

十速

# TM56F70C2

## *DATA SHEET*

### *Rev 0.91*

**tenx** reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. **tenx** does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. **tenx** products are not designed, intended, or authorized for use in life support appliances, devices, or systems. If Buyer purchases or uses **tenx** products for any such unintended or unauthorized application, Buyer shall indemnify and hold **tenx** and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that **tenx** was negligent regarding the design or manufacture of the part.

---

## AMENDMENT HISTORY

Version	Date	Description
0.90	2024/08	New Release
0.91	2025/01	*GPIO table optimization (p.37) *Add FIRC18.432 working current

## CONTENTS

<b>AMENDMENT HISTORY .....</b>	<b>2</b>
<b>CONTENTS.....</b>	<b>3</b>
<b>FEATURES .....</b>	<b>5</b>
<b>SYSTEM BLOCK DIAGRAM.....</b>	<b>8</b>
<b>PIN ASSIGNMENT DIAGRAM.....</b>	<b>9</b>
<b>PIN DESCRIPTIONS .....</b>	<b>11</b>
<b>FUNCTION DESCRIPTION.....</b>	<b>12</b>
1 CPU Core.....	12
1.1 ROM.....	12
1.1.1 Reset Vector (000h) .....	12
1.1.2 Interrupt Vector (004h) .....	12
1.1.3 Production Information Area and System Configuration (SYSCFG).....	13
1.1.4 Emulated EEPROM Area.....	14
1.1.5 ROM Low Power Mode .....	14
1.2 RAM and Special Function Registers .....	16
1.2.1 Bank .....	17
1.2.2 Directly Addressing and Indirect Addressing .....	17
1.3 Programming Counter (PC) and Stack.....	19
1.3.1 Programming Counter .....	19
1.3.2 Programming Counter Read and Write .....	19
1.3.3 Stack.....	19
1.4 ALU and Working (W) Register .....	21
1.5 STATUS Register (003h/083h/103h/183h) .....	21
1.6 Table Read.....	23
1.7 IAP and Emulated EEPROM .....	25
2 Reset .....	27
2.1 Power on Reset (POR) .....	27
2.2 Low Voltage Reset (LVR) .....	27
2.3 External Pin Reset (XRST) .....	27
2.4 Watchdog Timer Reset (WDTR) .....	28
3 Clock Circuitry and Operation Mode .....	29
4 Interrupt .....	33
5 I/O Port .....	37
5.1 GPIO (PA0-PA7, PB0-PB6) .....	37
5.2 OP0N / OPO / OP1N / VREXT .....	41
6 Peripheral Functional Block .....	42
6.1 Watchdog Timer (WDT).....	42
6.2 Wakeup Timer (WKT) .....	44
6.3 Timer0.....	46
6.4 Timer1 .....	51
6.5 T2:15-bit Timer .....	54

6.6	PWM .....	56
6.7	Analog-to-Digital Converter (ADC) .....	62
6.8	UART .....	66
6.9	Battery Charge Module (BCM) - DAC/Comparator/Amplifier.....	68
6.10	Cyclic Redundancy Check (CRC).....	73
6.11	In Circuit Emulation (ICE).....	74
<b>MEMORY MAP.....</b>		<b>75</b>
<b>INSTRUCTION SET .....</b>		<b>87</b>
<b>ELECTRICAL CHARACTERISTICS .....</b>		<b>101</b>
1.	Absolute Maximum Ratings .....	101
2.	DC Characteristics .....	101
3.	Clock Characteristics .....	103
4.	Reset Timing Characteristics .....	103
5.	Wakeup Timer (WKT) Timing Characteristics .....	103
6.	LVR Circuit Characteristics .....	104
7.	LVD Circuit Characteristics .....	104
8.	ADC Characteristics .....	105
9.	VBG Characteristics .....	105
10.	OPA Characteristics.....	105
11.	Comparator Characteristics .....	106
12.	Emulated EEPROM Characteristics .....	106
<b>CHARACTERISTICS GRAPHS .....</b>		<b>107</b>
<b>PACKAGING INFORMATION .....</b>		<b>110</b>

## FEATURES

1. **ROM: 4K\*16 bits Flash with Emulated EEPROM 32\*16 bits**
2. **RAM: 256 Bytes**
3. **STACK: 8 Levels**
4. **System Clock type selections:**
  - Built-in Fast RC oscillator (FIRC), 18.432 MHz
  - Built-in Slow RC oscillator (SIRC), 37 KHz
5. **System Clock Prescaler:**
  - System Clock can be divided by 1/2/4/8 option
6. **Power Saving Operation Mode**
  - FAST Mode: Slow-clock is enabled, Fast-clock keeps CPU running
  - SLOW Mode: Fast-clock can be disabled or enabled, Slow-clock keeps CPU running
  - IDLE Mode: Fast-clock and System clock stop. Slow-clock, T2, or Wake-up Timer keep running
  - STOP Mode: All clocks stop, T2 and Wake-up Timer stop
7. **3 Independent Timers**
  - Timer0
    - 8-bit timer divided by 1~256 pre-scale option / auto-reload / counter / interrupt / stop function
  - Timer1
    - 8-bit timer divided by 1~256 pre-scale option / auto-reload / interrupt / stop function
  - T2
    - IDLE mode wake-up timer or used as one simple 15-bit time base timer
    - 4 interrupt interval time options
    - Clock source: Slow-clock, Fsys/128, or FIRC/512

**8. Interrupt**

- Three External Interrupt pins
  - 1 pin is falling edge wake-up triggered & Interrupts
  - 2 pins are rising or falling edge wake-up triggered & Interrupt
- Timer0 / Timer1 / T2 / Wake-up Timer Interrupt
- ADC Interrupt
- PWM Interrupt
- UART Interrupt
- LVD Interrupt

**9. Wake-up Timer (WKT)**

- Clocked by built-in RC oscillator with 4 adjustable interrupt times
  - 28 ms / 55 ms / 111 ms / 221 ms @  $V_{CC}=5V$

**10. Watchdog Timer (WDT)**

- Clocked by built-in RC oscillator with 4 adjustable reset times
  - 221 ms / 443 ms / 1771 ms / 3542 ms @  $V_{CC}=5V$
- Watchdog timer can be disabled / enabled in STOP mode

**11. Three 16 bits PWMs**

- Independent PWM Duty
- Shared PWM Period
- PWM clock source: System clock ( $F_{sys}$ ), FIRC (18.432MHz), FIRC\*2 (36.864MHz)
- PWM0 supports complementary output (PWM0P, PWM0N) with non-overlap output option

**12. 12-bit ADC with 17 channels External Pin Input and 7 channels Internal Voltage Channel**

- Internal voltage channel: VR, OPA2TOADC,  $V_{TEMP}$ , LDO1.2V, VSS,  $V_{CC}/201$ ,  $V_{CC}/4$
- ADC reference voltage:  $V_{CC}$ , VR, LDO1.2V

**13. BCM**

- 14-bit DAC0 and comparator OPA0 are used for constant current control
- 14-bit DAC1 and comparator OPA1 are used for constant voltage control
- OPA2 is 1/10/20/50 times amplifier

**14. UART**

- Baud rate up to 115200
- Support single-wire mode

**15. Reset**

- Power On Reset (POR)
- Low Voltage Reset (LVR)
- External Pin Reset (XRST)
- Watchdog Timer Reset (WDTR)

**16. Low Voltage Reset (LVR) and Low Voltage Detection (LVD)**

- 16-Level Low Voltage Reset: 2.13V ~ 4.26V, can be disabled
- 15-Level Low Voltage Detection: 2.24V~4.20V, can be disabled, with hysteresis option.

**17. Operating Voltage: 2.2V~5.5V**

\*Power-up  $V_{CC}$  must exceed POR 2.0V and user selected LVR level, refer to the “Electrical Characteristics Graphs” to avoid entering ROM dead zone.

**18. Operating Temperature : -40°C to + 105°C****19. Table Read Instruction: 16-bit ROM data lookup table****20. 16-bit Cyclic Redundancy Check (CRC) function****21. Instruction set: 39 Instructions****22. I/O ports:**

- 15 GPIO
  - Open-Drain Output
  - CMOS Push-Pull Output
  - Schmitt Trigger Input with pull-up / pull-down resistor option
  - All I/O with High-Sink
  - 1/2 VCC (1/2 bias) Output
  - Support wake up function

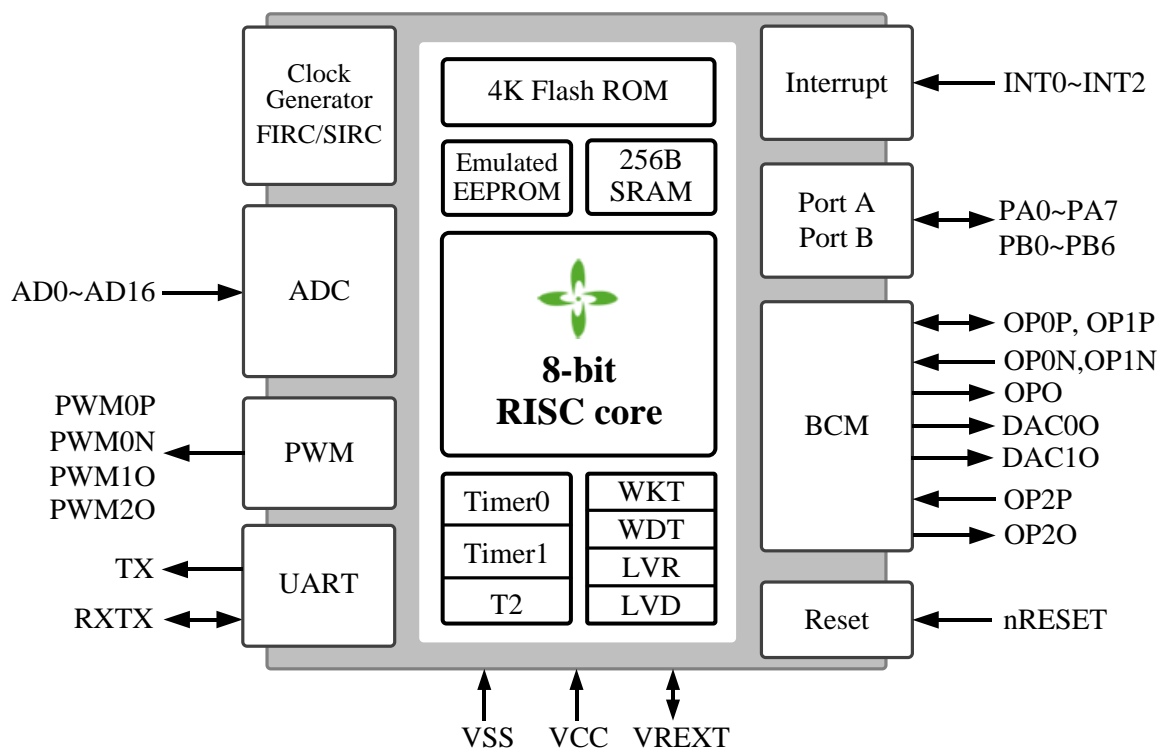
**23. Programming connectivity support 4-wire (ICP) or 6-wire program****24. Package Types:**

- 20-pin SOP (300 mil)
- 20-pin TSSOP (173 mil)
- 20-pin QFN20 (3\*3\*0.75-0.4mm)
- 16-pin SOP (150 mil)

**25. On-chip Debug/ICE Interface**

- Use 2-wire dedicated ICE pin, no GPIO occupied

## SYSTEM BLOCK DIAGRAM

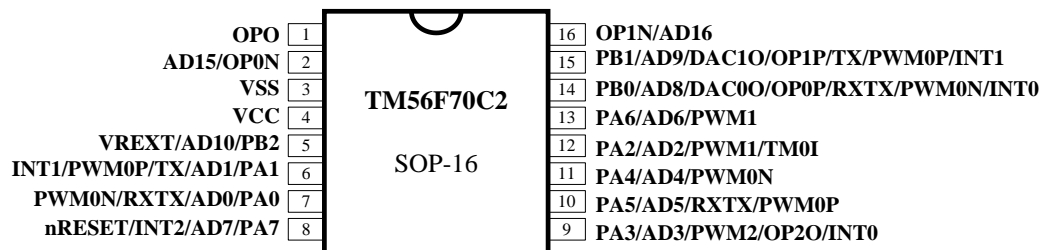
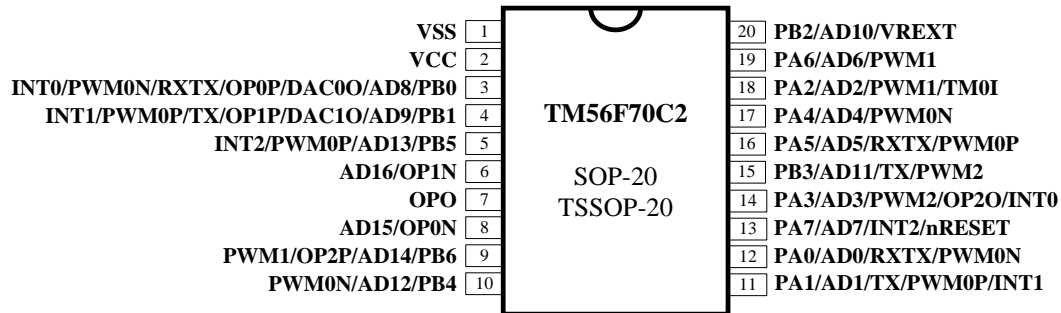


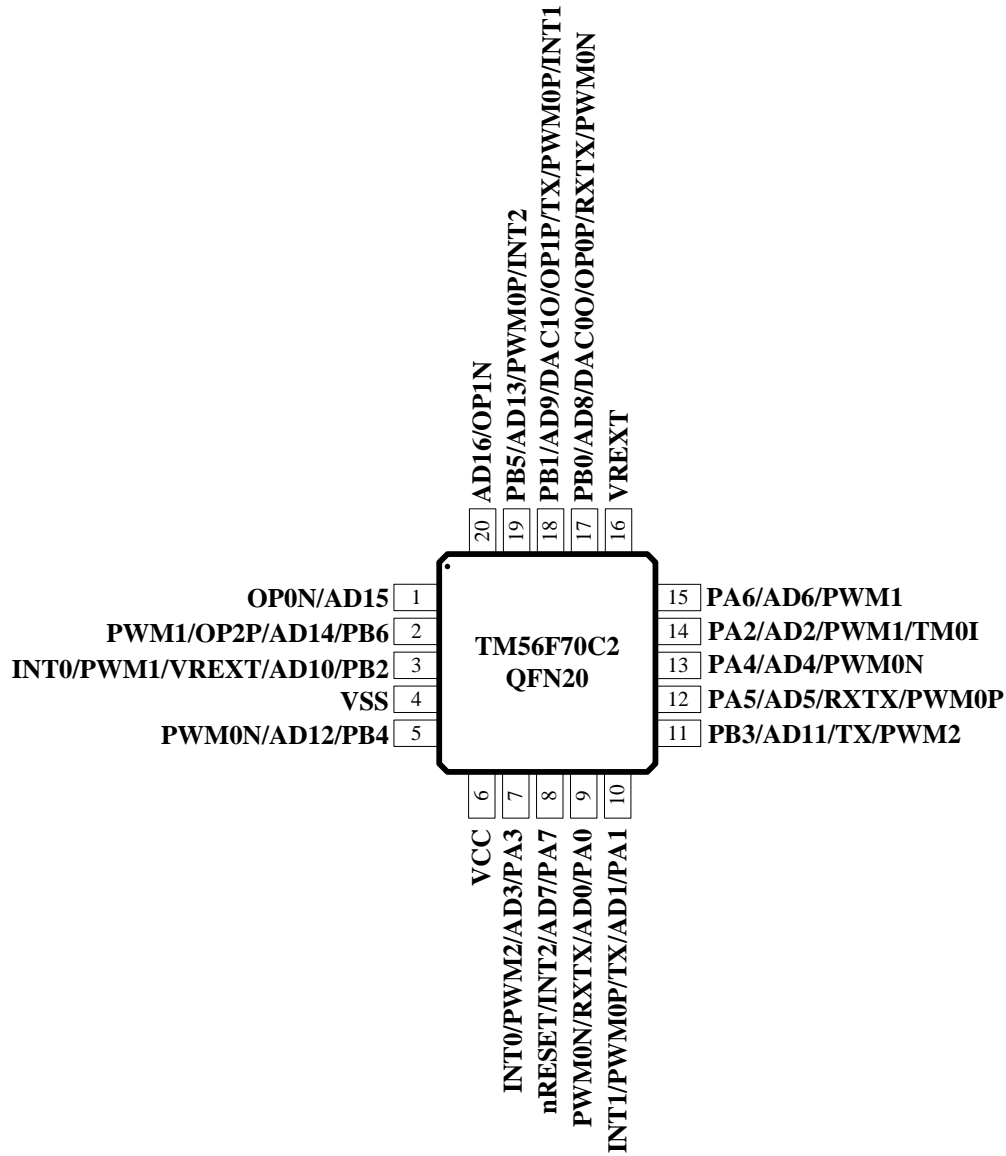
TM56F70C2 Block Diagram



## PIN ASSIGNMENT DIAGRAM

If it is a low-power application, all digital I/O (including unpinned or unused pins) should avoid being set to a high-impedance state.





## PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0~PA7 PB0~PB6	I/O	GPIO, function include Schmitt trigger input, CMOS push-pull output, open-drain output, $1/2V_{CC}$ output, pull-up/pull-down resistor, pin wake-up, etc.
nRESET	I	External active low reset
VCC, VSS	P	Power Voltage input pin and ground
INT0~INT2	I	External interrupt input
TM0I	I	Timer0's input in counter mode
PWM0P	O	PWM0 positive output
PWM0N	O	PWM0 negative output
PWM1	O	PWM1 output
PWM2	O	PWM2 output
AD0~16	I	ADC channel input
TX	O	UART serial data output
RXTX	I/O	UART serial data input, can also be used as output under single-wire mode
OPO	O	OPA0~OPA1 open drain output
OP2O	O	OPA2 output
OP1N	I	OPA1 inverting input
OP0N	I	OPA0 inverting input
OP1P/DAC1O	I/O	OPA1 non-inverting input or DAC1 output
OP0P/DAC0O	I/O	OPA0 non-inverting input or DAC0 output
OP2P	I	OPA2 non-inverting input
VREXT	I/O	Internal reference voltage LDO3V output or External reference voltage input

Programming pins:

6-wire: VCC / VSS / PA0 / PA1 / PA2 / PA4

4-wire: VCC / VSS / PA0 / PA1

\*All components of PA0 and PA1 need to be removed from the PCB during In-Circuit Programming.

## FUNCTION DESCRIPTION

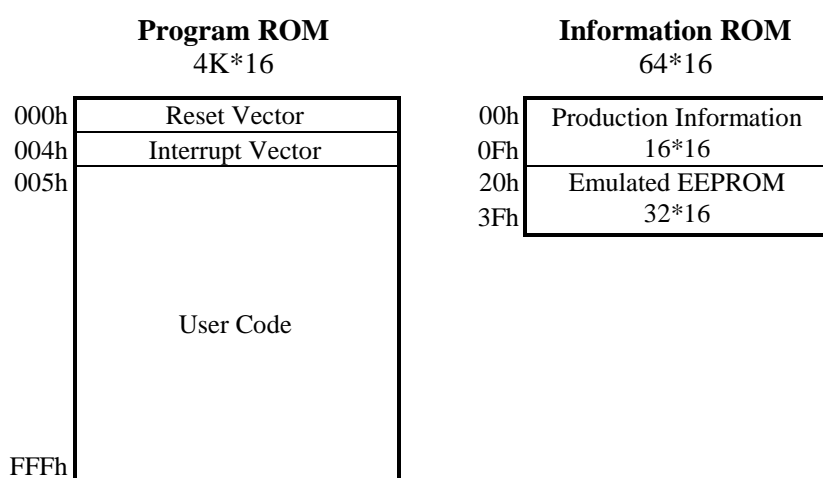
### 1 CPU Core

#### 1.1 ROM

The size of the Program ROM is 4K\*16, with an additional 64\*16 Information ROM.

Under the writer, when the PROTECT bit is set to 0, the PROM and information ROM can be read and written normally. When the PROTECT bit is set to 1, the PROM cannot be read, and only the information ROM is allowed to be read.

The PROTECT bit is only allowed to be cleared after the program ROM has been erased to 0.



##### 1.1.1 Reset Vector (000h)

After reset, system will restart the program counter (PC) at the address 000h, all registers will revert to the default value.

##### 1.1.2 Interrupt Vector (004h)

When an interrupt occurs, the program counter (PC) will be pushed onto the stack and jumps to address 004h.

### 1.1.3 Production Information Area and System Configuration (SYSCFG)

The production information area is placed at the beginning of the information ROM and stores production code, checksum values, trim values and other information. The 16-bit system configuration (SYSCFG) is also placed here, as shown in the table below.

Default Value		0000_0000_0000_0000	
Bit		Description	
SYSCFG	15	<b>PROTECT</b> : Code protection selection	
		0	Disable
		1	Enable
	13-12	<b>WDTE</b> : WatchDog Timer Reset Enable	
		0X	Disable
		10	Enable in FAST/SLOW mode, Disable in IDLE/STOP mode
		11	Always Enable
	11-8	<b>LVRS</b> : Low Voltage Reset Selection	
		0000	2.13V
		0001	2.26V
		0010	2.40V
		0011	2.54V
		0100	2.69V
		0101	2.83V
		0110	2.97V
		0111	3.11V
		1000	3.26V
		1001	3.40V
		1010	3.54V
		1011	3.68V
		1100	3.84V
		1101	3.98V
		1110	4.12V
		1111	4.26V
	7	<b>XRSTE</b> : External Pin (PA7) Reset Enable	
		0	Disable (PA7 as I/O pin)
		1	Enable
	6	<b>SOPAN</b> : OPAN Input Switch Selection	
		0	No change
		1	Swapping the input of OPAN (OPA0N <-> OPA1N)
	5	<b>FRCPSC</b> : FIRC prescaler (PWMCLK = FIRC and FIRC*2 are not affected by this bit)	
		0	FIRC is 18.432MHz
		1	FIRC is 9.216MHz
	4	<b>PORSEL</b> : Power on Reset (POR) Selection	
		0	Power on Reset enable 100% duty cycle
		1	Power on Reset enable 1/16 duty cycle
	3-0	Reserved	

#### 1.1.4 Emulated EEPROM Area

The Emulated EEPROM area is placed in the second half of the information ROM. Different simulated EEPROM area ranges are defined through the register IAPEN. Users can write data here through IAP and read data through Table Read. For details on the usage of IAP, see the “IAP and Emulated EEPROM” chapter.

#### 1.1.5 ROM Low Power Mode

The default is high speed mode, ROM can reduce power consumption by switching modes. Before changing the ROM mode through the ROMODS register, the user must first write any value to the RDSTP register to suspend ROM reading for a total of 4 system cycles to ensure that the ROM mode switch is successfully completed. The example is as follows.

Example: Switch to ROM low power mode, system clock < 1MHz.

( These three lines of code must be executed continuously, and no other code can be inserted in between )

```
; Write any value to RDSTP register
      MOVWX    RDSTP
;Set ROMODS = 00
      MOVLW    11111100h
      ANDWX    PWRCTL2
```

Example: Switch to ROM medium power mode, system clock < 4MHz.

( These three lines of code must be executed continuously, and no other code can be inserted in between )

```
; Write any value to RDSTP register
      MOVWX    RDSTP
;Set ROMODS = 01
      MOVLW    11111101h
      ANDWX    PWRCTL2
```

105h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL2	GPR2		—	—	—	HSINK	ROMODS	
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	—	0	1	1	1	1

105h.2 **ROMODS:** ROM mode selection  
 11: High speed mode  
 01: Medium power mode, Fsys < 4MHz  
 00: Low power mode, Fsys < 1MHz

106h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RDSTP	RDSTP							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

002h.7~0 **RDSTP:** Read stop  
 Before changing the ROM mode, the user must first write any value to this register to suspend ROM reading for a total of 4 system cycles to ensure that the ROM mode switch is successfully completed.

## 1.2 RAM and Special Function Registers

The table of Special Function Registers (SFR) and RAM is as follows. It is divided into 4 BANKs. Some commonly used registers will be placed in multiple BANKs to reduce the frequency of switching BANKs. The RAM includes 20h~7Fh, A0h~FFh, 120h~17Fh, the size is 256 bytes, of which 0F0h~0FFh, 170h~17Fh, 1F0h~1FFh all point to positions 070~7Fh.

【BANK0】 000~07Fh		【BANK1】 080h~0FFh		【BANK2】 100h~17Fh		【BANK3】 180h~1FFh	
000h	<b>INDF</b>	080h	<b>INDF</b>	100h	<b>INDF</b>	180h	<b>INDF</b>
001h	<b>TM0</b>	081h	<b>OPTION</b>	101h	<b>TM0</b>	181h	<b>OPTION</b>
002h	<b>PCL</b>	082h	<b>PCL</b>	102h	<b>PCL</b>	182h	<b>PCL</b>
003h	<b>STATUS</b>	083h	<b>STATUS</b>	103h	<b>STATUS</b>	183h	<b>STATUS</b>
004h	<b>FSR</b>	084h	<b>FSR</b>	104h	<b>FSR</b>	184h	<b>FSR</b>
005h	<b>PAD</b>	085h	<b>PAMOD10</b>	105h	<b>PWRCTL2</b>	185h	<b>DPL</b>
006h	<b>PBD</b>	086h	<b>PAMOD32</b>	106h	<b>RDSTP</b>	186h	<b>DPH</b>
007h		087h	<b>PAMOD54</b>	107h		187h	<b>CRCDL</b>
008h		088h	<b>PAMOD76</b>	108h		188h	<b>CRCDH</b>
009h		089h	<b>PWMCTL</b>	109h	<b>LVRPD</b>	189h	<b>CRCIN</b>
00Ah	<b>SFR0A</b>	08Ah	<b>SFR0A</b>	10Ah	<b>SFR0A</b>	18Ah	<b>SFR0A</b>
00Bh	<b>INTIE</b>	08Bh	<b>INTIE</b>	10Bh	<b>INTIE</b>	18Bh	<b>INTIE</b>
00Ch	<b>INTIF</b>	08Ch	<b>PBMOD10</b>	10Ch	<b>SFR10C</b>	18Ch	<b>TABR</b>
00Dh	<b>INTIE1</b>	08Dh	<b>PBMOD32</b>	10Dh	<b>CFG07</b>	18Dh	
00Eh	<b>INTIF1</b>	08Eh	<b>PBMOD54</b>	10Eh	<b>BGTRIM</b>	18Eh	
00Fh	<b>CLKCTL</b>	08Fh	<b>PBMOD76</b>	10Fh	<b>IRCF</b>	18Fh	
010h	<b>TM0RLD</b>	090h		110h	<b>OP0TRIM</b>	190h	<b>IAPCTL</b>
011h	<b>TM0CTL</b>	091h	<b>OPTION2</b>	111h	<b>OP1TRIM</b>	191h	<b>IAPEN</b>
012h	<b>TM1</b>	092h	<b>PWMPRDH</b>	112h	<b>OP2TRIM</b>	192h	<b>IAPDT</b>
013h	<b>TM1RLD</b>	093h	<b>PWMPRDL</b>	113h	<b>RDCTL</b>	193h	<b>IAPDTH</b>
014h	<b>TM1CTL</b>	094h	<b>PWM0DH</b>	114h	<b>BCMCTL3</b>	194h	
015h	<b>T2CTL</b>	095h	<b>PWM0DL</b>	115h	<b>PWRCTL</b>	195h	<b>SCON</b>
016h	<b>LVCTL</b>	096h	<b>PWM1DH</b>	116h		196h	<b>SBUF</b>
017h	<b>ADCDH</b>	097h	<b>PWM1DL</b>	117h		197h	<b>UARTCTL</b>
018h	<b>ADCTL</b>	098h	<b>PWM2DH</b>	118h		198h	<b>UARTCTL2</b>
019h	<b>ADCTL2</b>	099h	<b>PWM2DL</b>	119h		199h	
01Ah	<b>BCMCTL</b>	09Ah		11Ah		19Ah	
01Bh	<b>BCMCTL2</b>	09Bh		11Bh		19Bh	
01Ch	<b>DAC0DH</b>	09Ch		11Ch		19Ch	
01Dh	<b>DAC0DL</b>	09Dh		11Dh		19Dh	
01Eh	<b>DAC1DH</b>	09Eh		11Eh		19Eh	
01Fh	<b>DAC1DL</b>	09Fh		11Fh		19Fh	
020h		0A0h		120h		1A0h	
	RAM Bank0 area (80 Bytes)		RAM Bank1 area (80 Bytes)		RAM Bank2 area (80 Bytes)		Reserved
06Fh		0EFh		16Fh		1EFh	
070h	common area (16 Bytes)	0F0h	accesses 070h~07Fh	170h	accesses 070h~07Fh	1F0h	accesses 070h~07Fh
07Fh		0FFh		17Fh		1FFh	

**RAM and Special Function Register Table**



### 1.2.1 Bank

The purpose of registers RP1 and RP0 is to switch BANK.

[RP1, RP0] (03h.6~5)	BANK
00	0
01	1
10	2
11	3

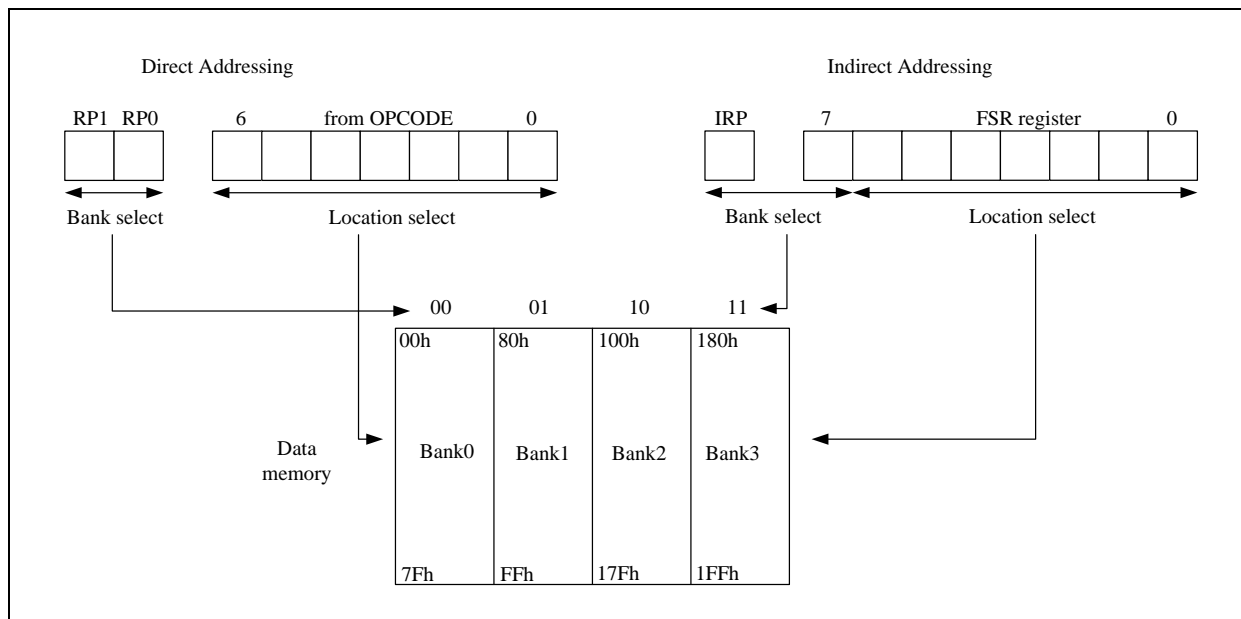
**Keeping RP0=RP1=0 in the beginning of the F/W code and using the new instruction set.**

The advantage of using new instruction is user can ignore the bank location of registers and the code size can be saved. The new instruction is almost the same as the old instruction. By replacing the “F” to “X” in the instruction set can easily use the new instruction without switching the bank. The instruction replacement table is as follows.

