十速

# TM57MT21

## *DATA SHEET*

## *Rev 0.94*

# AMENDMENT HISTORY

| Version | Date | Description |
|---|---|---|
| V0.90 | May, 2015 | New release. |
| V0.91 | Feb, 2016 | 1. PIN SUMMARY error correction<br>2. Add ATK control description<br>3. Add pin mode control related SFR<br>4. Others error correction |
| V0.92 | Jan, 2017 | 1. Modify operate voltage condition<br>2. Modify operate frequency condition<br>3. Modify Graphs: LVR vs. Temp.<br>4. Others error correction |
| V0.93 | Nov, 2017 | 1. Modify operate voltage condition |
| V0.94 | Mar, 2018 | 1. Add MSOP-8, MSOP10, QFN-20 package type |

# CONTENTS

**AMENDMENT HISTORY** ..... **2**
**FEATURES** ..... **5**
**BLOCK DIAGRAM** ..... **8**
**PIN ASSIGNMENT** ..... **9**
**PIN DESCRIPTION** ..... **11**
**PIN SUMMARY** ..... **12**
**FUNCTION DESCRIPTION** ..... **13**
**1. CPU Core** ..... **13**
1.1 Clock Scheme and Instruction Cycle ..... 13
1.2 Addressing Mode ..... 14
1.3 Programming Counter (PC) and Stack ..... 16
1.4 ALU and Working (W) Register ..... 17
1.5 STATUS Register (F-Plane 03H) ..... 18
1.6 Interrupt ..... 19
**2. Chip Operation Mode** ..... **23**
2.1 Reset ..... 23
2.2 System Configuration Register (SYSCFG) ..... 23
2.3 MTP Program ROM ..... 24
2.4 Power-Down Mode ..... 24
2.5 Dual System Clock ..... 25
2.6 Dual System Clock Modes Transition ..... 26
**3. Peripheral Functional Block** ..... **29**
3.1 Watchdog Timer (WDT) /Wakeup Timer (WKT) ..... 29
3.2 Timer0: 8-bit Timer/Counter with Pre-scale (PSC) ..... 31
3.3 Timer1: 8-bit Timer with Pre-scale (PSC) ..... 36
3.4 PWM0 ..... 39
3.5 Touch Key ..... 43
3.6 S/W controlled LCD Driver ..... 55
3.7 Specific Purpose Slave I2C Interface ..... 60
3.8 System Clock Oscillator ..... 65
**4. I/O Port** ..... **66**
4.1 PA0-6, PB0-7, PD7 ..... 66
4.2 PA7 ..... 75
**MEMORY MAP** ..... **76**
**F-Plane** ..... **76**
**R-Plane** ..... **81**

**Instruction Set** ..... **85**
**Electrical Characteristics** ..... **97**
**1. Absolute Maximum Ratings** ..... **97**
**2. DC Characteristics** ..... **97**
**3. Clock Timing** ..... **98**
**4. Reset Timing Characteristics** ..... **98**
**5. Characteristic Graphs** ..... **99**
**Packaging information** ..... **103**
**8-MSOP Package Dimension** ..... **104**
**10-MSOP Package Dimension** ..... **105**
**16-DIP Package Dimension** ..... **106**
**16-SOP Package Dimension** ..... **107**
**20-DIP Package Dimension** ..... **108**
**20-SOP Package Dimension** ..... **109**
**20-QFN Package Dimension** ..... **110**

# FEATURES

1. MTP: 2K x 14 bits MTP ROM (Support ISP uses 5 wires)
2. RAM: 184 x 8 bits
3. STACK: 6 Levels
4. I/O Ports: Three bit-programmable I/O ports (Max. 17 pins)
5. Two Independent Timers
   - Timer0
     - 8-bit Timer0 with divided by 1~256 pre-scale option/auto-reload/counter/interrupt/stop function
   - Timer1
     - 8-bit Timer1 with divided by 1~256 pre-scale option/auto-reload/interrupt stop function
6. One 8-bit PWM0 with pre-scale/period-adjustment/clear and hold function/positive-negative-output function
7. 14-channel Touch Key
   - 1~8 Key H/W auto scan, upper / lower boundary adjustable for each key
   - Interrupt / Wake-up CPU while key is pressed
8. Specific purpose slave I2C interface with interrupt function
9. Software controlled COM0~3 LCD driver with 1/2 bias
10. PA0~PA6, PB6~PB7, PD7 individual pin low level wake up
11. System Oscillation Sources (Fsys)
    - Fast-clock
      - FIRC (Fast Internal RC) :1/2/4 MHz
    - Slow-clock
      - SIRC (Slow Internal RC) : 2.1/4.2/8.5/17 KHz@$V_{DD}$=3V
12. Power Saving Operation Modes
    - FAST Mode: Fast-clock keeps CPU running
    - SLOW Mode: Fast-clock stops, Slow-clock keeps CPU running
    - IDLE Mode: Fast-clock and CPU stop, Wake-up Timer or Auto Touch Key keep running
    - STOP Mode: All Clocks Stop, Wake-up Timer and Auto Touch Key Stop
13. Dual System Clock
    - FIRC + SIRC

**14.** Reset Sources
- Power On Reset
- Watchdog Reset
- Low Voltage Reset
- External pin Reset

**15.** 1-Level Low Voltage Reset: 1.9V (can be disabled)

**16.** Operation Voltage: Low Voltage Reset Level to 4.0V
- Fsys=2 MHz, 1.3V ~ 4.0V
- Fsys=4 MHz, 1.5V ~ 4.0V

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

**18.** Interrupts
- Three External Interrupt pins
  - Two pin is falling edge triggered
  - One pin is rising or falling edge triggered
- Timer0 / Timer1/Wake-up Timer Interrupt
- I2C Interrupt
- Auto Touch Key Interrupt

**19.** Watchdog Timer (WDT) /Wake-up Timer (WKT)
- Clocked by built-in RC oscillator with 4 adjustable Reset/Interrupt time options
  - $V_{DD}$=3V, WDT/WKT=240ms/120ms/60ms/30ms
- Watchdog timer can be disabled/enabled in Power-down mode

**20.** I/O Port Modes
- Open-Drain Output
- CMOS Push-Pull Output
- Schmitt Trigger Input with pull-up resistor option

**21.** Instruction set: 36 Instructions

**22.** Package Types:
- 20-pin DIP (300 mil)
- 20-pin SOP (300 mil)
- 20-pin QFN (4x4x0.75-0.5mm)
- 16-pin DIP (300 mil)
- 16-pin SOP (150 mil)
- 10-pin MSOP (118 mil)
- 8-pin MSOP (118 mil)

**23.** Supported EV board on ICE

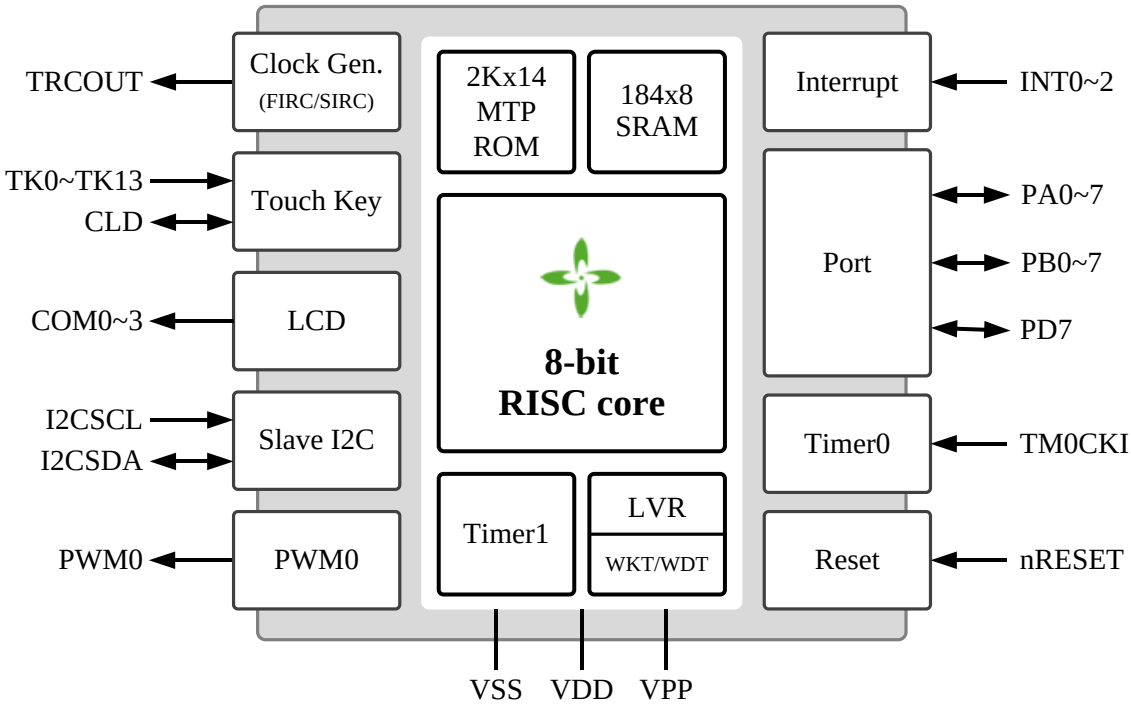
- EV board: EV8206

**24. Comparison Table:**

| | | EV2806 | TM57MT21 | TM57MT21A |
|---|---|---|---|---|
| **EV Board** | | – | EV2806 | EV2806 |
| **LVR** | | – | 1.9V<br>(can be disabled) | 1.9V / 2.2V<br>(can be disabled) |
| **Touch Key (S/W mode)** | **Channel** | 14 key | 14 key | 14 key |
| | **Reference Capacitor** | O<br>(TKCHS=15) | O<br>(TKCHS=15) | O<br>(TKCHS=15) |
| | **Interrupt** | X | X | O<br>(triggered when TKEOC=1) |
| **Touch Key (H/W mode)** | **Channel** | Scan 1~8 key | Scan 1~8 key | Scan 1~8 key |
| | **Scan** | multiple 1 / 2 / 4 time(s) | multiple 1 / 2 / 4 time(s) | multiple 1 / 2 / 4 time(s) |
| | **Interrupt** | O<br>(triggered after all keys scanned and pressed) | O<br>(triggered after all keys scanned and pressed) | O<br>(triggered when any key pressed) |
| | **ATKPOL**[1] | X | X | O |
| | **ATKSIT**[2] | X<br>(share with WKTPSC) | X<br>(share with WKTPSC) | O<br>(independent SFR) |
| | **ATKDT**[3] | 8 bits Binary<br>(ex. ATKDT=00001000b) | 8 bits Binary<br>(ex. ATKDT=00001000b) | 3 bits BCD<br>(ex. ATKDT=011b) |
| | **TKDH/DL**[4] **readable** | X | X | O |

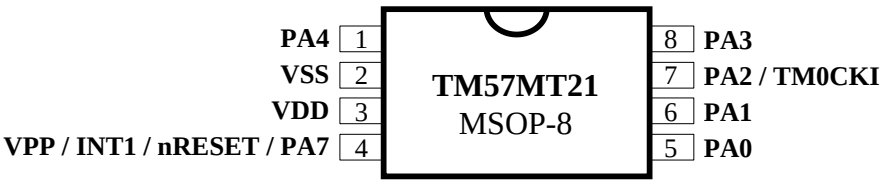
Note:
1. ATKPOL : Touch Key auto scan polarity flag
2. ATKSIT: Touch Key auto scan interval time
3. ATKDT: Touch Key auto scan result manifestation
4. TKDH / TKDL: Touch Key 10 bits data count

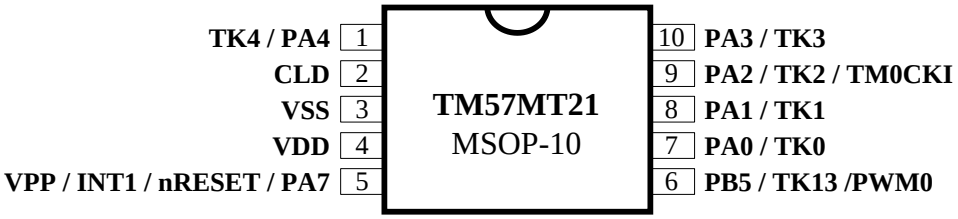
# BLOCK DIAGRAM

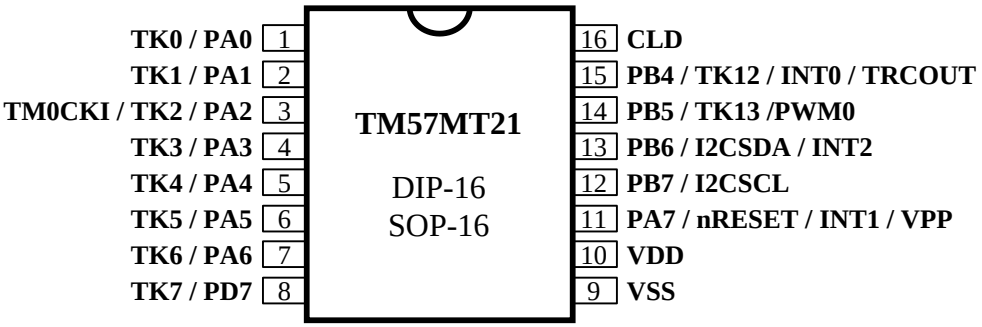


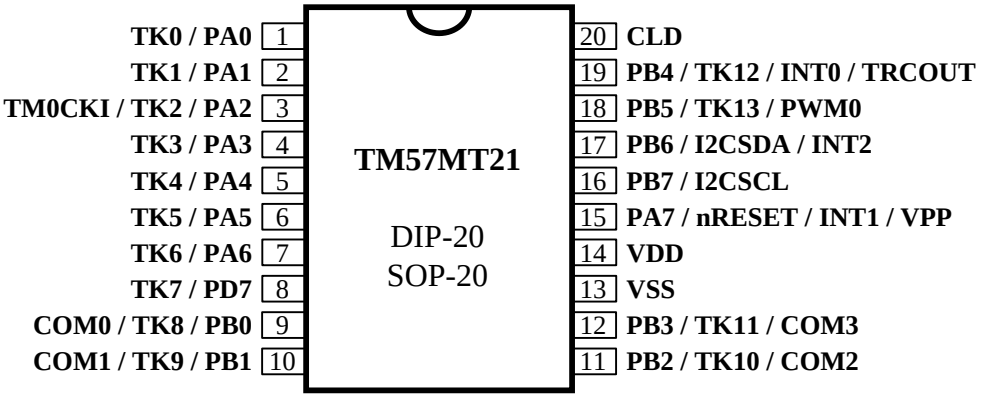

**TM57MT21 Block Diagram**

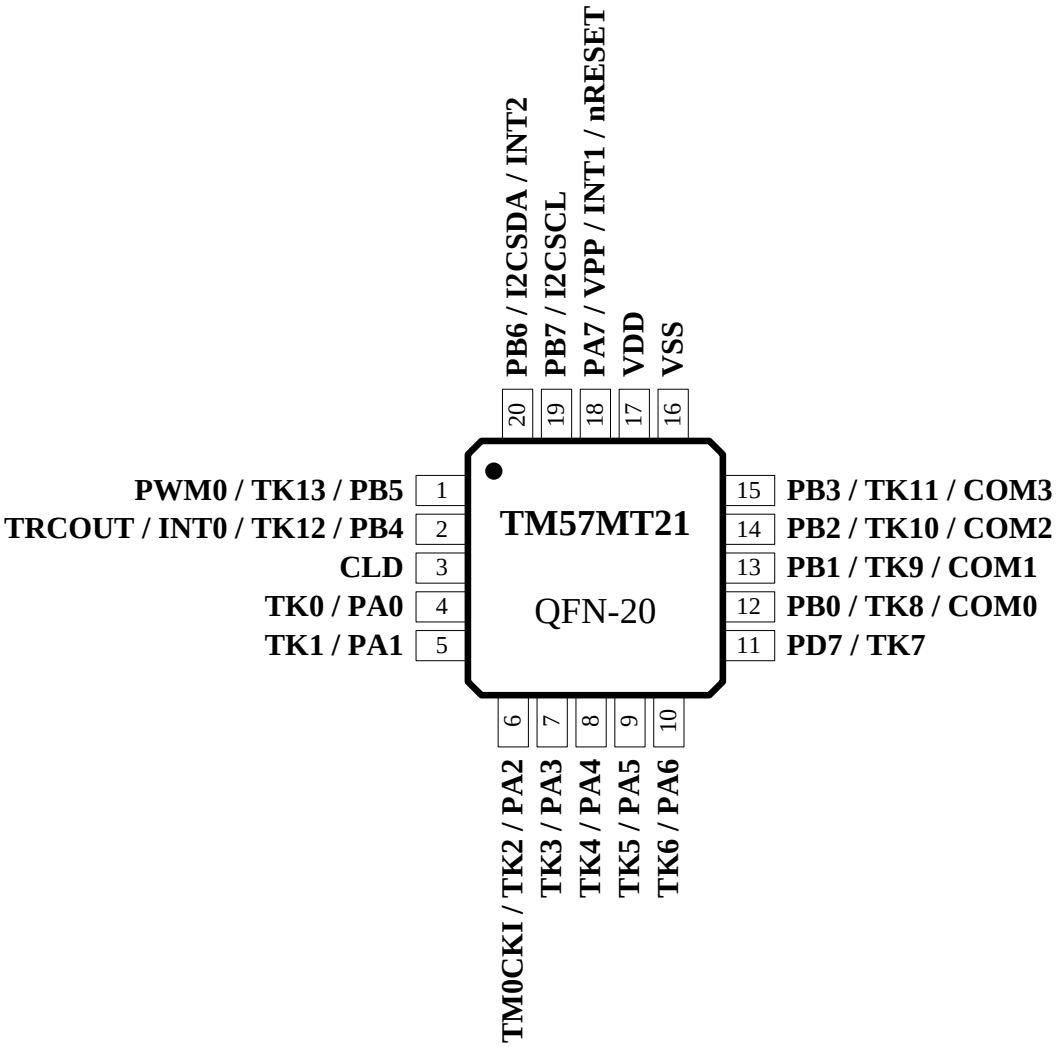
# PIN ASSIGNMENT




PB6 / I2CSDA / INT2
PB7 / I2CSCL
PA7 / VPP / INT1 / nRESET
VDD
VSS
20
19
18
17
16
PWM0 / TK13 / PB5
TRCOUT / INT0 / TK12 / PB4
CLD
TK0 / PA0
TK1 / PA1
1
2
3
4
5
TM57MT21
QFN-20
15
14
13
12
11
PB3 / TK11 / COM3
PB2 / TK10 / COM2
PB1 / TK9 / COM1
PB0 / TK8 / COM0
PD7 / TK7
6
7
8
9
10
TM0CKI / TK2 / PA2
TK3 / PA3
TK4 / PA4
TK5 / PA5
TK6 / PA6

