十速

TM57MT28

DATA SHEET

Rev 0.90

AMENDMENT HISTORY

Version	Date	Description
V0.90	Nov, 2018	New release.

CONTENTS

FAMILY OVERVIEW

None

FEATURES

1. **Operating Voltage**
   - V_{BAT}=1.8V~3.6V
2. **Program ROM: 2K x 14 bits MTP (Multi Time Programmable ROM)**
3. **RAM: 368 x 8 bits**
4. **STACK: 6 Levels**
5. **I/O ports: Maximum 18 programmable I/O pins**
   - Open-Drain Output
   - CMOS Push-Pull Output
   - Schmitt Trigger Input with pull-up resistor option
6. **System Oscillation Sources (Fsys)**
   - Fast-clock
     - FIRC (Fast Internal RC): 4MHz (can be trimmed)
   - Slow-clock
     - SIRC (Slow Internal RC): 27 KHz @VCC=3V (SIRC always enable)
   - System Oscillation Sources can be divided by 1/2/4/16 as System Clock (Fsys)
   - Dual System Clock Switching between Fast-clock and Slow-clock
     - FIRC + SIRC
7. **Power Saving Operation Mode**
   - FAST Mode: Fast-clock keeps CPU running.
   - SLOW Mode: Slow-clock keeps CPU running. Fast-clock can be disabled or enabled.
   - IDLE Mode: Fast-clock and CPU stop. Slow-clock keeps running for Touch Key.
8. **Two Independent Timers**
   - Timer0
     - 8-bit timer with divided by 1~256 pre-scale option / counter / auto-reload / interrupt / stop function
   - Timer1
     - 8-bit timer with divided by 1~256 pre-scale option / auto-reload / interrupt / stop function
9. **Interrupt**
   - One External Interrupt pins (PA7)
     - 1 pin is falling edge wake-up triggered & interrupt
   - I2C / TK / Timer1 / Timer0 / WKT (wake-up) interrupt
10. **Wake-up Timer (WKT)**
    - Clocked by built-in RC oscillator with 4 adjustable interrupt time

19.2 ms / 38.4 ms / 76.8 ms / 153.6 ms @VCC=3V

11. Watchdog Timer (WDT)

- Clocked by built-in RC oscillator with 4 adjustable reset time
  38.4ms / 76.8ms / 307.2ms / 614.4ms @VCC=3V

12. PWM*4

- PWM0:
  - 8+2 bits PWM0 with duty-adjustable and period-adjustable function
  - PWM0 clock source: System clock (Fsys) or FIRC (4 MHz), with 1~128 pre-scale option
- PWM1A/PWM1B/PWM1C
  - 8 bits PWM1 with three groups independent duty-adjustable settings and one group shared period-adjustable controlled
  - PWM0 clock source: System clock (Fsys) or FIRC (4 MHz), with 1~128 pre-scale option

13. 16-channel Touch Key

- Built-in algorithm called ETK can scan automatically and determine whether press event occurred
- Interrupt if all Touch Key scan done.
- Four kinds of TK I2C commands to transfer data by I2C interface without interrupt.

14. I2C Interface

- Specific purpose slave I2C interface with interrupt function

15. Four types Reset

- Power On Reset
- Watchdog Reset
- Low Voltage Reset
- External Pin Reset

16. 2-Level Low Voltage Reset: 2.2V / 1.8V

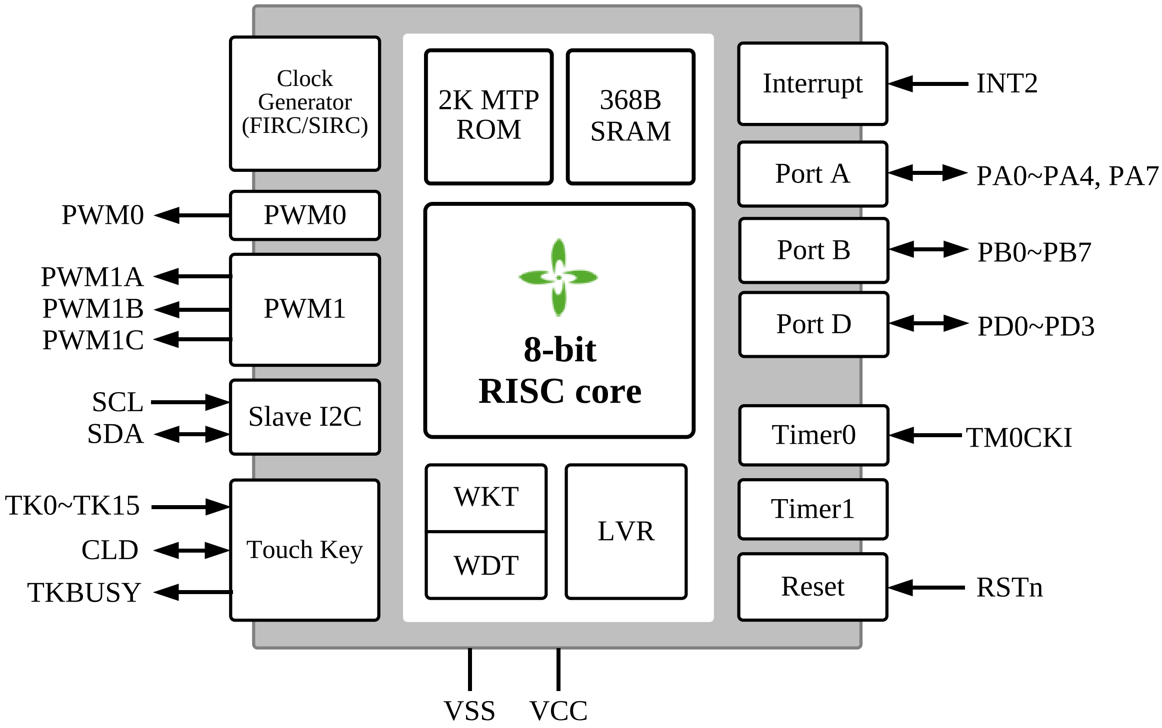
- LVR can’t be disable
- LVR active 1.2ms every 19.2ms

17. Operating Temperature Range: -40°C to +85°C

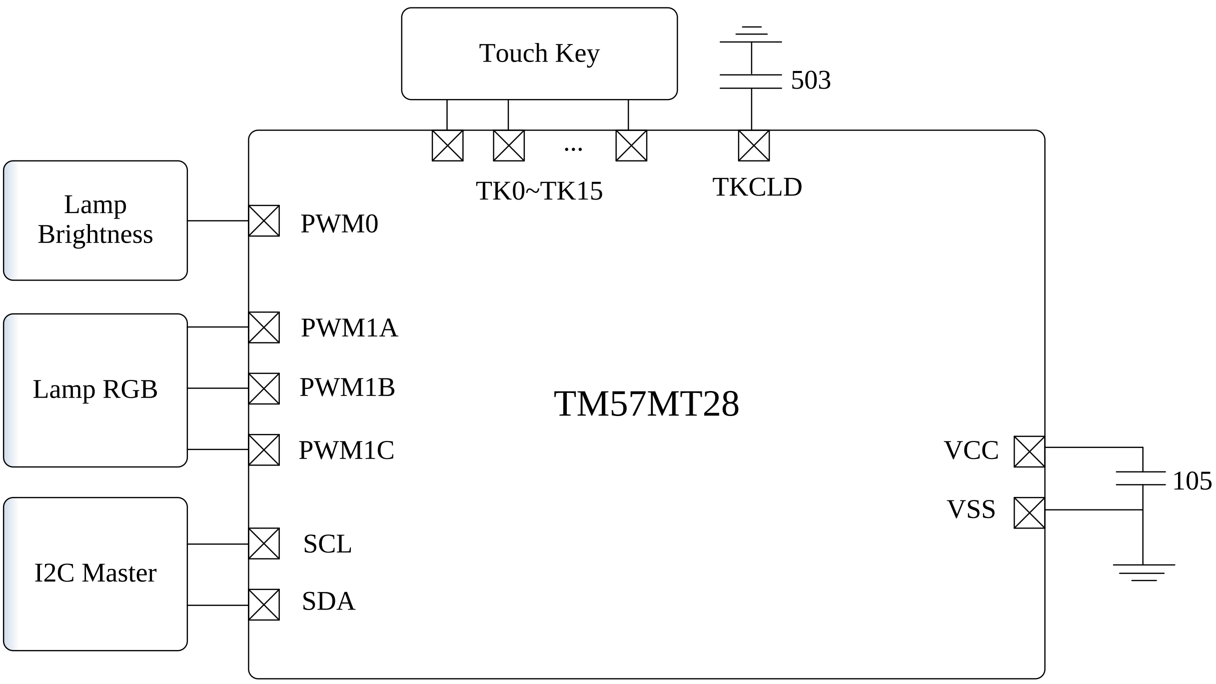
18. Package Types:

- SOP-24/20/16
- QFN-20

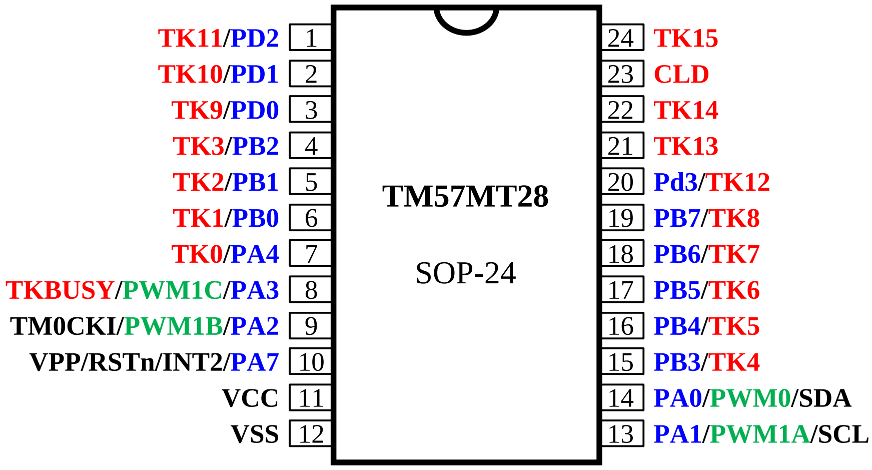
BLOCK DIAGRAM

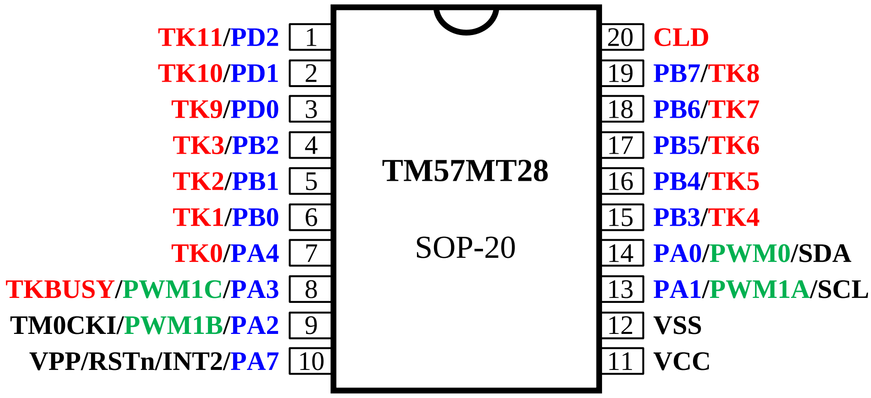


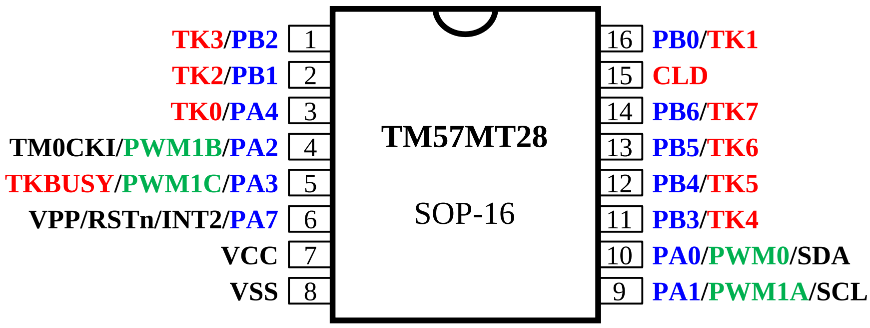
APPLICATION CIRCUIT



PIN ASSIGNMENT







PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0~PA4 PB0~PB7 PD0~PD3	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software.
PA7	I/O	Bit-programmable I/O port for Schmitt-trigger input or open-drain output. Pull-up resistors are assignable by software.
nRESET	I	External active low reset, internal pull-high
VCC, VSS	P	Power Voltage input pin and ground
VPP	I	PROM programming high voltage input
INT2	I	External interrupt input
TM0CKI	I	Timer0's input in counter mode
PWM0	O	PWM0 output
PWM1A	O	PWM1A output
PWM1B	O	PWM1B output
PWM1C	O	PWM1C output
TK0~TK15	I	Touch Key input
TKCLD	I/O	Touch Key external CLD
TKBUSY	O	Touch Key state information output
SDA	I/O	I2C serial data pin
SCL	I	I2C serial clock input

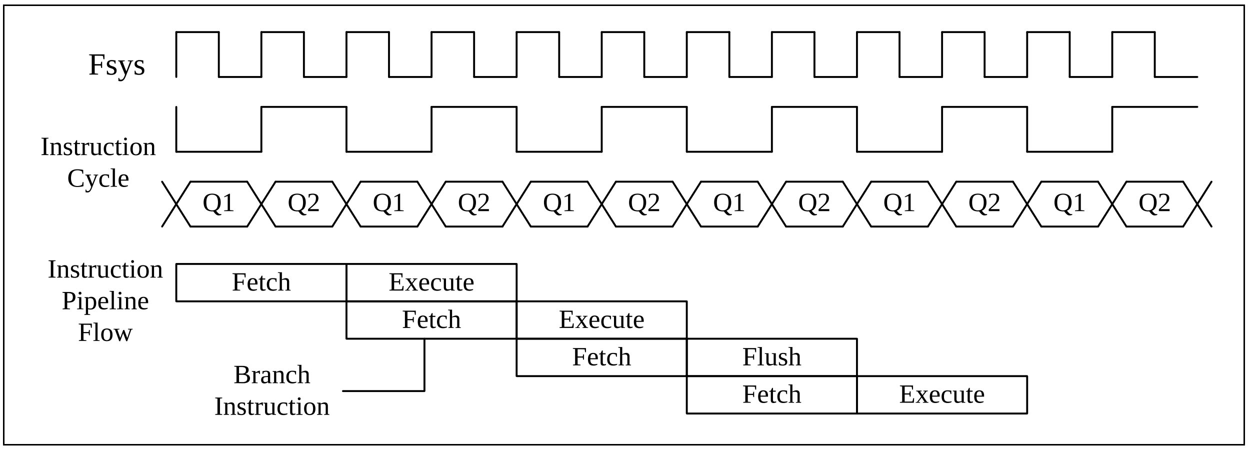
Note: Programming pins are list below. It is better to remove the PCB components connected to these pins during In-Circuit-Programming.
8 wire mode: VCC/VSS/PA0/PA1/PA2/PA3/PA4/PA7(VPP)
5 wire mode: VCC/VSS/PA0/PA1/PA7(VPP)

FUNCTIONAL DESCRIPTION

1. CPU Core

1.1 Clock Scheme and Instruction Cycle

The system clock is internally divided by two to generate Q1 state and Q2 state for each instruction cycle. The Programming Counter (PC) is updated at Q1 and the instruction is fetched from program ROM and latched into the instruction register in Q2. It is then decoded and executed during the following Q1-Q2 cycle. Branch instructions take two cycles since the fetch instruction is ‘flushed’ from the pipeline, while the new instruction is being fetched and then executed.



1.2 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.

1.3 Programming Counter (PC) and Stack

The Programming Counter is 11-bit wide capable of addressing a 2Kx14 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 the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL/GOTO instructions, PC loads 11 bits address from instruction word. For RET/RETI/RETLW instructions, PC retrieves its content from the top level STACK. For the other instructions updating PC [7:0], the PC [10:8] keeps unchanged. Therefore, the data of a lookup table must be located with the same PC [10:8].

The STACK is 11-bit wide and 6-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

For table lookup, the device offer the powerful table read instructions TABRL, TABRH to return the 14-bit ROM data into W by setting the DPTR={DPH, DPL} F-Plane registers.

Example: To look up the MTP data located "TABLE"

```
        ORG     000H          ; Reset Vector
        GOTO    START         ; Goto user program address
START:
        MOVLW   00H
        MOVWF   INDEX         ; Set lookup table's address (INDEX)
LOOP:
        MOVFW   INDEX         ; Move INDEX value to W register
        CALL    TABLE         ; To Lookup data (W = 55H when INDEX = 00H)
        …
        INCF    INDEX, 1      ; Increment the INDEX for next address
        …
        GOTO    LOOP          ; Goto LOOP label

        ORG     X00H          ; X = 1, 2, 3, 4, 5, 6, 7
TABLE:
        ADDWF   PCL, 1        ; (Addr = X00H) 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
```

Note: The chip defines 256 ROM addresses as one page, so that ROM has eight pages, 000H~0FFH, 100H~1FFH, 200H~2FFH, 300H~3FFH, 400H~4FFH, 500H~5FFH, 600H~6FFH, and 700H~7FFH. On the other words, PC [10:8] can be defined 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, 4, 5, 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.

Example: To look up the MTP data located "TABLE" by TABRL and TABRH instructions

```
        ORG     000H                 ; Reset Vector
        GOTO    START                ; Goto user program address
START:
        MOVLW   (TABLE >>8) & 0xff   ; Get high byte address of TABLE label
        MOVWF   DPH                  ; DPH (F1E.2~0) = 02H
        MOVLW   (TABLE) & 0xff       ; Get low byte address of TABLE label
        MOVWF   DPL                  ; DPL (F1D.7~0) = 80H
LOOP:
        TABRL                        ; W = 86H when DTPR = {DPH, DPL} = 0280H
        TABRH                        ; W = 19H when DTPR = {DPH, DPL} = 0280H
        …
        INCF    DPL, 1               ; Increment the DPL for next address
        …
        GOTO    LOOP                 ; Goto LOOP label

        ORG     280H
TABLE:
        DT      0x1986               ; 14-bit ROM data
        DT      0x3719               ; 14-bit ROM data
```

F02	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	PCL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

F02.7~0 **PCL:** Low-byte of Program Counter (PC[7:0])

F0A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCH	–	–	–	–	–	PCH		
R/W	–	–	–	–	–	R	R	R
Reset	–	–	–	–	–	0	0	0

F0A.2~0 **PCH:** High-byte of Program Counter (PC[10:8])

F1D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DPL	DPL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

F1D.7~0 **DPL:** Table read low address, data ROM pointer (DPTR[7:0])

F1E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DPH	–	–	–	–	–	DPH		
R/W	–	–	–	–	–	R/W	R/W	R/W
Reset	–	–	–	–	–	0	0	0

F1E.2~0 **DPH:** Table read high address, data ROM pointer (DPTR[10:8])

1.4 STATUS Register (F-Plane 03H)

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 BCF, BSF and MOVWF 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

<table>
<tr><td>Bit</td><td colspan="2">Description</td></tr>
<tr><td>7</td><td colspan="2">GB1: General Purpose Bit 1</td></tr>
<tr><td>6</td><td colspan="2">GB0: General Purpose Bit 0</td></tr>
<tr><td>5</td><td colspan="2">RAMBK: SRAM Bank Selection
0: SRAM Bank0
1: SRAM Bank1</td></tr>
<tr><td>4</td><td colspan="2">TO: Time Out Flag
0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction
1: WDT time out occurs</td></tr>
<tr><td>3</td><td colspan="2">PD: Power Down Flag
0: after Power On Reset, LVR Reset, or CLRWDT instruction
1: after SLEEP instruction</td></tr>
<tr><td>2</td><td colspan="2">Z: Zero Flag
0: the result of a logic operation is not zero
1: the result of a logic operation is zero</td></tr>
<tr><td rowspan="3">1</td><td colspan="2">DC: Decimal Carry Flag or Decimal /Borrow Flag</td></tr>
<tr><td>ADD instruction</td><td>SUB instruction</td></tr>
<tr><td>0: no carry
1: a carry from the low nibble bits of the result occurs</td><td>0: a borrow from the low nibble bits of the result occurs
1: no borrow</td></tr>
<tr><td rowspan="3">0</td><td colspan="2">C: Carry Flag or /Borrow Flag</td></tr>
<tr><td>ADD instruction</td><td>SUB instruction</td></tr>
<tr><td>0: no carry
1: a carry occurs from the MSB</td><td>0: a borrow occurs from the MSB
1: no borrow</td></tr>
</table>

◇Example: Write immediate data into STATUS register.

```
MOVLW    00H
MOVWF    STATUS       ; Clear STATUS register.
```

◇Example: Bit addressing set and clear STATUS register.

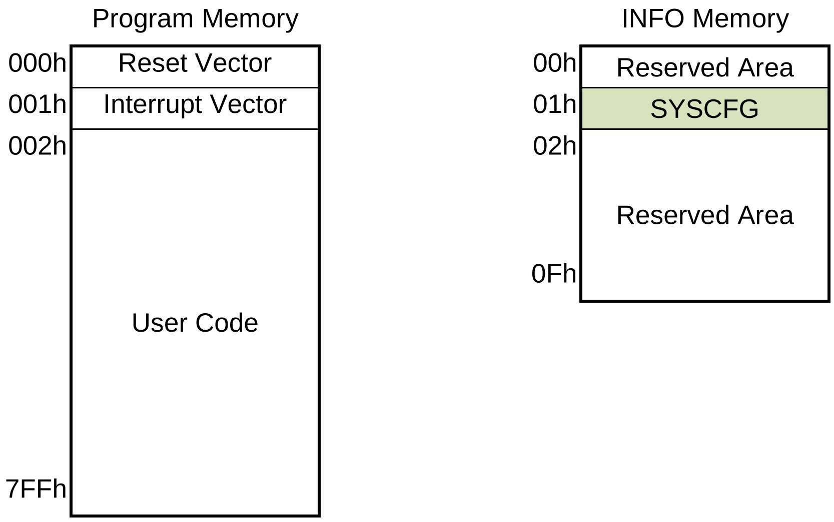
```
BSF      STATUS, 0    ; Set   C = 1.
BCF      STATUS, 0    ; Clear C = 0.
```

◇Example: Determine the C flag by BTFSS instruction.

```
BTFSS    STATUS, 0    ; Check the carry flag
GOTO     LABEL_1      ; If C = 0, goto label_1
GOTO     LABEL_2      ; If C = 1, goto label_2
```

2. Program ROM (MTP)

The MTP Program ROM of this device is 2K words, with an extra INFO area to store the SYSCFG and manufacture data. The MTP ROM can be written multi-times and can be read as long as the PROTECT bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but can be written only when PROTECT is cleared or MTP ROM is blank. That is, unprotect the PROTECT bit can be done only if the Program ROM area is blank. The tenx certified writer can do the above actions with the sophisticated software.



The System Configuration Register (SYSCFG) is located at MTP INFO area. The SYSCFG determines the option for initial condition of MCU. It is written by MTP Writer only. User can select chip operation mode by SYSCFG register.

