TM56M1511

TM56M1531

DATA SHEET

Rev 0.91

AMENDMENT HISTORY

Version	Date	Description
0.90	Mar, 2023	New Release
0.91	Mar, 2024	Update FIRC frequency accuracy(p.83)

CONTENTS

COMPARSION TABLE

	EV8239B	TM56P8336	TM56M1531
EV board	–	EV8239B	EV8239B
Touch Key	TK end of conversion, CTKCKO keep running	TK end of conversion, CTKCKO stop running and keep 1	TK end of conversion, CTKCKO stop running and keep 1

FEATURES

1. **ROM:**
   - 2K x 16 bits MTP
2. **RAM: 256 x 8 bits**
3. **STACK: 8 Levels**
4. **System Clock type selections**
   - Fast clock from Internal RC (FIRC, 16MHz)
   - Slow clock from Internal RC (SIRC, 80 KHz @4V)
5. **System Clock Prescaler**
   - System Clock can be divided by 1/2/4/8 option
6. **Power Saving Operation Mode**
   - FAST Mode: CPU running at Fast-clock
   - SLOW Mode: CPU running at Slow-clock
   - IDLE Mode: Fast-clock and CPU stop; Slow-clock keep running.
   - STOP Mode: All clocks and CPU stop.
7. **Two 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
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 / Wake-up Timer Interrupt
   - PWM Interrupt
   - Touch Key interrupt
   - LVD interrupt

9. Wake-up Timer (WKT)

- Clocked by built-in RC oscillator with 4 adjustable interrupt times
  - 12 ms / 25 ms / 50 ms / 100 ms @ 4V

10. Watchdog Timer (WDT)

- Clocked by built-in RC oscillator with 4 adjustable reset times
  - 100 ms / 200 ms / 800 ms / 1600 ms @ 4V
- Watchdog timer can be disabled / enabled in STOP mode

11. Five 16-bit PWMs (shared periods)

- PWM0, PWM1, PWM2, PWM4, PWM5
  - 16 bits, individual duty-adjustable, shared period-adjustable controlled PWM
  - PWM clock source: System clock (Fsys), FIRC (16 MHz) or FIRC*2 (32MHz)

12. Touch Key (M1531 Only)

- Two different architectures: CTK and STK
- 4 channel Touch Key with data counter
- 1 external CLD for CTK
- 1 internal reference capacitor

13. Reset Sources

- Power on Reset
- Watchdog Timer Reset
- Low Voltage Reset
- External Pin Reset

14. Low Voltage Reset (LVR) and Low Voltage Detection Flag (LVD)

- 16-Level Low Voltage Reset: 2.05V ~ 4.15V
- 15-Level Low Voltage Detection Flag: 2.20V ~ 4.15V

15.Operating Voltage

- Fsys= 16 MHz, 1.9V~5.5V @LVR disable @25°C. Suggest LVR 2.05V @-40°C~105°C

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

16. Operating Temperature Range : -40°C~105°C

17. Table Read Instruction: 16-bit ROM data lookup table

18. Integrated 16-bit Cyclic Redundancy Check (CRC) function

19. Instruction set: 39 Instructions

20. I/O ports:

- Maximum 6 programmable I/O pins
  - Open-Drain Output
  - CMOS Push-Pull Output
  - Schmitt Trigger Input with pull-up resistor option
  - All I/O with High-Sink
  - 1/2 bias output

21. Programming connectivity support 5-wire (ICP) or 8-wire program

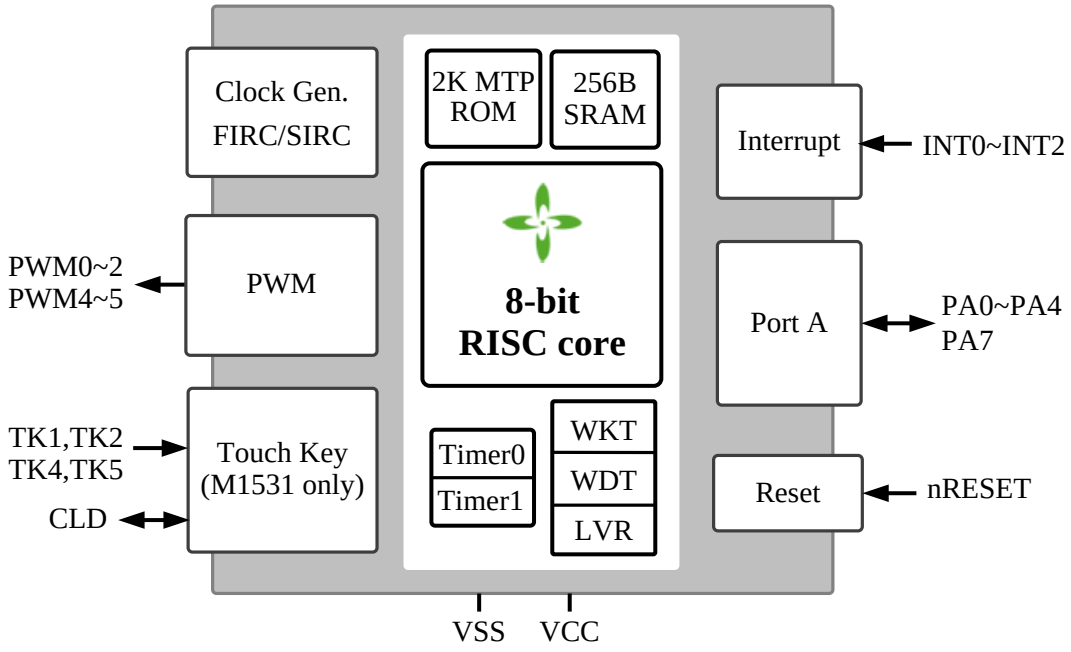
22. Package Types:

- 8-pin SOP (150 mil)
- 6-pin SOT23-6

23. Supported EV board on ICE

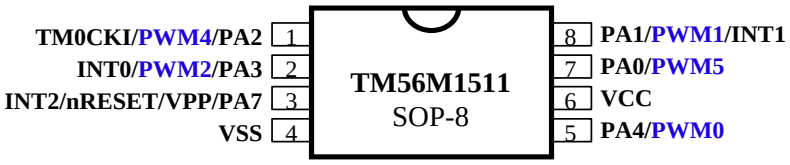
EV board: EV8239B

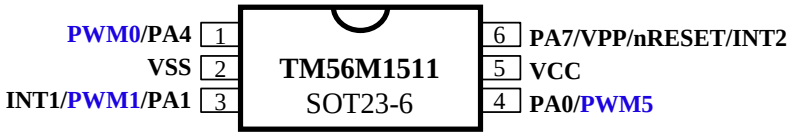
SYSTEM BLOCK DIAGRAM

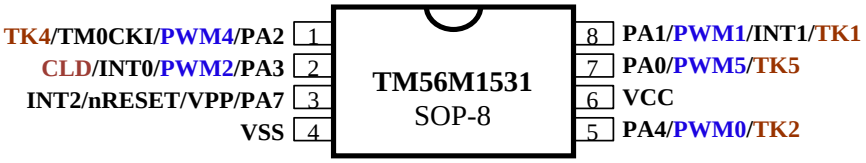


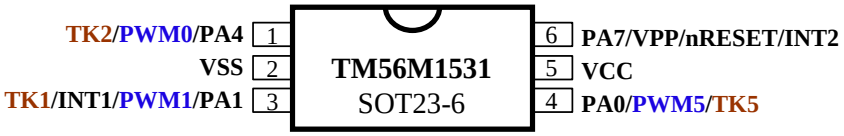
TM56M1511/31 Block Diagram

PIN ASSIGNMENT DIAGRAM









PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0~PA4, PA7	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output, open-drain output or $1/2V_{CC}$ output. Pull-up/Pull-down resistors are assignable by software.
nRESET	I	External active low reset
VCC, VSS	P	Power Voltage input pin and ground
VPP	P	Program high voltage input
INT0–INT2	I	External interrupt input
TM0CKI	I	Timer0's input in counter mode
PWM0~2,PWM4~5	O	16 bit PWM output
TK1,TK2,TK4,TK5	I	Touch Key input
CLD	I/O	Touch Key charge collection capacitor connection pin

Programming pins:

Normal mode: VCC / VSS / VPP / PA0 / PA1 / PA2 / PA3 / PA4

ICP mode: VCC / VSS / VPP / PA0 / PA1 - When using ICP (In-circuit Program) mode, the PCB needs to remove all components of PA0, PA1.

PIN SUMMARY

SOP8-A	Pin Name	Type	GPIO			Alternate Function		
			Input	Output				
TM56M1511/31			Ext. Interrupt	O.D	P.P	PWM	TK (M1531 Only)	MISC
1	PA2/PWM4/TM0CKI/TK4	I/O		○	○	○	○	TM0CKI
2	PA3/PWM2/INT0/CLD	I/O	○	○	○	○	○	
3	PA7/VPP/nRESET/INT2	I/O	○	○	○			nRESET
4	VSS	P						
5	PA4/PWM0/TK2	I/O		○	○	○	○	
6	VCC	P						
7	PA0/PWM5/TK5	I/O		○	○	○	○	
8	PA1/PWM1/INT1/TK1	I/O	○	○	○	○	○	

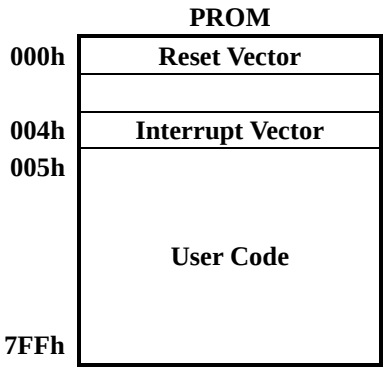
Symbol：P.P. = COM Push-Pull Output
O.D. = Open Drain Output

FUNCTION DESCRIPTION

1 CPU Core

1.1 Program ROM

The MTP (Multi-time Programmable) ROM are 2K words, with an extra INFO area to store the SYSCFG. The ROM can be written as long as the **PROT** (CFGWH[15]) bit of SYSCFG is not 1.



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.

1.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at INFO area; it contains a 16 bits register (CFGWH). The SYSCFG determines the option for initial condition of CPU. User can select LVR operation Mode and chip operation mode by SYSCFG register. The 15th bit of CFGWH is code-protected selection bit. If this bit is 1, the data in PROM will be protected when user reads PROM.

Bit		**15~0**	
Default Value		**1111_1111_1111_1111**	
Bit		Description	
CFGWH	15	**PROT**: Code protection selection	
		0	Disable
		1	Enable
	14	Tenx Reserved	
		0	
		1	
	13-12	**WDTE**: WDT Reset Enable	
		0X	Disable
		10	Enable in FAST/SLOW mode, Disable in IDLE/STOP mode
		11	Always Enable
	11-8	**LVR**: Low Voltage Reset Mode	
		0000	Low Voltage Reset 2.05V
		0001	Low Voltage Reset 2.20V
		0010	Low Voltage Reset 2.30V
		0011	Low Voltage Reset 2.45V
		0100	Low Voltage Reset 2.60V
		0101	Low Voltage Reset 2.75V
		0110	Low Voltage Reset 2.90V
		0111	Low Voltage Reset 3.00V
		1000	Low Voltage Reset 3.15V
		1001	Low Voltage Reset 3.30V
		1010	Low Voltage Reset 3.45V
		1011	Low Voltage Reset 3.60V
		1100	Low Voltage Reset 3.70V
		1101	Low Voltage Reset 3.85V
		1110	Low Voltage Reset 4.00V
		1111	Low Voltage Reset 4.15V
	7	**XRSTE**: External Pin (PA7) Reset Enable	
		0	Disable (PA7 as I/O pin)
		1	Enable
	6-0	Tenx Reserved	

1.3 RAM Addressing Mode

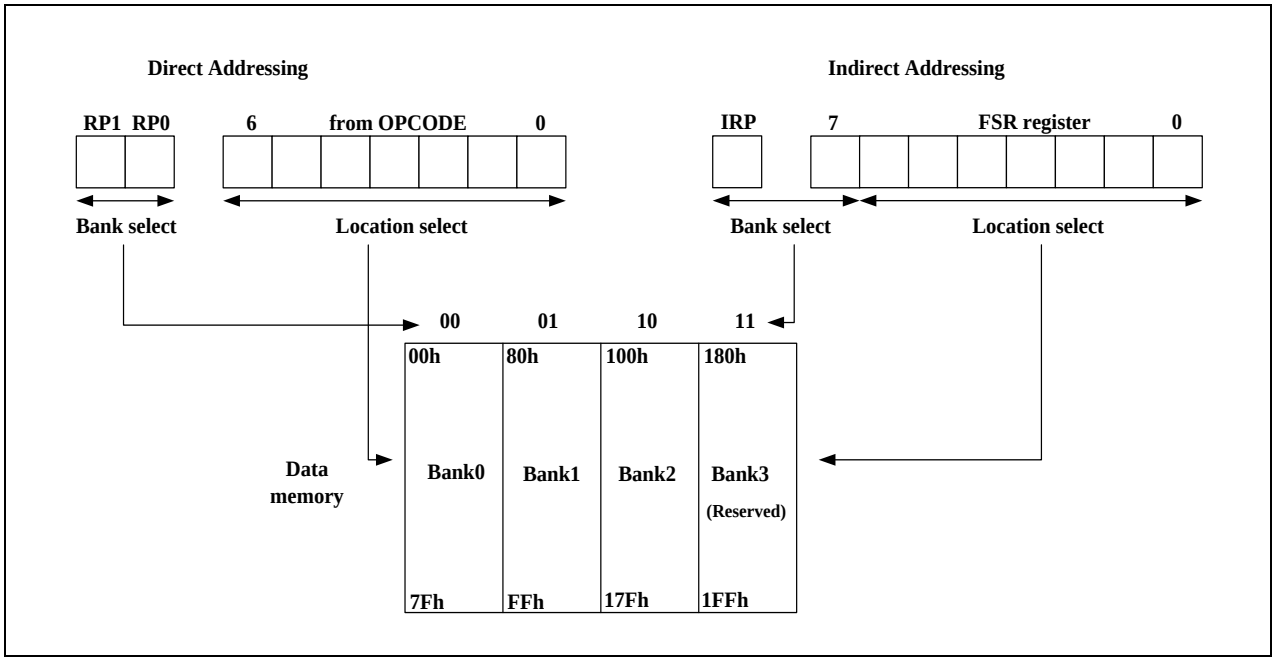
There is one Data Memory Plane in CPU. The Plane is partitioned into four banks. Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for Special Function Register (SFR). Above the SFR are General Purpose Registers, implemented as static RAM. All implemented banks contain Special Function Registers. Some frequently used Special Function Registers from one bank may be mirrored in another bank for code reduction and quicker access.

Bit RP1 and RP0 (STATUS[6:5]) are the bank select bits.

[RP1, RP0]	BANK
00	0
01	1
10	2
11	3

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 the File Select Register, FSR.

If FSR=0, that is, pointing to a virtual location, then reading the FSR will get 0, and writing the FSR will be an invalid instruction (although the 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

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.

For example:

BC**F**	TM0IE	➔	BC**X**	TM0IE
DEC**F**	CNT, 1	➔	DEC**X**	CNT, 1
INC**F**SZ	RAM25, 0	➔	INC**X**SZ	RAM25, 0
MOVW**F**	PAMODL	➔	MOVW**X**	PAMODL
RL**F**	RAMA0, 0	➔	RL**X**	RAMA0, 0
SWAP**F**	ADCTL, 0	➔	SWAP**X**	ADCTL, 0

【BANK0】 00~7Fh		【BANK1】 80h~FFh		【BANK2】 100h~17Fh		【BANK3】 180h~1FFh	
00h	**INDF**	80h	**INDF**	100h	**INDF**	180h	**INDF**
01h	TM0	81h	OPTION	101h	TM0	181h	OPTION
02h	**PCL**	82h	**PCL**	102h	**PCL**	182h	**PCL**
03h	**STATUS**	83h	**STATUS**	103h	**STATUS**	183h	**STATUS**
04h	**FSR**	84h	**FSR**	104h	**FSR**	184h	**FSR**
05h	PAD	85h	PAMOD10	105h		185h	DPL
06h		86h	PAMOD32	106h		186h	DPH
07h		87h	PAMOD54	107h		187h	CRCDL
08h		88h	PAMOD76	108h		188h	CRCDH
09h		89h	PWMCTL	109h	LVRPD	189h	CRCIN
0Ah	**PCLATH**	8Ah	**PCLATH**	10Ah	**PCLATH**	18Ah	**PCLATH**
0Bh	**INTIE**	8Bh	**INTIE**	10Bh	**INTIE**	18Bh	**INTIE**
0Ch	INTIF	8Ch		10Ch	PCH	18Ch	TABR
0Dh	INTIE1	8Dh		10Dh		18Dh	
0Eh	INTIF1	8Eh		10Eh	BGTRIM	18Eh	
0Fh	CLKCTL	8Fh		10Fh	IRCF	18Fh	
10h	TM0RLD	90h		110h		190h	
11h	TM0CTL	91h	OPTION2	111h		191h	
12h	TM1	92h	PWMPRDH	112h		192h	
13h	TM1RLD	93h	PWMPRDL	113h		193h	
14h	TM1CTL	94h	PWM0DH	114h		194h	
15h		95h	PWM0DL	115h		195h	
16h	LVCTL	96h	PWM1DH	116h		196h	
17h		97h	PWM1DL	117h		197h	
18h		98h	PWM2DH	118h		198h	
19h		99h	PWM2DL	119h		199h	
1Ah	TKDL	9Ah		11Ah		19Ah	
1Bh	TKDH	9Bh		11Bh		19Bh	
1Ch	TKTMRL	9Ch	PWM4DH	11Ch		19Ch	
1Dh	TKMFS	9Dh	PWM4DL	11Dh		19Dh	
1Eh	TKCTL	9Eh	PWM5DH	11Eh		19Eh	
1Fh	TKCTL2	9Fh	PWM5DL	11Fh		19Fh	
20h	RAM Bank0 area (80 Bytes)	A0h	RAM Bank1 area (80 Bytes)	120h	RAM Bank2 area (80 Bytes)	1A0h	RAM Bank3 area (80 Bytes)
6Fh		EFh		16Fh		1EFh	
70h	**common area 16 Bytes**	F0h	**accesses 70h~7Fh**	170h	**accesses 70h~7Fh**	1F0h	**accesses 70h~7Fh**
7Fh		FFh		17Fh		1FFh	