# PIN DESCRIPTION

| Name | In/Out | Pin Description |
|---|---|---|
| PA0–PA6 | 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 resistor is always assignable. |
| PB0–PB7 | 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. |
| PD7 | I/O | Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistor is assignable by software. |
| nRESET | I | External active low reset |
| TRCOUT | O | Touch key clock output |
| VDD, VSS | P | Power Voltage input pin and ground |
| VPP | I | PROM programming high voltage input |
| INT0–INT2 | I | External interrupt input |
| PWM0 | O | PWM0 output |
| TM0CKI | I | Timer0's input in counter mode |
| TK0–TK13 | I | Touch key input |
| CLD | I/O | Touch Key charge collection capacitor connection pin |
| COM0–COM3 | O | LCD Common output |
| I2CSCL | I | I2C serial clock input |
| I2CSDA | I/O | I2C serial data pin |

## PIN SUMMARY

| Pin Number | | | | | Pin Name | Type | GPIO | | | | Function After Reset | Alternate Function | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Input | | Output | | | | | | | |
| 20-SOP/DIP | 20-QFN | 16-SOP/DIP | 10-MSOP | 8-MSOP | | | Weak Pull-up | Ext. Interrupt | O.D | P.P | | PWM | Touch Key | LCD | I2C | MISC |
| 1 | 4 | 1 | 7 | 5 | PA0/TK0 | I/O | O | | O | O | PA0 | | O | | | |
| 2 | 5 | 2 | 8 | 6 | PA1/TK1 | I/O | O | | O | O | PA1 | | O | | | |
| 3 | 6 | 3 | 9 | 7 | PA2/TK2/TM0CKI | I/O | O | | O | O | PA2 | | O | | | TM0CKI |
| 4 | 7 | 4 | 10 | 8 | PA3/TK3 | I/O | O | | O | O | PA3 | | O | | | |
| 5 | 8 | 5 | 1 | 1 | PA4/TK4 | I/O | O | | O | O | PA4 | | O | | | |
| 6 | 9 | 6 | - | - | PA5/TK5 | I/O | O | | O | O | PA5 | | O | | | |
| 7 | 10 | 7 | - | - | PA6/TK6 | I/O | O | | O | O | PA6 | | O | | | |
| 8 | 11 | 8 | - | - | PD7/TK7 | I/O | O | | O | O | PD7 | | O | | | |
| 9 | 12 | - | - | - | PB0/TK8/COM0 | I/O | O | | O | O | PB0 | | O | O | | |
| 10 | 13 | - | - | - | PB1/TK9/COM1 | I/O | O | | O | O | PB1 | | O | O | | |
| 11 | 14 | - | - | - | PB2/TK10/COM2 | I/O | O | | O | O | PB2 | | O | O | | |
| 12 | 15 | - | - | - | PB3/TK11/COM3 | I/O | O | | O | O | PB3 | | O | O | | |
| 13 | 16 | 9 | 3 | 2 | VSS | P | | | | | | | | | | |
| 14 | 17 | 10 | 4 | 3 | VDD | P | | | | | | | | | | |
| 15 | 18 | 11 | 5 | 4 | PA7/INT1/nRESET/VPP | I/O | O | O | O | | PA7 | | | | | nRESET |
| 16 | 19 | 12 | - | - | PB7/I2CSCL | I/O | O | | O | O | PB7 | | | | O | |
| 17 | 20 | 13 | - | - | PB6/I2CSDA/INT2 | I/O | O | O | O | O | PB6 | | | | O | |
| 18 | 1 | 14 | 6 | - | PB5/TK13/PWM0 | I/O | O | | O | O | PB5 | O | O | | | |
| 19 | 2 | 15 | - | - | PB4/TK12/INT0/TRCOUT | I/O | O | O | O | O | PB4 | | O | | | TRCOUT |
| 20 | 3 | 16 | 2 | - | CLD | I/O | | | | | | | O | | | |

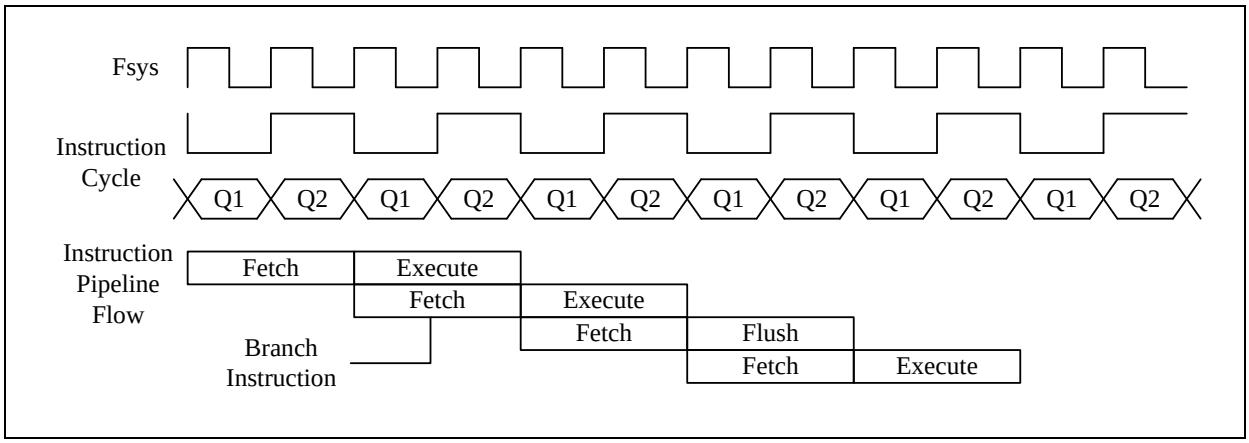
Symbol：P.P. = COM Push-Pull Output
O.D. = Open Drain Output

# FUNCTION DESCRIPTION

## 1. CPU Core

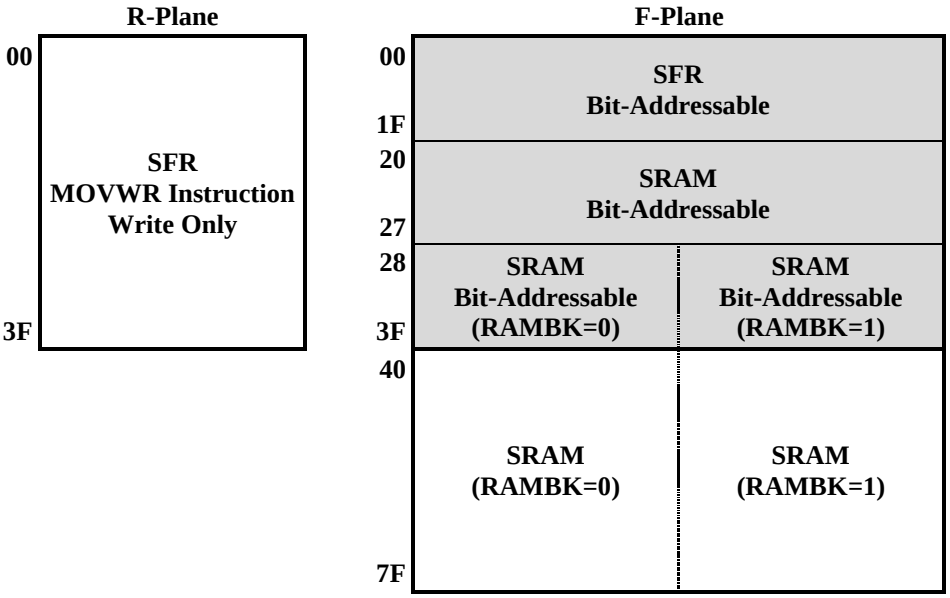
### 1.1 Clock Scheme and Instruction Cycle

The system clock (Fsys) 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 Addressing Mode

There are two Data Memory Planes in CPU, R-Plane and F-Plane. The registers in R-Plane are write-only. The "MOVWR" instruction copy the W-register's content to R-Plane registers by direct addressing mode. The lower locations of F-Plane are reserved for the 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 (F04.7~0) 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. And there are two RAM banks can be selected by RAMBK (F03.5).



◇Example: Write immediate data into R-Plane register

```
MOVLW   AAH     ; Move immediate AAH into W register
MOVWR   05H     ; Move W value into R-Plane location 05H
```

◇Example: Write immediate data into F-Plane register

```
MOVLW   55H     ; Move immediate 55H into W register
MOVWF   20H     ; Move W value into F-Plane location 20H
```

◇Example: Move F-Plane location 20H data into W register

```
MOVFW   20H     ; To get a content of F-Plane location 20H to W
```

◇Example: Clear SRAM Bank0 data by indirect addressing mode

```
        MOVLW   20H             ; W=20H (SRAM start address)
        MOVWF   FSR             ; Set start address of user SRAM into FSR register
        BCF     STATUS, 5       ; Set RAMBK=0
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 user SRAM?
        BTFSS   STATUS, 2       ; Check the Z flag
        GOTO    LOOP            ; If Z=0, goto LOOP label
        …                       ; If Z=1, exit LOOP
```

### 1.3 Programming Counter (PC) and Stack

The Programming Counter is 11-bit wide capable of addressing a 2K x 14 program 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.

◇ Example: To look up the PROM 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, …, 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: TM57MT21 defines 256 ROM addresses as one page, so that TM57MT21 has 8 pages, 000H~0FFH, 100H~1FFH, 200H~2FFH, …, 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, …, 6, 7). If a lookup table has fewer data, it needs not set the starting address at X00H, just only confirm all lookup table data are located at the same page.

### 1.4 ALU and Working (W) Register

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

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

### 1.5 STATUS Register (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. The RAMBK bit is used to the SRAM Bank selection.

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

◇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

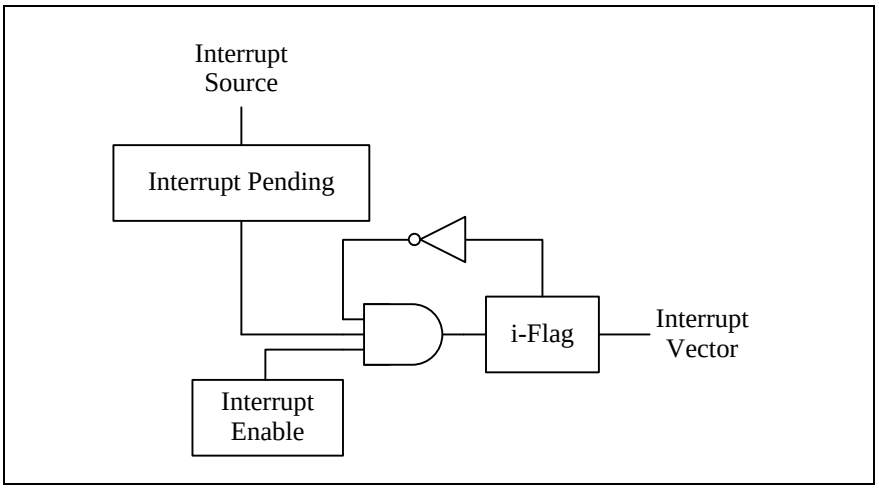
```
BTFSS   STATUS, 0       ; Check the C flag
GOTO    LABEL_1         ; If C=0, goto LABEL_1 label
GOTO    LABEL_2         ; If C=1, goto LABEL_2 label
```

### 1.6 Interrupt

The TM57MT21 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 TM57MT21 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 (INTIE), 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 INT0 (PB4) interrupt request with rising edge trigger

```
            ORG     000H            ; Reset Vector
            GOTO    START           ; Goto user program address

            ORG     001H            ; All interrupt vector
            GOTO    INT             ; If INT0 (PB4) input occurred rising edge

            ORG     002H
START:
            MOVLW   xxxxxx00B
            MOVWR   PBMODH          ; Select INT0 Pin Mode as Mode0
                                    ; Open drain output low or input with Pull-up
            MOVLW   xxx1xxxxB
            MOVWF   PBD             ; Release INT0, it becomes Schmitt-trigger
                                    ; input with input pull-up resistor
            MOVLW   0001 0xxxB
            MOVWR   MR0B            ; Set INT0 interrupt trigger as rising edge
            MOVLW   1111111 0B
            MOVWF   INTIF           ; Clear INT0 interrupt request flag
            MOVLW   0000000 1B
            MOVWF   INTIE           ; Enable INT0 interrupt
MAIN:
            …
            GOTO    MAIN

INT:
            MOVWF   20H             ; Store W data to SRAM 20H
            MOVFW   STATUS          ; Get STATUS data
            MOVWF   21H             ; Store STATUS data to SRAM 21H

            BTFSS   INT0IF          ; Check INT0IF bit
            GOTO    EXIT_INT        ; INT0IF=0, exit interrupt subroutine
            …                       ; INT0 interrupt service routine
            MOVLW   1111111 0B
            MOVWF   INTIF           ; Clear INT0 interrupt request flag
EXIT_INT:
            MOVFW   21H             ; Get SRAM 21H data
            MOVWF   STATUS          ; Restore STATUS data
            MOVFW   20H             ; Restore W data
            RETI                    ; Return from interrupt
```

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F08.7 **I2CIE**: I2C receive/transmit data finished interrupt enable
0: disable
1: enable

F08.6 **ATKIE**: Auto Touch Key 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 (PB6) pin interrupt enable
0: disable
1: enable

F08.1 **INT1IE**: INT1 (PA7) pin interrupt enable
0: disable
1: enable

F08.0 **INT0IE**: INT0 (PB4) pin interrupt enable
0: disable
1: enable

| F09 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIF | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F09.7 **I2CIF**: I2C interrupt event pending flag
This bit is set by H/W while
a. I2CRCD0 or I2CRCD1 receive new 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

F09.6 **ATKIF**: Auto Touch Key interrupt event pending flag
This bit is set by H/W while Key's TK Data Count is over the pre-set compare threshold range, write 0 to this bit will clear this flag

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

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

F09.3 **WKTIF**: WKT interrupt event pending flag
This bit is set by H/W while WKT time out, write 0 to this bit will clear this flag

F09.2 **INT2IF**: INT2 interrupt event pending flag
This bit is set by H/W at INT2 pin's falling edge, write 0 to this bit will clear this flag

F09.1 **INT1IF**: INT1 interrupt event pending flag
This bit is set by H/W at INT1 pin's falling edge, write 0 to this bit will clear this flag

F09.0 **INT0IF**: INT0 interrupt event pending flag
This bit is set by H/W at INT0 pin's falling/rising edge, write 0 to this bit will clear this flag

| R0B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MR0B | – | – | – | INT0EDG | – | TRCNOE | WKTPSC | |
| R/W | – | – | – | W | – | W | W | |
| Reset | – | – | – | 0 | – | 0 | 1 | 1 |

R0B.4 **INT0EDG:** INT0 pin (PB4) edge interrupt event
0: falling edge to trigger
1: rising edge to trigger

# 2. Chip Operation Mode

## 2.1 Reset

The TM57MT21 can be RESET in four ways.