Bit	Description	
13	**PROTECT**: Code protection selection	
	1	Enable
	0	Disable
12	**XRSTE**: External pin (PA7) reset enable	
	1	Enable
	0	Disable (PA7 as input I/O pin)
11-10	**LVR**: Low voltage reset mode (No disable option, active 1.2ms every 19.2ms)	
	x1	LVR level = 1.8V
	x0	LVR level = 2.2V
9-8	**WDTE**: WDT reset enable	
	11	WDT always enable
	10	WDT enable in FAST/SLOW mode, disable in IDLE mode
	0x	WDT disable
7-0	Tenx Reserved	

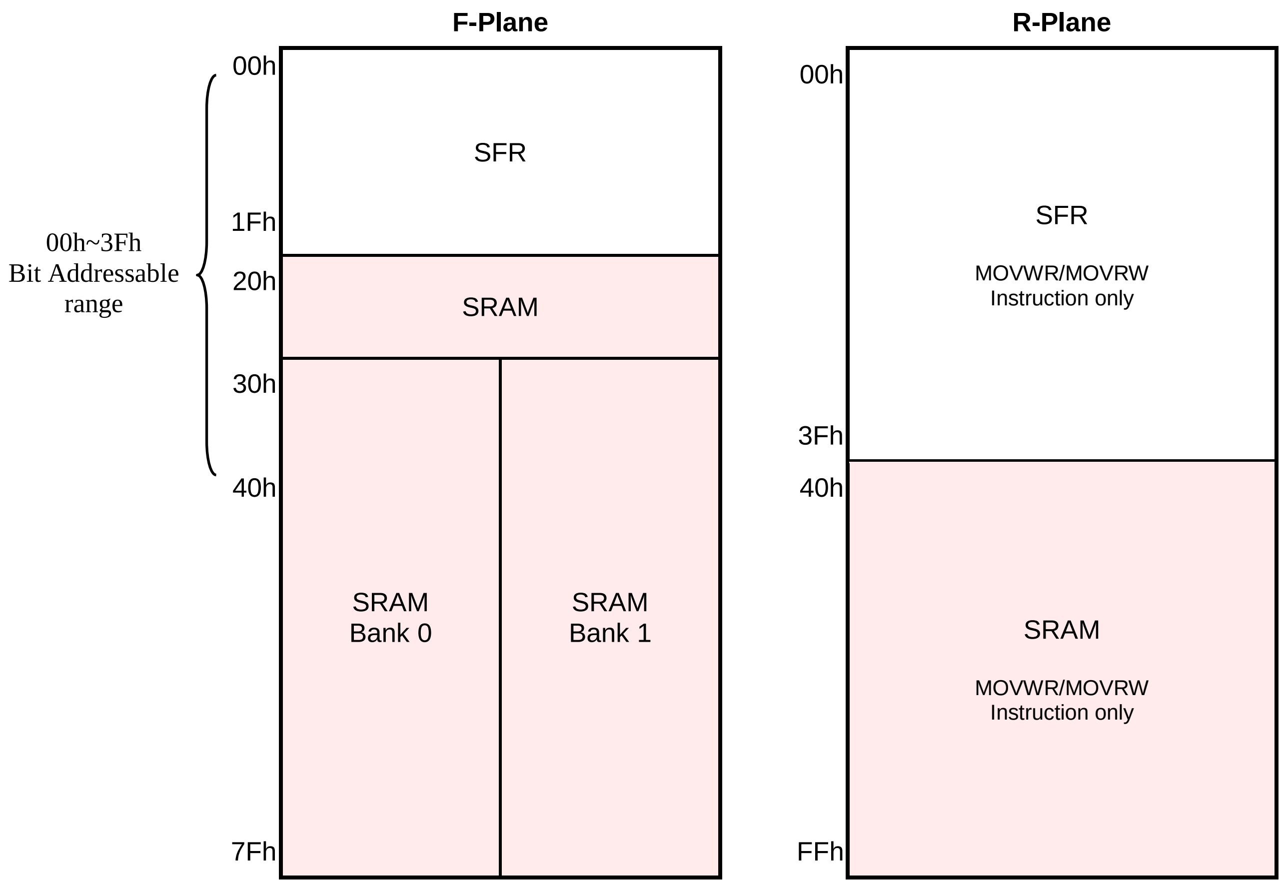
SYSCFG table

3. Data Memory (RAM and SFR)

There are two Data Memory Planes in the chip, F-Plane and R-Plane.

The lower locations of F-Plane are reserved for Special-Function-Register (SFR). Above the SFR is General Purpose Data Memory, implemented as static RAM. F-Plane can be addressed directly or indirectly. Indirect Addressing is made by INDF register. The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a pointer). The first half of F-Plane is bit-addressable, while the second half of F-Plane is not bit-addressable.

R-Plane can also be addressed directly or indirectly. Indirect Addressing is made by INDR register. The INDR register is not a physical register. Addressing INDR actually addresses the register whose address is contained in the RSR register (RSR is a pointer). The R-Plane is not bit-addressable and only supports the MOVWR, MOVRW byte operating instructions.



F-Plane	8/0	9/1	A/2	B/3	C/4	D/5	E/6	F/7
00h	INDF	TM0	PCL	STATUS	FSR	PAD	PBD	PDD
08h	INTIE	INTIF	PCH	CLKCTL	MF0C	PWM0DH	PWM0DL	TKCTL2
10h	PRES_SWH	PRES_SWL	PRES_HWH	PRES_HWL	TM1	TKCTL	I2CRCD0	I2CRCD1
18h	I2CTXD0	I2CTXD1	I2CCTL	I2CFLG	RSR	DPL	DPH	IRCF

R-Plane	8/0	9/1	A/2	B/3	C/4	D/5	E/6	F/7
00h	INDR	TM0RLD	TM0CTL	PWRDN	WDTCLR	PAMODH	PAMODL	PBMODH
08h	PBMODL		PDMODL	MR0B		PWM0PRD	PWM1PRD	TSTREG
10h	PWMCTL	PWM1AD	PWM1BD	PWM1CD	TM1CTL	TM1RLD	TKCHSEL	TKDL
18h	TKBLDH	BSLINEDL	TKCTL3	COLDWAIT	HOTWAIT	TKRSTH	TKRSTL	

Example: Clear R-Plane by indirectly addressing mode

```
        MOVLW   20H     ; W = 20H
        MOVWF   RSR     ; Set R-Plane address to RSR register
LOOP:
        MOVLW   00H
        MOVWR   INDR    ; Clear R-Plane 20H
```

◇Example: Clear all F-plane SRAM data by indirectly addressing mode

```
        MOVLW   20H     ; W = 20H (SRAM start address)
        MOVWF   FSR     ; Set start address of user SRAM into FSR register
LOOP:
        MOVLW   00H
        MOVWF   INDF    ; Clear user SRAM data
        INCF    FSR, 1  ; Increment the FSR for next address
        MOVLW   80H     ; W = 80H (SRAM end address)
        XORWF   FSR, 0  ; Check the FSR is end address of F-plane SRAM
        BTFSS   ZFLAG   ; Check the Zero flag
        GOTO    LOOP    ; If Z = 0, goto LOOP label
        …               ; If Z = 1, exit LOOP
```

F00	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDF	INDF							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

F00.7~0 **INDF:** Not a physical register, addressing INDF actually point to the F-Plane register whose address is contained in the FSR register

F04	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR	GB2	FSR						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

F04.7 **GB2:** General purpose bit

F04.6~0 **FSR:** F-Plane file select register, indirect address mode pointer

F1C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RSR	RSR							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

F1C.7~0 **RSR:** R-Plane file select register, indirect address mode pointer

R00	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDR	INDR							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

R00.7~0 **INDR:** Not a physical register, addressing INDR actually point to the R-Plane register whose address is contained in the RSR register

4. Power Management

The Power-down mode includes IDLE mode. It is activated by SLEEP instruction. During the Power-down mode, the system clock and peripherals stop to minimize power consumption, whether the WKT Timer are working or not depend on F/W setting. The Power-down mode can be terminated by Reset or enabled Interrupts.

R03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRDN	PWRDN							
R/W	W							
Reset	–	–	–	–	–	–	–	–

R03.7~0 **PWRDN:** Write this register to enter Power-down (IDLE) mode

5. Reset

This device can be RESET in four ways.

- Power-On Reset (POR)
- Low Voltage Reset (LVR)
- External Pin Reset (XRST) (PA7)
- Watchdog Reset (WDTR)

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

5.1 Power-on Reset (POR)

Power-on Reset ensures that all registers will be initialized after device power on.

5.2 Low Voltage Reset (LVR)

Low Voltage Reset works when supply voltage is below the threshold level voltage. There are two kinds of threshold level voltage defined by the SYSCFG register.

5.3 External Pin Reset (XRST)

External Pin Reset can be disabled or enabled by the SYSCFG register. It needs to keep at least 2 SIRC clock cycle long to be seen by the device.

5.4 Watchdog Timer Reset (WDTR)

WDT Reset can be disabled or enabled by the SYSCFG register. It runs in Fast/Slow mode and runs or stops in IDLE mode. WDT overflow time period can be defined by WDTPSC. WDT can be cleared by other device reset or CLRWDT instruction. The TO bit of STATUS is set when WDT Reset occurred. TO bit will be cleared after Power-on Reset, LVR Reset, or execution of CLRWDT/SLEEP instruction.

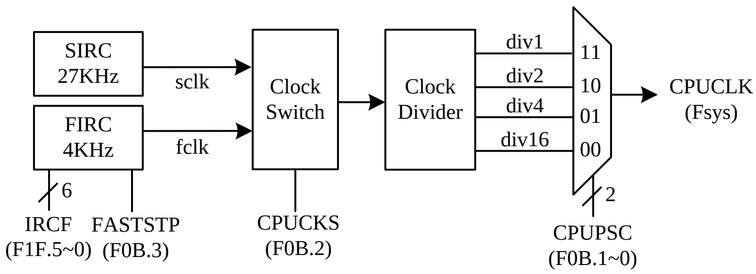
◇ Example: Handling WDT timeout condition

```
ORG 000H
            GOTO        START           ; Jump to user program address.
ORG 010H
START:
            BTFSS       STATUS, TO      ; If TO bit is set, then execute WDT timeout process
            GOTO        NEXT
WDT_Timeout_Process:
            …
            CLRWDT                      ; Clearing WDT is recommended
            …
NEXT:
            …
```

6. Clock Circuitry and Operation Mode

6.1 Dual System Clock

The device is designed with dual-clock system. There are two kinds of clock source, that is, SIRC (Slow Internal RC) and FIRC (Fast Internal RC). Each clock source can be applied to CPU kernel as system clock. SIRC can't be closed in TM57MT28 for TK. Refer to the figure below.



Clock Scheme Block Diagram

FAST Mode:

In FAST mode, CPUCLK is Fast-clock.

The device enters to the FAST mode by setting the CPUCKS=1. In this mode, the system clock source is FIRC. The PWM0 and PWM1 blocks can be driven by CPUCLK (Fsys) or FIRC 4MHz.

SLOW Mode:

In SLOW mode, CPUCLK is Slow-clock.

After power on reset, the device enters to the SLOW mode, which is the default system clock source in the device. In this mode, Slow-clock is enabled, and Fast-clock can be stopped or run by setting FASTSTP. The PWM0 and PWM1 blocks can be driven by CPUCLK (Fsys) or FIRC 4MHz.

IDLE Mode:

In IDLE mode, CPUCLK is stopped.

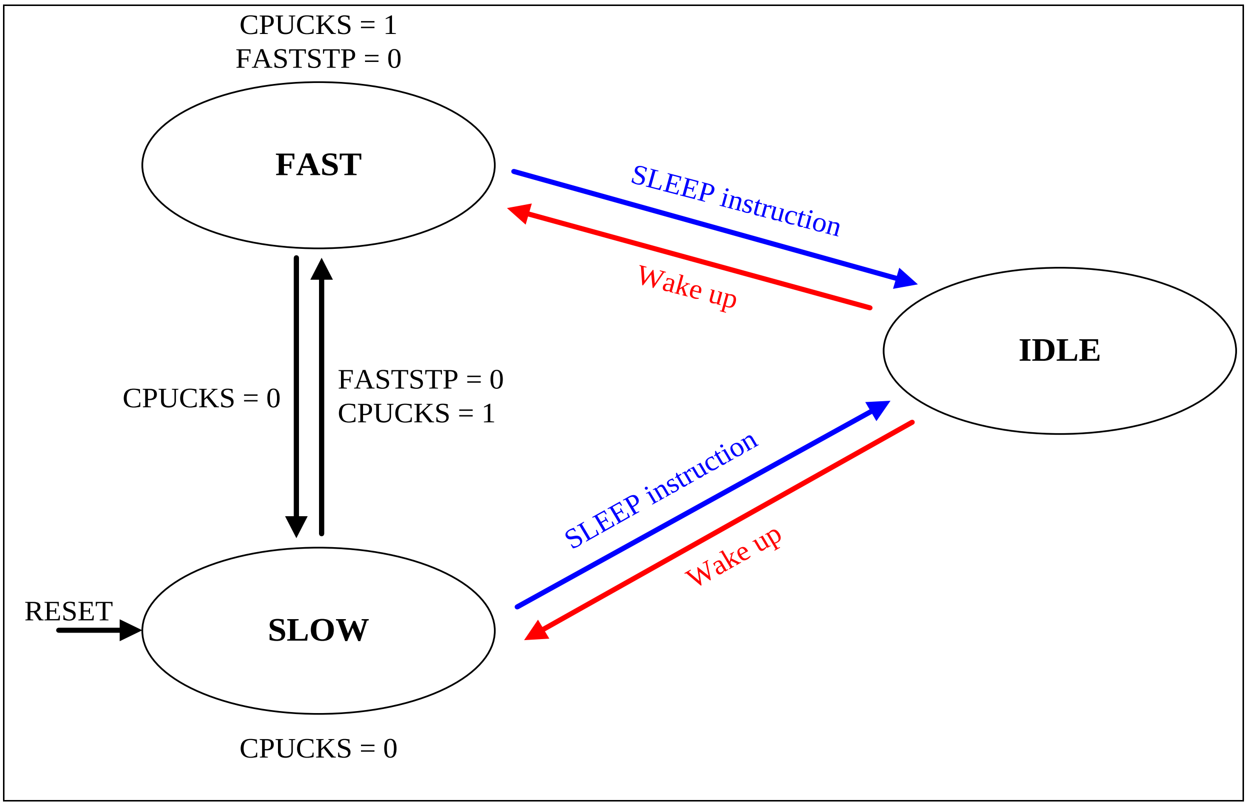
After executing the SLEEP instruction, the device enters to the IDLE mode, SIRC can't be stopped because the ETK blocks will still wakeup periodically for auto-scanning, TK scan frequency can be set by relative TK SFR.

Mode	CPUCLK	FIRC	SIRC	TM0	TM1	PWM0/PWM1	TK
FAST mode	FIRC	Run	Run	CPUCLK/2	CPUCLK/2	CPUCLK or FIRC4M	Run
SLOW mode	SIRC	Set by FASTSTP	Run	CPUCLK/2	CPUCLK/2	CPUCLK or FIRC4M	Run
IDLE mode	Stop	Stop	Run	Stop	Stop	Stop	Run

CLK & relative Functions Table

6.2 Dual System Clock Modes Transition

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



CPU Operation Block Diagram

- **FAST mode switches to SLOW mode**

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

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

◇Example: Switch operating mode from FAST mode to SLOW mode.

```
BCF     CPUCKS      ; Switch system clock source to Slow-clock
BSF     FASTSTP     ; Disable Fast-clock for power-saving (optional)
```

- **SLOW mode switches to FAST mode**

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

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

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

◇Example: Switch operating mode from SLOW mode to FAST mode

```
BCF     FASTSTP     ; Enable Fast-clock.
BSF     CPUCKS      ; Switch system clock source to Fast-clock
```

- **IDLE mode Setting**

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

(1) Execute SLEEP instruction

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

```
SLEEP               ; Enter IDLE mode.
```

IDLE mode can be woken up by I2CIF, TKIF2, TKIF1, TKTOIF, WKTIF and INT2IF.

F0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	-	-	-	-	FASTSTP	CPUCKS	CPUPSC	
R/W	-	-	-	-	R/W	R/W	R/W	
Reset	-	-	-	-	1	0	3	

F0B.3 **FASTSTP**: Fast-clock stop
0: Fast-clock is running
1: Fast-clock stops running

F0B.2 **CPUCKS**: System clock source select
0: Slow-clock
1: Fast-clock

F0B.1~0 **CPUPSC**: System clock source prescaler. System clock source
00: divided by 16
01: divided by 4
10: divided by 2
11: divided by 1

R03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRDN	PWRDN							
R/W	W							
Reset	–	–	–	–	–	–	–	–

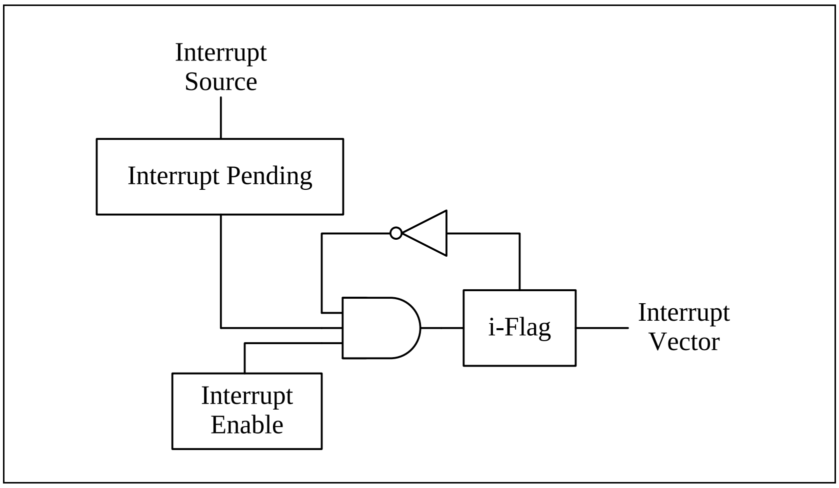
R03.7~0 **PWRDN:** Write this register to enter Power Down mode.

7. Interrupt

This device has 1 level, 1 vector and 8 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual pending flag; no matter its interrupt enable control bit is 0 or 1. Because device has only 1 vector, there is not an interrupt priority register. The interrupt priority is determined by F/W.

If the corresponding interrupt enable bit has been set, it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. In the mean while, a “CALL 001” 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 INT2 (PA7) interrupt request.

```
ORG  000H                                    ; Reset vector.
          GOTO      START                    ; Goto user program address.
ORG  001H                                    ; All interrupt vector.
          GOTO      INT_SUBROUTINE           ;
ORG  002H
START:
          MOVLW     x0xxxxxxB       ; Select INT2 (PA7) pin mode as Mode0
          MOVWR     PAMODH          ; Open drain output low or input with Pull-up
          BSF       PAD7            ; set PA7 as input with Pull-up resistor
          MOVLW     11111011B
          MOVWF     INT2IF          ; Clear INT2 interrupt flag
          BSF       INT2IE          ; Enable INT2 interrupt.
MAIN:
          …
          GOTO      MAIN

  INT_SUBROUTINE:
```

```
        MOVWF     GPR0          ; Push routine to Save W and STATUS data to buffers.
        MOVFW     STATUS        ; F-Plane 03H
        MOVWF     GPR1

        BTFSS     INT2IF        ; Check INT2IF bit.
        GOTO      EXIT_INT      ; INT2IF = 0, exit interrupt vector.
        …                       ; INT2 interrupt service routine.
        MOVLW     11111011B
        MOVWF     INT2IF        ; Clear INT2 interrupt flag
        GOTO      EXIT_INT

EXIT_INT:
        MOVFW     GPR1          ; POP Routine W and STATUS data from buffers.
        MOVWF     STATUS
        MOVFW     GPR0
        RETI
```

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	TKIE2	TKIE1	TM1IE	TM0IE	WKTIE	INT2IE	TKTOIE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F08.7 **TKIE2:** All Touch Key scan done interrupt enable
0: disable
1: enable

F08.6 **TKIE1:** Touch Key Hot/Cold switch interrupt enable
0: disable
1: enable

F08.5 **TM1IE:** Timer1 interrupt enable
0: disable
1: enable

F08.4 **TM0IE:** Timer0 interrupt enable
0: disable
1: enable

F08.3 **WKTIE:** Wakeup Timer interrupt enable
0: disable
1: enable

F08.2 **INT2IE:** INT2 (PA7) interrupt enable
0: disable
1: enable

F08.1 **TKTOIE:** Touch Key time out interrupt enable
0: disable
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	TKIF2	TKIF1	TM1IF	TM0IF	WKTIF	INT2IF	TKTOIF	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F09.7 **TKIF2:** All Touch Key scan done interrupt flag. Write 0 to clear.

F09.6 **TKIF1:** Touch Key Hot/Cold switch interrupt flag. Write 0 to clear.

F09.5 **TM1IF:** Timer1 interrupt event pending flag
This bit is set by H/W while Timer1 overflows. Write 0 to clear.

F09.4 **TM0IF:** Timer0 interrupt event pending flag
This bit is set by H/W while Timer0 overflows. Write 0 to clear.

F09.3 **WKTIF:** Wakeup Timer interrupt event pending flag
This bit is set by H/W while Wakeup Timer is timeout. Write 0 to clear.

F09.2 **INT2IF:** INT2 (PA7) pin falling interrupt pending flag
This bit is set by H/W at INT2 pin's falling edge. Write 0 to clear.

F09.1 **TKTOIF:** Touch Key time out interrupt flag. Write 0 to clear.

F1A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	-	-	-	I2CIE	ICEEN	-	I2CID	
R/W	-	-	-	R/W	R/W	-	R/W	
Reset	-	-	-	0	0	-	0	

F1A.4 **I2CIE:** Slave I2C interface interrupt enable

F1B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	-	I2CIF	TXD1F	TXD0F	RCD1OVF	RCD1F	RCD0OVF	RCD0F
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	0	0	0	0	0	0	0

F1B.6 **I2CIF:** Slave I2C interface interrupt flag. Write 0 to clear.

8. I/O Port

8.1 PA0-4, PB0-PB7, PD0-PD3

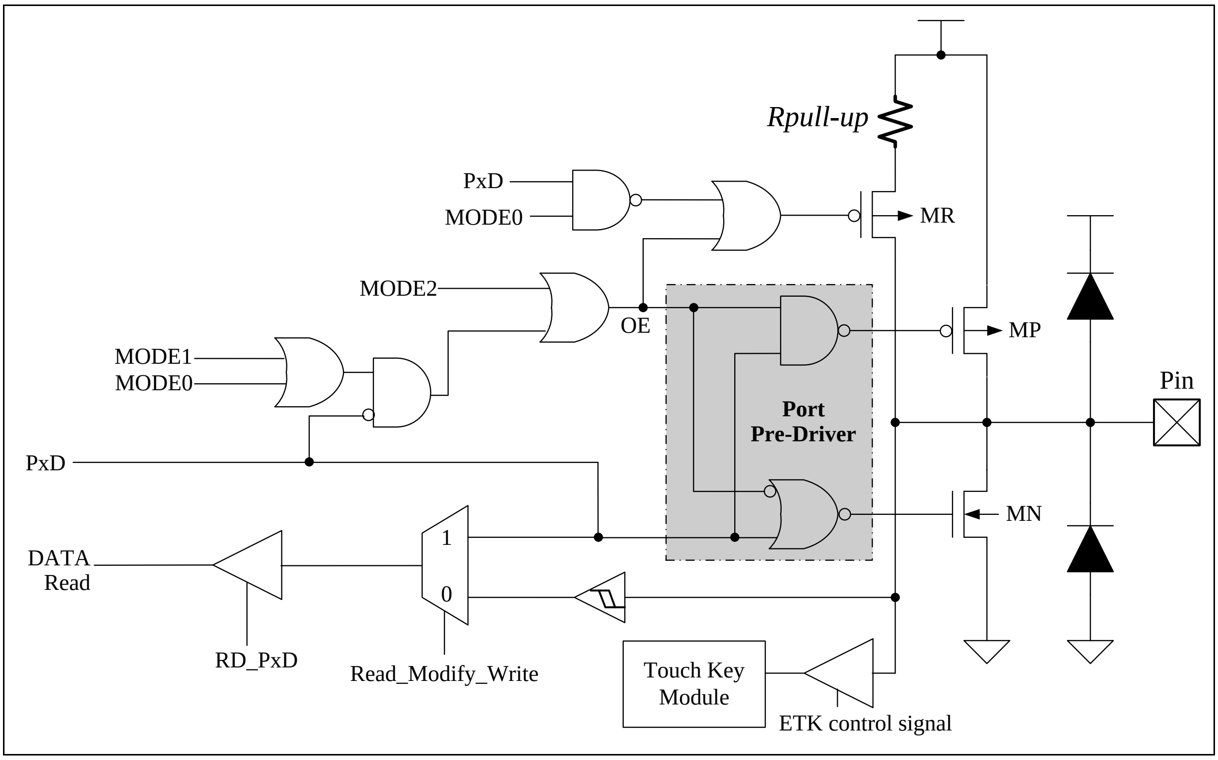
These pins can be used as Schmitt-trigger input, CMOS push-pull output or Open-drain output. The pull-up resistor is assignable to each pin by S/W setting. User can set each pin by their pin mode register. There are 4 kinds of pin modes Mode0, Mode1, Mode2 and Mode3 for each pin can be selected.

Pin Mode	Pin function	PxD SFR data	Pin State	Resistor Pull-up	Digital Input
Mode 0	Open Drain Output Low	0	Drive Low	N	N
	Input with Pull-up	1	Pull-High	Y	Y
Mode 1	Open Drain Output Low	0	Drive Low	N	N
	Input without Pull-up	1	Hi-Z	N	Y
Mode 2	CMOS Push-Pull Output	0	Drive Low	N	N
		1	Drive High	N	N
Mode 3	Alternative function, such as LCD, PWM and RFC	1	–	N	N

I/O Pin Function Table (except PA7)

If a pin is used for Schmitt-trigger input, S/W must set the I/O pin to Mode0 or Mode1 and set the corresponding Port Data SFR to 1 to disable the pin's output driving circuitry. Beside I/O port function, each pin has one or more alternative functions, such as PWM. Most of the functions are activated by setting the individual pin mode control SFR to Mode3.

Reading the pin data has different meaning. In "Read-Modify-Write" instruction, CPU actually reads the output data register. In the other instructions, CPU reads the pin state. The so-called "Read-Modify-Write" instruction includes BSF, BCF and all instructions using F-Plane as destination.