BC <b>F</b>	TM0IE	→	BC <b>X</b>	TM0IE
DEC <b>F</b>	CNT, 1	→	DEC <b>X</b>	CNT, 1
INC <b>F</b> SZ	RAM25, 0	→	INC <b>X</b> SZ	RAM25, 0
MOV <b>W</b> <b>F</b>	PAMODL	→	MOV <b>W</b> <b>X</b>	PAMODL
RL <b>F</b>	RAMA0, 0	→	RL <b>X</b>	RAMA0, 0
SWAP <b>F</b>	ADCTL, 0	→	SWAP <b>X</b>	ADCTL, 0

### 1.2.2 Directly Addressing and Indirect Addressing

The plane can be addressed directly or indirectly. The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing. Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = ‘0’) will read 00h. Writing to the INDF register indirectly results in a no operation (although status bit may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS[7]). Refer to the figure below.



**Direct / Indirect Addressing**

◇ Example: read / write register by using direct addressing (**force RP0=RP1=0**)

CLKCTL	equ	00Fh	; SFR in Bank0
TM1	equ	012h	; SFR in Bank0
OPTION2	equ	091h	; SFR in Bank1
LVRPD	equ	109h	; SFR in Bank2
IRCF	equ	10Fh	; SFR in Bank2
DPL	equ	185h	; SFR in Bank3
RAM020	equ	020h	; RAM in Bank0
RAM0A0	equ	0A0h	; RAM in Bank1
MOVXW	TM1		; read TM1 (Bank0) to W
MOVXW	OPTION2		; read OPTION2 (Bank1) to W
MOVXW	IRCF		; read IRCF (Bank2) to W
MOVXW	DPL		; read DPL (Bank3) to W
MOVLW	16h		; W = 16h
MOVWX	RAM020		; RAM[0x020] = W = 16h
MOVWX	RAM0A0		; RAM[0x0A0] = W = 16h
MOVLW	37h		; W = 37h
MOVWX	LVRPD		; LVRPD = W = 37h, force LVR/POR disable
MOVXW	CLKCTL		; read SFR CLKCTL (00Fh) to W
MOVXW	IRCF		; read SFR IRCF (10Fh) to W
MOVLW	0Bh		; W = 0Bh
MOVWX	CLKCTL		; CLKCTL (00Fh) = W = 0Bh
MOVWX	IRCF		; IRCF (10Fh) = W = 0Bh

◇ Example: read / write register by using indirect addressing (**force RP0=RP1=0**)

BSX	IRP		; IRP = 1 => Bank2/3
MOVLW	0Fh		; W = 0Fh
MOVWX	FSR		; FSR = W = 0Fh
MOVXW	INDF		; read SFR IRCF (10Fh) to W
BSX	IRP		; IRP = 1 => Bank2/3
MOVLW	0Fh		; W = 0Fh
MOVWX	FSR		; FSR = W = 0Fh
MOVLW	0Bh		; W = 0Bh
MOVWX	INDF		; IRCF (10Fh) = W = 0Bh
BCX	IRP		; IRP = 0 => Bank0/1
MOVLW	0Fh		; W = 0Fh
MOVWX	FSR		; FSR = W = 0Fh
MOVXW	INDF		; read SFR CLKCTL (00Fh) to W
BCX	IRP		; IRP = 0 => Bank0/1
MOVLW	0Fh		; W = 0Fh
MOVWX	FSR		; FSR = W = 0Fh
MOVLW	0Bh		; W = 0Bh
MOVWX	INDF		; CLKCTL (00Fh) = W = 0Bh

## 1.3 Programming Counter (PC) and Stack

### 1.3.1 Programming Counter

The program counter has a total of 12 bits and is used to address the 4Kx16 program ROM.

When a program instruction is executed, the program counter will contain the address of the next program instruction to be executed. The program counter usually continues to increment by one unless a reset, interrupt, call, jump, or return instruction is encountered.

The initial setup reset vector (000h) and interrupt vector (004h) are used for program counter initialization and interrupt events. For CALL instructions and GOTO instructions, the program counter loads the lower 11-bit address from the instruction word and the upper 1-bit address from register SFR0A.3. For return (RET/RETI/RETLW) instructions, the program counter retrieves its contents from the top of the stack.

Before executing the CALL/GOTO instruction, if the target address is greater than 2K, SFR0A.3 must be set, otherwise SFR0A.3 remains at 0.

This chip also provides additional new instructions LCALL and LGOTO to replace the CALL and GOTO instructions. When using LCALL and LGOTO, the user does not need to worry about the destination address, just keep SFR0A.3 to 0. The instruction replacement table is as follows.

CALL	TABLE	➔	LCALL	TABLE
GOTO	TABLE	➔	LGOTO	TABLE

### 1.3.2 Programming Counter Read and Write

The upper byte of the program counter (PC[11:8]) can be read from the PCH register (10Ch.3~0).

The low byte of the program counter (PC[7:0]) data can be read and written through the PCL register (002h/082h/102h/182h).

Use the PCH\_LAT function:

The default setting of the chip is that when the CPU executes an "instruction that will modify PCL", PC[11:8] is provided by the register PCH\_LAT.

Disable the PCH\_LAT function:

When the user writes 1C to the register SFR10C, the chip will disable the PCH\_LAT function. When the CPU executes an "instruction that modifies PCL", it will leave PC[11:8] unchanged for easy table lookup. Please note that the PCH\_LAT function can only be disabled when the user uses assembly code. Users of C language cannot disable this function.

Restore PCH\_LAT function:

When the user writes any other value to SFR10C, the system resumes the PCH\_LAT function.

### 1.3.3 Stack

The stack is 12 bits wide and 8 levels deep, used to store the address of program instructions. When calling (CALL/LCALL) instructions and interrupt events, they will be pushed into the stack in order. According to the first-in-last-out principle, when the return (RET/RETI/RETLW) instructions are executed, they will be popped back into the stack in order.

002h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	PCL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

002h.7~0 **PCL**: Programming Counter(PC) data bit 7~0

00Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SFR0A	GPR				PCH_LAT			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	0

00Ah.3~0 **PCH\_LAT**: Program counter(PC) high byte write buffer

When the CPU executes any instruction that will modify PCL, the PC[11:8] value is provided by the temporary register PCH\_LAT. This function can be turned off by register SFR10C.

10Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SFR10C	SFR10C							
R/W	W				R/W			
Reset	0	0	0	0	0	0	0	0

10Ch.7~0 **SFR10C**:

Use the PCH\_LAT function:

The default setting of the chip is that when the CPU executes an "instruction that will modify PCL", PC[11:8] is provided by the register PCH\_LAT.

Disable the PCH\_LAT function:

When the user writes 1C to the register SFR10C, the chip will disable the PCH\_LAT function.

When the CPU executes an "instruction that modifies PCL", it will leave PC[11:8] unchanged for easy table lookup. Please note that the PCH\_LAT function can only be disabled when the user uses assembly code. Users of C language cannot disable this function.

Restore PCH\_LAT function:

When the user writes any other value to SFR10C, the system resumes the PCH\_LAT function.

10Ch.3~0 **PCH**: Program counter(PC) data bits 11~8, which are the high 4-bit value of the program counter.

## 1.4 ALU and Working (W) Register

The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. In two-operand instructions, typically one operand is the W register, which is an 8-bit non-addressable register used for ALU operations. The other operand is either a file register or an immediate constant. In single operand instructions, the operand is either W register or a file register. Depending on the instruction executed, the ALU may affect the values of Carry (C), Digit Carry (DC), and Zero (Z) Flags in the STATUS register. The C and DC flags operate as a /Borrow and /Digit Borrow, respectively, in subtraction.

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

## 1.5 STATUS Register (003h/083h/103h/183h)

This register contains the arithmetic status of ALU and the Reset status. The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. It is recommended, therefore, that only BCX, BSX and MOVWX instructions are used to alter the STATUS Register because these instructions do not affect those bits.

STATUS	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Bit	Description							
7	<b>IRP</b> : Register Bank Select bit (used for indirect addressing) 0 = Bank 0,1 (000h - 0FFh) 1 = Bank 2,3 (100h - 1FFh)							
6:5	<b>RP1:RP0</b> : Register Bank Select bits (used for direct addressing) 00 = Bank 0 (000h - 07Fh) 01 = Bank 1 (080h - 0FFh) 10 = Bank 2 (100h - 17Fh) 11 = Bank 3 (180h - 1FFh) Each bank is 128 bytes							
4	<b>TO</b> : Time Out Flag 0: after Power On Reset or CLRWD T/SLEEP instruction 1: WDT time out occurs							
3	<b>PD</b> : Power Down Flag 0: after Power On Reset or CLRWD T instruction 1: after SLEEP instruction							
2	<b>Z</b> : Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero							
1	<b>DC</b> : Decimal Carry Flag or Decimal / Borrow Flag							
	ADD instruction				SUB instruction			
	0: no carry 1: a carry from the low nibble bits of the result occurs				0: a borrow from the low nibble bits of the result occurs 1: no borrow			
0	<b>C</b> : Carry Flag or /Borrow Flag							
	ADD instruction				SUB instruction			
	0: no carry 1: a carry occurs from the MSB				0: a borrow occurs from the MSB 1: no borrow			

◇ Example: Write immediate data into STATUS register.

```
MOVLW    00h
MOVWX    STATUS           ; Clear STATUS register
```

◇ Example: Bit addressing set and clear STATUS register.

```
BSX      STATUS, 0        ; Set C=1
BCX      STATUS, 0        ; Clear C=0
```

◇ Example: Determine the C flag by BTXSS instruction.

```
BTXSS    STATUS, 0        ; Check the carry flag
LGOTO    LABEL_1          ; If C=0, goto LABEL_1
LGOTO    LABEL_2          ; If C=1, goto LABEL_2
```

## 1.6 Table Read

The device can read the PROM value through the instruction TABRL / TABRH or the register TABR, and the read value will be stored in the register W.

The function needs to be enabled through register IAPEN. When not in use, please disable the function through the register IAPEN.

Example: Find PROM data located at "TABLE1"

```

        MOVLW    47H
        MOVWX    IAPEN

        MOVLW    00h
        MOVWX    INDEX            ; Set lookup address
        MOVLW    1Ch              ; Disable PCG_LAT function
        MOVWX    SFR10C

        MOVXW    INDEX
        LCALL    TABLE1          ; Find data and get W=33h
        ...
        INCX     INDEX, 1         ; next address
        LCALL    TABLE1          ; Find data and get W=44h
        ...

        MOVLW    33H
        MOVWX    IAPEN

ORG     X00h
TABLE1:
        ADDWX    PCL, 1           ; Add W to PCL and return the result to PCL.
        RETLW    33h             ; return W=33h
        RETLW    44h             ; return W=44h
        RETLW    55h             ; return W=55h

```

\*The chip defines 256 ROM addresses as one page, so the ROM has 16 pages, 000h~0FFh, 100h~1FFh,..., F00h~FFFh. The lookup table must be on the same page to avoid getting wrong data. Therefore, when a lookup table is started at X00h (X = 1, 2, 3, ... , E, F), the lookup table can have up to 255 data. Of course, if there is less data in the lookup table, you don't need to set the starting address to X00h, but just make sure that all the lookup table data is on the same page.

Example: Read PROM data located at "TABLE2"

```
MOVLW    47H
MOVWX    IAPEN
```

```
MOVLW    (TABLE2 >>8) & 0xff
MOVWX    DPH
MOVLW    (TABLE2) & 0xff
MOVWX    DPL                ; DPTR = {DPH, DPL} = TABLE2
```

; Method 1: Read the table through the instruction TABRL / TABRH

```
TABRL                ; Read PROM low byte data to W (W = 86h)
TABRH                ; Read PROM high byte data to W (W = 19h)
```

...

; Method 2: Read the table through the special function register TABR

```
MOVLW    01h                ; TABR = 01h is equivalent to instruction TABRL
MOVWX    TABR                ; Read PROM low byte data to TABR and W
                                ; TABR = W = 86h

MOVLW    02h                ; TABR = 02h is equivalent to instruction TABRH
MOVWX    TABR                ; Read PROM high byte data to TABR and W
                                ; TABR = W = 19h
```

...

```
MOVLW    33H
MOVWX    IAPEN
```

TABLE2:

```
.DT        0x1986                ; 16-bit ROM data
```

18Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TABR	TABR							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

18Ch.7~0 **TABR**: Table Read

When the user writes 01h to TABR, the W register will get the lower eight bits of the data in the address pointed to by DPTR.

When the user writes 02h to TABR, the W register will get the upper eight bits of the data in the address pointed to by DPTR.

In Assembly code, user can table read by TABRL/TABRH instruction or writing TABR register.

In C code, user can only table read by writing TABR register.

191h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IAPEN	IAPEN							
R/W	W							
Reset	0	0	0	0	0	0	0	0

191h.7~0 **IAPEN**: Function selection of Table Read and IAP

Write 47h to enable Main ROM Table Read and IAP functions

Write 50h to enable INFO ROM address 6'h20~ 6'h3F Table Read and IAP functions

Writing 33h will disable Table Read and IAP functions



## 1.7 IAP and Emulated EEPROM

First, IAPEN must be set. For example, writing 50H to IAPEN can enable the IAP function and treat 20H~3FH of the Information ROM as an area that can be written by IAP and read by Table read, called the Emulated EEPROM area. DPTR represents the address pointed to by writing and reading. 16 bits can be written at a time. The value to be written needs to be placed in the registers IAPDTH and IAPDTL. Fill in IAPDTH first and then IAPDTL. When filling in the value of IAPDTL, the hardware will perform writing.

The value of the Emulated EEPROM area can be read through the register TABR or the instructions TABRH/TABRL.

when not in use, please close IAPEN.

Example: Write 16'hAA55 to Emulated EEPROM and read it

; IAP Write Time-Out selection

MOVLW 00000001B

MOVWX IAPCTL

;Enable the IAP function and treat 20H~3FH of the Information ROM as an area that can be written by IAP and read by Table read, called the Emulated EEPROM area

MOVLW 50H

MOVWX IAPEN

;Set Write/Read Address = Data pointer register (DPTR) = 30h

MOVLW 00H

MOVWX DPH

MOVLW 30H

MOVWX DPL

;Write

MOVLW 0XAA

MOVWX IAPDTH ;Set data high byte

MOVLW 0X55

MOVWX IAPDT ;Set data low byte and write

NOP ;delay 1 clock cycle

;Read

MOVLW 02H ;=TABRH instruction

MOVWX TABR ;=TABRH instruction

MOVLW 01H ;=TABRL instruction

MOVWX TABR ;=TABRL instruction

;Disable IAP function

MOVLW 33H

MOVWX IAPEN

190h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IAPCTL							IAPTE	
R/W							R/W	R/W
Reset							0	0

190h.1~0 **IAPTE**: IAP Write Time-Out selection  
 00: Disable, 01: 3.5ms, 10: 14ms, 11: 28ms

191h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IAPEN	IAPEN							
R/W	W							
Reset	0	0	0	0	0	0	0	0

191h.7~0 **IAPEN**: Function selection of Table Read and IAP  
 Write 47h to enable Main ROM Table Read and IAP functions  
 Write 50h to enable INFO ROM address 6'h20~ 6'h3F Table Read and IAP functions  
 Writing 33h will disable Table Read and IAP functions

192h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IAPDTL	IAPDTL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

192h.7~0 **IAPDTL**: IAP Data low byte  
 When the user writes to this register, the hardware will automatically write the 16-bit value {IAPDTH, IAPDTL} to the location pointed to by DPTR.

193h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IAPDTH	IAPDTH							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

193h.7~0 **IAPDTH**: IAP Data high byte

## 2 Reset

This device can be RESET in four ways.

Power-On-Reset (POR)

Low Voltage Reset (LVR)

External Pin Reset (XRST)

Watchdog Timer Reset (WDTR)

Resets can be caused by Power on Reset (POR), External Pin Reset (XRST), Watchdog Timer Reset (WDTR), or Low Voltage Reset (LVR). The SYSCFG controls the Reset functionality. After Reset, the SFRs are returned to their default value, the program counter (PC) is cleared, and the system starts running from the reset vector 000h place. The TO and PD flags at status register (STATUS) are indicate system reset status.

### 2.1 Power on Reset (POR)

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values.

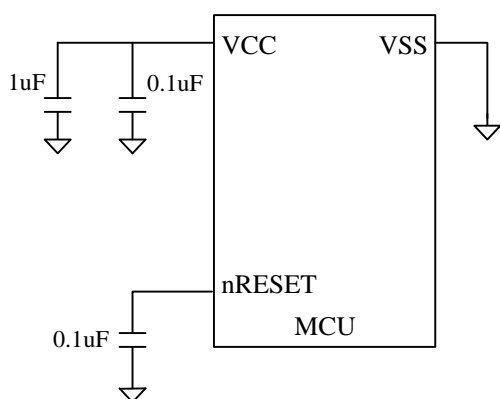
### 2.2 Low Voltage Reset (LVR)

The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are 16 threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG. Different Fsys have different system minimum operating voltage, reference to Operating Voltage of DC characteristics, if current system voltage is low than minimum operating voltage and lower LVR is selected, then the system maybe enters dead-band and error occurs.

### 2.3 External Pin Reset (XRST)

External pin reset can be disabled or enabled through the SYSCFG. It needs to be maintained for at least 2 SIRC clock cycles before it can be detected by the chip and trigger the reset action. XRST will return all control registers to their default resets, while the TO/PD flags are not affected by reset.

The external reset pin is active low, and a good external reset circuit can protect the system from operating under abnormal power conditions.

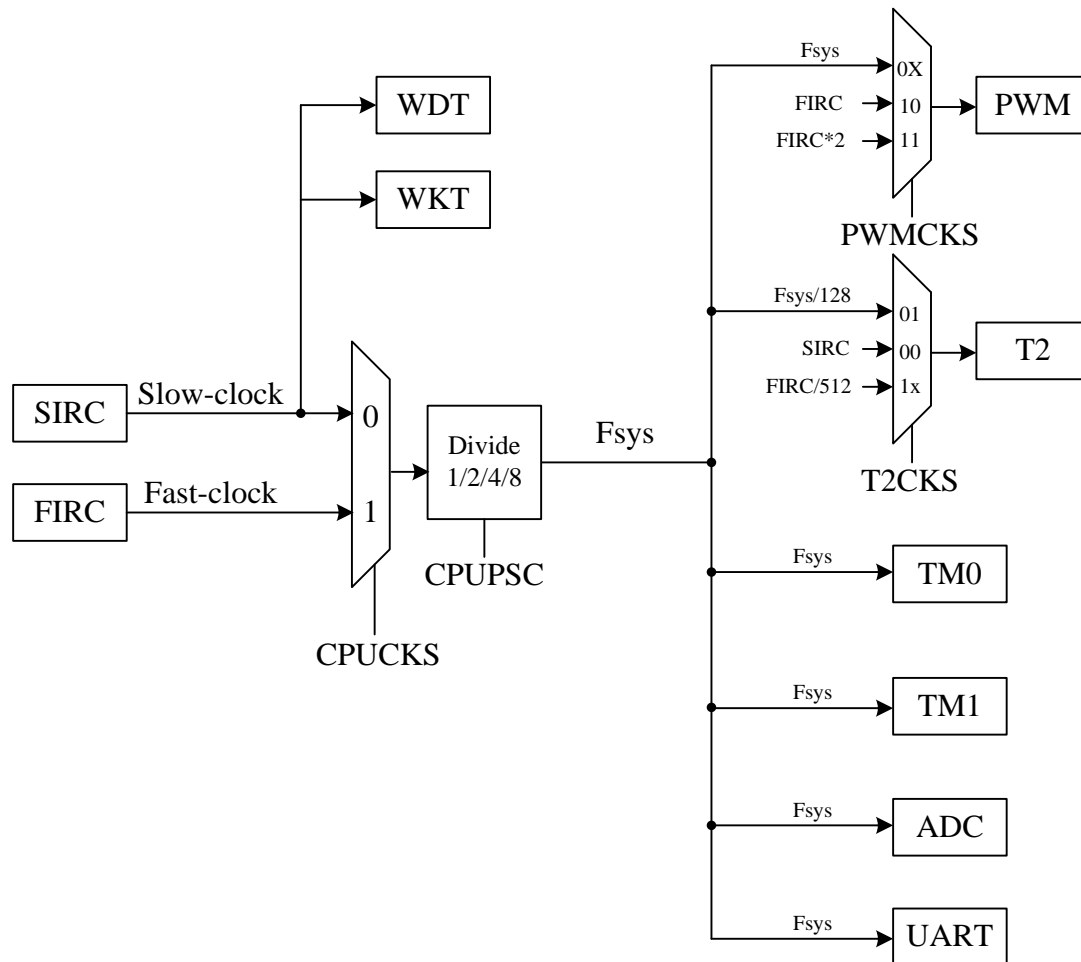


## **2.4 Watchdog Timer Reset (WDTR)**

The Watchdog Timer reset can be disabled or enabled by SYSCFG. Set WDT\_PSC to define the period during which WDT reset occurs. WDT reset counter can be cleared by device Reset or CLRWDTR instruction. WDT reset also set all the control registers to their default value. The TO/PD flags are not affected by WDT resets.

### 3 Clock Circuitry and Operation Mode

The device is designed using a dual clock system. The System clock (Fsys) can be selected from Slow-clock or Fast-clock. The clock sources of each peripheral are shown in the figure below.



**Clock Source Diagram**

**FAST Mode:**

In this mode, the chip is executed using Fast-clock as System clock (Fsys).

If you want to enter SLOW mode, first set SLOWSTP to 0, then set CPUCKS to 0, so that the device will switch to SLOW mode.

Example: Switch FAST mode to SLOW mode

BCX	SLOWSTP	; Slow-clock enable
BCX	CPUCKS	; Fsys = Slow-clock

**SLOW Mode:**

In this mode, the chip is executed using Slow-clock as System clock (Fsys).

After the device power-on or reset, System clock will enter SLOW mode.

The user can choose to turn the Fast-clock on or off through the FASTSTP bit

If you want to enter Fast mode, first set FASTSTP to 0, then set CPUCKS to 1, so that the device will switch to FAST mode.

Example: Switch SLOW mode to FAST mode

BCX	FASTSTP	; Fast-clock enable
BSX	CPUCKS	; Fsys = Fast-clock

**IDLE Mode:**

If the device goes to sleep with SLOWSTP=0 or WKTIE=1 or WDTE=3, the Slow-clock will still continue to oscillate during sleep, which is called IDLE mode.

Users can put the device to sleep by executing the SLEEP instruction. In the sleep state, Fast-Clock must stop oscillating.

After a fast device wakes up from sleep state, it will return to the mode before waking up.

Example: Switch FAST/SLOW mode to IDLE mode.

BCX	SLOWSTP	; Slow-clock will keep running after executing the SLEEP instruction
SLEEP		; executing SLEEP instruction

**STOP Mode:**

If the device goes to sleep with SLOWSTP=0 and WKTIE=0 and WDTE=0,1,2, Slow-clock will stop oscillating. In this case, the CPU is the most power-saving, which is called STOP mode.

Users can put the device to sleep by executing the SLEEP instruction. In the sleep state, Fast-Clock must stop oscillating.

After a fast device wakes up from sleep state, it will return to the mode before waking up.

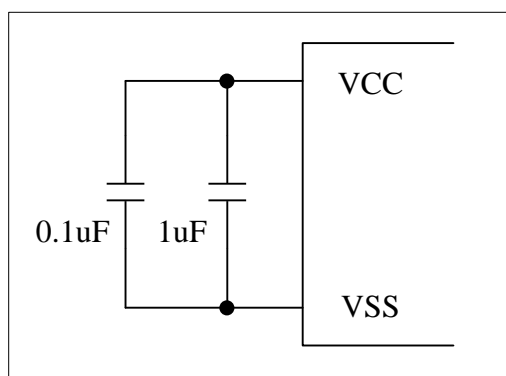
Example: Switch FAST/SLOW mode to STOP mode.

BSX	SLOWSTP	; Slow-clock will stop after executing the SLEEP instruction
MOVLW	00000000b	; close WKT
MOVWX	INTIE	
SLEEP		; executing SLEEP instruction

Mode	System clock (Fsys)	Built-in Fast RC oscillator (FIRC)	Built-in Slow RC oscillator (SIRC)
FAST mode	Fast-clock	V	V
SLOW mode	Slow-clock	by FASTSTP	V
IDLE mode	X	X	V
STOP mode	X	X	X

**Clock Mode Table**
**Power Supply Bypass Capacitor:**

Since power supply noise will degrade the performance of the internal clock oscillator, it is recommended to place the power supply bypass capacitors 1 uF and 0.1 uF close to the VCC/VSS pin, which can improve the stability of the clock and the entire system.


**Power supply bypass capacitor**

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.3 **WKTIE:** Wakeup Timer interrupt enable and Wakeup Timer enable  
 0: disable  
 1: enable

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	—	—	—	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	—	—	—	R/W	R/W	R/W	R/W	
Reset	—	—	—	0	1	0	1	1

- 0Fh.4     **SLOWSTP**: Stop Slow-clock after execute SLEEP instruction  
           0: Slow-clock keeps running after execute SLEEP instruction  
           1: Slow-clock stops running after execute SLEEP instruction
- 0Fh.3     **FASTSTP**: Fast-clock stop  
           0: Fast-clock is running  
           1: Fast-clock stops running
- 0Fh.2     **CPUCKS**: System clock source selection  
           0: Slow-clock  
           1: Fast-clock
- 0Fh.1~0   **CPUPSC**: System clock source prescaler. System clock source  
           00: divided by 8  
           01: divided by 4  
           10: divided by 2  
           11: divided by 1

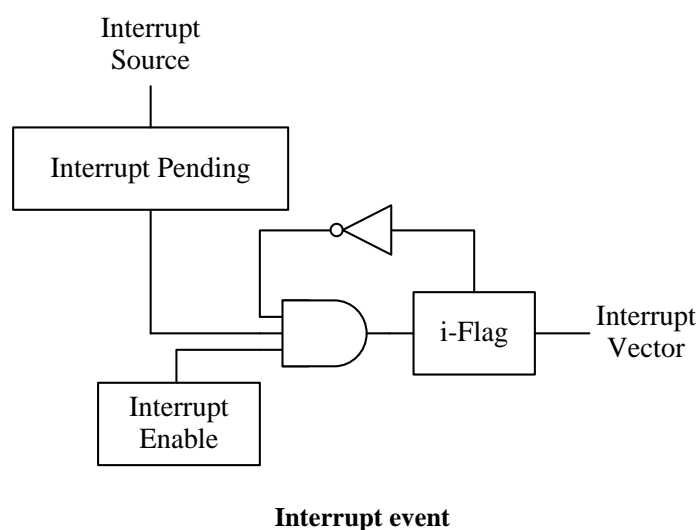


## 4 Interrupt

The Chip has 1 level, 1 vector and 11 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual pending flag, no matter its enable control bit is 0 or 1.

If the corresponding interrupt enable bit has been set, it would trigger CPU to service the interrupt. CPU accepts interrupt at the end of current executed instruction cycle. In the meanwhile, a “LCALL 004” instruction is inserted to CPU, and i-flag is set to prevent recursive interrupt nesting.

The i-flag is cleared in the instruction after the “RETI” instruction. That is, at least one instruction in main program is executed before service the pending interrupt. The interrupt event is level triggered. F/W must clear the interrupt event register while serving the interrupt routine.



◇ Example: Setup INT1 (PA1) interrupt request with rising edge trigger

```

ORG      000h                ; Reset Vector
LGOTO    START                ; Goto user program address

ORG      004h                ; All interrupt vector
LGOTO    INT                   ; If INT1 (PA1) input occurred rising edge

ORG      005h
START:
MOVLW    0000xxxxb
MOVWX    PAMOD10              ; Select INT1 Pin Mode as mode 0000b
                                   ; Open drain output low or input with Pull-up

MOVLW    xxxxxx1xb
MOVWX    PAD                   ; Release INT1, it becomes Schmitt-trigger
                                   ; input with input pull-up resistor

MOVLW    xx1xxxxxb
MOVWX    OPTION                ; Set INT1 interrupt trigger as rising edge
MOVLW    1111101b
MOVWX    INTIF                 ; Clear INT1 interrupt request flag
MOVLW    00000010b
MOVWX    INTIE                 ; Enable INT1 interrupt

MAIN:
...
LGOTO    MAIN

INT:
MOVWX    20h                   ; Store W data to SRAM 20h
MOVXW    STATUS                ; Get STATUS data
MOVWX    21h                   ; Store STATUS data to SRAM 21h

BTXSC    INT1IF                ; Check INT1IF bit
LCALL    INT1_SUB              ; INT1IF = 1, jump to INT1 interrupt service routine
...

EXIT_INT:
MOVXW    21h                   ; Get SRAM 21h data
MOVWX    STATUS                ; Restore STATUS data
MOVXW    20h                   ; Restore W data
RETI                          ; Return from interrupt

INT1_SUB:                      ; INT1 interrupt service routine
...
MOVLW    1111101b
MOVWX    INTIF                 ; Clear INT1 interrupt request flag
RET

```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- 0Bh.7 **ADCIE:** ADC interrupt enable  
0: disable  
1: enable
- 0Bh.6 **T2IE:** T2 interrupt enable  
0: disable  
1: enable
- 0Bh.5 **TM1IE:** Timer1 interrupt enable  
0: disable  
1: enable
- 0Bh.4 **TM0IE:** Timer0 interrupt enable  
0: disable  
1: enable
- 0Bh.3 **WKTIE:** Wakeup Timer interrupt enable and Wakeup Timer enable  
0: disable  
1: enable
- 0Bh.2 **INT2IE:** INT2 interrupt enable  
0: disable  
1: enable
- 0Bh.1 **INT1IE:** INT1 interrupt enable  
0: disable  
1: enable
- 0Bh.0 **INT0IE:** INT0 interrupt enable  
0: disable  
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- 0Ch.7 **ADCIF:** ADC interrupt event pending flag  
This bit is set by H/W after ADC end of conversion, write 0 to this bit will clear this flag
- 0Ch.6 **T2IF:** T2 interrupt event pending flag  
This bit is set by H/W while T2 overflows, write 0 to this bit will clear this flag
- 0Ch.5 **TM1IF:** Timer1 interrupt event pending flag  
This bit is set by H/W while Timer1 overflows, write 0 to this bit will clear this flag
- 0Ch.4 **TM0IF:** Timer0 interrupt event pending flag  
This bit is set by H/W while Timer0 overflows, write 0 to this bit will clear this flag
- 0Ch.3 **WKTIF:** Wakeup Timer interrupt event pending flag  
This bit is set by H/W while Wakeup Timer is timeout, write 0 to this bit will clear this flag
- 0Ch.2 **INT2IF:** INT2 pin falling interrupt pending flag  
This bit is set by H/W at INT2 pin's falling edge, write 0 to this bit will clear this flag
- 0Ch.1 **INT1IF:** INT1 pin falling/rising interrupt pending flag  
This bit is set by H/W at INT1 pin's falling/rising edge, write 0 to this bit will clear this flag
- 0Ch.0 **INT0IF:** INT0 pin falling/rising interrupt pending flag  
This bit is set by H/W at INT0 pin's falling/rising edge, write 0 to this bit will clear this flag

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	EA	UARTIE	—	—	—	—	PWMIE	LVDIE
R/W	R/W	R/W	—	—	—	—	R/W	R/W
Reset	1	0	—	—	—	—	0	0

0Dh.7 **EA:** Global interrupt enable

0: disable

1: enable

0Dh.6 **UARTIE:** UART interrupt enable

0: disable

1: enable

0Dh.1 **PWMIE:** PWM interrupt enable

0: disable

1: enable

0Dh.0 **LVDIE:** LVD interrupt enable

0: disable

1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	—	UARTIF	—	—	—	—	PWMIF	LVDIF
R/W	—	R	—	—	—	—	R/W	R/W
Reset	—	0	—	—	—	—	0	0

0Eh.6 **UARTIF:** UART interrupt event pending flag

This bit is set by H/W when UART transmission/reception is completed, write 0 to TI/RI flag will clear this flag

0Eh.1 **PWMIF:** PWM interrupt event pending flag

This bit is set by H/W after PWM period counter roll over, write 0 to this bit will clear this flag

0Eh.0 **LVDIF:** LVD interrupt event pending flag

This bit is set by H/W after  $V_{CC} < V_{LVD}$ , write 0 to this bit will clear this flag

## 5 I/O Port

### 5.1 GPIO (PA0-PA7, PB0-PB6)

The chip has various pin modes and their functions are shown in the table below. In this table, Pin Mode is defined by PAMODx, PBMODx, Pin Data is defined by PAD, PBD.

