU. The default of the EN_VDDAPU is turned on to enable the pull-high circuit to pull high the VDDA to VDD when regulator is turned off. After turning on the regulator, the pull-high circuit will be turned off automatically until the regulator being turned off. If turn off the EN_VDDAPU, the VDDA pin will be floating when regulator is turned off.

The typical capacitance for C3 is 1μF~10μF. For large load current, large capacitor should be used to increase the output voltage stability.

Note. Except the VDDA pin is used for input, the EN_VDDAPU should be turn on（default）.

12.4 Analog Bias Circuit

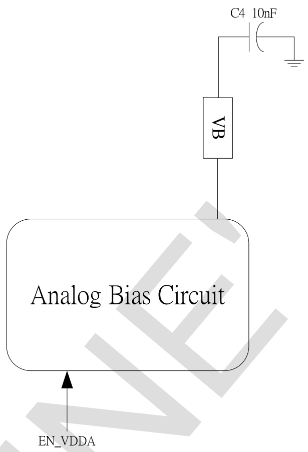


Figure 12-4: Analog Bias Circuit

Before enabling the analog block, EN_VDDA must be set. When the internal voltage doubler is used, a 10nF capacitor must be connected between pin VB and VSS for reducing voltage doubler's noise.

12.5 Analog Common Voltage Generator

Analog common voltage generator is used to generate the analog common voltage, AGND.

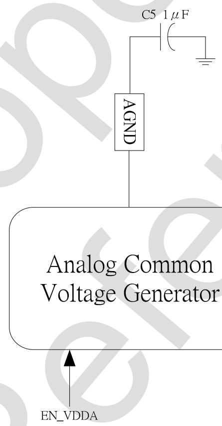


Figure 12-5: ***Analog*** *Common* ***Voltage Generator***

When EN_VDDA = 1, analog common voltage generator is enabled. AGND voltage is about 1/2 VDDA.

12.6 Low Battery Detector

Low battery detector is used for VDD low voltage detection. FS98O02 embeds a voltage divider which can generate 2.3V、2.4V、2.5V、2.6V and 2.7V. A multiplexer is used to connect the voltage divides to component input. The multiplexer's output is compares with 1.2V. The Control register flags are SILB[2:0] and the EN_LB.

The output flag is SVD which is for read only. Please see Figure 12-6

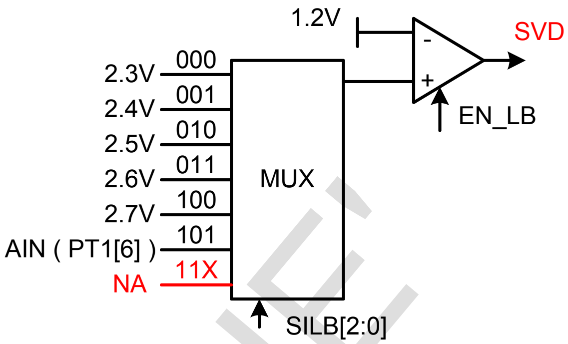


Figure 12-6: Low Battery Detector Block

Table 12-2: Low battery detector voltage detection selection table

SILB[2:0]	Detection Voltage	The SVD=1 Condition
000	VDD = 2.3V±100mV	VDD ≥ 2.2V
001	VDD = 2.4V±100mV	VDD ≥ 2.3V
010	VDD = 2.5V±100mV	VDD ≥ 2.4V
011	VDD = 2.6V±100mV	VDD ≥ 2.5V
100	VDD = 2.7V±100mV	VDD ≥ 2.6V
101	PT1.6 = 1.2V±100mV	PT1.6 ≥ 1.1V
11X	NA	NA

12.7 LCD Bias Circuit

V3, V2, V1 in Figure 12-7 are the output voltages of the LCD bias circuit. The voltages to VSS are about V_{RLCDin}, 2/3 * V_{RLCDin} and 1/3 * V_{RLCDin}, and the voltages are used in LCD driver.

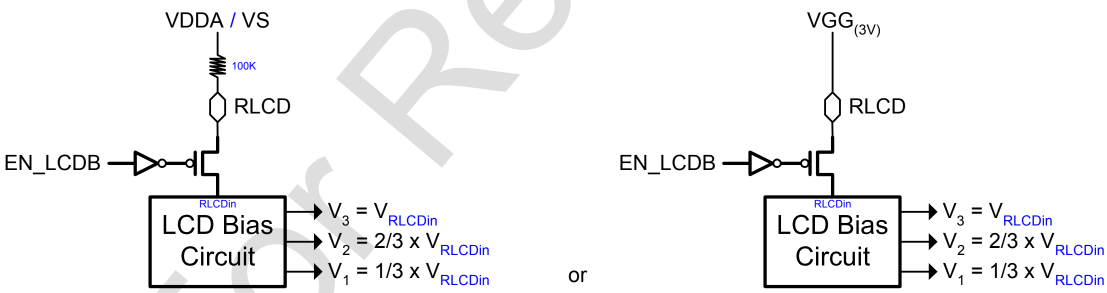


***Figure 12-7**: LCD Bias Circuit*

When EN_LCDB = 1, the LCD bias circuit is enabled. When EN_LCDB = 0, the LCD bias circuit is disabled, and then the LCD driver will not function.

13. Clock System

The clock system offers several clocks to some important blocks in FS98O02, such as CPU clock, ADC sample frequency, beeper clock, voltage doubler operating frequency, etc. Only with the clock signals from the clock system, the FS98O02 can work normally.

Table 13-1: Clock System Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
09h	PCK					S_CHCK[1:0]		S_BEEP	EN_OSC4M	---- 0000	---- 0000

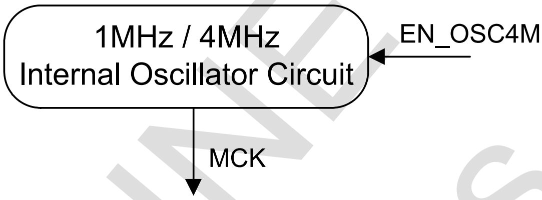


Figure 13-1: Internal Clock for CPU and Other Blocks

13.1 Oscillator State

MCK is the heart of the clock system. Almost all clock signals are derived from the MCK. If we stop MCK, many clock signals will be stopped. We may use “sleep” instruction to disable MCK as described in Table 13-2.

When EN_OSC4M = 0(default), the MCK is about 1MHz. When EN_OSC4M = 1, the MCK is about 4MHz.

*Table 13-2: **How to Use MCK***

Sleep	MCK
1	Disable
0	Enable

13.2 CPU Instruction Cycle

The CPU in FS98O02 has only one mode of instruction cycle. That is MCK/4 (~4us for 1MHz MCK or ~1us for 4MHz MCK).

13.3 ADC Sample Frequency

ADC sample frequency decides the sampling rate of the input signal. The sampling rate of ADC in FS98O02 is fixed to MCK/25.

*Table 13-3: ADC **Sampling** Frequency*

ADC Sample Frequency (ADCF)
MCK/25 (~**40kHz** for 1MHz MCK or ~**160kHz** for 4MHz MCK)

13.4 Beeper Clock

We may set beeper clock by select the proper value of S_BEEP as in Table 13-4.

*Table 13-4: **Setting Beeper Clock***

S_BEEP	Beeper Clock
1	**MCK/500** for 1MHz MCK or **MCK/2000** for 4MHz MCK (~2kHz)
0	**MCK/312** for 1MHz MCK or **MCK/1248** for 4MHz MCK (~3.2kHz)

13.5 Voltage Doubler Operation Frequency

The voltage doubler operation frequency is related to the load capability. There is an internal S_PCK signal deciding the voltage doubler operating frequency.