I/O Port Structure (except PA7)

◇Example: Set PA0 as Schmitt-trigger input with pull-up (Mode0)

```
MOVLW    xxxxxxx1B
MOVWF    PAD
MOVLW    xxxxxx00B
MOVWR    PAMODL
```

◇ Example: Set PA0 as Schmitt-trigger input without pull-up (Mode1)

```
MOVLW    xxxxxxx1B
MOVWF    PAD
MOVLW    xxxxxx01B
MOVWR    PAMODL
```

◇ Example: Set PA0 as CMOS push-pull output mode (Mode2)

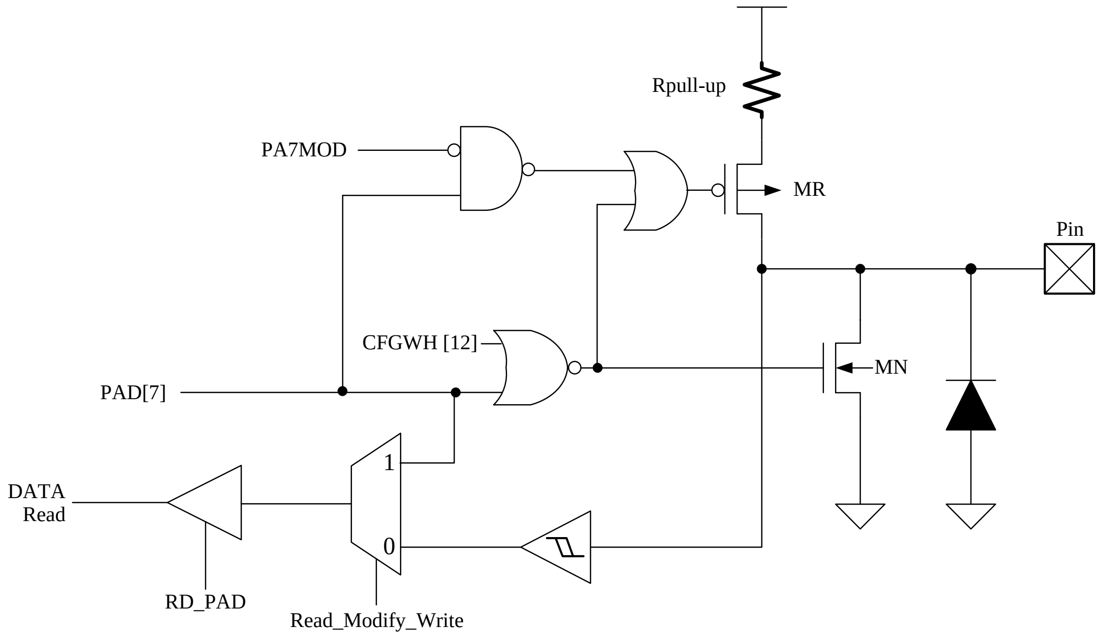
```
MOVLW    xxxxxx10B
MOVWR    PAMODL
```

◇Example: Set PA0 as PWM0 push-pull output mode (Mode3)

```
MOVLW    xxxxxx11B
MOVWR    PAMODL
```

8.2 PA7

PA7 can be used in Schmitt-trigger input or open-drain output which is setting by the PAD[7] (F05.7) bit. When the PAD[7] bit is set, PA7 is assigned as Schmitt-trigger input mode, otherwise is assigned as open-drain output mode and output low. The pull-up resistor is controlled by PA7MOD (R05.6) bit and the default value is enabled (i.e. PA7MOD=0) after system reset. When SYSCFG[12] is set, PA7 is only used in Schmitt-trigger input for external active low reset.

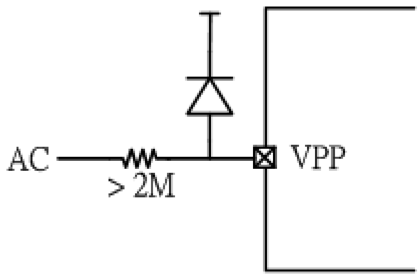


PA7 Structure

How to control PA7 status can be concluded as following list.

CFGWH.12	PA7MOD	PAD[7]	Pin State	Pull-up	MODE
0	0	0	Low	No	open-drain output without pull-high
0	0	1	High	Yes	input with pull-high
0	1	0	Low	No	open-drain output without pull-high
0	1	1	Hi-Z	No	input without pull-high
1	0	1	High	Yes	reset input with pull-high

Note: PA7(VPP) has no high voltage protection diode, need an external diode and resistor to achieve AC zero corssing detection. The reference circuit is shown below.



Zero crossing detector circuit for VPP pin

8.3 Relative Function

Pin	I2C interface	External interrupt	TM0	TK	Other function
PA0	I2CSDA				PWM0
PA1	I2CSCL				PWM1A
PA2			TM0CKI		PWM1B
PA3				TKBUSY	PWM1C
PA4				TK0	
PA7		XINT2			
PB0				TK1	PINWAKUP
PB1				TK2	PINWAKUP
PB2				TK3	PINWAKUP
PB3				TK4	PINWAKUP
PB4				TK5	PINWAKUP
PB5				TK6	PINWAKUP
PB6				TK7	PINWAKUP
PB7				TK8	PINWAKUP
PD0				TK9	
PD1				TK10	
PD2				TK11	
PD3				TK12	

PA0-4, PA7 Function Table

	State	Necessary Setting
I2CSDA	Input/Output	MODE0(with internal pull-up resister) or MODE1 (need external pull-up resister)
I2CSCL*1	Input	MODE0 and PAD1=1 (with internal pull-up resister) or MODE1 and PAD1=1
XINT2*1	Input	MODE0 and PAD7=1 (with internal pull-up resister) or MODE1 and PAD7=1
TM0CKI*1	Input	MODE0 and PAD2=1 (with internal pull-up resister) or MODE1 and PAD2=1
TKBUSY	Output	TKBUSYOE=1
TK0~TK12*2	Input	MODE0 and PAD4=1(with internal pull-up resister)
PWM0	Output	MODE3
PWM1A	Output	MODE3
PWM1B	Output	MODE3

PWM1C	Output	MODE3
PINWAKUP	Input	MODE3

PA0-4, PA7, PB0-PB7,PD0-PD3 Function necessary setting

*1 When user sets MODE1 and PADx=1 for input state, there is no internal pull-up resister, user can connect external pull-up resister by themselves if need.

*2 HW scan each TK channel in order, during the time of scanning TK0~TK12, internal pull-up resister will be closed automatically by HW for analog input.

F05	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

F05.7~0 **PAD:** PA data

F06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBD	PBD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

F06.7~0 **PBD:** PB data

F07	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDD	PDD							
R/W	R/W							
Reset	-	-	-	-	1	1	1	1

F07.7~0 **PDD:** PD data

R05	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMODH	–	PA7MOD	-		-		PA4MOD	
R/W	–	R/W	-		-		R/W	
Reset	–	0	-	-	-	-	0	1

R05.6 **PA7MOD**: PA7 pull-up resistor enable
0: the pin pull-up resistor is enabled
1: the pin pull-up resistor is disabled

R05.1~0 **PA4MOD**: PA4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

PAMODL	PA3MOD		PA2MOD		PA1MOD		PA0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R06.7~6 **PA3MOD**: PA3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R06.5~4 **PA2MOD**: PA2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R06.3~2 **PA1MOD**: PA1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2,
11: Mode3

R06.1~0 **PA0MOD**: PA0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R07	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODH	PB7MOD		PB6MOD		PB5MOD		PB4MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R07.7~6 **PB7MOD**: PB7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R07.5~4 **PB6MOD**: PB6 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R07.3~2 **PB5MOD**: PB5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2,
11: Mode3

R07.1~0 **PB4MOD**: PB4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODL	PB3MOD		PB2MOD		PB1MOD		PB0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R08.7~6 **PB3MOD**: PB3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R08.5~4 **PB2MOD**: PB2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R08.3~2 **PB1MOD**: PB1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2,
11: Mode3

R08.1~0 **PB0MOD**: PB0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R0A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMODL	PD3MOD		PD2MOD		PD1MOD		PD0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R0A.7~6 **PD3MOD**: PD3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R0A.5~4 **PD2MOD**: PD2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

R0A.3~2 **PD1MOD**: PD1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2,
11: Mode3

R0A.1~0 **PD0MOD**: PD0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

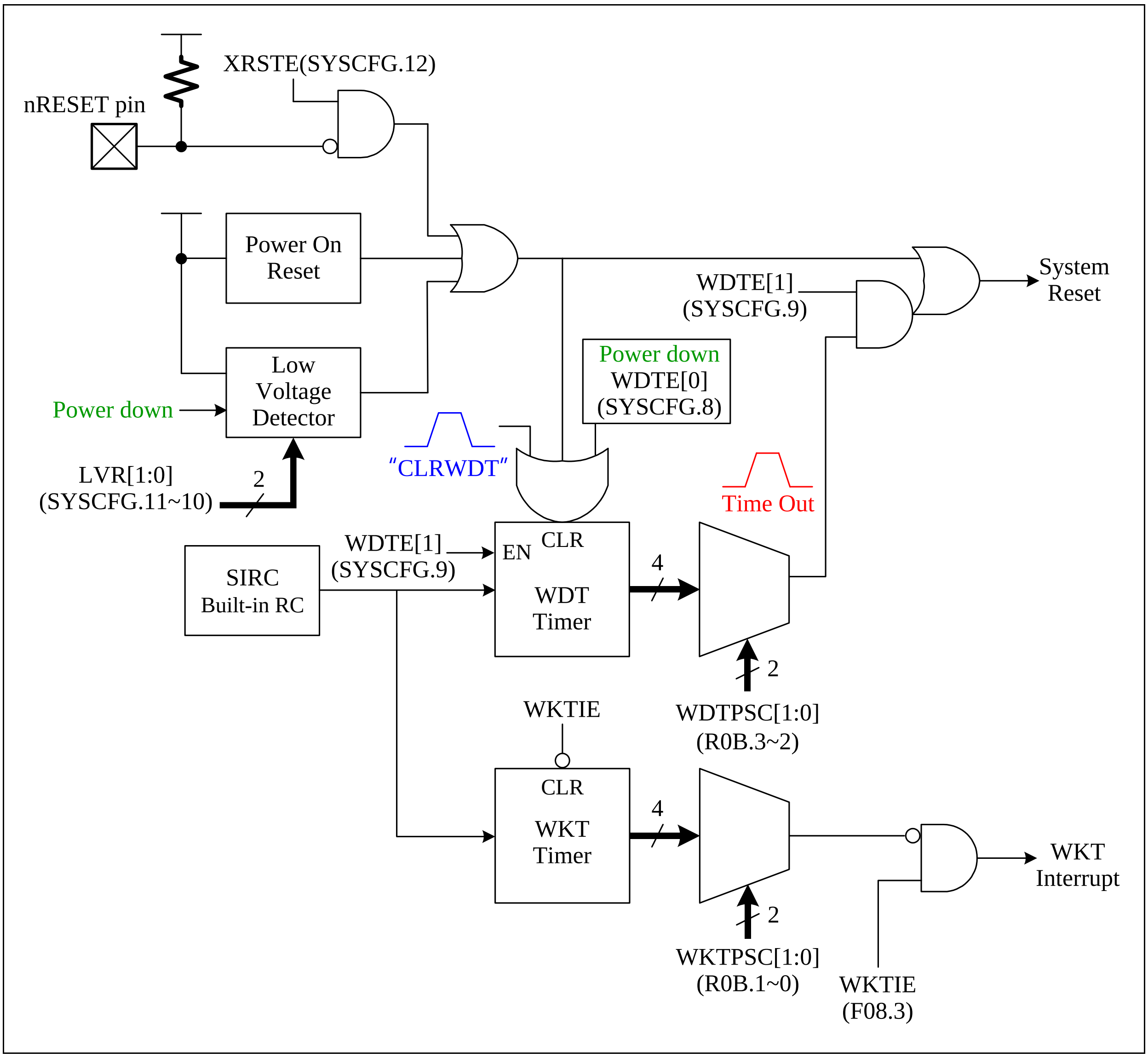
9. Peripheral Functional Block

9.1 Watchdog Timer (WDT) Reset / Wakeup (WKT) Timer Interrupt

The WDT and WKT share the same built-in internal RC Oscillator and have individual own counters. The overflow period of WDT, WKT can be selected by individual prescaler (WDTPSC[1:0], WKTPSC[1:0]).

The WDT timer is cleared by the CLRWDT instruction and moving any value into WDTCLR (R04) is to clear watchdog timer. If the Watchdog is enabled (SYSCFG[9], WDTE[1]=1), the WDT generates the chip reset signal. In IDLE mode, the WDT is only enabled when WDTE[1:0]=11B. Otherwise it will be disabled and stopped for power saving.

The WKT timer is also an internal timer. When WKT overflow, it will generate overflow time out flag “WKTIF” (F09.3). The WKT timer is cleared/stopped by WKTIE=0. Set WKTIE=1, the WKT timer will generate WKT overflow time out interrupt and always count at any CPU operating mode.



WDT/WKT Block Diagram

◇Example: Clear watchdog timer by CLRWDT instruction.

```
MAIN:
        …                                ; Execute program.
        CLRWDT                           ; Execute CLRWDT instruction.
        …
        GOTO        MAIN
```

◇ Example: Clear watchdog timer by write WDTCLR register.

```
MAIN:
        …                                ; Execute program.
        MOVWR       WDTCLR               ; Write any value into WDTCLR register.
        …
        GOTO        MAIN
```

◇ Example: Set WKT period and interrupt function.

```
        MOVLW       xxxxxx10B            ; set WKT period
        MOVWR       MR0B

        MOVLW       11110111B            ; Clear WKT interrupt flag by using byte operation
                                         ; Don’t use bit operation “BCF  WKTIF” to clear flag
        MOVWF       INTIF                ; F-Plane 09H

        MOVLW       00001000B            ; Enable WKT interrupt function
        MOVWF       INTIE
```

F03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	GB1	GB0	RAMBK	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

F03.4 **TO:** WDT time out flag, read-only
0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instructions
1: WDT time out occurs

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	TKIE2	TKIE1	TM1IE	TM0IE	WKTIE	INT2IE	TKTOIE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F08.3 **WKTIE:** Wakeup Timer interrupt enable
0: disable
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	TKIF2	TKIF1	TM1IF	TM0IF	WKTIF	INT2IF	TKTOIF	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F09.3 **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

R04	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WDTCLR	WDTCLR							
R/W	W							
Reset	–	–	–	–	–	–	–	–

R04.7~0 **WDTCLR:** Write this register to clear WDT timer

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	HWAUTO	-	-	-	WDTPSC		WKTPSC	
R/W	R/W	-	-	-	R/W		R/W	
Reset	0	-	-	-	1	1	1	1

R0B.3~2 **WDTPSC:** WDT period
00: 38.4 ms
01: 76.8 ms
10: 307.2 ms
11: 614.4ms

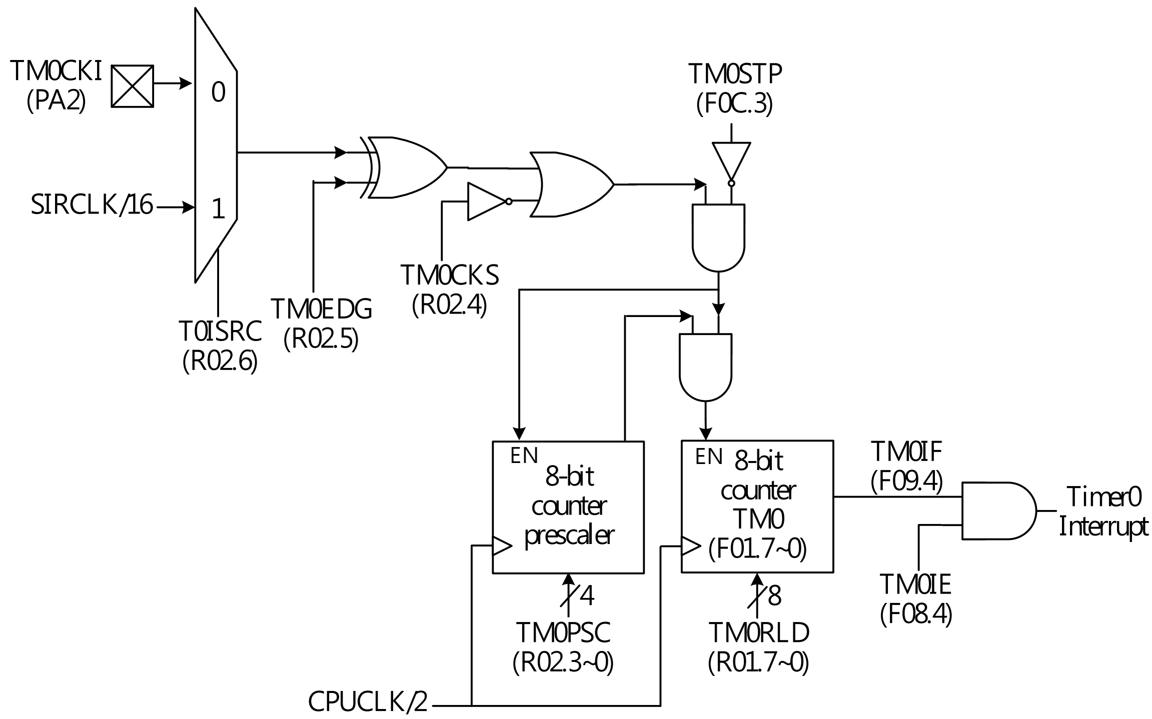
R0B.1~0 **WKTPSC:** WKT period
00: 19.2 ms
01: 38.4 ms
10: 76.8 ms
11: 153.6 ms

9.2 Timer0

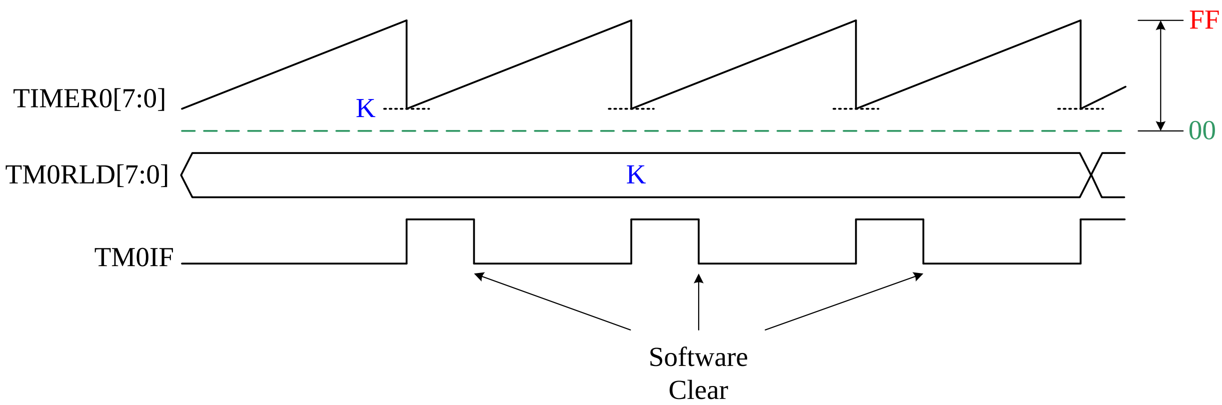
Timer0 is an 8-bit register at F-Plane 01h. It can be read or written as any other register of F-Plane, increases itself periodically and automatically rolls over and the clock source is CPUCLK/2.

If TM0CKS=0, the enable signal of the timer0 counter will keep turning on, and the mode of Timer0 is called Timer mode. If TM0CKS=1, the enable signal of timer0 counter is controlled by TM0CKI pin or SIRCLK/16, and the mode of Timer0 is called Counter mode.

Timer0 can be paused if TM0STP bit is set. User can adjust the rate by TM0PSC. Timer0 generates interrupt flag TM0IF and reload the new data from TM0RLD when its count rolls over. It generates Timer0 Interrupt if TM0IE bit is set.



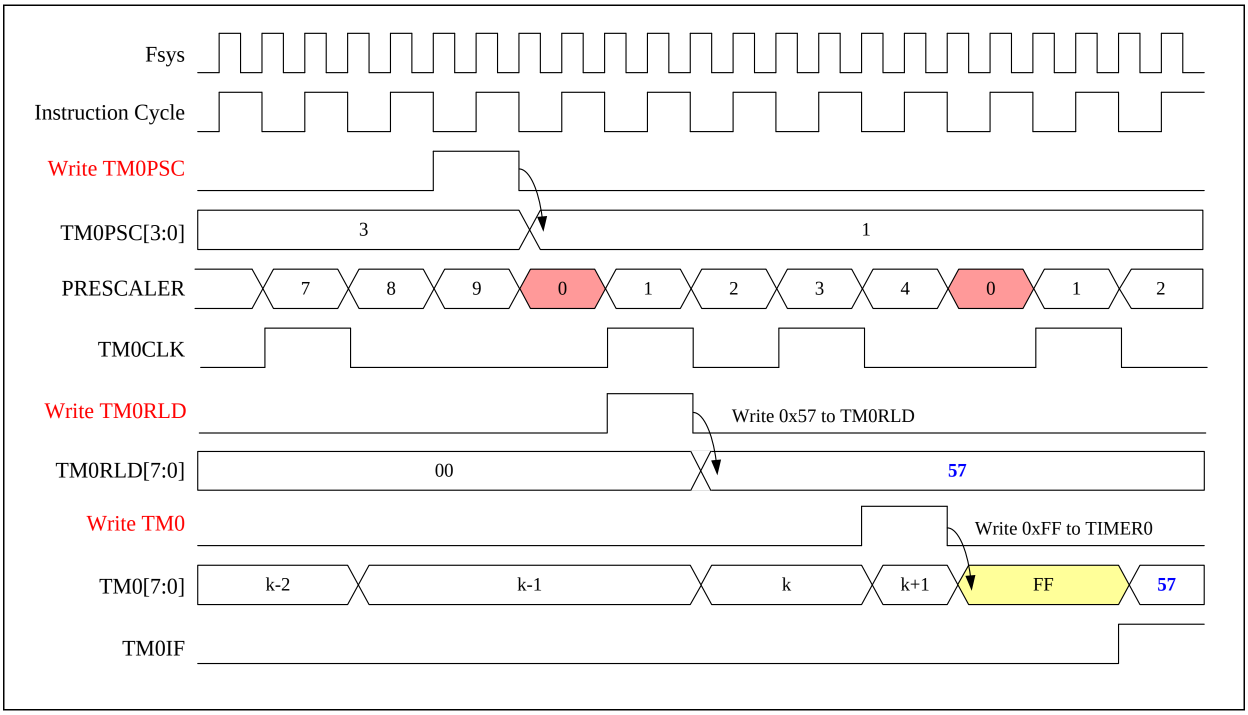
Timer0 Block Diagram



Timer0 Reload Diagram

Timer Mode:

If TM0CKS=0, Timer0 is in Timer mode.



Timer0 works in Timer mode (TM0CKS=0)

◇Example:

```
; Setup Timer0 configuration
        BSF       CPUCKS          ; Set Fast-clock as system clock
        MOVLW     00000101B       ; Timer0 work in Timer mode, scaling factor is 32
        MOVWR     TM0CTL
        MOVLW     7FH             ; Set Timer0 reload data
        MOVWR     TM0RLD

; Initialize Timer0
        BSF       TM0STP          ; Disable Timer0 counting
        MOVLW     00H
        MOVWF     TM0             ; Clear Timer0 content

; Enable Timer0 and interrupt function.
        MOVLW     11101111B       ; Clear Timer0 interrupt flag
        MOVWF     INTIF
        BSF       INTIE           ; Enable Timer0 interrupt function
        BCF       TM0STP          ; Enable Timer0 counting
```

If CPUCLK is FCLK, CPUPSC = div1, Timer0 clock source freq. is CPUCLK/2 = 4MHz / 2 = 2MHz

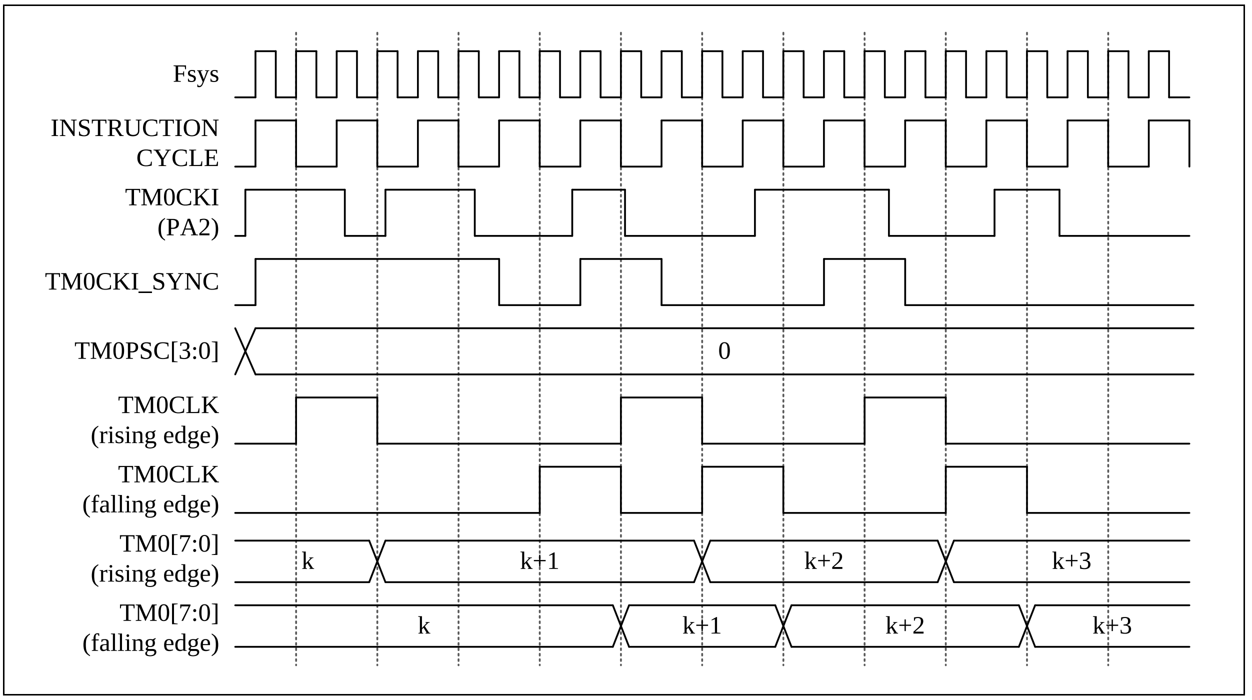
Timer0 interrupt freq. = Timer0 clock source freq. / Timer0 scaling factor / (256-TM0RLD)

= 2 MHz / 32 / (256-128) = 512Hz

Counter Mode:

If TM0CKS=1, Timer0 is in Counter mode. If T0ISRC=0, Counter mode source of Timer0 is TM0CKI pin. If T0ISRC=1, Counter mode source of Timer0 is SRCLK/16.