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

```
TM1       equ     12H       ;SFR in Bank0
OPTION2   equ     91H       ;SFR in Bank1
IRCF      equ     10FH      ;SFR in Bank2
DPL       equ     185H      ;SFR in Bank3
RAM20     equ     20H       ;RAM in Bank0
RAMA0     equ     A0H       ;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
MOVWX     RAM20             ; W = 16h write to RAM[0x20]
MOVWX     RAMA0             ; W = 16h write to RAM[0xA0]

MOVLW     37H
MOVWX     LVRPD             ; LVRPD = W = 37h, force LVR/POR disable

MOVXW     CLKCTL            ; read SFR CLKCTL (0Fh) to W
MOVXW     IRCF              ; read SFR IRCF (10Fh) to W

MOVLW     0BH
MOVWX     CLKCTL            ; CLKCTL (0Fh) = 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 (0Fh) to W

BCX       IRP               ; IRP = 0 => Bank0/1
MOVLW     0FH               ; W = 0Fh
MOVWX     FSR               ; FSR = W =0Fh
MOVLW     0BH               ; W = 0Bh
MOVWX     INDF              ; CLKCTL (0Fh) = W = 0Bh
```

1.4 Programming Counter (PC) and Stack

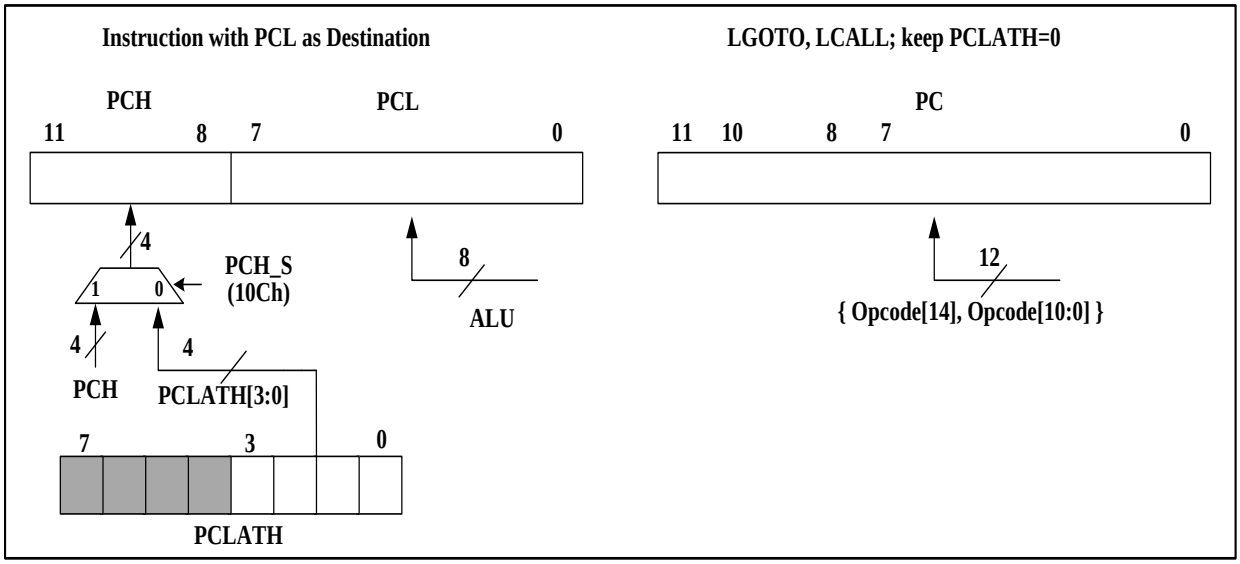
The Programming Counter is 11-bit wide and capable of addressing a 2K x 16 MTP ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except for the following cases. The Reset Vector (000h) and the Interrupt Vector (004h) are provided for PC initialization and Interrupt. For CALL/GOTO instruction, PC loads lower 11 bits address from instruction word. For RET/RETI/RETLW instruction, PC retrieves its content from the top level STACK.

Similar as RAM Addressing Mode (refer 1.3), the Chip provides new instruction set LCALL/LGOTO to replace CALL/GOTO instruction set.

The low byte data of the Programming Counter (PC[7:0]) can be read and written by PCL register (002h/082h/102h/182h). The high byte data of Programming Counter (PC[11:8]) can only be read by PCH register (10Ch). The internal flag PCH_S is used to select the source of PCH, when executing any instruction with the PCL register as the destination. Write 0x1C to PCH register can set PCH_S, write others value to PCH register will clear PCH_S. After reset, the PCH_S is cleared.

When PCH_S is cleared to '0', executing any instruction with the PCL register as the destination simultaneously causes PCH to be replaced by the contents of the PCLATH (00Ah/08Ah/10Ah/18Ah) register. This allows the entire contents of the program counter to be changed by writing the desired high byte to the PCLATH register. When the low byte is written to the PCL register, all contents of program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

When PCH_S is setting to '1', executing any instruction with the PCL register as the destination the low byte is written to the PCL register and will not change the PCH. It is recommended to setting PCH_S to '1' when using any instruction with the PCL register as the destination, but C language doesn't support this function.



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 data bit 7~0

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

00Ah.2~0 **PCLATH:** Write Buffer for the high byte of the Program Counter.

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

10Ch.7~0 **PCH (W):** Programming Counter high byte source selection when instruction with PCL as destination is executed
write 0x1C to set PCH_S = 1: PCH keep the original value
write others to clear PCH_S = 0: PCH is from PCLATH

10Ch.3~0 **PCH (R):** Programming Counter data bit 11~8

The STACK is 11-bit wide and 8-level in depth. The LCALL instruction and hardware interrupt will push STACK level in order, while the RET/RETI/RETLW instruction pops STACK level in order. For table lookup, the device offer the powerful table read instructions TABRL, TABRH to return the 16-bit ROM data into W register by setting DPTR={DPH, DPL} registers. It also offers another way to read the16-bit ROM data into W register by setting TABR (18Ch) for C language.

Example: To look up the PROM data located “TABLE1” and “TABLE2”.

```
ORG         000h                    ;  Reset Vector
            LGOTO       START

START:
            MOVLW       00h
            MOVWX       INDEX       ; Set lookup table’s address
            MOVLW       1Ch         ; Write 1Ch to PCH to set PCH_S flag
            MOVWX       PCH
LOOP:
            MOVXW       INDEX       ; Move index value to W register
            LCALL       TABLE1      ; To lookup data
            …
            INCX        INDEX, 1    ; Increment the index address for next address
            …
            LGOTO       LOOP        ; Go to LOOP label
            …
            MOVLW       (TABLE2 >>8) & 0xff
            MOVWX       DPH         ; DPH register (F186h.3~0)
            MOVLW       (TABLE2) & 0xff
            MOVWX       DPL         ; DPL register (F185h.7~0)
; Table Read by instructions TABRL / TABRH
            TABRL                   ; Read PROM low byte data to W (W = 86h)
            TABRH                   ; Read PROM high byte data to W (W = 19h)
```

```
                …
; Table Read by SFR TABR
                MOVLW     01h             ; TABR = 01h = instruction TABRL
                MOVWX     TABR            ; Read PROM low byte data to TABR (TABR = 86h)
                MOVXW     TABR            ; Read TABR to W (W = 86)
                MOVLW     02h             ; TABR = 02h = instruction TABRH
                MOVWX     TABR            ; read PROM high byte data to TABR (TABR = 19h)
                MOVXW     TABR            ; read TABR to W (W = 19)
                …
ORG             X00h
TABLE1:
                ADDWX     PCL, 1          ; Add the W with PCL, the result back in PCL.
                RETLW     55h             ; W=55h when return
                RETLW     56h             ; W=56h when return
                RETLW     58h             ; W=58h when return
                …
TABLE2:
                .DT       0x1986          ; 16-bit ROM data
                .DT       0x3719
                …
```

Note: The chip define 256 ROM address as one page, so that ROM has 8 pages, 000h~0FFh, 100h~1FFh, …, 700h~7FFh. On the other words, PC[10:8] can be define as page. A lookup table must be located at the same page to avoid getting wrong data. Thus, the lookup table has maximum 255 data for above example with starting a lookup table at X00h (X = 1, 2, 3, …, 6, 7). If a lookup table has fewer data, it needs not setting the starting address at X00h, but only confirms all lookup table data are located at the same page.

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 1. TABR write 01h = instruction TABRL
2. TABR write 02h = instruction TABRH
3. After step.1 or step.2, read TABR to get main ROM table read value
After step.1, read TABR to get EEPROM value (when EEPEN = E2h)

Table Read for ASM: instruction TABRL / TABRH or register TABR

Table Read for C: using register TABR

1.4.1 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.

1.4.2 STATUS Register (03H/83H/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	**IRP**: Register Bank Select bit (used for indirect addressing) 0 = Bank 0,1 (00h - FFh) 1 = Bank 2,3 (100h - 1FFh)							
6:5	**RP1:RP0**: Register Bank Select bits (used for direct addressing) 00 = Bank 0 (00h - 7Fh) 01 = Bank 1 (80h - FFh) 10 = Bank 2 (100h - 17Fh) 11 = Bank 3 (180h - 1FFh) Each bank is 128 bytes							
4	**TO**: Time Out Flag 0: after Power On Reset or CLRWDT/SLEEP instruction 1: WDT time out occurs							
3	**PD**: Power Down Flag 0: after Power On Reset or CLRWDT instruction 1: after SLEEP instruction							
2	**Z**: Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero							
1	**DC**: 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	**C**: 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
```

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)

A reset may be caused by a power on reset (POR), an external pin reset (XRST), a watchdog timer reset (WDTR), or a low voltage reset (LVR). CFGWH controls the reset function. After reset, the SFR is restored to its default value, the program counter (PC) is cleared, and the system starts to run from the reset vector 000H. The TO flag of the status register (STATUS) indicates the system watchdog timer 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 CFGWH register. See the following LVR Table. User must also consider the lowest operating voltage of operating frequency.

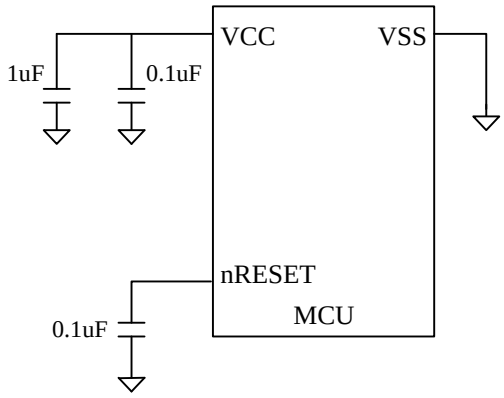
CFGWH.11~8	LVR level
0000	Low Voltage Reset 2.05V
0001	Low Voltage Reset 2.20V
0010	Low Voltage Reset 2.30V
0011	Low Voltage Reset 2.45V
0100	Low Voltage Reset 2.60V
0101	Low Voltage Reset 2.75V
0110	Low Voltage Reset 2.90V
0111	Low Voltage Reset 3.00V
1000	Low Voltage Reset 3.15V
1001	Low Voltage Reset 3.30V
1010	Low Voltage Reset 3.45V
1011	Low Voltage Reset 3.60V
1100	Low Voltage Reset 3.70V
1101	Low Voltage Reset 3.85V
1110	Low Voltage Reset 4.00V
1111	Low Voltage Reset 4.15V

Different F_{SYS} 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)

The External Pin Reset can be disabled or enabled by XRSTE at CFGWH register. External pin reset should be kept low for at least 2 SIRC clock cycles to ensure reset can active. The External Pin Reset also sets all the control registers to their default value but the TO/PD flags will not affected by these resets.

External reset pin (nRESET) is low level active. The system is running when reset pin is high level voltage input. The reset pin receives the low voltage and the system is reset. The external reset can reset the system during power on duration, and good external reset circuit can protect the system to avoid operating at inappropriate power condition.



2.4 Watchdog Timer Reset (WDTR)

The WDT reset can be disabled or enabled through the CFGWH register. Set WDTPSC (81h.3~2) to define the period during which WDT reset occurs. WDT reset counter can be cleared by device Reset or CLRWDT instruction. WDT reset also set all the control registers to their default value.

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

Mode	CFGWH[13:12]		Watch dog Timer
	WDTE[1]	WDTE[0]	
Normal Mode	0	0	Stop
	0	1	Stop
	1	0	Run
	1	1	Run
IDLE/STOP 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   0000011b
MOVWX   OPTION          Select WDT Time out=200 ms @4V
…
```

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
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	-	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	-	R/W		R/W	R/W
Reset	0	0	0	-	1	1	1	1

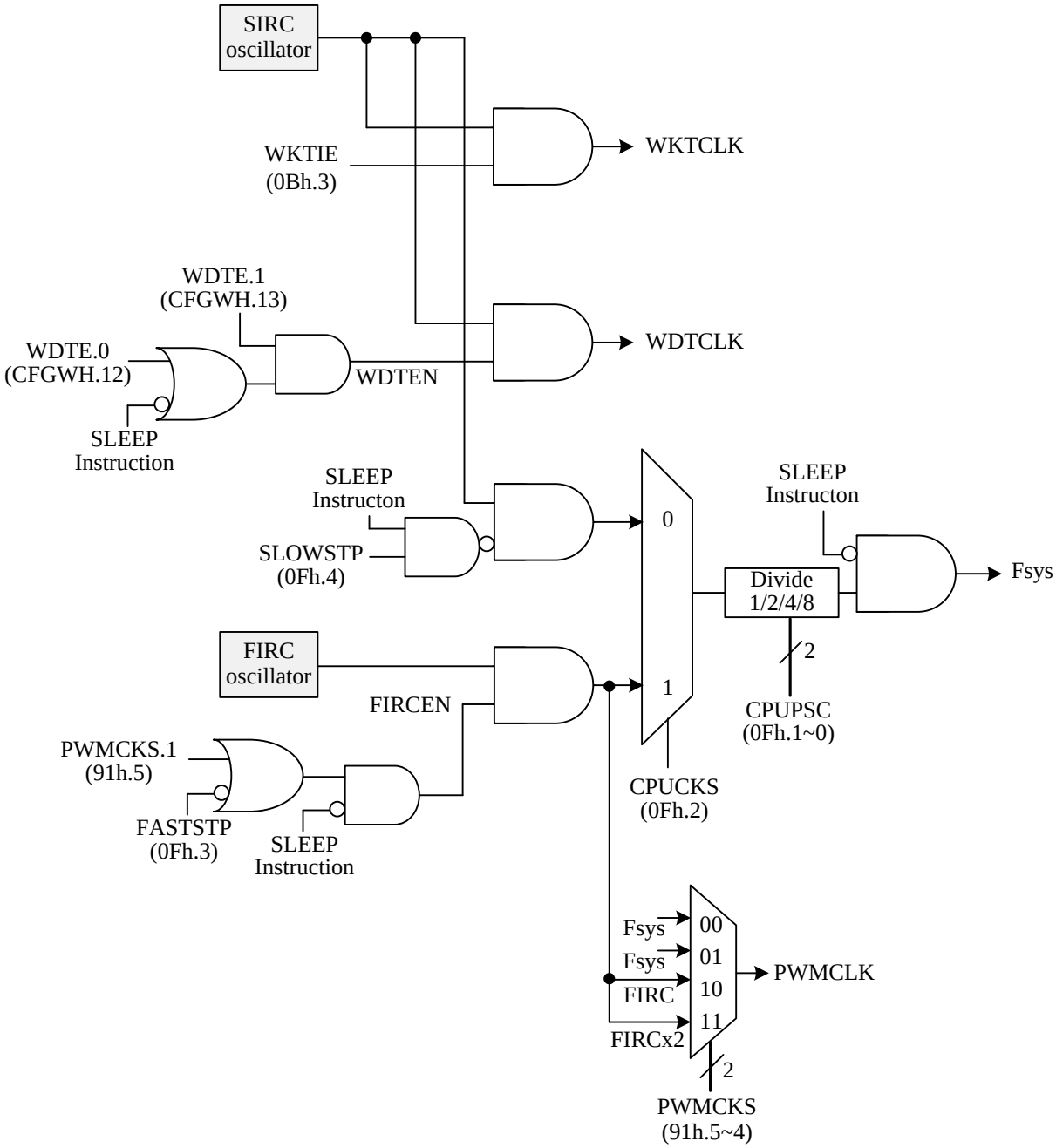
81h.3~2 **WDTPSC:** WDT period
00: 100 ms 01: 200 ms 10: 800 ms 11: 1600 ms @VCC=4V

3 Clock Circuitry and Operation Mode

3.1 System Clock

The device is designed with a dual clock system. There are two clock sources, SIRC (Slow Internal RC) clock and FIRC (Fast Internal RC) clock. Each clock source can be applied to the CPU core as a system clock (Fsys).

When the user issues the SLEEP command, the system will enter the IDLE mode or the STOP mode. When the system does not use the SIRC oscillator at all, it is the most power-saving at this time, which we call the STOP mode. Refer to the Figure as below.



Clock Scheme Block Diagram

After a CPU reset, the device runs in slow mode with 90 KHz SIRC. The user should select an appropriate clock frequency for the safety of chip operation. A higher VCC allows the chip to run at a higher system clock frequency.

CLKCTL (0Fh) SFR controls system clock operation. The hardware will automatically prevent out-of-spec settings for this register. It is recommended that users write this SFR bit by bit, and do not set FASTSTP=1 and CPUCKS=1 at the same time.

The frequency of FIRC can be adjusted by IRCF (10Fh). With this function, we can adjust the frequency of FIRC after power up. Each IC may have a different IRCF default value to ensure FIRC=16 MHz after power-on reset.

FAST Mode:

In this mode, the program is executed using FIRC as CPU clock (F_{SYS}). The Timer0, Timer1 blocks are also driven by Fast-clock. The PWM0 block can be driven by Fsys, FIRC (16MHz), or FIRC*2 (32 MHz) by setting PWMCKS (91h.5~4).

SLOW Mode:

After power-on or reset, the system clock (Fsys) uses a slow clock by default, and the user can stop Fast-clock (by FASTSTP=1, for power saving) or run (by FASTSTP=0). Except that the PWM module can select other clock sources, the clock sources of all peripheral modules (Timer0, Timer1, etc.) at this time use slow clocks.