*Table 13-5: **Voltage Doubler Operation Frequency***

S_PCK	Voltage Doubler Operation Frequency
1	**MCK/50** for 1MHz MCK or **MCK/200** for 4MHz MCK (~20kHz)

13.6 Timer and LCD Module Input Clock

The timer and LCD module input clock in **FS98O02A** and **FS98O02B** is MCK/2000 for 1MHz MCK or MCK/8000 for 4MHz MCK. The timer and LCD module input clock in **FS98O02C** is MCK/25 for 1MHz or 4MHz MCK as described in Table 13-6.

*Table 13-6: **Timer and LCD Module Input Clock***

Timer and LCD Module Input Clock TMCLK
MCK/2000 for 1MHz MCK or **MCK/8000** for 4MHz MCK (~500Hz) for **FS98O02A** and **FS98O02B**
MCK/25 for 1MHz or 4MHz MCK for **FS98O02C**

13.7 OPAM Chopper Input Clock

The OPAM chopper input clock in FS98O02 is selected by internal S_CH1CK as described in Table 13-7.

*Table 13-7: **OPAMP chopper Input Clock***

S_CH1CK[1:0]	OPAMP chopper mode (input operation)
00	+Offset
01	-Offset
10	**MCK/500** for 1MHz MCK or **MCK/2000** for 4MHz MCK chopper frequency
11	**MCK/1000** for 1MHz MCK or **MCK/4000** for 4MHz MCK chopper frequency

14. I/O Port

We may set the I/O port to be input port or output port in FS98O02 for our applications. We may also set PT1 [7] as buzzer output to drive the external buzzer to generate sounds. The buzzer's beeper frequency is show in Table 13-4.

Table 14-1: I/O Port Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
06h	INTF								E0IF	---0 --00	---0 --00
07h	INTE	GIE							E0IE	0--0 --00	0--0 --00
20h	PT1	PT1 [7:0]								uuuu uuuu	uuuu uuuu
21h	PT1EN	PT1EN [7:4]								0000 ----	0000 ----
22h	PT1PU	PT1PU [7:0]								0000 0000	0000 0000
23h	PT1MR	BPE						E0M [1:0]		0--- --00	0--- --00
24h	PT2					PT2 [3:0]				---- uuuu	---- uuuu

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
25h	PT2EN					PT2EN [3:0]				---- 0000	---- 0000
26h	PT2PU					PT2PU [3:0]				---- 0000	---- 0000

14.1 PT1

PT1 is 4-bit Input port and 4-bit I/O port with pull-up resistor enable control.

PT1 [N] is an input port when PT1EN [N] = “0”; PT1 [N] is an output port when PT1EN [N] = “1”. When PT1EN [7] = “1” and BPE = “1”, PT1 [7] is used as the buzzer output.

When PT1PU [N] = “0”, PT1 [N] has no pull-up resistor; When PT1PU [N] = “1”, PT1 [N] has a pull-up resistor.

PT1 [0] can be used as an external interrupt source. Interrupt mode of PT1 [0] is controlled by E0M [1:0]. When E0M [1:0] = “00”, PT1 [0] is for negative edge trigger interrupt input; and “01” for positive edge. “10” & “11” for interrupt during other time.

If VPP = 12V , PT1[1] use for judge if SPI or instruction programmer EPROM, if PT1[1] connect to VSS then used instruction programmer EPROM function. If PT1[1] connect to VDD then used SPI programmer EPROM.

If VPP = 0 ~3.6V , PT1[1] is a input port.

PT1 [N] has Schmitt-trigger input.

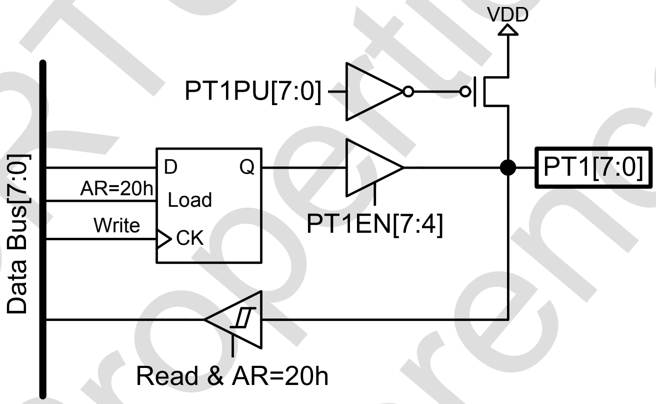


Figure 14-1: the Block Diagram of Port 1

14.2 PT2

PT2 is 4-bit I/O port with pull-up resistor enable control. PT2 [N] is an input port when PT2EN [N] = “0”; PT2 [N] is an output port when PT2EN [N] = “1”. When PT2PU [N] = “0”, PT2 [N] has no pull-up resistor; When PT2PU [N] = “1”, PT2 [N] has a pull-up resistor. PT2 [N] has Schmitt-trigger input.

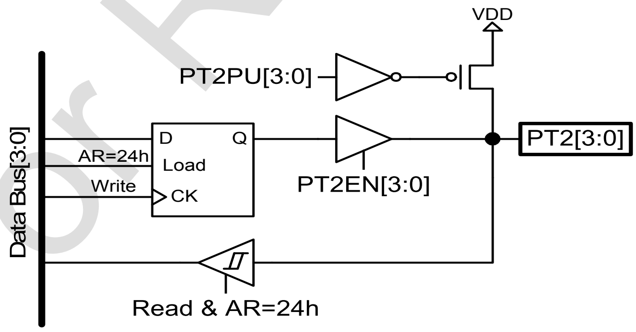


Figure 14-2: the Block Diagram of Port 2

15. 8-bit Timer

The 8-bit timer in FS98O02 is usually used for timer interrupt in applications. We may have a periodic internal interrupt to do some periodic procedure in CPU by using the 8-bit timer properly.

Table 15-1: 8-bit Timer Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
06h	INTF				TMIF					---0 --00	---0 --00
07h	INTE	GIE			TMIE					0--0 --00	0--0 --00
0Dh	TMOUT	TMOUT [7:0]								0000 0000	0000 0000
0Eh	TMCON	TMRST				TMEN	INS [2:0]			1000 0000	1u00 0000
0Fh	TMMOD	TMMOD [7:0]								0000 0000	0000 0000

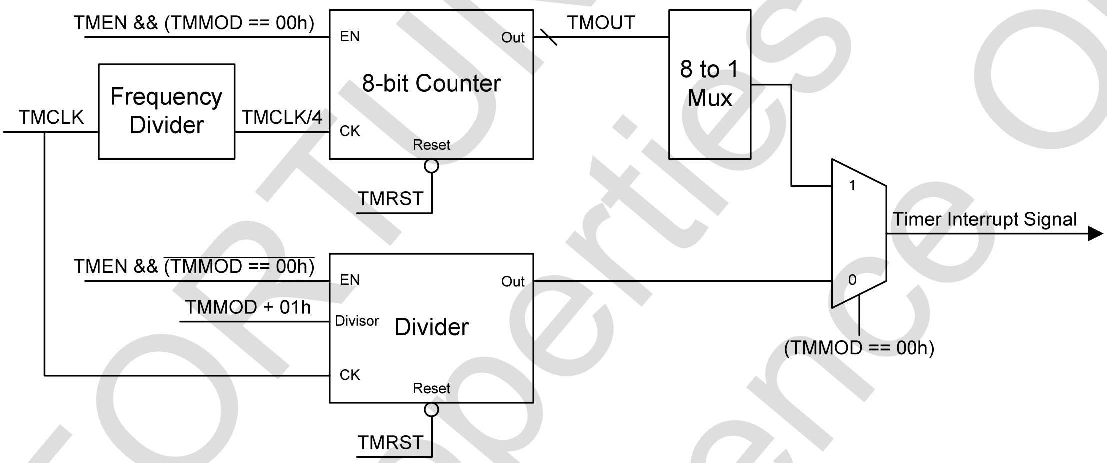


Figure 15-1: The 8-bit Timer Block Diagram

After writing a “0” to bit 7 of address 0Eh, the CPU will send a low pulse to TMRST and reset the 8-bit counter and TMCLK divider. And then the bit 7 of address 0EH will be set to “1”.