Pin Mode	Pin Data	Description	Digital Output	Digital Input	Pin Wakeup
<b>0000b</b>	0	Output low	ON	OFF	OFF
	1	Input with Pull-up resistor	OFF	ON	OFF
<b>0001b</b>	0	Output low	ON	OFF	OFF
	1	Input high impedance	OFF	ON	OFF
<b>0010b</b>	0	Output Low	ON	OFF	OFF
	1	Output High	ON	OFF	OFF
<b>0011b</b>	X	Analog signal ADC / OP0P / OP1P / OP2P / VREXT / VBGO	OFF	OFF	OFF
<b>0100b</b>	0	Output low	ON	OFF	OFF
	1	Input with Pull-down resistor	OFF	ON	OFF
<b>0101b</b>	0	Output low	ON	OFF	OFF
	1	Input high impedance	OFF	ON	OFF
<b>0110b</b>	0	Output Low	ON	OFF	OFF
	1	Output High	ON	OFF	OFF
<b>0111b</b>	X	PWM output	ON	OFF	OFF
<b>1000b</b>	0	Output low	ON	OFF	OFF
	1	Input with Pull-up resistor	OFF	ON	ON
<b>1001b</b>	0	Output low	ON	OFF	OFF
	1	Input high impedance	OFF	ON	ON
<b>1010b</b>	0	Output Low	ON	OFF	OFF
	1	Output High	ON	OFF	OFF
<b>1011b</b>	X	Reserved	OFF	OFF	OFF-
<b>1100b</b>	0	Output low	ON	OFF	OFF
	1	Input with Pull-down resistor	OFF	ON	ON
<b>1101b</b>	0	Output low	ON	OFF	OFF
	1	Input high impedance	OFF	ON	ON
<b>1110b</b>	0	Output Low	ON	OFF	OFF
	1	Output High	ON	OFF	OFF
<b>1111b</b>	X	Analog output 1/2 V <sub>CC</sub> (1/2 bias)	OFF	OFF	OFF

**GPIO Function Table**

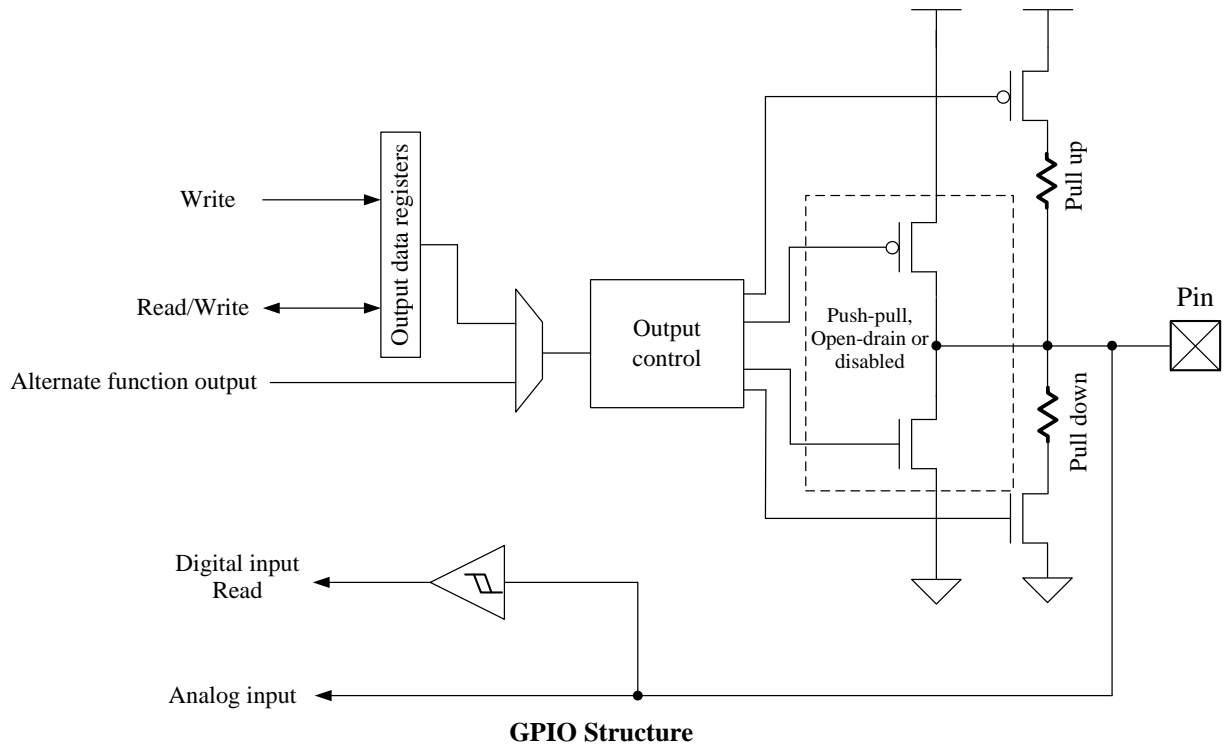
Mode 1 turns off the digital output and enables the digital input. Mode 3 turns off the digital output and turns off the digital input. Both Mode 1 and Mode 3 can be used for analog signals. However, because mode 1 enables digital input, it may consume more power when used with analog signals. It is recommended that analog signals such as ADC / OP0P / OP1P / OP2P / VREXT / VBGO use Mode 3.

The default setting of all general IO (GPIO) is mode 1. PB0~PB2 are not high impedance because they have analog signal output inside by default. PB0 defaults to output 60mV of DAC0, PB1 defaults to

output 2.4V of DAC1, and the PB2 default output is VR voltage value (3V). Please pay attention to whether there is any conflict with the external circuit when using it. If PB0~PB2 are to be used for analog signals, it is recommended to switch to mode 3 after power-on to save power consumption.

Pin Name	PAxMOD / PBxMOD		
	0011b (Analog in/out)	0111b (Digital output)	1111b (Analog output)
PA0	ADC0	PWM0N	1/2 bias
PA1	ADC1	PWM0P	1/2 bias
PA2	ADC2	PWM1	1/2 bias
PA3	ADC3/OP2O	PWM2	1/2 bias
PA4	ADC4	PWM0N	1/2 bias
PA5	ADC5	PWM0P	1/2 bias
PA6	ADC6	PWM1	1/2 bias
PA7	ADC7	PWM2	1/2 bias
PB0	ADC8/OP0P/DAC0O	PWM0N	1/2 bias
PB1	ADC9/OP1P/DAC1O	PWM0P	1/2 bias
PB2	ADC10/VREXT	PWM1	1/2 bias
PB3	ADC11	PWM2	1/2 bias
PB4	ADC12	PWM0N	1/2 bias
PB5	ADC13	PWM0P	1/2 bias
PB6	ADC14/OP2P	PWM1	1/2 bias

**GPIO Special Function Table**



85h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD10	PA1MOD				PA0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

86h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD32	PA3MOD				PA2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

87h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD54	PA5MOD				PA4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

88h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD76	PA7MOD				PA6MOD			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	1

88h.7~4 **PA7MOD ~ PA0MOD:** PA7~PA0 Pin Mode Control

88h.3~0 0000: Open drain or digital input with pull-up

87h.7~4 0001: Open drain or digital input

87h.3~0 0010: CMOS Push-pull

86h.7~4 0011: Analog input/output

86h.3~0 0100: Open drain or digital input with pull-down

85h.7~4 0101: Open drain or digital input

85h.3~0 0110: CMOS Push-pull

0111: Alternate function output

1000: Open drain or digital input with pull-up and pin-changed wakeup

1001: Open drain or digital input and pin-changed wakeup

1010: CMOS Push-pull

1011: Reserved

1100: Open drain or digital input with pull-down and pin-changed wakeup

1101: Open drain or digital input and pin-changed wakeup

1110: CMOS Push-pull

1111: 1/2 V<sub>CC</sub> (1/2 bias)

8Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD10	PB1MOD				PB0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD32	PB3MOD				PB2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD54	PB5MOD				PB4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD76	-				PB6MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Fh.3~0 **PB6MOD ~ PB0MOD**: PB6~PB0 Pin Mode Control

8Eh.7~4 0000: Open drain or digital input with pull-up

8Eh.3~0 0001: Open drain or digital input

8Dh.7~4 0010: CMOS Push-pull

8Dh.3~0 0011: Analog input

8Ch.7~4 0100: Open drain or digital input with pull-down

8Ch.3~0 0101: Open drain or digital input

0110: CMOS Push-pull

0111: Alternate function output

1000: Open drain or digital input with pull-up and pin-changed wakeup

1001: Open drain or digital input and pin-changed wakeup

1010: CMOS Push-pull

1011: Reserved

1100: Open drain or digital input with pull-down and pin-changed wakeup

1101: Open drain or digital input and pin-changed wakeup

1110: CMOS Push-pull

1111: 1/2 V<sub>CC</sub> (1/2 bias)



05h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

05h.7~0 **PAD**: PA7~PA0 pin data

06h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBD	-	PBD						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

06h.6~0 **PBD**: PB6~PB0 pin data

105h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL2	GPR2		-	-	-	HSINK	ROMODS	
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	-	0	1	1	1	1

105h.2 **HSINK**: All GPIO high sink current selection  
 0: low sink current  
 1: high sink current

## 5.2 OP0N / OPO / OP1N / VREXT

The function is fixed as analog signal, so no pin mode setting is required.

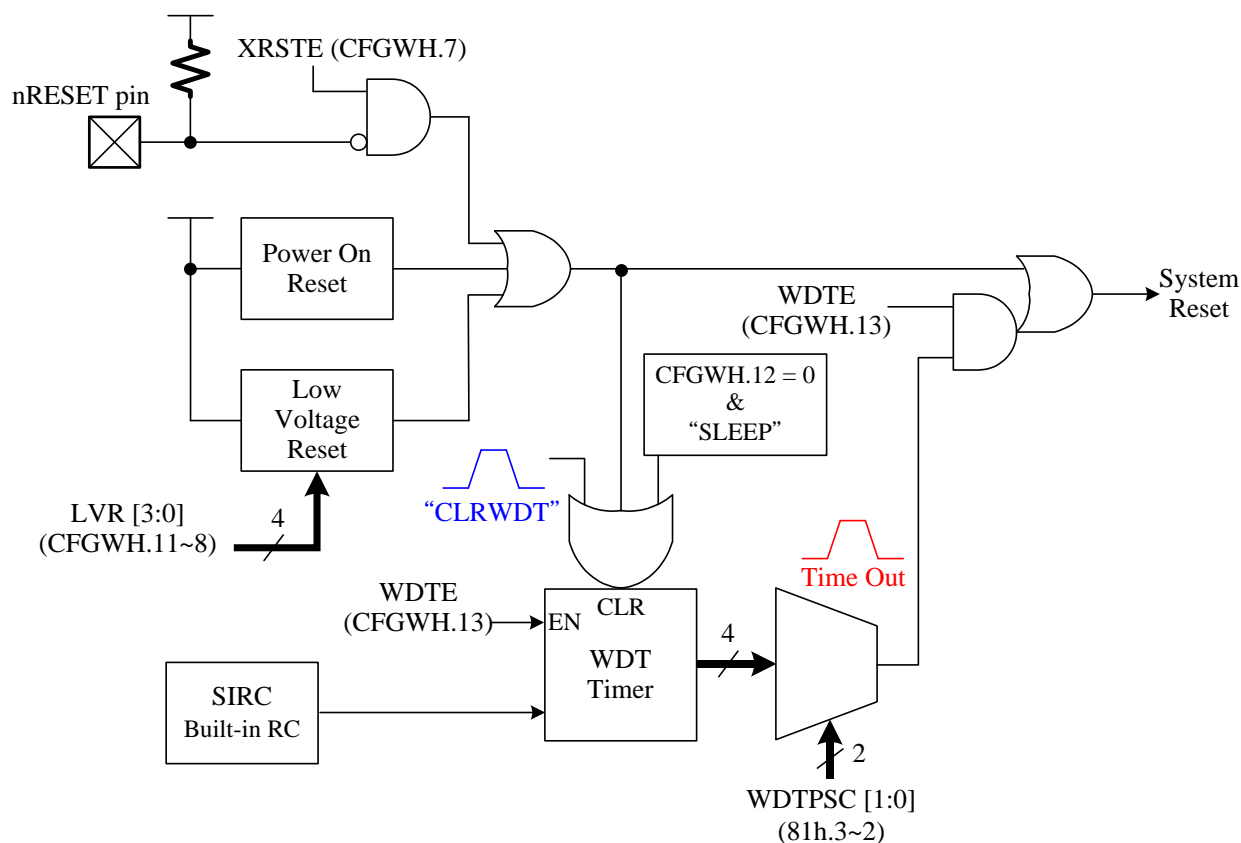
Pin Name	Analog Signal
OP0N	OP0N/ADC15
OPO	OPO
OP1N	OP1N/ADC16
VREXT	VR

## 6 Peripheral Functional Block

### 6.1 Watchdog Timer (WDT)

The watchdog (WDT) uses the internal SIRC oscillator and has a separate counter. The overflow period of the WDT can be selected by the prescaler WDTOSC.

The WDT timer is cleared by the CLRWDT instruction. If the watchdog is enabled and the watchdog counter overflows, the WDT will generate a chip reset signal.



**WDT Block Diagram**

The WDT's behavior in different Mode is shown as below table.

Mode	SYSCFG[13:12]		WDT
	WDTE[1]	WDTE[0]	
Normal Mode	0	0	Stop
	0	1	Stop
	1	0	Run
	1	1	Run
Power-down Mode (SLEEP)	0	0	Stop
	0	1	Stop
	1	0	Stop
	1	1	Run

Watchdog clear is controlled by CLRWDT instruction.

◇ Example: Clear watchdog timer by CLRWDT instruction.

```

MAIN:    ...                               ; Execute program.
          CLRWDT                           ; Execute CLRWDT instruction.
          ...
          LGOTO    MAIN
    
```

◇ Example: Setup WDT time.

```

          MOVLW    00000111b
          MOVWX    OPTION                   ; Select WDT reset period
          ...
    
```

03h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

03h.4 **TO:** WDT time out flag, read-only  
 0: after Power On Reset or CLRWDT / SLEEP instructions  
 1: WDT time out occurs

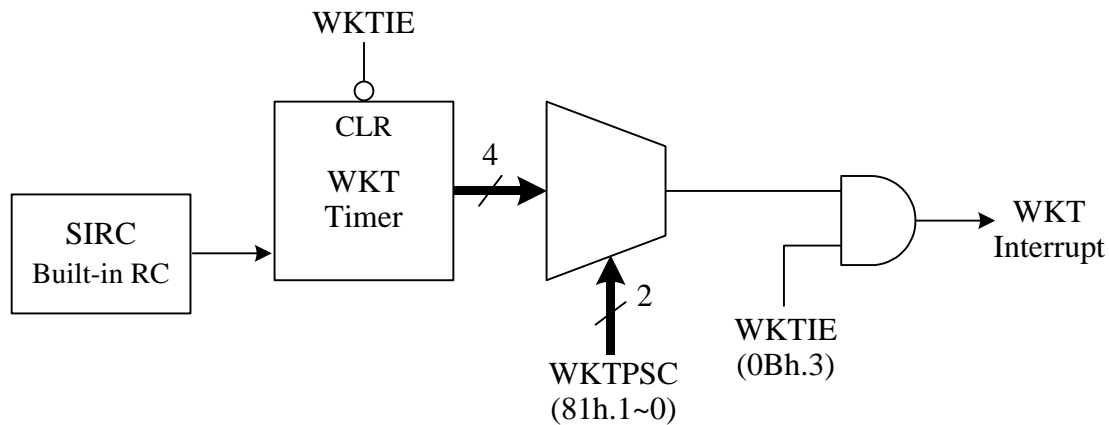
81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EDG	—	WDT PSC		WKTPSC	
R/W	R/W	R/W	R/W	—	R/W		R/W	
Reset	0	0	0	—	1	1	1	1

81h.3~2 **WDT PSC:** WDT period (@V<sub>CC</sub>=5V)  
 00: 221 ms  
 01: 443 ms  
 10: 1771 ms  
 11: 3542 ms

## 6.2 Wakeup Timer (WKT)

The wakeup timer (WKT) uses the internal SIRC oscillator and has a separate counter. The overflow period of WKT can be selected by the prescaler WKT<sub>PSC</sub>.

The WKT timer is an interval timer, and a WKT interrupt flag (WKTIF) will be generated when the WKT timer times out. The WKT timer is cleared/stopped by WKTIE=0. When WKTIE=1 is set, the WKT timer will keep counting regardless of the CPU operation mode.



WKT Block Diagram

◇ Example: Set WKT period and interrupt function.

```

MOVLW    00000110b
MOVWXX   OPTION           ; Select WKT period
MOVLW    11110111b
MOVWXX   INTIF            ; Clear WKT interrupt flag by using byte operation
                                ; Don't use bit operation "BCX WKTIF" to clear

BSX      WKTIE            ; Enable WKT interrupt function
  
```

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.3 **WKTIF:** Wakeup Timer interrupt event pending flag

This bit is set by H/W while Wakeup Timer is timeout, write 0 to this bit will clear this flag

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.3 **WKTIE:** Wakeup Timer interrupt enable and Wakeup Timer enable

0: disable

1: enable

81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EDG	–	WDTPSC		WKTTPSC	
R/W	R/W	R/W	R/W	–	R/W		R/W	
Reset	0	0	0	–	1	1	1	1

81h.1~0 **WKTTPSC:** WKT period (@V<sub>CC</sub>=5V)

00: 28 ms

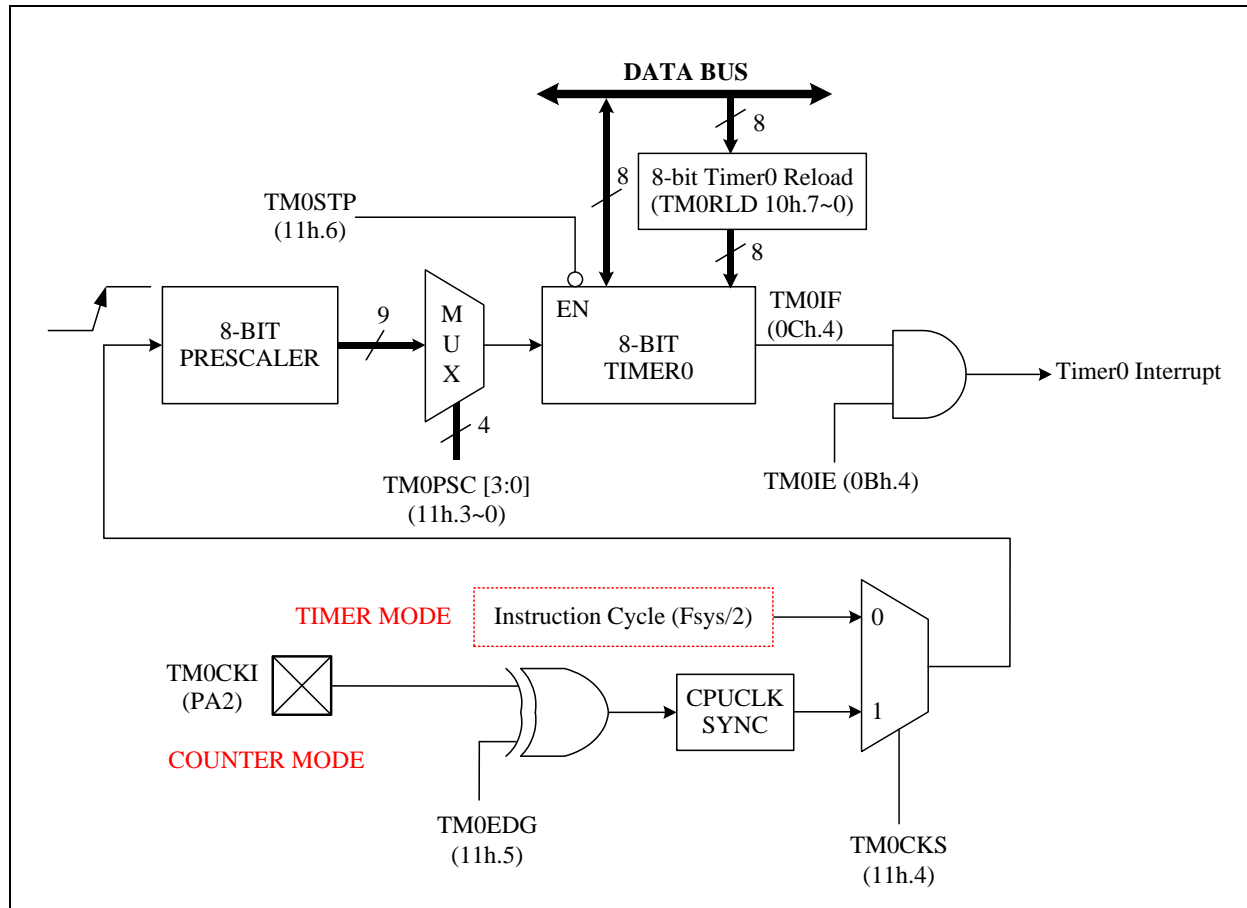
01: 55 ms

10: 111 ms

11: 221 ms

### 6.3 Timer0

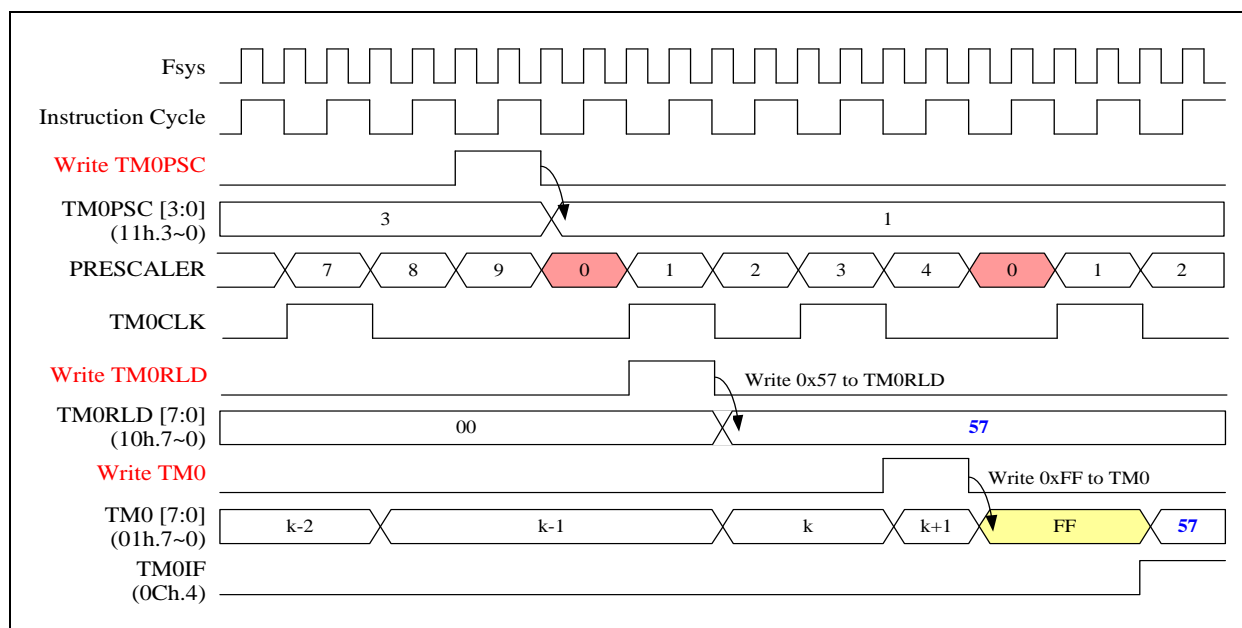
The TM0 (01h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be  $F_{sys}/2$  or TM0I (PA2) rising/falling input. The Timer0 increase rate is determined by "Timer0 Pre-Scale" (TM0PSC) register. The Timer0 always generates TM0IF (0Ch.4) when its count rolls over. It generates Timer0 Interrupt if TM0IE (0Bh.4) is set. Timer0 can be stopped counting if the TM0STP (11h.6) bit is set.



Timer0 Block Diagram

The following timing diagram describes the Timer0 works in pure Timer mode.

When the Timer0 prescaler (TM0PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer0 count. TM0CLK is the internal signal that causes the Timer0 to increase by 1 at the end of TM0CLK. TM0WR is also the internal signal that indicates the Timer0 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer0 counts from FFh to TM0RLD, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set.



**Timer0 works in Timer mode (TM0CKS=0)**

The equation of TM0 interrupt time value is as following:

$$\text{TM0 interrupt interval cycle time} = F_{\text{sys}} / 2 / \text{TM0PSC} / (256 - \text{TM0RLD})$$

◇ Example: Setup Timer0 work in Timer mode, if  $F_{\text{sys}} = 8 \text{ MHz}$

; Setup Timer0 clock source and divider

MOVLW	00x <u>00101</u> b	; TM0CKS = 0, Timer0 clock is instruction cycle
MOVWX	TM0CTL	; TM0PSC = 0101b, divided by 32

; Setup Timer0 reload data

MOVLW	80h	
MOVWX	TM0RLD	; Set Timer0 reload data = 128

; Setup Timer0

BSX	TM0STP	; Timer0 stops counting
CLR X	TM0	; Clear Timer0 content

; Enable Timer0 and interrupt function

MOVLW	111 <u>0</u> 1111b	
MOVWX	INTIF	; Clear Timer0 request interrupt flag
BSX	TM0IE	; Enable Timer0 interrupt function
BCX	TM0STP	; Enable Timer0 counting

$$\text{Timer0 interrupt frequency} = F_{\text{sys}} / 2 / \text{TM0PSC} / (256 - \text{TM0RLD}),$$

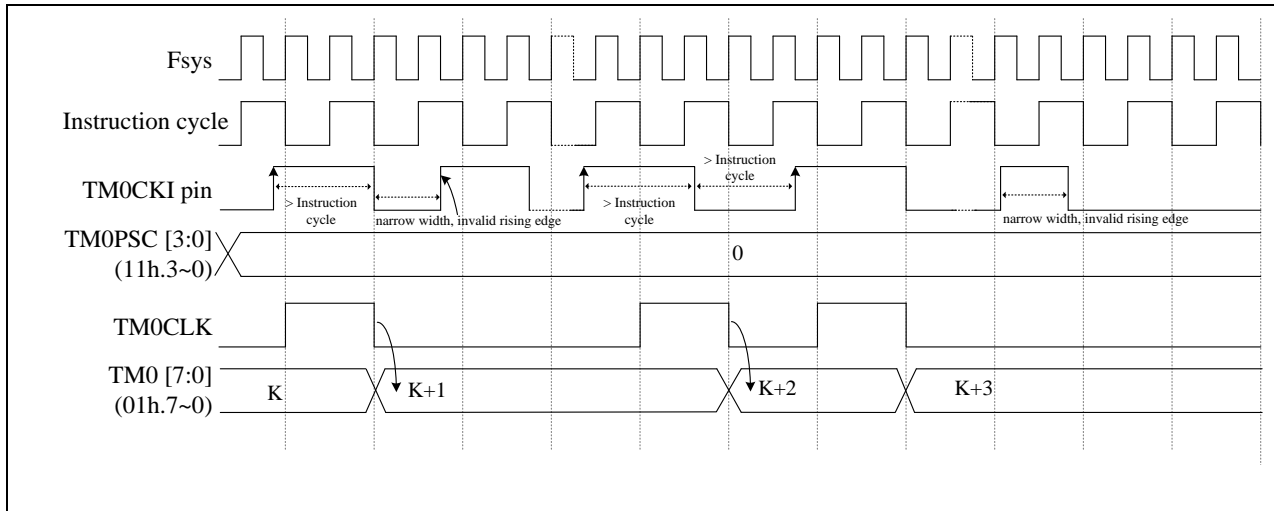
$$F_{\text{sys}} = 8 \text{ MHz}, \text{TM0PSC} = \text{div } 32, \text{TM0RLD} = 128$$

$$\text{Timer0 interrupt frequency} = 8 \text{ MHz} / 2 / 32 / (256 - 128) = 0.976 \text{ KHz}$$



The following timing diagram describes the Timer0 works in Counter mode.

If TM0CKS=1 then Timer0 counter source clock is from TM0I pin. TM0I signal is synchronized by instruction cycle ( $F_{sys}/2$ ) that means the high/low time durations of TM0I must be longer than one instruction cycle time ( $F_{sys}/2$ ) to guarantee each TM0I's change will be detected correctly by the synchronizer.