- Power-On-Reset
- Low Voltage Reset (LVR)
- External Pin Reset (PA7)
- Watchdog Reset (WDT)

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values. The LVR level is selected by the SYSCFG register value. The Low Voltage Reset features static reset when supply voltage is below a threshold level. There is only one threshold level 1.9V can be selected. The External Pin Reset and Watchdog Reset can be disabled or enabled by the SYSCFG register. These two resets also set all the control registers to their default reset value. The TO/PD flag is not affected by these resets.

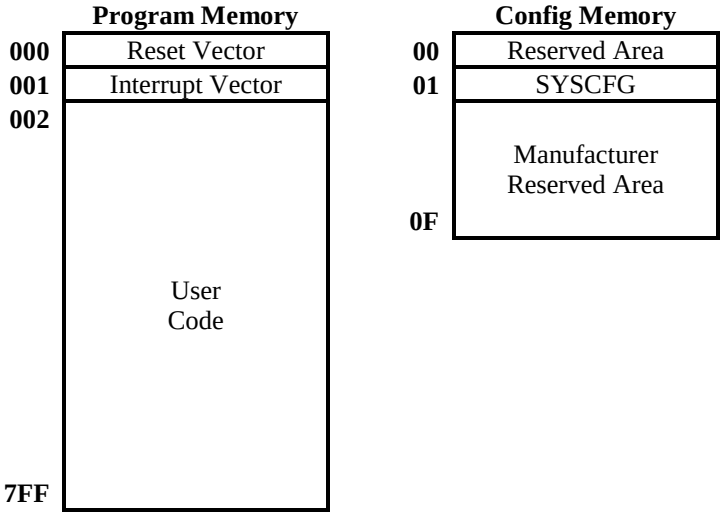
## 2.2 System Configuration Register (SYSCFG)

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 LVR threshold voltage and chip operation mode by SYSCFG register. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in MTP will be protected, when user reads MTP.

| **Bit** | **13~0** | |
|---|---|---|
| **Default Value** | **0x_00xx_00xx_xxxx** | |
| Bit | Description | |
| 13 | **PROTECT**: Code Protection Selection | |
| | 0 | Disable |
| | 1 | Enable |
| 12 | **Reserved** | |
| 11-10 | **LVR**: Low Voltage Reset Mode | |
| | 00 | LVR level=1.9V, always enabled |
| | 01 | LVR disabled |
| | 10 | Reserved |
| | 11 | LVR level=1.9V, disabled in Power-down Mode |
| 9-8 | **Reserved** | |
| 7 | **XRSTE**: External Pin (PA7) Reset Enable | |
| | 0 | PA7 as Input PIN |
| | 1 | Enable |
| 6 | **WDTE**: WDT Reset Enable | |
| | 0 | Disable |
| | 1 | Enable |
| 5-0 | **Reserved** | |

## 2.3 MTP Program ROM

The MTP ROM of this device is 2K words, with an extra INFO area to store the SYSCFG. 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 not set or MTP ROM is erased. That is, un-protect the PROTECT bit needs the erased MTP ROM.




## 2.4 Power-Down Mode

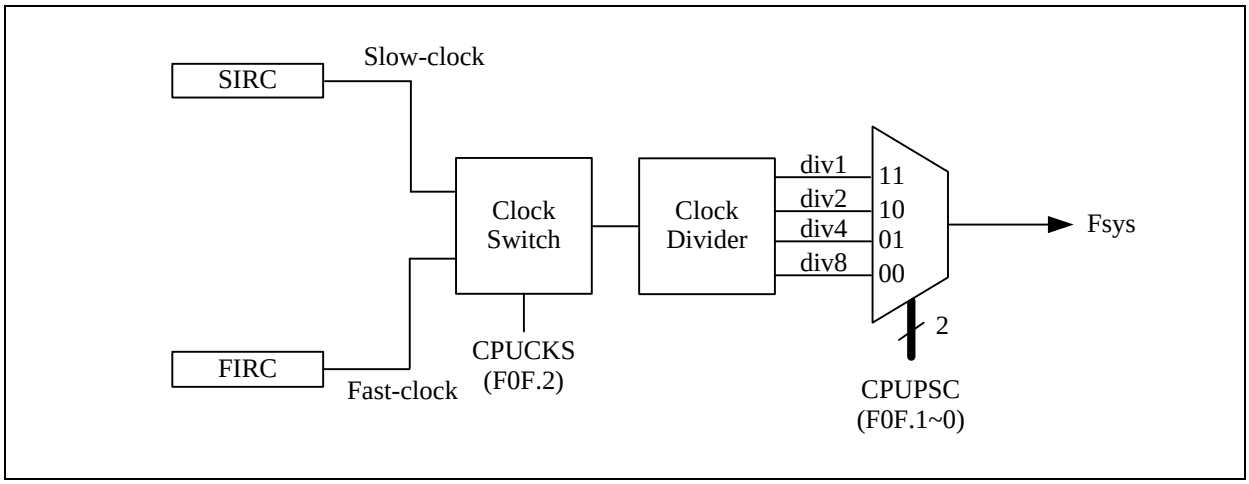
The Power-down mode includes IDLE Mode and STOP Mode. It is activated by SLEEP instruction. During the Power-down mode, the system clock and peripherals stop to minimize power consumption, whether the WDT/WKT Timer are working or not depend on F/W setting. The Power-down mode can be terminated by Reset, or enabled Interrupts (External pins, WKT, ATK and I2C interrupts) or PA0-PA6, PB6-PB7 and PD7 pins low level wake up.

| 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 (STOP/IDLE) Mode

## 2.5 Dual System Clock

TM57MT21 is designed with dual-clock system. There are two kinds of clock source, SIRC (Slow Internal RC) Clock and FIRC (Fast Internal RC) Clock. Both of them can be applied to CPU kernel as system clock source. Refer to the figure as below.




**FAST Mode:**

TM57MT21 enters FAST mode by setting the CPUCKS (F0F.2). In this mode, the system clock source is FIRC. The Timer0, Timer1 and PWM0 blocks are driven by Fast-clock.

**SLOW Mode:**

After power on or reset, TM57MT21 enters SLOW mode, the default system clock source is SIRC. In this mode, Fast-clock is stopped and Slow-clock is enabled for power saving. All peripheral blocks (Timer0, Timer1 and PWM0, etc…) are driven by Slow-clock.

**IDLE Mode:**

When SLOWSTP (F0F.4) is cleared, the TM57MT21 will enter the "IDLE Mode" after executing the SLEEP instruction. In this mode, the Slow-clock will keep running to provide clock to WKT/WDT or Auto Touch Key block. CPU stops fetching code and all blocks are stop.

Another way to keep clock oscillation in IDLE mode is setting WKTIE=1 (F08.3) before executing the SLEEP instruction. In such condition, the Slow-clock will also keep running no matter SLOWSTP is set or cleared. It is possible to keep WKT working for wake-up CPU periodically in the IDLE mode, which is useful for low power mode Touch Key detection.

The third way to keep clock oscillation in IDLE mode is setting TKAUTO≠00b (F13.4~3) before executing the SLEEP instruction. In such condition, The Auto Touch Key will keep working.

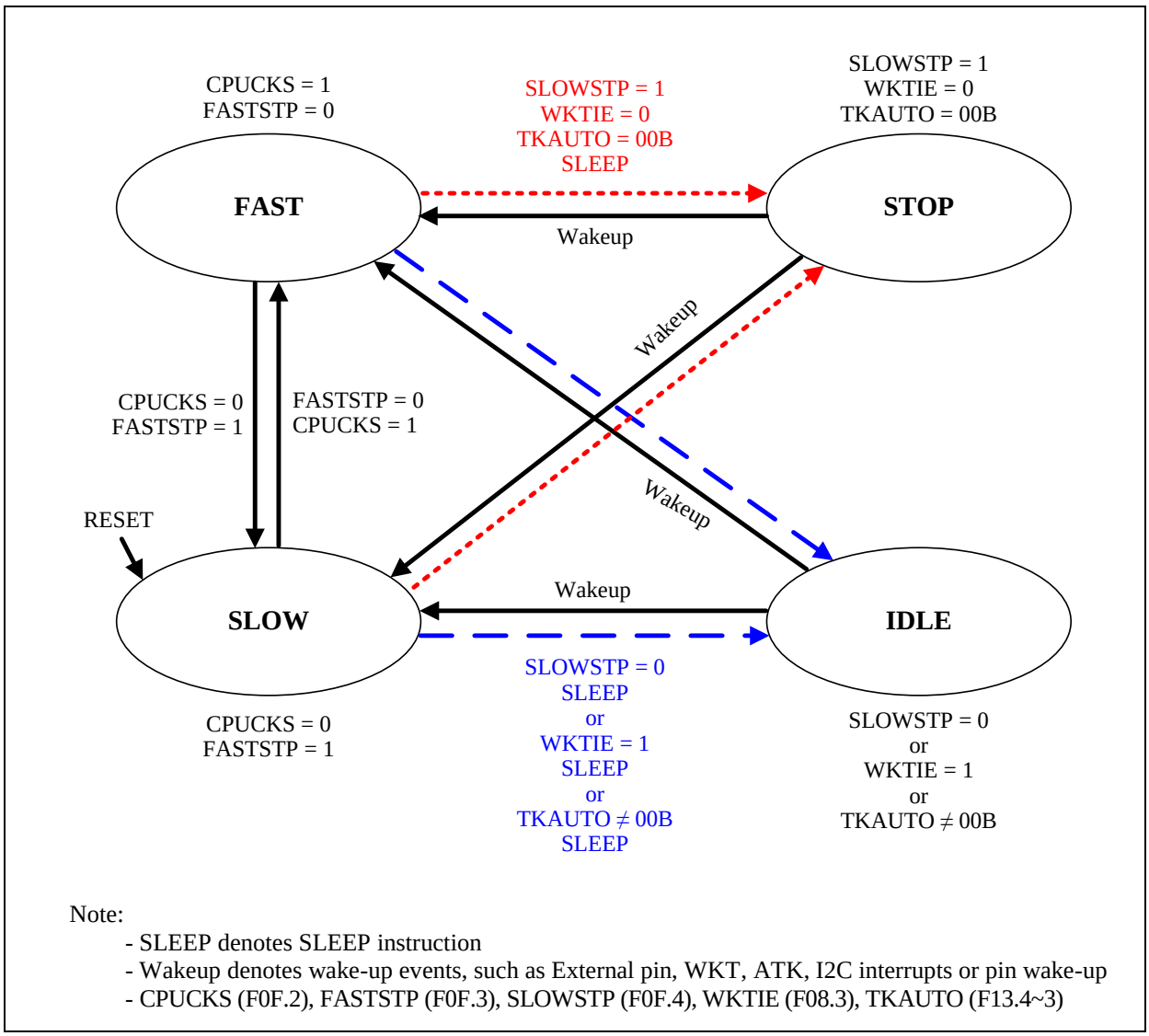
**STOP Mode:**

When SLOWSTP is set, WKTIE and TKAUTO are both cleared, all blocks will be turned off and the TM57MT21 will enter the "STOP Mode" after executing the SLEEP instruction. STOP mode is similar to IDLE mode. The difference is all clock oscillators either Fast-clock or Slow-clock are stopped and no clocks are generated.

## 2.6 Dual System Clock Modes Transition

TM57MT21 is operated in one of four modes: FAST Mode, SLOW Mode, IDLE Mode, and STOP Mode.

**Modes Transition Diagram:**




Note:

- SLEEP denotes SLEEP instruction
- Wakeup denotes wake-up events, such as External pin, WKT, ATK, I2C interrupts or pin wake-up
- CPUCKS (F0F.2), FASTSTP (F0F.3), SLOWSTP (F0F.4), WKTIE (F08.3), TKAUTO (F13.4~3)

**CPU Mode & Clock Functions Table:**

| Mode | Oscillator | Fsys | Fast-clock | Slow-clock | TM0/TM1 PWM0 | WKT | ATK | Wakeup event |
|---|---|---|---|---|---|---|---|---|
| **FAST** | FIRC | Fast-clock | Run | Run | Run | Run/Stop | Run/Stop | – |
| **SLOW** | SIRC | Slow-clock | Stop | Run | Run | Run/Stop | Run/Stop | – |
| **IDLE** | SIRC | Stop | Stop | Run | Stop | Run/Stop | Run/Stop | WKT/ATK/I2C/IO |
| **STOP** | Stop | Stop | Stop | Stop | Stop | Stop | Stop | I2C/IO |

**FAST Mode transits to SLOW Mode:**

The source clock of Slow-clock is SIRC. The following steps are suggested to be executed by order when FAST mode transits to SLOW mode:

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

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

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

```
BCF     CPUCKS      ; Switch system clock source to Slow-clock
BSF     FASTSTP     ; Stop Fast-clock
```

**SLOW Mode transits to FAST Mode:**

The source clock of Fast-clock is FIRC. The following steps are suggested to be executed by order when SLOW mode transits 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) Enable Slow-clock (SLOWSTP=0)

(2) Execute SLEEP instruction

IDLE mode can be woken up by interrupts (External pins, WKT, ATK or I2C) or PA0-PA7, PB6-PB7 and PD7 pins low level wake up.

◇Example: Switch operating mode to IDLE mode

```
BCF     SLOWSTP     ; Enable Slow-clock
SLEEP               ; Enter IDLE mode
```

**STOP Mode Setting:**

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

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

(2) Disable WDT/WKT (WKTIE=0)

(3) Disable ATK (TKAUTO=00b)

(4) Execute SLEEP instruction

STOP mode can be woken up by interrupt (External pins or I2C) or PA0-PA7, PB6-PB7 and PD7 pins low level wake up.

◇Example: Switch operating mode to STOP mode

```
BSF     SLOWSTP         ; Stop Slow-clock
BCF     WKTIE           ; Disable WDT/WKT
MOVLW   xxx00xxxB       ; Disable ATK
MOVWF   ATKCTL
SLEEP                   ; Enter STOP mode
```

| F0F | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| CLKCTL | – | – | – | SLOWSTP | FASTSTP | CPUCKS | CPUPSC | |
| R/W | – | – | – | R/W | R/W | R/W | R/W | |
| Reset | – | – | – | 0 | 0 | 0 | 1 | 1 |

F0F.4 **SLOWSTP**: Slow-clock stop
0: Slow-clock is running
1: Slow-clock stops running in Power-down mode

F0F.3 **FASTSTP**: Fast-clock stop
0: Fast-clock is running
1: Fast-clock stops running

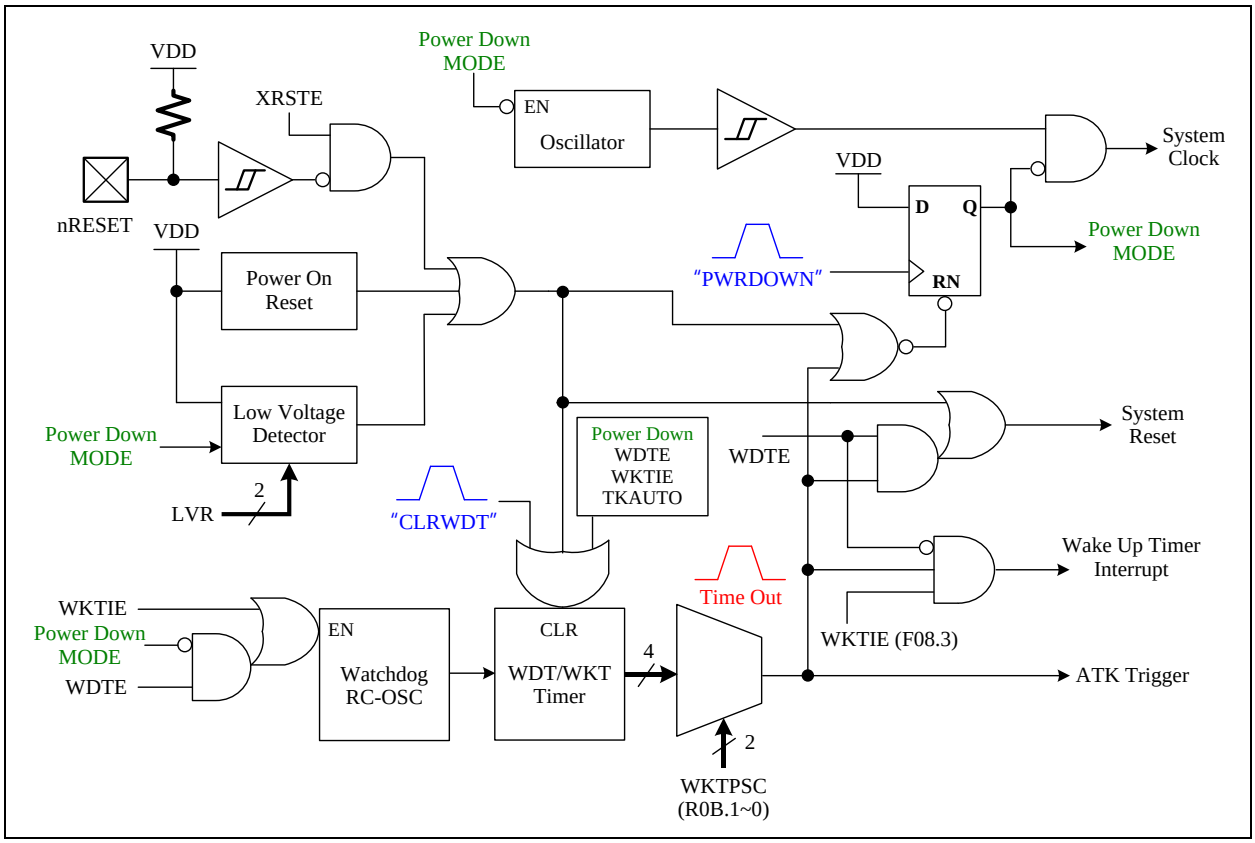
F0F.2 **CPUCKS**: System clock source select
0: Slow-clock
1: Fast-clock (forbid using, when CPUPSC=3)