When TMEN = 1, the register value of TMMOD is zero or not that controls which one of the 8-bit counter or TMCLK divider is enabled. When TMMOD = 8’h00, the 8-bit counter is enabled and the TMCLK divider is disabled. Otherwise, if the TMMOD has been set not zero value, the TMCLK divider is enabled and 8-bit counter is disabled. When TMEN = 0, the 8-bit counter and TMCLK divider are all stopped.

INS [2:0] and TMMOD select timer interrupt source. When TMMOD = 8’h00, the selection codes are described in Table 15-3. Otherwise the selection codes are described in Table 15-4.

Table 15-2: The Timer Input Clock TMCLK

Product	The TMCLK for Int. OSC=1MHz	The TMCLK for Int. OSC=4MHz
FS98O02A / FS98O02B	500Hz	500Hz
FS98O02C	40KHz	160KHz

Table 15-3: Setting Timer Interrupt Source by INS

INS [2:0]	Timer Interrupt Source for FS98O02A / B	Timer Interrupt Source for FS98O02C
000	TMOUT [0] （TMCLK/8：~62.5Hz）	TMOUT [0] （TMCLK/8：5KHz or 20KHz）
001	TMOUT [1] （TMCLK/16：~31.25Hz）	TMOUT [1] （TMCLK/16：2.5KHz or 10KHz）
010	TMOUT [2] （TMCLK/32：~15.625Hz）	TMOUT [2] （TMCLK/32：1.25KHz or 5KHz）

011	TMOUT [3] （TMCLK/64：~7.813Hz）	TMOUT [3] （TMCLK/64：625Hz or 2.5KHz）
100	TMOUT [4] （TMCLK/128：~3.91Hz）	TMOUT [4] （TMCLK/128：312.5Hz or 1.25KHz）
101	TMOUT [5] （TMCLK/256：~1.953Hz）	TMOUT [5] （TMCLK/256：156.25Hz or 625Hz）
110	TMOUT [6] （TMCLK/512：~0.977Hz）	TMOUT [6] （TMCLK/512：78.125Hz or 312.5Hz）
111	TMOUT [7] （TMCLK/1024：~0.488Hz）	TMOUT [7] （TMCLK/1024：39.0625Hz or 156.25Hz）

Table 15-4: Setting Timer Interrupt Source by TMMOD

TMMOD（0x0F）	Timer Interrupt Source
8’h00	The Timer Interrupt Source is selected by INS[2:0]
8’h01	TMCLK / 2
8’h02	TMCLK / 3
8’h03	TMCLK / 4
⋮	⋮
8’hFE	TMCLK / 255
8’hFF	TMCLK / 256

When TMMOD = 8’h00 and INS[2:0] = 3’b000, the timer interrupt function is equal to TMMOD = 8’h07 because of the timer interrupt sources are TMCLK/8. In the same manner, the timer interrupt source of the TMMOD = 8’h00 and INS[2:0] = 3’b001 is equal to TMMOD = 8’h0F. And the timer interrupt source of the TMMOD = 8’h00 and INS[2:0] = 3’b101 is equal to TMMOD = 8’hFF, etc.

TMOUT [7:0] is the output of the 8-bit counter. It is a read-only register.

16. ADC

The ADC in the IC is a ΔΣ ADC with fully differential inputs and fully differential reference voltage inputs. Its maximum digital output code is ±15625.

The conversion equation is: Dout = 15625 * G * (VIH – VIL + Vio) / (VRH – VRL + Vro). VIH is the ADC positive input voltage. VIL is the ADC negative input voltage. Vio is the ADC input offset voltage. VRH is the ADC positive reference voltage. VRL is the ADC negative reference voltage. Vro is the ADC reference offset voltage. And (VRH – VRL + Vro) > 0. When G * (VIH – VIL + Vio) / (VRH – VRL + Vro) ≧ 1, Dout = 15625. When G * (VIH – VIL + Vio) / (VRH – VRL + Vro) ≦ -1, Dout = -15625.

16.1 ADC Digital Output Code Format

ADO [23:8] is the ADC digital output code. The digital output code is in 2’s complement format, and the ADO [23], the most significant bit (MSB) of ADO represents the sign of the code. For example, if ADO [23:8] = E2F7h, then Dout = -(not(E2F7h) + 1) = -7433.

16.2 ADC Linear Range

The ΔΣ ADC is close to saturation state when G * (VIH – VIL + Vio) / (VRH – VRL + Vro) is close to ±1. The ADC has good linearity when G * (VIH – VIL + Vio) / (VRH – VRL + Vro) is within ± 0.95.

16.3 ADC Control Register

There are some ADC control related registers in FS98O02. There will be some more detail descriptions about using the ADC control register to get the proper ADC operation in users' applications.

Table 16-1: ADC** Control **Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
06h	INTF							ADIF		---0 --00	---0 --00
07h	INTE	GIE						ADIE		0--0 --00	0--0 --00
10h	ADOH	ADO [23:16]								0000 0000	0000 0000
11h	ADOM	ADO [15:8]								0000 0000	0000 0000
12h	ADOL	ADO [7:0]								0000 0000	0000 0000
13h	ADCON					ADRST	ADM [2:0]			---- 0000	---- 0000
2Fh	NETG					ADG [1:0]		ADEN	AZ	---- 0000	---- 0000

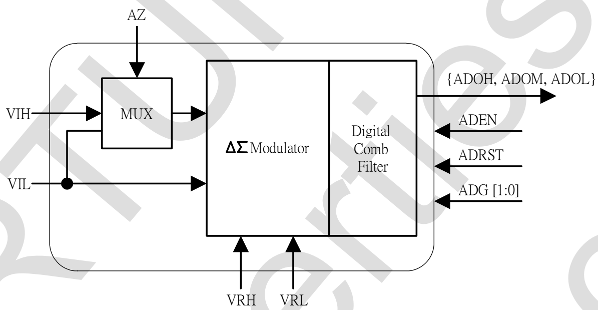


Figure 16-1: the Block Diagram of ADC

The ADC contains a ΔΣ modulator and a digital comb filter. When ADEN = 1, the ΔΣ modulator (ADC) will be enabled. When ADEN = 0, the ΔΣ modulator (ADC) will be disabled. When ADRST = 1, the digital comb filter will be enabled. When ADRST = 0, the digital comb filter will be reset.

16.3.1 ADC Output Rate and Settling Time

The ΔΣ ADC is generally an over-sampling ADC. The ADC's each digital output code is the result of sampling the input signal N times and processed by DSP. The ADC's sampling frequency is decided by ADCF.

ADM decides when to send out a 16-bit digital code after sampling N times, and raises an interrupt signal every time the ADC produces a digital output code. In fact, the ADC's each digital output code is the result from the previous 2*N times sampling results. If any of the ADC's input, reference voltage, ADG or AZ is switched, the first two output codes are normally not stable, and the third output code and later codes are stable for calculation.

The ADC's output rate is selected by ADM [2:0] as described in Table 16-2.

Table 16-2: ADC Output Rate

ADM [2:0]	ADC Output Rate
000	ADCF/125 (~320Hz for 1MHz MCK or ~1.28kHz for 4MHz MCK)
001	ADCF/250 (~160Hz for 1MHz MCK or ~640Hz for 4MHz MCK)

010	ADCF/500 (~80Hz for 1MHz MCK or ~320Hz for 4MHz MCK)
011	ADCF/1000 (~40Hz for 1MHz MCK or ~160Hz for 4MHz MCK)
100	ADCF/2000 (~20Hz for 1MHz MCK or ~80Hz for 4MHz MCK)
101	ADCF/4000 (~10Hz for 1MHz MCK or ~40Hz for 4MHz MCK)
110	ADCF/8000 (~5Hz for 1MHz MCK or ~20Hz for 4MHz MCK)

16.3.2 ADC Input Offset

The ADC input offset voltage, Vio drifts with temperature and common mode voltage at the inputs. When AZ = 0, the ΔΣ modulator's differential inputs are (VIH, VIL); when AZ = 1, the ΔΣ modulator's differential inputs are (VIL, VIL). We can set AZ = 1 to measure the ADC's offset.