**Timer0 works in Counter mode for TM0I (TM0EDG=0), TM0CKS=1**

◇ Example: Setup TM0 work in Counter mode and clock source from TM0I pin (PA2)

; Setup Timer0 clock source and divider

MOVLW	00 <b>110000</b> B	; TM0EDG = 1, counting edge is falling edge
MOVWX	TM0CTL	; TM0CKS = 1, Timer0 clock is TM0I
		; TM0PSC = 0000b, divided by 1

; Setup Timer0

BSX	TM0STP	; Timer0 stops counting
CLRXL	TM0	; Clear Timer0 content

; Enable Timer0 and read Timer0 counter

BCX	TM0STP	; Enable Timer0 counting
...		
BSX	TM0STP	; Timer0 stops counting
MOVXW	TM0	; Read Timer0 content

01h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0	TM0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

01h.7~0 **TM0**: Timer0 content

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.4 **TM0IE**: Timer0 interrupt enable

0: disable  
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.4 **TM0IF**: Timer0 interrupt event pending flag

This bit is set by H/W while Timer0 overflows, write 0 to this bit will clear this flag

10h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0RLD	TM0RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

10h.7~0 **TM0RLD**: Timer0 reload data

11h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	—	TM0STP	TM0EDG	TM0CKS	TM0PSC			
R/W	—	R/W	R/W	R/W	R/W			
Reset	—	0	0	0	0	0	0	0

11h.6 **TM0STP**: Stop Timer0

0: Timer0 runs  
1: Timer0 stops

11h.5 **TM0EDG**: Timer0 prescaler counting edge for TM0I pin

0: rising edge  
1: falling edge

11h.4 **TM0CKS**: Timer0 prescaler clock source

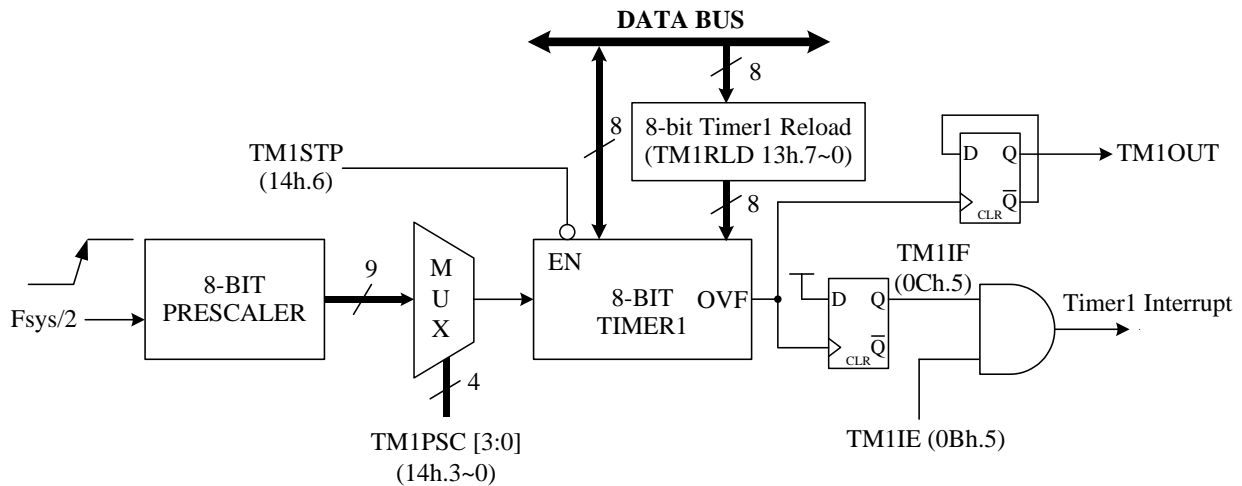
0: Fsys/2  
1: TM0I pin (PA2 pin)

11h.3~0 **TM0PSC**: Timer0 prescaler. Timer0 prescaler clock source divided by

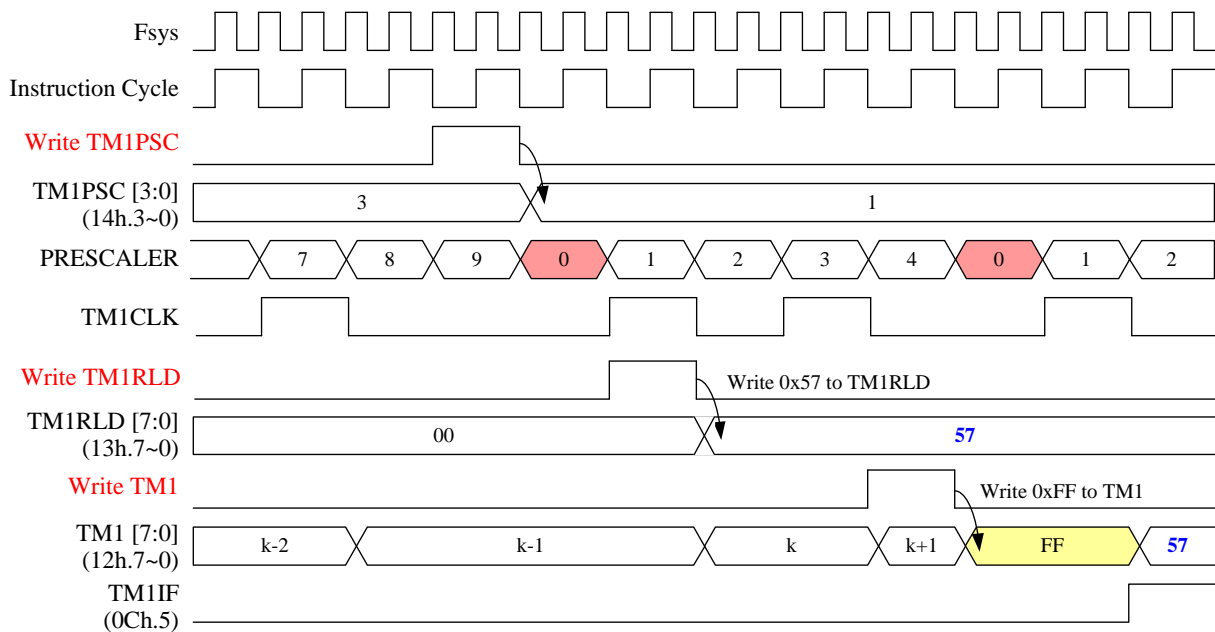
0000: 1	0001: 2	0010: 4	0011: 8
0100: 16	0101: 32	0110: 64	0111: 128
1xxx: 256			

## 6.4 Timer1

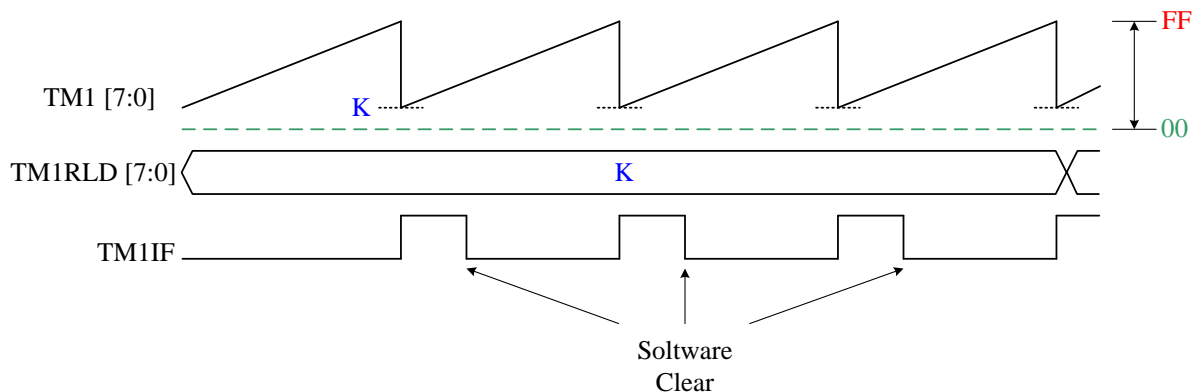
The TM1 (12h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock ( $F_{sys}/2$ ). The Timer1 increase rate is determined by TM1PSC register. It generates Timer1 interrupt if the TM1IE bit is set. Timer1 can be stopped counting if the TM1STP bit is set. TM1OUT is an output signal that toggles when Timer1 overflow.



**Timer1 Block Diagram**



**Timer1 Timing Diagram**



**Timer1 Reload Diagram**

◇ Example: CPU is running in SLOW mode,  $F_{sys} = \text{Slow-clock} / \text{CPUPSC} = 90 \text{ KHz} / 2 = 45 \text{ KHz}$

; Setup Timer1 clock source and divider

```
MOVLW    00000011b
MOVWX    TM1CTL           ; TM1PSC = 0011b, divided by 8
```

; Setup Timer1 reload data

```
MOVLW    FFh
MOVWX    TM1RLD           ; Set Timer1 reload data = 255
```

; Setup Timer1

```
BSX      TM1STP           ; Timer1 stops counting
CLR      TM1              ; Clear Timer1 content
```

; Enable Timer1 and interrupt function

```
MOVLW    11011111b
MOVWX    INTIF            ; Clear Timer1 request interrupt flag
BSX      TM1IE            ; Enable Timer1 interrupt function
BCX      TM1STP           ; Enable Timer1 counting
```

Timer1 interrupt frequency =  $F_{sys} / 2 / \text{TM1PSC} / (256 - \text{TM1RLD})$ ,

$F_{sys} = 45 \text{ KHz}$ , TM1PSC = div 8, TM1RLD = 255

Timer1 interrupt frequency =  $45 \text{ KHz} / 2 / 8 / (256 - 255) = 2.81 \text{ KHz}$

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.5 **TM1IE**: Timer1 interrupt enable  
0: disable  
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.5 **TM1IF**: Timer1 interrupt event pending flag  
This bit is set by H/W while Timer1 overflows, write 0 to this bit will clear this flag

12h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1	TM1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

12h.7~0 **TM1**: Timer1 content

13h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1RLD	TM1RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

13h.7~0 **TM1RLD**: Timer1 reload data

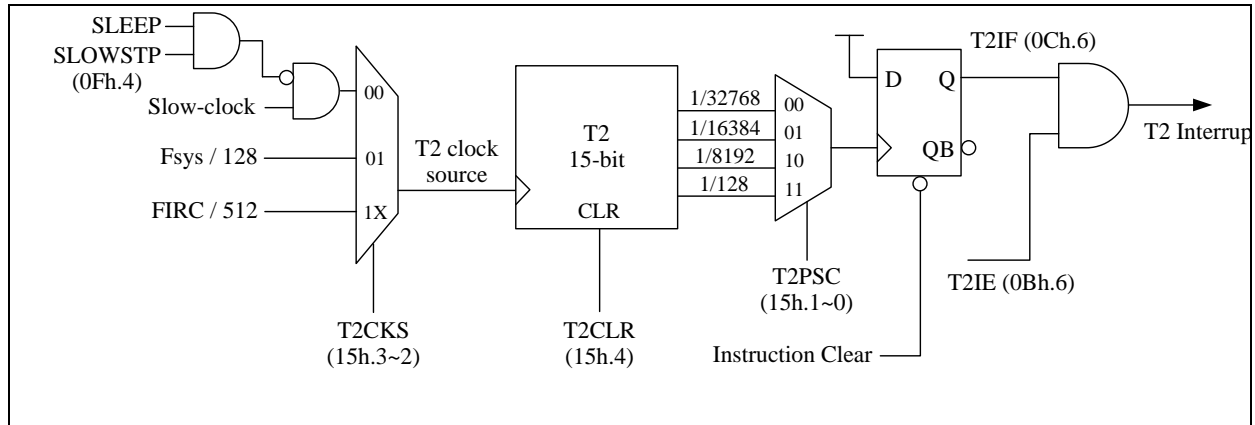
14h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1CTL	—	TM1STP	—	—	TM1PSC			
R/W	—	R/W	—	—	R/W			
Reset	—	0	—	—	0	0	0	0

14h.6 **TM1STP**: Stop Timer1  
0: Timer1 runs  
1: Timer1 stops

14h.3~0 **TM1PSC**: Timer1 prescaler. Timer1 prescaler clock source divided by  
0000: 1      0001: 2      0010: 4      0011: 8  
0100: 16      0101: 32      0110: 64      0111: 128  
1xxx: 256

## 6.5 T2:15-bit Timer

The T2 is a 15-bit counter and the clock sources are from Slow-clock, Fsys/128, or FIRC/512. It is used to generate time base interrupt and T2 counter block clock. The T2 content cannot be read by instructions. It generates interrupt flag T2IF (0Ch.6) with the clock divided by 32768/16384/8192/128 depends on T2PSC[1:0] (15h.1~0) register bits. The following figure shows the block diagram of T2.



**T2 Block Diagram**

Example: CPU is running at FAST mode, Fsys = 9.216 MHz

; Setup T2 clock source and divider

MOVLW	00000 <b>101</b> b	; T2CKS(15h.3~2) = 1, T2 clock source is Fsys/128
MOVWX	T2CTL	; T2PSC(15h.1~0) = 1, divided by 16384
BSX	T2CLR	; T2CLR = 1, clear T2 counter

; Enable T2 interrupt function

MOVLW	<b>10</b> 111111b	; Clear T2 request interrupt flag
MOVWX	INTIF	; Enable T2 interrupt function
BSX	T2IE	; T2CLR = 0, Enable T2 counting
BCX	T2CLR	

T2 clock source is Fsys / 128 = 9.216 MHz / 128 = 72000 Hz, T2PSC = 16384

T2 frequency = 72000 Hz / 16384 = 4.395 Hz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.6 **T2IE:** T2 interrupt enable  
0: disable  
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.6 **T2IF:** T2 interrupt event pending flag  
This bit is set by H/W while T2 overflows, write 0 to this bit will clear this flag

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	SCKTYP	FCKTYPE	—	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
Reset	0	0	—	0	1	0	1	1

0Fh.4 **SLOWSTP:** Stop Slow-clock after execute SLEEP instruction  
0: Slow-clock keeps running after execute SLEEP instruction  
1: Slow-clock stops running after execute SLEEP instruction

15h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CTL	—	—	—	T2CLR	T2CKS		T2PSC	
R/W	—	—	—	R/W	R/W		R/W	
Reset	—	—	—	0	0	0	0	0

15h.4 **T2CLR:** Clear and stop T2  
0: T2 runs  
1: T2 clear and stops

15h.3~2 **T2CKS:** T2 clock source selection  
00: Slow-clock  
01: Fsys/128  
1x: FIRC/512

15h.1~0 **T2PSC:** T2 prescaler. T2 clock source divided by  
00: 32768  
01: 16384  
10: 8192  
11: 128

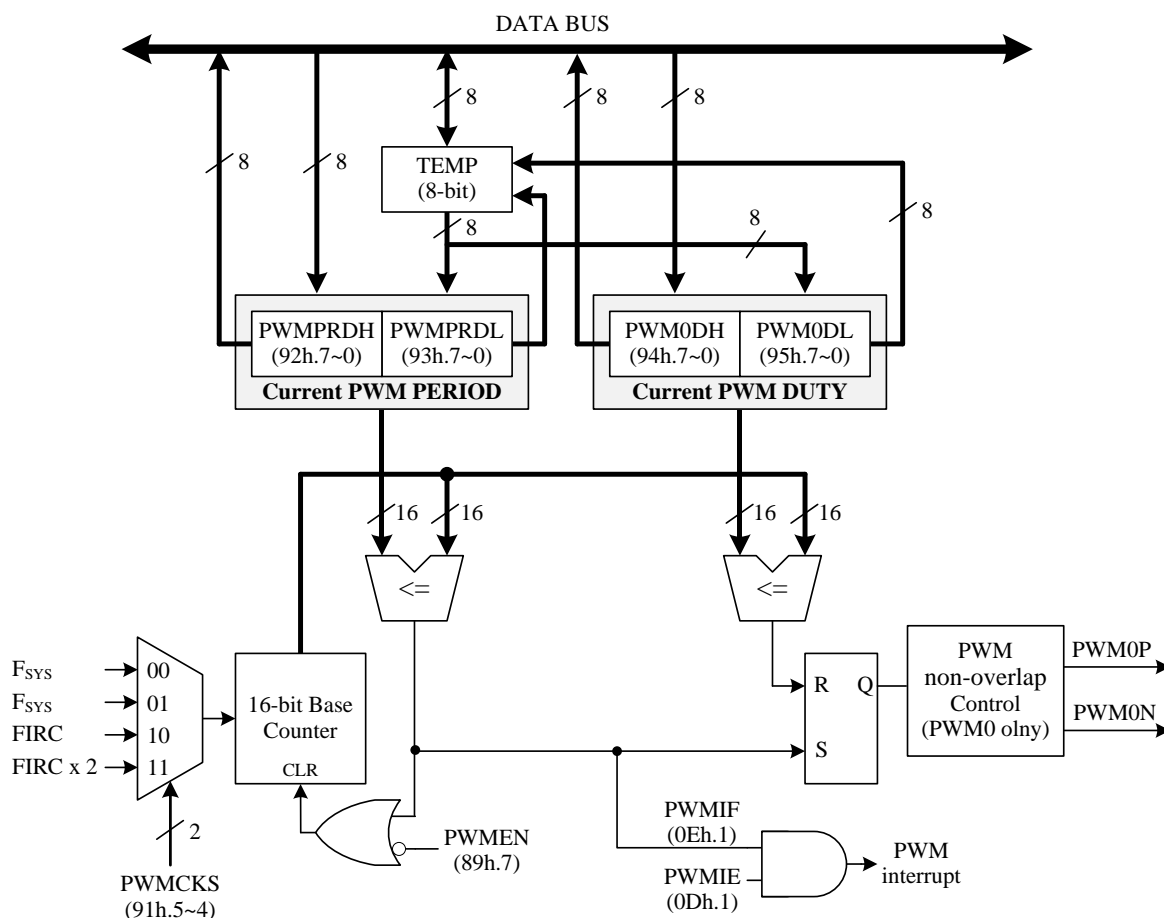
## 6.6 PWM

There are 3 PWMs in this chip. PWM0~PWM2 have independent 16-bit duty control register, and share a set of 16-bit period register. The PWM can generate varies frequency waveform with 65536 duty resolution on the basis of the PWM clock. The PWM clock can select Fsys, FIRC (18.432 MHz), or FIRC\*2 (36.864MHz) as its clock source, the FIRC and FIRC\*2 frequencies used here will not be affected by FRCPS (SYSCFG.5). The following takes PWM0 as an example for description.

The 16-bit PWMPRD, PWM0D registers both have a low byte and high byte structure. The high bytes can be directly accessed, but the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to notes is that data transfer to and from the 8-bit buffer and its related low byte only takes place when write or read operation to its corresponding high bytes is executed. **Briefly speaking, write low byte first and then high byte; read high byte first and then low byte.**

If PWMEN is cleared, the PWM0~2 will be cleared and stopped, otherwise the PWM0~2 remain running. The PWM0 structure is shown as follow. The PWM0 duty cycle can be changed by writing to PWM0DH and PWM0DL. The PWM0 output signal resets to a low level whenever the 16-bit base counter matches the 16-bit PWM0 duty register {PWM0DH, PWM0DL}. The PWM0 period can be set by writing the period value to the PWMPRDH and PWMPRDL registers. After writing the PWM0DH or PWMPRDH register, H/W will update PWM period and duty immediately. PWM0~2 share an interrupt flag, and an interrupt flag is generated at the end of the period.

Only PWM0 has dead-zone control, and is divided into PWM0P and PWM0N outputs, and the remaining PWM1~PWM2 have no non-overlap control. User can use pin mode setting to output PWM to the corresponding IO pin, refer to Chapter 5 for more information on pin settings.

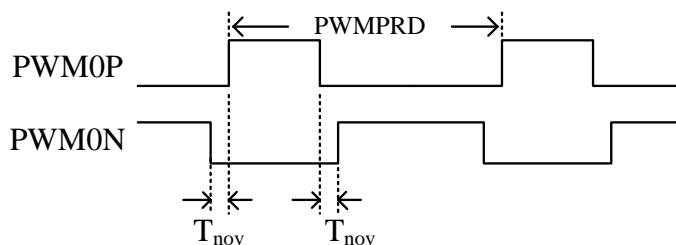




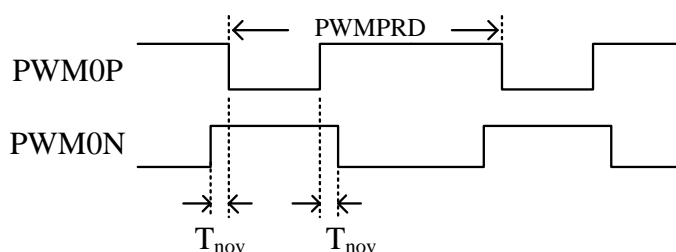
### PWM0 Block Diagram

Only PWM0 can be output via PWM0P and PWM0N with four different modes. The edges of the PWM pulse can be separated with 16 different time non-overlap clocks intervals ( $T_{nov}$ ). The width of  $T_{nov}$  can be selected by PWM0DZ (89h.3~0) within 0~15 PWM clock. The default output form is Mode0. The waveforms of the four output modes are shown below.

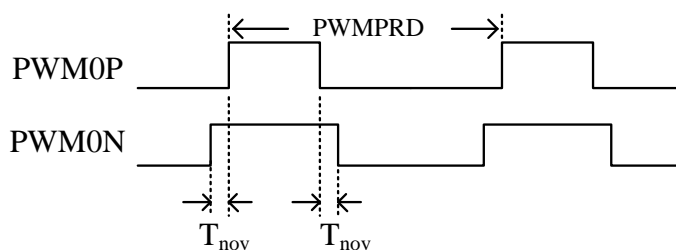
Mode0



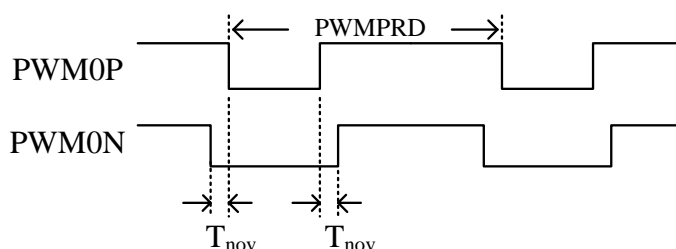
Mode1



Mode2



Mode3



### PWM0 Waveform Modes

◇ Example:

; Setup Pin mode

```

MOV LW    01110111b      ;
MOV WX    PAMOD54          ; PA4 引脚作为 PWM0N
                                ; PA5 引脚作为 PWM0P

```

; Setup PWM0 clock source select

```

MOV LW    xx10xxxxb      ;
MOV WX    OPTION2          ; FIRC 18.432 MHz as PWM clock source

```

; Setup PWM0 period and duty setting

```

MOV LW    FFh
MOV WX    PWMPRDL          ; write sequence: PWMPRDL then PWMPRDH
MOV LW    7Fh
MOV WX    PWMPRDH          ; Set PWM period = 7FFFh

MOV LW    00h
MOV WX    PWM0DL           ; write sequence: PWM0DL then PWM0DH
MOV LW    40h
MOV WX    PWM0DH           ; Set PWM0 duty = 4000h

```

; Setup PWM0 enable and dead zone control

```

MOV LW    10000000b      ; 89h.7 = 1, PWM0 enable
MOV WX    PWMCTL           ; 89h.5~4 = 0, PWM0 Mode0 output
                                ; 89h.3~0 = 0, PWM0 dead zone output disable

```

Example:

PWM0 clock source = FIRC 18.432 MHz, PWM period = 7FFFh, PWM duty = 4000h

PWM0 output frequency =  $18.432 \text{ MHz} / (\text{period} + 1) = 18.432 \text{ MHz} / 32768 = 563 \text{ Hz}$ .

PWM0 output duty =  $\text{duty} / (\text{period} + 1) = 50 \%$ .

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	EA	UARTIE	—	—	—	—	PWMIE	LVDIE
R/W	R/W	R/W	—	—	—	—	R/W	R/W
Reset	1	0	—	—	—	—	0	0

0Dh.1 **PWMIE:** PWM interrupt enable  
 0: disable  
 1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	—	UARTIF	—	—	—	—	PWMIF	LVDIF
R/W	—	R	—	—	—	—	R/W	R/W
Reset	—	0	—	—	—	—	0	0

0Eh.1 **PWMIF:** PWM interrupt event pending flag  
 This bit is set by H/W after PWM period counter roll over, write 0 to this bit will clear this flag

89h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWMEN	—	PWM0OM		PWM0DZ			
R/W	R/W	—	R/W		R/W			
Reset	0	—	0	0	0	0	0	0

89h.7 **PWMEN:** PWM0~2 enable  
 0: disable  
 1: enable

89h.5~4 **PWM0OM:** PWM0 output mode selection  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3

89h.3~0 **PWM0DZ:** PWM0 non-overlap control  
 0000: no non-overlap  
 0001: non-overlap width are 1 PWM clock cycle  
 0010: non-overlap width are 2 PWM clock cycles  
 ...  
 1111: non-overlap width are 15 PWM clock cycles

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION2	—	—	PWMCKS		—	INT2SEL	INT1SEL	INT0SEL
R/W	—	—	R/W		—	R/W	R/W	R/W
Reset	—	—	0	0	—	0	0	0

91h.5~4 **PWMCKS:** PWM clock source selection  
 00: Fsys  
 01: Fsys  
 10: FIRC (18.432 MHz)  
 11: FIRC x 2 (36.864 MHz)

92h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDH	PWMPRDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

92h.7~0 **PWMPRDH**: PWM0~2 period high byte  
 write sequence: PWMPRDL then PWMPRDH  
 read sequence: PWMPRDH then PWMPRDL

93h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDL	PWMPRDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

93h.7~0 **PWMPRDL**: PWM0~2 period low byte  
 write sequence: PWMPRDL then PWMPRDH  
 read sequence: PWMPRDH then PWMPRDL

94h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	PWM0DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

94h.7~0 **PWM0DH**: PWM0 duty high byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

95h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

95h.7~0 **PWM0DL**: PWM0 duty low byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

96h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DH	PWM1DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

96h.7~0 **PWM1DH**: PWM1 duty high byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

97h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DL	PWM1DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

97h.7~0 **PWM1DL**: PWM1 duty low byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DH	PWM2DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

98h.7~0 **PWM2DH:** PWM2 duty high byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

99h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DL	PWM2DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

99h.7~0 **PWM2DL:** PWM2 duty low byte  
 write sequence: PWMxDL then PWMxDH  
 read sequence: PWMxDH then PWMxDL

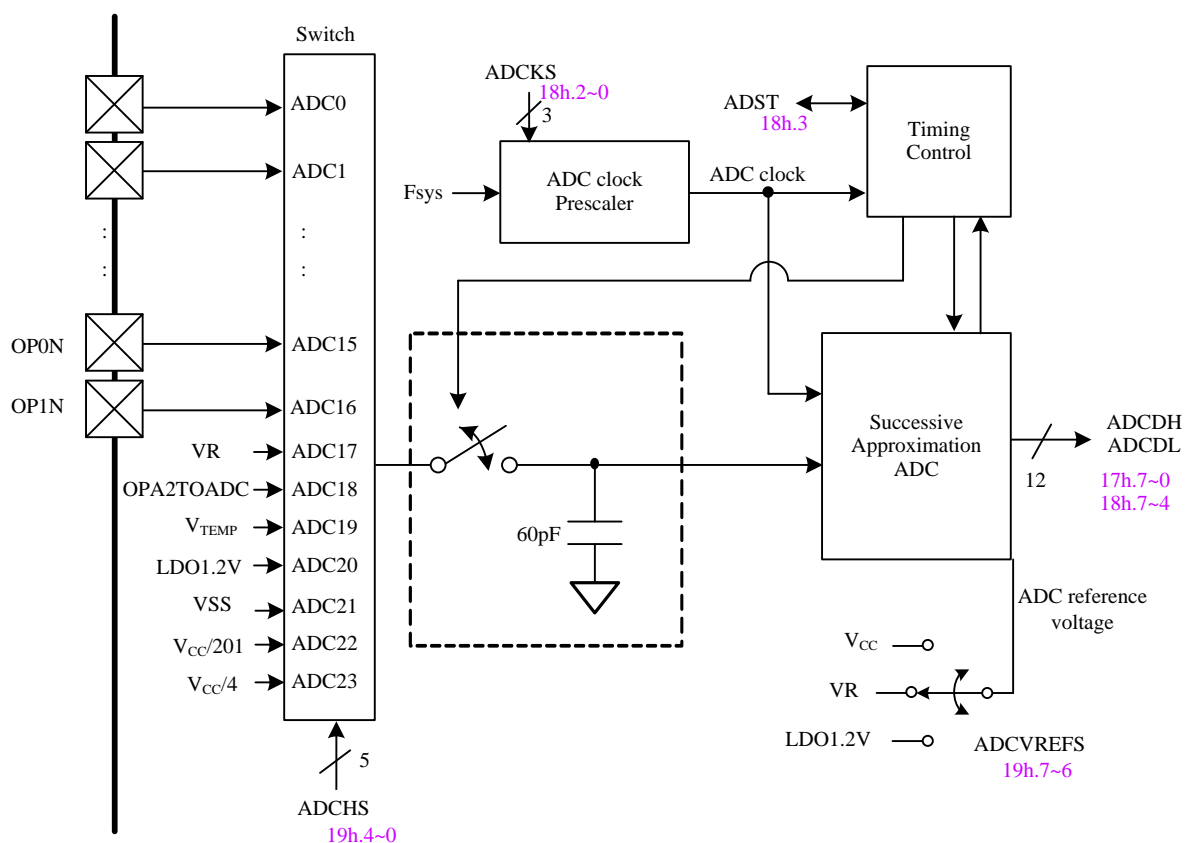
## 6.7 Analog-to-Digital Converter (ADC)

This 12-bit ADC (analog-to-digital converter) consists of an analog input multiplexer with 17 external channels, control registers, clock generator, 12-bit successive approximation register, and output data register.

The user needs to set ADCHS to select the input channel of the ADC, and set ADCKS to select a suitable ADC clock frequency. When the ADC uses a low-voltage reference voltage source, for example, when ADCVREF is set to LDO1.2V, the ADC clock frequency must be less than 0.5MHz. Please refer to the "Electrical Characteristics" section for further information on ADC clock frequency.

The ADC reference voltage source can be configured as  $V_{CC}$ , VR or LDO1.2V through ADVREFS. When the reference voltage source switches to VR or LDO1.2V, an internal preparation stabilization time of 200us is required.

The user starts ADC conversion by setting the ADST control bit. ADST remains 1 during conversion. After the conversion is completed, H/W will automatically clear the ADST bit. When the ADC conversion is completed, ADST will return to 0, so the user can know whether the ADC conversion has been completed by reading ADST.



When using GPIOs such as ADC0~ADC14 as the input pins of the ADC, the corresponding pin mode should be set to 0011b. When using ADC15 and ADC16, they are connected to OP0N and OP1N respectively, and there is no need to set the pin mode.

The input pin of the ADC can also select an internal reference voltage. This device has a variety of internal reference voltages to choose from, including VR, OPA2TOADC,  $V_{TEMP}$ , LDO1.2V...etc. The  $V_{TEMP}$  voltage is used for temperature sensing, and the control items are the registers SVBIAS and SBFIN. OPA2TOADC is the output of OPA2. The VR voltage source is determined by the LDO3VPD register. When LDO3VPD is 0, the source of VR is the LDO3V voltage. When LDO3VPD is 1, the

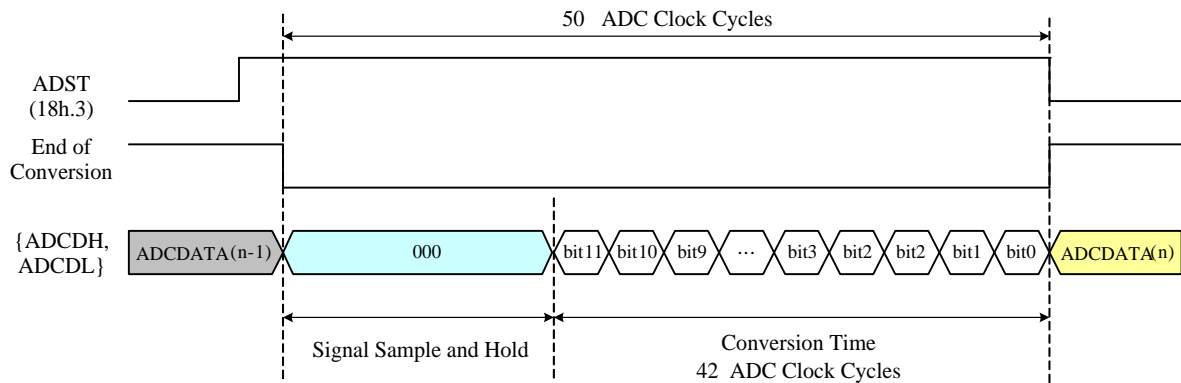
source of VR is the VRXT PAD external voltage. For more information about VR, LDO3V, LDO1.2V and OPA2 please refer to the "Battery Charging Module" chapter.

The following are the relevant usage restrictions:

When the ADC uses VR and LDO3VPD is 0, the VREXT PAD must be connected to a capacitor to stabilize the voltage.

When the ADC uses VR and LDO3VPD is 1, the ADC cannot use LDO1.2V at this time.

When the ADC uses LDO1.2V, LDO3VPD must be 0 at this time.