F0F.1~0 **CPUPSC**: System clock source prescaler. System clock source
00: divided by 8
01: divided by 4
10: divided by 2
11: divided by 1 (forbid using, when CPUCKS=1)

# 3. Peripheral Functional Block

## 3.1 Watchdog Timer (WDT) /Wakeup Timer (WKT)

The WDT and WKT share the same internal RC Timer. The overflow period of WDT/WKT can be selected from 30 ms to 240 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (SYSCFG[6], WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out flag WKTIF" (F09.3). It generates WKT overflow time out interrupt if the WKTIE" (F08.3) bit is set. If TKAUTO≠00b, the WKT also generates the ATK trigger signal for TK H/W auto mode. If WKTIE is cleared (no matter WDTE is 1 or 0) and TKAUTO=00b, the internal RC Timer stops for power saving in Stop mode. If WKTIE, WDTE and TKAUTO are all cleared, the internal RC Timer will also stop for power saving in normal mode. Refer to the following table and figure.




The WDT and WKT's behavior in different Mode are shown as below table.

| Mode | | WDTE | WKTIE | TKAUTO | Watchdog RC Oscillator |
|---|---|---|---|---|---|
| Normal Mode | | 0 | 0 | 00 | Stop |
| | | 0 | 1 | | Run |
| | | 1 | 0 | | |
| | | 1 | 1 | | |
| | | x | x | 01/10/11 | Run |
| Power-down Mode | IDLE Mode | x | x | 01/10/11 | Run |
| | | x | 1 | x | |
| | STOP Mode | x | 0 | 00 | Stop |

| 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 | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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 | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F09.3 **WKTIF**: WKT interrupt event pending flag
This bit is set by H/W while WKT time out, 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 | – | – | – | – | – | – | – | – |

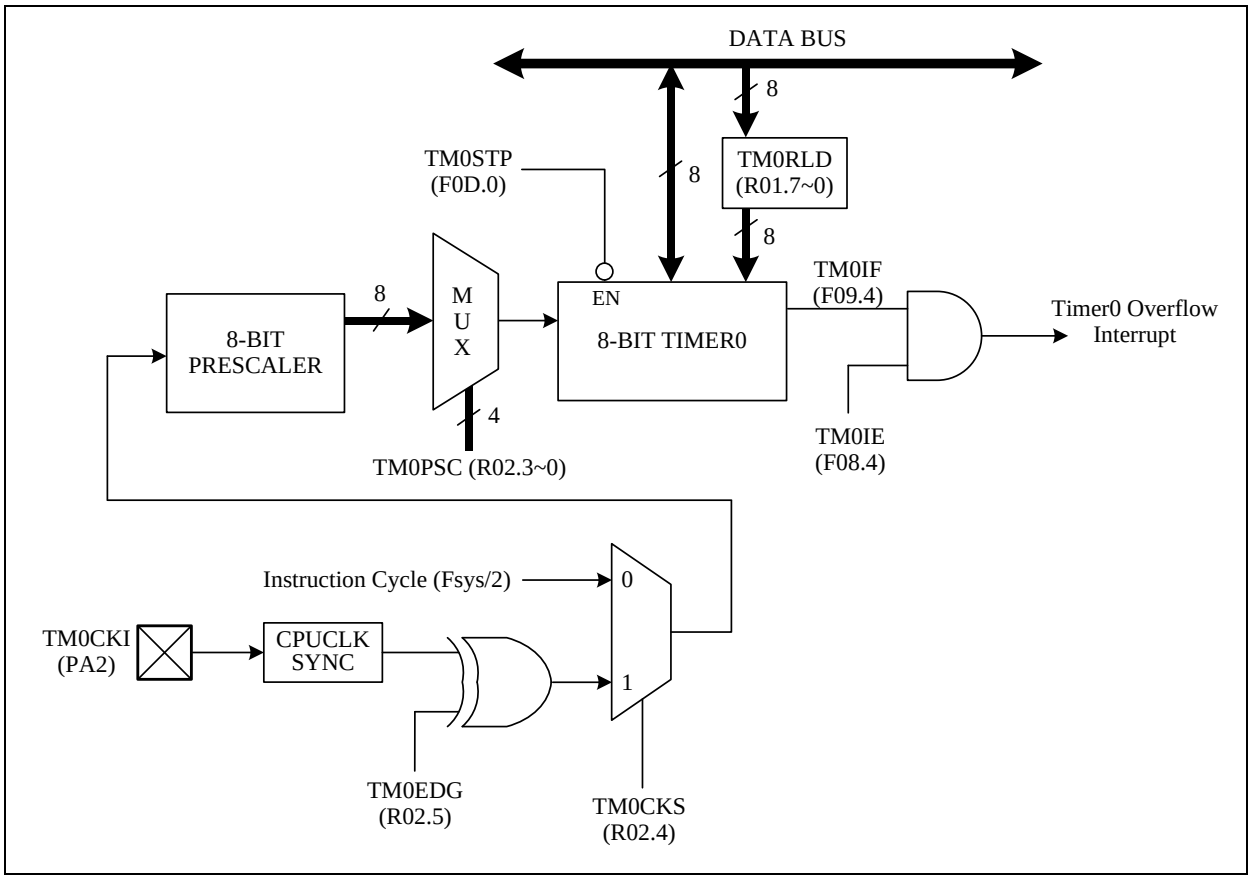
R04.7~0 **WDTCLR:** Write this register to clear WDT

| R0B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MR0B | – | – | – | INT0EDG | – | TRCNOE | WKTPSC | |
| R/W | – | – | – | W | – | W | W | |
| Reset | – | – | – | 0 | – | 0 | 1 | 1 |

R0B.1~0 **WKTPSC:** WDT/WKT pre-scale select:
00: 30ms
01: 60ms
10: 120ms
11: 240ms

### 3.2 Timer0: 8-bit Timer/Counter with Pre-scale (PSC)

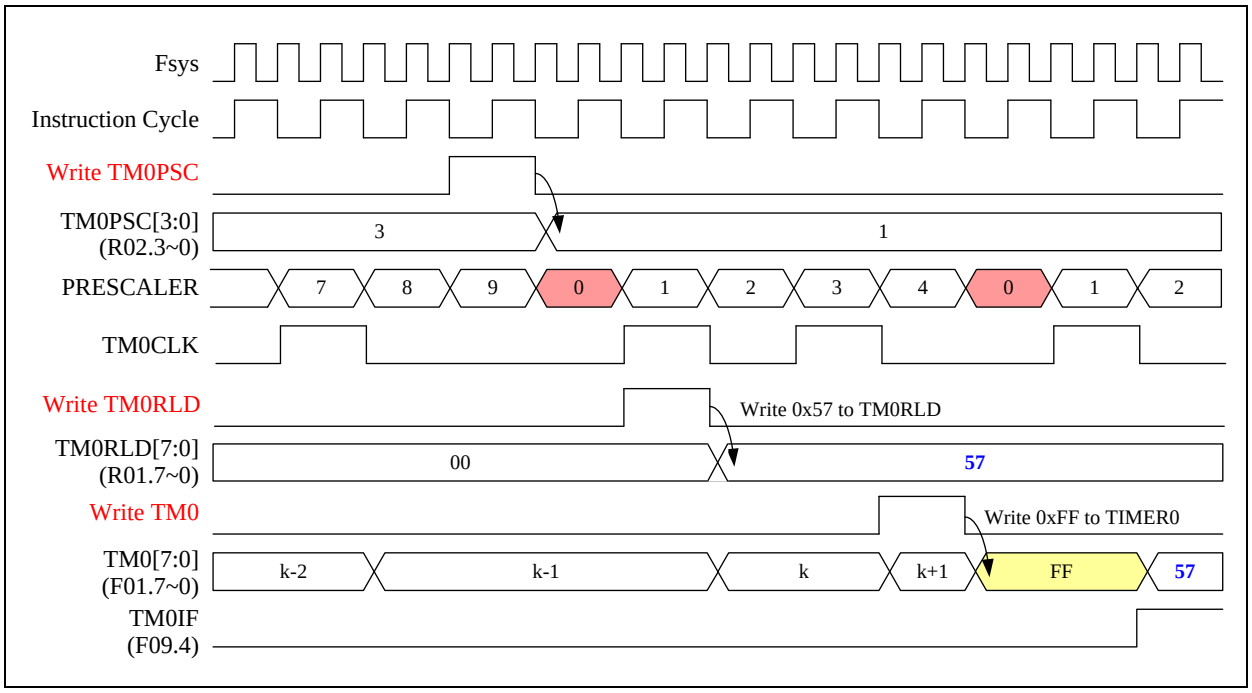
The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TM0CKI (PA2) rising/falling input. The Timer0's increasing rate is determined by the TM0PSC (R02.3~0). The Timer0 can generate interrupt flag TM0IF (F09.4) and also reload the new data from TM0RLD (R01.7~0) when it rolls over. It generates Timer0 interrupt if the TM0IE (F08.4) bit is set. Timer0 can be stopped counting if the TM0STP (F0D.0) bit is set.




**Timer0 Block Diagram**

**Timer Mode:**

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




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

The equation of Timer0 interrupt frequency is as following:

Timer0 interrupt frequency=Instruction cycle frequency/TM0PSC/ (256-TM0RLD)

◇ Example: Setup Timer0 work in Timer mode, Fsys=Fast-clock/CPUPSC=FIRC 8 MHz/2=4 MHz

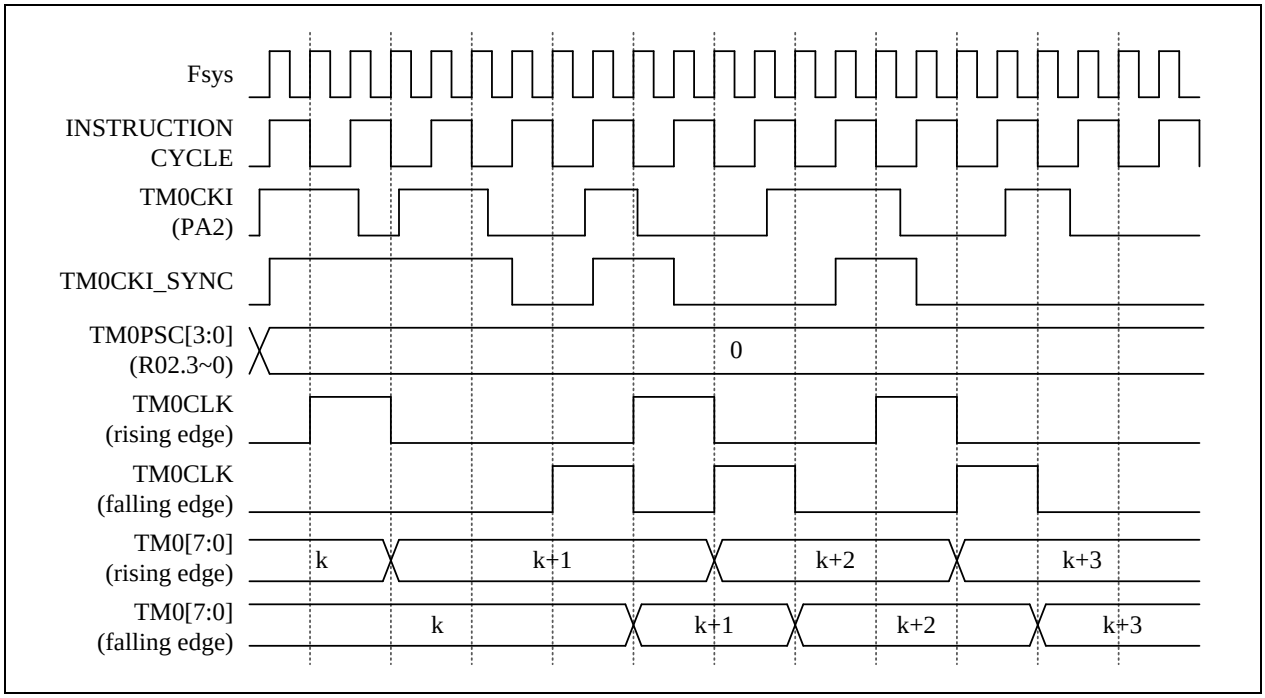
```
; Setup Timer0 clock source and divider
        MOVLW   00000B
        MOVWF   CLKCTL          ; CPUPSC=10b, divided by 2
        BSF     CPUCKS          ; Set Fast-clock as system clock
        MOVLW   00x0 0100B      ; TM0CKS=0, Timer0 clock is instruction cycle
        MOVWR   TM0CTL          ; TM0PSC=0100b, divided by 16

; Setup Timer0 reload data
        MOVLW   80H
        MOVWR   TM0RLD          ; Set Timer0 reload data=128

; Setup Timer0
        BSF     TM0STP          ; Timer0 stops counting
        CLRF    TM0             ; Clear Timer0 content

; Enable Timer0 and interrupt function
        MOVLW   1110 1111B
        MOVWF   INTIF           ; Clear Timer0 request interrupt flag
        BSF     TM0IE           ; Enable Timer0 interrupt function
        BCF     TM0STP          ; Enable Timer0 counting
```

Timer0 clock source is Fsys/2=4 MHz/2=2 MHz, Timer0 divided by 16

Timer0 interrupt frequency=2 MHz/16/ (256-128) =976.56 Hz

**Counter Mode:**

If TM0CKS=1, then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle that means the high/low time durations of TM0CKI must be longer than one instruction cycle time to guarantee each TM0CKI's 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) for TM0CKI**