IDLE/STOP Mode:

After the user executes the SLEEP instruction, the device will turn off the system clock. If the SIRC oscillator is still running, it is called IDLE mode, if the SIRC oscillator is stopped, it is called STOP mode.

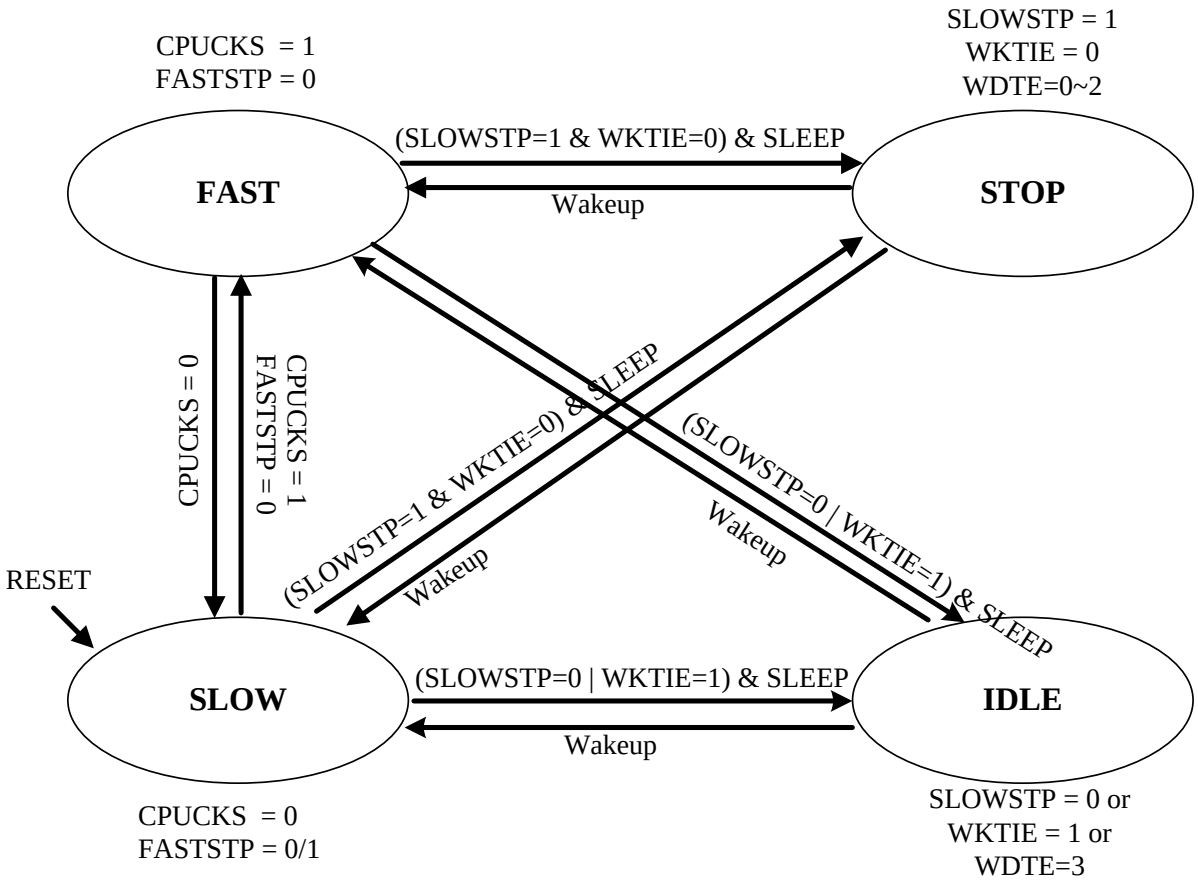
SIRC oscillators are used in the following projects:

1. WKT (controlled by WKTIE)
2. WDT (controlled by WDTE)
3. System Clock (Fsys)

When the above items are all stopped, and SLOWSTP=1, the SIRC oscillator will be stopped, the device is the most power-saving at this time, called STOP mode.

3.2 Dual System Clock Modes Transition

The device is operated in one of four modes: FAST mode, SLOW mode, IDLE mode, and STOP mode.



Note:

- SLEEP denotes SLEEP instruction
- Wakeup denotes wake-up events, include INT2~0, Pin wake up and WKT interrupt.
- SLOWSTP (0Fh.4), FASTSTP (0Fh.3), CPUCKS (0Fh.2), WKTIE (0Bh.3)
- CPU will not enter STOP mode if WDTE (CFGWH.13~12) = 3

CPU Operation Block Diagram

CPU Mode & Clock Functions Table:

Mode	FIRC oscillator	SIRC oscillator	Fsys/ TM0/TM1	WKT	WDT	How to wake up
FAST	Run	Run	Run	Run	Run	-
SLOW	Set by FASTSTP	Run	Run	Run	Run	-
IDLE	Stop	Run	Stop	Set by WKTIE	Set by WDTE	WKT/IO
STOP	Stop	Stop	Stop	Stop	Stop	IO

- **FAST mode switches to SLOW mode**

The following steps are suggested to be executed by order when FAST mode switches to SLOW mode:

(1) Switch to Slow-clock (CPUCKS=0)

(2) Stop Fast-clock (FASTSTP=1)

◇Example: Switch FAST mode to SLOW mode.

```
BCX     CPUCKS      ; Fsys=Slow-clock
BSX     FASTSTP     ; Disable Fast-clock
```

- **SLOW mode switches to FAST mode**

SLOW mode can be enabled by CPUCKS=0 in CLKCTL register. The following steps are suggested to be executed by order when SLOW mode switches to FAST mode:

(1) Enable Fast-clock (FASTSTP=0)

(2) Switch to Fast-clock (CPUCKS=1)

◇Example: Switch SLOW mode to FAST mode (The Fast-clock stop).

```
BCX     FASTSTP     ; Enable Fast-clock
NOP
BSX     CPUCKS      ; Fsys=Fast-clock
```

- **IDLE/STOP mode Setting**

The IDLE mode can be configured by following setting in order:

(1) Enable WKT (WKTIE=1)

(2) Execute SLEEP instruction

IDLE mode can be woken up by External interrupt, Pin wake up or WKT interrupt.

◇Example: Switch FAST/SLOW mode to IDLE mode.

```
BSX     WKTIE       ; Enable WKT counting
SLEEP               ; Enter IDLE mode
```

- **STOP Mode Setting**

The STOP mode can be configured by following setting in order:

(1) Stop Slow-clock (SLOWSTP=1)

(2) Stop WKT (WKTIE=0)

(3) Execute SLEEP instruction

STOP mode can be woken up only by External pin interrupt or Pin wake up.

Note: CPU will not enter STOP mode if WDTE (CFGWH.13~12) = 3

◇ Example: Switch FAST/SLOW mode to STOP mode.

```
BSX     SLOWSTP     ; Disable Slow-clock after execute SLEEP instruction
BCX     WKTIE       ; Disable WKT counting
SLEEP               ; Enter STOP mode.
```

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

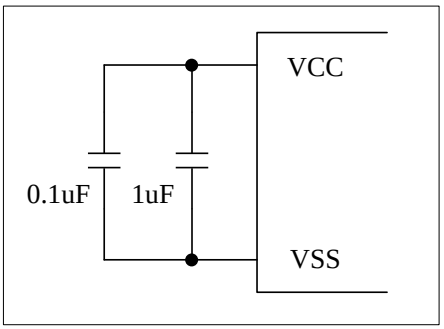
0Fh.3 **FASTSTP**: Fast-clock stop
0: Fast-clock is running
1: Fast-clock stops running

0Fh.2 **CPUCKS**: System clock source select
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

3.3 Bypass capacitors

In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 16 MHz system clock. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VCC/VSS pins improves the stability of clock and the overall system.



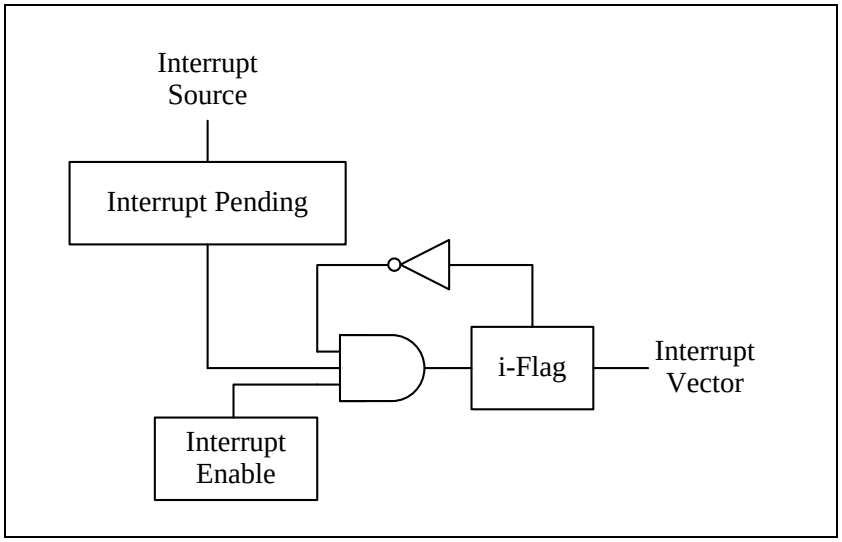
Internal RC Mode

4 Interrupt

TM56M1511 has 1 level, 1 vector and 9 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 (INTIE.5~0, INTIE1.2~0) 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           ; Go to 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   111111 0 1b
                MOVWX   INTIF           ; Clear INT1 interrupt request flag
                MOVLW   000000 1 0b
                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_SUBROUTINE ; if INT1IF = 1, jump to INT1 subroutine
                …                       ;
                MOVXW   21h             ; Get SRAM 21h data
                MOVWX   STATUS          ; Restore STATUS data
                MOVXW   20h             ; Restore W data
                RETI                    ; Return from interrupt

INT1_SUBROUTINE:                        ; INT1 interrupt service routine
                …                       ;
                MOVLW   111111 0 1b
                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	-	-	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	0	0	0	0	0	0

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
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	-	-	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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

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	–	–	–	–	–	TKIE	PWMIE	LVDIE
R/W	–	–	–	–	–	R/W	R/W	R/W
Reset	–	–	–	–	–	0	0	0

0Dh.2 **TKIE:** Touch Key interrupt enable
0: disable
1: enable

0Dh.1 **PWMIE:** PWM0~5 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	–	–	–	–	–	TKIF	PWMIF	LVDIF
R/W	–	–	–	–	–	R/W	R/W	R/W
Reset	–	–	–	–	–	0	0	0

0Eh.2 **TKIF:** Touch key interrupt event pending flag
This bit is set by H/W after touch key end of conversion, write 0 to this bit 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 PA0-PA4, PA7

Each IO has 4-bit pin mode setting. The mode setting can include the following functions: open drain output, cmos output, pull-up resistor, pull-down resistor, pin changed wake-up, PWMO and so on.

These pins can be operated in different modes as below table.

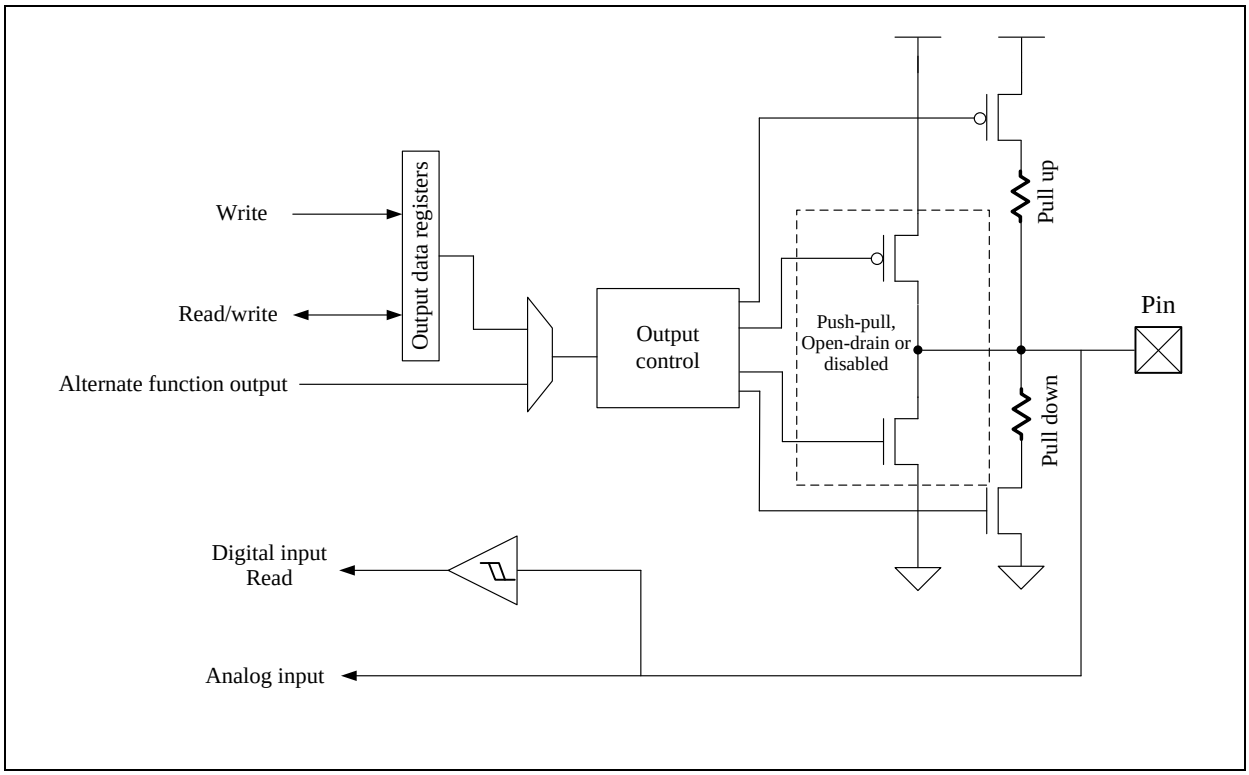
PA7 does not provide functions such as pull-down resistor, 1/2 bias, PWMO and CTKCKO, so please do not select xx11b or x100b for the PIN mode setting of PA7.

PAMOD	PAD	Pin Function	Pin State	Pull-up Resistor	Pull-down Resistor	Digital Input	Pin changed Wakeup
0000b	0	Open Drain	Drive Low	-	-	-	-
	1	Input	Pull-up	Y	-	Y	-
0001b	0	Open Drain	Drive Low	-	-	-	-
	1		Hi-Z	-	-	Y	-
0010b	0	CMOS Output	Drive Low	-	-	-	-
	1		Drive High	-	-	-	-
0011b	X	Tenx Reserved	Hi-Z	-	-	-	-
0100b	0	Open Drain	Drive Low	-	-	-	-
	1	Input	Pull- down	-	Y	Y	-
0101b	0	Open Drain	Drive Low	-	-	-	-
	1		Hi-Z	-	-	Y	-
0110b	0	CMOS Output	Drive Low	-	-	-	-
	1		Drive High	-	-	-	-
0111b	X	Function CMOS output for PWMO	-	-	-	-	-
1000b	0	Open Drain	Drive Low	-	-	-	-
	1	Input	Pull-up	Y	-	Y	Y
1001b	0	Open Drain	Drive Low	-	-	-	-
	1		Hi-Z	-	-	Y	Y
1010b	0	CMOS Output	Drive Low	-	-	-	-
	1		Drive High	-	-	-	-
1011b	X	Function CMOS output for CTKCKO	-	-	-	-	-
1100b	0	Open Drain	Drive Low	-	-	-	-
	1	Input	Pull- down	-	Y	Y	Y
1101b	0	Open Drain	Drive Low	-	-	-	-
	1		Hi-Z	-	-	Y	Y
1110b	0	CMOS Output	Drive Low	-	-	-	-
	1		Drive High	-	-	-	-
1111b	X	Analog output for 1/2 bias (1/2 V_{CC})	-	-	-	-	-

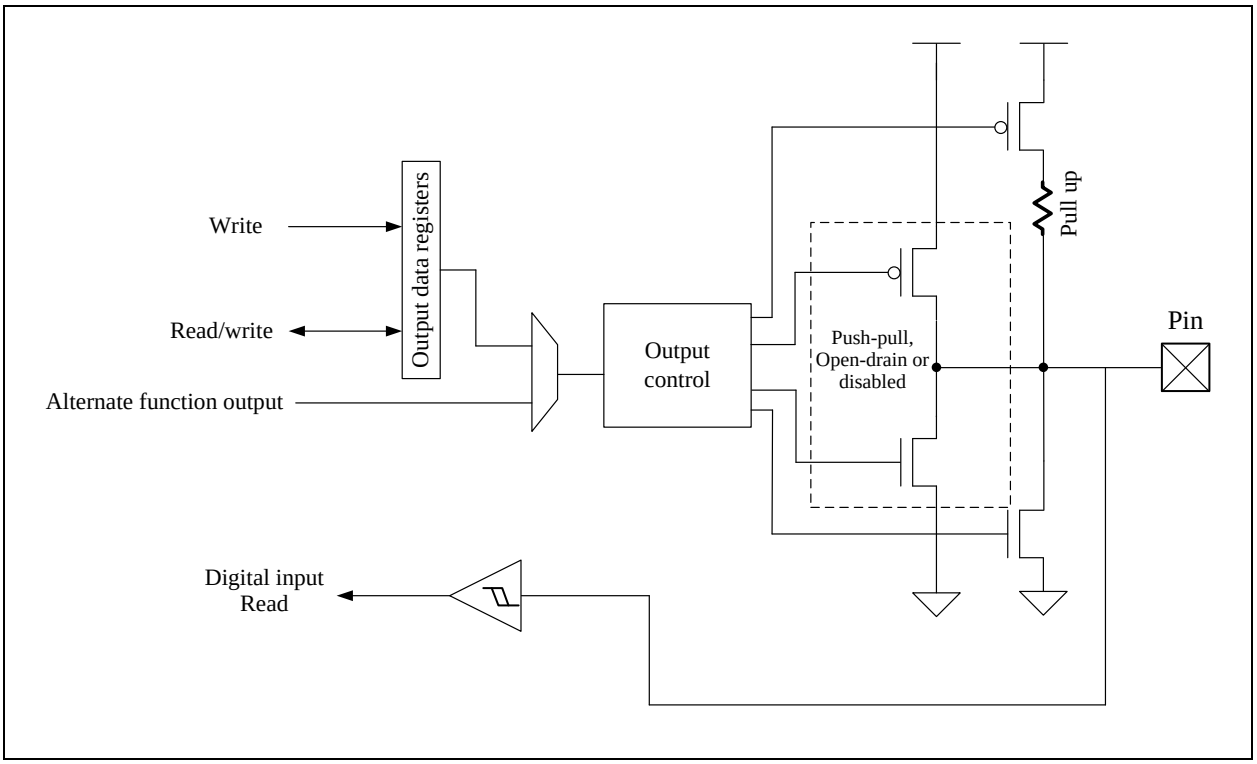
General I/O Pin Function Table

Pin Name	PAMOD					
	0111b (Digital output)	1011b (Digital output)	1111b (Analog output)	xx10b & PADx=1	xx10b & PADx=0	xx10b & PADx=0
PA0	PWM5O	CTKCKO	1/2 bias	TK5 (CTK)	TK5 (STK)	
PA1	PWM1O	CTKCKO	1/2 bias	TK1 (CTK)	TK1 (STK)	
PA2	PWM4O	CTKCKO	1/2 bias	TK4 (CTK)	TK4 (STK)	
PA3	PWM2O		1/2 bias			CLD
PA4	PWM0O	CTKCKO	1/2 bias	TK2 (CTK)	TK2 (STK)	
PA7						