Counter mode source signal is synchronized by instruction cycle (CPUCLK/2) that means the high/low time durations of Counter mode source must be longer than one instruction cycle time to guarantee each Counter mode source change will be detected correctly by the synchronizer. The following timing diagram describes the Timer0 works in Counter mode.



Timer0 works in Counter mode (TM0CKS=1 and T0ISRC=0)

◇Example: Setup Timer0 work in Counter mode and the counter source is TM0CKI pin

```
; Setup Timer0 configuration
        MOVLW     00000000B     ;Set Counter mode (TM0CKI)
        MOVWR     TM0CTL

; Initialize Timer0
        BSF       TM0STP        ; Disable Timer0 counting
        MOVLW     00H
        MOVWF     TM0           ; Clear Timer0 content

; Start Timer0 count and read Timer0 counter.
        BCF       TM0STP        ; Enable Timer0 counting
        NOP
        NOP
        NOP
        BSF       TM0STP        ; Disable Timer0 counting
        MOVFW     TM0           ; Read Timer0 content
```

F01	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

F01.7~0 **TM0:** Timer0 content

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	TKIE2	TKIE1	TM1IE	TM0IE	WKTIE	INT2IE	TKTOIE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F08.4 **TM0IE:** Timer0 interrupt enable
0: disable
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	TKIF2	TKIF1	TM1IF	TM0IF	WKTIF	INT2IF	TKTOIF	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F09.4 **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

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	-	-	-	VCCFLT	TM0STP	TM1STP	PWM1CLR	PWM0CLR
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	0	0	0	0	0

F0C.3 **TM0STP:** Timer0 counter stop
0: Release
1: Stop counting

R01	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

R01.7~0 **TM0RLD:** Timer0 reload data

R02	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	-	T0ISRC	TM0EDG	TM0CKS	TM0PSC			
R/W	-	R/W	R/W	R/W	R/W			
Reset	-	0	0	0	0	0	0	0

R02.6 **T0ISRC:** Timer0 source select of counter mode
0: T0CKI (PA2)
1: SRCLK/16

R02.5 **TM0EDG:** TM0CKI (PA2) edge selection for Timer0 prescaler count
0: TM0CKI rising edge for Timer0 prescaler count
1: TM0CKI falling edge for Timer0 prescaler count

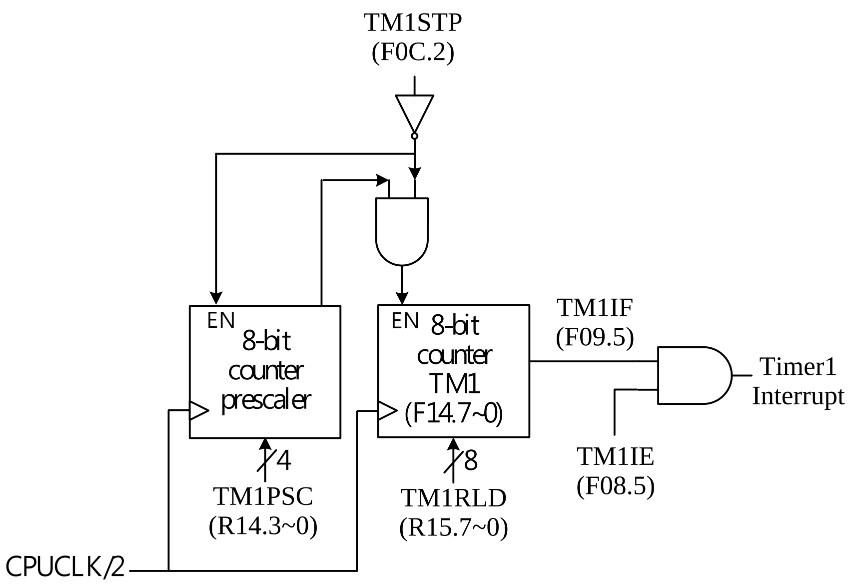
R02.4 **TM0CKS:** Timer0 mode select
0 : Timer mode (Fsys/2)
1 : Counter mode (T0CKI or SRCLK/16)

R02.3~0 **TM0PSC:** Timer0 prescaler. Timer0 clock source
0000: divided by 1

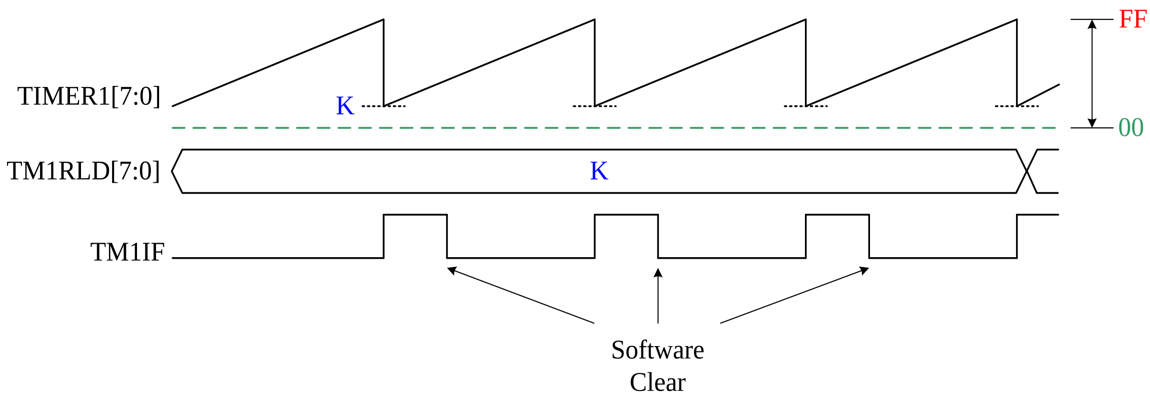
0001: divided by 2
0010: divided by 4
0011: divided by 8
0100: divided by 16
0101: divided by 32
0110: divided by 64
0111: divided by 128
1xxx: divided by 256

9.3 Timer1

Timer1 is an 8-bit register of F-Plane. It can be read or written as any other register of F-Plane. It is almost the same as Timer0, except Timer1 doesn't have Counter Mode. Timer1 increases itself periodically and automatically rolls over and the clock source is CPUCLK/2. Timer1's increasing rate is determined by TM1PSC. Timer1 can generate interrupt flag TM1IF and also reload the new data from TM1RLD when it rolls over. It generates Timer1 interrupt if the TM1IE bit is set. Timer1 can be paused if TM1STP bit is set.

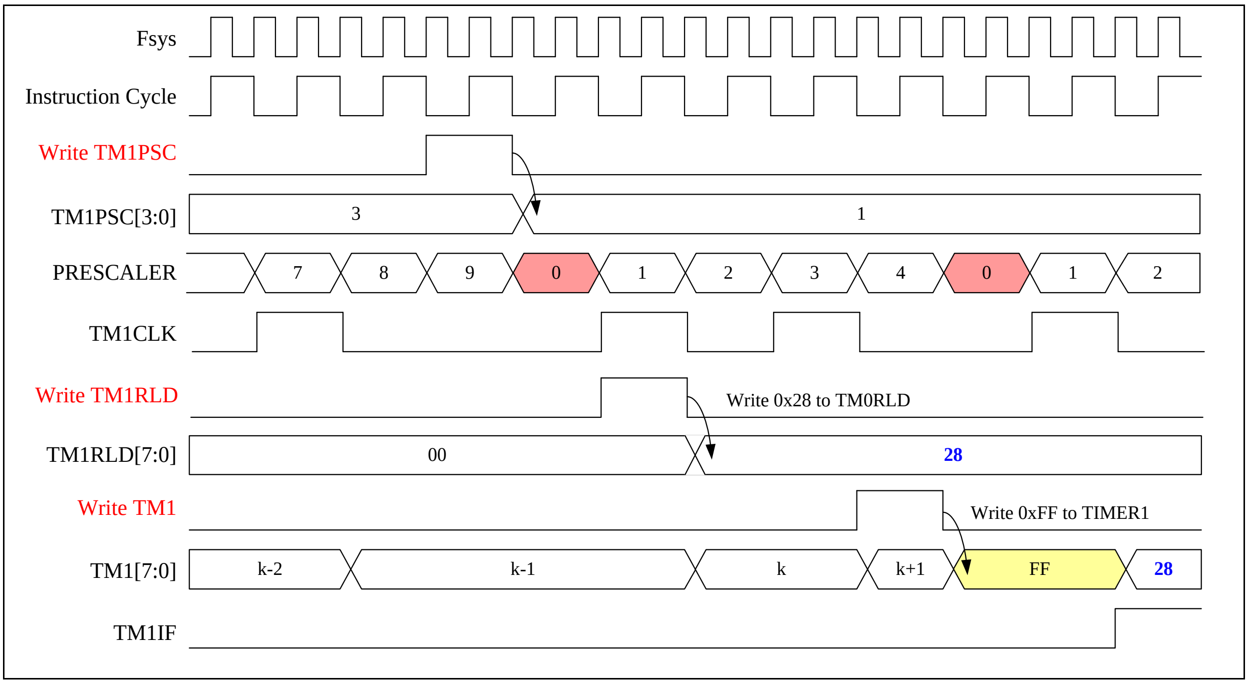


Timer1 Block Diagram



Timer1 Reload Diagram

Timer mode:



Timer1 Timing Diagram

◇Example:

```
; Setup Timer1 configuration
        BSF       CPUCKS       ; Set Fast-clock as system clock
        MOVLW     00000101B    ; TM1PSC = 5 (divided by 32)
        MOVWR     TM1CTL

        MOVLW     7FH
        MOVWR     TM1RLD
; Initialize Timer
        BSF       TM1STP       ; Stop Timer1 counting
        MOVLW     00H
        MOVWF     TM1

; Enable Timer1 timer and interrupt function.
        MOVLW     11011111B    ; Clear Timer1 interrupt flag
        MOVWF     INTIF        ;
        BSF       INTIE        ; Enable Timer1 interrupt function.
        BCF       TM1STP       ; Enable Timer1 counting
```

If CPUCLK is FCLK, CPUPSC = div1, Timer1 clock source freq. is CPUCLK/2 = 4KHz / 2 = 2MHz

Timer1 interrupt freq. = Timer1 clock source freq. / Timer1 scaling factor / (256-TM1RLD)

= 2MHz / 32 / (256-128) = 512Hz

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	TKIE2	TKIE1	TM1IE	TM0IE	WKTIE	INT2IE	TKTOIE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F08.5 **TM1IE:** Timer1 interrupt enable
0: disable
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	TKIF2	TKIF1	TM1IF	TM0IF	WKTIF	INT2IF	TKTOIF	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F09.5 **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

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	-	-	-	VCCFLT	TM0STP	TM1STP	PWM1CLR	PWM0CLR
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	0	0	0	0	0

F0C.2 **TM1STP:** Timer1 counter stop
0: Release
1: Stop counting

F14	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

F14 **TM1:** Timer1 content

R15	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

R15.7~0 **TM1RLD:** Timer1 reload data

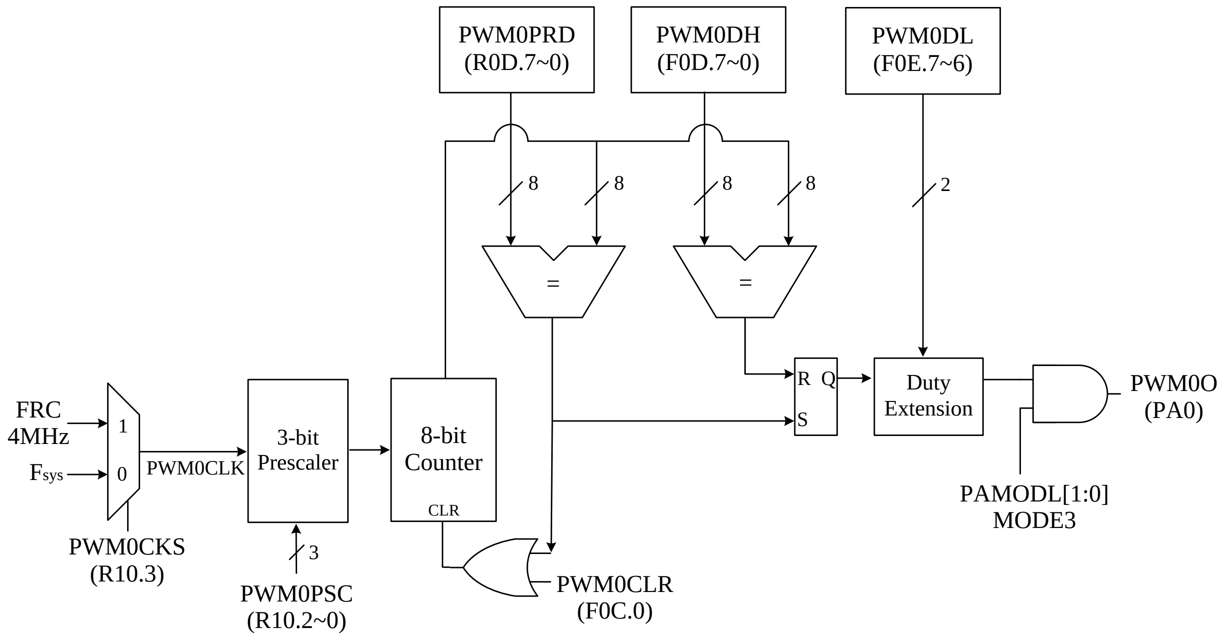
R14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0C	-	-	-	-	TM1PSC			
R/W	-	-	-	-	R/W			
Reset	-	-	-	-	0			

R14.3~0 **TM1PSC:** Timer1 prescaler. Timer1 clock source (Fsys/2)
0000: divided by 1 (Fsys/2)
0001: divided by 2 (Fsys/4)
0010: divided by 4 (Fsys/8)
0011: divided by 8 (Fsys/16)
0100: divided by 16 (Fsys/32)
0101: divided by 32 (Fsys/64)
0110: divided by 64 (Fsys/128)
0111: divided by 128 (Fsys/256)
1xxx: divided by 256 (Fsys/512)

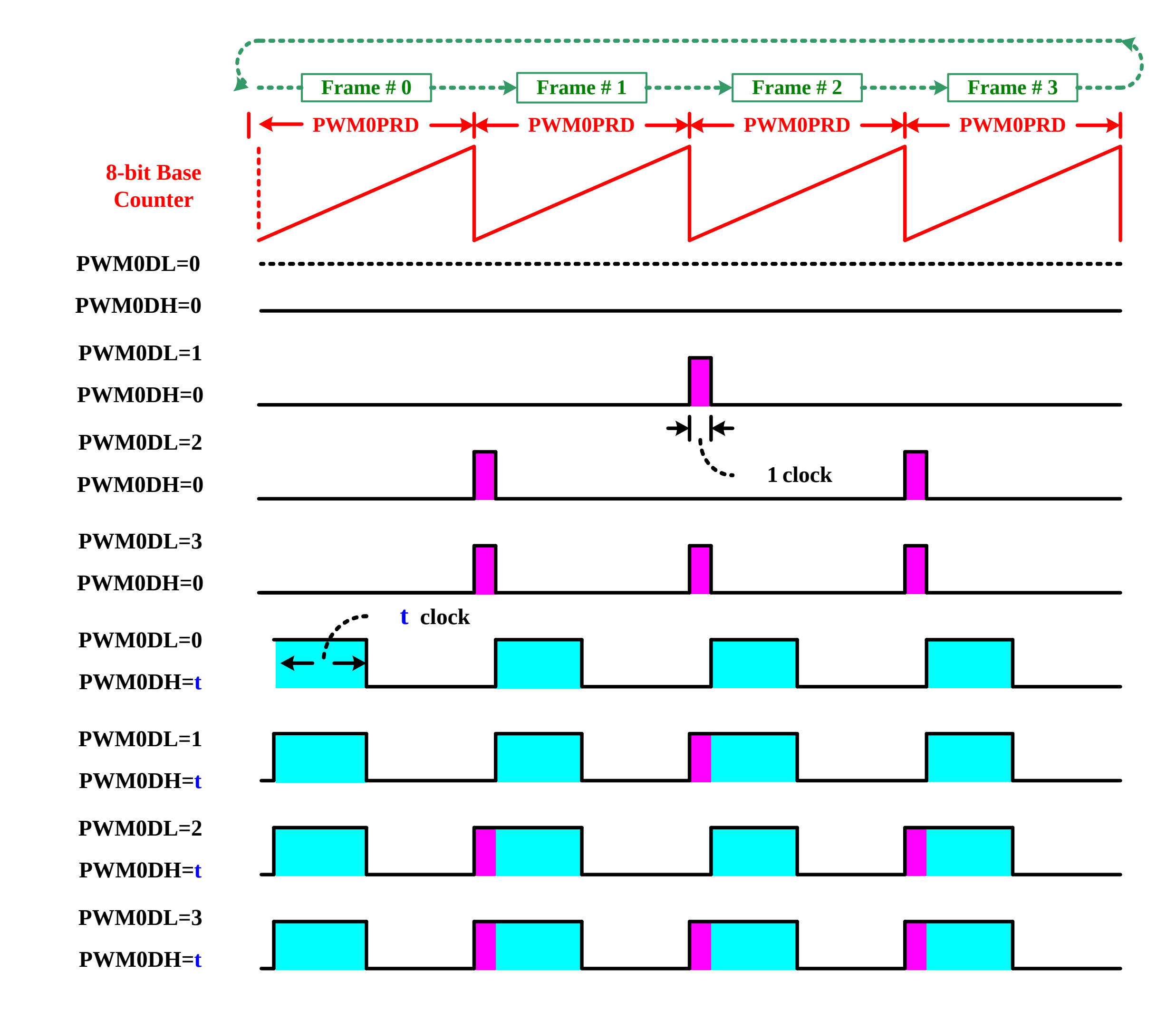
9.4 PWM0: (8+2) bits PWM

The PWM can generate various frequencies with 1024 duty resolution based on PWM0CLK, which can select Fsys or FRC 4 MHz, decided by PWM0CKS. A spread LSB technique allows PWM0 to run its frequency at “PWM0CLK divided by 256” instead of “PWM0CLK divided by 1024”, which means the PWM is 4 times faster than normal. The advantage of higher PWM frequency is that the post RC filter can transform the PWM signal to more stable DC voltage level. The PWM output signal reset to low level whenever the 8-bit base counter matches the 8-bit MSB of PWM duty register PWM0DH. When the base counter rolls over, the 2-bit LSB of PWM duty register PWM0DL decides whether to set the PWM output signal high immediately or set it high after one clock cycle delay.

The PWM0 period can be set by writing period value to PWM0PRD register. Note that changing PWM0PRD will immediately change the PWM0 period, which are different from PWM0DH/PWM0DL which has buffer to update the duty at the end of current period. The Programmer must pay attention to the current time to change PWM0PRD by observing the following figure. There is a digital comparator that compares the PWM0 counter and PWM0RD, if PWM0 counter is larger than PWM0PRD after setting PWM0PRD, a fault long PWM cycle will be generated because PWM0 counter must count to overflow then keep counting to PWM0PRD to finish the cycle.



PWM0 Block Diagram



PWM0 (8+2) bits Timing Diagram

◇ Example:

PWM0 clock source=FIRC4 MHz, PWM0PSC=/64, PWM0PRD=7FH,

PWM0DL=00H, PWM0DH=20H

PWM0 output frequency= 4 MHz / 64 / (PWM0PRD+1) = 4 MHz / 64 / 128 = 512 Hz.

PWM0 output duty = PWM0DH / (PWMPROD+1) = 32 / 128 = 25%

```
;Enable PWM0O output

        MOVLW   00000011b
        MOVWR   PAMODL
        BCF     FASTSTP
        BSF     PWM0CLR

;PWM0 clock source=FIRC(4MHz),PWM0 prescaler is div64

        MOVLW   00001110b
        MOVWR   PWMCTL
        MOVLW   7fh
        MOVWR   PWM0PRD
        MOVLW   00h
        MOVWF   PWM0DL
        MOVLW   20h
        MOVWF   PWM0DH
        BCF     PWM0CLR
```

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	-	-	-	VCCFLT	TM0STP	TM1STP	PWM1CLR	PWM0CLR
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	0	0	0	0	0

F0C.0 **PWM0CLR:** PWM0 Clear and Hold
0: PWM0 Running
1: PWM0 Clear and Hold

F0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	PWM0DH							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F0D.7~0 **PWM0DH**: PWM0 duty 8-bit MSB

F0E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL		-	-	-	-	-	-
R/W	R/W		-	-	-	-	-	-
Reset	0	-	-	-	-	-	-	-

F0E.7~0 **PWM0DL**: PWM0 duty 2-bit LSB

R06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

PAMODL	PA3MOD		PA2MOD		PA1MOD		PA0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R06.1~0 **PA0MOD**: PA0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PA0 as PWM0 push-pull output

R0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PRD	PWM0PRD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

R0D.7~0 **PWM0PRD**: PWM0 period data

R10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWM1CKS	PWM1PSC			PWM0CKS	PWM0PSC		
R/W	R/W	R/W			R/W	R/W		
Reset	0	0	0	0	0	0	0	0

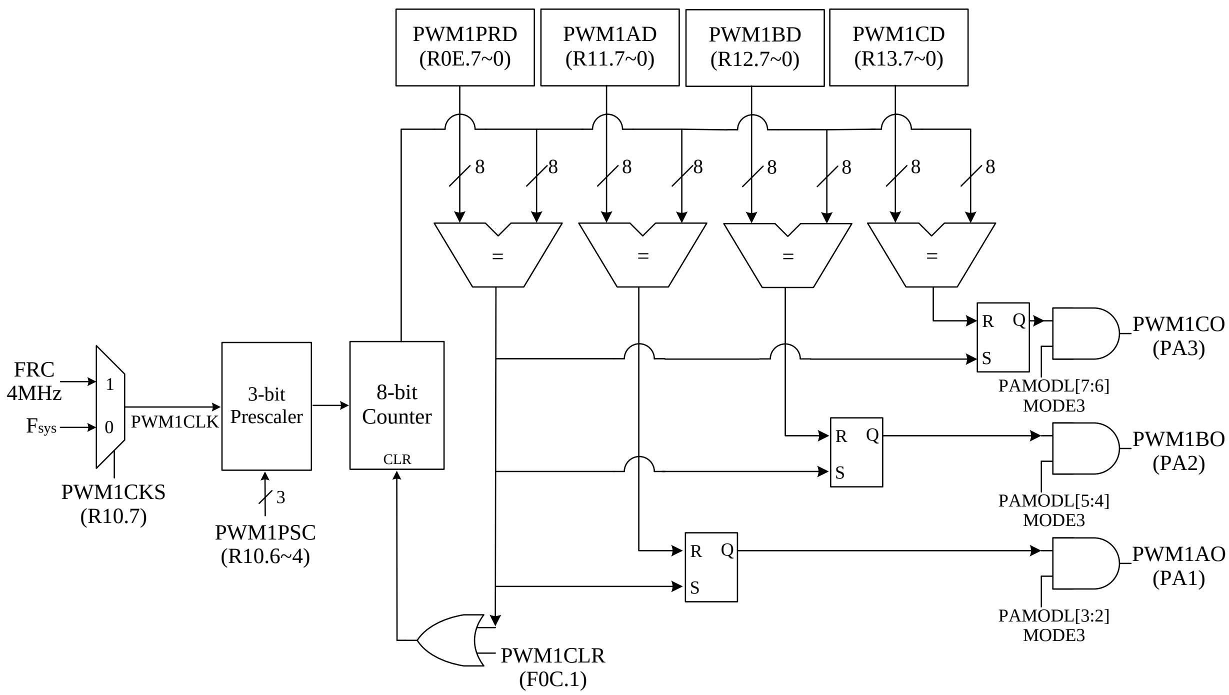
R10.3 **PWM0CKS**: PWM0 clock source
0: Fsys
1: FIRC (4MHz)

R10.2~0 **PWM0PSC**: PWM0 prescaler
000: divided by 1
001: divided by 2
010: divided by 4
…
111: divided by 128

9.5 PWM1A/PWM1B/PWM1C: 8 bits PWMs

PWM1A/PWM1B/PWM1C has independent duty and common period. PWM1 can generate various frequencies with 256 duty resolution based on PWM1CLK, which can select Fsys or FRC 4 MHz, decided by PWM1PSC. The PWM output signal reset to low level whenever the 8-bit base counter matches the 8-bit of PWM duty register PWM1AD/PWM1BD/PWM1CD.