Example:

[  $F_{sys} = FIRC/2 = 9.216 \text{ MHz}$  ]

ADC clock frequency = 1.152 MHz, ADC channel = ADC2 (PA2).

```

MOVLW    xxxx0011b          ; ADC2 (PA2) as ADC input
MOVW     PAMOD32

MOVLW    00000100b          ; ADCKS = Fsys/16, ADC clock = 1.152 MHz
MOVW     ADCTL

MOVLW    01x00010b           ; ADC reference voltage select VR
MOVW     ADCTL2              ; ADC input channel select ADC2

BSX      ADST                ; 18h.3 (ADST), ADC start conversion.
    
```

WAIT\_ADC:

```

BTXSC    ADST                ; Wait ADC conversion finish.
LGOTO    WAIT_ADC

MOVXW    ADCDH              ; Read ADC output data bit 11~4
MOVXW    ADCTL              ; Read ADC output data bit 3~0
...
    
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.7    **ADCIE**: ADC interrupt enable  
 0: disable  
 1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.7 **ADCIF**: ADC interrupt event pending flag

This bit is set by H/W after ADC end of conversion, write 0 to this bit will clear this flag

17h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCDH	ADCDH							
R/W	R							
Reset	–	–	–	–	–	–	–	–

17h.7~0 **ADCDH**: ADC output data bit 11~4

18h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL	ADCDL				ADST	ADCKS		
R/W	R				R/W	R/W		
Reset	–	–	–	–	0	0	0	0

18h.7~4 **ADCDL**: ADC output data bit 3~0

18h.3 **ADST**: ADC start bit.

0: H/W clear after end of conversion

1: ADC start conversion

18h.2~0 **ADCKS**: ADC clock frequency selection:

000: Fsys/256    100: Fsys/16

001: Fsys/128    101: Fsys/8

010: Fsys/64    110: Fsys/4

011: Fsys/32    111: Fsys/2

19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL2	ADVREFS		–	ADCHS				
R/W	R/W		–	R/W				
Reset	0	1	–	1	1	1	1	1

19h.7~6 **ADVREFS**: ADC reference voltage selection.

00: ADC reference voltage is V<sub>CC</sub>

01: ADC reference voltage is VR

10: ADC reference voltage is LDO1.2V

11: Reserved

19h.3~0 **ADCHS**: ADC channel selection

00000: ADC0 (PA0)    01000: ADC8 (PB0)    10000: ADC16 (OP1N)

00001: ADC1 (PA1)    01001: ADC9 (PB1)    10001: VR

00010: ADC2 (PA2)    01010: ADC10 (PB2)    10010: OPA2TOADC

00011: ADC3 (PA3)    01011: ADC11 (PB3)    10011: V<sub>TEMP</sub>

00100: ADC4 (PA4)    01100: ADC12 (PB4)    10100: LDO1.2V

00101: ADC5 (PA5)    01101: ADC13 (PB5)    10101: VSS

00110: ADC6 (PA6)    01110: ADC14 (PB6)    10110: V<sub>CC</sub>/201

00111: ADC7 (PA7)    01111: ADC15 (OP0N)    10111: V<sub>CC</sub>/4

115h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL	SVBIAS	SBFIN		SVBGT	VBGTOE	LDO3VPD		
R/W	R/W	R/W	R/W	R/W	R/W	R/W		



Reset	0	1	1	0	0	0	-	-
-------	---	---	---	---	---	---	---	---

- 115h.7     **SVBIAS:** Reference voltage of VT0 selection  
                  0:V<sub>CC</sub>   1:VR
- SBFIN:** ADC V<sub>TEMP</sub> selection  
                  00:V<sub>TEMP</sub> = VT0 (diode type)
- 115h.6~5   01:V<sub>TEMP</sub> = VT1 (BJT type)  
                  10:V<sub>TEMP</sub> = VBG1.2V  
                  11:V<sub>TEMP</sub> is disabled

## 6.8 UART

The device has a full-duplex or half-duplex (single-wire mode) asynchronous serial interface. The user can choose 8 or 9-bit data transmission. The UART baud rate is set by the user and can support up to 115200. When the UART data transmission is completed or the reception is completed, the UART interrupt can be triggered

When the UART1W bit is set to 1, the UART will operate in single-wire mode. In single-wire mode, only a single RXTX pin is used to transmit and receive data. When the RXEN bit is set to 0, the RXTX pin is used as a transmit pin, and when the RXEN bit is set to 1, the RXTX pin is used as a receive pin.

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	—	UARTIF	—	—	—	—	PWMIF	LVDIF
R/W	—	R	—	—	—	—	R/W	R/W
Reset	—	0	—	—	—	—	0	0

0Eh.6 **UARTIF:** UART interrupt event pending flag

This bit is set by H/W when UART transmission/reception is completed, write 0 to TI/RI flag will clear this flag

195h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SCON	UART9	—	RIMASK	RXEN	TX8	RX8	TI	RI
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

195h.7 **UART9:** Number of data transfer bits selection

0: 8-bit data transfer

1: 9-bit data transfer

195h.5 **RIMASK:** Receive flag mask control

If this bit is set, receive flag function is disable when RX8 is 0.

195h.4 **RXEN:** Receive function enable.

0:

When UART1W is set low, the RXTX pin is disabled.

When UART1W is set high, the RXTX pin is used as the TX pin.

1:

When UART1W is set low, the RXTX pin is used as the RX pin.

When UART1W is set high, the RXTX pin is used as the RX pin.

195h.3 **TX8:** (This bit is only valid when UART9=1)

This bit is the 9th value to be transmitted by TX pin.

195h.2 **RX8:** (This bit is only valid when UART9=1)

This bit is the 9th value received by RX pin.

195h.1 **TI:** Transmit flag.

Set by H/W when transmission is completed. SW needs to write 0 to clear it, writing 1 does nothing.

When TI=1 or RI=1, UARTIF will be set.

195h.0 **RI:** Receive flag.

Set by H/W when reception is completed. SW needs to write 0 to clear it, writing 1 does nothing.

When TI=1 or RI=1, UARTIF will be set.

196h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SBUF	SBUF							
R/W	R/W							
Reset	—	—	—	—	—	—	—	—

196h.7~0 **SBUF**: Serial UART transmit/receive data.

197h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UARTCTL	UARTBRP							
R/W	R/W							
Reset	—	—	—	—	—	—	—	—

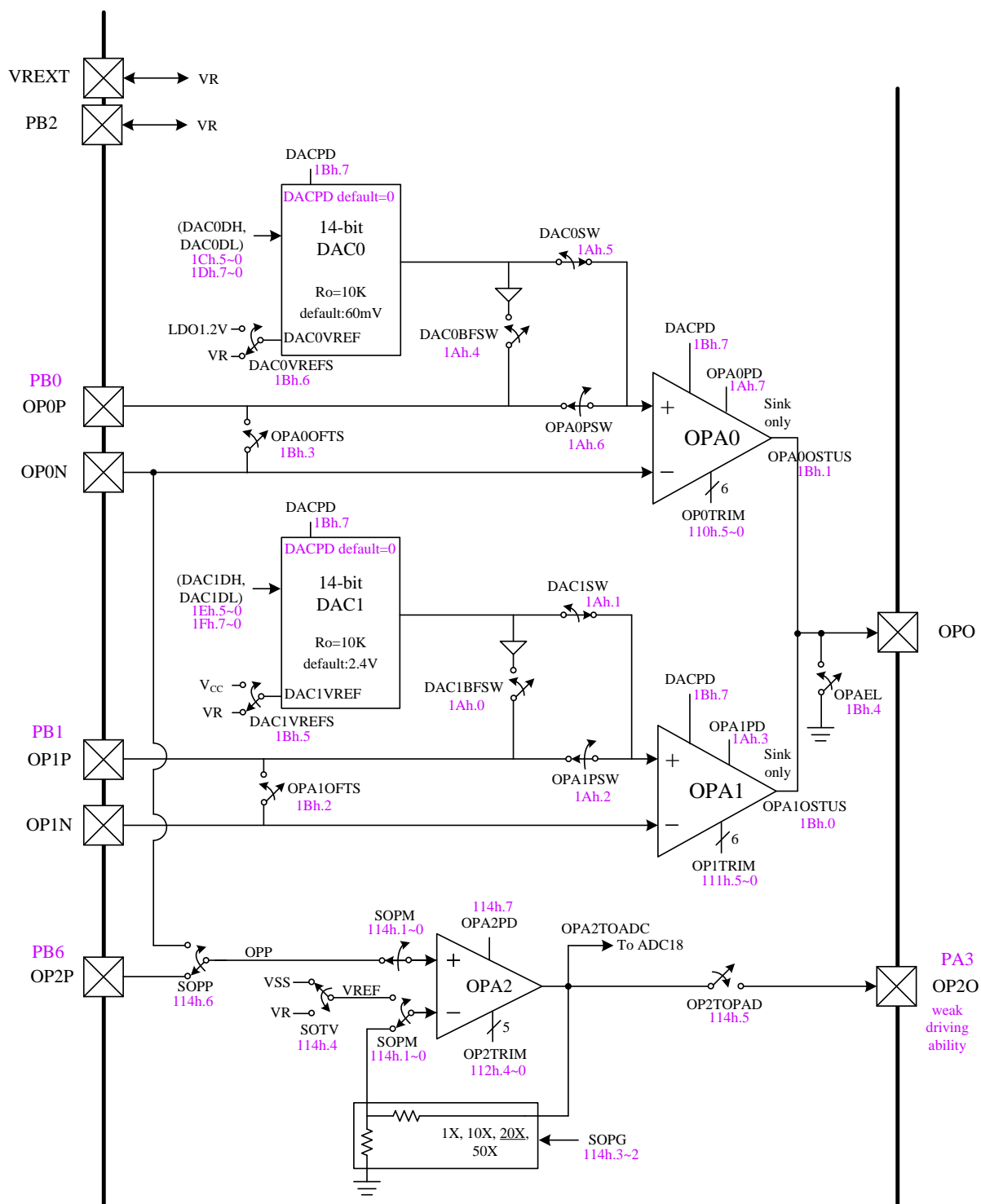
197h.7~0 **UARTBRP**: UART Baud Rate Prescaler.  
 UART Baud Rate =  $F_{sys}/16/UARTBRP$

198h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UARTCTL2	UART1W	—	TXS2	TXS1	TXS0	RXTXS2	RXTXS1	RXTXS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

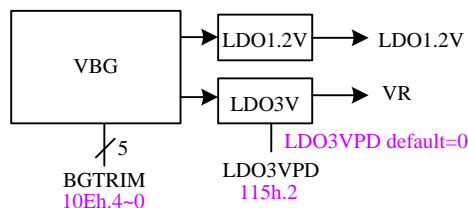
- 198h.7 **UART1W**: Single-wire mode enable  
 0: full-duplex communication  
 1: half-duplex communication (single-wire mode)
- 198h.5 **TXS2**: TX pin selection 2  
 0:disable 1:PB3 is used as TX pin
- 198h.4 **TXS1**: TX pin selection 1  
 0:disable 1:PB1 is used as TX pin
- 198h.3 **TXS0**: TX pin selection 0  
 0:disable 1:PA1 is used as TX pin
- 198h.2 **RXTXS2**: RXTX pin selection 2.  
 0: disable  
 1: If UART1W is set low, PB0 is used as RX pin.  
 If UART1W is set high and RXEN is set low, PB0 is used as TX pin.  
 If UART1W is set high and RXEN is set high, PB0 is used as RX pin.
- 198h.1 **RXTXS1**: RXTX pin selection 1.  
 0: disable  
 1: If UART1W is set low, PA5 is used as RX pin.  
 If UART1W is set high and RXEN is set low, PA5 is used as TX pin.  
 If UART1W is set high and RXEN is set high, PA5 is used as RX pin.
- 198h.0 **RXTXS0**: RXTX pin selection 0.  
 0: disable  
 1: If UART1W is set low, PA0 is used as RX pin.  
 If UART1W is set high and RXEN is set low, PA0 is used as TX pin.  
 If UART1W is set high and RXEN is set high, PA0 is used as RX pin.

## 6.9 Battery Charge Module (BCM) - DAC/Comparator/Amplifier

This chip has a battery charging module, which consists of two DACs, two comparators, and an amplifier. Among them, DAC0~1 provides the comparison reference value to the comparator OPA0~1. OPA0 is used for battery charging constant current, OPA1 is used for battery charging constant voltage, and OPA2 is used to amplify the battery charging current. The details are as shown below.



Battery Charge Module (SOPAN=0)



**Internal Reference Voltage Module**

Only one of PB2 and VREXT will be selected during packaging to provide an external interface for the chip's VR voltage.

When the LDO3VPD register is 0, the internal reference voltage LDO3V is enabled, and the VR voltage value is provided by the internal LDO3V. When the LDO3VPD register is 1, the internal reference voltage LDO3V is turned off, and the VR voltage value is provided by the outside of the chip (that is, PB2 or VREXT). The LDO3V output, internal VR signal, PB2 and VREXT PAD are connected to each other as shown in the block diagram. The VR signal is also provided to the ADC module for use. When the BCM module and the ADC module need to use VR at the same time, PB2 or VREXT must choose an external capacitor to stabilize the voltage.

After power-on, the default LDO3VPD register is 0, and LDO3V outputs a 3V reference voltage. At this time, the internal VR signal is equal to the LDO3V voltage value. The user will obtain the voltage value from PB2 or VREXT. When LDO3VPD is 1, the LDO3V output is floating, and the VR voltage value is input from PB2 or VREXT.

The two comparators OPA1 and OPA0 have the same structure, and the only difference is that the initial values of the DACs are different. The following description takes OPA0 as an example. The non-inverting input terminal is the fixed input voltage to be measured by OP0N PAD. The inverting input terminal is provided by DAC0 to provide the voltage comparison reference value to OPA0. When DAC0BFSW=1, DAC0 can be obtained from OP0P PAD. The output voltage and driving capacity are about 2mA. Due to load and process drift, the accuracy of the absolute value of the DAC0 output voltage obtained by OP0P PAD cannot be guaranteed. Alternatively, the user can also input the precise voltage value from OP0P PAD to provide the comparison reference value to OPA0.

The DAC output formula taking into account the internal voltage drop is:  $DACO = 20mV + DACVREF * (DACD / 16384)$ , where  $DACD = \{DACDH, DACDL\}$ .

The output characteristics of OPA0 and OPA1 are CMOS open drain output. Both are output to the OPO PAD. When the output of OP0O or OP1O is 0, the OPO will output 0, which means that battery overvoltage or battery overcurrent occurs at this time. event. In addition, the output values of OP0O and OP1O are stored in registers OPA0OSTUS and OPA0OSTUS respectively for users to read.

After power-on, the default output DAC0 voltage of PB0 is about 60mV, and the output DAC1 voltage of PB1 is about 2.4V.

For the compatibility requirements of different package wiring, when the SOPAN (SYSCFG.6) bit is set to 1, the OP0N and OP1N functions are exchanged.

The OPA2 input can be OP2P PAD or OP0N PAD. By default, OPA2 is used as a 20x amplifier. The gain of the amplifier can be adjusted by the register. The OPA2 output can be sent internally to the ADC for calculation, and can also be output to PAD. OP2O PAD has poor driving ability. When SOPM is set to 3, OPA2 is used as a voltage detector. The comparison voltage of this voltage detector can be VR or VSS.

Finally, there is a description of the IC's own power consumption and related register control. When OPA0PD is 1, turn off OPA0. When OPA1PD is 1, turn off OPA1. When DACPD is 1, the driving source including DAC0~1 and OPA0~1 will be turned off. When LDOC3VPD is 1, the LDOC3V

function is turned off. Including ADC, the LDO1P2V function will be automatically turned off when no module uses LDO1P2V. Including ADC, LVD and LVR, the VBG function will be automatically turned off when no module uses VBG.

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BCMCTL	OPA0PD	OPA0PSW	DAC0SW	DAC0BFSW	OPA1PD	OPA1PSW	DAC1SW	DAC1BFSW
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	0	0	1	1	0

- 1Ah.7 **OPA0PD:** OPA0 Power Down
- 1Ah.6 **OPA0PSW:** Turn on the switch for the path from OPA0P to OPA0 non-inverting input(V+)  
0:switch off 1:switch on
- 1Ah.5 **DAC0SW:** Turn on the switch for the path from DAC0 to OPA0 non-inverting input(V+)  
0:switch off 1:switch on
- 1Ah.4 **DAC0BFSW:** Turn on the switch for DAC0 output Buffer  
0:switch off 1:switch on
- 1Ah.3 **OPA1PD:** OPA1 Power Down
- 1Ah.2 **OPA1PSW:** Turn on the switch for the path from OPA1P to OPA1 non-inverting input(V+)  
0:switch off 1:switch on
- 1Ah.1 **DAC1SW:** Turn on the switch for the path from DAC1 to OPA1 non-inverting input(V+)  
0:switch off 1:switch on
- 1Ah.0 **DAC1BFSW:** Turn on the switch for DAC1 output Buffer  
0:switch off 1:switch on

1Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BCMCTL2	DACPD	DAC0VREFS	DAC1VREFS	OPAEL	OPA0OFTS	OPA1OFTS	OPA0OSTUS	OPA1OSTUS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R
Reset	0	1	1	0	0	0	-	-

- 1Bh.7 **DACPD:**DAC0,DAC1,OPA0,OPA1 bias Power Down
- 1Bh.6 **DAC0VREFS:** DAC0 reference voltage selection  
0:LDO1.2V 1:VR
- 1Bh.5 **DAC1VREFS:** DAC1 reference voltage selection  
0:V<sub>CC</sub> 1:VR
- 1Bh.4 **OPAEL:** Force OPO output low.
- 1Ah.3 **OPA0OFTS:** OPA0 non-inverting input(V+) is connected to inverting input(V-) for OPA0 trim
- 1Bh.2 **OPA1OFTS:** OPA1 non-inverting input(V+) is connected to inverting input(V-) for OPA1 trim
- 1Bh.1 **OPA0OSTUS:** OPA0 comparator output status
- 1Bh.0 **OPA1OSTUS:** OPA1 comparator output status

1Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DAC0DH			DAC0DH					
R/W			R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	0	0	0	0	0	0

- 1Ch.5~0 **DAC0DH:** DAC0 Data bit13~bit8

1Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DAC0DL	DAC0DL							
R/W	R/W							
Reset	1	1	0	1	1	0	1	0

1Dh.7~0 **DAC0DL**: DAC0 Data bit7~bit0  
Write DAC0DL first, then DAC0DH

1Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DAC1DH	–	–	DAC1DH					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	1	1	0	0	1	1

1Eh.5~0 **DAC1DH**: DAC1 Data bit13~bit8

1Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DAC1DL	DAC1DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	1	1	0	1	0

1Fh.7~0 **DAC1DL**: DAC1 Data bit7~bit0  
Write DAC1DL first, then DAC1DH

110h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OP0TRIM	OP0TRIM							
R/W			R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

110h.5~0 **OP0TRIM**: 6-bit OPA0 trim value

111h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OP1TRIM	OP1TRIM							
R/W			R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

111h.5~0 **OP1TRIM**: 6-bit OPA1 trim value

112h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OP2TRIM	OP2TRIM							
R/W				R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

112h.4~0 **OP2TRIM**: 5-bit OPA2 trim value

114h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BCMCTL3	OPA2PD	SOPP	OP2TOPAD	SOTV	SOPG		SOPM	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	0	0	0

114h.7 **OPA2PD:** OPA2 Power Down

114h.6 **SOPP:** OPA2 OPP selection.

0:OP2P (PB6) 1:OP0N (if SOPAN=0) or OP1N (if SOPAN=1)

**OP2TOPAD:** OPA2 to iopad output enable.

114h.5 (weak driving capability)

0:switch off 1:switch on

114h.4 **SOTV:** OPA2 VREF selection when OPA2 as comparator

0: VSS 1: VR

114h.3~2 **SOPG:** OPA2 negative feedback gain selection

00: 1X 01: 10X 10: 20X 11: 50X

**SOPM:** OPA2 operating mode selection

00: opa with negative feedback (non-inverting amplifier)

non-inverting input(V+) = OPP

01: comparator for OPA2 trim, OPP is disconnected,

non-inverting input(V+) = VREF+offset, inverting input(V-) = VREF

114h.1~0 10: opa with negative feedback for OPA2 trim, OPP is disconnected,

non-inverting input(V+) = VSS+offset, inverting input(V-) = VSS

output voltage = offset\*gain, gain=50

11: comparator as voltage level detector

non-inverting input(V+) = OPP, inverting input(V-) = VREF

115h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL	SVBIAS	SBFIN		SVBGT	VBGTOE	LDO3VPD		
R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	1	1	0	0	0	-	-

115h.4 **SVBGT:** VBGT(PA3) output selection

0:VBGT=V<sub>TEMP</sub> 1:VBGT=DACLDO1P2V (only for testing)

115h.3 **VBGTOE:** VBGT(PA3) output enable.

0:disable 1:enable (only for testing)

**LDOV3VPD:** LDO3V power down.

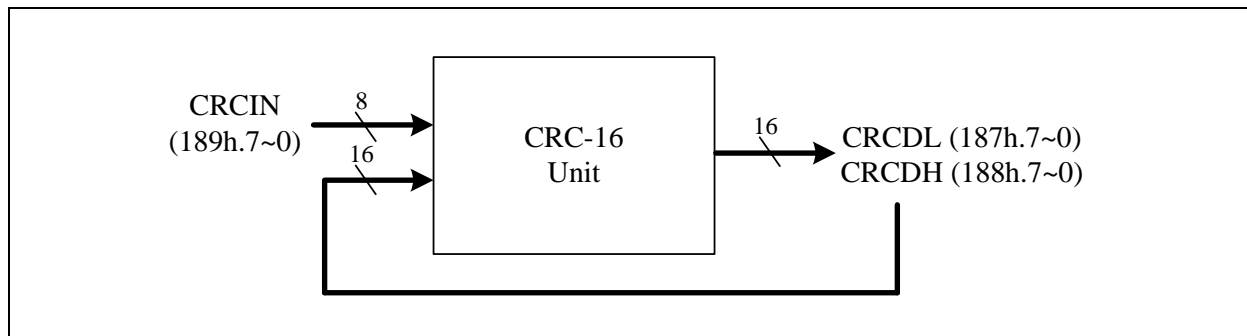
115h.2 0: LDO3V enable

1: LDO3V power down



## 6.10 Cyclic Redundancy Check (CRC)

The chip supports an integrated 16-bit Cyclic Redundancy Check function. The Cyclic Redundancy Check (CRC) calculation unit is an error detection technique test algorithm and uses to verify data transmission or storage data correctness. The CRC calculation takes an 8-bit data stream or a block of data as input and generates a 16-bit output remainder. The data stream is calculated by the same generator polynomial.



**CRC16 Block Diagram**

The CRC generator provides the 16-bit CRC result calculation based on the CRC-16-IBM polynomial. In this CRC generator, there is only one polynomial available for the numeric values calculation. It can't support the 16-bit CRC calculations based on any other polynomials. Each write operation to the CRCIN register creates a combination of the previous CRC value stored in the CRCDH and CRCDL registers. It will take one MCU instruction cycle to calculate.

CRC-16-IBM (Modbus) Polynomial representation:  $X^{16} + X^{15} + X^2 + 1$

187h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDL	CRCDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

187h.7~0 **CRCDL**: 16-bit CRC checksum data bit 7~0

188h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDH	CRCDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

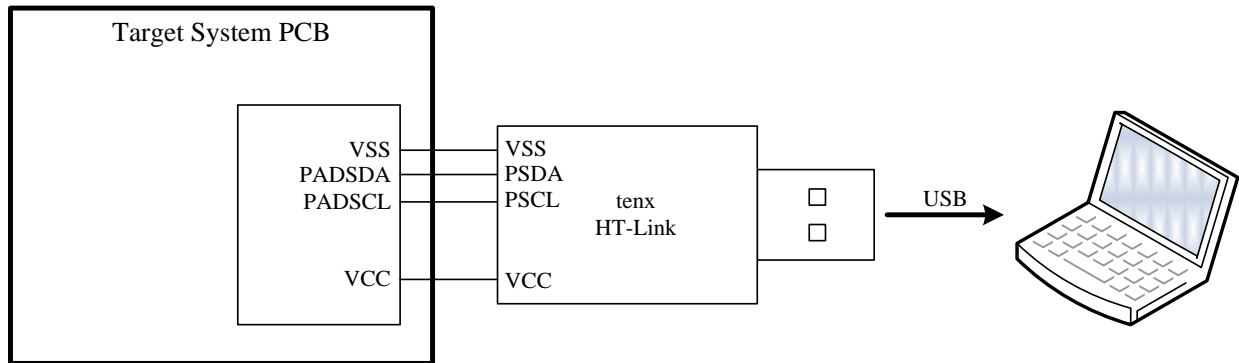
188h.7~0 **CRCDH**: 16-bit CRC checksum data bit 15~8

189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCIN	CRCIN							
W	W							
Reset	—	—	—	—	—	—	—	—

189h.7~0 **CRCIN**: CRC data input, write this register to start CRC calculation

### 6.11 In Circuit Emulation (ICE)

The device supports in-circuit emulation, to use ICE mode, the user needs to set the PROT bit low and connect the ICE dedicated pins (PADSDA, PADSCL) to the tenx proprietary EV module. The benefit of this is that the user can emulate the entire system without changing the onboard target device.



## MEMORY MAP

Name	Address	R/W	Rst	Description
<b>INDF (00h/80h/100h/180h)</b>				<b>Function related to: RAM W/R</b>
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register
<b>TM0 (01h/101h)</b>				<b>Function related to: Timer0</b>
TM0	01.7~0	R/W	00	Timer0 content
<b>PCL (02h/82h/102h/182h)</b>				<b>Function related to: Programming Counter (PC)</b>
PCL	02.7~0	R/W	00	Programming Counter data bit 7~0