◇ Example: Setup Timer0 works in Counter mode

```
; Setup Timer0 clock source and divider
        MOVLW   00110000B       ; TM0EDG=1, counting edge is falling edge
        MOVWR   TM0CTL          ; TM0CKS=1, Timer0 clock is TM0CKI
                                ; TM0PSC=0000b, divided by 1
; Setup Timer0
        BSF     TM0STP          ; Timer0 stops counting
        CLRF    TM0             ; Clear Timer0 content

; Enable Timer0 and read Timer0 counter
        BCF     TM0STP          ; Enable Timer0 counting
        …
        BSF     TM0STP          ; Timer0 stops 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 | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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 | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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

| F0D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MF0D | – | – | – | VDDFLT | EMIIMPV | PWM0CLR | TM1STP | TM0STP |
| R/W | – | – | – | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | 0 | 1 | 1 | 0 | 0 |

F0D.0 **TM0STP**: Timer0 counter stop
0: Timer0 is counting
1: Timer0 stops counting

| R01 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TM0RLD | TM0RLD | | | | | | | |
| R/W | 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 | – | – | TM0EDG | TM0CKS | TM0PSC | | | |
| R/W | – | – | W | W | W | | | |
| Reset | – | – | 0 | 0 | 0 | 0 | 0 | 0 |

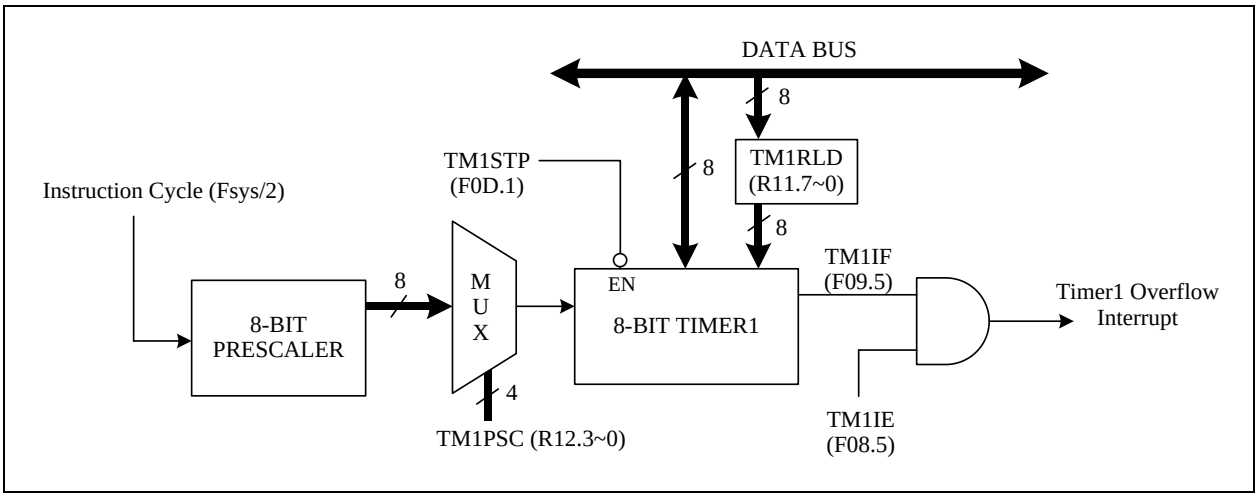
R02.5 **TM0EDG:** TM0CKI (PA2) edge selection for Timer0 prescaler count
0: TM0CKI rising edge for Timer0 prescaler count
1: TM0CKI falling edge for Timer0 prescaler count

R02.4 **TM0CKS:** Timer0 clock source select
0: Instruction Cycle (Fsys/2) as Timer0 prescaler clock
1: TM0CKI (PA2) as Timer0 prescaler clock

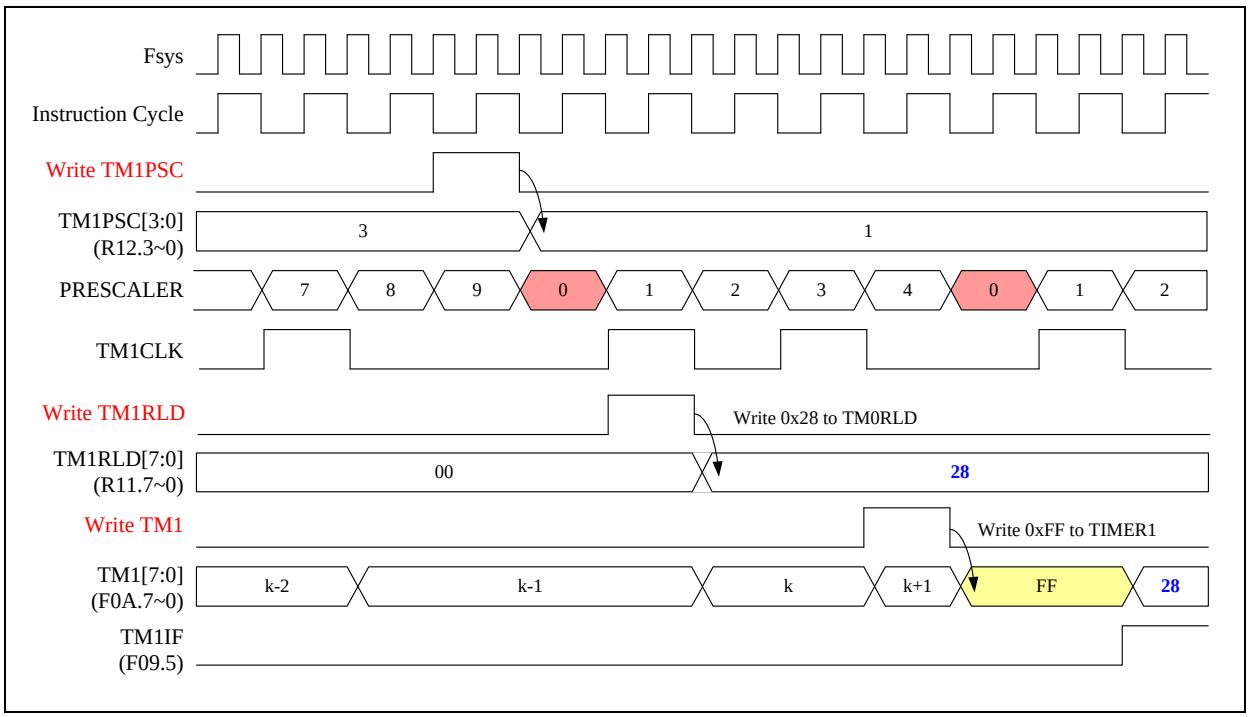
R02.3~0 **TM0PSC:** Timer0 prescaler. Timer0 clock source
0000: divided by 1
0001: divided by 2
0010: divided by 4
0011: divided by 8
0100: divided by 16
0101: divided by 32
0110: divided by 64
0111: divided by 128
1xxx: divided by 256

### 3.3 Timer1: 8-bit Timer with Pre-scale (PSC)

The Time1 is an 8-bit wide 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 based on the pre-scaled instruction cycle. The Timer1's increasing rate is determined by the TM1PSC (R12.3~0). The Timer1 can generate interrupt flag TM1IF (F09.5) and also reload the new data from TM1RLD (R11.7~0) when it rolls over. It generates Timer1 interrupt if the TM1IE (F08.5) bit is set. Timer1 can be stopped counting if the TM1STP (F0D.1) bit is set.




**Timer1 Block Diagram**




**Timer1 works in Timer mode**

When the Timer1 prescaler (TM1PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer1 count. TM1CLK is the internal signal that causes the Timer1 to increase by 1 at the end of TM1CLK. TM1WR is also the internal signal that indicates the Timer1 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer1 counts from FFh to TM1RLD data, TM1IF (Timer1 Interrupt Flag) will be set to 1 and generate interrupt if TM1IE (Timer1 Interrupt Enable) is set. The timing diagram describes the Timer1 works in pure Timer mode is shown in above.

The equation of Timer1 interrupt frequency is as following:

Timer1 interrupt frequency=Instruction cycle frequency/TM1PSC/ (256-TM1RLD)

◇Example: CPU is running in SLOW mode, Fsys=Slow-clock/CPUPSC=SIRC 17 KHz/2=8.5 KHz

```
; Setup Timer1 clock source and divider
        MOVLW   000000 10B      ; Set Slow-clock as system clock
        MOVWF   CLKCTL          ; CPUPSC=10b, divided by 2
        MOVLW   00000101B
        MOVWR   TM1CTL          ; TM1PSC=0101b, divided by 32

; Setup Timer1 reload data
        MOVLW   7CH
        MOVWR   TM1RLD          ; Set Timer1 reload data = 124

; Setup Timer1
        BSF     TM1STP          ; Timer1 stops counting
        CLRF    TM1             ; Clear Timer1 content

; Enable Timer1 and interrupt function
        MOVLW   110 11111B
        MOVWF   INTIF           ; Clear Timer1 request interrupt flag
        BSF     TM1IE           ; Enable Timer1 interrupt function
        BCF     TM1STP          ; Enable Timer1 counting
```

Timer1 clock source is Fsys/2=8.5 KHz/2=4.25 KHz, Timer1 divided by 32

Timer1 interrupt frequency=4.25 KHz/32/ (256-124) =1.006 Hz

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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 | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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

| F0A | 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 |

F0A.7~0 **TM1:** Timer1 content

| F0D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MF0D | – | – | – | VDDFLT | EMIIMPV | PWM0CLR | TM1STP | TM0STP |
| R/W | – | – | – | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | 0 | 1 | 1 | 0 | 0 |

F0D.1 **TM1STP**: Timer1 counter stop
0: Timer1 is counting
1: Timer1 stops counting

| R11 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TM1RLD | TM1RLD | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R11.7~0 **TM1RLD:** Timer1 reload data

| R12 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TM1CTL | – | – | – | – | TM1PSC | | | |
| R/W | – | – | – | – | W | | | |
| Reset | – | – | – | – | 0 | 0 | 0 | 0 |

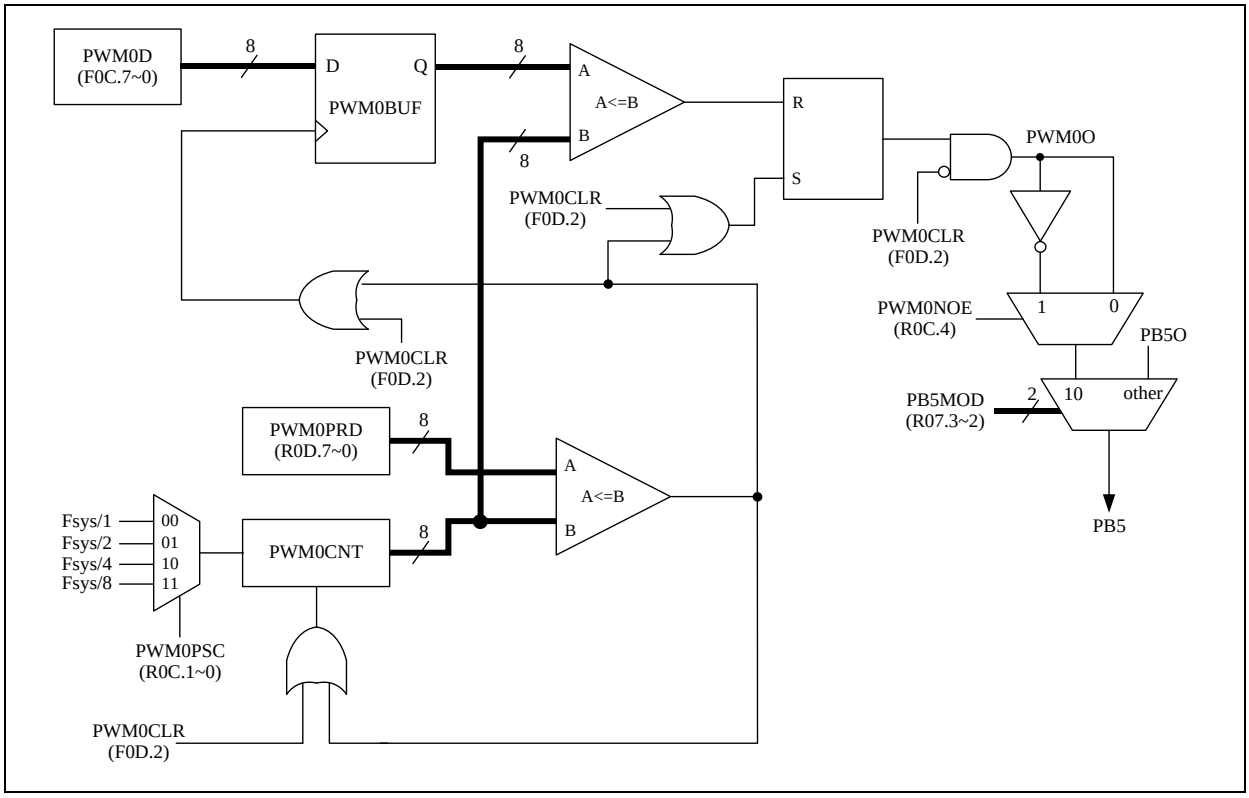
R12.3~0 **TM1PSC:** Timer1 prescaler. Timer1 clock source (Fsys/2)
0000: divided by 1
0001: divided by 2
0010: divided by 4
0011: divided by 8
0100: divided by 16
0101: divided by 32
0110: divided by 64
0111: divided by 128
1xxx: divided by 256

### 3.4 PWM0

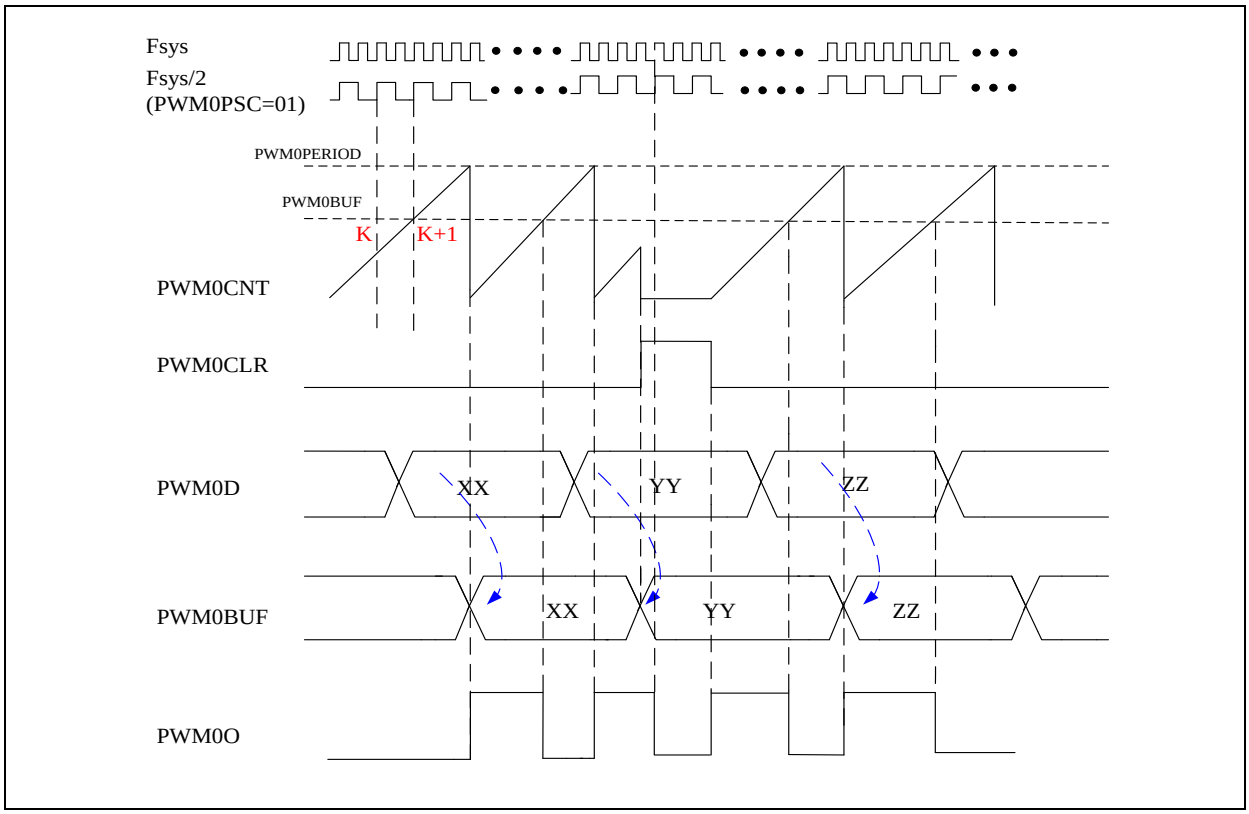
TM57MT21 has a built-in 8-bit PWM generator. The source clock comes from system clock (Fsys) divided by 1, 2, 4, and 8 according to PWM0PSC (R0C.1~0). The PWM0 duty cycle can be changed with writing to PWM0D (F0C.7~0). Writing to PWM0D will not change the current PWM duty until the current PWM period complete. When current PWM period is finish, the new value of PWM0D will be updated to the PWM0BUF.

The PWM0 will be output to PB5 if PB5's Pin Mode "PB5MOD" (R07.3~2) is set as Mode2. The PWM0 output can be set as CMOS push-pull output mode or open-drain output mode. When PB5MOD =2 and PBD[5] =0, the PWM0 output is CMOS push-pull output mode. When PB5MOD=2 and PBD[5] =1, the PWM0 output is open-drain output mode. The complement of PWM0, PWM0N, will be output to PB5 if PWM0NOE (R0C.2) is set. Setting the PWM0CLR (F0D.2) bit will clear the PWM0 counter and load the PWM0D to PWM0BUF, PWM0CLR bit must be cleared so that the PWM0 counter can count. Figure shows the block diagram of PWM0

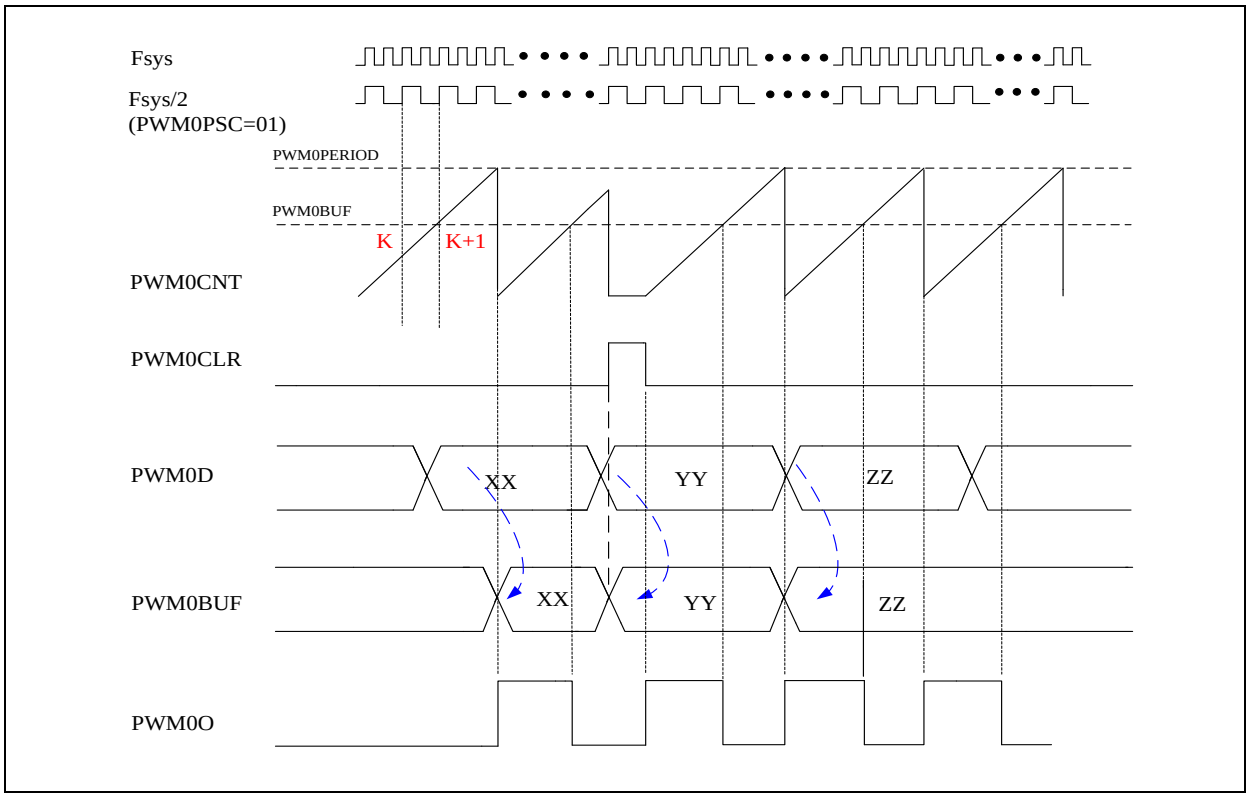
Note that the default value of PWM0CLR bit is '1'.




The next two Figures show the PWM0 waveforms. When PWM0CLR bit is set or PWM0BUF equals to zero, the PWM0 output is cleared to '0' no matter what its current status is. Once the PWM0CLR bit is cleared and PWM0BUF is not zero, the PWM0 output is set to '1' to begin a new PWM cycle. PWM0 output will be '0' when PWM0CNT is greater than or equal to PWM0BUF. PWM0CNT keeps counting up when equals to PWM0PRD (R0D.7~0), the PWM0 output is set to '1' again.




**PWM0 Timing (PWM0CLR before PWM0CNT reaches PWM0BUF)**




**PWM0 Timing (PWM0CLR after PWM0CNT over PWM0BUF)**

◇ Example: CPU is running in FAST mode, Fsys=Fast-clock/CPUPSC=FIRC 8 MHz/2=4 MHz