When the drifting is slow, we may set AZ = 1, and get Doff = 15625 * G * (Vio) / (VRH – VRL + Vro). When measuring input signal, Doff should be deducted.

16.3.3 ADC Gain

The ADC digital output code deducted by Doff is the ADC Gain. The ADC gain does not change as VDD changes. The suggested values for common mode voltages at ADC input and reference voltage are 1V~2V respect to VSS. ADG [1:0] can set ADC's input gain: 00 for 2/3, 01 for 1, 10 for 2, and 11 for 7/3.

16.3.4 ADC Resolution

The ADC resolution is mainly decided by ADM [2:0] (ADC output rate) and reference voltage, and the test results in FSC are as below for users' reference. When we set (VRH, VRL) = 0.4V, (VIH, VIL) = 0.2V, VRL = VIL = AGND, G = 1, and record ADO [23:8], we get the result in Table 16-3

. When we set (VRH, VRL) = VR, (VIH, VIL) = 1/2 * VR, VRL = VIL = AGND, G = 1, ADM [2:0] = 101, and record ADO [15:0], we get the result in Table 16-4.

Table 16-3

ADM [2:0]	000	001	010	011	100	101	110
Rolling counts	10	6	4	3	3	2	1

Table 16-4

VR (V)	0.05	0.1	0.2	0.3	0.4	0.6	0.8	1.0
Rolling counts	31	15	5	3	2	2	4	9

16.4 ADC Input Multiplexer and Low Pass Filter

There are several analog input multiplexers and a low pass filter in FS98O02. We may use the analog input multiplexers and the low pass filter properly to measure the input signals well.

Table 16-5: ADC Input Multiplexer and Low Pass Filter Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
2Ah	NETB		SINL [1:0]		SINH [1:0]		SFT [2]		SFT [0]	---0 00-0	---0 00-0
2Ch	NETD		SVRH [0]	SVRL [1:0]				SLVD	SVR	u000 uu00	u000 uu00