<b>STATUS (03h/83h/103h/183h)</b>				<b>Function related to: STATUS</b>
IRP	03.7	R/W	0	Register Bank Select bit (used for indirect addressing)
RP1	03.6	R/W	0	Register Bank Select bit 1 for direct addressing
RP0	03.5	R/W	0	Register Bank Select bit 0 for direct addressing
TO	03.4	R	0	WDT timeout flag, cleared by PWRST, 'SLEEP' or 'CLRWDWT' instruction
PD	03.3	R	0	Power down flag, set by 'SLEEP', cleared by 'CLRWDWT' instruction
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag
C	03.0	R/W	0	Carry flag
<b>FSR (04h/84h/104h/184h)</b>				<b>Function related to: RAM W/R</b>
FSR	04.7~0	R/W	-	File Select Register, indirect address mode pointer
<b>PAD (05h)</b>				<b>Function related to: Port</b>
PAD	05.7~0	R/W	FF	Port A data
<b>PBD (06h)</b>				<b>Function related to: Port</b>
PBD	06.7~0	R/W	FF	Port B data
<b>SFR0A (0Ah/8Ah/10Ah/18Ah)</b>				<b>Function related to: Programming Counter (PC)</b>
GPR	0A.7~4	R/W	0	General Purpose Register
PCH_LAT	0A.3~0	R/W	0	Program counter high byte write buffer When the CPU executes any instruction that will modify PCL, PC[11:8] value is provided by register PCH_LAT. This function can be disabled by register SFR10C.
<b>INTIE (0Bh/8Bh/10Bh/18Bh)</b>				<b>Function related to: Interrupt Enable</b>
ADCIE	0B.7	R/W	0	ADC interrupt enable 0: disable 1: enable
T2IE	0B.6	R/W	0	T2 interrupt enable 0: disable 1: enable
TM1IE	0B.5	R/W	0	Timer1 interrupt enable 0: disable 1: enable
TM0IE	0B.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
WKTIE	0B.3	R/W	0	Wakeup Timer interrupt enable and Wakeup Timer enable 0: disable 1: enable
INT2IE	0B.2	R/W	0	INT2 pin (PA7 or PB5) interrupt enable 0: disable 1: enable

Name	Address	R/W	Rst	Description
INT1IE	0B.1	R/W	0	INT1 pin (PA1 or PB1) interrupt enable 0: disable 1: enable
INT0IE	0B.0	R/W	0	INT0 pin (PA3 or PB2) interrupt enable 0: disable 1: enable
<b>INTIF (0Ch)</b>				<b>Function related to: Interrupt Flag</b>
ADCIF	0C.7	R/W	0	ADC interrupt flag, set by H/W after ADC end of conversion. write 0: clear this flag; write 1: no action
T2IF	0C.6	R/W	0	T2 interrupt event pending flag, set by H/W while T2 overflows write 0: clear this flag; write 1: no action
TM1IF	0C.5	R/W	0	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows write 0: clear this flag; write 1: no action
TM0IF	0C.4	R/W	0	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows write 0: clear this flag; write 1: no action
WKTIF	0C.3	R/W	0	WKT interrupt event pending flag, set by H/W while WKT time out. write 0: clear this flag; write 1: no action
INT2IF	0C.2	R/W	0	INT2 (PA7 or PB5) interrupt event pending flag, set by H/W at INT2 pin's falling edge write 0: clear this flag; write 1: no action
INT1IF	0C.1	R/W	0	INT1 (PA1 or PB1) interrupt event pending flag, set by H/W at INT1 pin's falling/rising edge write 0: clear this flag; write 1: no action
INT0IF	0C.0	R/W	0	INT0 (PA3 or PB2) interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge. write 0: clear this flag; write 1: no action
<b>INTIE1 (0Dh)</b>				<b>Function related to: Interrupt Enable</b>
EA	0D.7	R/W	1	Global interrupt enable 0: disable 1: enable
UARTIE	0D.6	R/W	0	UART interrupt enable 0: disable 1: enable
PWMIE	0D.1	R/W	0	PWM interrupt enable 0: disable 1: enable
LVDIE	0D.0	R/W	0	LVD interrupt enable 0: disable 1: enable
<b>INTIF1 (0Eh)</b>				<b>Function related to: Interrupt Flag</b>
UARTIF	0E.6	R	0	UART interrupt event pending flag, set by H/W when UART transmission/reception is completed. write 0 to TI/RI flag will clear this flag.
PWMIF	0E.1	R/W	0	PWM interrupt event pending flag, set by H/W after PWM period counter roll over. write 0: clear this flag; write 1: no action
LVDIF	0E.0	R/W	0	LVD interrupt event pending flag, set by H/W while $V_{CC} < V_{LVD}$ write 0: clear this flag; write 1: no action
<b>CLKCTL (0Fh)</b>				<b>Function related to: Clock</b>
SLOWSTP	0F.4	R/W	0	Stop Slow-clock after execute SLEEP instruction 0: Slow-clock keeps running after execute SLEEP instruction 1: Slow-clock stop running after execute SLEEP instruction
FASTSTP	0F.3	R/W	1	Stop Fast-clock 0: Fast-clock is running 1: Fast-clock stops running

Name	Address	R/W	Rst	Description
CPUCKS	0F.2	R/W	0	System clock(Fsys) source selection 0: Slow-clock 1: Fast-clock
CPUPSC	0F.1~0	R/W	3	System clock(Fsys) source prescaler. System clock source 00: div 8 01: div 4 10: div 2 11: div 1
<b>TM0RLD (10h)</b>				<b>Function related to: Timer0</b>
TM0RLD	10.7~0	R/W	00	Timer0 reload data
<b>TM0CTL (11h)</b>				<b>Function related to: Timer0</b>
TM0STP	11.6	R/W	0	Stop Timer0 0: Timer0 runs 1: Timer0 stops
TM0EDG	11.5	R/W	0	TM0I (PA2) edge 0: rising edge 1: falling edge
TM0CKS	11.4	R/W	0	Timer0 prescaler clock source 0: Fsys/2 1: TM0I (PA2)
TM0PSC	11.3~0	R/W	0	Timer0 prescaler. Timer0 prescaler clock source divided by 0000: 1 0011: 8 0110: 64 0001: 2 0100: 16 0111: 128 0010: 4 0101: 32 1xxx: 256
<b>TM1 (12h)</b>				<b>Function related to: Timer1</b>
TM1	12.7~0	R/W	00	Timer1 content
<b>TM1RLD (13h)</b>				<b>Function related to: Timer1</b>
TM1RLD	13.7~0	R/W	00	Timer1 reload data
<b>TM1CTL (14h)</b>				<b>Function related to: Timer1</b>
TM1STP	14.6	R/W	0	Stop Timer1 0: Timer1 runs 1: Timer1 stops
TM1PSC	14.3~0	R/W	0	Timer1 prescaler. Timer1 clock source (Fsys/2) divided by 0000: 1 0011: 8 0110: 64 0001: 2 0100: 16 0111: 128 0010: 4 0101: 32 1xxx: 256
<b>T2CTL (15h)</b>				<b>Function related to: T2</b>
T2CLR	15.4	R/W	0	Clear and stop T2 0: T2 runs 1: T2 clear and stops
T2CKS	15.3~2	R/W	0	T2 clock source selection 00: Slow-clock 11: Fsys/128 1x: FIRC/512 (9.216MHz/512)
T2PSC	15.1~0	R/W	0	T2 prescaler. T2 clock source divided by 00: 32768 01: 16384 10: 8192 11: 128
<b>LVCTL (16h)</b>				<b>Function related to: LVD / LVR</b>
LVDF	16.7	R	0	Low voltage detection flag 0: $V_{CC} > V_{LVD}$ 1: $V_{CC} < V_{LVD}$
LVDHYS	16.6	R/W	0	LVD Hysteresis 0: disable 1: enable
LVRSAV	16.5	R/W	1	POR/LVR will be disabled during IDLE/STOP mode to reduce power consumption
LVDSAV	16.4	R/W	1	LVD will be disabled during IDLE/STOP mode to reduce power consumption

Name	Address	R/W	Rst	Description
LVDS	16.3~0	R/W	0	LVD voltage ( $V_{LVD}$ ) selection 0000: disable    0100 : 2.65V    1000: 3.22V    1100: 3.78V 0001: 2.24V    0101: 2.79V    1001: 3.36V    1101: 3.92V 0010: 2.37V    0110: 2.93V    1010: 3.50V    1110: 4.06V 0011: 2.51V    0111: 3.07V    1011: 3.64V    1111: 4.20V
<b>ADCDH (17h)</b>				<b>Function related to: ADC</b>
ADCDH	17.7~0	R	-	ADC output data bit 11~4
<b>ADCTL (18h)</b>				<b>Function related to: ADC</b>
ADCDL	18.7~4	R	-	ADC output data bit 3~0
ADST	18.3	R/W	0	ADC start bit. 0: H/W clear after end of conversion 1: ADC start conversion
ADCKS	18.2~0	R/W	0	ADC clock frequency selection. 1MHz(Typ.) 000: Fsys/256    010: Fsys/64    100: Fsys/16    110: Fsys/4 001: Fsys/128    011: Fsys/32    101: Fsys/8    111: Fsys/2
<b>ADCTL2 (19h)</b>				<b>Function related to: ADC</b>
ADVREFS	19.7~6	R/W	01	ADC reference voltage selection 00: ADC reference voltage is $V_{CC}$ 01: ADC reference voltage is VR 10: ADC reference voltage is LDO1.2V 11: Reserved
ADCHS	19.4~0	R/W	1F	ADC channel selection 00000: ADC0 (PA0)    01000: ADC8 (PB0)    10000: ADC16 (OP1N) 00001: ADC1 (PA1)    01001: ADC9 (PB1)    10001: VR 00010: ADC2 (PA2)    01010: ADC10 (PB2)    10010: OPA2TOADC 00011: ADC3 (PA3)    01011: ADC11 (PB3)    10011: $V_{TEMP}$ 00100: ADC4 (PA4)    01100: ADC12 (PB4)    10100: LDO1.2V 00101: ADC5 (PA5)    01101: ADC13 (PB5)    10101: VSS 00110: ADC6 (PA6)    01110: ADC14 (PB6)    10110: $V_{CC}/201$ 00111: ADC7 (PA7)    01111: ADC15 (OP0N)    10111: $V_{CC}/4$
<b>BCMCTL (1Ah)</b>				<b>Function related to: BCM</b>
OPA0PD	1A.7	R/W	0	OPA0 Power Down
OPA0PSW	1A.6	R/W	1	Turn on the switch for the path from OPA0P to OPA0 non-inverting input(V+) 0:switch off 1:switch on
DAC0SW	1A.5	R/W	1	Turn on the switch for the path from DAC0 to OPA0 non-inverting input(V+) 0:switch off 1:switch on
DAC0BFSW	1A.4	R/W	0	Turn on the switch for DAC0 output Buffer 0:switch off 1:switch on
OPA1PD	1A.3	R/W	0	OPA1 Power Down
OPA1PSW	1A.2	R/W	1	Turn on the switch for the path from OPA1P to OPA1 non-inverting input(V+) 0:switch off 1:switch on
DAC1SW	1A.1	R/W	1	Turn on the switch for the path from DAC1 to OPA1 non-inverting input(V+) 0:switch off 1:switch on
DAC1BFSW	1A.0	R/W	0	Turn on the switch for DAC1 output Buffer 0:switch off 1:switch on
<b>BCMCTL2 (1Bh)</b>				<b>Function related to: BCM</b>
DACPD	1B.7	R/W	0	DAC0,DAC1,OPA0,OPA1 bias voltage Power Down
DAC0VREFS	1B.6	R/W	1	DAC0 reference voltage selection 0:LDO1.2V 1:VR
DAC1VREFS	1B.5	R/W	1	DAC1 reference voltage selection

Name	Address	R/W	Rst	Description
				0:V <sub>CC</sub> 1:VR
OPAEL	1B.4	R/W	0	Force OPO output low.
OPA0OFTS	1B.3	R/W	0	OPA0 non-inverting input (V+) is connected to the inverting input (V-) for OPA0 trim
OPA1OFTS	1B.2	R/W	0	OPA1 non-inverting input (V+) is connected to the inverting input (V-) for OPA1 trim
OPA0OSTUS	1B.1	R	-	OPA0 comparator output status
OPA1OSTUS	1B.0	R	-	OPA1 comparator output status
<b>DAC0DH (1Ch)</b>				<b>Function related to: BCM</b>
DAC0DH	1C.7~0	R/W	00	DAC0 Data bit13~bit8
<b>DAC0DL (1Dh)</b>				<b>Function related to: BCM</b>
DAC0DL	1D.7~0	R/W	DA	DAC0 Data bit7~bit0 Write DAC0DL first, then DAC0DH
<b>DAC1DH (1Eh)</b>				<b>Function related to: BCM</b>
DAC1DH	1E.7~0	R/W	33	DAC1 Data bit13~bit8
<b>DAC1DL (1Fh)</b>				<b>Function related to: BCM</b>
DAC1DL	1F.7~0	R/W	9A	DAC1 Data bit7~bit0 Write DAC1DL first, then DAC1DH
<b>User Data Memory</b>				
RAM	20~6F	R/W	-	RAM Bank0 area (80 Bytes)
RAM	70~7F	R/W	-	RAM common area (16 Bytes)

Name	Address	R/W	Rst	Description
<b>OPTION (81h/181h)</b>				<b>Function related to: STATUS / INT / WDT / WKT</b>
HWAUTO	81.7	R/W	0	Enter/Exit interrupt subroutine, HW auto Save/Restore WREG, FSR, TABR, PCH_LAT, DPL, DPH, and STATUS w/o TO, PD 0:disable 1: enable
INT0EDG	81.6	R/W	0	INT0 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
INT1EDG	81.5	R/W	0	INT1 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
WDTPSC	81.3~2	R/W	3	WDT period selections: 00: 221ms 01: 443ms 10: 1771ms 11: 3542ms @Vcc=5V
WKT PSC	81.1~0	R/W	3	WKT period selections: 00: 28ms 01: 55ms 10: 111ms 11: 221ms @Vcc=5V
<b>PAMOD10 (85h)</b>				<b>Function related to: Port</b>
PA1MOD	85.7~4	R/W	1	PA1 I/O mode control
PA0MOD	85.3~0	R/W	1	PA0 I/O mode control
<b>PAMOD32 (86h)</b>				<b>Function related to: Port</b>
PA3MOD	86.7~4	R/W	1	PA3 I/O mode control
PA2MOD	86.3~0	R/W	1	PA2 I/O mode control
<b>PAMOD54 (87h)</b>				<b>Function related to: Port</b>
PA5MOD	87.7~4	R/W	1	PA5 I/O mode control
PA4MOD	87.3~0	R/W	1	PA4 I/O mode control
<b>PAMOD76 (88h)</b>				<b>Function related to: Port</b>
PA7MOD	88.7~4	R/W	0	PA7 I/O mode control
PA6MOD	88.3~0	R/W	1	PA6 I/O mode control
<b>PWMCTL (89h)</b>				<b>Function related to: PWM</b>
PWMEN	89.7	R/W	0	PWM Clock Enable 0: PWM Clock Disable 1: PWM Clock Enable
PWM0OM	89.5~4	R/W	0	PWM0 output mode 00: Mode0 10: Mode2 01: Mode1 11: Mode3
PWM0DZ	89.3~0	R/W	0	PWM0 non-overlap control 0000: no non-overlap 0001: non-overlap width are 1 PWM clock cycle 0010: non-overlap width are 2 PWM clock cycles ... 1111: non-overlap width are 15 PWM clock cycles
<b>PBMOD10 (8Ch)</b>				<b>Function related to: Port</b>
PB1MOD	8C.7~4	R/W	1	PB1 I/O mode control
PB0MOD	8C.3~0	R/W	1	PB0 I/O mode control
<b>PBMOD32 (8Dh)</b>				<b>Function related to: Port</b>
PB3MOD	8D.7~4	R/W	1	PB3 I/O mode control
PB2MOD	8D.3~0	R/W	1	PB2 I/O mode control
<b>PBMOD54 (8Eh)</b>				<b>Function related to: Port</b>
PB5MOD	8E.7~4	R/W	1	PB5 I/O mode control
PB4MOD	8E.3~0	R/W	1	PB4 I/O mode control
<b>PBMOD76 (8Fh)</b>				<b>Function related to: Port</b>
-	8F.7~4	R/W	1	Reserved
PB6MOD	8F.3~0	R/W	1	PB6 I/O mode control
<b>OPTION2 (91h)</b>				<b>Function related to: PWM0/INT2/INT1/INT0</b>
PWMCKS	91.5~4	R/W	00	PWM Clock Source selection 0x: Fsys



Name	Address	R/W	Rst	Description
				10: FIRC (18.432 MHz) 11: FIRC*2 (36.864 MHz)
INT2SEL	91.2	R/W	0	INT2 pin selection 0: PA7 1: PB5
INT1SEL	91.1	R/W	0	INT1 pin selection 0: PA1 1: PB1
INT0SEL	91.0	R/W	0	INT0 pin selection 0: PA3 1: PB0
<b>PWMPRDH (92h)</b>				<b>Function related to: PWM</b>
PWMPRDH	92.7~0	R/W	FF	PWM Period bit 15~8
<b>PWMPRDL (93h)</b>				<b>Function related to: PWM</b>
PWMPRDL	93.7~0	R/W	FF	PWM Period bit 7~0
<b>PWM0DH (94h)</b>				<b>Function related to: PWM</b>
PWM0DH	94.7~0	R/W	80	PWM0 Duty bit 15~8
<b>PWM0DL (95h)</b>				<b>Function related to: PWM</b>
PWM0DL	95.7~0	R/W	00	PWM0 Duty bit 7~0
<b>PWM1DH (96h)</b>				<b>Function related to: PWM</b>
PWM1DH	96.7~0	R/W	80	PWM1 Duty bit 15~8
<b>PWM1DL (97h)</b>				<b>Function related to: PWM</b>
PWM1DL	97.7~0	R/W	00	PWM1 Duty bit 7~0
<b>PWM2DH (98h)</b>				<b>Function related to: PWM</b>
PWM2DH	98.7~0	R/W	80	PWM2 Duty bit 15~8
<b>PWM2DL (99h)</b>				<b>Function related to: PWM</b>
PWM2DL	99.7~0	R/W	00	PWM2 Duty bit 7~0
<b>User Data Memory</b>				
RAM	A0~EF	R/W	-	RAM Bank1 area (80 Bytes)

Name	Address	R/W	Rst	Description
<b>PWRCTL2 (105h) Function related to: ROM mode</b>				
GPR2	105.7~6	R/W	0	General purpose register
-	105.5	R	-	Reserved
-	105.4	R/W	0	Reserved
-	105.3	R/W	1	Reserved
HSINK	105.2	R/W	1	All GPIO high sink current selection 0: low sink current 1: high sink current
ROMODS	105.1~0	R/W	11	ROM mode selection 11: High speed mode 01: Medium power mode, Fsys < 4MHz 00: Low power mode, Fsys < 1MHz
<b>RDSTP (106h) Function related to: ROM mode</b>				
RDSTP	106.7~0	W	-	Before changing the ROM mode, the user must first write any value to this register to suspend ROM reading for a total of 4 system cycles to ensure that the ROM mode switch is successfully completed.
<b>LVRPD (109h) Function related to: LVR/POR</b>				
LVRPD	109.7~0	W	-	LVR and POR disable option selection. Write 37h to force both LVR and POR to be disabled Write 38h to force LVR to be disabled Write 39h to force POR to be disabled Writing other values will cancel this disable option
PORPDF	109.1	R	0	POR disable option flag 0: POR has not been set by the LVRPD register 1: POR has been set to force disabled by LVRPD register
LVRPDF	109.0	R	0	LVR disable option flag 0: LVR has not been set by the LVRPD register 1: LVR has been set to force disabled by LVRPD register
<b>SFR10C (10Ch) Function related to: Programming Counter (PC)</b>				
SFR10C	10C.7~0	W	-	Use the PCH_LAT function:  The default setting of the chip is that when the CPU executes an "instruction that will modify PCL", PC[11:8] is provided by the register PCH_LAT.  Disable PCH_LAT function:  When the user writes 1C to the register SFR10C, the chip will disable the PCH_LAT function. When the CPU executes an "instruction that modifies PCL", it will leave PC[11:8] unchanged for easy table lookup. Please note that the PCH_LAT feature can only be disabled if the user is using assembly code.  Restore PCH_LAT function:  When the user writes any other value to SFR10C, the system resumes the PCH_LAT function.
PCH	10C.3~0	R	0	Program counter(PC) data bits 11~8, which are the high 4-bit value of the program counter.
<b>CFG07 (10Dh) Function related to: ADC / BCM</b>				
CFG07	10D.4~0	R/W	CFG	Store 5-bit VBG trim value of LDO1.2V if user want to accurate VBG for LDO1.2V, please write the CFG07 value to BGTRIM (10Eh).
<b>BGTRIM (10Eh) Function related to: ADC / BCM</b>				

Name	Address	R/W	Rst	Description
BGTRIM	10E.4~0	R/W	CFG	5-bit VBG trim value (default use VBG trim value for LDO3V)
<b>IRCF (10Fh)</b>				<b>Function related to: Clock</b>
IRCF	10F.6~0	R/W	CFG	7-bit FIRC trim value
<b>OP0TRIM (110h)</b>				<b>Function related to: BCM</b>
OP0TRIM	110.5~0	R/W	CFG	6-bit OPA0 trim value
<b>OP1TRIM (111h)</b>				<b>Function related to: BCM</b>
OP1TRIM	111.5~0	R/W	CFG	6-bit OPA1 trim value
<b>OP2TRIM (112h)</b>				<b>Function related to: BCM</b>
OP2TRIM	112.4~0	R/W	CFG	5-bit OPA2 trim value
<b>RDCTL (113h)</b>				<b>Function related to: BCM</b>
RDCTL	113.1~0	R/W	01	Select the delay time for ROM reading. 00:4ns 01:8ns 10:12ns 11:16ns
<b>BCMCTL3 (114h)</b>				<b>Function related to: BCM</b>
OPA2PD	114.7	R/W	0	OPA2 Power Down
SOPP	114.6	R/W	0	OPA2 non-inverting input(V+) selection. 0:OP2P (PB6) 1:OP0N (if SOPAN=0) or OP1N (if SOPAN=1)
OP2TOPAD	114.5	R/W	0	OPA2 to iopad output enable. (weak driving capability) 0:switch off 1:switch on
SOTV	114.4	R/W	0	OPA2 VREF selection when OPA2 as comparator. 0: VSS 1: VR
SOPG	114.3~2	R/W	10	OPA2 negative feedback gain selection. 00: 1X 01: 10X 10: 20X 11: 50X
SOPM	114.1~0	R/W	00	OPA2 operating mode selection. 00: opa with negative feedback (non-inverting amplifier) non-inverting input(V+) = OPP 01: comparator for OPA2 trim, OPP is disconnected, non-inverting input(V+) = VREF+offset inverting input(V-) = VREF 10: opa with negative feedback for OPA2 trim, OPP is disconnected, non-inverting input(V+) = VSS+offset inverting input(V-) = VSS output voltage = offset*gain, gain=50 11: comparator as voltage level detector non-inverting input(V+) = OPP inverting input(V-) = VREF
<b>PWRCTL (115h)</b>				<b>Function related to: ADC / BCM</b>
SVBIAS	115.7	R/W	0	Reference voltage of VT0 selection 0:V <sub>CC</sub> 1:VR
SBFIN	115.6~5	R/W	11	ADC V <sub>TEMP</sub> selection for temperature sensing 00:V <sub>TEMP</sub> = VT0 (Diode type) 01:V <sub>TEMP</sub> = VT1 (BJT type) 10:V <sub>TEMP</sub> = VBG1.2V 11:V <sub>TEMP</sub> is disabled
SVBGT	115.4	R/W	0	VBGT(PA3) output selection (only for testing) 0:VBGT=V <sub>TEMP</sub> 1:VBGT=LDO1P2V
VBGTOE	115.3	R/W	0	VBGT(PA3) output enable. (only for testing)
LDO3VPD	115.2	R/W	0	LDO3V power down. 0: LDO3V enable 1: LDO3V power down

Name	Address	R/W	Rst	Description
User Data Memory				
RAM	120~16F	R/W	-	RAM Bank2 area (80 Bytes)

Name	Address	R/W	Rst	Description
<b>DPL (185h) Function related to: IAP / Table Read</b>				
DPL	185.7~0	R/W	00	TBL Data Pointer bit 7~0, DPTR={DPH,DPL}
<b>DPH (186h) Function related to: IAP / Table Read</b>				
DPH	186.3~0	R/W	00	TBL Data Pointer bit 11~8, DPTR={DPH,DPL}
<b>CRCDL (187h) Function related to: CRC16</b>				