Special function for Pin Table



General Pin Structure (exclude PA7)



PA7 Pin Structure

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, PA4~PA0 Pin Mode Control
87h.3~0 0000: Open drain with pull-up
86h.7~4 0001: Open drain
86h.3~0 0010: Push-pull
85h.7~4 0011: Analog input
85h.3~0 0100: Open drain with pull-down
0101: Open drain
0110: Push-pull
0111: Alternate function output
1000: Open drain with pull-up and pin-changed wakeup
1001: Open drain and pin-changed wakeup
1010: Push-pull
1011: CTKCKO output
1100: Open drain with pull-down and pin-changed wakeup
1101: Open drain and pin-changed wakeup
1110: Push-pull
1111:1/2 bias

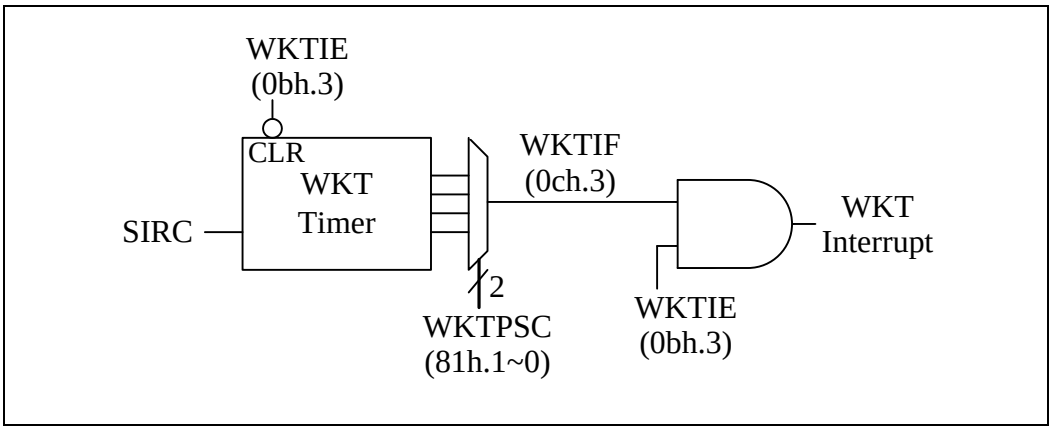
05h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

05h.7~0 **PAD**: PA7, PA4~PA0 data

6 Peripheral Functional Block

6.1 Wake up Timer (WKT)

The WKT is an interval timer, and the interval length can be adjusted by WKTPSC (81h.1~0), WKT will generate WKT Interrupt Flag (WKTIF) when WKT timeout. WKTIE is interrupt enable and also a switch that determines whether to start the WKT function. When WKTIE=0, WKT will be cleared and stopped. Since WKT has a certain power consumption, WKTIE=0 is one of the conditions for entering STOP mode.



WKT Block Diagram

◇ Example: Set WKT period and interrupt function.

```
MOVLW   000001xxb
MOVWX   OPTION          ; Select WKT period=50 ms @4V
MOVLW   1111x111b       ; Clear WKT interrupt flag by using byte operation
MOVWX   INTIF           ; 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	-	-	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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	-	-	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	0	0	0	0	0	0

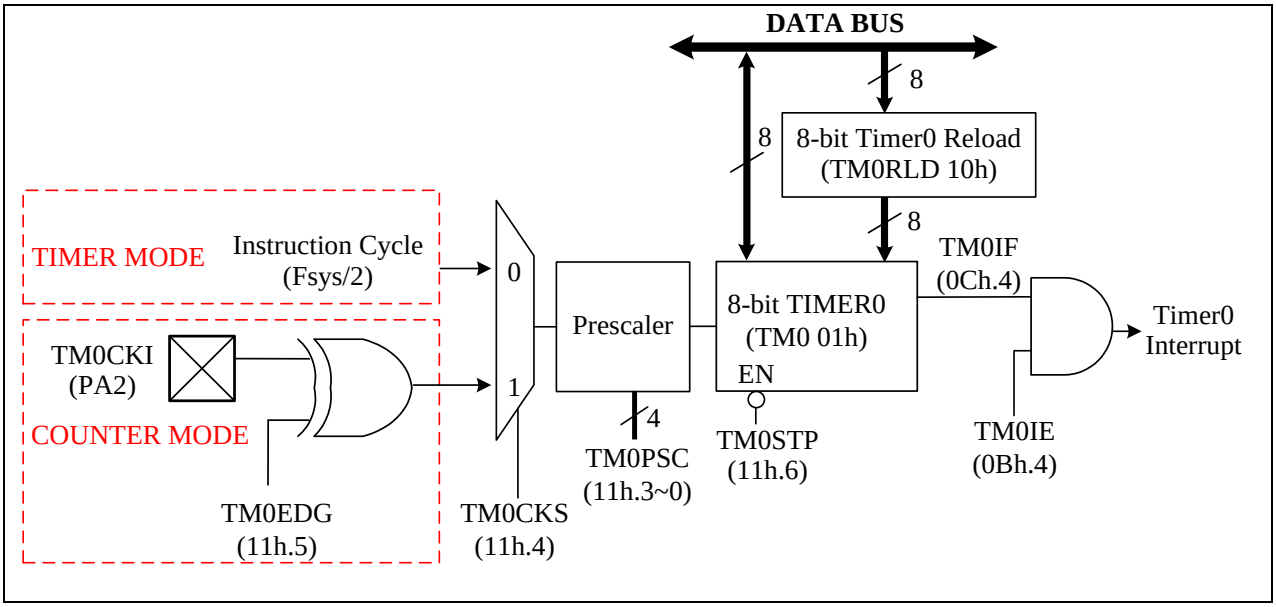
0Bh.3 **WKTIE:** Wakeup Timer interrupt enable
0: disable
1: enable

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

81h.1~0 **WKTPSC:** WKT period
00: 12 ms 01: 25 ms 10: 50 ms 11: 100 ms @ VCC=4V

6.2 Timer0

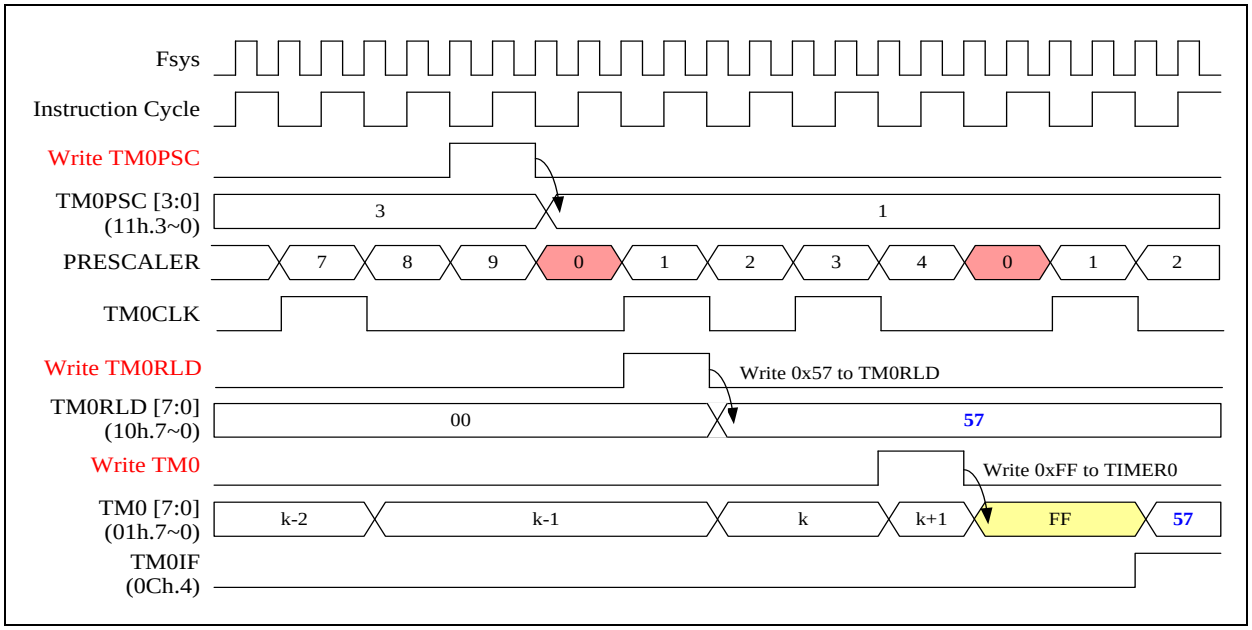
The Timer0 is an 8-bit wide register 01h (TM0). It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be Fsys/2 or TM0CKI (PA2) rising/falling input. The Timer0 increase rate is determined by “Timer0 Pre-Scale” (TM0PSC) register. The Timer0 always generates TM0IF when its count rolls over. It generates Timer0 Interrupt if (TM0IE) is set. Timer0 can be stopped counting if the TM0STP 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:

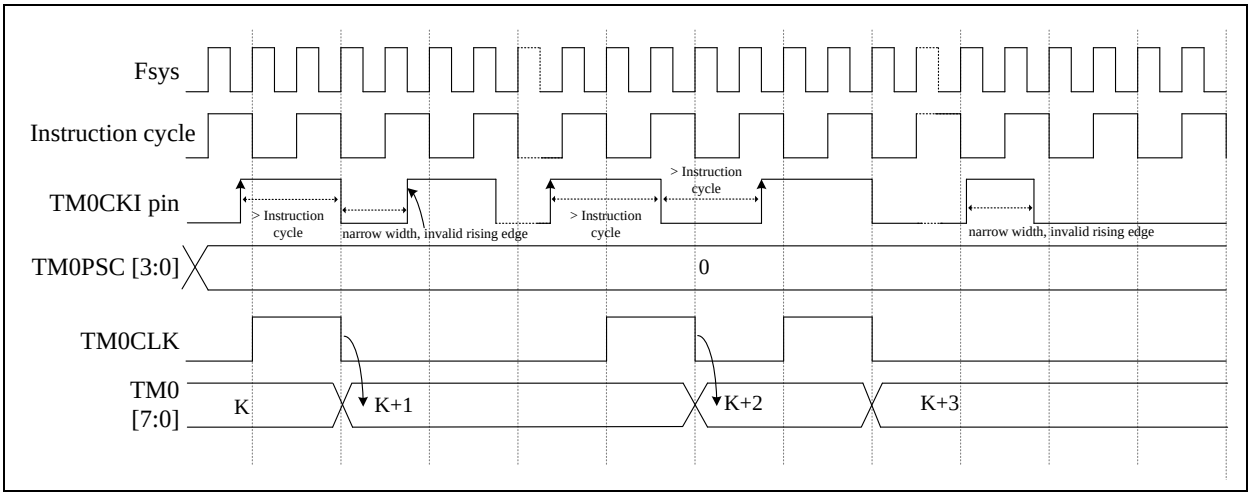
◇Example: Setup Timer0 work in Timer mode, if Fsys = 8 MHz

```
; Setup Timer0 clock source and divider
        MOVLW   00000101b       ; TM0CKS = 0, Timer0 clock source is instruction cycle (Fsys/2)
        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
        CLRX    TM0             ; Clear Timer0 content

; Enable Timer0 and interrupt function
        MOVLW   1110 1111b
        MOVWX   INTIF           ; Clear Timer0 request interrupt flag
        BSX     TM0IE           ; Enable Timer0 interrupt function
        BCX     TM0STP          ; Enable Timer0 counting
```

Timer0 interrupt frequency = Timer0 clock frequency / (256-TM1RLD)
= (Fsys / 2 / TM0PSC) / (256-TM0RLD)
= (128 KHz) / (256-128) = 1 KHz

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

If TM0CKS=1 then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle (Fsys/2) that means the high/low time durations of TM0CKI must be longer than one instruction cycle time (Fsys/2) to guarantee each TM0CKI's change will be detected correctly by the synchronizer.



Timer0 works in Counter mode for TM0CKI (TM0EDG=0) , TM0CKS=1

◇ Example: Setup TM0 work in Counter mode and clock source from TM0CKI pin (PA2)

```
; Setup Timer0 clock source and divider
        MOVLW   00110000B       ; TM0EDG = 1, counting edge is falling edge
        MOVWX   TM0CTL          ; TM0CKS = 1, Timer0 clock is TM0CKI
                                ; TM0PSC = 0000b, divided by 1
; Setup Timer0
        BSX     TM0STP          ; Timer0 stops counting
        CLRX    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	-	-	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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	-	-	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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:** Timer0 counter stop.
0: Release 1: Stop counting

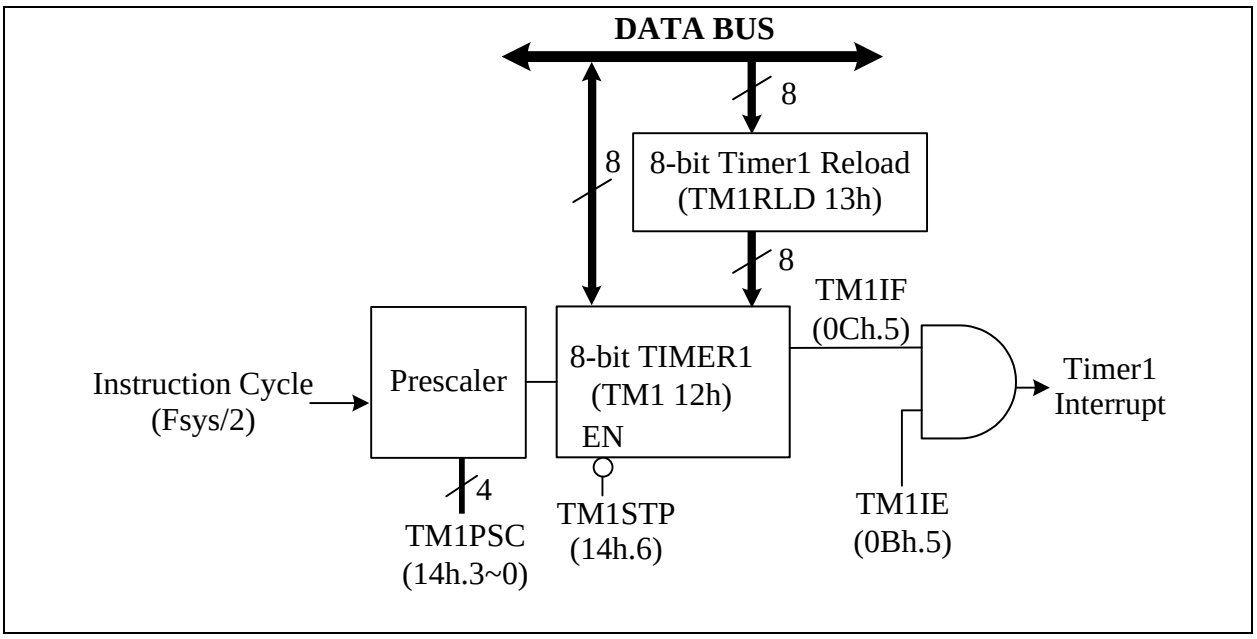
11h.5 **TM0EDG:** TM0CKI (PA2) edge.
0: rising edge 1: falling edge

11h.4 **TM0CKS:** Timer0 prescaler clock source
0:Fsys/2 1: TM0CKI pin (PA2)

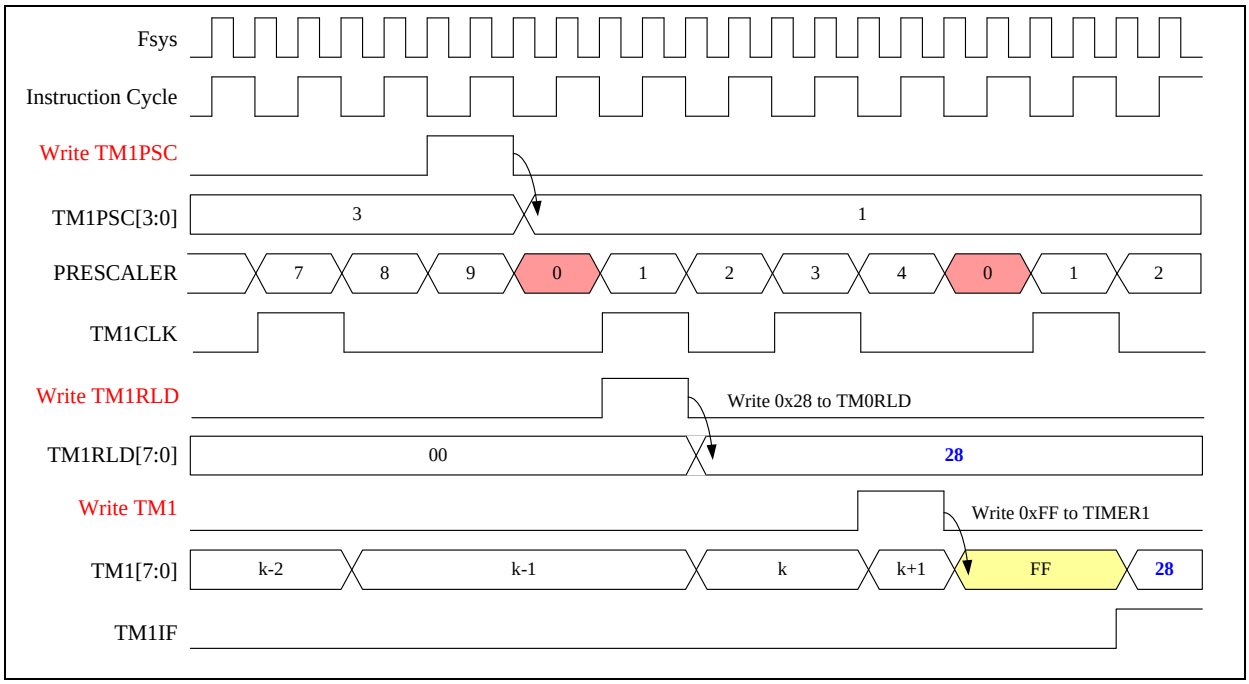
11h.3~0 **TM0PSC:** Timer0 prescaler. Timer0 clock source (Fsys/2 or TM0CKI) divided by
0000: 1 0011: 8 0110: 64
0001: 2 0100: 16 0111: 128
0010: 4 0101: 32 1xxx: 256