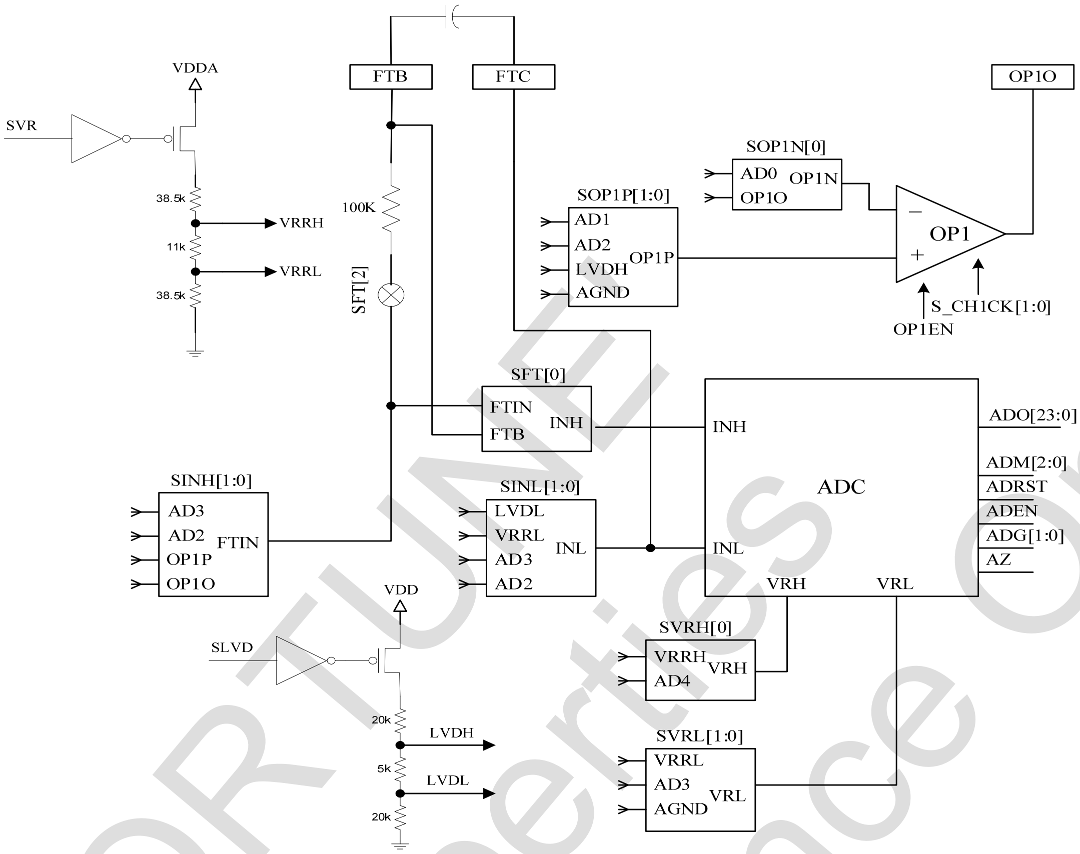


Figure 16-2: The FS98O02A / B ADC Input** Multiplexer **and Low Pass Filter

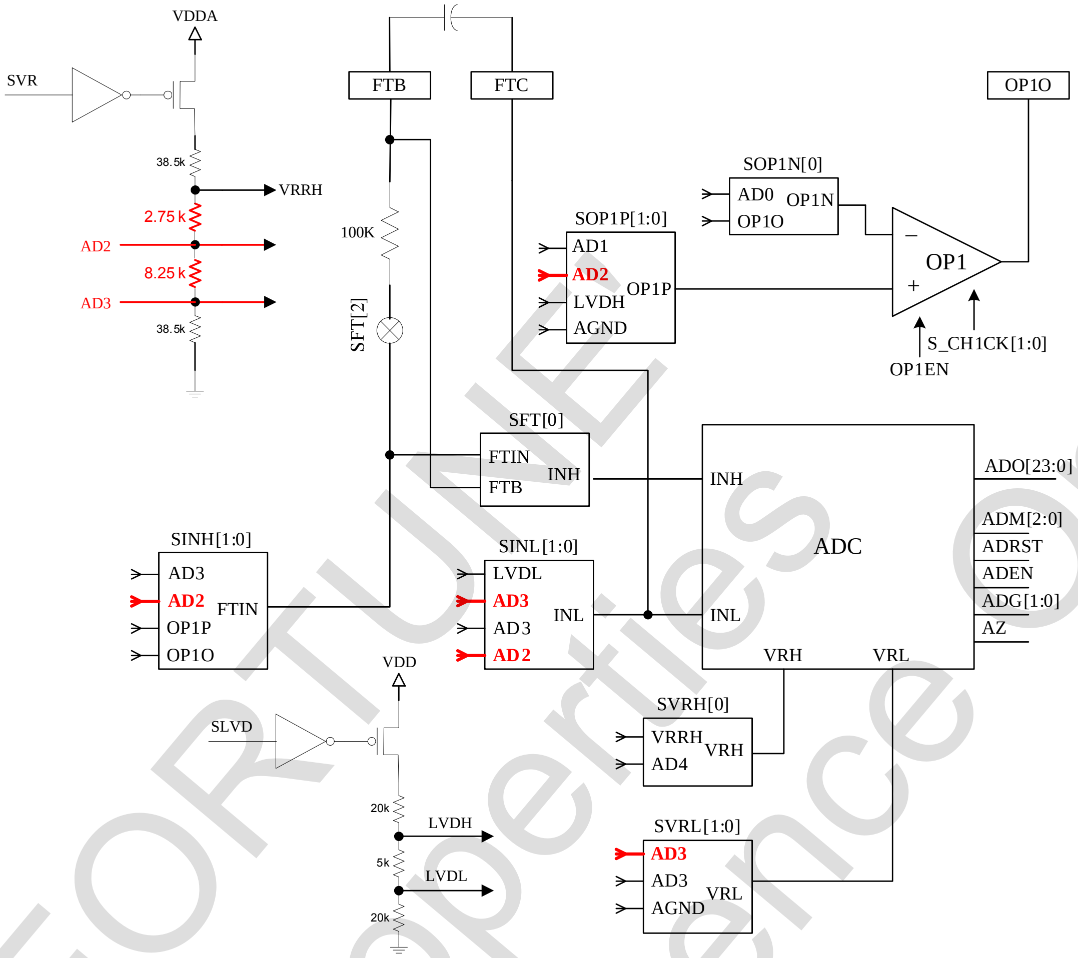


Figure 16-3: The FS98O02C ADC Input** Multiplexer **and Low Pass Filter

ADC Operation

2. Get the VGG (2 times VDDP / 3V or external Power Supply).
3. Get the VDDA (3.6V / 2.5V)
4. Enable the Analog Bias Circuit
5. Set SINH[2:0] and SFTA[2:0] to decide the ADC positive input port signal.(Table 16-6, Table 16-7 and Table 16-8)

Table 16-6: FTIN Selection table

SINH[1:0]	FTIN
00	OP1O
01	OP1P
10	AD2
11	AD3

Table 16-7: FTB selection table

SFT[2]	FTB[1]
0	ADC Low Pass Filter is disabled
1	ADC Low Pass Filter is enabled

Table 16-8: INH selection table

SFT[0]	INH (ADC positive input port signal)
0	FTB
1	FTIN

6. Set SINL[1:0] to decide the ADC negative input port signal. (Table 16-9)

Table 16-9: INL selection table

SINL[1:0]	INL for FS98O02A / B	INL for FS98O02C
00	AD2	AD2
01	AD3	AD3
10	VRRL	AD3
11	LVDL	LVDL

7. Set ADG[1:0] to decide the ADC input gain. (Table 16-10)

Table 16-10: ADG selection table

ADG[1:0]	ADC input gain
00	2/3
01	1
10	2
11	7/3

8. Set SVRH[1:0] to decide the ADC reference voltage positive input port signal. (Table 16-11)

Table 16-11: VRH selection table

SVRH[1:0]	VRH (ADC reference voltage positive input)
0	AD4
1	VRRH

9. Set SVRL[1:0] to decide the ADC reference voltage negative input port signal. (Table 16-12)

Table 16-12: VRL selection table

SVRL[1:0]	VRL for FS98O02A / B	VRL for FS98O02C
00	AGND	AGND
01	AD3	AD3
10	VRRL	AD3
11	VRRL	AD3

10. Set ADIE and GIE register flags to enable the ADC interrupt
11. Set ADEN register flag, the embedded Σ-Δ modulator will be enabled.
12. Set ADRST register flag, the comb filter will be enabled.
13. When the ADC interrupt happen, read the ADO[23:8] to get the ADC output.(ADO[23:22] are signed bits)

[1] *The input of ADC Low Pass Filter is FTIN, and the output is FTB*

14. Set AZ register flag to make the ADC positive and negative input port be internally short. Read the ADO[23:8] to get the ADC offset (The ADO should be zero if the offset is zero). Clear AZ register flag to make the ADC work normally.

16.5 OPAMP : OP1

Table 16-13: FS98O02 OPAMP register table

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
09h	PCK					S_CHCK[1:0]				---- 0000	---- 0000
33h	NETK					OP1EN	SOP1P[1:0]		SOP1N[0]	0000 0000	0000 0000

OPAMP Operation

1. Set SOP1P[1:0] to decide the OPAMP non-inverting input port signal. (Table 16-14)

Table 16-14: OP1P selection table

SOP1P[1:0]	OP1P (OPAMP non-inverting input)
00	AGND
01	LVDH
10	AD2
11	AD1

2. Set SOP1N[0] to decide the OPAMP inverting input port signal. (Table 16-15)

Table 16-15: OP1N selection table

SOP1N[0]	OP1N (OPAMP inverting input)
0	OP1O
1	AD0

3. Set S_CHCK[1:0] to decide the OPAMP chopper mode.

Table 16-16: chopper mode selection table

S_CHCK[1:0]	OPAMP chopper mode (input operation)
00	+Offset
01	-Offset
10	MCK/500 chopper frequency
11	MCK/1000 chopper frequency

4. Set OP1EN to enable the OPAMP.

16.5.1 ADC Pre-filter

The input signal is sampled by the ADC. When the input signal has a noise with frequency higher than the sampling frequency, the noise generates low frequency noises through sampling circuit. Hence it is recommended to pass the input signal through a low pass filter, in order to get a stable ADC output.

Inside the chip, there is an internal 100k ohm resistor, which is for the construction of a low pass filter through parallel connection with an external capacitor between two pins FTB and FTC. The capacitance is normally between 10nF and 50nF. Please note that a larger capacitance may cause too much delay time in input signal switching. SFT [2] decides if an input signal passes the low pass filter. SFT [0] decides whether the ADC's input is the signal through a low pass filter.

16.5.2 ADC Input Multiplexers

The positive and negative input signal terminals of the ADC and reference voltages of the ADC can be selected by analog multiplexers and set to status specified by users. The input signal terminals of the ADC can be selected by SIN [0] and SFT [2]. The reference voltages of the ADC can be selected by SVR.

17. Instruction Programmer EPROM

"TBLP" is an instruction to programmer EPROM, "MOVP" is an instruction to look up table from EPROM. In this function, PT1[1] must connect to VSS, and VPP must be 12V.

Table 17-1 FS98O02 Programmer EPROM register table

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
2Ch	NETD	EPMAT				ERV	EPBLK			u000 uu00	u000 uu00
0Ah	EADRH	PAR [15:8]								uuuu uuuu	uuuu uuuu
0Bh	EADRL	PAR [7:0]								uuuu uuuu	uuuu uuuu
0Ch	EDAH	EDATA [15:8]								uuuu uuuu	uuuu uuuu

NETD[3] : ERV is a status flag of programming voltage detection. When VPP = 12V, ERV = 1. Otherwise ERV = 0.
NETD[2] : EPBLK is a status flag of blank data checking generated by table look-up instruction MOVP. When data in EPROM address {EADRH, EADRL} equals to FFFFh, ELBLK=1. Otherwise ELBLK=0
NETD[7] : EPMAT is a EPROM data validation flag generated by instruction MOVP. When EPROM data in address {EADRH, EADRL} equals to the data in register {EDAH, WORK}, ELMAT=1. Otherwise ELMAT=0

EADRH[7:0] : Programmer Address MSB.
EADRL[7:0] : Programmer Address LSB.
EDAH[7:0] : Programmer Data MSB.
WORK[7:0] : Programmer Data LSB.