CRCDL	187.7~0	R/W	FF	16-bit CRC checksum data bit 7~0
<b>CRCDH (188h) Function related to: CRC16</b>				
CRCDH	188.7~0	R/W	FF	16-bit CRC checksum data bit 15~8
<b>CRCIN (189h) Function related to: CRC16</b>				
CRCIN	189.7~0	W	0	CRC data input, write this register to start CRC calculation
<b>TABR (18Ch) Function related to: Table Read</b>				
TABR	18C.7~0	R/W	0	<p>When the user writes 01h to TABR, the W register will get the lower eight bits of the data in the address pointed to by DPTR.</p> <p>When the user writes 02h to TABR, the W register will get the upper eight bits of the data in the address pointed to by DPTR.</p> <p>in Assembly code, user can table read by TABRL/TABRH instruction or writing TABR register.</p> <p>in C code, user can only table read by writing TABR register.</p>
<b>IAPCTL (190h) Function related to: IAP</b>				
IAPTE	190.1~0	R/W	00	IAP Write Time-Out function selection 00: Disable, 01: 3.5ms, 10: 14ms, 11: 28ms
<b>IAPEN (191h) Function related to: IAP</b>				
IAPEN	191.7~0	W	0	<p>Function selection of Table Read and IAP.</p> <p>Write 47h to enable Main ROM Table Read and IAP functions.</p> <p>Write 50h to enable INFO ROM address 6'h20~ 6'h3F Table Read and IAP functions.</p> <p>Writing 33h will disable Table Read and IAP functions.</p>
<b>IAPDTL (192h) Function related to: IAP</b>				
IAPDTL	192.7~0	R/W	0	<p>IAP Data Low byte</p> <p>When the user writes to this register, the hardware will automatically write the 16-bit value {IAPDTH, IAPDTL} to the location pointed to by DPTR.</p>
<b>IAPDTH (193h) Function related to: IAP</b>				
IAPDTH	193.7~0	R/W	0	IAP Data High byte
<b>SCON (195h) Function related to: UART</b>				
UART9	195.7	R/W	0	Number of data transfer bits select 0: 8-bit data transfer 1: 9-bit data transfer
RIMASK	195.5	R/W	0	<p>Receive flag mask control.</p> <p>If this bit is set, the receive flag function will be disabled when RX8 is 0</p>
RXEN	195.4	R/W	0	<p>Receive function enable.</p> <p>0: When UART1W is set low, the RXTX pin is disabled. When UART1W is set high, the RXTX pin is used as the TX pin.</p> <p>1: When UART1W is set low, the RXTX pin is used as the RX pin. When UART1W is set high, the RXTX pin is used as the RX pin.</p>
TX8	195.3	R/W	0	(This bit is only valid when UART9=1)

Name	Address	R/W	Rst	Description
				This bit is the 9th value to be transmitted by TX pin.
RX8	195.2	R/W	0	(This bit is only valid when UART9=1) This bit is the 9th value received by RX pin.
TI	195.1	R/W	0	Transmit flag. Set by H/W when transmission is completed. SW needs to write 0 to clear it, writing 1 does nothing. When TI=1 or RI=1, UARTIF will be set to 1.
RI	195.0	R/W	0	Receive flag. Set by H/W when reception is completed. SW needs to write 0 to clear it, writing 1 does nothing. When TI=1 or RI=1, UARTIF will be set to 1.
<b>SBUF (196h)</b>				<b>Function related to: UART</b>
SBUF	196.7~0	R/W	-	UART transmit/receive data.
<b>UARTCTL (197h)</b>				<b>Function related to: UART</b>
UARTBRP	197.7~0	R/W	0	Define UART Baud Rate Prescaler UART Baud Rate = Fsys/16/UARTBRP
<b>UARTCTL2 (198h)</b>				<b>Function related to: UART</b>
UART1W	198.7	R/W	0	Single-wire mode select 0: full-duplex communication 1: half-duplex communication (single-wire mode)
-	198.6	R/W	0	Reserved
TXS2	198.5	R/W	0	TX pin selection 2 0:disable 1:PB3 is used as TX pin
TXS1	198.4	R/W	0	TX pin selection 1 0:disable 1:PB1 is used as TX pin
TXS0	198.3	R/W	0	TX pin selection 0 0:disable 1:PA1 is used as TX pin
RXTXS2	198.2	R/W	0	RXTX pin selection 2. 0: disable 1: If UART1W is set low, PB0 is used as RX pin. If UART1W is set high and RXEN is set low, PB0 is used as TX pin. If UART1W is set high and RXEN is set high, PB0 is used as RX pin.
RXTXS1	198.1	R/W	0	RXTX pin selection 1. 0: disable 1: If UART1W is set low, PA5 is used as RX pin. If UART1W is set high and RXEN is set low, PA5 is used as TX pin. If UART1W is set high and RXEN is set high, PA5 is used as RX pin.
RXTXS0	198.0	R/W	0	RXTX pin selection 0. 0: disable 1: If UART1W is set low, PA0 is used as RX pin. If UART1W is set high and RXEN is set low, PA0 is used as TX pin. If UART1W is set high and RXEN is set high, PA0 is used as RX pin.

## INSTRUCTION SET

Each instruction is a 16-bit word divided into an Op Code, which specifies the instruction type, and one or more operands, which further specify the operation of the instruction. The instructions can be categorized as byte-oriented, bit-oriented and literal operations list in the following table.

For byte-oriented instructions, “f” represents the address designator and “d” represents the destination designator. The address designator is used to specify which address in Program memory is to be used by the instruction. The destination designator specifies where the result of the operation is to be placed. If “d” is “0”, the result is placed in the W register. If “d” is “1”, the result is placed in the address specified in the instruction.

For bit-oriented instructions, “b” represents a bit field designator, which selects the number of the bit affected by the operation, while “f” represents the address designator. For literal operations, “k” represents the literal or constant value.

Field/Legend	Description
f	Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field, 0: Working register, 1: Register file
W	Working Register
Z	Zero Flag
C	Carry Flag or/Borrow Flag
DC	Decimal Carry Flag or Decimal/Borrow Flag
PC	Program Counter
TOS	Top Of Stack
GIE	Global Interrupt Enable Flag (i-Flag)
[]	Option Field
()	Contents
.	Bit Field
B	Before
A	After
←	Assign direction

Mnemonic		Op Code	Cycle	Flag Affect	Description
<b>Byte-Oriented File Register Instruction</b>					
ADDW <del>X</del>	f, d	ff00 0111 dfff ffff	1	C, DC, Z	Add W and "f"
ANDW <del>X</del>	f, d	ff00 0101 dfff ffff	1	Z	AND W with "f"
CLR <del>X</del>	f	ff00 0001 1fff ffff	1	Z	Clear "f"
CLRW		0000 0001 0100 0000	1	Z	Clear W
COM <del>X</del>	f, d	ff00 1001 dfff ff ff	1	Z	Complement "f"
DEC <del>X</del>	f, d	ff00 0011 dfff ffff	1	Z	Decrement "f"
DEC <del>X</del> SZ	f, d	ff00 1011 dfff ffff	1 or 2	-	Decrement "f", skip if zero
INC <del>X</del>	f, d	ff00 1010 dfff ffff	1	Z	Increment "f"
INC <del>X</del> SZ	f, d	ff00 1111 dfff ffff	1 or 2	-	Increment "f", skip if zero
IORW <del>X</del>	f, d	ff00 0100 dfff ffff	1	Z	OR W with "f"
MOV <del>X</del>	f, d	ff00 1000 dfff ffff	1	Z	Move "f"
MOV <del>X</del> W	f	ff00 1000 0fff ffff	1	Z	Move "f" to W
MOVW <del>X</del>	f	ff00 0000 1fff ffff	1	-	Move W to "f"
RL <del>X</del>	f, d	ff00 1101 dfff ffff	1	C	Rotate left "f" through carry
RR <del>X</del>	f, d	ff00 1100 dfff ffff	1	C	Rotate right "f" through carry
SUBW <del>X</del>	f, d	ff00 0010 dfff ffff	1	C, DC, Z	Subtract W from "f"
SWAP <del>X</del>	f, d	ff00 1110 dfff ffff	1	-	Swap nibbles in "f"
TST <del>X</del>	f	ff00 1000 1fff ffff	1	Z	Test if "f" is zero
XORW <del>X</del>	f, d	ff00 0110 dfff ffff	1	Z	XOR W with "f"
<b>Bit-Oriented File Register Instruction</b>					
BC <del>X</del>	f, b	ff11 00bb bfff ffff	1	-	Clear "b" bit of "f"
BS <del>X</del>	f, b	ff11 01bb bfff ffff	1	-	Set "b" bit of "f"
BT <del>X</del> SC	f, b	ff11 10bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if clear
BT <del>X</del> SS	f, b	ff11 11bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if set
<b>Literal and Control Instruction</b>					
ADDLW	k	0001 1100 kkkk kkkk	1	C, DC, Z	Add Literal "k" and W
ANDLW	k	0001 1011 kkkk kkkk	1	Z	AND Literal "k" with W
L <del>C</del> ALL	k	kk10 0kkk kkkk kkkk	2	-	Call subroutine "k"
CLRWD <del>T</del>		0001 1110 0000 0100	1	TO, PD	Clear Watch Dog Timer
L <del>G</del> OTO	k	kk10 1kkk kkkk kkkk	2	-	Jump to branch "k"
IORLW	k	0001 1010 kkkk kkkk	1	Z	OR Literal "k" with W
MOVLW	k	0001 1001 kkkk kkkK	1	-	Move Literal "k" to W
NOP		0000 0000 0000 0000	1	-	No operation
RET		0000 0000 0100 0000	2	-	Return from subroutine
RETI		0000 0000 0110 0000	2	-	Return from interrupt
RETLW	k	0001 1000 kkkk kkkk	2	-	Return with Literal in W
SLEEP		0001 1110 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
SUBLW	k	0001 1111 kkkk kkkk	1	C, DC, Z	Subtract W from literal
TABRH		0000 0000 0101 1000	2	-	Lookup ROM high data to W
TABRL		0000 0000 0101 0000	2	-	Lookup ROM low data to W
XORLW	k	0001 1101 kkkk kkkk	1	Z	XOR Literal "k" with W



<b>ADDLW</b>	<b>Add Literal "k" and W</b>	
Syntax	ADDLW k	
Operands	k : 00h ~ FFh	
Operation	$(W) \leftarrow (W) + k$	
Status Affected	C, DC, Z	
OP-Code	0001 1100 kkkk kkkk	
Description	The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register.	
Cycle	1	
Example	ADDLW 0x15	B : W =0x10 A : W =0x25

<b>ADDWX</b>	<b>Add W and "f"</b>	
Syntax	ADDWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	$(\text{destination}) \leftarrow (W) + (f)$	
Status Affected	C, DC, Z	
OP-Code	ff00 0111 dfff ffff	
Description	Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	ADDWX FSR, 0	B : W =0x17, FSR =0xC2 A : W =0xD9, FSR =0xC2

<b>ANDLW</b>	<b>Logical AND Literal "k" with W</b>	
Syntax	ANDLW k	
Operands	k : 00h ~ FFh	
Operation	$(W) \leftarrow (W) \text{ AND } k$	
Status Affected	Z	
OP-Code	0001 1011 kkkk kkkk	
Description	The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	ANDLW 0x5F	B : W =0xA3 A : W =0x03

<b>ANDWX</b>	<b>AND W with "f"</b>	
Syntax	ANDWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	$(\text{destination}) \leftarrow (W) \text{ AND } (f)$	
Status Affected	Z	
OP-Code	ff00 0101 dfff ffff	
Description	AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	ANDWX FSR, 1	B : W =0x17, FSR =0xC2 A : W =0x17, FSR =0x02

---

**BCX Clear "b" bit of "f"**


---

Syntax	BCX f [,b]	
Operands	f : 000h ~ 1FFh, b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	ff11 00bb bfff ffff	
Description	Bit 'b' in register 'f' is cleared.	
Cycle	1	
Example	BCX FLAG_REG, 7	B : FLAG_REG =0xC7 A : FLAG_REG =0x47

---

**BSX Set "b" bit of "f"**


---

Syntax	BSX f [,b]	
Operands	f : 000h ~ 1FFh, b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	ff11 01bb bfff ffff	
Description	Bit 'b' in register 'f' is set.	
Cycle	1	
Example	BSX FLAG_REG, 7	B : FLAG_REG =0x0A A : FLAG_REG =0x8A

---

**BTXSC Test "b" bit of "f", skip if clear(0)**


---

Syntax	BTXSC f [,b]	
Operands	f : 000h ~ 1FFh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) =0	
Status Affected	-	
OP-Code	ff11 10bb bfff ffff	
Description	If bit 'b' in register 'f' is 1, then the next instruction is executed. If bit 'b' in register 'f' is 0, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTXSC FLAG, 1 TRUE LGOTO SUB1 FALSE ...	B : PC =LABEL1 A : if FLAG.1 =0, PC =FALSE if FLAG.1 =1, PC =TRUE

---

**BTXSS Test "b" bit of "f", skip if set(1)**


---

Syntax	BTXSS f [,b]	
Operands	f : 000h ~ 1FFh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) =1	
Status Affected	-	
OP-Code	ff11 11bb bfff ffff	
Description	If bit 'b' in register 'f' is 0, then the next instruction is executed. If bit 'b' in register 'f' is 1, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTXSS FLAG, 1 TRUE LGOTO SUB1 FALSE ...	B : PC =LABEL1 A : if FLAG.1 =0, PC =TRUE if FLAG.1 =1, PC =FALSE

---

**CLR X                      Clear "f"**


---

Syntax	CLR X f	
Operands	f : 000h ~ 1FFh	
Operation	(f) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	ff00 0001 1fff ffff	
Description	The contents of register 'f' are cleared and the Z bit is set.	
Cycle	1	
Example	CLR X FLAG_REG	B : FLAG_REG =0x5A A : FLAG_REG =0x00, Z =1

---

**CLR W                      Clear W**


---

Syntax	CLR W	
Operands	-	
Operation	(W) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	0000 0001 0100 0000	
Description	W register is cleared and Z bit is set.	
Cycle	1	
Example	CLR W	B : W =0x5A A : W =0x00, Z =1

---

**CLR WDT                      Clear Watchdog Timer**


---

Syntax	CLR WDT	
Operands	-	
Operation	WDT Timer ← 00h	
Status Affected	TO, PD	
OP-Code	0001 1110 0000 0100	
Description	CLR WDT instruction clears the Watchdog Timer	
Cycle	1	
Example	CLR WDT	B : WDT counter =? A : WDT counter =0x00

---

**COM X                      Complement "f"**


---

Syntax	COM X f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← ( $\bar{f}$ )	
Status Affected	Z	
OP-Code	ff00 1001 dfff ffff	
Description	The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	COM X REG1, 0	B : REG1 =0x13 A : REG1 =0x13, W =0xEC

<b>DECX</b>	<b>Decrement "f"</b>
Syntax	DECX f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) $\leftarrow$ (f) - 1
Status Affected	Z
OP-Code	ff00 0011 dfff ffff
Description	Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	DECX CNT, 1 B : CNT =0x01, Z =0 A : CNT =0x00, Z =1

<b>DECXSZ</b>	<b>Decrement "f", Skip if 0</b>
Syntax	DECXSZ f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) $\leftarrow$ (f) - 1, skip next instruction if result is 0
Status Affected	-
OP-Code	ff00 1011 dfff ffff
Description	The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2 cycle instruction.
Cycle	1 or 2
Example	LABEL1 DECXSZ CNT, 1 LGOTO LOOP CONTINUE B : PC =LABEL1 A : CNT =CNT - 1 if CNT =0, PC =CONTINUE if CNT $\neq$ 0, PC =LABEL1 + 1

<b>INCX</b>	<b>Increment "f"</b>
Syntax	INCX f [,d]
Operands	f : 000h ~ 1FFh
Operation	(destination) $\leftarrow$ (f) + 1
Status Affected	Z
OP-Code	ff00 1010 dfff ffff
Description	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
Cycle	1
Example	INCX CNT, 1 B : CNT =0xFF, Z =0 A : CNT =0x00, Z =1

<b>INCXSZ</b>	<b>Increment "f", Skip if 0</b>
Syntax	INCXSZ f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) $\leftarrow$ (f) + 1, skip next instruction if result is 0
Status Affected	-
OP-Code	ff00 1111 dfff ffff
Description	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2 cycle instruction.
Cycle	1 or 2
Example	<div> <div> <b>LABEL1 INCXSZ CNT, 1</b>  <b>LGOTO LOOP</b>  <b>CONTINUE</b> </div> <div> <b>B : PC =LABEL1</b>  <b>A : CNT =CNT + 1</b>  if CNT =0, PC =CONTINUE  if CNT <math>\neq</math>0, PC =LABEL1 + 1 </div> </div>

<b>IORLW</b>	<b>Inclusive OR Literal with W</b>
Syntax	IORLW k
Operands	k : 00h ~ FFh
Operation	(W) $\leftarrow$ (W) OR k
Status Affected	Z
OP-Code	0001 1010 kkkk kkkk
Description	The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register.
Cycle	1
Example	<div> <div> <b>IORLW 0x35</b> </div> <div> <b>B : W =0x9A</b>  <b>A : W =0xBF, Z =0</b> </div> </div>

<b>IORWX</b>	<b>Inclusive OR W with "f"</b>
Syntax	IORWF f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) $\leftarrow$ (W) OR k
Status Affected	Z
OP-Code	ff00 0100 dfff ffff
Description	Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
Cycle	1
Example	<div> <div> <b>IORWX RESULT, 0</b> </div> <div> <b>B : RESULT =0x13, W =0x91</b>  <b>A : RESULT =0x13, W =0x93, Z =0</b> </div> </div>

<b>LCALL</b>	<b>Call subroutine "k"</b>
Syntax	LCALL k
Operands	k : 0000h ~ 1FFFh
Operation	Operation: TOS $\leftarrow$ (PC) + 1, PC.12~0 $\leftarrow$ k
Status Affected	-
OP-Code	kk10 0kkk kkkk kkkk
Description	LCALL Subroutine. First, return address (PC+1) is pushed onto the stack. The 13-bit immediate address is loaded into PC bits <12:0>. LCALL is a two-cycle instruction.
Cycle	2
Example	LABEL1 LCALL SUB1      B : PC =LABEL1 A : PC =SUB1, TOS =LABEL1 + 1

<b>LGOTO</b>	<b>Unconditional Branch</b>
Syntax	LGOTO k
Operands	k : 0000h ~ 1FFFh
Operation	PC.12~0 $\leftarrow$ k
Status Affected	-
OP-Code	kk10 1kkk kkkk kkkk
Description	LGOTO is an unconditional branch. The 13-bit immediate value is loaded into PC bits <12:0>. LGOTO is a two-cycle instruction.
Cycle	2
Example	LABEL1 LGOTO SUB1      B : PC =LABEL1 A : PC =SUB1

<b>MOVX</b>	<b>Move f</b>
Syntax	MOVX f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) $\leftarrow$ (f)
Status Affected	Z
OP-Code	ff00 1000 dfff ffff
Description	The contents of register 'f' are moved to a destination dependent upon the status of d. If d=0, destination is W register. If d=1, the destination is file register f itself. d=1 is useful to test a file register, since status flag Z is affected.
Cycle	1
Example	MOVX FSR,0      B : FSR =0xC2, W =? A : FSR =0xC2, W =0xC2

<b>MOVXW</b>	<b>Move "f" to W</b>
Syntax	MOVXW f
Operands	f : 000h ~ 1FFh
Operation	(W) $\leftarrow$ (f)
Status Affected	Z
OP-Code	ff00 1000 0fff ffff
Description	The contents of register 'f' are moved to W register.
Cycle	1
Example	MOVXW FSR      B : FSR =0xC2, W =? A : FSR =0xC2, W =0xC2

---

**MOVLW                      Move Literal to W**


---

Syntax	MOVLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← k	
Status Affected	-	
OP-Code	0001 1001 kkkk kkkk	
Description	The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.	
Cycle	1	
Example	MOVLW 0x5A	B : W =? A : W =0x5A

---

**MOVWX                      Move W to 'f'**


---

Syntax	MOVWX f	
Operands	f : 000h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

---

**NOP                          No Operation**


---

Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	0000 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

---

**RET                          Return from Subroutine**

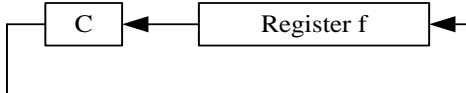

---

Syntax	RET	
Operands	-	
Operation	PC ← TOS	
Status Affected	-	
OP-Code	0000 0000 0100 0000	
Description	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.	
Cycle	2	
Example	RET	A : PC =TOS



<b>RETI</b>	<b>Return from Interrupt</b>	
Syntax	RETI	
Operands	-	
Operation	PC ← TOS, GIE ← 1	
Status Affected	-	
OP-Code	0000 0000 0110 0000	
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is a two-cycle instruction.	
Cycle	2	
Example	RETI	A : PC =TOS, GIE =1

RETLW	Return with Literal in W														
Syntax	RETLW k														
Operands	k : 00h ~ FFh														
Operation	PC $\leftarrow$ TOS, (W) $\leftarrow$ k														
Status Affected	-														
OP-Code	0001 1000 kkkk kkkk														
Description	The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.														
Cycle	2														