The PWM1A/1B/1C common period can be set by PWM1PRD register. Note that changing PWM1PRD will immediately change the PWM1 period. The Programmer must pay attention to the current time to change PWM1PRD by observing the following figure. There is a digital comparator that compares the PWM1A/1B/1C counter and PWM1PRD. If PWM1A/1B/1C counter is larger than PWM1PRD after setting the PWM1PRD, a fault long PWM cycle will be generated because PWM1A/1B/1C counter must count to overflow then keep counting to PWM1PRD to finish the cycle.



PWM1 Block Diagram

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	-	-	-	VCCFLT	TM0STP	TM1STP	PWM1CLR	PWM0CLR
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	0	0	0	0	0

F0C.1 **PWM1CLR:** PWM1 Clear and Hold
0: PWM1 Running
1: PWM1 Clear and Hold

R06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMODL	PA3MOD		PA2MOD		PA1MOD		PA0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R06.7~6 **PA3MOD**: PA3 Pin Mode Control
00: Mode0

01: Mode1
10: Mode2
11: Mode3, PA3 as PWM1C push-pull output

R06.5~4 **PA2MOD**: PA2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PA2 as PWM1B push-pull output

R06.3~2 **PA1MOD**: PA1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PA1 as PWM1A push-pull output

R0E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1PRD	PWM1PRD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

R0E.7~0 **PWM1PRD**: PWM1 period data

R10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWM1CKS	PWM1PSC			PWM0CKS	PWM0PSC		
R/W	R/W	R/W			R/W	R/W		
Reset	0	0	0	0	0	0	0	0

R10.7 **PWM1CKS**: PWM1 clock source
0: Fsys
1: FIRC (4MHz)

R10.6~4 **PWM1PSC**: PWM1 prescaler
000: divided by 1
001: divided by 2
010: divided by 4
…
111: divided by 128

R11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1AD	PWM1AD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

R11.7~0 **PWM1AD**: PWM1A duty

R12	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1BD	PWM1BD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

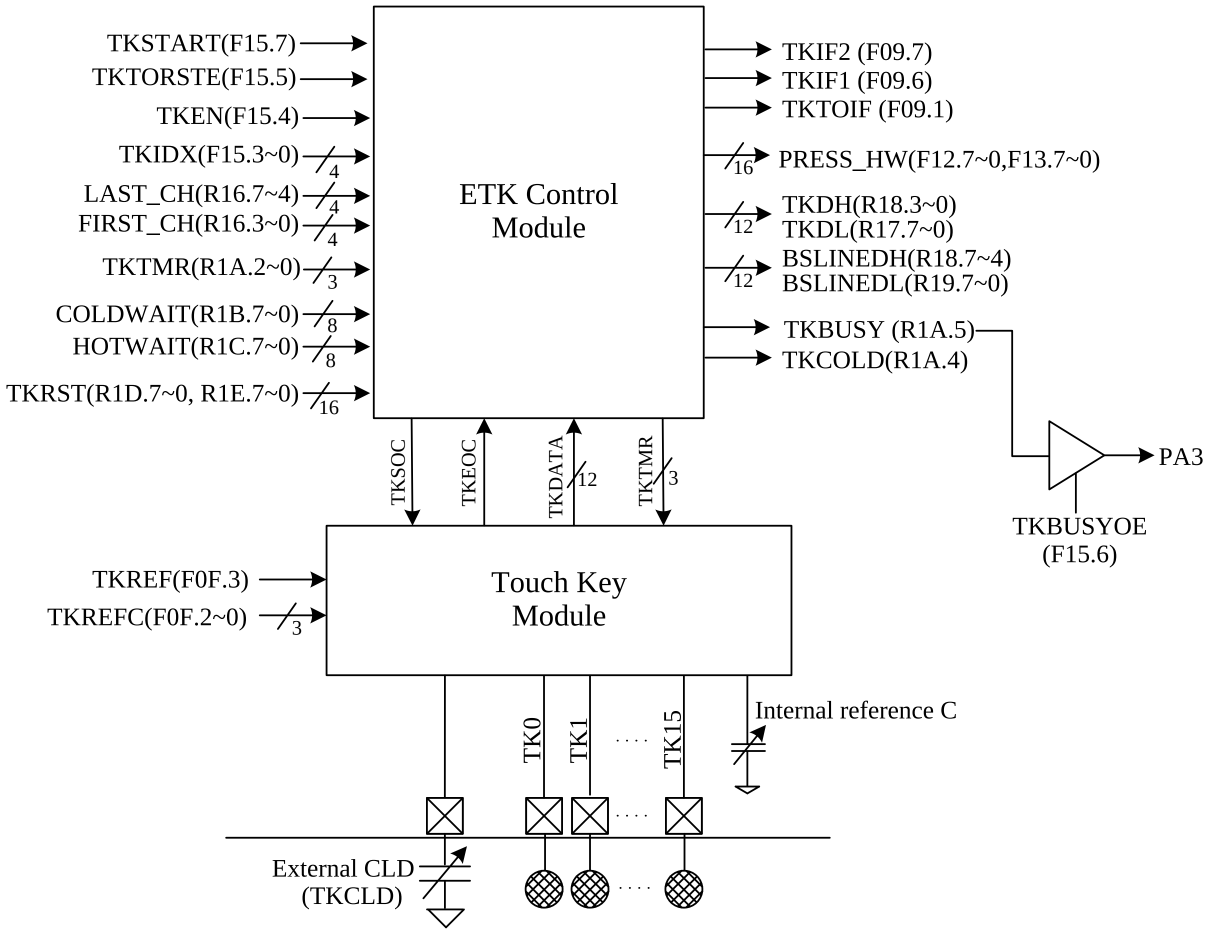
R12.7~0 **PWM1BD**: PWM1B duty

R13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1CD	PWM1CD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

R13.7~0 **PWM1CD**: PWM1C duty

9.6 Touch Key

The Touch Key offers an easy, simple and reliable method to implement finger touch detection. In most applications, it requires an external capacitor component (CLD). The device supports 16 channels Touch Key detection at most, and if some TK pin is not used, it has to be connected to VCC.



Touch Key Block Diagram

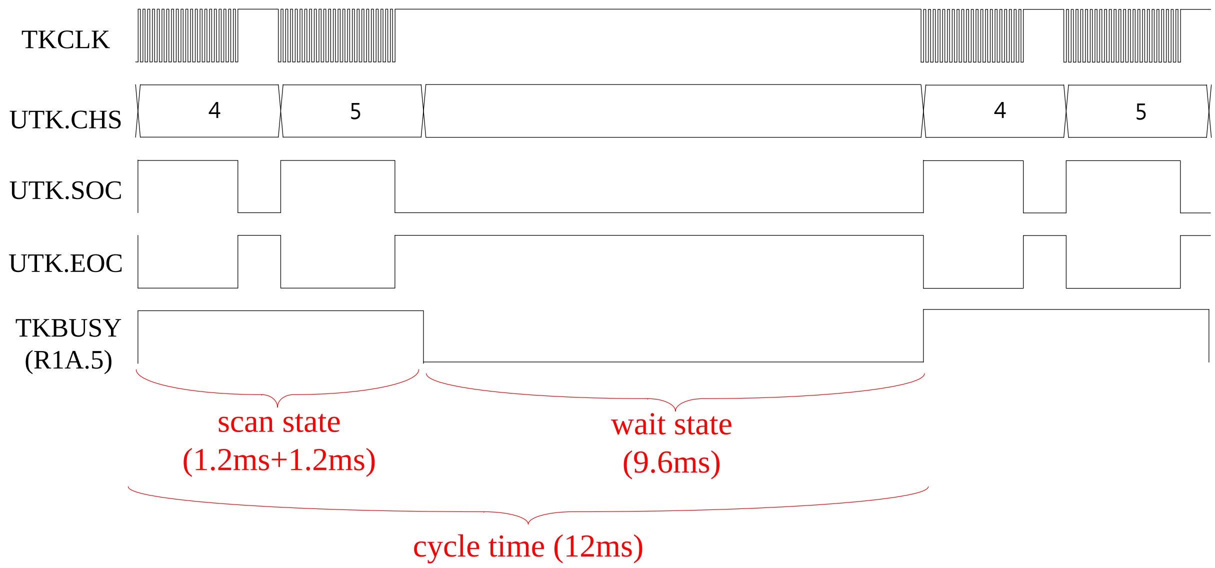
The hardware provides auto scan touch key function called ETK (Enhance auto Touch Key). This function include a series of algorithm, it can catch each touch key scan value (TKDATA) automatically, and adjust environment threshold wisely to decide whether press event occurred or not, user can read PRES_HW value to obtain the press result when ETK interrupt wake up the system.

First, assign TK channel range defined in TKCHSEL and the other TK setting after the chip power on, next, set TKEN to enable TK scan, then, wait at least a complete TK cycle time, last, set TKSTART to enable ETK algorithm for detecting press event. Once user set TKSTART bit, it can be stopped only if system reset.

ETK scan TK channels each cycle time automatically, and the cycle time consists of Scan state and Wait state, user can read TKBUSY bit to know current state is scan state or wait state. The time of Scan state is fixed to 1.2ms each TK channel and the value of Wait state time is depend on SFR selection.

Because the Touch Key clock is working and causing power consumption only when the system scans the touch key value, if we can make the interval more longer between two scan state, in the other words, make the time of Wait state more longer, then, the TK module can save more power. The time of Wait state depends on which current mode is, the hot mode is short and the cold mode is longer. The default mode is hot mode. If there is no variation of touch key pin capacitance occurred for 20 seconds long, the Wait state will switch to cold mode to reduce power consumption. On the other hand, if ETK detect any one of touch key pin capacitance change its value, the Wait state will return to the hot mode. User can obtain which current mode is by TKCOLD bit.

For example, if TKCHSEL=54h and HOTWAIT=8h, and TK is in hot mode, then a whole cycle time is (1.2+1.2+9.6) ms, the figure below shows a whole cycle time and relative signals.



Cycle time of TK

The Touch Key unit has an internal reference capacitor to simulate the TK pin behavior. Setting TKREF can get the TK data count of this reference capacitor. Since the internal capacitor is not affected by water or mobile phone, it is useful for comparing the environment background noise. Setting TKREFC can reduce/increase the value of the internal reference capacitor. It's important to note that the build-in internal capacitor doesn’t have independent registers to store TK data, Baseline data, and TKTMR data, so user have to choose an unused TK channel by TDIDX for storing these information before setting TKREF=1.

The Touch Key scanning result is stored into the 12-bit TK data count register called TKDH and TKDL, and the threshold of ETK algorithm called baseline data is stored into the 12-bit register called BSLINEDH and BSLINEDL, user can reset baseline data by TKRST. User also can reduce/increase TK data count by TKTMR to adjust the TK press sensitivity for adapting different system board circumstances. Because each TK channel have independent register to store TK data, Baseline data and TKTMR, in order to reduce the SFR space cost, hardware define a SFR called TDIDX to switch these registers, user has to use TKIDX to indicate that which TK channel will be chosen when user read TK data, read Baseline data or read/write TKTMR.

Due to synchronization issue, be careful to avoid changing TKTMR, TKHOT, TKCOLD, TKRST value in Scan state.

There is an additional function can be enabled by TKTORSTE bit. If TKTORSTE=1, when system detect more than 30 seconds press, all ETK module signal will be reset excluding TKSTART bit.

◇ Example:

Set Touch Key auto scan Channel = TK4~TK5, TKTMR of TK4 is 1, TKTMR of TK5 is 3.

User has to set TK auto scan range before start ETK and be careful to avoid changing TK setting in Scan state.

```
START:
        movlw  54h        ;7~4:last 3~0:first
        movwr  tkchsel    ;set ETK auto scan range = TK4~TK5
        movlw  08h
        movwr  hotwait
        movlw  40h
        movwr  coldwait
TK4 setting:
        movlw  04h        ;
        movwf  tkctl      ;set index=4
        movlw  01h        ;
        movwr  tkctl3     ;set tktmr of TK4 is 1
TK5 setting:
        movlw  05h        ;
        movwf  tkctl      ;set index=5
        movlw  03h        ;
        movwr  tkctl3     ;set tktmr of TK5 is 3

        movlw  10h        ;7:tkstart 6:tkbusyoe(pa3) 5:torste 4:tken 3~0:tkindex
        movwf  tkctl      ;set TKEN=1

        call   wait       ;wait at least a whole cycle time

        movlw  90h        ;7:tkstart 6:tkbusyoe(pa3) 5:torste 4:tken 3~0:tkindex
        movwf  tkctl      ;set TKSTART=1

        bcf    tkif2
        bsf    tkie2
        sleep
        …
```

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	TKIE2	TKIE	TM1IE	TM0IE	WKTIE	INT2IE	TKTOIE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F08.7 **TKIE2**: All Touch Key scan done interrupt enable
0: disable
1: enable

F08.6 **TKIE:** Touch Key Hot/Cold switch interrupt enable
0: disable
1: enable

F08.1 **TKTOIE:** Touch Key 30-sec time out interrupt enable
0: disable
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	TKIF2	TKIF	TM1IF	TM0IF	WKTIF	INT2IF	TKTOIF	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F09.7 **TKIF2**: All Touch Key scan done interrupt flag.
This interrupt flag is set while end of Scan state. Write 0 to clear.

F09.6 **TKIF**: Touch Key Hot/Cold switch interrupt flag. Write 0 to clear.

F09.1 **TKTOIF**: Touch Key 30-sec time out interrupt flag. If TKTORSTE=1, this interrupt flag is set while 30-second timeout occur. Write 0 to clear.

F0F	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL2	-	-	-	-	TKREF	TKREFC		
R/W	-	-	-	-	R/W	R/W		
Reset	-	-	-	-	0	4		

F0F.3 **TKREF**: Touch Key channel internal reference capacitor select
0: Touch key channel select to TK0~TK15
1: Touch key channel select to Internal reference capacitor

F0F.2~0 **TKREFC**: reduce/increase the value of the internal reference capacitor.
000: Smallest
…
111: Biggest

F10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PRES_SWH	PRES_SWH							
R/W	R/W							
Reset	0							

F10.7~0 **PRES_SWH**: Place TK15~TK8 press result by user. There is no function in PRES_SW, user can use own press algorithm and put the press result in PRES_SW.

F11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PRES_SWL	PRES_SWL							
R/W	R/W							
Reset	0							

F11.7~0 **PRES_SWL**: Place TK7~TK0 press result by user. There is no function in PRES_SW, user can use own press algorithm and put the press result in PRES_SW.

F12	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PRES_HWH	PRES_HWH							
R/W	R							
Reset	0							

F12.7~0 **PRES_HWH**: TK15~TK8 press result by built-in algorithm (ETK) 1: press, 0: no press

F13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PRES_HWL	PRES_HWL							
R/W	R							
Reset	0							

F13.7~0 **PRES_HWL**: TK7~TK0 press result by built-in algorithm (ETK) 1: press, 0: no press

F15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL	TKSTART	TKBUSYOE	TKTORSTE	TKEN	TKIDX			
R/W	R/W	R/W	R/W	R/W	R/W			
Reset	0	0	0	0	0			

F15.7 **TKSTART**: Set TKSTART to start ETK algorithm detect, HW will show the press result in PRES_HWH and PRES_HWL. Once user set TKSTART to start ETK, it can be stopped only if system reset.

F15.6 **TKBUSYOE**: Touch Key state information output enable (PA3)

F15.5 **TKTORSTE**: 30-second press timeout reset function enable

F15.4 **TKEN**: If TKEN=0, TK scan keep stop in Wait state (TKBUSY=0),
If TKEN=1, TK scan start.

F15.3~0 **TKIDX**: Touch Key index. To indicate that which TK channel will be chosen when user read TK data, Baseline data or read/write TKTMR.

R16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCHSEL	LAST_CH				FIRST_CH			
R/W	R/W				R/W			
Reset	dh				dh			

R16.7~4 **LAST_CH**: Touch Key last scan channel. TK scan range as follow
TK [FIRST_CH] ~TK [LAST_CH].

R16.3~0 **FIRST_CH**: Touch Key first scan channel. TK scan range as follow
TK [FIRST_CH] ~TK [LAST_CH].

R17	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKDL	TKDL							
R/W	R							
Reset	-							

R17.7~0 **TKDL**: TKDATA [7:0]. Touch key data low byte of TK [TKIDX].

R18	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKBLDH	BSLINEDH				TKDH			
R/W	R				R			
Reset	-				-			

R18.7~4 **BSLINEDH**: BSLINED [11:8]. Baseline data high byte of TK [TKIDX].

R18.3~0 **TKDH**: TKDATA [11:8]. Touch key data high byte of TK [TKIDX].

R19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BSLINEDL	BSLINEDL							

R/W	R
Reset	-

R19.7~0 **BSLINEDL**: BSLINED [7:0]. Baseline data low byte of TK [TKIDX].

R1A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL3	-	-	TKBUSY	TKCOLD	-	TKTMR		
R/W	-	-	R	R	-	R/W		
Reset	-	-	-	-	-	2		

R1A.5 **TKBUSY**: Touch Key state information.
0: Wait state
1: Scan state

R1A.4 **TKCOLD**: Touch Key mode information.
0: Wait state is at hot mode.
1: Wait state is at cold mode.

R1A.2~0 **TKTMR**: Touch Key conversion time select. TKTMR of TK [TKIDX].
000: Conversion time shortest
...
111: Conversion time longest

R1B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COLDWAIT	COLDWAIT							
R/W	R/W							
Reset	50h							

R1B.7~0 **COLDWAIT**: TK cold mode wait time
0: 2.4ms
1: 2.4ms
2: 4.8ms
...
ff: 612ms

R1C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
HOTWAIT	HOTWAIT							
R/W	R/W							
Reset	2ah							

R1C.7~0 **HOTWAIT**: TK hot mode wait time
0: 1.2ms
1: 1.2ms
2: 2.4ms
...
ff: 306ms

R1D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKRSTH	TKRSTH							
R/W	R/W							
Reset	0							

R1D.7~0 **TKRSTH**: TK baseline data reset. TKRST [15:8].

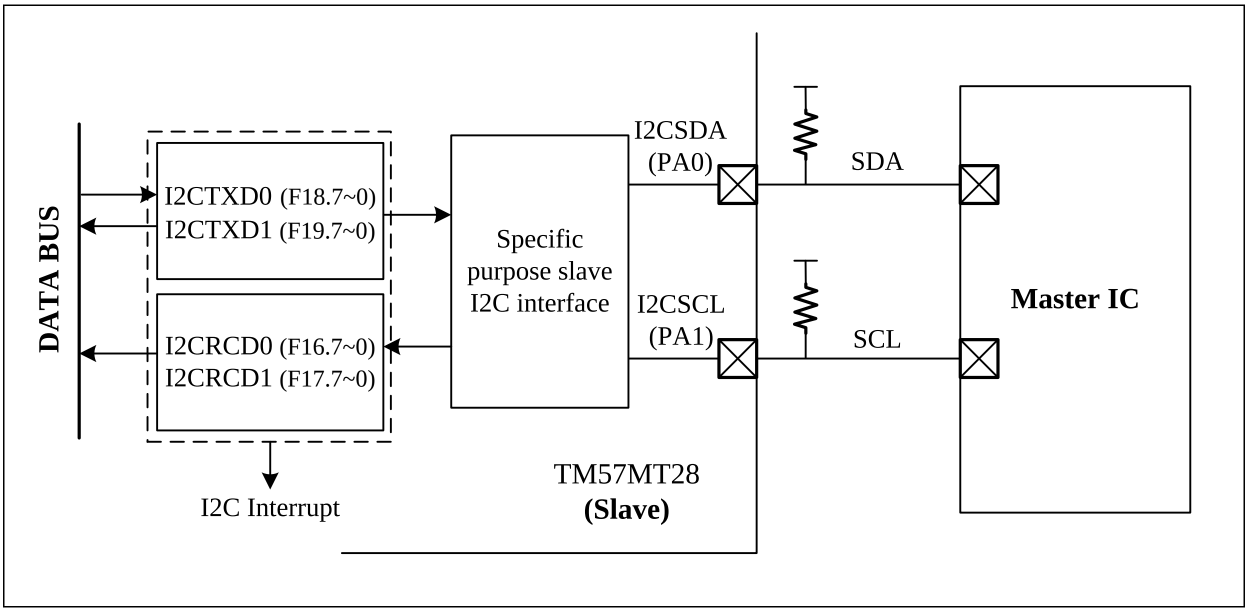
R1E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKRSTL	TKRSTL							
R/W	R/W							

Reset	0

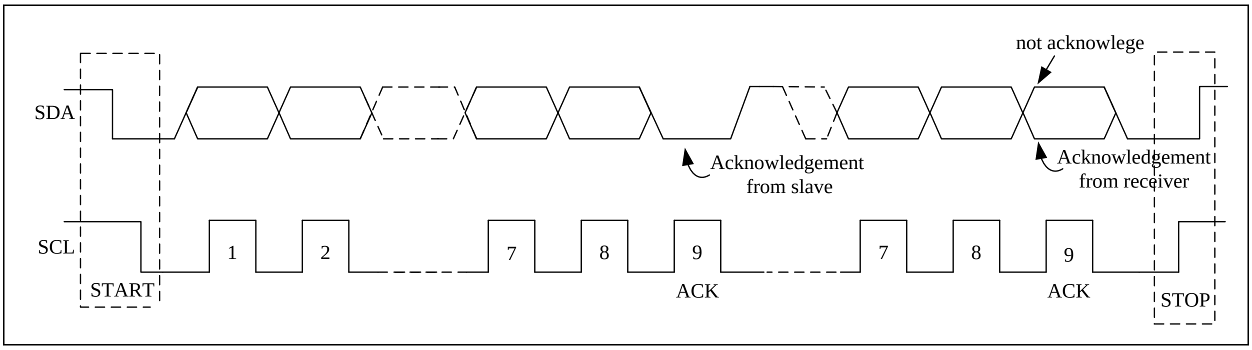
R1E.7~0 **TKRSTL**: TK baseline data reset. TKRST [7:0].

9.7 Specific Purpose Slave I2C Interface

Specific purpose slave I2C interface in TM57MT28 could be used for data transmission. This interface is based on a standard I2C (Inter-Integrated Circuit), and TM57MT28 is always as a slave mode. When the master mode (another IC or device) sends the correct ID through I2C, it can read data from the register I2CTXD0 (F18.7~0) and I2CTXD1 (F19.7~0) of TM57MT28 or write data to the register I2CRCD0 (F16.7~0) and I2CRCD1 (F17.7~0) of TM57MT28.