Note. 1. Please keep VDD = 3.2V ± 0.2V when executing Instruction Programmer EPROM.
2. Charge Pump must be on and delayed about 100ms for VGG=2xVDDP before the VPP connect to 12V and executing Instruction Programmer EPROM.

```
Initially::
        BSF     NETF , ENPUMP       ; Open Charge PUMP , For VDDA , VS , and Instruction Programmer EPROM
        CALL    delay100ms
        BSF     NETF , ENVDDA       ; Open VDDA , If VS has heavy Loading then delay about 200mS
        CALL    delay200ms
        BSF     NETF,ENVS           ; Open VS
          ……………………..
          ……………………..
          ……………………..

TBProgram:
            BTFSS   NETD,ERV            ; Check VPP = 12 V
            GOTO    ERR_VPP
            MOVLW   HIGH TABLE          ; Programmer EPROM MSB Address
            MOVWF   EADRH
            MOVFW   LOW TABLE           ; Programmer EPROM LSB Address
            MOVWF   EADRL
            MOVP    0                   ; Read EPROM Data , not Increment Address
            BTFSS   NETD,EPBLK          ; Check EPROM if 0xFFFF
            GOTO    ERR_EMPTY
            MOVFW   ADCO+1              ; Programmer   MSB Data
            MOVWF   EDAH
            MOVFW   ADCO                ; Programmer   LSB Data
            TBLP    64                  ; Programmer EPROM , 64   is a const value
            NOP
            MOVP    1                   ; Read EPROM Data and Increment Address ,[EADRH , EADRL]+1
            BTFSS   NETD,EPMAT          ; Check if Programmer Match
            GOTO    ERR_MATCH
            ………….                       ; Next step
ERR_EMPTY:
            ………….
ERR_VPP:
            ………….
ERR_MATCH:
            ………….
```

Example 17-1: *Programmer* ***EPROM Example***

18. LCD Driver

The LCD driver in FS98O02 has 4 commons and 12 segments, and the LCD can drive 4 x 12, 48 dots LCD.

The LCD common driver waveform is show in Figure 18-1.

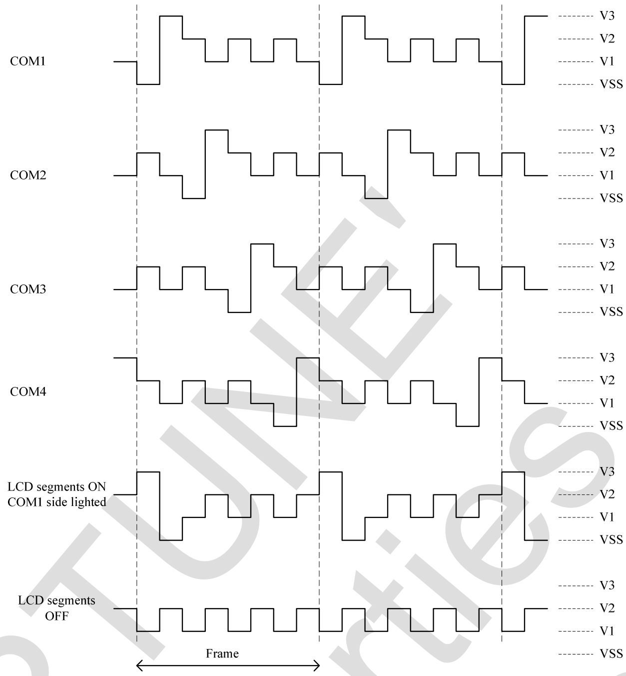


Figure 18-1: LCD** Common **Driver Waveform

Table 18-1: LCD Control Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
40h	LCD1	SEG1 [3:0]				SEG0 [3:0]				uuuu uuuu	uuuu uuuu
41h	LCD2	SEG3 [3:0]				SEG2 [3:0]				uuuu uuuu	uuuu uuuu
42h	LCD3	SEG5 [3:0]				SEG4 [3:0]				uuuu uuuu	uuuu uuuu
43h	LCD4	SEG7 [3:0]				SEG6 [3:0]				uuuu uuuu	uuuu uuuu
44h	LCD5	SEG9 [3:0]				SEG8 [3:0]				uuuu uuuu	uuuu uuuu
45h	LCD6	SEG11 [3:0]				SEG10 [3:0]				uuuu uuuu	uuuu uuuu
2Eh	NETF			EN_LCDB	LCDEN					0000 0100	0000 0100

- LCD1~LCD6 is the LCD display data area.
- LCDEN =1 starts the LCD clock and then we may use the LCD. If LCDEN = 0, the LCD driver will be disabled.

When EN_LCDB = 1, the LCD bias circuit is enabled. When EN_LCDB = 0, the LCD bias circuit is disabled, and the LCD driver will not function.

The LCD frame frequency is selected by internal LCDCKS [1:0] as described in Table 18-2.

Table 18-2: Setting LCD Frame Frequency

LCDCKS [1:0]	LCD Frame Frequency
01	LCD Input Frequency/16 (~31.25Hz)

Here we demonstrate the way to light on the dots on the LCD. If we set LCD1 = “0110-1101b”, it means we set SEG1 [3:0] = “0110b” and SEG0 [3:0] = “1101b”. Then, the dots on the cross position of SEG1 and COM3, COM2 will be on; also, the dots on the cross position of SEG0 and COM4, COM3, COM1 will be on. We may use the same method to light on the dots on the cross position of SEG0 to SEG7 and COM4 to COM1.

19. CPU Reset

The FS98O02 CPU has three reset signals and they are external reset (RST_), low voltage reset (LVR), and watchdog time out reset. When resetting, the CPU's program counter (PC) is reset to 0. After reset finished, the CPU starts working. Table 19-1 shows the CPU's internal registers status after reset.

Table 19-1: the CPU's Internal Registers Status after Reset

Address	Name	Reset State	WDT Reset State
00h	IND0	uuuu uuuu	uuuu uuuu
02h	FSR0	uuuu uuuu	uuuu uuuu
04h	STATUS	---0 0uuu	---u 1uuu
05h	WORK	uuuu uuuu	uuuu uuuu
06h	INTF	---0 –00	---0 --00
07h	INTE	0--0 –00	0--0 --00
09h	PCK	---- 0000	---- 0000
0Ah	EADRH	uuuu uuuu	uuuu uuuu
0Bh	EADRL	uuuu uuuu	uuuu uuuu
0Ch	EDAH	uuuu uuuu	uuuu uuuu
0Dh	TMOUT	0000 0000	0000 0000
0Eh	TMCON	1000 0000	1u00 0000
0Fh	TMMOD	0000 0000	0000 0000
10h	ADOH	0000 0000	0000 0000
11h	ADOM	0000 0000	0000 0000
12h	ADOL	0000 0000	0000 0000
13h	ADCON	---- 0000	---- 0000
1Fh	SVD	---- ---1	---- ---1
20h	PT1	uuuu uuuu	uuuu uuuu
21h	PT1EN	0000 ----	0000 ----
22h	PT1PU	0000 0000	0000 0000
23h	PT1MR	00-- --00	00-- --00
24h	PT2	---- uuuu	---- uuuu
25h	PT2EN	---- 0000	---- 0000
26h	PT2PU	---- 0000	---- 0000
2Ah	NETB	-000 00-0	-000 00-0
2Ch	NETD	u000 uu00	u000 uu00
2Eh	NETF	0000 0100	0000 0100
2Fh	NETG	---- 0000	---- 0000
33h	NETK	0000 0000	0000 0000
40h	LCD1	uuuu uuuu	uuuu uuuu
41h	LCD2	uuuu uuuu	uuuu uuuu
42h	LCD3	uuuu uuuu	uuuu uuuu
43h	LCD4	uuuu uuuu	uuuu uuuu
44h	LCD5	uuuu uuuu	uuuu uuuu
45h	LCD6	uuuu uuuu	uuuu uuuu

Note 1: "u" means unknown or unchanged. "-" means unimplemented, read as "0".
Note 2: The "Reset State" indicates the registers state after external reset and low voltage reset.
Note 3: The "WDT Reset State" indicates the registers state after watchdog time out reset.

19.1 External Reset

The CPU has a "VPP/RST/TST" pin for external reset usage. When "VPP/RST/TST" is in logic "low" state (about 0 ~ 0.3V), the CPU will go into external reset status. The external R/C circuit for reset is shown as following. When VDD changes from “low” to “high”, the CPU external reset status will be released, and the CPU will be in normal operating condition.