6.3 Timer1

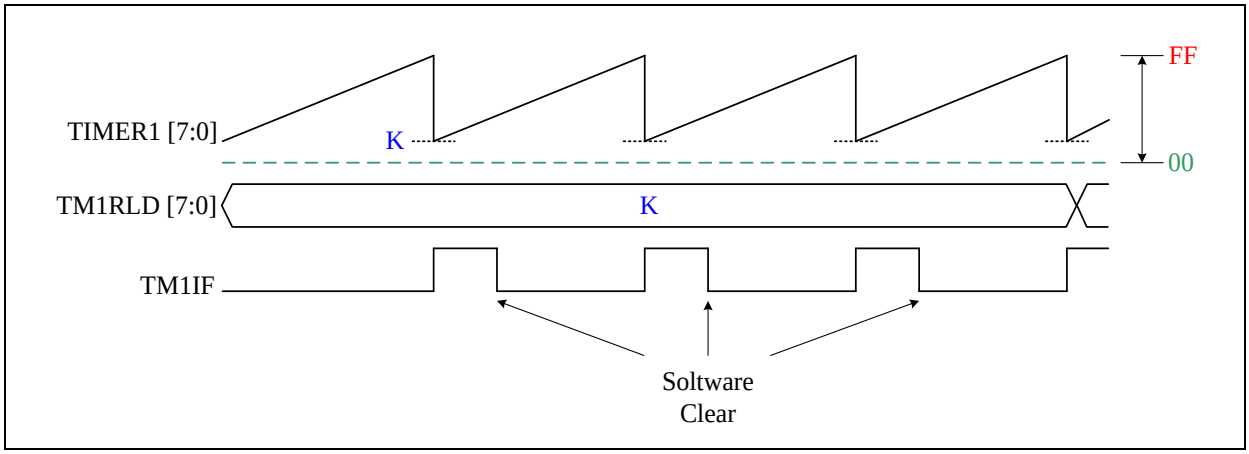
The Timer1 is an 8-bit wide register. It can be read or written as any other register. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock (Fsys/2). The Timer1 increase rate is determined by TM1PSC register. It generates Timer1 interrupt if the TM1IE bit is set. Timer1 can be stopped counting if the TM1STP bit is set.



Timer1 Block Diagram



Timer1 Timing Diagram



Timer1 Reload Diagram

The equation of TM1 interrupt time value is as following:

◇Example: Setup Timer1 work in Timer mode, if Fsys = 8 MHz

```
; Setup Timer1 clock source and divider
        MOVLW   00000101b       ; Timer1 clock source is instruction cycle (Fsys/2)
        MOVWX   TM1CTL          ; TM1PSC = 0101b, divided by 32

; Setup Timer1 reload data
        MOVLW   FFh
        MOVWX   TM0RLD          ; Set Timer1 reload data = 255

; Setup Time1
        BSX     TM1STP          ; Timer1 stops counting
        CLRX    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 = Timer1 clock frequency / (256-TM1RLD)
= (Fsys / 2 / TM0PSC) / (256-TM0RLD)
= (128 KHz) / (256-128) = 1 KHz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	-	-	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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	-	-	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	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 offset value while it rolls over

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:** Timer1 counter stop
0: Release
1: Stop counting

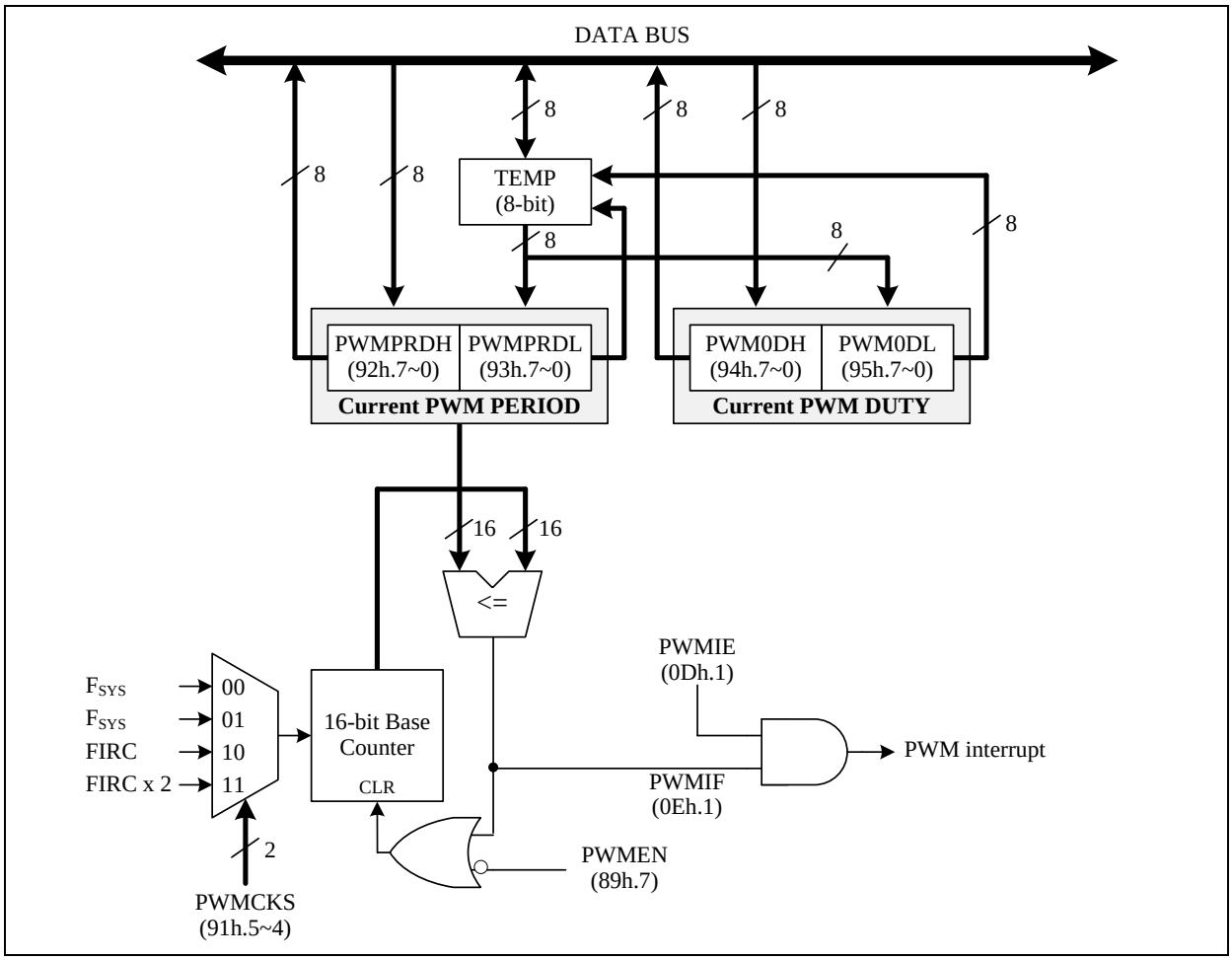
14h.3~0 **TM1PSC:** Timer1 prescaler. Timer1 clock source (Fsys/2) divided by
0000: 1 0011: 8 0110: 64
0001: 2 0100: 16 0111: 128
0010: 4 0101: 32 1xxx: 256

6.4 PWM: 16 bits PWM

There are five PWMs in this chip. All PWM have independent 16bit 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 FIRC*2, FIRC or F_{SYS} as its clock source.

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, all PWM will be cleared and stopped, otherwise all PWM 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. All PWM share an interrupt flag, and an interrupt flag is generated at the end of the period.



PWM0 Block Diagram

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	–	–	–	–	TKIE	PWMIE	LVDIE
R/W	–	–	–	–	–	R/W	R/W	R/W
Reset	–	–	–	–	–	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	–	–	–	–	–	TKIF	PWMIF	LVDIF
R/W	–	–	–	–	–	R/W	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	–	–	–	–	–	–	–
R/W	R/W	–	–	–	–	–	–	–
Reset	0	–	–	–	–	–	–	–

89h.7 **PWMEN:** PWM enable
0: disable
1: enable

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION2	–	–	PWMCKS		–	–	–	–
R/W	–	–	R/W		–	–	–	–
Reset	–	–	0	0	–	–	–	–

91h.5~4 **PWMCKS:** PWM clock source select
00: F_{SYS}
01: F_{SYS}
10: FIRC
11: FIRC x 2 (VCC>2.3V@25 ℃)

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:** PWM 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:** PWM 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

9Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DH	PWM4DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Ch.7~0 **PWM4DH:** PWM4 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

9Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DL	PWM4DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

9Dh.7~0 **PWM4DL:** PWM4 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DH	PWM5DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Eh.7~0 **PWM5DH:** PWM5 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

9Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DL	PWM5DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

9Fh.7~0 **PWM5DL:** PWM5 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

6.5 Touch Key (M1531 Only)

The device support 4 channels touch key detection. The Touch Key offers two easy, simple and reliable methods to implement finger touch detection. One structure is STK, and the other is CTK. In most applications, it doesn't require any external component in STK mode. Even though in CTK mode, it only requires an external capacitor component (CLD).

The pin status during TK scanning is automatically switched by the hardware, while the pin status during TK idle period needs to be set by the user. It is strongly recommended to set the pins as shown in the table below. Please refer to Section 5 for details.

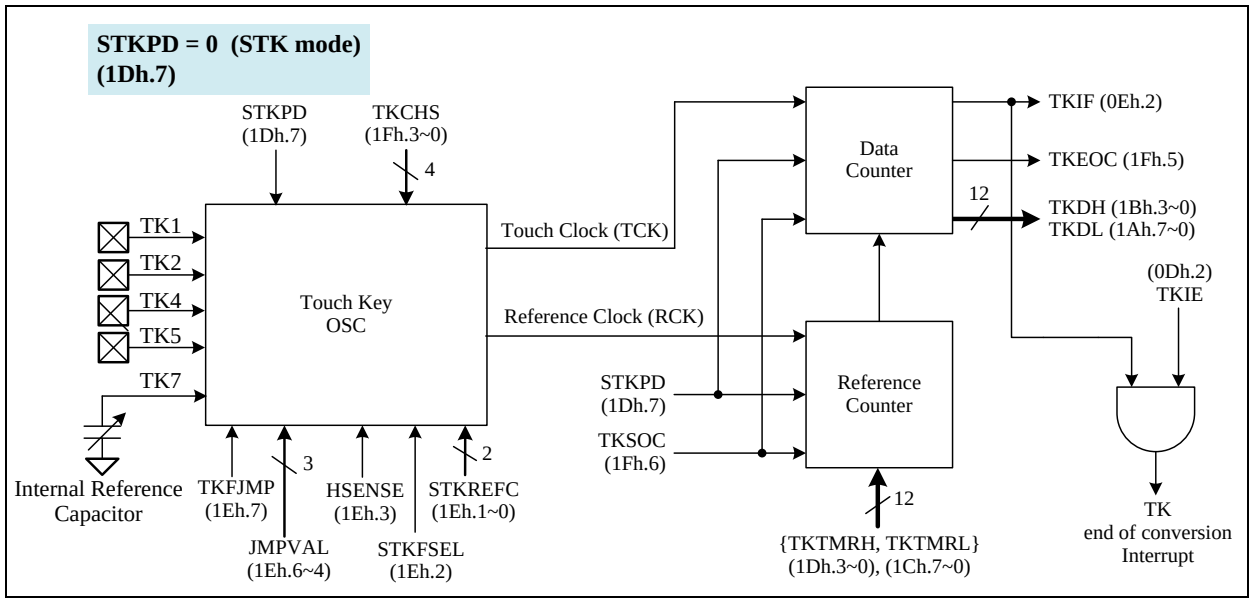
With these pin settings, the mutual influence of the TK key can be reduced, and the discharge speed of the external capacitor CLD can be accelerated.

Pin Name	Pin Mode Setting			
	PAMOD=1011b (Digital output)	PAMOD=xx10b & PAD=1	PAMOD=xx10b & PAD=0	PAMOD=xx10b & PAD=0
PA0	CTKCKO	TK5 (CTK)	TK5 (STK)	
PA1	CTKCKO	TK1 (CTK)	TK1 (STK)	
PA2	CTKCKO	TK4 (CTK)	TK4 (STK)	
PA3				CLD
PA4	CTKCKO	TK2 (CTK)	TK2 (STK)	
PA7				

6.5.1 STK

Set STKPD=0, you can choose to relax the vibration structure and touch the button. In STK mode, there are two oscillators: the reference clock (RCK) and the touch clock (TCK). They are connected to the reference counter and data counter respectively. The frequency of RCK can be adjusted by setting STKREFC, the reference counter is used to control the conversion time. The number of RCK oscillation cycles (0 to 4096) required is determined by setting TKTMR from the start of touch key conversion to the end. After the conversion is over, the user can get TKDATA (TKDH, TKDL) from the data counter. TKDATA is affected by finger touch. When finger touch slows down TCK, the value of TKDATA is less than the value without finger touch. Depending on the TKDATA, the user can check if it was touched. On the other hand, setting STKFSEL can adjust the overall frequency of the TK system (including TCK and RCK). STKREFC only controls the frequency of RCK. Set TKFJMP=1, the system will automatically change the TCK frequency randomly.

There is a built-in reference capacitor inside the touch key unit to simulate the behavior of the keys. Set TKCHS= 7 or 15, the system will switch to the built-in reference capacitor, and start to touch the button to convert to get the TKDATA of this reference capacitor. Because the internal capacitance is never affected by water or cell phones, it is useful for comparing ambient background noise. The touch key clock frequency can be adjusted automatically by the internal hardware by setting TKFJMP=1, which can effectively improve the high interference characteristics of the touch key.



When starting the scan, the user first sets STKPD = 0, and then sets TKSOC to 1 to start the touch key conversion. TKEOC = 0 means the conversion is in progress. TKEOC = 1 indicates that the conversion is completed, and the touch key count result is stored in the 12-bit TK data counters TKDH and TKDL. The TKSOC bit can be automatically cleared after TKEOC=1 at the end of the conversion.

TKIF will be activated when the touch key function is enabled for the first time (STKPD = 0), user should clear TKIF after STKPD = 0.

Example:

```
BCX     STKPD       ; STKPD=0 (select STK)
MOVLW   09H
MOVWX   TKCTL       ; HSENSE=1, STKREFC=1
                    ;
MOVLW   04H
MOVWX   TKMFS       ; TKTMR=400h
MOVLW   00H
MOVWX   TKTMRL      ; TKTMR=400h
MOVLW   A1H
MOVWX   TKCTL2      ; Select TK1
MOVLW   11111011B
MOVWX   INTIF1      ; Clear TKIF
MOVLW   00000100B
IORWX   INTIE1      ; SET TKIE
BSX     TKSOC
```

01Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKDL	TKDL							
R/W	R							
Reset	1	1	1	1	1	1	1	1

1Ah.7 **TKDL:** Touch Key counter data 7~0

01Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKDH	–	–	–	–	TKDH			
R/W	–	–	–	–	R			
Reset	–	–	–	–	1	1	1	1

1Bh.7 **TKDH:** Touch Key counter data 11~8

01Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKTMRL	TKTMRL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

1Ch.7 **TKTMR:** STK Scan length adjustment bit 7~0.
12-bit TKTMR: 000: shortest, FFF: longest

01Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMFS	STKPD	–	–	–	TKTMRH			
R/W	R/W	–	–	–	R/W			
Reset	1	–	–	–	0	0	0	0

1Dh.7 **STKPD:** STK Power Down.
0: STK enable and CTK disable.
1: STK disable.

1Dh.3~0 **TKTMRH:** STK Scan length adjustment bit 11~8.
12-bit TKTMR: 000: shortest, FFF: longest.

01Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL	TKFJMP	JMPVAL			HSENSE	STKFSEL	STKREFC	
R/W	R/W	R/W			R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

1Eh.7 **TKFJMP:** Touch Key clock frequency auto change select
0: disable, determined by SFR JMPVAL
1: enable, Touch Key clock frequency auto-change

1Eh.6~4 **JMPVAL:** Touch Key clock frequency select (only available in TKFJMP=0)

1Eh.3 **HSENSE:** STK channel sensitivity select
0: normal mode
1: high sensitivity

1Eh.2 **STKFSEL:** STK clock (RCK/TCK) frequency selection.
0: STK clock frequency is the slowest.
1: STK clock frequency is the fastest.

1Eh.1~0 **STKREFC:** STK reference clock (RCK) capacitor select.
0: smallest (conversion time shortest)
3: biggest (conversion time longest)

01Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL2	CTKPD	TKSOC	TKEOC	–	TKCHS			
R/W	R/W	R/W	R	–	R/W			
Reset	1	0	1	–	1	1	1	1

1Fh.6 **TKSOC:** Touch Key start conversion, HW clear while end of conversion
0: disable
1: enable, Touch Key start conversion

1Fh.5 **TKEOC:** Touch Key end of conversion

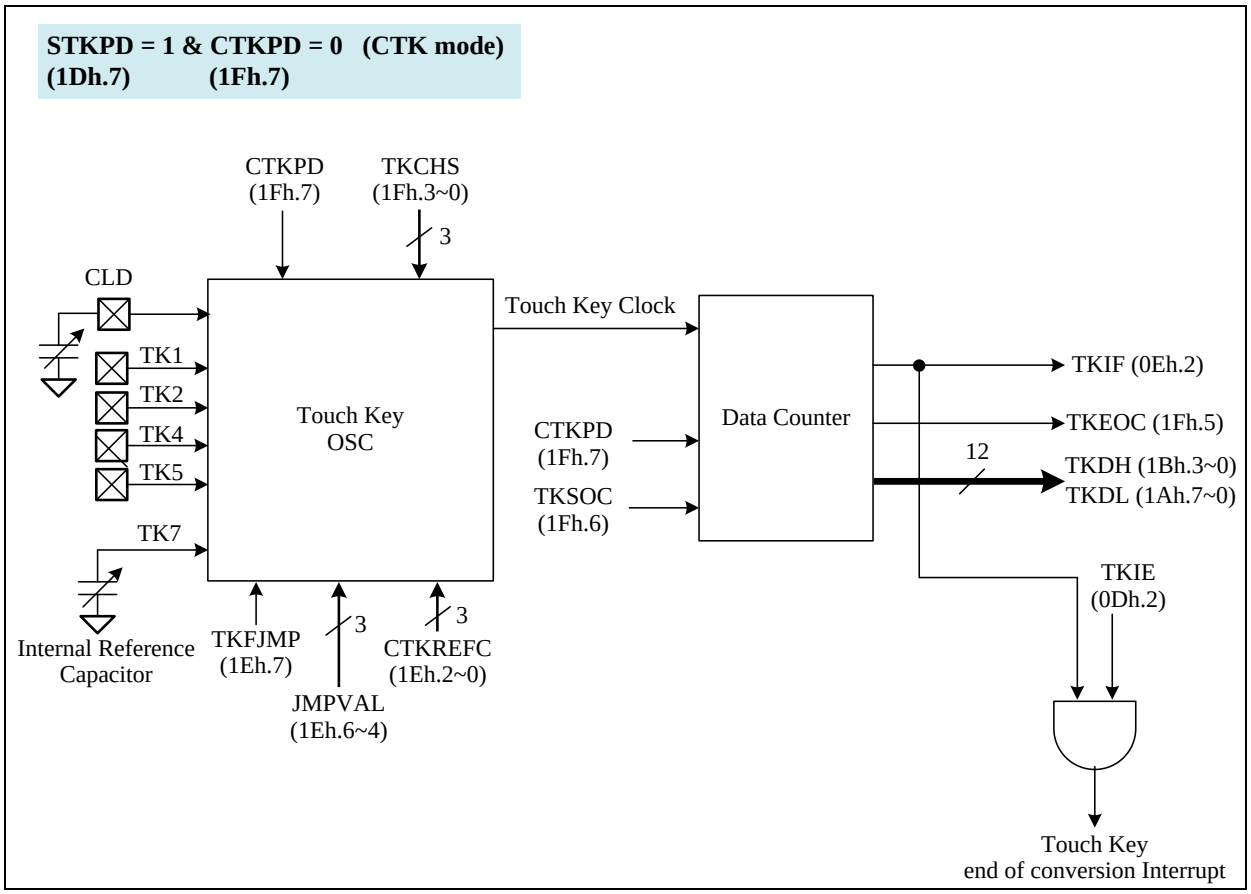
1Fh.3~0 **TKCHS:** Touch Key channel select

x000: reserved	x100: TK4
x001: TK1	x101: TK5
x010: TK2	x110: reserved
x011: reserved	x111: Internal reference capacitor

6.5.2 CTK

Set STKPD=1 and CTKPD=0, user can choose the external capacitive structure touch button. In CTK mode, an external capacitor needs to be connected to the CLD pin. To start the scan, the user needs to assign CTKPD=0, and then set the TKSOC bit to start the touch key conversion. The TKSOC bit can be automatically cleared when the conversion ends. TKEOC=0 means the conversion is in progress. TKEOC=1 means the conversion is over, and the count value of the touch key is stored in the 12-bit TK data count (TKDH and TKDL). After TKEOC=1, the user must wait at least 50μs for the next conversion. If the converted value is outside the period range, decreasing/increasing CTKREFC can decrease/increase TK data counts to suit system board conditions.

There is a built-in reference capacitor inside the touch key unit to simulate the behavior of the keys. Set TKCHS= 7 or 15, the system will switch to the built-in reference capacitor, and start to touch the button to convert to get the TKDATA of this reference capacitor. Because the internal capacitance is never affected by water or cell phones, it is useful for comparing ambient background noise. The touch key clock frequency can be adjusted automatically by the internal hardware by setting TKFJMP=1, which can effectively improve the high interference characteristics of the touch key.



Example:

Touch Key channel select Reference TK, and select speedup CLD discharging mode.

```
        MOVLW   00101011b       ; PA3 (CLD) = output low, PA2 (TK4) = CTKCKO
        MOVWX   PAMOD32         ; PA3 setting for speed up CLD discharging
        BCX     PAD,3           ; set PA3 output low

        MOVLW   10000101b       ; for CTK clock auto change
        MOVWX   TKCTL           ; CTK conversion time select 5

        MOVLW   00001111b       ; CTK Touch Key operating
        MOVWX   TKCTL2          ; Touch Key channel select Reference TK
        BSX     TKSOC           ; Touch Key start conversion
                                ; (need delay 5us)
        LCALL   WAIT_TK