```
; Setup PWM0 prescaler, period, and duty
        BSF     PWM0CLR         ; PWM0CLR=1, PWM0 clear and hold

; CLK_setting
        MOVLW   00000 1 10B     ; CPUCKS=1, select Fast-clock as system clock
        MOVWF   CLKCTL          ; CPUCKS=10b,System clock source divided by 2

; PINMODE _setting
        MOVLW   xxxx10xxB       ; PB5MOD=10b
        MOVWR   PBMODH          ; set PB5's Pin Mode as Mode2 (PWM0 output)
        BCF     PBD, 5          ; set PWM output as CMOS push-pull output mode

; PWM_setting
        MOVLW   00000 0 01B     ; PWM0NOE=0, PWM0 positive output to PB5
        MOVWR   PWM0CTL         ; PWM0PSC=01b, divided by 2

        MOVLW   FFH
        MOVWR   PWM0PRD         ; Set PWM0 period=FFh+1=256

        MOVLW   80H
        MOVWF   PWM0D           ; Set PWM0 duty=80h=128
        BCF     PWM0CLR         ; PWM0CLR=0, PWM0 is running
```

PWM0 output duty=PWM0D/ (PWM0PRD+1) =128/ (255+1) =1/2

PWM clock=Fsys=4 MHz, PWM clock divided by 2

PWM0 output frequency=4 MHz/2/ (255+1)  7812.5 Hz

| F0C | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PWM0D | PWM0D | | | | | | | |
| R/W | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F0C.7~0 **PWM0D**: PWM0 duty

| F0D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MF0D | – | – | – | VDDFLT | EMIIMPV | PWM0CLR | TM1STP | TM0STP |
| R/W | – | – | – | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | – | 0 | 1 | 1 | 0 | 0 |

F0D.2 **PWM0CLR**: PWM0 clear and hold
0: PWM0 is running
1: PWM0 is clear and hold

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

R07.3~2 **PB5MOD**: PB5 I/O mode control
00: Mode0
01: Mode1
10: Mode2, PWM0 Output
PBD[5] =1, Open Drain output
PBD[5] =0, CMOS output
11: Mode3

| R0C | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PWM0CTL | – | – | – | – | – | PWM0NOE | PWM0PSC | |
| R/W | – | – | – | – | – | W | W | |
| Reset | – | – | – | – | – | 0 | 0 | 0 |

R0C.2 **PWM0NOE**: PWM0 output select
0: positive output
1: negative output

R0C.1~0 **PWM0PSC**: PWM0 prescaler, PWM0 clock source (Fsys)
00: divided by 1
01: divided by 2
10: divided by 4
11: divided by 8

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

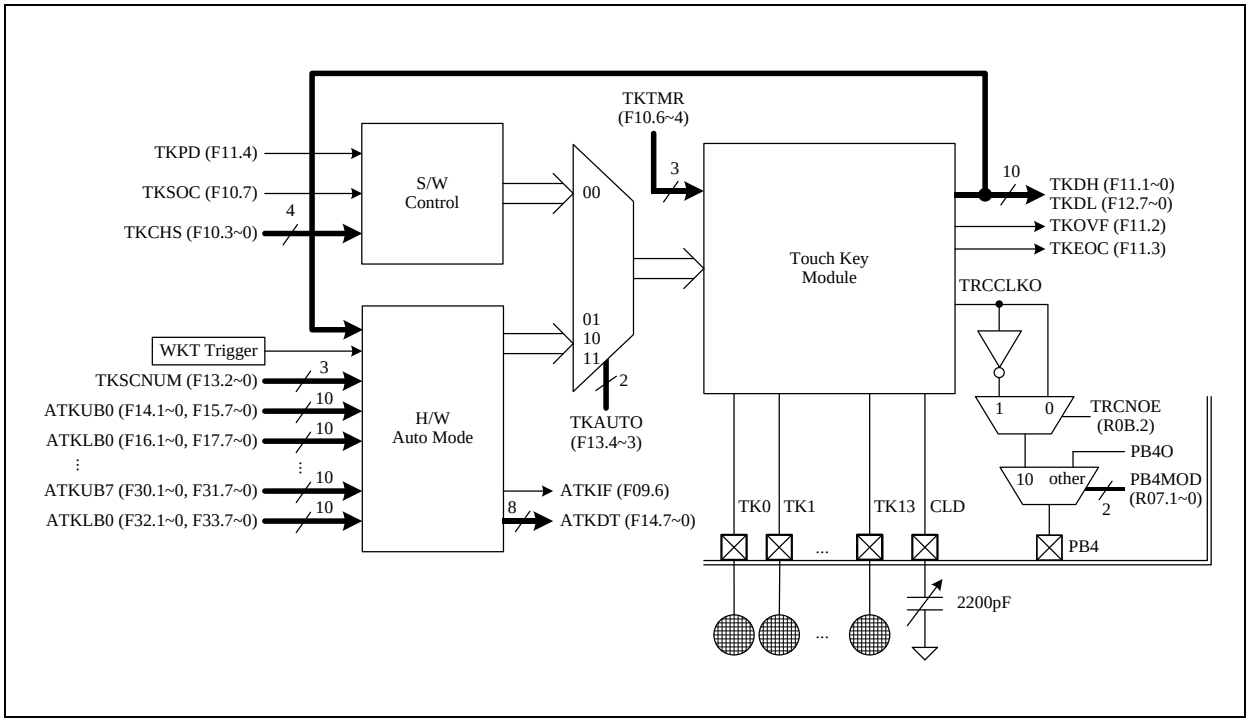
R0D.7~0 **PWM0PRD**: PWM0 period data

### 3.5 Touch Key

The Touch Key offers an easy, simple and reliable method to implement finger touch applications. For most applications, it only requires an external capacitor component on CLD pin. The device support 14 channels touch key detection with S/W manual mode and 8 channels with H/W auto mode (ATK). Only one mode can be active at a time.

To use the Touch Key, user must setup the Pin Mode (***see section 4***) correctly as below table. Setting Mode0 for Touch Key pin can pull up the pin and reduce the Key's mutual interference. While a TK pin is under scanning, TM57MT21will disable the pull up resistor automatically. TM57MT21 can also output the Touch Key clock to PB4 when PB4's Pin Mode PB4MOD (R07.1~0) is set as Mode2. If TRCNOE (R0B.2) is set, the negative Touch Key clock will output to PB4.

| Pin Mode Setting for Touch Key | TK0~7 | TK8~13 |
|---|---|---|
| Pin is not Touch Key | Mode0, 1, 2, 3 | Mode0, 1, 2, 3 |
| Pin is Touch Key, Idling | Mode0 | Mode0 |
| Pin is Touch Key, S/W Scanning | Mode0 | Mode0 |
| Pin is Touch Key, H/W Auto Scan | Mode0 | N/A |




**Touch Key Block Diagram**

### 3.5.1 S/W Manual Mode Touch Key Detection

All Touch Key (TK0~TK13) can be used for S/W mode, it can be select by TKCHS (F10.3~0) bits. To start the S/W mode, user assigns TKAUTO=00b (F13.4~3) and TKPD=0 (F11.4), then set the TKSOC (F10.7) bit to start touch key conversion, the TKSOC bit will be automatically cleared while end of conversion. However, if the system clock is too slow, H/W might lose the auto clear TKSOC capability. "TKEOC=0" means conversion is in process, while "TKEOC=1" means the conversion is finish. After TKEOC's (F11.3) edge rising, user must wait at least 10 μs for next conversion. The touch key counting values is stored into the 10 bits touch key data count "TKDH (F11.1~0), TKDL (F12.7~0)". If TKOVF (F11.2) is set, it means the conversion transaction exceeds period time. Reduce/Increase TKTMR (F10.6~4) can reduce/increase touch key data count to adapt the system board circumstances.

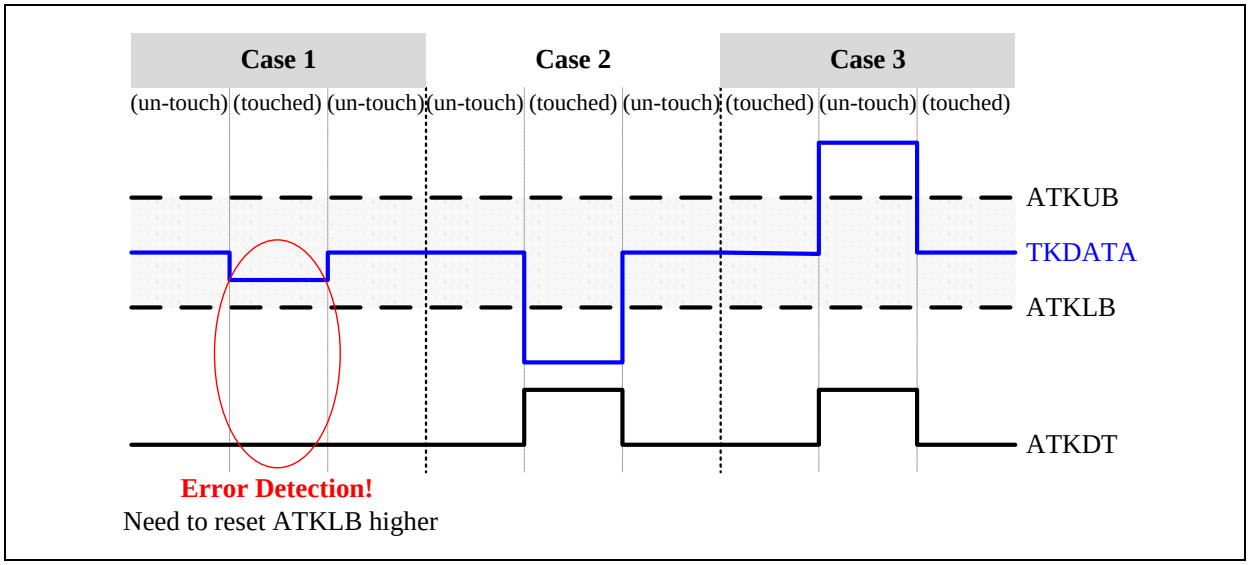
The Touch Key unit has an internal built-in reference capacitor to simulate the KEY behavior. Set TKCHS = 15 and start the S/W scan mode can get the TK Data Count of this reference capacitor. Since the internal capacitor never affected by water or mobile phone, it is useful for comparing the environment background noise.

◇ Example: S/W Mode, Touch key channel=TK5 (PA5).

```
        MOVLW   xxxx00xxB       ; PAMODH[3:2] =00b
        MOVWR   PAMODH          ; Set PA5MOD as Mode0 for touch key input
        BSF     PAD, 5          ; Set PA5 is input with pull-up

        MOVLW   0 100 0101B     ; TKSOC=0
        MOVWF   TKCTL1          ; TKTMR=4, TKCHS=5 (TK5)

        BCF     TKPD            ; TKPD=0
        :
        BSF     TKSOC           ; TKSOC=1, touch key start conversion
        NOP

WAIT_TK:
        BTFSS   TKEOC           ; Polling TKEOC
        GOTO    WAIT_TK         ; Waiting touch key conversion finish

        MOVFW   TKCTL2          ; Read TKDH[1:0]
        MOVFW   TKDL            ; Read TKDL[7:0]
```

### 3.5.2 H/W Auto Mode Touch Key Detection

Only TK0~TK7 are eligible for H/W auto mode by setting TKSCNUM (F13.2~0). This function can work in Idle mode and save the S/W effort as well as minimize the chip current consumption. To use this function, user need to set TKAUTO≠0, TKPD=1 and PB4MOD≠Mode2 (disable Touch Key clock output) to enable H/W fully control the TK unit. If TKAUTO is set to "1", the WKT will generate an overflow flag after WKT time out to trigger the touch key H/W auto mode starting. And then H/W automatically detects the TK0~TK7's TK Data Count at every 30/60/120/240 ms rate by WKTPSC (R0B.1~0). TM57MT21 also offers the multi-time scanning function. If TKAUTO is set to "2", H/W will scan each key 2 times and plus the scanning results together. H/W will scan each key 4 times when TKAUTO is set to "3". It is effective to reduce environment influence. The examples are shown in below. If a keys' TK Data Count is less than the pre-set compare lower boundary (ATKLB0~7) or more than the pre-set compare upper boundary (ATKUB0~7), H/W will record the compare result in the ATKDT (F14.7~0). If ATKDT bits are not equal to "0", H/W will also generate interrupt flag ATKIF (F9.6) after touch key module ends conversion. It generates auto touch key interrupt and wake up CPU if the ATKIE (F8.6) bit is set. User can switch the TK module to S/W Manual Mode after the TK interrupt and identify/confirm the Key touch event.




◇Example: TKAUTO=1 (normal ATK scanning), TKSCNUM=7 (scan 8 keys)

| Scan Sequence: | TK0 | TK1 | TK2 | TK3 | TK4 | TK5 | TK6 | TK7 |
|---|---|---|---|---|---|---|---|---|
| TKDATA: | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits |
| ATKUB/ATKLB: | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits | 10 bits |

◇Example: TKAUTO=2 (2 times scanning), TKSCNUM=4 (scan 5 keys)

| Scan Sequence: | TK0 | TK0 | TK1 | TK1 | TK2 | TK2 | TK3 | TK3 | TK4 | TK4 |
|---|---|---|---|---|---|---|---|---|---|---|
| TKDATA: | (9 bits+9 bits) | | (9 bits+9 bits) | | (9 bits+9 bits) | | (9 bits+9 bits) | | (9 bits+9 bits) | |
| ATKUB/ATKLB: | 10 bits | | 10 bits | | 10 bits | | 10 bits | | 10 bits | |

◇Example: TKAUTO=3 (4 times scanning), TKSCNUM=2 (scan 3 keys)

| Scan Sequence: | TK0 | TK0 | TK0 | TK0 | TK1 | TK1 | TK1 | TK1 | TK2 | TK2 | TK2 | TK2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TKDATA: | (8 bits+8 bits+8 bits+8 bits) | | | | (8 bits+8 bits+8 bits+8 bits) | | | | (8 bits+8 bits+8 bits+8 bits) | | | |

ATKUB/ATKLB: 10 bits 10 bits 10 bits

◇ Example: H/W Auto Mode, Touch key auto scan number is 3 keys, 2 times scanning