The signal from the “VPP/RST/TST” pin to CPU should remain in logic “low” state for more than 2μs to reset the CPU. If the signal from the “VPP/RST/TST” pin to CPU is in “low” state less than 2μs, the CPU will not be reset. Figure 19-2 shows the minimum reset period to reset the CPU.

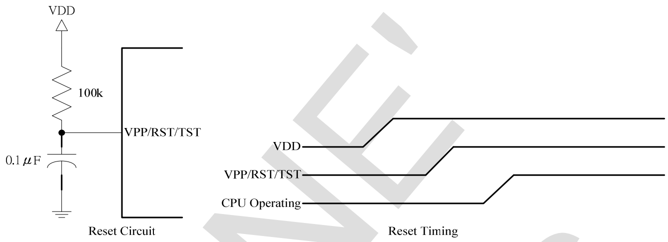


Figure 19-1: the Reset** Circuit **and the Reset Timing

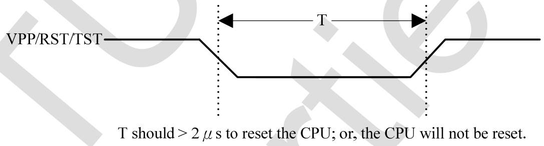


Figure 19-2: the Minimum** Reset **Period to Reset the CPU

19.2 Low Voltage Reset

To avoiding the CPU in an abnormal power status that makes the CPU unable to reset and causes the CPU operating abnormally, there’s a low voltage reset circuit embedded in FS98O02. When the voltage of VDD is less than LVR threshold low voltage, the CPU enters reset state; and when the voltage of VDD comes back above the LVR threshold high voltage, the CPU will be in normal operating condition.

19.3 Watchdog Time Out Reset

The watchdog timer in FS98O02 is usually used to monitor if the CPU is in normal operation. If the CPU is not in normal operation, we may use the watchdog timer to raise a watchdog time our reset and make the CPU to be back in normal operation. The watchdog timer may be used to some period wakeup-and-measuring applications to save the power for some long time monitoring portable applications.

Table 19-2: Watchdog Time Out Reset Related Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	WDT Reset State
04h	STATUS					TO				---0 0uuu	---u 1uuu
0Eh	TMCON		WDTEN	WTS [1:0]						1000 0000	1u00 0000

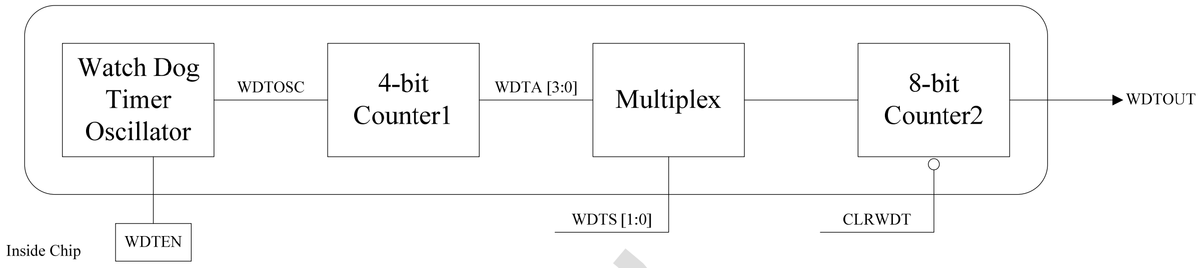


Figure 19-3: the Block Diagram of Watchdog Timer

When starting watch dog timer (WDT), it is necessary to set WDTEN bit. Once WDTEN be set, that can't be clear by any program instruction except hardware reset (RST PIN Set to low). When the CPU executes CLRWDT instruction, the WDT counter may be reset. The clock input of the WDT comes from an internal independent R/C oscillator. The WDT output can be selected by WTS, as shown in the following table. When the WDT outputs the logic "high", the CPU will enter reset status and set TO bit to 1.

Table 19-3: Setting the Frequency of WDTOUT

Typical Frequency of WDTOSC is about 1kHz.		
WTS [1:0]	**Frequency of WDTOUT**	**Typical Frequency of WDTOUT**
00	FWDTOSC / 4096	0.244Hz
01	FWDTOSC / 2048	0.488Hz
10	FWDTOSC / 1024	0.977Hz
11	FWDTOSC / 512	1.953Hz

20. Halt and Sleep Mode

20.1 Halt Mode

After the CPU executes a HALT instruction, the CPU program counter (PC) stops counting until the CPU receives an internal or external interrupt signal. To avoid program errors caused by Interrupt return, it is necessary to add a NOP instruction after the HALT instruction to guarantee the program's normal execution as described in Example20-1.

```
HALT
NOP
```

Example20-1: Halt Mode

20.2 Sleep Mode

After the CPU executes a SLEEP instruction, all oscillators stop working until the CPU receives an external interrupt signal or the CPU is reset. To avoid program errors caused by Interrupt return, it is necessary to add a NOP instruction after the SLEEP instruction to guarantee the program's normal execution as described in Example 20-2.

```
SLEEP
NOP
```

Example 20-2: Sleep Mode

To make sure that the CPU have the minimum power consumption in SLEEP mode, it is necessary to disable all the power management and analog circuits before executing a SLEEP instruction, and to make sure all the I/O ports are in VDD or VSS voltage levels. There are some parasitic diodes between VDDA and analog input ports as in Figure 20-1. When the voltage regulator is disabled and VDD is gradually going to VSS voltage level, it is necessary to keep AD0~AD4 in floating or VSS voltage level.

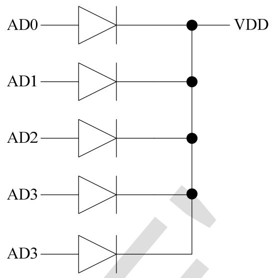


Figure 20-1: the Parasitic Diodes between VDD and Analog Input Ports

Example 20-3 is a recommended example program for user's reference before the CPU executes the SLEEP instruction.

```
CLRF     NETB
CLRF     NETD
CLRF     NETF
CLRF     NETG
MOVLW    01h
MOVWF    PT1PU
MOVLW    00h
MOVWF    PT1MR
MOVLW    0FEh
MOVWF    PT1EN
CLRF     PT1                ; Set PT1 [7:1] output low, PT1 [0] Input with pull up
CLRF     INTF
MOVLW    081h               ; Enable external Interrupt
MOVWF    INTE
SLEEP
NOP
```

Example 20-3: Example Program before the CPU Executes the SLEEP Instruction

21. Instruction Set

The FS98O02 instruction set consists of 37 instructions. Each instruction is a 16-bit word with an OPCODE and one or more operands. The detail descriptions are as below.

21.1 Instruction Set Summary

Table 21-1: Instruction Set Summary Table

Instruction	Operation	Cycle	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

Instruction	Operation	Cycle	Flag
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
HALT	CPU Stop	1	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] –Ċ	1	C, DC, Z
XORLW k	[W] ← [W] XOR k	1	Z
XORWF f, d TBLP k MOVP k	[Destination] ← [W] XOR [f] [EADRH,EADRL] ← EDAH,WORK (k is const) [EADRH,EADRL]→ EDAH,WORK ; (EADRH,EADRL) + k	1 (k*2)+1 2	Z None None

- **Note**
  - f: memory address. f may be 00h to 7Fh.
  - W: work register.
  - k: literal field, constant data or label.
  - d: destination select. If d = 0, store result in W. If d = 1, store result in memory address f.
  - b: bit select. b may be 0 to 7.
  - [f]: the content of memory address f.
  - PC: program counter.
  - C: Carry flag.
  - DC: Digit carry flag.
  - Z: Zero flag.
  - PD: power down flag.
  - TO: watchdog time out flag.
  - WDT: watchdog timer counter.