WAIT_TK:
        BTXSS   TKEOC           ; Wait TK conversion finish
        LGOTO   WAIT_TK
        RET
```

01Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKDL	TKDL							
R/W	R							
Reset	1	1	1	1	1	1	1	1

1Ah.7 **TKDL:** Touch Key counter data 7~0

01Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKDH	–	–	–	–	TKDH			
R/W	–	–	–	–	R			
Reset	–	–	–	–	1	1	1	1

1Bh.7 **TKDH:** Touch Key counter data 11~8

01Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL	TKFJMP	JMPVAL			HSENSE	CTKREFC		
R/W	R/W	R/W			R/W	R/W		
Reset	0	0	0	0	0	0	0	0

1Eh.7 **TKFJMP:** Touch Key clock frequency auto change select
0: disable, determined by SFR JMPVAL
1: enable, Touch Key clock frequency auto-change

1Eh.6~4 **JMPVAL:** Touch Key clock frequency select (only available in TKFJMP=0)

1Eh.2~0 **CTKREFC:** CTK conversion time.
0: smallest
7: longest

01Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL2	CTKPD	TKSOC	TKEOC	–	TKCHS			
R/W	R/W	R/W	R	–	R/W			
Reset	1	0	1	–	1	1	1	1

1Fh.7 **CTKPD:** CTK power down.
0: CTK enable if STKPD=1.
1: CTK disable.

1Fh.6 **TKSOC:** Touch Key start conversion, HW clear while end of conversion
0: disable
1: enable, Touch Key start conversion

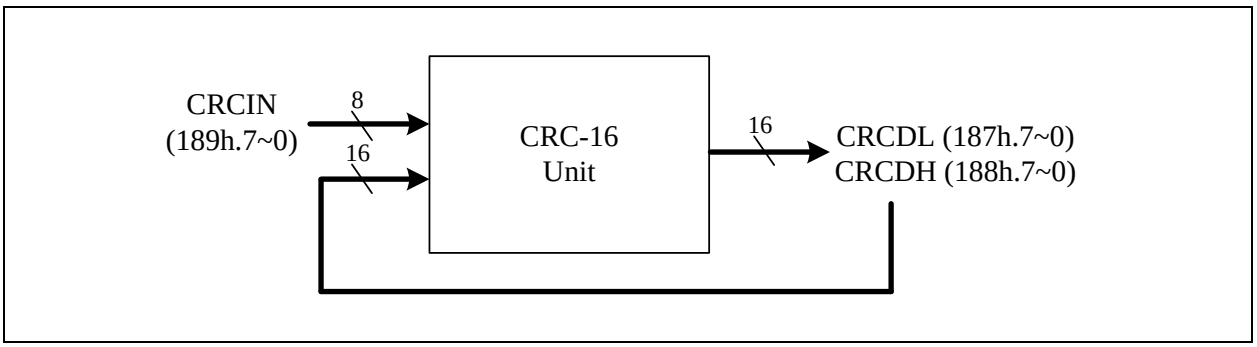
1Fh.5 **TKEOC:** Touch Key end of conversion

1Fh.3~0 **TKCHS:** Touch Key channel select

x000: reserved	x100: TK4
x001: TK1	x101: TK5
x010: TK2	x110: reserved
x011: reserved	x111: Internal reference capacitor

6.6 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 a 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:** write this register to start CRC calculation

MEMORY MAP

Name	Address	R/W	Rst	Description
INDF (00h/80h/100h/180h)				**Function related to: RAM W/R**
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
TM0 (01h/101h)				**Function related to: Timer0**
TM0	01.7~0	R/W	0	Timer0 content
PCL (02h/82h/102h/182h)				**Function related to: PROGRAM COUNT**
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
STATUS (03h/83h/103h/183h)				**Function related to: STATUS**
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 'CLRWDT' instruction
PD	03.3	R	0	Power down flag, set by 'SLEEP', cleared by 'CLRWDT' 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
FSR (04h/84h/104h/184h)				**Function related to: RAM W/R**
FSR	04.7~0	R/W	-	File Select Register, indirect address mode pointer.
PAD (05h)				**Function related to: Port A**
PAD	05.7~0	R	-	Port A pin or "data register" state
		W	FF	Port A output data register
PCLATH (0Ah/8Ah/10Ah/18Ah)				**Function related to: PROGRAM COUNT**
GPR	0A.7~5	R/W	0	General Purpose Register
PCLATH	0A.2~0	R/W	0	Write Buffer for the high byte of the Program Counter
INTIE (0Bh/8Bh/10Bh/18Bh)				**Function related to: Interrupt 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, set 0 to clear & disable WKT timer 0: disable 1: enable
INT2IE	0B.2	R/W	0	INT2 pin (PA7) interrupt enable 0: disable 1: enable
INT1IE	0B.1	R/W	0	INT1 pin (PA1) interrupt enable 0: disable 1: enable
INT0IE	0B.0	R/W	0	INT0 pin (PA3) interrupt enable 0: disable 1: enable
INTIF (0Ch)				**Function related to: Interrupt Flag**
TM1IF	0C.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	write 0: clear this flag; write 1: no action
TM0IF	0C.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	write 0: clear this flag; write 1: no action
WKTIF	0C.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	write 0: clear this flag; write 1: no action

Name	Address	R/W	Rst	Description
INT2IF	0C.2	R	-	INT2 (PA7) interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	write 0: clear this flag; write 1: no action
INT1IF	0C.1	R	-	INT1 (PA1) interrupt event pending flag, set by H/W at INT1 pin's falling/rising edge
		W	0	write 0: clear this flag; write 1: no action
INT0IF	0C.0	R	-	INT0 (PA3) interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge
		W	0	write 0: clear this flag; write 1: no action
INTIE1 (0Dh)				**Function related to: Interrupt Enable**
TKIE	0D.2	R/W	0	Touch Key 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
INTIF1 (0Eh)				**Function related to: Interrupt Flag**
TKIF	0E.2	R	-	Touch Key interrupt flag
		W	0	write 0: clear this flag; write 1: no action
PWMIF	0E.1	R	-	PWM interrupt flag
		W	0	write 0: clear this flag; write 1: no action
LVDIF	0E.0	R	-	LVD interrupt flag
		W	0	write 0: clear this flag; write 1: no action
CLKCTL (0Fh)				**Function related to: System Clock (Fsys)**
SLOWSTP	0F.4	R/W	0	Slow-clock Stop control in STOP mode. 0: Slow-clock run 1: Slow-clock stop
FASTSTP	0F.3	R/W	1	Fast-clock Stop control 0: Fast-clock run 1: Fast-clock stop
CPUCKS	0F.2	R/W	0	Select Fast-clock 0: Fsys=Slow-clock 1: Fsys=Fast-clock
CPUPSC	0F.1~0	R/W	11	Fsys Prescaler, 00: div 8 01: div 4 10: div 2 11: div 1
TM0RLD (10h)				**Function related to: TM0**
TM0RLD	10.7~0	R/W	0	Timer0 reload data.
TM0CTL (11h)				**Function related to: TM0**
TM0STP	11.6	R/W	0	Stop Timer0 0: Timer0 runs 1: Timer0 stops
TM0EDG	11.5	R/W	0	TM0CKI (PA2) edge 0: rising edge 1: falling edge
TM0CKS	11.4	R/W	0	Timer0 prescaler clock source 0: Fsys/2 1: TM0CKI (PA2)
TM0PSC	11.3~0	R/W	0	Timer0 prescaler. Timer0 clock source (Fsys/2 or TMCKI) divided by 0000: 1 0011: 8 0110: 64 0001: 2 0100: 16 0111: 128 0010: 4 0101: 32 1xxx: 256
TM1 (12h)				**Function related to: Timer1**
TM1	12.7~0	R/W	0	Timer1 content
TM1RLD (13h)				**Function related to: Timer1**
TM1RLD	13.7~0	R/W	0	Timer1 reload data

Name	Address	R/W	Rst	Description
TM1CTL (14h)				**Function related to: Timer1**
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
LVCTL (16h)				**Function related to: LVD/LVR**
LVDF	16.7	R	0	Low voltage detection flag, set by H/W while VCC ≦LVD
LVDHYS	16.6	R/W	0	LVD Hysteresis. 0: disable 1: enable
LVRSAV	16.5	R/W	1	POR/LVR auto power off in STOP/IDLE mode
LVDSAV	16.4	R/W	1	LVD auto power off in STOP/IDLE mode
LVDS	16.3~0	R/W	00	LVD select 0000: Disable 0100 : 2.60V 1000: 3.15V 1100: 3.70V 0001: 2.20V 0101: 2.75V 1001: 3.30V 1101: 3.85V 0010: 2.30V 0110: 2.90V 1010: 3.45V 1110: 4.00V 0011: 2.45V 0111: 3.00V 1011: 3.60V 1111: 4.15V
TKDL (1Ah)				**Function related to: STK/CTK**
TKDL	1A.7~0	R	-	Touch Key Counter Data 7~0
TKDH (1Bh)				**Function related to: STK/CTK**
TKDH	1B.3~0	R	-	Touch Key Counter Data 11~8
TKTMRL(1Ch)				**Function related to: STK**
TKTMRL	1C.7~0	R/W	FF	STK Scan length adjustment bit 7~0. 12-bit TKTMR: 000: shortest, FFF: longest
TKMFS (1Dh)				**Function related to: STK**
STKPD	1D.7	R/W	1	STK Power Down. 0: STK enable and CTK disable. 1: STK disable.
TKTMRH	1D.3~0	R/W	0	STK Scan length adjustment bit 11~8. 12-bit TKTMR: 000: shortest, FFF: longest
TKCTL (1Eh)				**Function related to: STK/CTK**
TKFJMP	1E.7	R/W	0	Touch Key clock frequency auto change select 0: disable, clock frequency determine by SFR JMPVAL 1: enable
JMPVAL	1E.6~4	R/W	0	Touch Key clock frequency select. (only available in TKFJMP=0)
HSENSE	1E.3	R/W	0	STK channel sensitivity select 0: normal mode 1: high sensitivity
STKFSEL	1E.2	R/W	0	STK clock (RCK/TCK) frequency selection. 0: STK clock frequency is the slowest 1: STK clock frequency is the fastest
STKREFC	1E,1~0	R/W	0	STK reference clock (RCK) capacitor select. 0:small (conversion time shortest) 3:biggest (conversion time longest)
CTKREFC	1E,2~0	R/W	0	CTK conversion time. 0: smallest 7: longest
TKCTL2 (1Fh)				**Function related to: STK/CTK**
CTKPD	1F.7	R/W	1	CTK Power Down. 0: CTK enable if STKPD=1. 1: CTK disable.

Name	Address	R/W	Rst	Description
TKSOC	1F.6	R/W	0	Touch Key Start of Conversion, HW clear while end of conversion
TKEOC	1F.5	R	1	Touch Key End of Conversion
TKCHS	1F.3~0	R/W	F	Touch Key Channel Select. x000: reserved x100: TK4 x001: TK1 x101: TK5 x010: TK2 x110: reserved x011: reserved x111: Internal reference capacitor
User Data Memory				
RAM	20~6F	R/W	-	RAM Bank0 area (80 Bytes)
RAM	70~7F	R/W	-	RAM common area (16 Bytes)
OPTION (81h/181h)				**Function related to: STATUS/INT0/INT1/WDT/WKT**
HWAUTO	81.7	R/W	0	Enter interrupt vector, HW Save/Restore WREG 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	11	WDT period selections: 00: 100mS 01: 200mS 10: 800mS 11: 1600mS @4V
WKTPSC	81.1~0	R/W	11	WKT period selections: 00: 12mS 01: 25mS 10: 50mS 11: 100mS @4V
PAMOD10 (85h)				**Function related to: Port A**
PA1MOD	85.7~4	R/W	0001	PA1~PA0 I/O mode control
PA0MOD	85.3~0	R/W	0001	
PAMOD32 (86h)				**Function related to: Port A**
PA3MOD	86.7~4	R/W	0001	PA3~PA2 I/O mode control
PA2MOD	86.3~0	R/W	0001	
PAMOD54 (87h)				**Function related to: Port A**
-	87.7~4	R/W	0001	PA4 I/O mode control
PA4MOD	87.3~0	R/W	0001	
PAMOD76 (88h)				**Function related to: Port A**
PA7MOD	88.7~4	R/W	0000	PA7 I/O mode control
-	88.3~0	R/W	0001	
PWMCTL (89h)				**Function related to: PWM0**
PWMEN	89.7	R/W	0	PWM Clock Enable 0: Disable 1: Enable
OPTION2 (91h)				**Function related to: PWM0**
PWMCKS	91.5~4	R/W	00	PWM Clock Source Select 0x: Fsys 10: FIRC (16MHz) 11: FIRC*2 (32MHz) (VCC>2.3V@25 ℃)
PWMPRDH (92h)				**Function related to: PWM**
PWMPRDH	92.7~0	R/W	FF	PWM Period Msb 8bit
PWMPRDL (93h)				**Function related to: PWM**
PWMPRDL	93.7~0	R/W	FF	PWM Period Lsb 8bit
PWM0DH (94h)				**Function related to: PWM0**

Name	Address	R/W	Rst	Description
PWM0DH	94.7~0	R/W	80	PWM0 Duty Msb 8bit
PWM0DL (95h)				**Function related to: PWM0**
PWM0DL	95.7~0	R/W	00	PWM0 Duty Lsb 8bit
PWM1DH (96h)				**Function related to: PWM1**
PWM1DH	96.7~0	R/W	80	PWM1 Duty Msb 8bit
PWM1DL (97h)				**Function related to: PWM1**
PWM1DL	97.7~0	R/W	00	PWM1 Duty Lsb 8bit
PWM2DH (98h)				**Function related to: PWM2**
PWM2DH	98.7~0	R/W	80	PWM2 Duty Msb 8bit
PWM2DL (99h)				**Function related to: PWM2**
PWM2DL	99.7~0	R/W	00	PWM2 Duty Lsb 8bit
PWM4DH (9Ch)				**Function related to: PWM4**
PWM4DH	9C.7~0	R/W	80	PWM4 Duty Msb 8bit
PWM4DL (9Dh)				**Function related to: PWM4**
PWM4DL	9D.7~0	R/W	00	PWM4 Duty Lsb 8bit
PWM5DH (9Eh)				**Function related to: PWM5**
PWM5DH	9E.7~0	R/W	80	PWM5 Duty Msb 8bit
PWM5DL (9Fh)				**Function related to: PWM5**
PWM5DL	9F.7~0	R/W	00	PWM5 Duty Lsb 8bit
User Data Memory				
RAM	A0~EF	R/W	-	RAM Bank1 area (80 Bytes)
LVRPD (109h)				**Function related to: LVR/POR**
LVRPD	109.7~0	W	0	Write 37h to force LVR+POR Disable Write 38h to force LVR Disable, POR still enable Write 39h to force POR Disable, LVR still enable Write others LVR and POR enable
PORPDF	109.1	R	0	POR force power down flag 0: POR enable 1: POR is forced power down
LVRPDF	109.0	R	0	LVR force power down flag 0: LVR enable 1: LVR is forced power down
PCH (10Ch)				**Function related to: PCH**
PCH	10C.7~0	W	0	Programming Counter high byte source selection when instruction with PCL as destination is executed write 0x1C to set PCH_S = 1: PCH keep the original value write others to clear PCH_S = 0: PCH is from PCLATH
PCH	10C.2~0	R	-	Program counter data bit 10~8
BGTRIM (10Eh)				**Function related to: Bandgap**
BGTRIM	10E.4~0	R/W	CFG	VBG trim value
IRCF (10Fh)				**Function related to: Internal RC**
IRCF	10F.6~0	R/W	CFG	FIRC trim value
User Data Memory				
RAM	120~16F	R/W	-	RAM Bank2 area (80 Bytes)

DPL (185h)				**Function related to: Table Read**
DPL	185.7~0	R/W	00	TBL Data Pointer 7~0
DPH (186h)				**Function related to: Table Read**
DPH	186.3~0	R/W	00	TBL Data Pointer 11~8
CRCDL (187h)				**Function related to: CRC16**
CRCDL	187.7~0	R/W	FF	CRC16 Data 7~0
CRCDH (188h)				**Function related to: CRC16**
CRCDH	188.7~0	R/W	FF	CRC16 Data 15~8
CRCIN (189h)				**Function related to: CRC16**
CRCIN	189.7~0	W	0	CRC input
TABR (18Ch)				**Function related to: Table Read**
TABR	18C.7~0	R/W	0	1. TABR write 01h = opcode TABRL 2. TABR write 02h = opcode TABRH 3. After step.1 or step.2, read TABR to get main ROM table read value *Table Read for ASM: instruction TABRL / TABRH or register TABR* *Table Read for C: using register TABR*

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
Byte-Oriented File Register Instruction					
ADDWX	f, d	ffOO 0111 dfff ffff	1	C, DC, Z	Add W and "f"
ANDWX	f, d	ffOO 0101 dfff ffff	1	Z	AND W with "f"
CLRX	f	ffOO 0001 1fff ffff	1	Z	Clear "f"
CLRW		0000 0001 0100 0000	1	Z	Clear W
COMX	f, d	ffOO 1001 dfff ffff	1	Z	Complement "f"