```
        ORG     000H            ; Reset Vector
        GOTO    START           ; Goto user program address

        ORG     001H            ; All interrupt vector
        GOTO    INT

        ORG     002H
START:
        MOVLW   xxxxx111B       ; Set PAD[2:0] =111b
        MOVWF   PAD
        MOVLW   xx00 00 00B     ; Set PA2~0 Pin Mode as Mode0
        MOVWR   PAMODL

        MOVLW   xxxxxx00B       ; Make sure PB4 Pin Mode is not Mode2
        MOVWR   PBMODH          ; Disable TK clock output

        MOVLW   0 100xxxxB      ; TKSOC=0, TKTMR=4
        MOVWF   TKCTL1          ; ATK will control the TK channels automatically

        MOVLW   xxx00 010B      ; TKSCNUM=010b (TK auto scan number=3)
        MOVWF   ATKCTL          ; TKAUTO=00b (disable H/W Auto Mode)

        BSF     TKPD            ; TKPD=1

SET_BOUNDARY:
        MOVLW   01H             ; Set TK0 compare upper boundary
        MOVWR   ATKUB0H         ; Set ATKUB0=450
        MOVLW   C2H
        MOVWR   ATKUB0L

        MOVLW   01H             ; Set TK0 compare lower boundary
        MOVWR   ATKLB0H         ; Set ATKLB0=350
        MOVLW   5EH
        MOVWR   ATKLB0L

        MOVLW   01H             ; Set TK1 compare upper boundary
        MOVWR   ATKUB1H         ; Set ATKUB1=380
        MOVLW   7CH
        MOVWR   ATKUB1L

        MOVLW   01H             ; Set TK1 compare lower boundary
        MOVWR   ATKLB1H         ; Set ATKLB1=340
        MOVLW   54H
        MOVWR   ATKLB1L

        MOVLW   03H             ; Set TK2 compare upper boundary
        MOVWR   ATKUB2H         ; Set ATKUB2=1023 (don't care upper boundary)
        MOVLW   FFH
```

```
            MOVWR   ATKUB2L

            MOVLW   01H             ; Set TK2 compare lower boundary
            MOVWR   ATKLB2H         ; Set ATKLB2=392
            MOVLW   88H
            MOVWR   ATKLB2L

            MOVLW   000x0x11B
            MOVWR   MR0B            ; WKTPSC=11b, WKT period=240ms

            MOVLW   10111111B
            MOVWF   INTIF           ; Clear ATK interrupt request flag
            BSF     ATKIE           ; Enable ATK interrupt

            MOVLW   xxx10 010B      ; TKAUTO=10b
            MOVWF   ATKCTL          ; TK H/W auto Mode 2 times scanning
MAIN:
            SLEEP                   ; Set system into IDLE mode
            :
            :
            GOTO    MAIN

INT:
            MOVWF   20H             ; Store W data to SRAM 20H
            MOVFW   STATUS          ; Get STATUS data
            MOVWF   21H             ; Store STATUS data to SRAM 21H

            BTFSC   ATKIF           ; Check ATKIF bit
            GOTO    INT_ATK
            …
EXIT_INT:
            MOVFW   21H             ; Get SRAM 21H data
            MOVWF   STATUS          ; Restore STATUS data
            MOVFW   20H             ; Restore W data
            RETI                    ; Return from interrupt

INT_ATK:
            MOVLW   10111111B
            MOVWF   INTIF           ; Clear ATK interrupt request flag
            MOVFW   ATKDT
            MOVWF   22H             ; Store ATK scan result to SRAM 22H
            GOTO    EXIT_INT
```

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F08.6 **ATKIE**: Auto Touch Key Interrupt Enable
0: disable
1: enable

| F09 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIF | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F09.6 **ATKIF**: Auto Touch Key interrupt event pending flag
This bit is set by H/W while Key's TK Data Count is over the pre-set compare threshold range, write 0 to this bit will clear this flag

| F10 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TKCTL1 | TKSOC | TKTMR | | | TKCHS | | | |
| R/W | R/W | R/W | | | R/W | | | |
| Reset | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |

F10.7 **TKSOC**: Touch key start of conversion, rising edge to start
H/W auto cleared while end of conversion

F10.6~4 **TKTMR**: Touch key conversion time
000: shortest
…
111: longest

F10.3~0 **TKCHS**: Touch key channel select
0000: TK0 (PA0)
0001: TK1 (PA1)
0010: TK2 (PA2)
0011: TK3 (PA3)
0100: TK4 (PA4)
0101: TK5 (PA5)
0110: TK6 (PA6)
0111: TK7 (PD7)
1000: TK8 (PB0)
1001: TK9 (PB1)
1010: TK10 (PB2)
1011: TK11 (PB3)
1100: TK12 (PB4)
1101: TK13 (PB5)
1110: Undefined
1111: Internal reference capacitor

| F11 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TKCTL2 | – | – | – | TKPD | TKEOC | TKOVF | TKDH | |
| R/W | – | – | – | R/W | R | R | R | |
| Reset | – | – | – | 1 | 0 | 0 | 0 | 0 |

F11.4 **TKPD**: Touch key power down
0: Touch key running
1: Touch key power down

F11.3 **TKEOC**: Touch key end of conversion
0: conversion is in process
1: end of conversion

F11.2 **TKOVF**: Touch key counter overflow flag
0: not overflow
1: overflow

F11.1~0 **TKDH**: Touch key data MSB[9:8]

| F12 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TKDL | TKDL | | | | | | | |
| R/W | R | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F12.7~0 **TKDL**: Touch key data LSB[7:0]

| F13 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKCTL | – | – | – | TKAUTO | | TKSCNUM | | |
| R/W | – | – | – | R/W | | R/W | | |
| Reset | – | – | – | 0 | 0 | 1 | 1 | 1 |

F13.4~3 **TKAUTO:** Touch key auto scan mode enable
00: disable H/W Auto Mode
01: enable H/W Auto Mode (1 time scanning)
10: enable H/W Auto Mode (2 times scanning)
11: enable H/W Auto Mode (4 times scanning)

F13.2~0 **TKSCNUM**: Touch key auto scan channel number
000: only scan 1 channel (TK0)
001: scan 2 channels (TK0 ~ TK1)
010: scan 3 channels (TK0 ~ TK2)
011: scan 4 channels (TK0 ~ TK3)
100: scan 5 channels (TK0 ~ TK4)
101: scan 6 channels (TK0 ~ TK5)
110: scan 7 channels (TK0 ~ TK6)
111: scan 8 channels (TK0 ~ TK7)

| F14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKDT | ATKDT | | | | | | | |
| R/W | R | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F14.7~0 **ATKDT**: Touch key auto scan result
xxxx_xxx1: TK0 has a touch event
xxxx_xx1x: TK1 has a touch event
xxxx_x1xx: TK2 has a touch event
xxxx_1xxx: TK3 has a touch event
xxx1_xxxx: TK4 has a touch event
xx1x_xxxx: TK5 has a touch event
x1xx_xxxx: TK6 has a touch event
1xxx_xxxx: TK7 has a touch event

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

R07.1~0 **PB4MOD**: PB4 I/O mode control
00: Mode0
01: Mode1
10: Mode2, Touch Key clock output
11: Mode3

| R0B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MR0B | – | – | – | INT0EDG | – | TRCNOE | WKTPSC | |
| R/W | – | – | – | W | – | W | W | |
| Reset | – | – | – | 0 | – | 0 | 1 | 1 |

R0B.2 **TRCNOE:** Touch Key clock output select
0: positive output
1: negative output

| R14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB0H | – | – | – | – | – | – | ATKUB0H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R14.1~0 **ATKUB0H:** Auto Touch Key TK0 upper boundary MSB[9:8]

| R15 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB0L | ATKUB0L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R15.7~0 **ATKUB0L:** Auto Touch Key TK0 upper boundary LSB[7:0]

| R16 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB0H | – | – | – | – | – | – | ATKLB0H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R16.1~0 **ATKLB0H:** Auto Touch Key TK0 lower boundary MSB[9:8]

| R17 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB0L | ATKLB0L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R17.7~0 **ATKLB0L:** Auto Touch Key TK0 lower boundary LSB[7:0]

| R18 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB1H | – | – | – | – | – | – | ATKUB1H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R18.1~0 **ATKUB1H:** Auto Touch Key TK1 upper boundary MSB[9:8]

| R19 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB1L | ATKUB1L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R19.7~0 **ATKUB1L:** Auto Touch Key TK1 upper boundary LSB[7:0]

| R1A | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB1H | – | – | – | – | – | – | ATKLB1H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R1A.1~0 **ATKLB1H:** Auto Touch Key TK1 lower boundary MSB[9:8]

| R1B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB1L | ATKLB1L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R1B.7~0 **ATKLB1L:** Auto Touch Key TK1 lower boundary LSB[7:0]

| R1C | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB2H | – | – | – | – | – | – | ATKUB2H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R1C.1~0 **ATKUB2H:** Auto Touch Key TK2 upper boundary MSB[9:8]

| R1D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB2L | ATKUB2L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R1D.7~0 **ATKUB2L:** Auto Touch Key TK2 upper boundary LSB[7:0]

| R1E | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB2H | – | – | – | – | – | – | ATKLB2H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R1E.1~0 **ATKLB2H:** Auto Touch Key TK2 lower boundary MSB[9:8]

| R1F | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB2L | ATKLB2L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R1F.7~0 **ATKLB2L:** Auto Touch Key TK2 lower boundary LSB[7:0]

| R20 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB3H | – | – | – | – | – | – | ATKUB3H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R20.1~0 **ATKUB3H:** Auto Touch Key TK3 upper boundary MSB[9:8]

| R21 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB3L | ATKUB3L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R21.7~0 **ATKUB3L:** Auto Touch Key TK3 upper boundary LSB[7:0]

| R22 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB3H | – | – | – | – | – | – | ATKLB3H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R22.1~0 **ATKLB3H:** Auto Touch Key TK3 lower boundary MSB[9:8]

| R23 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB3L | ATKLB3L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R23.7~0 **ATKLB3L:** Auto Touch Key TK3 lower boundary LSB[7:0]

| R24 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB4H | – | – | – | – | – | – | ATKUB4H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R24.1~0 **ATKUB4H:** Auto Touch Key TK4 upper boundary MSB[9:8]

| R25 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB4L | ATKUB4L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R25.7~0 **ATKUB4L:** Auto Touch Key TK4 upper boundary LSB[7:0]

| R26 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB4H | – | – | – | – | – | – | ATKLB4H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R26.1~0 **ATKLB4H:** Auto Touch Key TK4 lower boundary MSB[9:8]

| R27 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB4L | ATKLB4L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R27.7~0 **ATKLB4L:** Auto Touch Key TK4 lower boundary LSB[7:0]

| R28 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB5H | – | – | – | – | – | – | ATKUB5H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R28.1~0 **ATKUB5H:** Auto Touch Key TK5 upper boundary MSB[9:8]

| R29 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB5L | ATKUB5L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R29.7~0 **ATKUB5L:** Auto Touch Key TK5 upper boundary LSB[7:0]

| R2A | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB5H | – | – | – | – | – | – | ATKLB5H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R2A.1~0 **ATKLB5H:** Auto Touch Key TK5 lower boundary MSB[9:8]

| R2B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB5L | ATKLB5L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R2B.7~0 **ATKLB5L:** Auto Touch Key TK5 lower boundary LSB[7:0]

| R2C | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB6H | – | – | – | – | – | – | ATKUB6H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R2C.1~0 **ATKUB6H:** Auto Touch Key TK6 upper boundary MSB[9:8]

| R2D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB6L | ATKUB6L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R2D.7~0 **ATKUB6L:** Auto Touch Key TK6 upper boundary LSB[7:0]

| R2E | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB6H | – | – | – | – | – | – | ATKLB6H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R2E.1~0 **ATKLB6H:** Auto Touch Key TK6 lower boundary MSB[9:8]

| R2F | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB6L | ATKLB6L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R2F.7~0 **ATKLB6L:** Auto Touch Key TK6 lower boundary LSB[7:0]

| R30 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB7H | – | – | – | – | – | – | ATKUB7H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

R30.1~0 **ATKUB7H:** Auto Touch Key TK7 upper boundary MSB[9:8]

| R31 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKUB7L | ATKUB7L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

R31.7~0 **ATKUB7L:** Auto Touch Key TK7 upper boundary LSB[7:0]

| R32 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB7H | – | – | – | – | – | – | ATKLB7H | |
| R/W | – | – | – | – | – | – | W | |
| Reset | – | – | – | – | – | – | 0 | 0 |

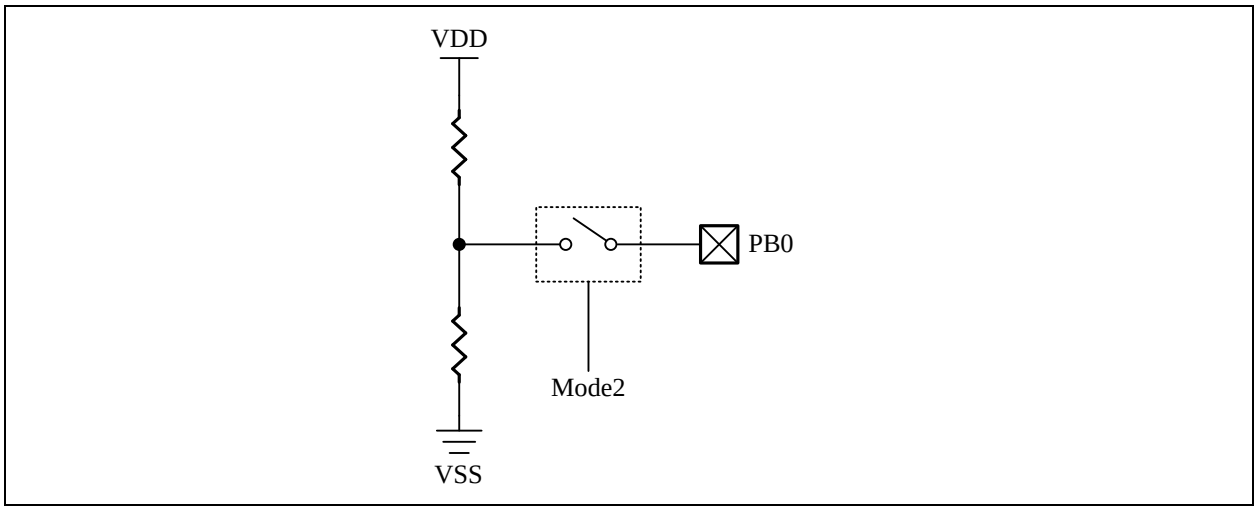
R32.1~0 **ATKLB7H:** Auto Touch Key TK7 lower boundary MSB[9:8]

| R33 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| ATKLB7L | ATKLB7L | | | | | | | |
| R/W | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

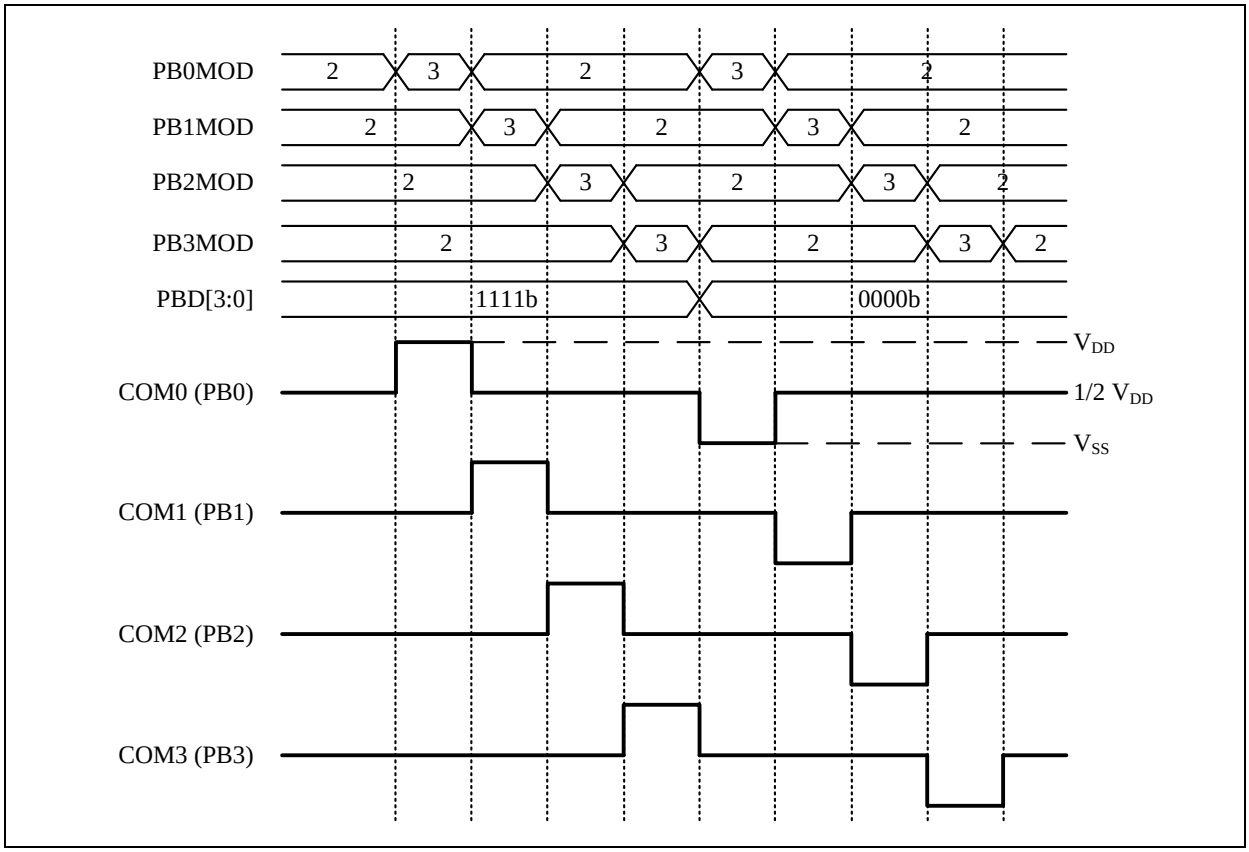
R33.7~0 **ATKLB7L:** Auto Touch Key TK7 lower boundary LSB[7:0]

### 3.6 S/W controlled LCD Driver

The TM57MT21 supports an S/W controlled method to driving LCD. It is capable of driving the LCD panel with 52 dots (Max.) by 4 Commons (COM) and 13 Segments (SEG). The PB0~PB3 are used for Common pins COM0~COM3 and others pins (PA0~PA7, PB4~PB7, PD7) can be used for Segment pins. It is capable of driving 1/2 bias when PB0~PB3's Pin Mode are set to Mode2. Refer to the following figures.

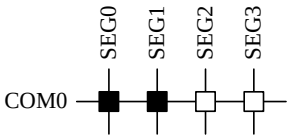


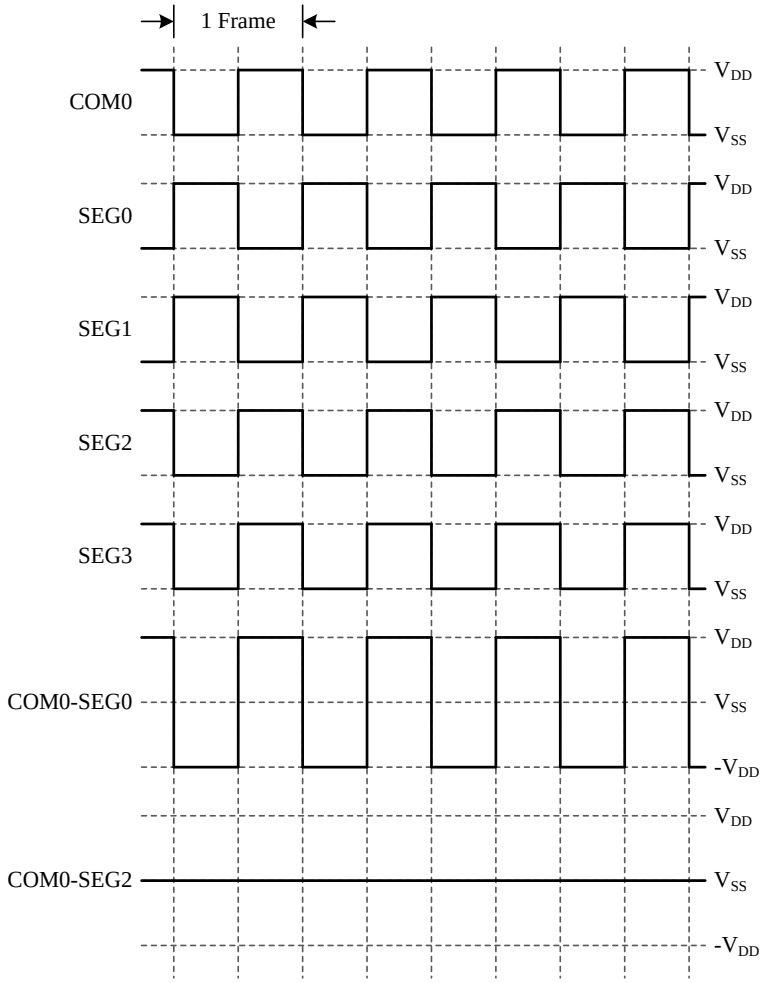

**LCD COM0 Circuit**



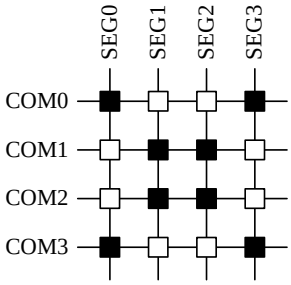

**S/W Controlled LCD COM Scanning**

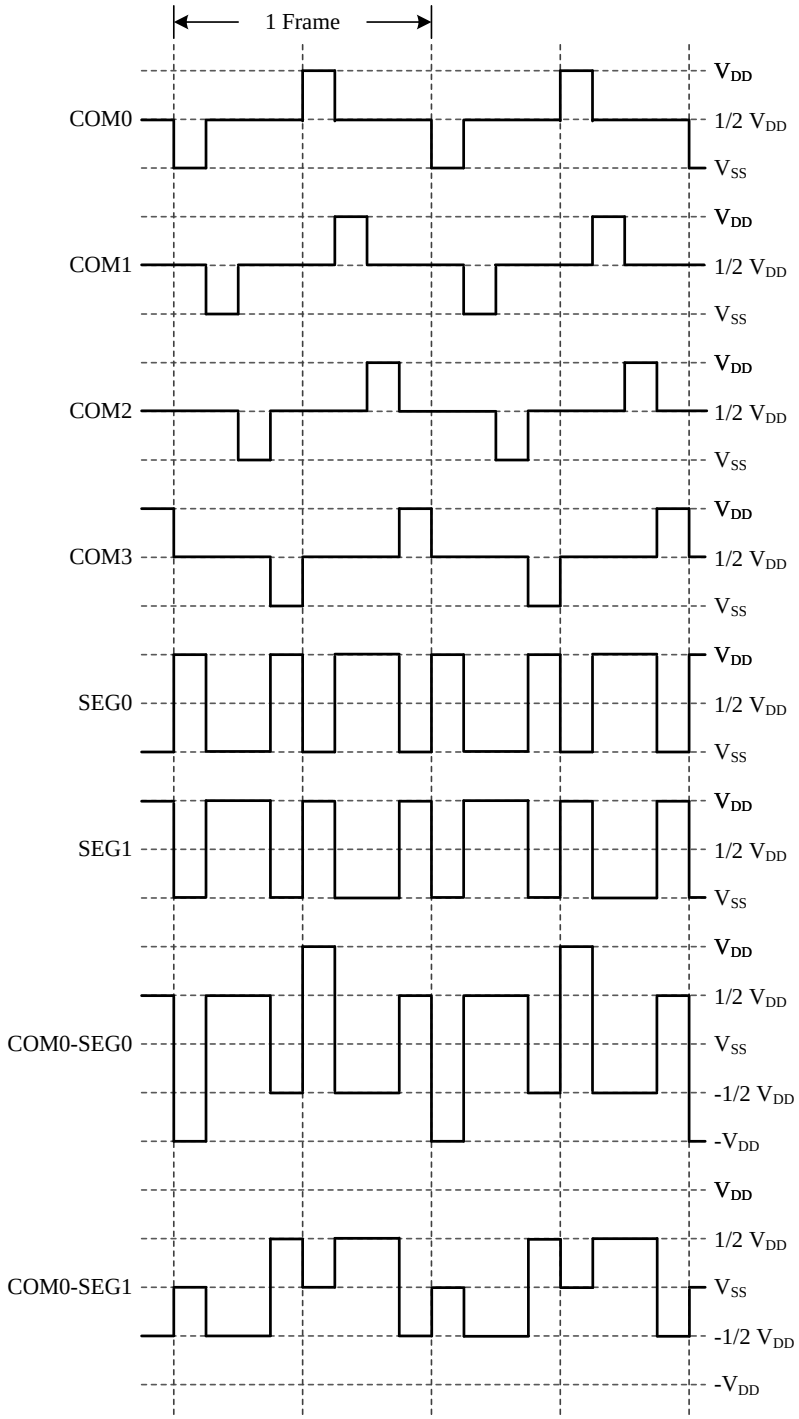
**Static Waveform**



### 1/4 Duty, 1/2 Bias Output Waveform



◇ Example: 1/4 Duty, 1/2 Bias, LCD COM scanning