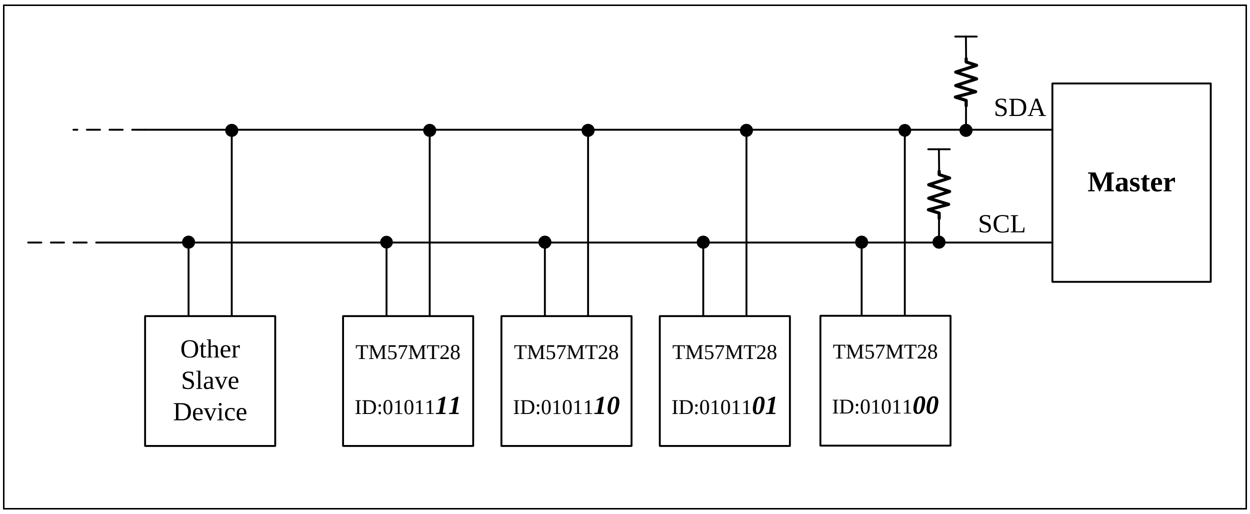


Slave I2C Interface Block Diagram

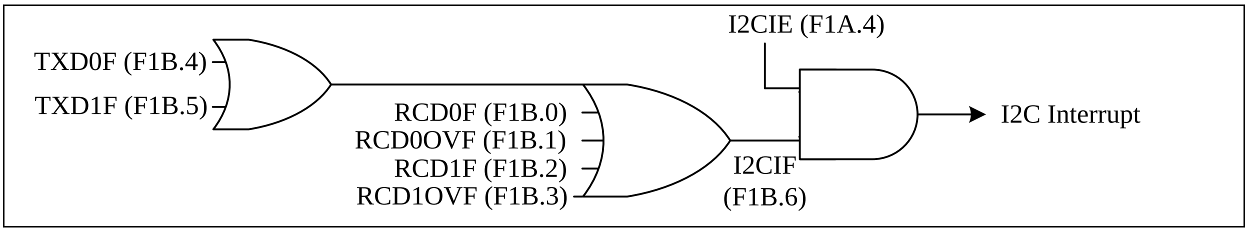


I2C Protocol

To use the slave I2C interface, the I2CEN (F1A.3) bit has to be set. TM57MT28 supports 4 slave device IDs by setting I2CID (F1A.1~0). TM57MT28 can generate the transmitting flag TXD0F (F1B.4) and TXD1F (F1B.5) when data transmitting finished. It generates the receiving flag RCD0F (F1A.0) and RCD1F (F1A.2) when data receiving finished. It can also generate the receiving overflow flag RCD0OVF (F1A.1) and RCD1OVF (F1A.3) when data receiving finished but the receiving flag is not cleared. If one of those I2C flags is set, the I2C interrupt flag I2CIF (F1B.6) will be generated. It generates I2C interrupt if the I2CIE (F1A.4) bit is set. Refer to the following table and figure.



I2C Parallel Connection Application Circuit



Slave I2C Interrupt Block Diagram

RCDxOVF	RCDxF	I2CIF	STATE
0	0	0	IDLE
0	1	1	Data received to I2CRCDx register
1	1	1	Data overflow occurred at I2CRCDx register

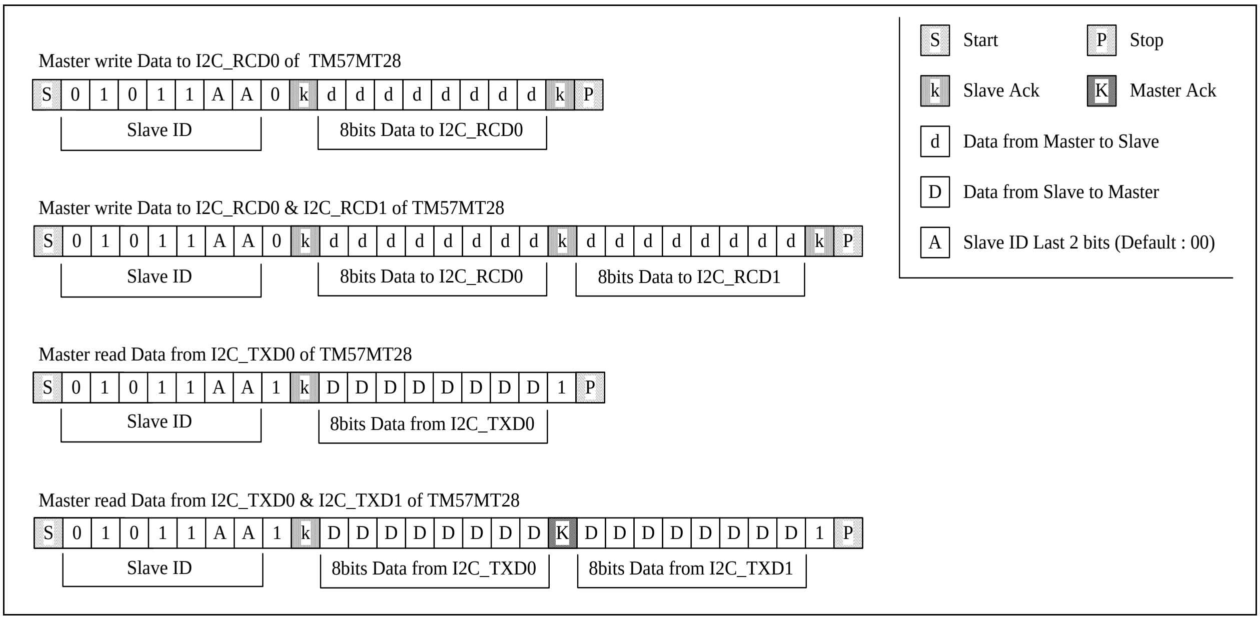


Table of TM57MT28 I2C Commands

F16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CRCD0	I2CRCD0							
R/W	R							
Reset	0	0	0	0	0	0	0	0

F16.7~0 **I2CRCD0:** The receiving register 0 of slave I2C

F17	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CRCD1	I2CRCD1							
R/W	R							
Reset	0	0	0	0	0	0	0	0

F17.7~0 **I2CRCD1:** The receiving register 1 of slave I2C

F18	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CTXD0	I2CTXD0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F18.7~0 **I2CTXD0:** The transmitting register 0 of slave I2C

F19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CTXD1	I2CTXD1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F19.7~0 **I2CTXD1:** The transmitting register 1 of slave I2C

F1A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CCTL	-	-	-	I2CIE	I2CEN	-	I2CID	
R/W	-	-	-	R/W	R/W	-	R/W	
Reset	-	-	-	0	0	-	0	–

F1A.4 **I2CIE:** Slave I2C interrupt enable
0: disable
1: enable

F1A.3 **I2CEN:** Slave I2C SDA output enable
0: disable
1: enable

F1A.1~0 **I2CID**: Slave I2C ID last 2 bits

F1B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CFLAG	-	I2CIF	TXD1F	TXD0F	RCD1OVF	RCD1F	RCD0OVF	RCD0F
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	0	0	0	0	0	0	0

F1B.6 **I2CIF**: Slave I2C interrupt event pending flag
This bit is set by H/W while
a. I2CRCD0 or I2CRCD1 receive data finished
b. I2CRCD0 or I2CRCD1 data overflow occurred
c. I2CTXD0 or I2CTXD1 data transmit finished
write 0 to this bit will clear this flag and slave I2C related flags

F1B.5 **TXD1F**: Slave I2C transmitting data register 1 flag
This bit is set by H/W while I2CTXD1 data transmitting finished, write 0 to this bit will clear this flag

F1B.4 **TXD0F**: Slave I2C transmitting data register 0 flag
This bit is set by H/W while I2CTXD0 data transmitting finished, write 0 to this bit will clear this flag

F1B.3 **RCD1OVF**: Slave I2C receiving data register 1 overflow flag
This bit is set by H/W while receiving data to I2CRCD1 overflow, write 0 to this bit will clear this flag

F1B.2 **RCD1F**: Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, write 0 to this bit will clear this flag

F1B.1 **RCD0OVF**: Slave I2C receiving data register 0 overflow flag
This bit is set by H/W while receiving data to I2CRCD0 overflow, write 0 to this bit will clear this flag

F1B.0 **RCD0F**: Slave I2C receiving data register 0 flag
This bit is set by H/W while data receiving to I2CRCD0 finished, write 0 to this bit will clear this flag

9.8 Touch Key Slave I2C Command

TM57ME28 define four kinds of TK slave I2C command. Before using TK slave I2C command, user has to set I2CEN = 1 to make I2C SDA output enable. TK slave I2C command doesn't have any interrupt or wakeup, and the slave will not check the ACK which the master transmits in Data read of TK I2C command.

The master can only transfer data in Wait state of ETK, and the master can get state information from PA3 after setting TKBUSYOE = 1.

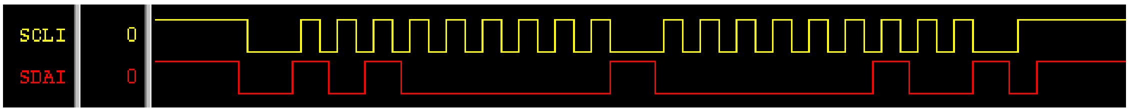
	TK I2C Command	Slave address (hex)	d (data)
1. Data Write	S-1010X000k-dddddddddk-P	a0	tkindex[3:0]
2. Data Read	S-1010X001k-ddddddddK-ddddddddK-ddddddddK-P	a1	{baselineh,tkdatah} tkdatal baselinel
3. Data Read	S-1010X011k-ddddddddK-ddddddddK-P	a3	PRESS_SW[15:0]
4. Data Read	S-1010X101k-ddddddddK-ddddddddK-P	a5	PRESS_HW[15:0]

S : I2C start P : I2C stop k : I2C slave ack, 0 mean success K : I2C master ack, 1 mean last transfer d : data

1. Data Write: S-1010X000k-dddddddddk-P

   The master can use this command to assign a TK index value by I2C interface. If user wants to use this command, user has to set TESTBIT0=1 (R0F.0 = 1) to make the source of TK index switch from SFR to I2C Data Write input.

   First data [7:0] = {4'b0, TKIDX [3:0]}



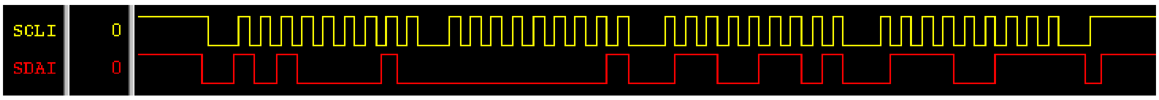
2. Data Read: S-1010X001k-ddddddddK-ddddddddK-ddddddddK-P

   The master can use this command to obtain TK data and baseline data by I2C interface.

   First data [7:0] = {Baseline [11:8], TKDATA [11:8]}

   Second data [7:0] = TKDATA [7:0]

   Third data [7:0] = Baseline [7:0]



3. Data Read: S-1010X011k-ddddddddK-ddddddddK-P

Master can use this command to obtain PRESS_SW by I2C interface.

First data [7:0] = PRES_SW [15:8]

Second data [7:0] = PRES_SW [7:0]

4. Data Read: S-1010X101k-ddddddddK-ddddddddK-P

Master can use this command to obtain PRESS_HW by I2C interface.

First data [7:0] = PRES_HW [15:8]

Second data [7:0] = PRES_HW [7:0]

F15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKCTL	TKSTART	TKBUSYOE	TKTORSTE	TKEN	TKIDX			
R/W	R/W	R/W	R/W	R/W	R/W			
Reset	0	0	0	0	0			

F15.6 **TKBUSYOE:** Touch Key state information output enable (PA3)

F1A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CCTL	-	-	-	I2CIE	I2CEN	-	I2CID	
R/W	-	-	-	R/W	R/W	-	R/W	
Reset	-	-	-	0	0	-	0	–

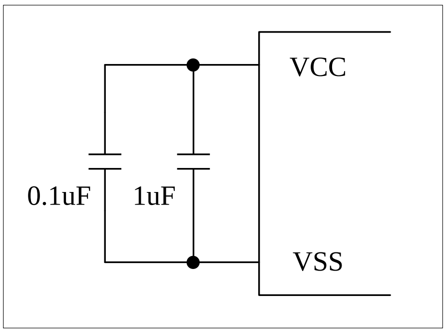
F1A.3 **I2CEN:** Slave I2C SDA output enable
0: disable
1: enable

R0F	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TSTREG	-	-	-	-	-	-	-	TESTBIT0
R/W	-	-	-	-	-	-	-	W
Reset	-	-	-	-	-	-	-	0

R0F.0 **TESTBIT0**: If user wants to use TK I2C Data Write command, user has to set TESTBIT0=1 to make the source of TK index switch from SFR to I2C Data Write input.
0: SFR is the source of TKIDX value
1: TK I2C Data Write input is the source of TKIDX value

9.9 System Clock Oscillator

System clock can be operated in two different oscillation modes. The two oscillation modes are FIRC and SIRC. In the Fast Internal RC mode (FIRC), the on-chip oscillator generates 4 MHz system clock that can be trimmed by IRCF (F1F.5~0). It can separate the frequency into 64 steps. In the Slow Internal RC mode (SIRC), the on-chip oscillator generates 27 KHz system clock. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 μF and 0.1 μF very close to VCC/VSS pins to improve the stability of clock and the overall system.



Internal RC Mode

F1F	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRCF			IRCF					
R/W			R/W					
Reset			by SYSTEM setting					

F1F.5~0 **IRCF:** FIRC frequency adjustment

MEMORY MAP

F-Plane

Name	Address	R/W	Rst	Description
(F00) INDF				**Function related to: FRAM R/W**
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
(F01) TM0				**Function related to: Timer0**
TM0	01.7~0	R/W	0	Timer0 content
(F02) PCL				**Function related to: Program Counter**
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
(F03) STATUS				**Function related to: STATUS**
GB1	03.7	R/W	0	General purpose bit 1
GB0	03.6	R/W	0	General purpose bit 0
RAMBK	03.5	R/W	0	SRAM Bank selection, 0: Bank0, 1: Bank1
TO	03.4	R	0	WDT timeout flag
PD	03.3	R	0	Power-down mode flag
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag or Decimal / Borrow flag
C	03.0	R/W	0	Carry flag or / Borrow flag
(F04) FSR				**Function related to: FRAM R/W**
GB2	04.7	R/W	0	General purpose bit 2
FSR	04.6~0	R/W	-	File Select Register, indirect address mode pointer
(F05) PAD				**Function related to: Port A**
PAD	05.7~0	R	-	Port A pin or “data register” state
		W	FF	Port A output data register
(F06) PBD				**Function related to: Port B**
PBD	06.7~0	R	-	Port B pin or “data register” state
		W	FF	Port B output data register
(F07) PDD				**Function related to: Port D**
PDD	07.7~0	R	-	Port D pin or “data register” state
		W	FF	Port D output data register
(F08) INTIE				**Function related to: Interrupt Enable**

Name	Address	R/W	Rst	Description
TKIE2	08.7	R/W	0	All Touch Key scan done interrupt enable 0: disable 1: enable
TKIE1	08.6	R/W	0	Touch Key Hot/Cold switch interrupt enable 0: disable 1: enable
TM1IE	08.5	R/W	0	Timer1 interrupt enable 0: disable 1: enable
TM0IE	08.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
WKTIE	08.3	R/W	0	Wakeup Timer interrupt enable 0: disable 1: enable
INT2IE	08.2	R/W	0	INT2 (PA7) pin interrupt enable 0: disable 1: enable
TKTOIE	08.1	R/W	0	Touch Key 30-sec time out interrupt enable 0: disable 1: enable
-	08.0	-	-	-
(F09) INTIF				**Function related to: Interrupt Flag**
TKIF2	09.7	R	-	All Touch Key scan done interrupt flag. This interrupt flag is set while end of Scan state.
		W	0	0: clear this flag 1: no action
TKIF1	09.6	R	-	Touch Key Hot/Cold switch interrupt flag.
		W	0	0: clear this flag 1: no action
TM1IF	09.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	0: clear this flag 1: no action
TM0IF	09.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	0: clear this flag 1: no action
WKTIF	09.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	0: clear this flag 1: no action
INT2IF	09.2	R	-	INT2 interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	0: clear this flag 1: no action
TKTOIF	09.1	R	-	Touch Key 30-sec time out interrupt flag. If TKTORSTE=1, this interrupt flag is set while 30-second timeout occur.
		W	0	0: clear this flag 1: no action

Name	Address	R/W	Rst	Description
-	09.0	R	-	-
		W	-	-
(F0A) PCH				**Function related to: Program Counter**
PCH	0a.2~0	R	0	Programming Counter MSB [10~8]
(F0B) CLKCTL				**Function related to: Clock**
FASTSTP	0b.3	R/W	1	Stop Fast-clock 0:no Stop 1:Stop
CPUCKS	0b.2	R/W	0	Select Fast-clock 0: Fsys=Slow-clock 1: Fsys=Fast-clock
CPUPSC	0b.1~0	R/W	3	Fsys Prescaler, 0: div 16 1: div 4 2: div 2 3: div 1
(F0C) MF0C				**Function related to: TM0/TM1/PWM/Filter**
VCCFLT	0c.4	R/W	0	Power noise Filter 0: disable 1: enable
TM0STP	0c.3	R/W	0	Timer0 counter stop 0: Timer0 is counting 1: Timer0 stops counting
TM1STP	0c.2	R/W	0	Timer1 counter stop 0: Timer1 is counting 1: Timer1 stops counting
PWM1CLR	0c.1	R/W	0	PWM1 clear and hold 0: PWM1 is running 1: PWM1 is cleared and hold
PWM0CLR	0c.0	R/W	0	PWM0 clear and hold 0: PWM0 is running 1: PWM0 is cleared and hold
(F0D) PWM0DH				**Function related to: PWM0**
PWM0DH	0d.7~0	R/W	0	PWM0 Duty MSB 8bit, PWM0DT[9:2]
(F0E) PWM0DL				**Function related to: PWM0**
PWM0DL	0e.7~6	R/W	0	PWM0 Duty LSB 2bit, PWM0DT[1:0]
(F0F) TKCTL2				**Function related to: Touch Key**
TKREF	0f.3	R/W	0	Touch Key channel reference capacitor select 0: Touch key channel select to TK0~TK15 1: Touch key channel select to Internal reference capacitor
TKREFC	0f.2~0	R/W	4	Reduce/Increase the value of the internal reference capacitor. 000: Smallest … 111: Biggest
(F10) PRES_SWH				**Function related to: Touch Key**
PRES_SWH	10.7~0	R/W	0	Place TK15~TK8 press result by user
(F11) PRES_HWL				**Function related to: Touch Key**
PRES_SWL	11.7~0	R/W	0	Place TK7~TK0 press result by user
(F12) PRES_HWH				**Function related to: Touch Key**
PRES_HWH	12.7~0	R	0	TK15~TK8 press result by built-in algorithm (ETK) 1: press, 0: no press
(F13) PRES_HWL				**Function related to: Touch Key**
PRES_HWL	13.7~0	R	0	TK7~TK0 press result by built-in algorithm (ETK) 1: press, 0: no press
(F14) TM1				**Function related to: Timer1**

Name	Address	R/W	Rst	Description
TM1	14.7~0	R/W	0	Timer1 content
(F15) TKCTL				**Function related to: Touch Key**
TKSTART	15.7	R/W	0	Set TKSTART to start ETK algorithm detect, HW will show the press result in PRES_HWH and PRES_HWL. Once user set TKSTART to start ETK, it can be stopped only if system reset. write 0: no action; write 1:set this flag
TKBUSYOE	15.6	R/W	0	Touch Key state information output enable (PA3)
TKTORSTE	15.5	R/W	0	30-second press timeout reset function enable.
TKEN	15.4	R/W	0	If TKEN=0, TK scan stop and keep in Wait state (TKBUSY=0), If TKEN=1, TK scan start.
TKIDX	15.3~0	R/W	0	Touch Key index. To indicate that which TK channel will be chosen when user Read TK Data, Read Baseline Data or Read/Write TKTMR.
(F16) I2CRCD0				**Function related to: Slave I2C**
I2CRCD0	16.7~0	R	0	The receiving register 0 of slave I2C
(F17) I2CRCD1				**Function related to: Slave I2C**
I2CRCD1	17.7~0	R	0	The receiving register 1 of slave I2C
(F18) I2CTXD0				**Function related to: Slave I2C**
I2CTXD0	18.7~0	R/W	0	The transmitting register 0 of slave I2C
(F19) I2CTXD1				**Function related to: Slave I2C**
I2CTXD1	19.7~0	R/W	0	The transmitting register 1 of slave I2C
(F1A) I2CCTL				**Function related to: Slave I2C**
I2CIE	1a.4	R/W	0	Slave I2C Interrupt Enable 0: Disable 1: Enable
I2CEN	1a.3	R/W	0	Slave I2C SDA output enable 0: disable 1: enable
-	1a.2	-	-	-
I2CID	1a.1~0	R/W	0	Slave I2C ID last 2 bits
(F1B) I2CFLG				**Function related to: Slave I2C**
I2CIF	1b.6	R	-	Slave I2C interrupt event pending flag This bit is set by H/W while a. I2CRCD0 or I2CRCD1 receive data finished b. I2CRCD0 or I2CRCD1 data overflow occurred c. I2CTXD0 or I2CTXD1 data transmit finished
		W	0	0: clear this flag 1: no action
TXD1F	1b.5	R	-	Slave I2C transmitting data register 1 flag, set by H/W while I2CTXD1 data transmitting finished
		W	0	0: clear this flag 1: no action
TXD0F	1b.4	R	-	Slave I2C transmitting data register 0 flag, set by H/W while I2CTXD0 data transmitting finished
		W	0	0: clear this flag 1: no action
RCD1OVF	1b.3	R	-	Slave I2C receiving data register 1 overflow flag, set by H/W while receiving data to I2CRCD1 overflow
		W	0	0: clear this flag 1: no action
RCD1F	1b.2	R	-	Slave I2C receiving data register 1 flag, set by H/W while data receiving

Name	Address	R/W	Rst	Description
				to I2CRCD1 finished
		W	0	0: clear this flag 1: no action
RCD0OVF	1b.1	R	-	Slave I2C receiving data register 0 overflow flag, set by H/W while receiving data to I2CRCD0 overflow
		W	0	0: clear this flag 1: no action
RCD0F	1b.0	R	-	Slave I2C receiving data register 0 flag, set by H/W while data receiving to I2CRCD0 finished
		W	0	0: clear this flag 1: no action
(F1C) RSR				**Function related to: RRAM R/W**
RSR	1c.7~0	R/W	0	R-Plane File Select Register
(F1D) DPL				**Function related to: Table Read**
DPL	1d.7~0	R/W	0	Table read low address, data ROM pointer (DPTR) low byte
(F1E) DPH				**Function related to: Table Read**
DPH	1e.7~0	R/W	0	Table read high address, data ROM pointer (DPTR) high byte
(F1F) IRCF				**Function relate to: Trim FIRC**
IRCF	1f.5~0	R/W	-	FIRC frequency adjustment:
User Data Memory				
FRAM	20~30	R/W	-	SRAM common area
	30~7f	R/W	-	SRAM Bank0 area
	30~7f	R/W	-	SRAM Bank1 area

R-Plane

Name	Address	R/W	Rst	Description
(R00) INDR				**Function related to: RRAM R/W**
INDR	00.7~0	R/W	-	Not a physical register, addressing INDR actually point to the register whose address is contained in the RSR register
(R01) TM0RLD				**Function related to: Timer0**
TM0RLD	01.7~0	R/W	0	Timer0 reload Data
(R02) TM0CTL				**Function related to: Timer0**
T0ISRC	02.6	R/W	0	Counter mode source select 0: T0CKI (PA2) 1: SRCLK/16
TM0EDG	02.5	R/W	0	Counter mode source edge, 0: rising edge, 1: falling edge
TM0CKS	02.4	R/W	0	Timer0 mode select 0 : Timer mode (Fsys/2) 1 : Counter mode (T0CKI or SRCLK/16)
TM0PSC	02.3~0	R/W	0	Timer0 prescaler. Timer0 clock source 0000: divided by 1 0001: divided by 2 0010: divided by 4 0011: divided by 8 0100: divided by 16 0101: divided by 32 0110: divided by 64 0111: divided by 128 1xxx: divided by 256
(R03) PWRDN				**Function related to: POWER DOWN**
PWRDN	03	W	-	Write this register to enter Power-down (IDLE) Mode
(R04) WDTCLR				**Function related to: WDT**
WDTCLR	04	W	-	Write this register to clear WDT timer
(R05) PAMODH				**Function related to: Port A**
PA7MOD	05.6	R/W	0	PA7 pull-up resistor enable 0: the pin pull-up resistor is enabled 1: the pin pull-up resistor is disabled
PA4MOD	05.1~0	R/W	01	PA4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3

Name	Address	R/W	Rst	Description
(R06) PAMODL				**Function related to: Port A**
PA3MOD	06.7~6	R/W	01	PA3 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA2MOD	06.5~4	R/W	01	PA2 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA1MOD	06.3~2	R/W	01	PA1 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA0MOD	06.1~0	R/W	01	PA0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
(R07) PBMODH				**Function related to: Port B**
PB7MOD	07.7~6	R/W	01	PB7 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB6MOD	07.5~4	R/W	01	PB6 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB5MOD	07.3~2	R/W	01	PB5 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB4MOD	07.1~0	R/W	01	PB4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
(R08) PBMODL				**Function related to: Port B**
PB3MOD	08.7~6	R/W	01	PB3 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB2MOD	08.5~4	R/W	01	PB2 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB1MOD	08.3~2	R/W	01	PB1 I/O mode control 00: Mode0