DECX	f, d	ffOO 0011 dfff ffff	1	Z	Decrement "f"
DECXSZ	f, d	ffOO 1011 dfff ffff	1 or 2	-	Decrement "f", skip if zero
INCX	f, d	ffOO 1010 dfff ffff	1	Z	Increment "f"
INCXSZ	f, d	ffOO 1111 dfff ffff	1 or 2	-	Increment "f", skip if zero
IORWX	f, d	ffOO 0100 dfff ffff	1	Z	OR W with "f"
MOVX	f,d	ffOO 1000 dfff ffff	1	Z	Move "f"
MOVXW	f	ffOO 1000 Offf ffff	1	Z	Move "f" to W
MOVWX	f	ffOO 0000 1fff ffff	1	-	Move W to "f"
RLX	f, d	ffOO 1101 dfff ffff	1	C	Rotate left "f" through carry
RRX	f, d	ffOO 1100 dfff ffff	1	C	Rotate right "f" through carry
SUBWX	f, d	ffOO 0010 dfff ffff	1	C, DC, Z	Subtract W from "f"
SWAPX	f, d	ffOO 1110 dfff ffff	1	-	Swap nibbles in "f"
TSTX	f	ffOO 1000 1fff ffff	1	Z	Test if "f" is zero
XORWX	f, d	ffOO 0110 dfff ffff	1	Z	XOR W with "f"
Bit-Oriented File Register Instruction					
BCX	f, b	ff11 00bb bfff ffff	1	-	Clear "b" bit of "f"
BSX	f, b	ff11 01bb bfff ffff	1	-	Set "b" bit of "f"
BTXSC	f, b	ff11 10bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if clear
BTXSS	f, b	ff11 11bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if set
Literal and Control Instruction					
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
LCALL	k	kk10 0kkk kkkk kkkk	2	-	Call subroutine "k"
CLRWDT		0001 1110 0000 0100	1	TO, PD	Clear Watch Dog Timer
LGOTO	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

ADDLW	**Add Literal "k" and W**	
Syntax	ADDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (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

ADDWX	**Add W and "f"**	
Syntax	ADDWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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

ANDLW	**Logical AND Literal "k" with W**	
Syntax	ANDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) 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

ANDWX	**AND W with "f"**	
Syntax	ANDWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) 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

CLRX	Clear "f"	
Syntax	CLRX 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	CLRX FLAG_REG	B : FLAG_REG =0x5A A : FLAG_REG =0x00, Z =1

CLRW	Clear W	
Syntax	CLRW	
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	CLRW	B : W =0x5A A : W =0x00, Z =1

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

COMX	Complement "f"	
Syntax	COMX 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	COMX REG1, 0	B : REG1 =0x13 A : REG1 =0x13, W =0xEC

DECX	**Decrement "f"**	
Syntax	DECX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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

DECXSZ	**Decrement "f", Skip if 0**	
Syntax	DECXSZ f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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 ≠0, PC =LABEL1 + 1

INCX	**Increment "f"**	
Syntax	INCX f [,d]	
Operands	f : 000h ~ 1FFh	
Operation	(destination) ← (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

INCXSZ	Increment "f", Skip if 0	
Syntax	INCXSZ f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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	LABEL1 INCXSZ CNT, 1 LGOTO LOOP CONTINUE	B : PC =LABEL1 A : CNT =CNT + 1 if CNT =0, PC =CONTINUE if CNT ≠0, PC =LABEL1 + 1

IORLW	Inclusive OR Literal with W	
Syntax	IORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (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	IORLW 0x35	B : W =0x9A A : W =0xBF, Z =0

IORWX	Inclusive OR W with "f"	
Syntax	IORWF f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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	IORWX RESULT, 0	B : RESULT =0x13, W =0x91 A : RESULT =0x13, W =0x93, Z =0

LCALL	Call subroutine "k"	
Syntax	LCALL k	
Operands	k : 0000h ~ 1FFFh	
Operation	Operation: TOS ← (PC) + 1, PC.12~0 ← 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

LGOTO	Unconditional Branch	
Syntax	LGOTO k	
Operands	k : 000h ~ 1FFFh	
Operation	PC.12~0 ← 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

MOVX	Move f	
Syntax	MOVX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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

MOVXW	Move "f" to W	
Syntax	MOVXW f	
Operands	f : 000h ~ 1FFh	
Operation	(W) ← (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

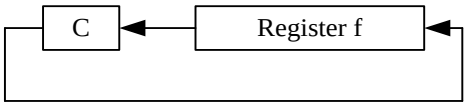
MOVWX	**Move W to "f"**	
Syntax	MOVWX f	
Operands	f : 000h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

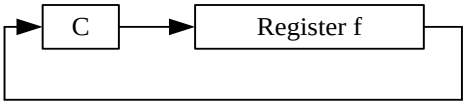
NOP	**No Operation**	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	0000 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

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

RETI	**Return from Interrupt**
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 ← TOS, (W) ← 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	LCALL TABLE B : W =0x07 : A : W =value of k8 TABLE ADDWX PCL, 1 RETLW k1 RETLW k2 : RETLW kn

RLX	**Rotate Left "f" through Carry**
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

RRX	Rotate Right "f" through Carry
Syntax	RRX f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	
Status Affected	C
OP-Code	ff00 1100 dfff ffff
Description	The contents of register 'f' are rotated one bit to the right through the Carry Flag. 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	RRX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =0111 0011, C =0

SLEEP	Go into Power-down mode, Clock oscillation stops
Syntax	SLEEP
Operands	-
Operation	-
Status Affected	TO, PD
OP-Code	001 1110 0000 0011
Description	Go into Power-down mode with the oscillator stops.
Cycle	1
Example	SLEEP -

SUBLW	Subtract W from Literal
Syntax	SUBLW k
Operands	k : 00h ~ FFh
Operation	(W) ← k - (W)
Status Affected	C, DC, Z
OP-Code	0001 1111 kkkk kkkk
Description	The W register is subtracted (2’s complement method) from the eight-bit literal “k”. The result is placed in the W register.
Cycle	1
Example	SUBLW 0x15 B : W =0x25 A : W =0xF0

SUBWX	**Subtract W from "f"**	
Syntax	SUBWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (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) ← (f.3~0), (destination.3~0) ← (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 REG, 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	MOVLW (TAB1&0xFF) MOVWX DPL ;Where DPL register MOVLW (TAB1>>8)&0xFF MOVWX DPH ;Where DPH register TABRL ;W =0x89 TABRH ;W =0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;ROM data 16 bits

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	TSTX REG1 B : REG1 =0, Z =? A : REG1 =0, Z =1

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	XORLW 0xAF B : W =0xB5 A : W =0x1A

XORWX	Exclusive OR W with "f"	
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	XORWX REG, 1	B : REG =0xAF, W =0xB5 A : REG =0x1A, W =0xB5

ELECTRICAL CHARACTERISTICS

1. Absolute Maximum Ratings (T_A =25°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	°C
Storage temperature	-65 to +150	

2. DC Characteristics (TA =25°C, VCC =5.0V, unless otherwise specified)

Parameter	Sym	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{cc}	Fsys = 16Mhz		1.9	–	5.5	V
		Fsys = 8Mhz		1.4	–	5.5	V
Input High Voltage	V_{IH}	All Input	V_{CC} = 3~5V	0.6Vcc	–	Vcc	V
Input Low Voltage	V_{IL}	All Input	V_{CC} = 3~5V	Vss	–	0.2Vcc	V
I/O port Source Current	I_{OH}	All I/O pin (except PA7)	V_{CC} = 5V,V_{OH} = 4.5V	–	TBD	–	mA
			V_{CC} = 3V,V_{OH} = 2.7V	–	TBD	–	
		PA7	V_{CC} = 5V,V_{OH} = 4.5V	–	TBD	–	
			V_{CC} = 3V,V_{OH} = 2.7V	–	TBD	–	
I/O port Sink Current	I_{OL}	All I/O pin (except PA7)	V_{CC} = 5V,V_{OL} = 0.5V	–	TBD	–	mA
			V_{CC} = 3V,V_{OL} = 0.3V	–	TBD	–	
		PA7	V_{CC} = 5V,V_{OL} = 0.5V	–	TBD	–	
			V_{CC} = 3V,V_{OL} = 0.3V	–	TBD	–	
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}	–	–	1	uA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0V	–	–	–1	uA

Parameter	Sym	Conditions		Min	Typ	Max	Unit
Power Supply Current (No Load)	I_{CC}	FAST mode FIRC 16 MHz	V_{CC} = 5V	–	6.4	–	mA
			V_{CC} = 3V		3.7		
		FAST mode FIRC 8 MHz	V_{CC} = 5V	–	3.8	–	
			V_{CC} = 3V		2.3		
		FAST mode FIRC 4 MHz	V_{CC} = 5V	–	3.2	–	
			V_{CC} = 3V	–	1.9	–	
		SLOW mode SIRC/1 FIRC STOP	V_{CC} = 5V	–	1.7	–	
			V_{CC} = 3V	–	1.3	–	
		IDLE mode LVRSAV = 0 LVDSAV=0	V_{CC} = 5V	–	96	–	uA
			V_{CC} = 3V	–	60	–	
		IDLE mode LVRSAV = 1 LVDSAV=1	V_{CC} = 5V	–	20		uA
			V_{CC} = 3V	–	6		
		STOP mode LVRSAV = 1 LVDSAV=1	V_{CC} = 5V	–	0.1		uA
			V_{CC} = 3V	–	0.1		
Pull-up Resistor	R_{UP}	V_{IN} = 0 V Ports A	V_{CC} = 5V	–	TBD	–	KΩ
			V_{CC} = 3V	–	TBD	–	
		V_{IN} = 0 V PA7	V_{CC} = 5V	–	TBD	–	
			V_{CC} = 3V	–	TBD	–	

3. Clock Timing

Parameter	Condition	Min	Typ	Max	Unit
FIRC Frequency (*)	-20°C ~ 85°C, V_{CC} = 3.0 ~ 5.0V	-5%	16	+1.5%	MHz
	-20°C ~ 85°C, V_{CC} = 4.0 V	-3%	16	+1.5%	
	0°C ~ 70°C, V_{CC} = 4.0 V	-2%	16	+1.5%	
	25°C, V_{CC} = 3.0 ~ 5.0 V	-1.2%	16	+1.2%	

(*) FIRC frequency can be divided by 1/2/4/8.

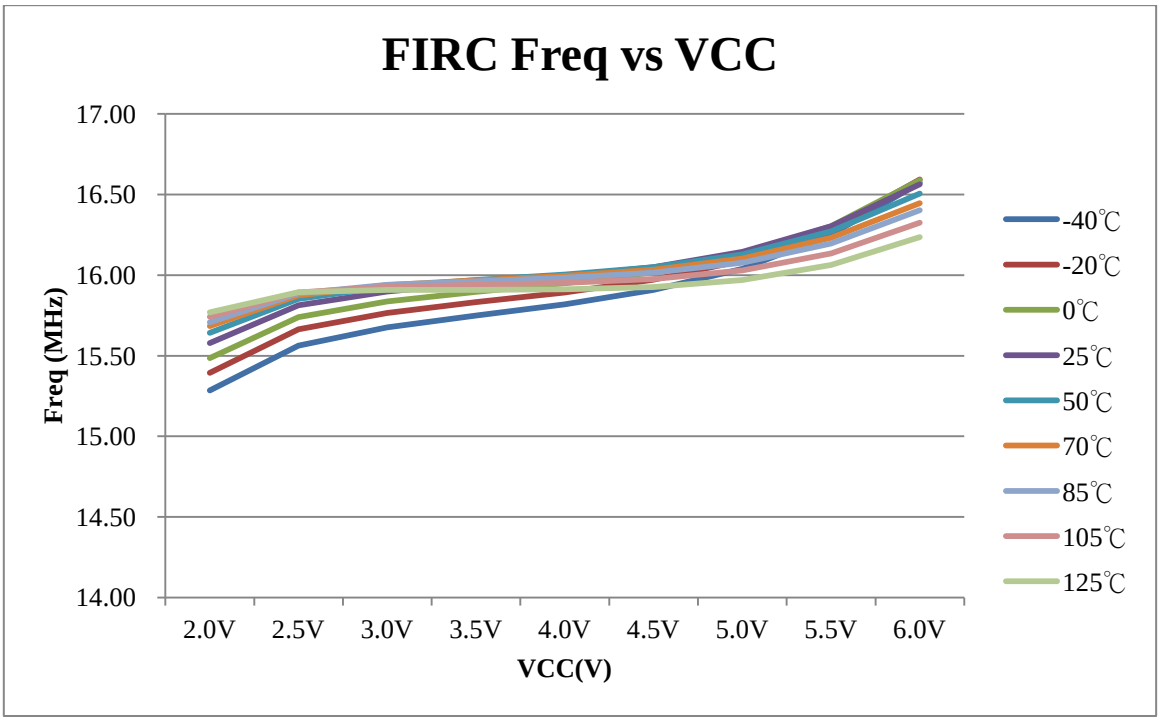
4. Reset Timing Characteristics (T_A = 25°C)

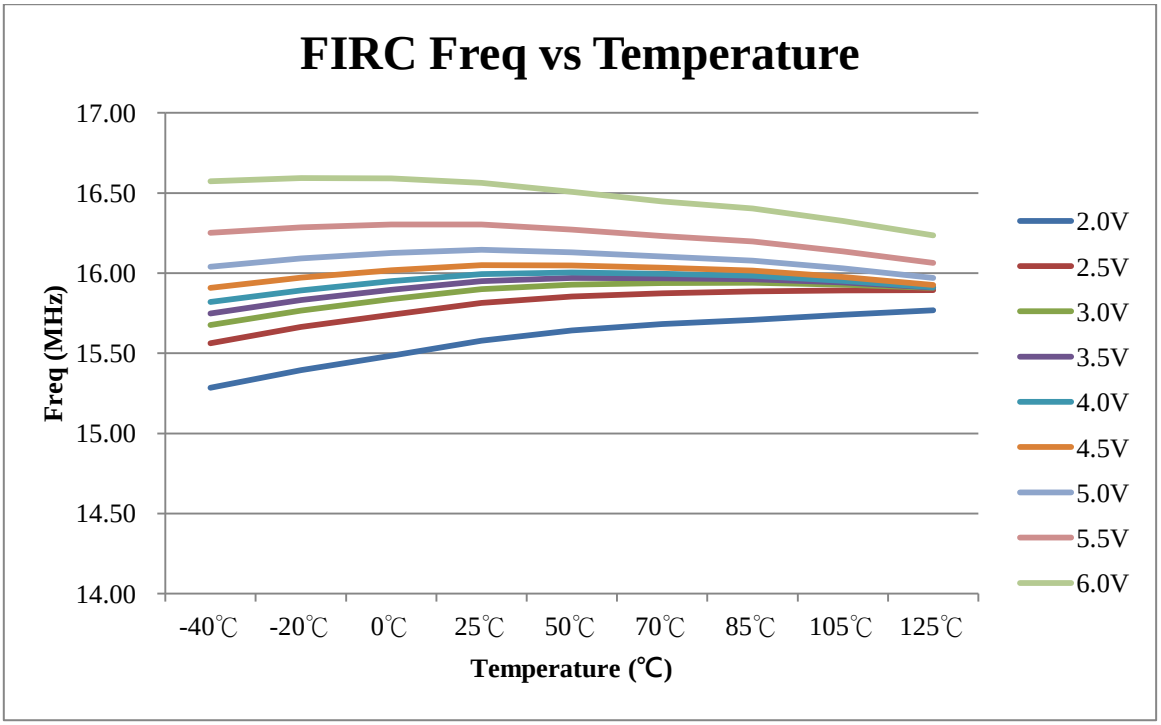
Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input V_{CC} = 5 V ±10 %	–	30	–	μs
WDT time	V_{CC} = 4 V, WDTPSC = 11	–	1600	–	ms
WKT time	V_{CC} = 4 V, WKTPSC = 11	–	100	–	ms
CPU start up time	V_{CC} = 4 V	–	25	–	ms

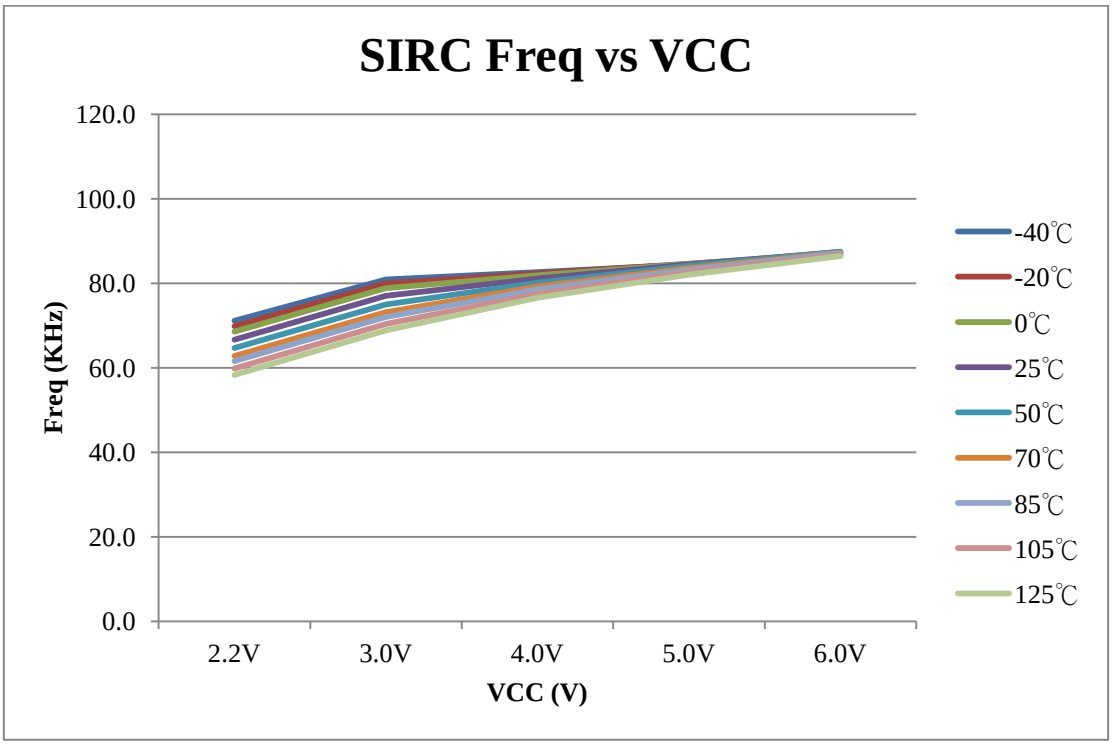
5. LVR and LVD Circuit Characteristics (TA = 25°C)

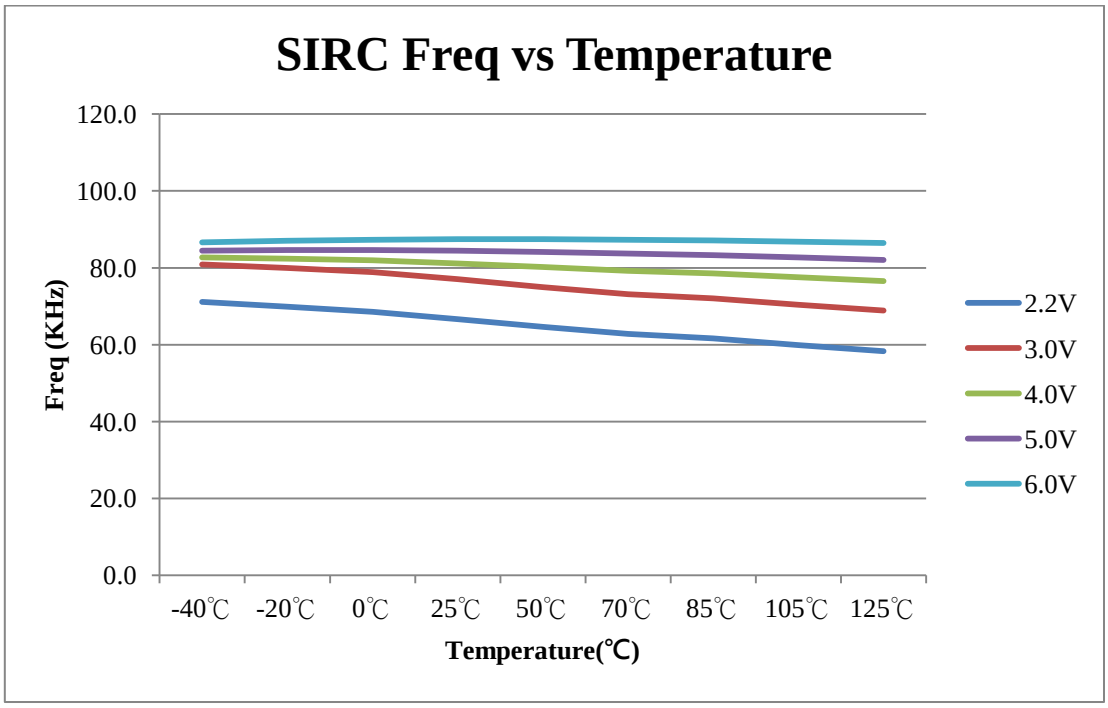
Parameter	Symbol	Conditions	Min	Typ	Max	Unit
LVR Reference Voltage	V_{LVR}	T_A=25°C	–	2.05	–	V
			–	2.20	–	
			–	2.30	–	
			–	2.45	–	
			–	2.60	–	
			–	2.75	–	
			–	2.90	–	
			–	3.00	–	
			–	3.15	–	
			–	3.30	–	
			–	3.45	–	
			–	3.60	–	
			–	3.70	–	
			–	3.85	–	
			–	4.00	–	
			–	4.15	–	
LVR Hysteresis Window	V_{HYS_LVR}	T_A=25°C	–	0	–	mV
Low Voltage Detection time	T_{LVR}	T_A=25°C	100	–	–	µs
LVD Reference Voltage	V_{LVD}	T_A=25°C	–	2.20	–	V
			–	2.30	–	
			–	2.45	–	
			–	2.60	–	
			–	2.75	–	
			–	2.90	–	
			–	3.00	–	
			–	3.15	–	
			–	3.30	–	
			–	3.45	–	
			–	3.60	–	
			–	3.70	–	
			–	3.85	–	
			–	4.00	–	
			–	4.15	–	
LVD Hysteresis Window	V_{HYS_LVD}	LVDHYS=0	–	0	–	mV
		LVDHYS=1, LVD_{th}=2.20V		30		
		LVDHYS=1, LVD_{th}=3.15V		50		
		LVDHYS=1, LVD_{th}=4.15V	–	70	–	
Low Voltage Detection time	T_{LVD}	T_A=25°C	100	–	–	µs

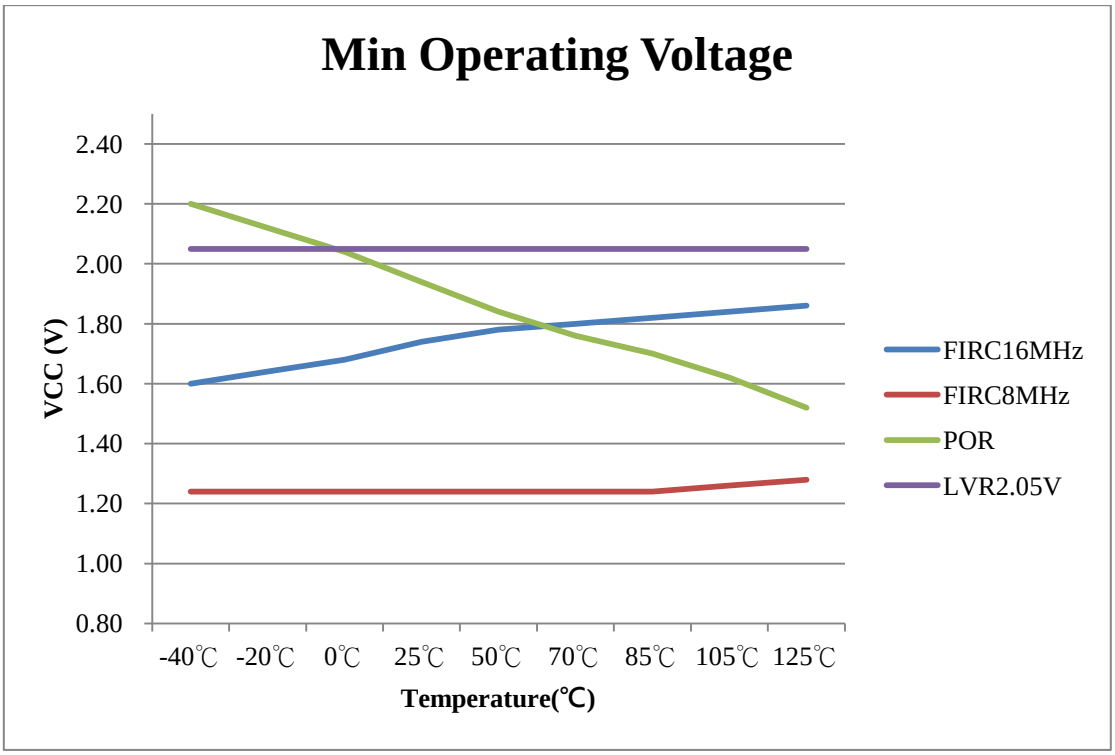
6. Electrical Characteristics Graphs






SIRC Freq vs VCC
120.0
100.0
80.0
60.0
40.0
20.0
0.0
Freq (KHz)
2.2V
3.0V
4.0V
5.0V
6.0V
VCC (V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃


SIRC Freq vs Temperature
120.0
100.0
80.0
60.0
40.0
20.0
0.0
Freq (KHz)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃
Temperature(℃)
2.2V
3.0V
4.0V
5.0V
6.0V


Min Operating Voltage
2.40
2.20
2.00
1.80
1.60
1.40
1.20
1.00
0.80
VCC (V)
-40℃ -20℃ 0℃ 25℃ 50℃ 70℃ 85℃ 105℃ 125℃
Temperature(℃)
FIRC16MHz
FIRC8MHz
POR
LVR2.05V

PACKAGING INFORMATION

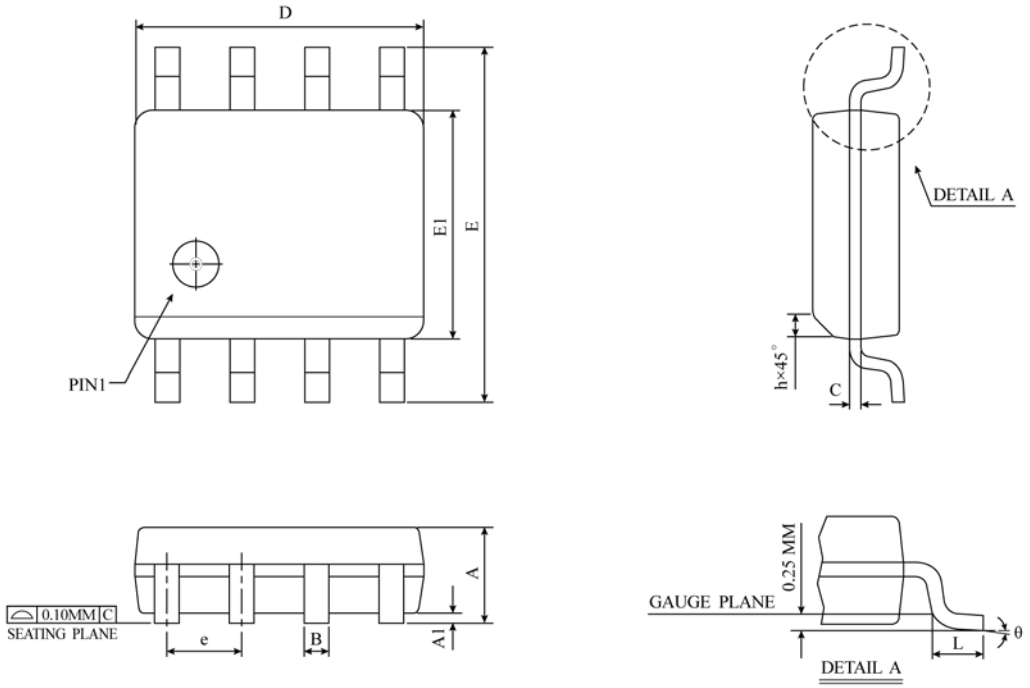
Please note that the package information provided is for reference only. Since this information is frequently updated, users can contact Sales to consult the latest package information and stocks.

The ordering information:

Ordering number	Package
TM56M1511-MTP	Wafer / Dice blank chip
TM56M1511-COD	Wafer / Dice with code
TM56M1511-MTP-14	SOP 8 pin (150mil)
TM56M1511-MTP-A8	SOT23-6

Ordering number	Package
TM56M1531-MTP	Wafer / Dice blank chip
TM56M1531-COD	Wafer / Dice with code
TM56M1531-MTP-14	SOP 8 pin (150mil)
TM56M1531-MTP-A8	SOT23-6

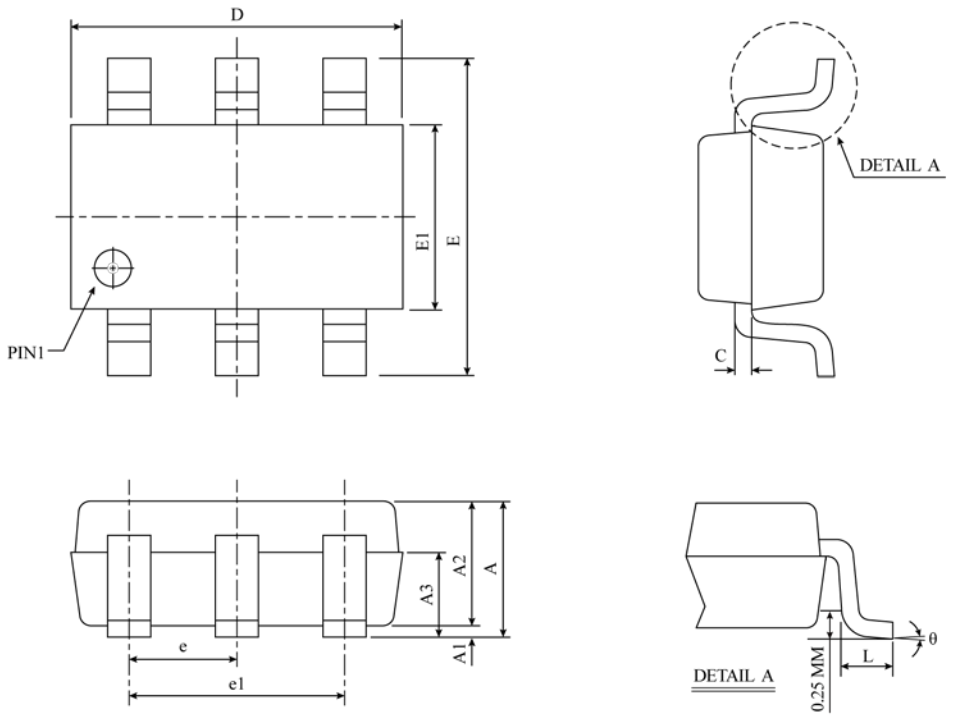
SOP-8 (150mil) Package Dimension



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	4.80	4.90	5.00	0.1890	0.1939	0.1988
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 (AA)					

⚠ * 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.

SOT23-6 Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	1.45	-	-	0.057
A1	0	0.08	0.15	0	0.003	0.006
A2	0.90	1.10	1.30	0.035	0.043	0.051
A3	0.60	0.65	0.70	0.024	0.026	0.028
c	0.12	0.16	0.19	0.005	0.006	0.007
D	2.82	2.92	3.02	0.111	0.115	0.119
E	2.70	2.90	3.10	0.106	0.114	0.122
E1	1.52	1.62	1.72	0.060	0.064	0.068
e	0.85	0.95	1.05	0.033	0.037	0.041
e1	1.80	1.90	2.00	0.071	0.075	0.079
L	0.35	0.48	0.60	0.014	0.019	0.024
θ	0°	4°	8°	0°	4°	8°
JEDEC	M0-178 (AB)					

* NOTES : ALL DIMENSIONS REFER TO JEDEC STANDARD MO-178 AB
DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.