Example	<table border="0"> <tr> <td>LCALL TABLE</td> <td>B : W =0x07</td> </tr> <tr> <td>          :</td> <td>A : W =value of k8</td> </tr> <tr> <td>TABLE ADDWX PCL, 1</td> <td></td> </tr> <tr> <td>      RETLW k1</td> <td></td> </tr> <tr> <td>      RETLW k2</td> <td></td> </tr> <tr> <td>      :</td> <td></td> </tr> <tr> <td>      RETLW kn</td> <td></td> </tr> </table>	LCALL TABLE	B : W =0x07	:	A : W =value of k8	TABLE ADDWX PCL, 1		RETLW k1		RETLW k2		:		RETLW kn	
LCALL TABLE	B : W =0x07														
:	A : W =value of k8														
TABLE ADDWX PCL, 1															
RETLW k1															
RETLW k2															
:															
RETLW kn															

<b>RLX</b>	<b>Rotate Left 'f' through Carry</b>
Syntax	RLX f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	
Status Affected	C
OP-Code	ff00 1101 dfff ffff
Description	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	RLX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W     =1100 1100, C =1





**SUBWX**
**Subtract W from 'f'**

Syntax	SUBWX f[,d]		
Operands	f : 000h ~ 1FFh, d : 0, 1		
Operation	(destination) $\leftarrow$ (f) - (W)		
Status Affected	C, DC, Z		
OP-Code	ff00 0010 dfff ffff		
Description	Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.		
Cycle	1		
Example	SUBWX REG1, 1		
		B : REG1 =0x03, W =0x02, C=?, Z=?	
		A : REG1 =0x01, W =0x02, C=1, Z=0	
	SUBWX REG1, 1		
		B : REG1 =0x02, W =0x02, C=?, Z=?	
		A : REG1 =0x00, W =0x02, C=1, Z=1	
	SUBWX REG1, 1		
		B : REG1 =0x01, W =0x02, C=?, Z=?	
		A : REG1 =0xFF, W =0x02, C=0, Z=0	

**SWAPX**
**Swap Nibbles in 'f'**

Syntax	SWAPX f[,d]		
Operands	f : 000h ~ 1FFh, d : 0, 1		
Operation	(destination,7~4) $\leftarrow$ (f.3~0), (destination.3~0) $\leftarrow$ (f.7~4)		
Status Affected	-		
OP-Code	ff00 1110 dfff ffff		
Description	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.		
Cycle	1		
Example	SWAPX REG1, 0		
		B : REG1 =0xA5	
		A : REG1 =0xA5, W =0x5A	

**TABRH**
**Return DPTR high byte to W**

Syntax	TABRH		
Operands	-		
Operation	(W) ← ROM[DPTR] high byte content, Where DPTR = {DPH[max:8], DPL[7:0]}		
Status Affected	-		
OP-Code	0000 0000 0101 1000		
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.		
Cycle	2		
Example	MOVLW	(TAB1&0xFF)	
	MOVWX	DPL	;Where DPL is register
	MOVLW	(TAB1>>8)&0xFF	
	MOVWX	DPH	;Where DPH is register
	TABRL		;W =0x89
	TABRH		;W =0x37
	ORG 0234H		
	TAB1:		
DT	0x3789, 0x2277	;ROM data 16 bits	

---

**TABRL                      Return DPTR low byte to W**


---

Syntax	TABRL		
Operands	-		
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR = {DPH[max:8], DPL[7:0]}		
Status Affected	-		
OP-Code	0000 0000 0101 0000		
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.		
Cycle	2		
Example	<pre> MOVLW    (TAB1&amp;0xFF) MOVWX    DPL                      ;Where DPL is register MOVLW    (TAB1&gt;&gt;8)&amp;0xFF MOVWX    DPH                      ;Where DPH is register  TABRL                      ;W =0x89 TABRH                      ;W =0x37  ORG 0234H  TAB1: DT        0x3789, 0x2277          ;ROM data 16 bits </pre>		

---

**TSTX                      Test if 'f' is zero**


---

Syntax	TSTX f		
Operands	f : 000h ~ 1FFh		
Operation	Set Z flag if (f) is 0		
Status Affected	Z		
OP-Code	ff00 1000 1fff ffff		
Description	If the content of register 'f' is 0, Zero flag is set to 1.		
Cycle	1		
Example	<pre> TSTX REG1                      B : REG1 =0, Z =?                                 A : REG1 =0, Z =1 </pre>		

---

**XORLW                      Exclusive OR Literal with W**


---

Syntax	XORLW k		
Operands	k : 00h ~ FFh		
Operation	(W) ← (W) XOR k		
Status Affected	Z		
OP-Code	0001 1101 kkkk kkkk		
Description	The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.		
Cycle	1		
Example	<pre> XORLW 0xAF                      B : W =0xB5                                 A : W =0x1A </pre>		

<b>XORWX</b>	<b>Exclusive OR W with 'f'</b>
Syntax	XORWX f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) ← (W) XOR (f)
Status Affected	Z
OP-Code	ff00 0110 dfff ffff
Description	Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	<div> XORWX REG1, 1 <div> B : REG1 =0xAF, W =0xB5 A : REG1 =0x1A, W =0xB5 </div> </div>

## ELECTRICAL CHARACTERISTICS

### 1. Absolute Maximum Ratings ( $T_A = 25^\circ\text{C}$ )

Parameter	Rating	Unit
Supply voltage	$V_{SS} - 0.3$ to $V_{SS} + 5.5$	V
Input voltage	$V_{SS} - 0.3$ to $V_{CC} + 0.3$	
Output voltage	$V_{SS} - 0.3$ to $V_{CC} + 0.3$	
Output current high per 1 PIN	-25	mA
Output current high per all PIN	-80	
Output current low per 1 PIN	+30	
Output current low per all PIN	+150	
Maximum operating voltage	5.5	V
Operating temperature	-40 to +105	$^\circ\text{C}$
Storage temperature	-65 to +150	

### 2. DC Characteristics ( $T_A = 25^\circ\text{C}$ , $V_{CC} = 5.0\text{V}$ , unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Operating Voltage	$V_{CC}$	$F_{sys} = 18.432\text{ MHz}$		2.2	—	5.5	V
		$F_{sys} = 9.216\text{ MHz}$		2.0	—	5.5	
Input High Voltage	$V_{IH}$	All Input	$V_{CC} = 3.0 \sim 5.0\text{V}$	$0.6V_{CC}$	—	$V_{CC}$	V
Input Low Voltage	$V_{IL}$	All Input	$V_{CC} = 3.0 \sim 5.0\text{V}$	$V_{SS}$	—	$0.2V_{CC}$	V
I/O port Source Current	$I_{OH}$	All I/O pin	$V_{CC} = 5.0\text{V}$ , $V_{OH} = 4.5\text{V}$	6	12	—	mA
			$V_{CC} = 3.0\text{V}$ , $V_{OH} = 2.7\text{V}$	2.5	5	—	
I/O port Sink Current	$I_{OL}$	All I/O pin (HSINK=1)	$V_{CC} = 5.0\text{V}$ , $V_{OL} = 0.5\text{V}$	38	75	—	mA
			$V_{CC} = 3.0\text{V}$ , $V_{OL} = 0.3\text{V}$	18	35	—	
		All I/O pin (HSINK=0)	$V_{CC} = 5.0\text{V}$ , $V_{OL} = 0.5\text{V}$	22	43	—	mA
			$V_{CC} = 3.0\text{V}$ , $V_{OL} = 0.3\text{V}$	10	19	—	
Input Leakage Current (pin high)	$I_{ILH}$	All Input	$V_{IN} = V_{CC}$	—	—	1	$\mu\text{A}$
Input Leakage Current (pin low)	$I_{ILL}$	All Input	$V_{IN} = 0\text{V}$	—	—	-1	$\mu\text{A}$
I/O pull-up resister	$R_{UP}$	$V_{IN} = 0\text{V}$ Ports A, B	$V_{CC} = 5.0\text{V}$		35.8		$\text{K}\Omega$
			$V_{CC} = 3.0\text{V}$		36.4		

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Operating Current	$I_{DD}$	FAST mode FIRC18.432 MHz $V_{CC} = 5.0V$	–	4.3	–	mA
		$V_{CC} = 3.0V$	–	2.5	–	
		FAST mode FIRC 9.216 MHz $V_{CC} = 5.0V$	–	2.5	–	
		$V_{CC} = 3.0V$	–	1.5	–	
		FAST mode FIRC 4.608 MHz $V_{CC} = 5.0V$	–	1.9	–	
		$V_{CC} = 3.0V$	–	1.2	–	
		FAST mode FIRC 2.304 MHz $V_{CC} = 5.0V$	–	1.6	–	$\mu A$
		$V_{CC} = 3.0V$	–	1.0	–	
		SLOW mode FIRC disable ROMODS=00 LDO3V disable LDO1.2V disable POR/LVR enable LVD enable $V_{CC} = 5.0V$	–	140	–	
		$V_{CC} = 3.0V$	–	95	–	
		SLOW mode FIRC disable ROMODS=00 LDO3V disable LDO1.2V disable POR/LVR disable LVD enable $V_{CC} = 5.0V$	–	95	–	
		$V_{CC} = 3.0V$	–	60	–	
		SLOW mode FIRC disabled ROMODS=00 LDO3V disable LDO1.2V disable POR/LVR disable LVD disable $V_{CC} = 5.0V$	–	18	–	
		$V_{CC} = 3.0V$	–	9	–	
		IDLE mode POR/LVR disable LVD enable $V_{CC} = 5.0V$	–	80	–	
		$V_{CC} = 3.0V$	–	50	–	
		IDLE mode POR/LVR disable LVD disable $V_{CC} = 5.0V$	–	6.5	–	
		$V_{CC} = 3.0V$	–	2.0	–	
		STOP mode POR/LVR disable LVD disable $V_{CC} = 5.0V$	–	–	0.1	
		$V_{CC} = 3.0V$	–	–	0.1	

### 3. Clock Characteristics

Parameter	Condition	Min.	Typ.	Max.	Unit
FIRC Frequency (*)	$T_A = -40^{\circ}\text{C} \sim 105^{\circ}\text{C}$ $V_{CC} = 3.0 \sim 5.5\text{V}$	-2%	18.432	+6%	MHz
	$T_A = -40^{\circ}\text{C} \sim 105^{\circ}\text{C}$ $V_{CC} = 5.0\text{V}$	-1.5%	18.432	+1%	
	$T_A = 25^{\circ}\text{C}$ $V_{CC} = 5.0\text{V}$	-0.5%	18.432	+0.5%	

\*System clock( $F_{\text{sys}}$ ) can be divided by 1/2/4/8.

Parameter	Condition	Min.	Typ.	Max.	Unit
SIRC Frequency (*)	$T_A = 25^{\circ}\text{C}$ $V_{CC} = 5.0\text{V}$	-	37	-	KHz
	$T_A = 25^{\circ}\text{C}$ $V_{CC} = 3.0\text{V}$	-	33	-	

\*System clock( $F_{\text{sys}}$ ) can be divided by 1/2/4/8.

### 4. Reset Timing Characteristics ( $T_A = 25^{\circ}\text{C}$ )

Parameter	Conditions	Min.	Typ.	Max.	Unit
RESET Input Low width	Input $V_{CC} = 5.0\text{V} \pm 10\%$	—	30	—	$\mu\text{s}$
CPU start up time	$V_{CC} = 5.0\text{V}$	—	51	—	ms
	$V_{CC} = 3.0\text{V}$	—	57	—	
WDT time	$V_{CC} = 5.0\text{V}$ , WDTPSC = 11b	—	3542	—	ms
	$V_{CC} = 3.0\text{V}$ , WDTPSC = 11b	—	3972	—	

### 5. Wakeup Timer (WKT) Timing Characteristics ( $T_A = 25^{\circ}\text{C}$ )

Parameter	Conditions	Min.	Typ.	Max.	Unit
WKT time	$V_{CC} = 5.0\text{V}$ , WKTPSC = 11b	—	221	—	ms
	$V_{CC} = 3.0\text{V}$ , WKTPSC = 11b	—	248	—	

**6. LVR Circuit Characteristics ( $T_A = 25^\circ\text{C}$ )**

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
LVR Voltage	$\text{LVR}_{\text{th}}$	SYSCFG.11~8 = 0000b	–	2.13	–	V
		SYSCFG.11~8 = 0001b	–	2.26	–	
		SYSCFG.11~8 = 0010b	–	2.40	–	
		SYSCFG.11~8 = 0011b	–	2.54	–	
		SYSCFG.11~8 = 0100b	–	2.69	–	
		SYSCFG.11~8 = 0101b	–	2.83	–	
		SYSCFG.11~8 = 0110b	–	2.97	–	
		SYSCFG.11~8 = 0111b	–	3.11	–	
		SYSCFG.11~8 = 1000b	–	3.26	–	
		SYSCFG.11~8 = 1001b	–	3.40	–	
		SYSCFG.11~8 = 1010b	–	3.54	–	
		SYSCFG.11~8 = 1011b	–	3.68	–	
		SYSCFG.11~8 = 1100b	–	3.84	–	
		SYSCFG.11~8 = 1101b	–	3.98	–	
		SYSCFG.11~8 = 1110b	–	4.12	–	
		SYSCFG.11~8 = 1111b	–	4.26	–	
LVR Hysteresis Window	$V_{\text{HYS\_LVR}}$	–	–	20	–	mV
Low Voltage Detection time	$T_{\text{LVR}}$	–	100	–	–	$\mu\text{s}$

**7. LVD Circuit Characteristics ( $T_A = 25^\circ\text{C}$ )**

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
LVD Voltage	$\text{LVD}_{\text{th}}$	LVDS = 0001b	–	2.24	–	V
		LVDS = 0010b	–	2.37	–	
		LVDS = 0011b	–	2.51	–	
		LVDS = 0100b	–	2.65	–	
		LVDS = 0101b	–	2.79	–	
		LVDS = 0110b	–	2.93	–	
		LVDS = 0111b	–	3.07	–	
		LVDS = 1000b	–	3.22	–	
		LVDS = 1001b	–	3.36	–	
		LVDS = 1010b	–	3.50	–	
		LVDS = 1011b	–	3.64	–	
		LVDS = 1100b	–	3.78	–	
		LVDS = 1101b	–	3.92	–	
		LVDS = 1110b	–	4.06	–	
		LVDS = 1111b	–	4.20	–	
LVD Hysteresis Window	$V_{\text{HYS\_LVD}}$	LVDHYS = 0	–	20	–	mV
		LVDHYS = 1 (LVDS=0001b)	–	40	–	
		LVDHYS = 1 (LVDS=1111b)	–	80	–	
Low Voltage Detection time	$T_{\text{LVD}}$	–	100	–	–	$\mu\text{s}$



**8. ADC Characteristics** ( $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 3.0\text{V}$  to  $5.5\text{V}$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Conditions	Min.	Typ.	Max.	Units
Total Accuracy	$V_{CC} = 5.0\text{V}$ , $V_{SS} = 0\text{V}$ , $F_{\text{ADC}} = 1\text{ MHz}$	—	$\pm 3$	$\pm 13$	LSB
Integral Non-Linearity		—	$\pm 3.2$	$\pm 15$	
Differential Non-Linearity		—	$\pm 1$	$\pm 4$	
Max Input Clock freq. ( $F_{\text{ADC}}$ )	Source impedance ( $R_s < 10\text{K ohm}$ )	—	—	4	MHz
	Source impedance ( $R_s < 20\text{K ohm}$ )	—	—	2	
	Source impedance ( $R_s < 50\text{K ohm}$ )	—	—	1	
	Source is internal voltage	—	—	4	
Conversion Time	$F_{\text{ADC}} = 2\text{ MHz}$	—	25	—	$\mu\text{s}$
$V_{CC}/4$ reference voltage	$25^\circ\text{C}$ , $V_{CC} = 3.0\text{V} \sim 5.5\text{V}$	-1%	$0.25V_{CC}$	+1%	V
Input Voltage	—	$V_{SS}$	—	$V_{CC}$	V

**9. VBG Characteristics** ( $V_{CC} = 5.0\text{V}$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Conditions	Min.	Typ.	Max.	Units
Bandgap Reference Voltage (LDO1.2V)	$25^\circ\text{C}$	-1%	1.2	+1%	V
	$-20^\circ\text{C} \sim 105^\circ\text{C}$	-1.5%	1.2	+1.5%	V
Bandgap Reference Voltage (VR, LDO3V)	$25^\circ\text{C}$	-1%	3.0	+1%	V
	$-20^\circ\text{C} \sim 105^\circ\text{C}$	-1.5%	3.0	+1.5%	V

**10. OPA Characteristics** ( $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Conditions	Min.	Typ.	Max.	Units
Power supply	—	2.2	—	5.5	V
$V_{\text{icm}}$	—	0.1	—	$V_{CC}-0.7$	V
$V_{\text{os2}}$	After trim	—	2	—	mV
$\Delta V_{\text{os}}/\Delta T$	After trim	—	4	8	$\mu\text{V}/^\circ\text{C}$
AVOL	$R_L = 1\text{M ohm}$ , $C_L = 100\text{ pF}$ , $V_i = 0.1$ to $4\text{V}$ , $V_o = 1$ to $4\text{V}$	—	100	—	dB
GBW	$R_L = 1\text{M ohm}$ , $C_L = 100\text{ pF}$	—	2	—	MHz
CMRR	$V_o = 2\text{V}$	—	80	—	dB
PSRR	$V_o = 2\text{V}$	—	80	—	dB
ICC	Gain = 1, OPP = 5V, OPO > 2.5V at $V_{CC} = 5\text{V}$	—	200	—	$\mu\text{A}$
SR	No load	—	1.2	—	V/ $\mu\text{s}$
IOH	Gain = 1, OPP = 5V, OPO > 2.5V at $V_{CC} = 5\text{V}$	—	8	—	mA
IOL	Gain = 1, OPP = 5V, OPO > 2.5V at $V_{CC} = 5\text{V}$	—	14	—	mA

**11. Comparator Characteristics** ( $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 3.0\text{V}$  to  $5.5\text{V}$ ,  $V_{SS} = 0\text{V}$ )

The VSS voltage used by the DAC0/DAC1 module will be raised by about 10mV.

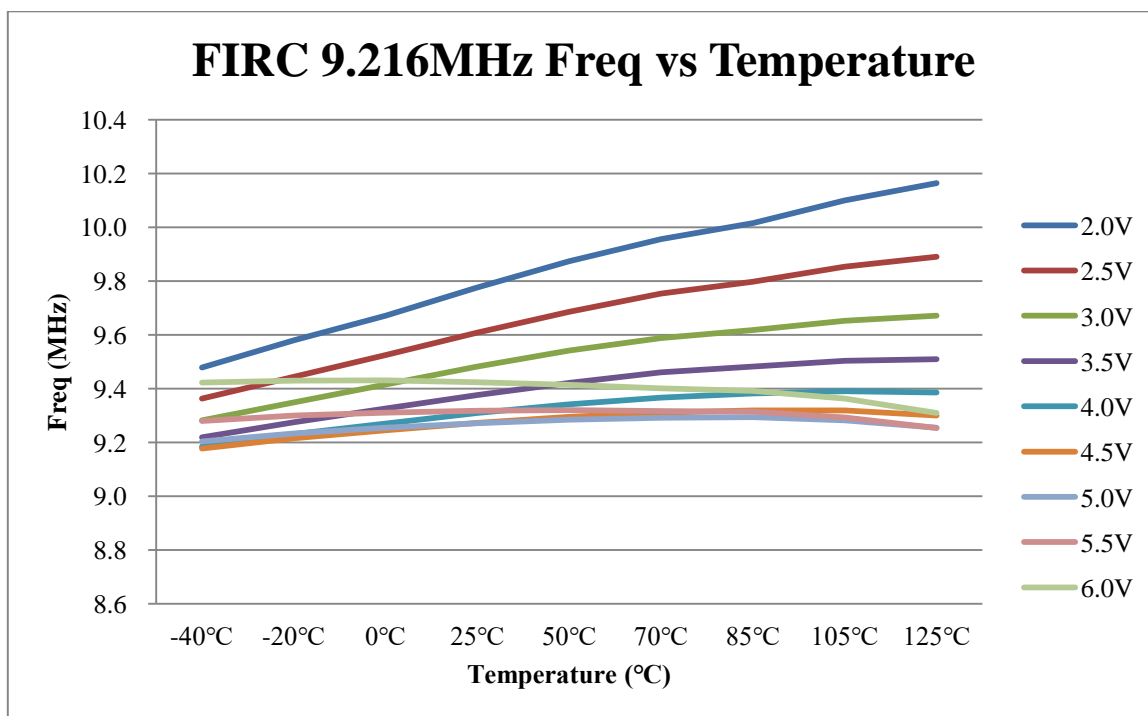
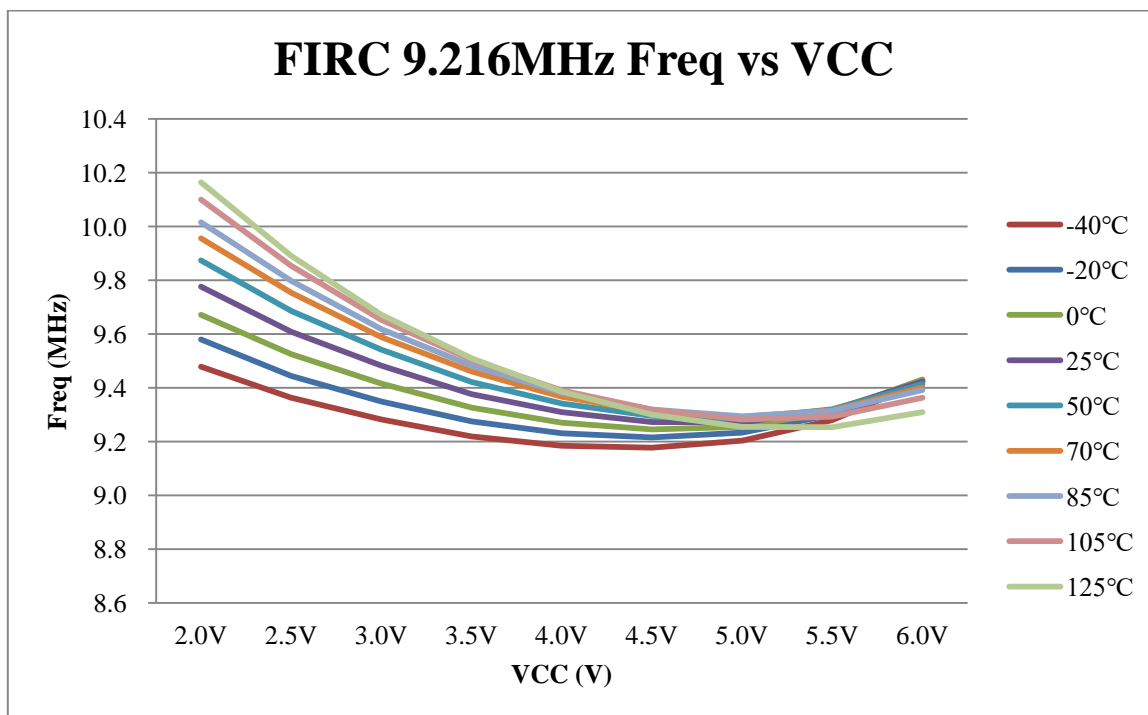
Parameter	Conditions	Min.	Typ.	Max.	Units
Power supply	—	2.2	—	5.5	V
Quiescent Current	$V_{CC} = 5.0\text{V}$	—	100	—	$\mu\text{A}$
DAC Current	$V_{CC} = 5.0\text{V}$	60	—	220	$\mu\text{A}$
$V_{OS\_CMP}$	$V_{CC} = 5.0\text{V}$	-15	—	15	mV
$V_{CM\_CMP}$	$V_{CC} = 5.0\text{V}$	0	—	$V_{CC}-0.5$	V
$V_{HYS\_CMP}$	$V_{CC} = 5.0\text{V}$	5	10	20	mV

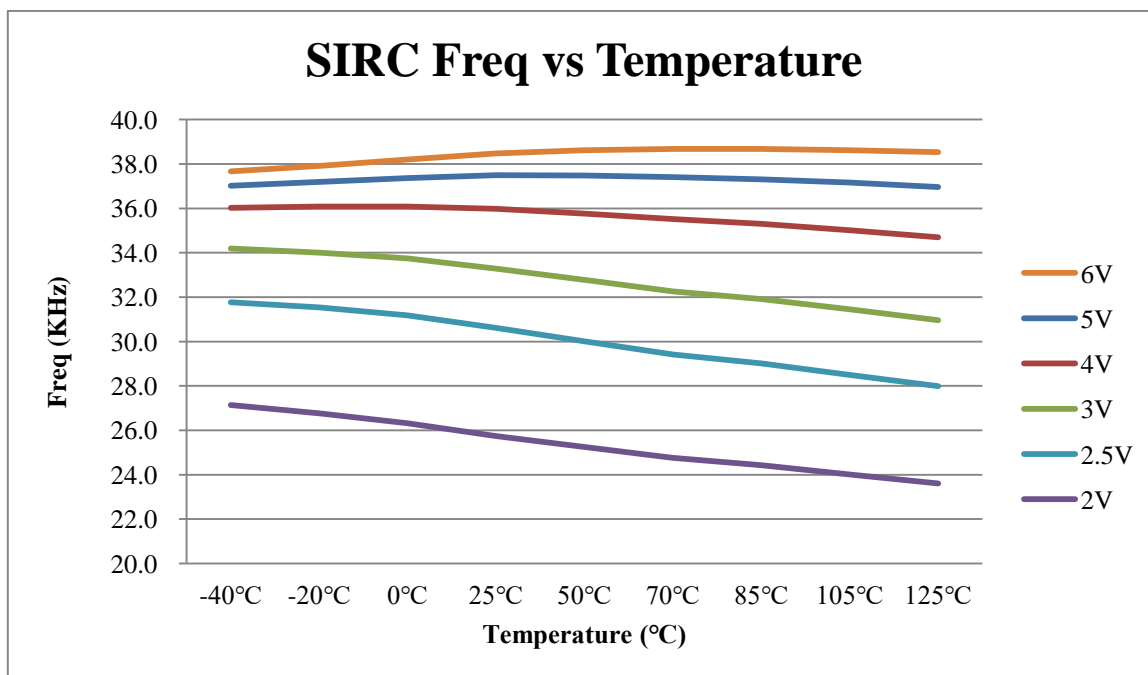
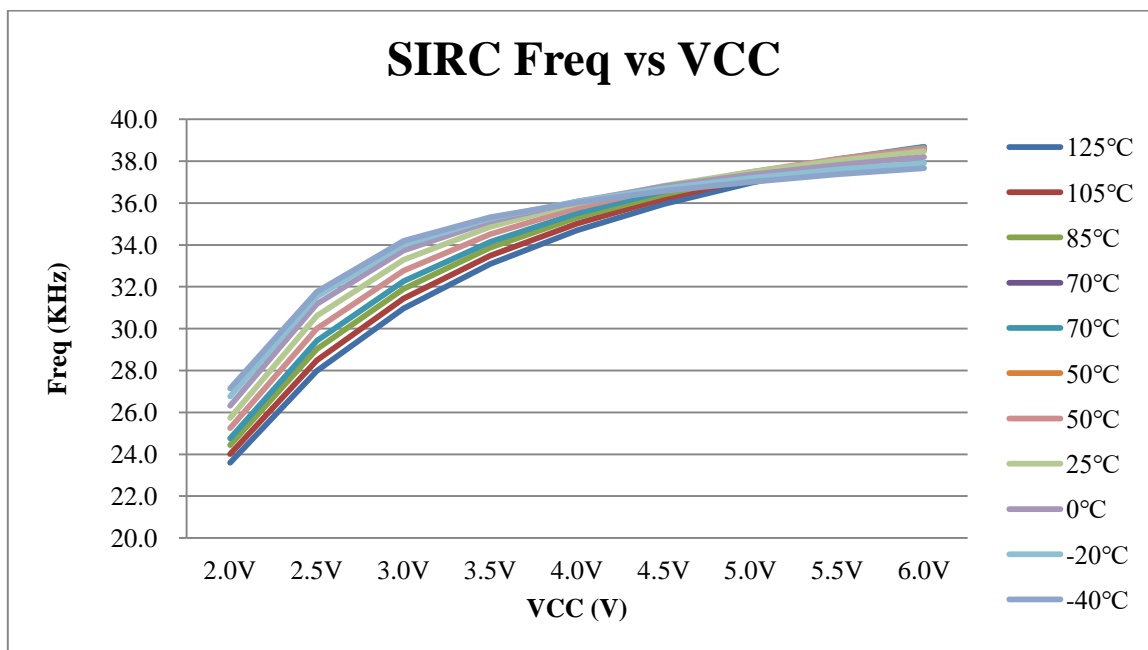
**12. Emulated EEPROM Characteristics**

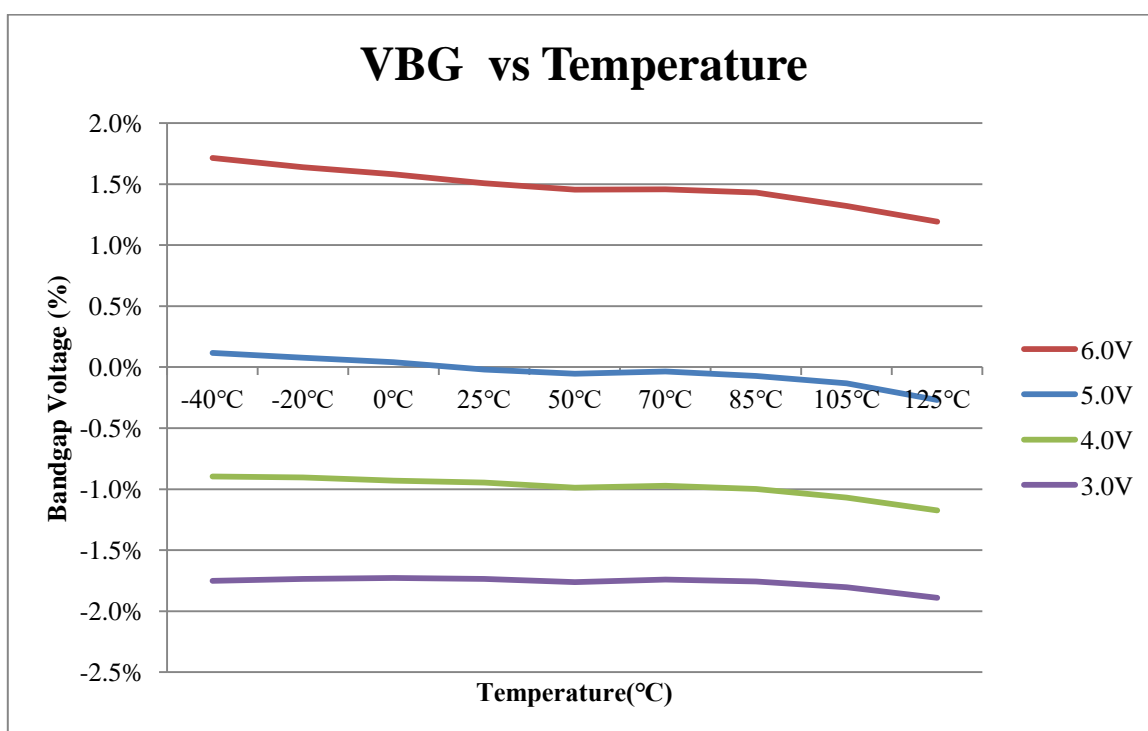
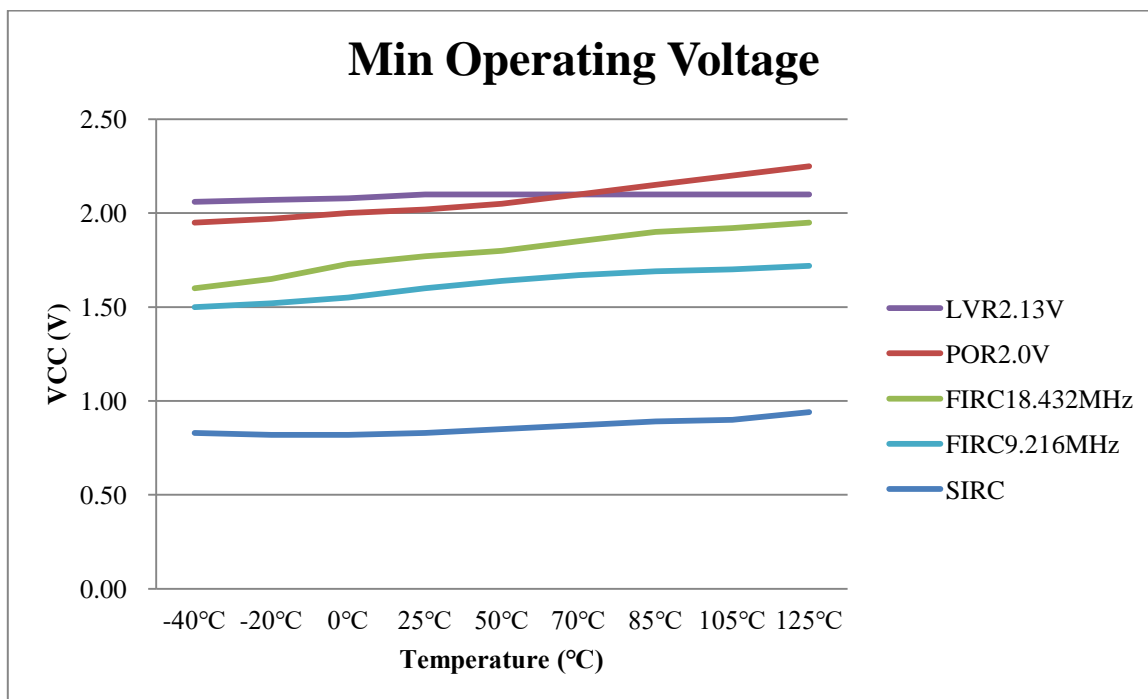
Parameter	Conditions	Min	Typ	Max	Unit
Write Voltage	$25^\circ\text{C}$	4.5	5	5.5	V
Write Endurance*	$-20^\circ\text{C}\sim 85^\circ\text{C}$	0.5K	—	—	cycles
	$25^\circ\text{C}$	1K	—	—	

\*The value of this parameter is based on the characteristics of tested samples.

## Characteristics Graphs







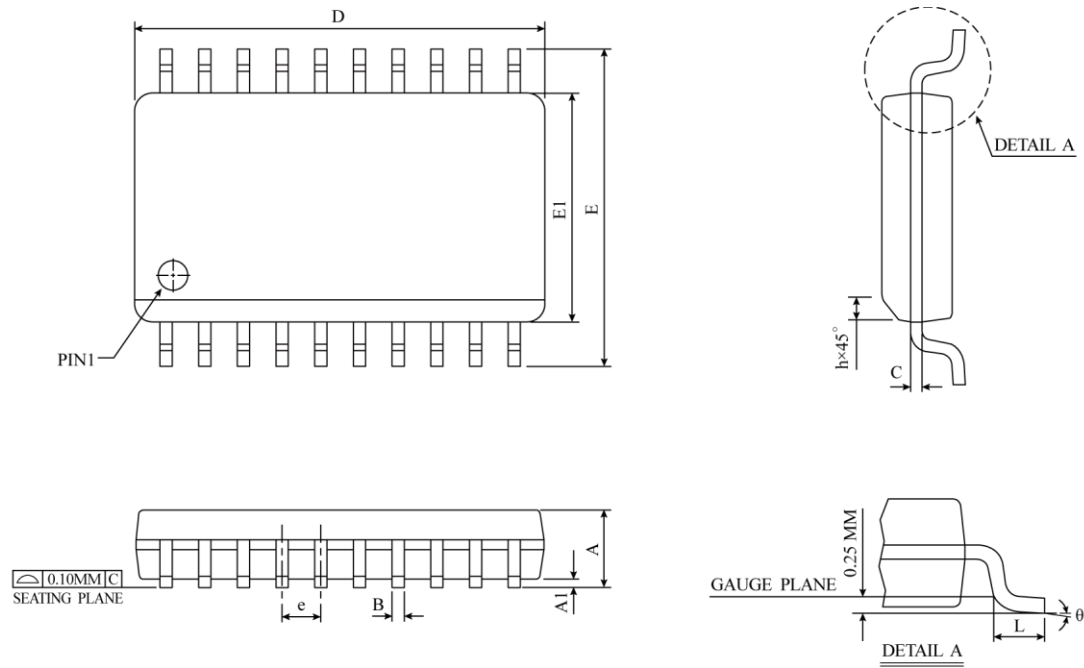
## PACKAGING INFORMATION

Please note that the package information provided is for reference only. Since this information is frequently updated, users can contact Sales to consult the latest package information and stocks.

The ordering information:

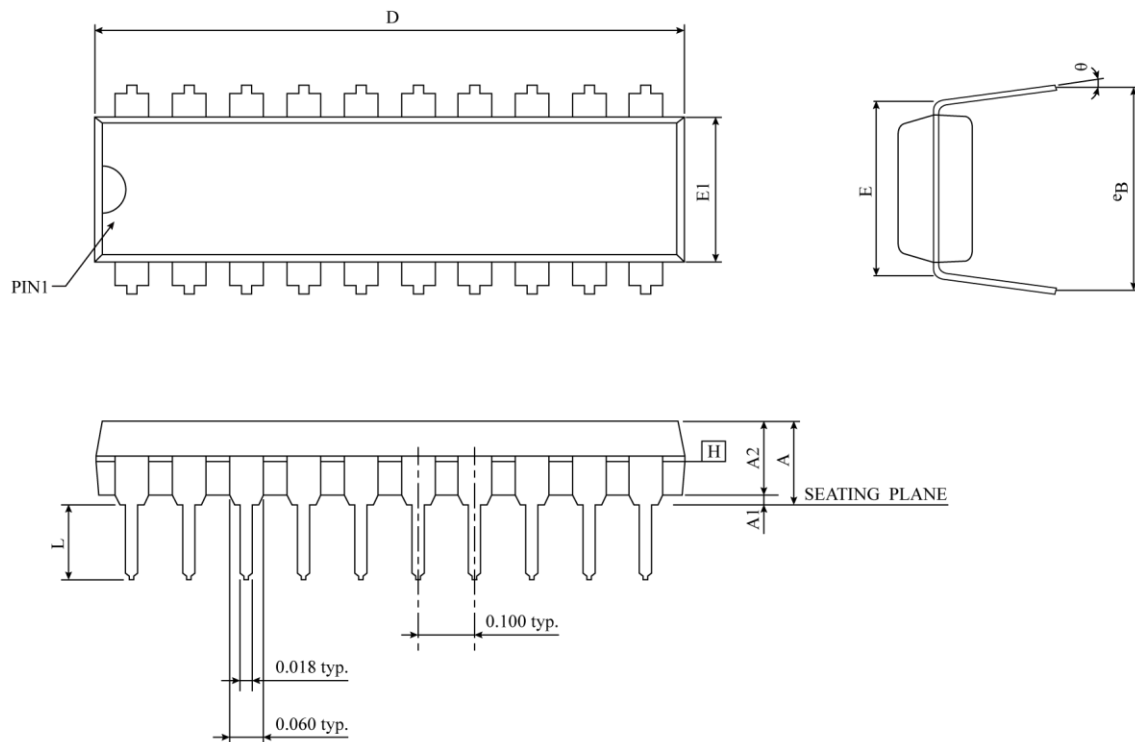
Ordering number	Package
TM56F70C23S	SOP 20-pin (300 mil)
TM56F70C23T	TSSOP 20-pin (173 mil)
TM56F70C23Q	QFN 20-pin (3*3*0.75-0.4mm)
TM56F70C22S	SOP 16-pin (150 mil)

# SOP-20 (300 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	2.35	2.50	2.65	0.0926	0.0985	0.1043
A1	0.10	0.20	0.30	0.0040	0.0079	0.0118
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.23	0.28	0.32	0.0091	0.0108	0.0125
D	12.60	12.80	13.00	0.4961	0.5040	0.5118
E	10.00	10.33	10.65	0.3940	0.4425	0.4910
E1	7.40	7.50	7.60	0.2914	0.2953	0.2992
e	1.27 BSC			0.050 BSC		
h	0.25	0.50	0.75	0.0100	0.0195	0.0290
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-013 (AC)					

△ \* NOTES : DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.  
MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL  
NOT EXCEED 0.15 MM ( 0.006 INCH ) PER SIDE.

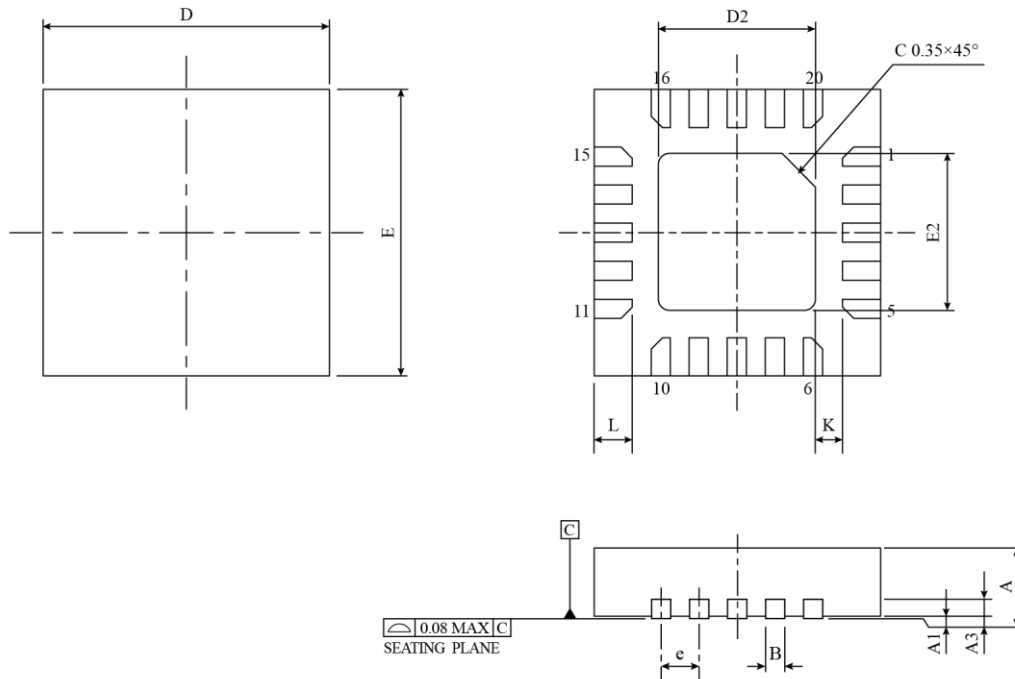
**DIP-20 (300mil) Package Dimension**


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	4.445	-	-	0.175
A1	0.381	-	-	0.015	-	-
A2	3.175	3.302	3.429	0.125	0.130	0.135
D	25.705	26.061	26.416	1.012	1.026	1.040
E	7.620	7.747	7.874	0.300	0.305	0.310
E1	6.223	6.350	6.477	0.245	0.250	0.255
L	3.048	3.302	3.556	0.120	0.130	0.140
eB	8.509	9.017	9.525	0.335	0.355	0.375
θ	0°	7.5°	15°	0°	7.5°	15°
JEDEC	MS-001 (AD)					

**NOTES :**

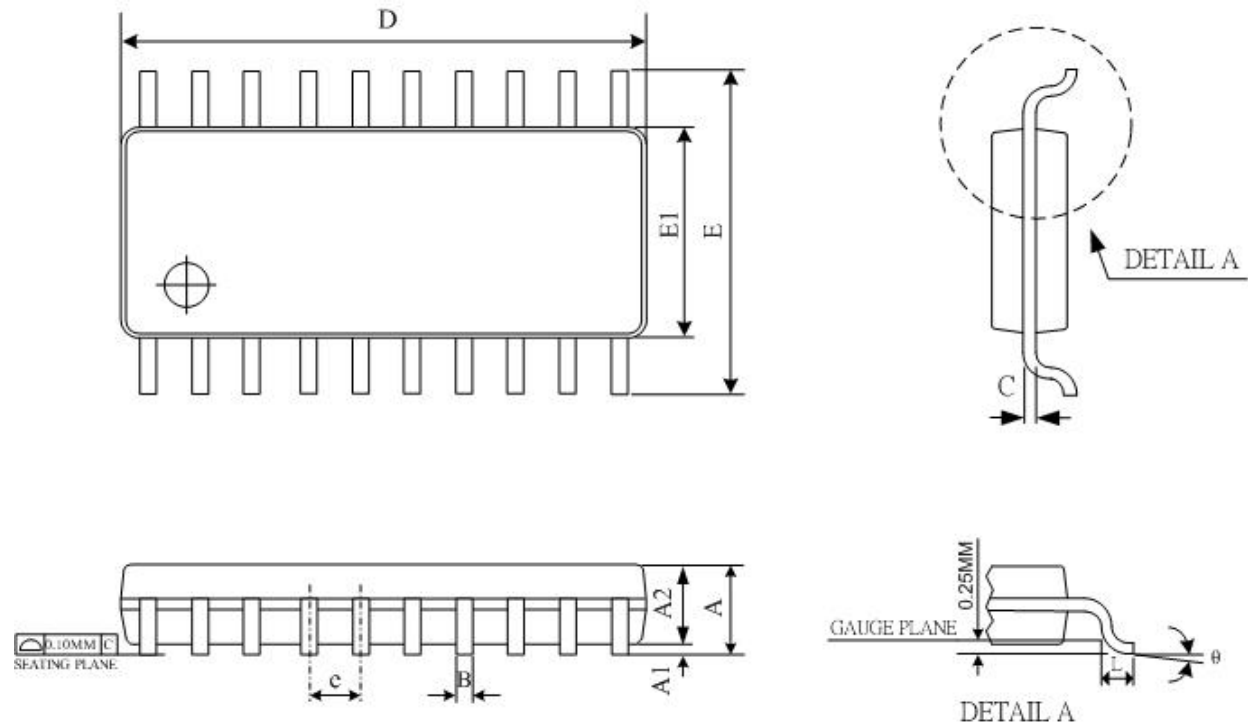
1. "D", "E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOT EXCEED .010 INCH.
2. eB IS MEASURED AT THE LEAD TIPS WITH THE LEADS UNCONSTRAINED.
3. POINTED OR ROUNDED LEAD TIPS ARE PREFERRED TO EASE INSERTION.
4. DISTANCE BETWEEN LEADS INCLUDING DAM BAR PROTRUSIONS TO BE .005 INCH MINIMUM.
5. DATUM PLANE  $\square$  COINCIDENT WITH THE BOTTOM OF LEAD, WHERE LEAD EXITS BODY.



**QFN-20 (3\*3\*0.75-0.4mm) Package Dimension**


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.70	0.75	0.80	0.028	0.030	0.031
A1	0.00	0.02	0.05	0.000	0.001	0.002
A3	0.203 REF.			0.008 REF.		
B	0.15	0.20	0.25	0.006	0.008	0.010
D	3.00 BSC			0.118 BSC		
E	3.00 BSC			0.118 BSC		
e	0.40 BSC			0.016 BSC		
K	0.20	-	-	0.008	-	-
E2	1.60	1.65	1.70	0.063	0.065	0.067
D2	1.60	1.65	1.70	0.063	0.065	0.067
L	0.30	0.40	0.50	0.012	0.016	0.020
JEDEC						

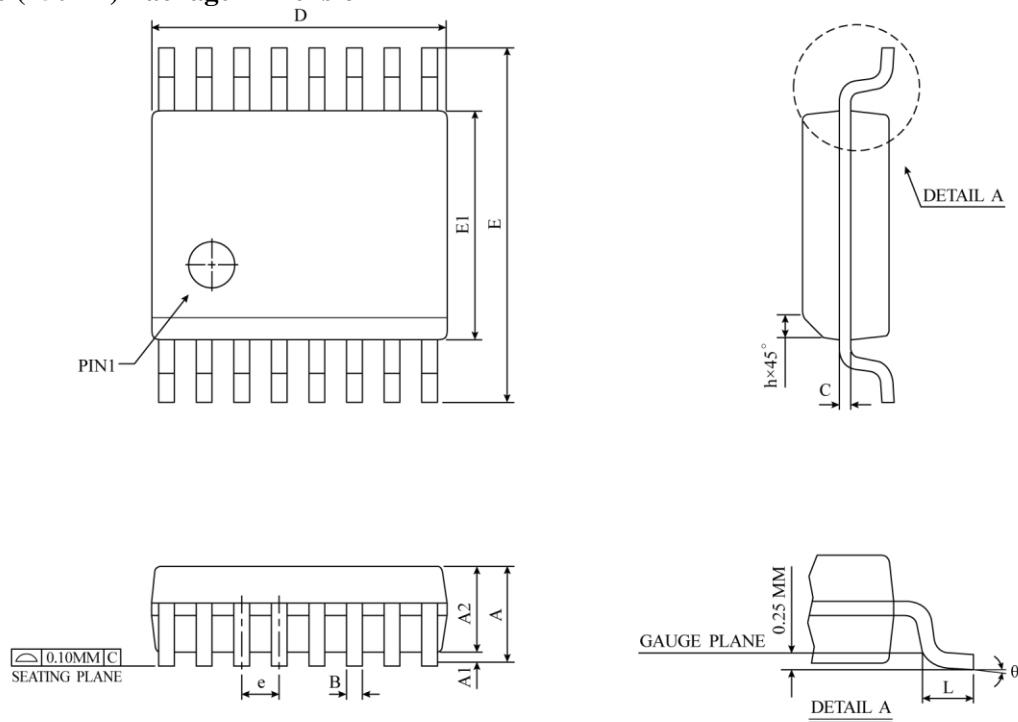
- △ \*NOTES : 1. ALL DIMENSION ARE IN MILLIMETERS.  
 2. DIMENSION B APPLIES TO METALLIZED TERMINAL AND IS MEASURED BETWEEN 0.15mm AND 0.30mm FROM THE TERMINAL TIP.  
 IF THE TERMINAL HAS THE OPTIONAL RADIUS ON THE OTHER END OF THE TERMINAL, THE DIMENSION B SHOULD NOT BE MEASURED IN THAT RADIUS AREA.  
 3. BILATERAL COPLANARITY ZONE APPLIES TO THE EXPOSED HEAT SINK SLUG AS WELL AS THE TERMINALS.

**TSSOP-20 (173 mil) Package Dimension**


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	1.2	-	-	0.047
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	0.8	0.93	1.05	0.031	0.036	0.041
B	0.19	-	0.3	0.007	-	0.012
D	6.4	6.5	6.6	0.252	0.256	0.260
E	6.25	6.4	6.55	0.246	0.252	0.258
E1	4.3	4.4	4.5	0.169	0.173	0.177
e	0.65 BSC			0.026 BSC		
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0 °		8 °	0 °		8 °
JEDEC	MO-153 AC REV.F					

**Notes :**

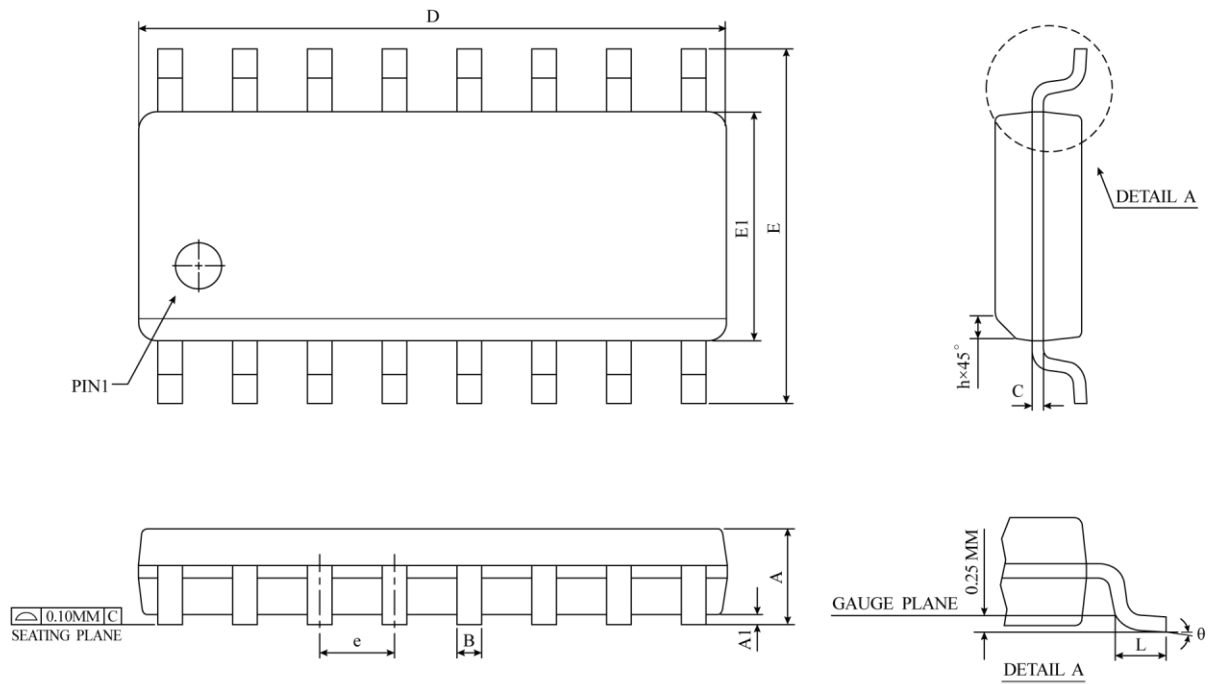
- 1.DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS OR GATE BURRS SHALL NOT EXCEED 0.15 PER SIDE.
- 2.DIMENSION "E1" DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION. INTERLEAD FLASH OR PROTRUSION SHALL NOT EXCEED 0.25 PER SIDE.
- 3.DIMENSION "B" DOES NOT INCLUDE DAMBAR PROTRUSION.ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08MM TOTAL IN EXCESS OF THE "B" DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OF THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD IS 0.07MM.

**SSOP-16 (150mil) Package Dimension**


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.053	0.061	0.069
A1	0.10	0.18	0.25	0.004	0.007	0.010
A2	-	-	1.50	-	-	0.059
B	0.20	0.25	0.30	0.008	0.010	0.012
C	0.18	0.22	0.25	0.007	0.009	0.010
D	4.80	4.90	5.00	0.189	0.193	0.197
E	5.79	6.00	6.20	0.228	0.236	0.244
E1	3.81	3.90	3.99	0.150	0.154	0.157
e	0.635 BSC			0.025 BSC		
L	0.41	0.84	1.27	0.016	0.033	0.050
θ	0°	4°	8°	0°	4°	8°
JEDEC	M0-137 (AB)					

△ \*NOTES : DIMENSION "D" DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS,  
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT  
EXCEED 0.15 MM ( 0.006 INCH ) PER SIDE.

# SOP-16 (150 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	9.80	9.90	10.00	0.3859	0.3898	0.3937
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AC)					

△ \* NOTES : DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.  
 MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL  
 NOT EXCEED 0.15 MM ( 0.006 INCH ) PER SIDE.