Name	Address	R/W	Rst	Description
				01: Mode1 10: Mode2 11: Mode3
PB0MOD	08.1~0	R/W	01	PB0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
(R0A) PDMODL				**Function related to: Port D**
PD3MOD	0a.7~6	R/W	01	PD3 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD2MOD	0a.5~4	R/W	01	PD2 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD1MOD	0a.3~2	R/W	01	PD1 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD0MOD	0a.1~0	R/W	01	PD0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
(R0B) MR0B				**Function related to: HWAUTO / WDT / WKT**
HWAUTO	0b.7	R/W	0	Save/Restore W register and STATUS register w/o TO, PD 0:disable 1:enable
WDTPSC	0b.3~2	R/W	11	WDT pre-scale option 00: 38.4ms 01: 76.8ms 10: 307.2ms 11:614.4ms
WKTPSC	0b.1~0	R/W	11	WKT pre-scale option 00: 19.2ms 01:38.4ms 10: 76.8ms 11: 153.6ms
(R0D) PWM0PRD				**Function related to: PWM0**
PWM0PRD	0d.7~0	R/W	FF	PWM0 period data
(R0E) PWM1PRD				**Function related to: PWM1**
PWM1PRD	0e.7~0	R/W	FF	PWM1 period data
(R0F) TSTREG				**Function related to: TEST**
TESTBIT0	0f. 0	W	0	If user wants to use TK I2C Data Write command, user has to set TESTBIT0=1 to make the source of TK index switch from SFR to I2C Data Write input. 0: SFR is the source of TKIDX value 1: TK I2C Data Write input is the source of TKIDX value
(R10) PWMCTL				**Function related to: PWM0 / PWM1**
PWM1CKS	10.7	R/W	0	PWM1 clock source

Name	Address	R/W	Rst	Description
				0: Fsys 1: FIRC (4MHz)
PWM1PSC	10.6~4	R/W	00	PWM1 prescaler 000: divided by 1 001: divided by 2 010: divided by 4 … 111: divided by 128
PWM0CKS	10.3	R/W	0	PWM0 clock source 0: Fsys 1: FIRC (4MHz)
PWM0PSC	10.2~0	R/W	00	PWM0 prescaler 000: divided by 1 001: divided by 2 010: divided by 4 … 111: divided by 128
(R11) PWM1AD				**Function related to: PWM1**
PWM1AD	11.7~0	R/W	00	PWM1A Duty
(R12) PWM1BD				**Function related to: PWM1**
PWM1BD	12.7~0	R/W	00	PWM1B Duty
(R13) PWM1CD				**Function related to: PWM1**
PWM1CD	13.7~0	R/W	00	PWM1C Duty
(R14) TM1CTL				**Function related to: Timer1**
TM1PSC	14.3~0	R/W	0	Timer1 prescaler. Timer1 clock source (Fsys/2) 0000: divided by 1 0001: divided by 2 0010: divided by 4 0011: divided by 8 0100: divided by 16 0101: divided by 32 0110: divided by 64 0111: divided by 128 1xxx: divided by 256
(R15) TM1RLD				**Function related to: Timer1**
TM1RLD	15.7~0	R/W	0	Timer1 reload Data
(R16) TKCHSEL				**Function related to: Touch Key**
LAST_CH	16.7~4	R/W	dh	Touch Key last scan channel TK scan range as follow : TK[FIRST_CH] ~TK[LAST_CH]
FIRST_CH	16.3~0	R/W	dh	Touch Key first scan channel. TK scan range as follow : TK[FIRST_CH] ~TK[LAST_CH]
(R17) TKDL				**Function related to: Touch Key**
TKDL	17.7~0	R	-	TKDATA [7:0]. Touch key data low byte of TK [TKIDX].
(R18) TKBLDH				**Function related to: Touch Key**
BSLINEDH	18.7~4	R	-	BSLINED [11:8]. Baseline data high byte of TK [TKIDX].
TKDH	18.3~0	R	-	TKDATA [11:8]. Touch key data high byte of TK [TKIDX].
(R19) BSLINEDL				**Function related to: Touch Key**
BSLINEDL	19.7~0	R	-	BSLINED [7:0]. Baseline data low byte of TK [TKIDX].
(R1A) TKCTL3				**Function related to: Touch Key**
TKBUSY	1a.5	R	-	Touch Key state information. 0: Wait state 1: Scan state
TKCOLD	1a.4	R	-	Touch Key Wait state mode information. 0: Wait state is at hot mode. 1: Wait state is at cold mode.
TKTMR	1a.2~0	R/W	2	Touch Key conversion time select. TKTMR of TK [TKIDX].

Name	Address	R/W	Rst	Description
				000: Conversion time shortest ... 111: Conversion time longest
(R1B) COLDWAIT				**Function related to: Touch Key**
COLDWAIT	1b.7~0	R/W	50h	TK cold mode wait time 0: 2.4ms 1: 2.4ms 2: 4.8ms ... ff: 612ms
(R1C) HOTWAIT				**Function related to: Touch Key**
HOTWAIT	1c.7~0	R/W	2ah	TK hot mode wait time 0: 1.2ms 1: 1.2ms 2: 2.4ms ... ff: 306ms
(R1D) TKRSTH				**Function related to: Touch Key**
TKRSTH	1d.7~0	R/W	0	TKRST [15:8]:TK15~TK8 baseline data reset.
(R1E) TKRSTL				**Function related to: Touch Key**
TKRSTL	1e.7~0	R/W	0	TKRST [7:0]:TK7~TK0 baseline data reset.

INSTRUCTION SET

Each instruction is a 14-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” or “r” 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	F-Plane Register File Address
r	R-Plane Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field. 0 : Working register 1 : Register file
TO	WDT Time Out Flag
PD	Power Down Flag
W	Working Register
Z	Zero Flag
C	Carry Flag
DC	Decimal Carry 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					
ADDWF	f,d	00 0111 dfff ffff	1	C, DC, Z	Add W and "f"
ANDWF	f,d	00 0101 dfff ffff	1	Z	AND W with "f"
CLRF	f	00 0001 1fff ffff	1	Z	Clear "f"
CLRW		00 0001 0100 0000	1	Z	Clear W
COMF	f,d	00 1001 dfff ffff	1	Z	Complement "f"
DECF	f,d	00 0011 dfff ffff	1	Z	Decrement "f"
DECFSZ	f,d	00 1011 dfff ffff	1 or 2	-	Decrement "f", skip if zero
INCF	f,d	00 1010 dfff ffff	1	Z	Increment "f"
INCFSZ	f,d	00 1111 dfff ffff	1 or 2	-	Increment "f", skip if zero
IORWF	f,d	00 0100 dfff ffff	1	Z	OR W with "f"
MOVFW	f	00 1000 0fff ffff	1	-	Move "f" to W
MOVRW	r	01 1111 00rr rrrr	1	-	Move "r" to W
MOVWF	f	00 0000 1fff ffff	1	-	Move W to "f"
MOVWR	r	01 1110 00rr rrrr	1	-	Move W to "r"
RLF	f,d	00 1101 dfff ffff	1	C	Rotate left "f" through carry
RRF	f,d	00 1100 dfff ffff	1	C	Rotate right "f" through carry
SUBWF	f,d	00 0010 dfff ffff	1	C, DC, Z	Subtract W from "f"
SWAPF	f,d	00 1110 dfff ffff	1	-	Swap nibbles in "f"
TESTZ	f	00 1000 1fff ffff	1	Z	Test if "f" is zero
XORWF	f,d	00 0110 dfff ffff	1	Z	XOR W with "f"
Bit-Oriented File Register Instruction					
BCF	f,b	01 000b bbff ffff	1	-	Clear "b" bit of "f"
BSF	f,b	01 001b bbff ffff	1	-	Set "b" bit of "f"
BTFSC	f,b	01 010b bbff ffff	1 or 2	-	Test "b" bit of "f", skip if clear
BTFSS	f,b	01 011b bbff ffff	1 or 2	-	Test "b" bit of "f", skip if set
Literal and Control Instruction					
ADDLW	k	01 1100 kkkk kkkk	1	C, DC, Z	Add Literal "k" and W
ANDLW	k	01 1011 kkkk kkkk	1	Z	AND Literal "k" with W
CALL	k	10 kkkk kkkk kkkk	2	-	Call subroutine "k"
CLRWDT		01 1110 0000 0100	1	TO, PD	Clear Watch Dog Timer
GOTO	k	11 kkkk kkkk kkkk	2	-	Jump to branch "k"
IORLW	k	01 1010 kkkk kkkk	1	Z	OR Literal "k" with W
MOVLW	k	01 1001 kkkk kkkk	1	-	Move Literal "k" to W
NOP		00 0000 0000 0000	1	-	No operation
RET		00 0000 0100 0000	2	-	Return from subroutine
RETI		00 0000 0110 0000	2	-	Return from interrupt
RETLW	k	01 1000 kkkk kkkk	2	-	Return with Literal in W
TABRL		00 0000 0101 0000	2	-	Lookup ROM low data to W
TABRH		00 0000 0101 1000	2	-	Lookup ROM high data to W
SLEEP		01 1110 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
XORLW	k	01 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	01 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

ADDWF — Add W and "f"

Syntax	ADDWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (W) + (f)
Status Affected	C, DC, Z
OP-Code	00 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	ADDWF 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	01 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

ANDWF — AND W with "f"

Syntax	ANDWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (W) AND (f)
Status Affected	Z
OP-Code	00 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	ANDWF FSR, 1 — B : W = 0x17, FSR = 0xC2 A : W = 0x17, FSR = 0x02

BCF — Clear "b" bit of "f"

Syntax	BCF f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	01 000b bbff ffff	
Description	Bit 'b' in register 'f' is cleared.	
Cycle	1	
Example	BCF FLAG_REG, 7	B : FLAG_REG = 0xC7 A : FLAG_REG = 0x47

BSF — Set "b" bit of "f"

Syntax	BSF f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	01 001b bbff ffff	
Description	Bit 'b' in register 'f' is set.	
Cycle	1	
Example	BSF FLAG_REG, 7	B : FLAG_REG = 0x0A A : FLAG_REG = 0x8A

BTFSC — Test "b" bit of "f", skip if clear(0)

Syntax	BTFSC f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 0	
Status Affected	-	
OP-Code	01 010b bbff 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 BTFSC FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = FALSE if FLAG.1 = 1, PC = TRUE

BTFSS — Test "b" bit of "f", skip if set(1)

Syntax	BTFSS f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 1	
Status Affected	-	
OP-Code	01 011b bbff 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 BTFSS FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = TRUE if FLAG.1 = 1, PC = FALSE

CALL — Call subroutine "k"

Syntax	CALL k
Operands	k : 000h ~ FFFh
Operation	Operation: TOS ← (PC) + 1, PC.11~0 ← k
Status Affected	-
OP-Code	10 kkkk kkkk kkkk
Description	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 12-bit immediate address is loaded into PC bits <11:0>. CALL is a two-cycle instruction.
Cycle	2
Example	LABEL1 CALL SUB1 — B : PC = LABEL1 A : PC = SUB1, TOS = LABEL1 + 1

CLRF — Clear "f"

Syntax	CLRF f
Operands	f : 00h ~ 7Fh
Operation	(f) ← 00h, Z ← 1
Status Affected	Z
OP-Code	00 0001 1fff ffff
Description	The contents of register 'f' are cleared and the Z bit is set.
Cycle	1
Example	CLRF 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	00 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/WKT Timer ← 00h
Status Affected	TO, PD
OP-Code	01 1110 0000 0100
Description	CLRWDT instruction clears the Watchdog/Wakeup Timer
Cycle	1
Example	CLRWDT — B : WDT counter = ? A : WDT counter = 0x00

COMF — Complement "f"

Syntax	COMF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← ($\bar{f}$)	
Status Affected	Z	
OP-Code	00 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	COMF REG1, 0	B : REG1 = 0x13 A : REG1 = 0x13, W = 0xEC

DECF — Decrement "f"

Syntax	DECF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (f) - 1	
Status Affected	Z	
OP-Code	00 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	DECF CNT, 1	B : CNT = 0x01, Z = 0 A : CNT = 0x00, Z = 1

DECFSZ — Decrement "f", Skip if 0

Syntax	DECFSZ f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (f) - 1, skip next instruction if result is 0	
Status Affected	-	
OP-Code	00 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 DECFSZ CNT, 1 GOTO LOOP CONTINUE	B : PC = LABEL1 A : CNT = CNT – 1 if CNT = 0, PC = CONTINUE if CNT ≠ 0, PC = LABEL1 + 1

GOTO — Unconditional Branch

Syntax	GOTO k	
Operands	k : 000h ~ FFFh	
Operation	PC.11~0 ← k	
Status Affected	-	
OP-Code	11 kkkk kkkk kkkk	
Description	GOTO is an unconditional branch. The 12-bit immediate value is loaded into PC bits <11:0>. GOTO is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 GOTO SUB1	B : PC = LABEL1 A : PC = SUB1

INCF — Increment "f"

Syntax	INCF f [,d]
Operands	f : 00h ~ 7Fh
Operation	(destination) ← (f) + 1
Status Affected	Z
OP-Code	00 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	INCF CNT, 1 — B : CNT = 0xFF, Z = 0 A : CNT = 0x00, Z = 1

INCFSZ — Increment "f", Skip if 0

Syntax	INCFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (f) + 1, skip next instruction if result is 0
Status Affected	-
OP-Code	00 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 INCFSZ CNT, 1 GOTO 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	01 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

IORWF — Inclusive OR W with "f"

Syntax	IORWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (W) OR k
Status Affected	Z
OP-Code	00 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	IORWF RESULT, 0 — B : RESULT = 0x13, W = 0x91 A : RESULT = 0x13, W = 0x93, Z = 0

MOVFW — Move "f" to W

Syntax	MOVFW f	
Operands	f : 00h ~ 7Fh	
Operation	(W) ← (f)	
Status Affected	-	
OP-Code	00 1000 0fff ffff	
Description	The contents of register 'f' are moved to W register.	
Cycle	1	
Example	MOVFW 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	01 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

MOVRW — Move "r" to W

Syntax	MOVRW r	
Operands	r : 00h ~ 3Fh	
Operation	(W) ← (r)	
Status Affected	-	
OP-Code	01 1111 00rr rrrr	
Description	The contents of register 'r' are moved to W register.	
Cycle	1	
Example	MOVRW MR0B	B : MR0B = 0x0F, W = ? A : MR0B = 0x0F, W = 0x0F

MOVWF — Move W to "f"

Syntax	MOVWF f	
Operands	f : 00h ~ 7Fh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWF REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

MOVWR	**Move W to "r"**	
Syntax	MOVWR r	
Operands	r : 00h ~ 3Fh	
Operation	(r) ← (W)	
Status Affected	-	
OP-Code	01 1110 00rr rrrr	
Description	Move data from W register to register 'r'.	
Cycle	1	
Example	MOVWR REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

NOP	**No Operation**	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	00 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

RET	**Return from Subroutine**	
Syntax	RET	
Operands	-	
Operation	PC ← TOS	
Status Affected	-	
OP-Code	00 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	00 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	01 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	

```
CALL  TABLE                  B : W = 0x07
         :                   A : W = value of k8
TABLE  ADDWF  PCL, 1
       RETLW  k1
       RETLW  k2
         :
       RETLW  kn
```

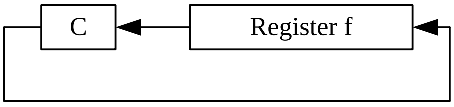
TABRL — Return DPTR low byte to W

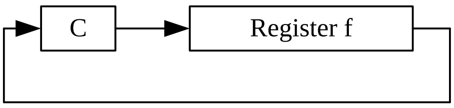
Syntax	TABRL
Operands	-
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR={DPH[max:8],DPL[7:0]}
Status Affected	-
OP-Code	00 0000 0101 0000
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	

```
:
:
MOVLW    (TAB1&0xFF)
MOVWF    DPL                ; Where DPL is F-plane register
MOVLW    (TAB1>>8)&0xFF
MOVWF    DPH                ; Where DPH is F-plane register

TABRL                       ; W=0x89
TABRH                       ; W=0x37


ORG      0234H
TAB1:
.DT        0x3789, 0x2277   ;ROM data 14 bits
```

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	00 0000 0101 1000
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2

RLF	**Rotate Left "f" through Carry**
Syntax	RLF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 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	RLF REG1, 0 — B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 1100 1100, C = 1

RRF	**Rotate Right "f" through Carry**
Syntax	RRF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 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	RRF 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	01 1110 0000 0011
Description	Go into Power-down mode with the oscillator stops.
Cycle	1
Example	SLEEP — -

SUBWF — Subtract W from "f"

Syntax	SUBWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (f) – (W)
Status Affected	C, DC, Z
OP-Code	00 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	SUBWF REG1, 1 — B : REG1 = 0x03, W = 0x02, C = ?, Z = ? A : REG1 = 0x01, W = 0x02, C = 1, Z = 0
	SUBWF REG1, 1 — B : REG1 = 0x02, W = 0x02, C = ?, Z = ? A : REG1 = 0x00, W = 0x02, C = 1, Z = 1
	SUBWF REG1, 1 — B : REG1 = 0x01, W = 0x02, C = ?, Z = ? A : REG1 = 0xFF, W = 0x02, C = 0, Z = 0

SWAPF — Swap Nibbles in "f"

Syntax	SWAPF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination,7~4) ← (f.3~0), (destination.3~0) ← (f.7~4)
Status Affected	-
OP-Code	00 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	SWAPF REG, 0 — B : REG1 = 0xA5 A : REG1 = 0xA5, W = 0x5A

TESTZ — Test if "f" is zero

Syntax	TESTZ f
Operands	f : 00h ~ 7Fh
Operation	Set Z flag if (f) is 0
Status Affected	Z
OP-Code	00 1000 1fff ffff
Description	If the content of register 'f' is 0, Zero flag is set to 1.
Cycle	1
Example	TESTZ 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	01 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

XORWF	Exclusive OR W with "f"	
Syntax	XORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	00 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	XORWF 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} + 3.6	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	3.6	V
Operating temperature	-40 to +85	°C
Storage temperature	-65 to +150	

2. DC Characteristics (T_A = 25°C, V_{CC} = 1.3V to 3.6V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{CC}	FAST mode, 25°C, Fsys = 4 MHz		1.8	–	3.6	V
		FAST mode, 25°C, Fsys = 2 MHz		1.8	–	3.6	
		SLOW mode, 25°C, Fsys = 27 KHz		1.8	–	3.6	
Input High Voltage	V_{IH}	All Input, except PA7	V_{CC} = 3V	0.6V_{CC}	–	–	V
		PA7		0.7V_{CC}	–	–	
Input Low Voltage	V_{IL}	All Input	V_{CC} = 3V	–	–	0.2V_{CC}	V
I/O Port Source Current	I_{OH}	All Output	V_{CC} = 3V, V_{OH} = 0.9V_{CC}	2	4	–	mA
I/O Port Sink Current	I_{OL}	All Output	V_{CC} = 3V, V_{OL} = 0.1V_{CC}	5	10	–	mA
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}	–	–	1	μA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0V	–	–	-1	
Supply Current	I_{DD}	FAST mode	V_{CC} = 3V, FIRC = 4 MHz	–	0.7	–	mA
			V_{CC} = 3V, FIRC = 2 MHz	–	0.4	–	
			V_{CC} = 3V, FIRC = 1 MHz	–	0.3	–	
			V_{CC} = 3V, FIRC = 0.25 MHz	–	0.2	–	
		SLOW mode,	V_{CC} = 3V, SIRC = 27 KHz	–	6	–	μA
		IDLE mode,	V_{CC} = 3V, SIRC = 27 KHz	–	1.6	–	

System Clock Frequency	Fsys	V_{CC} = 3.6V @25℃	–	–	4.1	MHz
		V_{CC} = 2.5V @25℃	–	–	3.9	
LVR Reference Voltage	V_{LVR}	T_A = 25°C	–	1.8	–	V
LVR Hysteresis Voltage	V_{HYST}	T_A = 25°C	–	±0.1	–	V
Low Voltage Detection time	t_{LVR}	T_A = 25°C	100	–	–	μs
Pull-Up Resistor	R_P	V_{IN} = 0 V, V_{CC} = 3V, All Pins except PA7	–	230	–	KΩ
		V_{IN} = 0 V, V_{CC} = 3V, PA7	–	230	–	

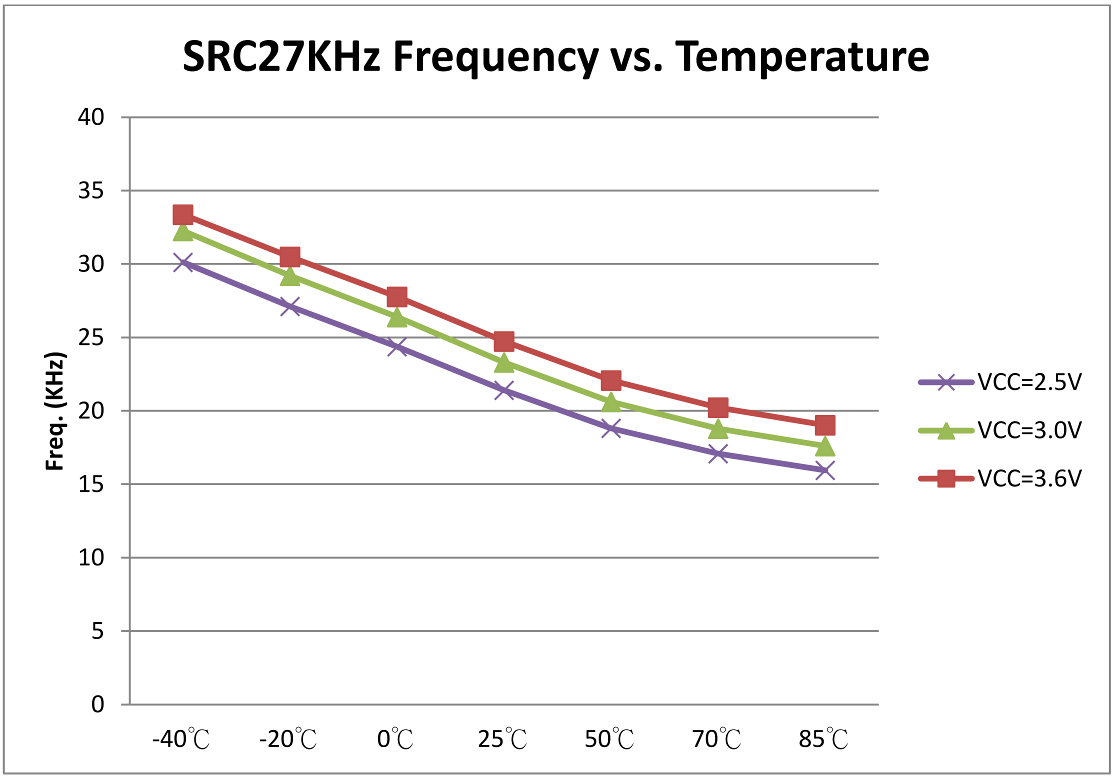
3. Clock Timing (T_A = -40°C to +85°C)

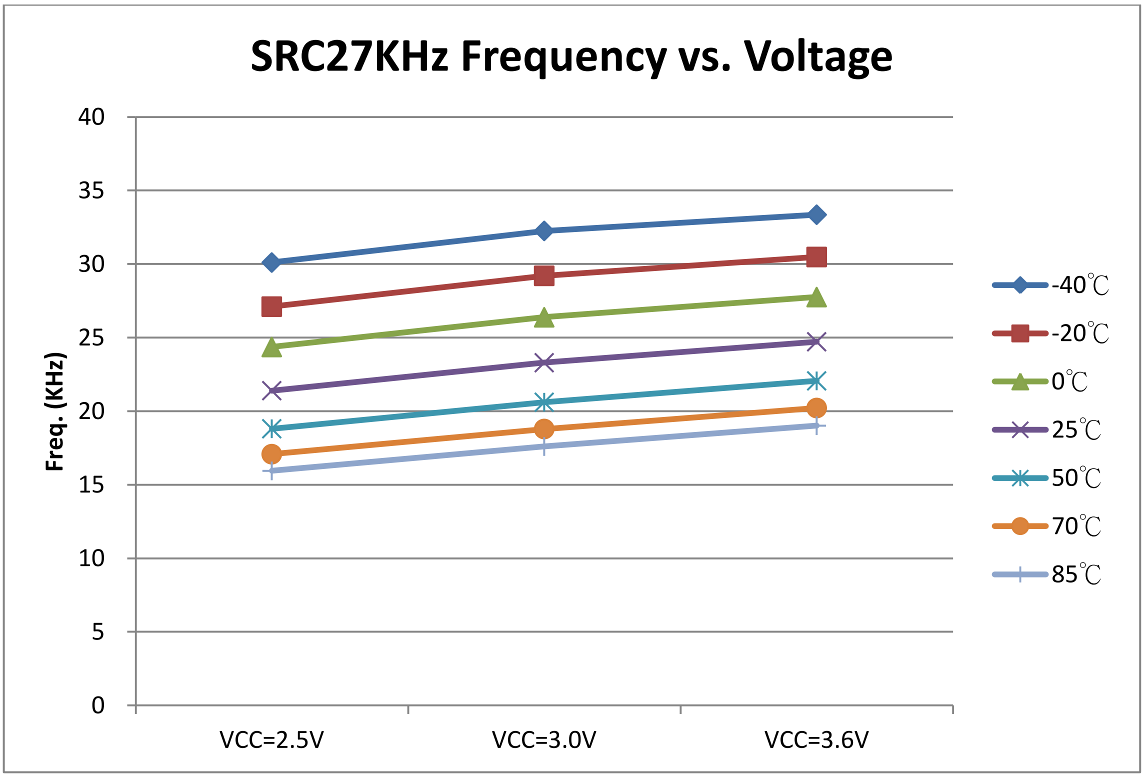
Parameter	Condition	Min	Typ	Max	Unit
Fast Internal RC Frequency	25℃, V_{CC} = 2.5 ~ 3.6V	3.9	–	4.1	MHz
	-40℃ ~ 85℃, V_{CC} = 2.5 ~ 3.6V	3.7	–	4.1	