21.2 Instruction Description

The instruction descriptions are sort 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: **ADDLW 08h**	Before instruction: W = 08h After instruction: W = 10h

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: **ADDPCW**	Before instruction: W = 7Fh, PC = 0212h After instruction: PC = 0292h
Example 2: **ADDPCW**	Before instruction: W = 80h, PC = 0212h After instruction: PC = 0193h
Example 3: **ADDPCW**	Before instruction: W = FEh, PC = 0212h After instruction: PC = 0211h

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: **ADDWF OPERAND, 0**	Before instruction: OPERAND = C2h W = 17h After instruction: OPERAND = C2h W = D9h
Example 2: **ADDWF OPERAND, 1**	Before instruction: OPERAND = C2h W = 17h After instruction: OPERAND = D9h W = 17h

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 **ADDWFC OPERAND,1**	Before instruction: 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: **ANDLW 5Fh**	Before instruction: 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: **ANDWF OPERAND,0**	Before instruction: W = 0Fh, OPERAND = 88h After instruction: W = 08h, OPERAND = 88h
Example 2: **ANDWF OPERAND,1**	Before instruction: 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: **BCF FLAG, 2**	Before instruction: 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: **BSF FLAG, 2**	Before instruction: 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: Node **BTFSC FLAG, 2** OP1 : OP2 :	Before instruction: PC = address (Node) After instruction: 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: Node **BTFSS FLAG, 2** OP1 : OP2 :	Before instruction: PC = address (Node) After instruction: 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 $0 \le f \le 255$ $d \in [0,1]$
Operation	$[f] \leftarrow NOT([f])$
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 $0 \le f \le 255$ $d \in [0,1]$
Operation	$[Destination] \leftarrow [f] - 1$
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 $0 \le f \le FFh$ $d \in [0,1]$
Operation	$[Destination] \leftarrow [f] - 1$, 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 $[FLAG] \neq 0$ PC = address(OP2)

GOTO	Unconditional Branch
Syntax	GOTO k $0 \le k \le 1FFFh$
Operation	$PC \leftarrow k$
Flag Affected	None
Description	The immediate address is loaded into PC.
Cycle	2
Example: **GOTO THERE**	After instruction: PC = address (THERE)

HALT	Stop CPU Core Clock
Syntax	HALT
Operation	CPU Stop

Flag Affected	None
Description	CPU clock is stopped. Oscillator is running. CPU can be waked up by internal and external interrupt sources.
Cycle	1

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: **INCF OPERAND,0**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = 24h, OPERAND = 23h
Example 2: **INCF OPERAND,1**	Before instruction: 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: Node **INCFSZ FLAG, 1** OP1 : OP2 :	Before instruction: PC = address (Node) After instruction: [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: **IORLW 85H**	Before instruction: 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: **IORWF OPERAND,1**	Before instruction: 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: **MOVFW OPERAND**	Before instruction: 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: **MOVLW 23H**	Before instruction: W = 88h After instruction: W = 23h

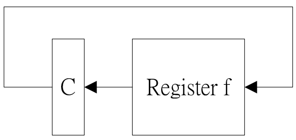
MOVWF	Move W to f
Syntax	MOVWF f 0 ≤ f ≤ FFh
Operation	[f] ← [W]
Flag Affected	None
Description	Move data from the W register to [f].
Cycle	1
Example: **MOVWF OPERAND**	Before instruction: W = 88h, OPERAND = 23h After instruction: W = 88h, OPERAND = 88h

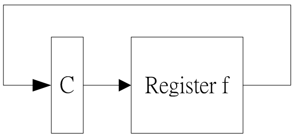
NOP	No Operation
Syntax	NOP
Operation	No Operation
Flag Affected	None
Description	No operation. NOP is used for one instruction cycle delay.
Cycle	1

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: **RETFIE**	After instruction: 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: CALL TABLE : TABLE ADDWF PC **RETLW k0** **RETLW k1** : **RETLW kn**	Before instruction: WREG = 0x07 After instruction: WREG = value of k7

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: **Return**	After instruction: 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

TBLP	Exclusive Programmer EPROM
Syntax	TBLP k k = 32 , that is const
Operation	[EADRH,EADRL]←EDAH,WORK
Flag Affected	None
Description	Exclusive Programmer EPROM. EPROM Address is in EADRH and EADRL. Programmer Data is in EDAH and WORK.
Cycle	(k*2)+1
Example: **TBLP k**	k is const For FS98O01 Chip , k = 32

MOVP	Exclusive Read EPROM Data
Syntax	MOVP k k is increment EPROM Address value.
Operation	[EADRH,EADRL]→EDAH,WORK
Flag Affected	None
Description	Exclusive Read EPROM data. EPROM Address is in EADRH and EADRL. MSB and LSB After MOVP Data Put in EDAH and WORK. MSB and LSB
Cycle	2
Example: **MOVP k**	After instruction: EPROM Data put in EDAH (MSB) and WORK (LSB) EPROM Address increment k (EADRH,EADRL)+k

Package Information

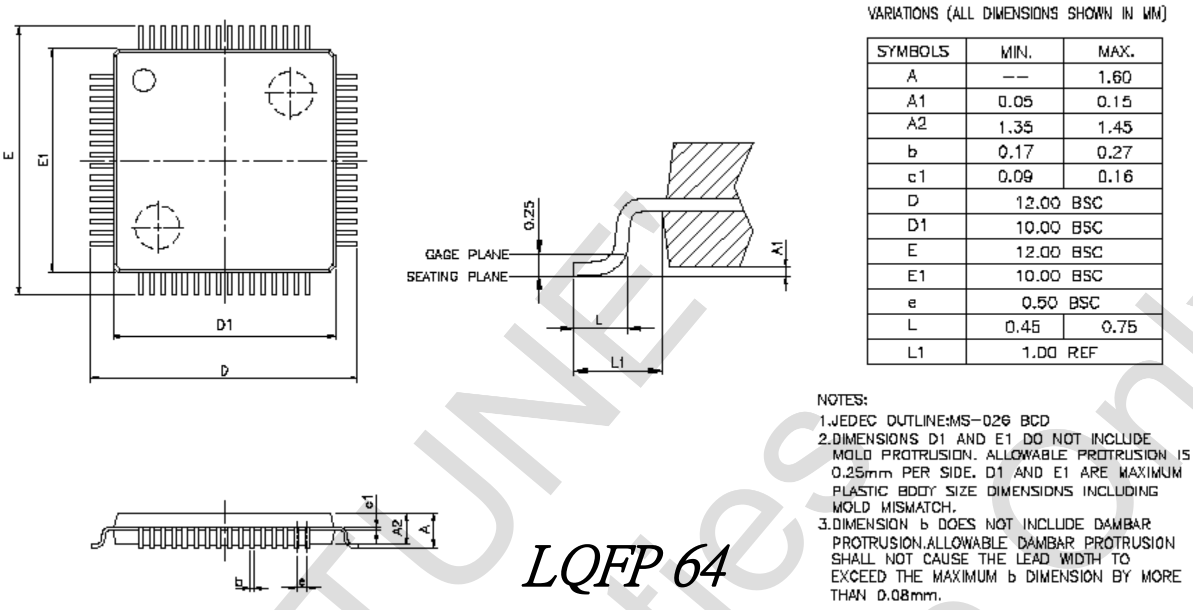


Figure 22-1: LQFP64 Package Outline

22. Ordering Information

Product Number	Package Type
FS98O02	Die form (49-pin), 64-pin LQFP (Green package)

23. Revision History

Version	Date	Page	Description
1.0	2010/06/21	All	Officially released version 1.0.
1.1	2010/8/25	11	The op offset value changed from 1mv to 1.5 mv.
1.2	2010/10/29	10	4MHz FRC Minimum value modify from 3.2MHz to 3.0MHz. 2.5V VDDA Minimum value modify from 2.4V to 2.37V.
1.3	2011/09/29	All	Add the FS98O02C Function Spec
1.4	2014/01/14	14	Revise OPAMP Characteristics
1.5	2014/05/22	2	Revised company address