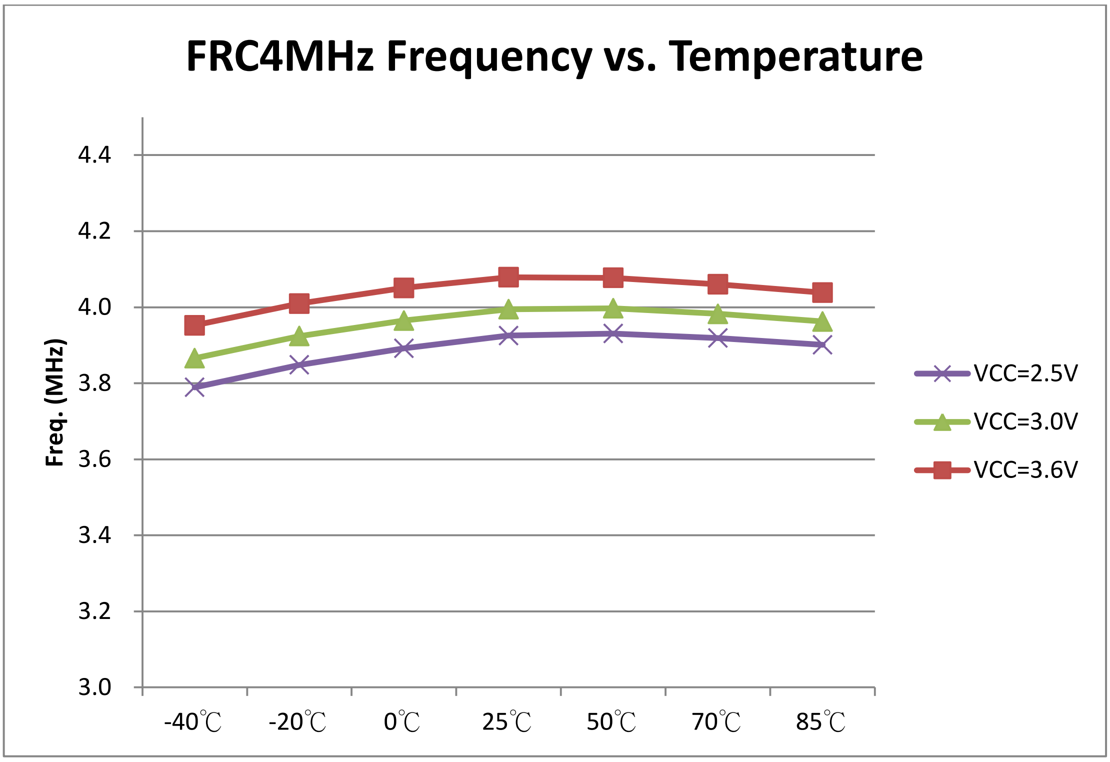
4. Reset Timing Characteristics (T_A = -40°C to +85°C, V_{CC} = 3V)

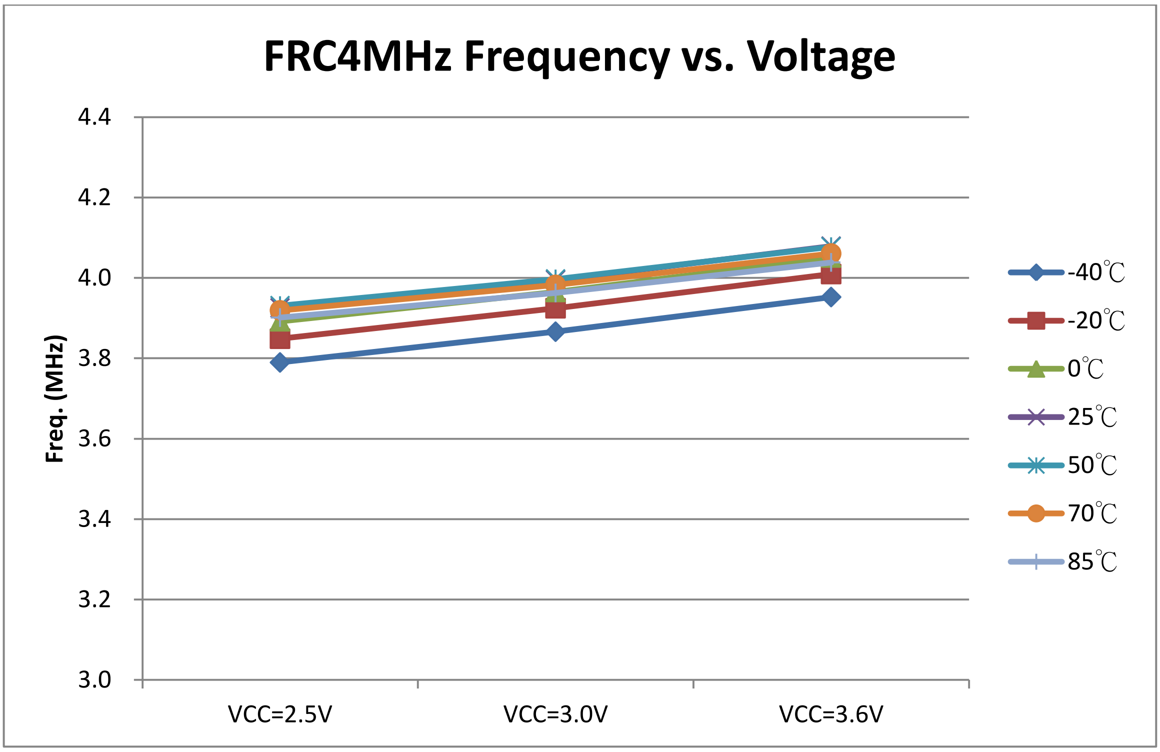
Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input V_{CC} = 3 V ±10 %	3	–	–	μs
WKT time	V_{CC} = 3V, WKTPSC = 3	–	153.6	–	ms
WDT time	V_{CC} = 3V, WDTPSC = 3	–	614.4	–	ms
CPU start up time	V_{CC} = 3V	–	35	–	ms

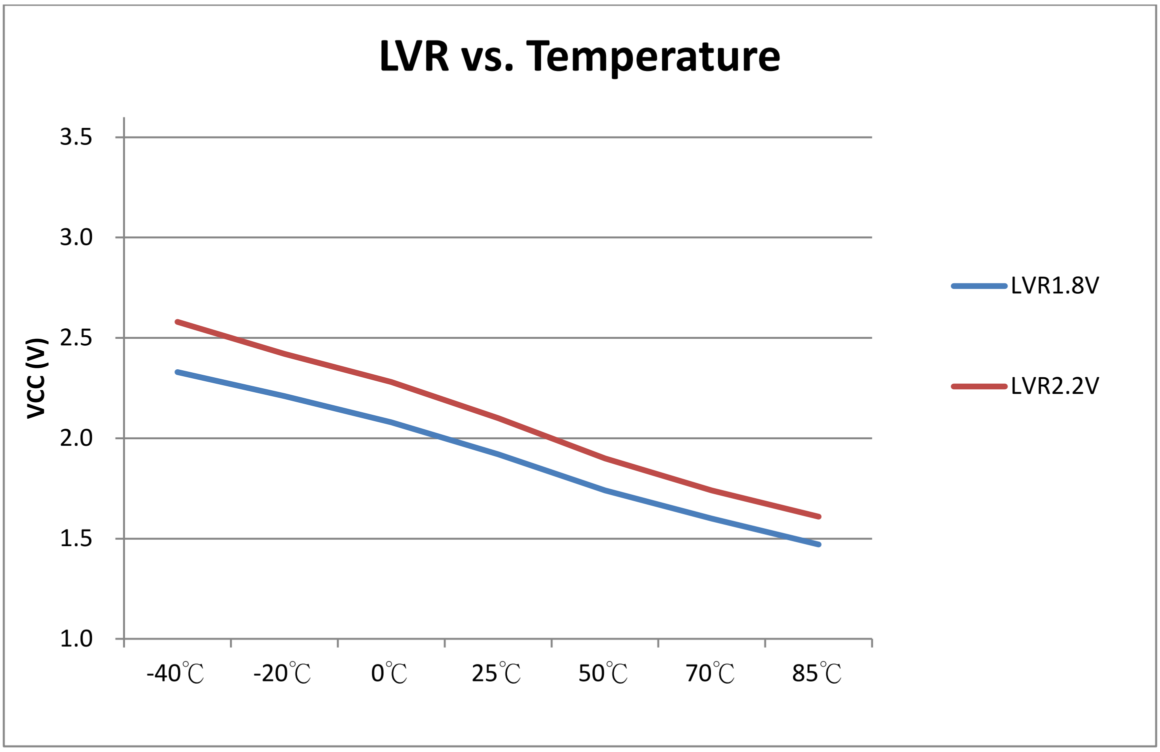
5. Characteristic Graphs



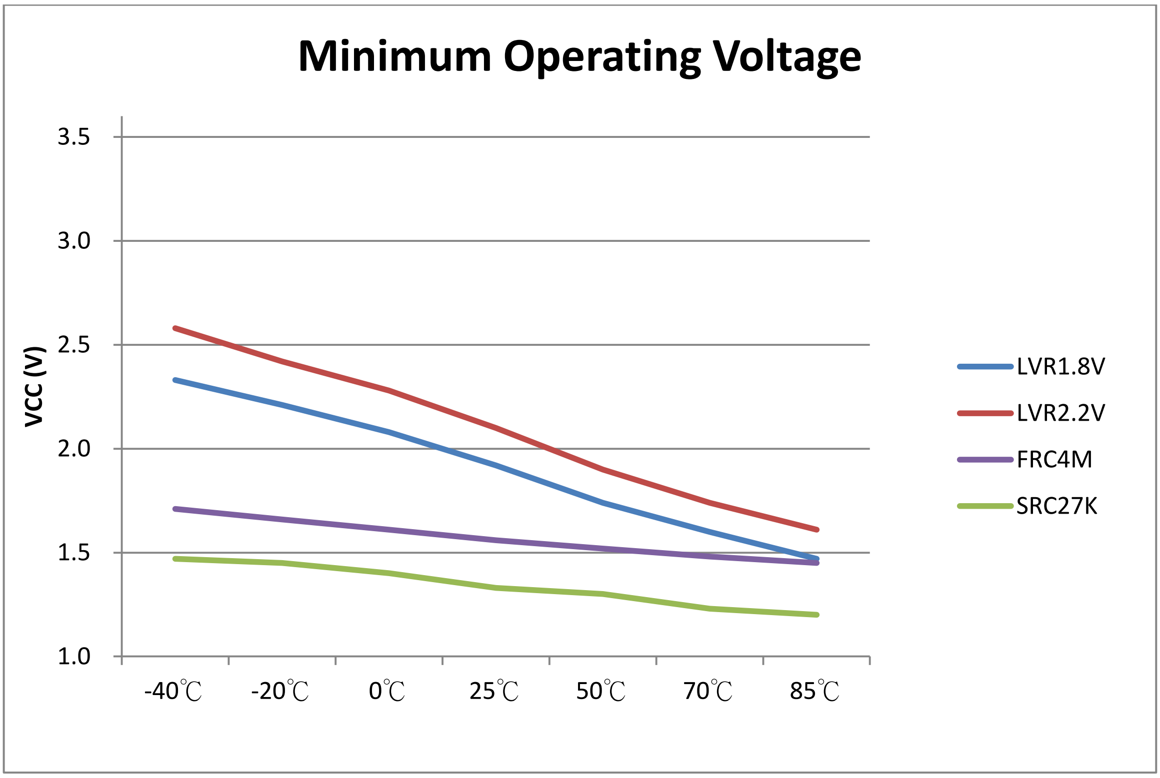



FRC4MHz Frequency vs. Temperature
Freq. (MHz)
4.4
4.2
4.0
3.8
3.6
3.4
3.2
3.0
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
VCC=2.5V
VCC=3.0V
VCC=3.6V


FRC4MHz Frequency vs. Voltage
Freq. (MHz)
4.4
4.2
4.0
3.8
3.6
3.4
3.2
3.0
VCC=2.5V
VCC=3.0V
VCC=3.6V
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃



Note. Due to the variation of manufacturing process, this LVR will slightly vary between different chips.



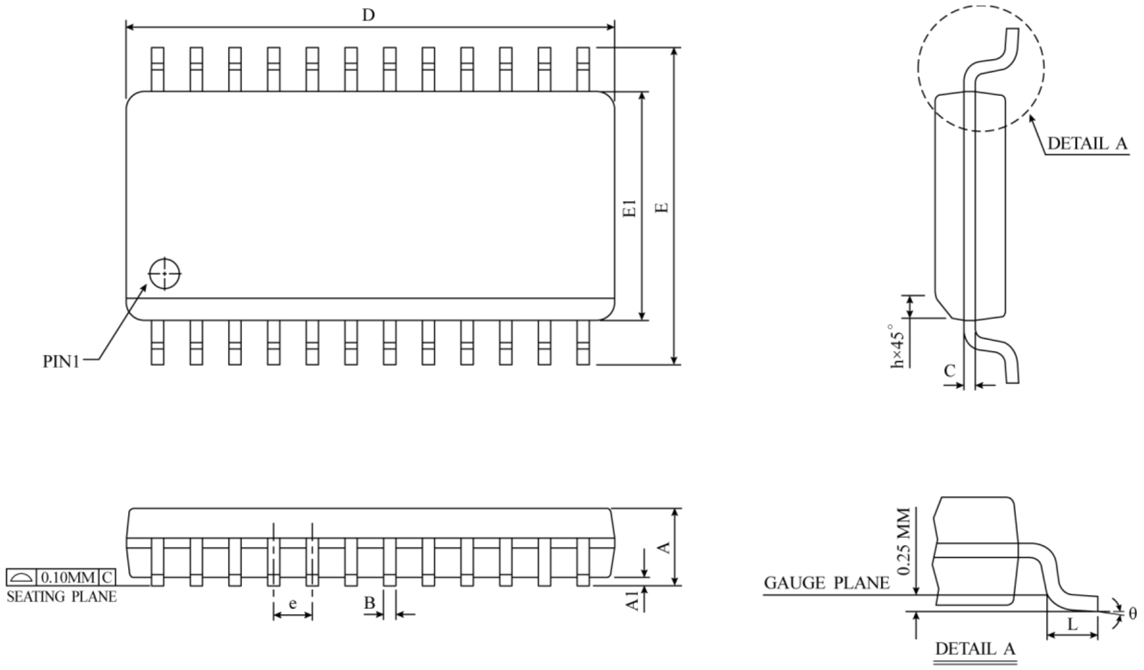
Note. Due to the variation of manufacturing process, this LVR will slightly vary between different chips.

PACKAGING INFORMATION

The ordering information:

Ordering number	Package
TM57ME28-MTP	Wafer / Dice blank chip
TM57ME28-COD	Wafer / Dice with code
TM57ME28-MTP-22	SOP 24 pin (300mil)
TM57ME28-MTP-21	SOP 20-pin (300 mil)
TM57ME28-MTP-16	SOP 16-pin (150 mil)
TM57ME28-MTP-97	QFN 20-pin(4*4*0.75-0.5mm)

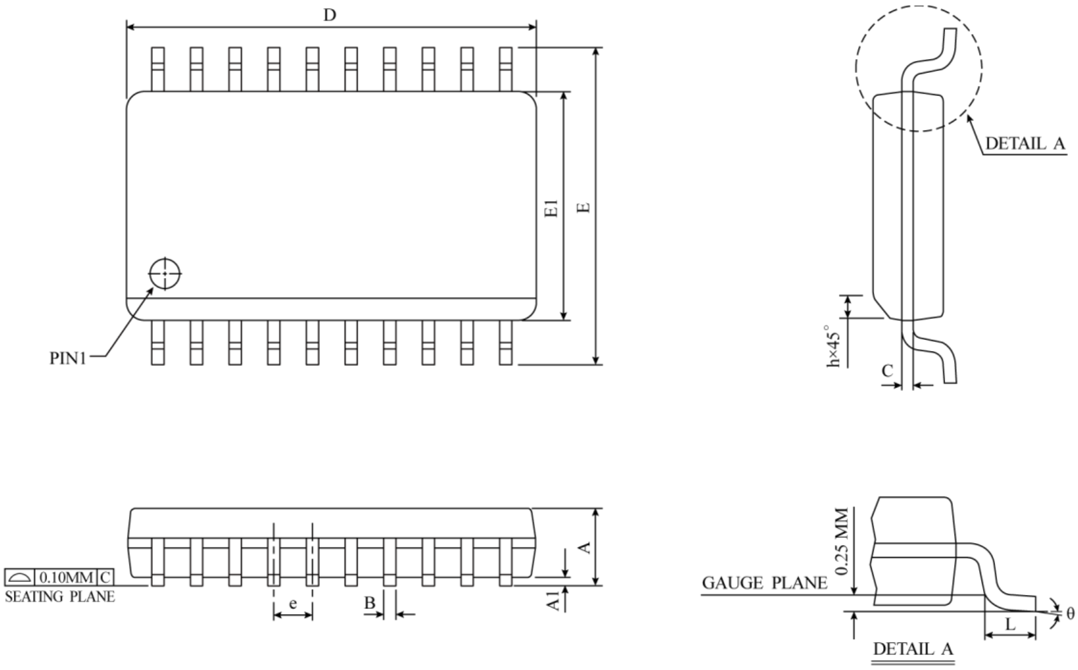
SOP-24 (300mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	2.35	2.50	2.65	0.0926	0.0985	0.1043
A1	0.10	0.20	0.30	0.0040	0.0079	0.0118
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.23	0.28	0.32	0.0091	0.0108	0.0125
D	15.20	15.40	15.60	0.5985	0.6063	0.6141
E	10.00	10.33	10.65	0.3940	0.4425	0.4910
E1	7.40	7.50	7.60	0.2914	0.2953	0.2992
e	1.27 BSC			0.050 BSC		
h	0.25	0.50	0.75	0.0100	0.0195	0.0290
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-013 (AD)					

A * 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.

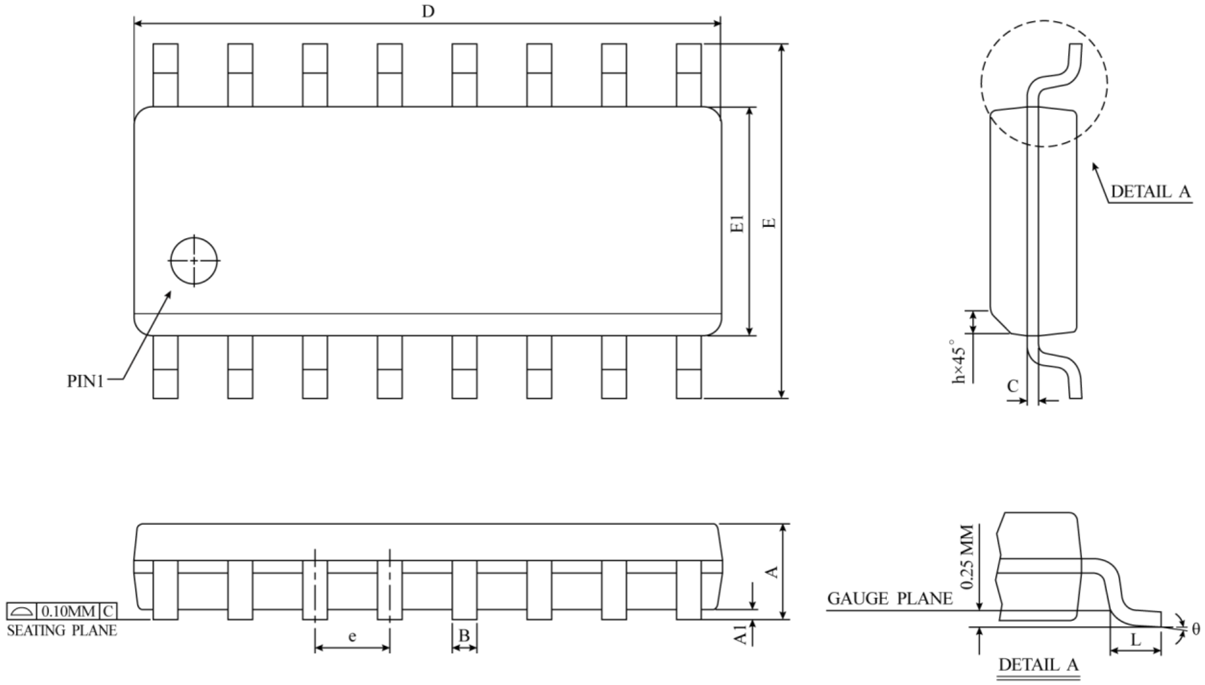
SOP-20 (300mil) Package Dimension



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

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

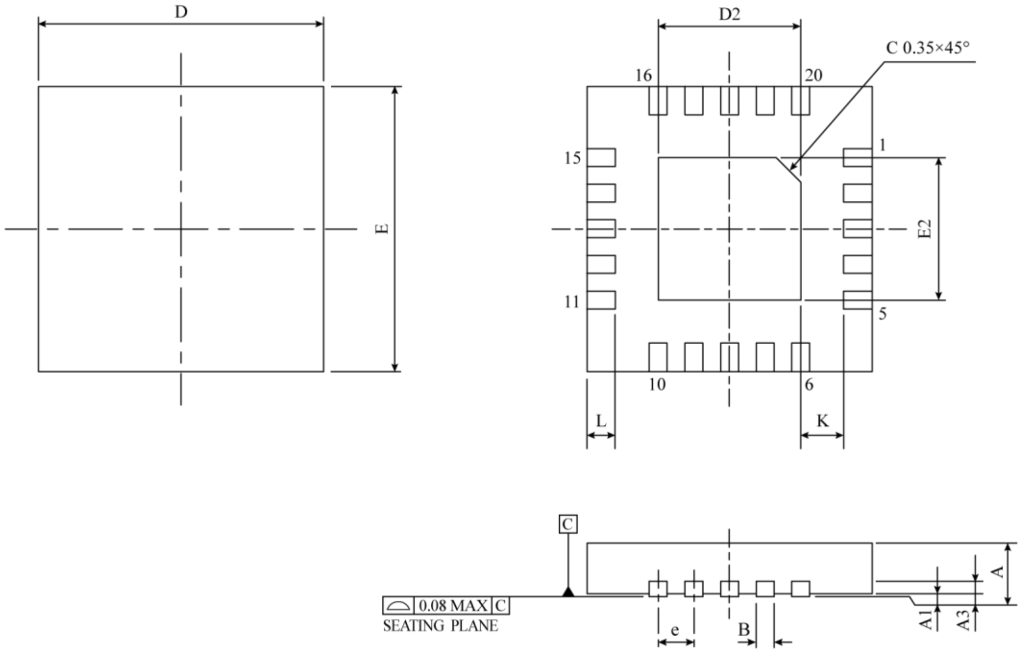
SOP-16 (150mil) Package Dimension



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

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

QFN 20 (4*4*0.75-0.5mm) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.70	0.75	0.80	0.028	0.030	0.031
A1	0.00	0.03	0.05	0.000	0.001	0.002
A3	0.20 REF.			0.008 REF.		
B	0.20	0.25	0.30	0.008	0.001	0.012
D	4.00 BSC			0.157 BSC		
E	4.00 BSC			0.157 BSC		
e	0.50 BSC			0.020 BSC		
K	0.20	-	-	0.008	-	-
E2	2.40	2.48	2.55	0.094	0.097	0.100
D2	2.40	2.48	2.55	0.094	0.097	0.100
L	0.35	0.45	0.55	0.014	0.018	0.022
JEDEC	W(V) GGD-11					

⚠ * NOTES : DIMENSION B APPLIES TO METALLIZED TERMINAL AND IS MEASURED BETWEEN 0.15mm AND 0.30mm FROM THE TERMINAL TIP. IF THE TERMINAL HAS THE TERMINAL, THE DIMENSION B SHOULD NOT BE MEASURED IN THAT RADIUS AREA.
BILATERAL COPLANARITY ZONE APPLIES TO THE EXPOSED HEAT SINK SLUG AS WELL AS THE TERMINALS.