```
SET_MODE:
        MOVLW   10 10 10 10B
        MOVWR   PBMODL          ; Set PB0 - PB3 as Mode2 (1/2 VDD output)

FRAME_SCAN:
        MOVLW   xxxx1111B
        MOVWF   PBD
COM0_H:                         ;One frame scan start
        MOVLW   10 10 10 11B    ; Set PB0 as Mode3 (CMOS output)
        MOVWR   PBMODL          ; PB1, PB2, PB3 as Mode2
COM1_H:
        MOVLW   10 10 11 10B    ; Set PB1 as Mode3
        MOVWR   PBMODL          ; PB0, PB2, PB3 as Mode2
COM2_H:
        MOVLW   10 11 10 10B    ; Set PB2 as Mode3
        MOVWR   PBMODL          ; PB0, PB1, PB3 as Mode2
COM3_H:
        MOVLW   11 10 10 10B    ; Set PB3 as Mode3
        MOVWR   PBMODL          ; PB0, PB1, PB2 as Mode2

        MOVLW   xxxx0000B
        MOVWF   PBD
COM0_L:
        MOVLW   10 10 10 11B    ; Set PB0 as Mode3
        MOVWR   PBMODL          ; PB1, PB2, PB3 as Mode2
COM1_L:
        MOVLW   10 10 11 10B    ; Set PB1 as Mode3
        MOVWR   PBMODL          ; PB0, PB2, PB3 as Mode2
COM2_L:
        MOVLW   10 11 10 10B    ; Set PB2 as Mode3
        MOVWR   PBMODL          ; PB0, PB1, PB3 as Mode2
COM3_L:
        MOVLW   11 10 10 10B    ; Set PB3 as Mode3
        MOVWR   PBMODL          ; PB0, PB1, PB2 as Mode2
                                ;One frame scan end
```

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

R08.7~6 **PB3MOD**: PB3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

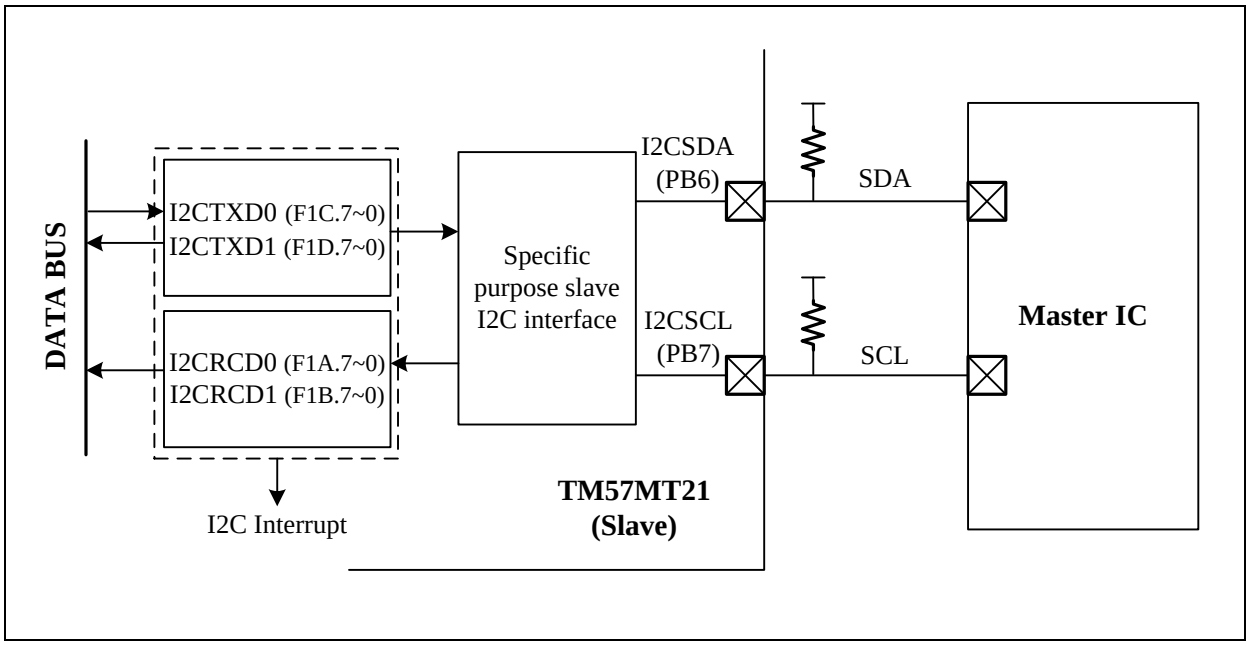
R08.5~4 **PB2MOD**: PB2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

R08.3~2 **PB1MOD**: PB1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

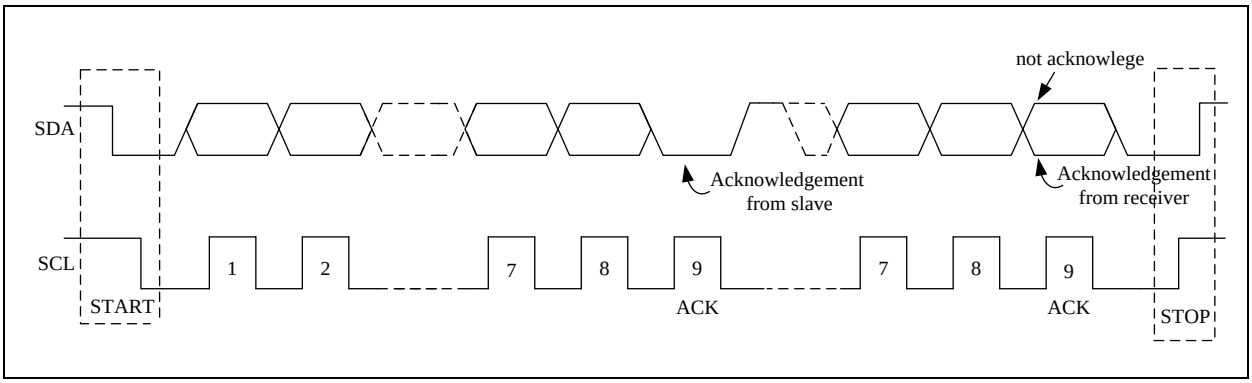
R08.1~0 **PB0MOD**: PB0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

### 3.7 Specific Purpose Slave I2C Interface

Specific purpose slave I2C interface in TM57MT21 could be used for data transmission. This interface is based on a standard I2C (Inter-Integrated Circuit), and TM57MT21 is always as a slave mode. When the master node (another IC or device) sends the correct ID through I2C, it can read data from the register I2CTXD0 (F1C.7~0) and I2CTXD1 (F1D.7~0) of TM57MT21 or write data to the register I2CRCD0 (F1A.7~0) and I2CRCD1 (F1B.7~0) of TM57MT21.

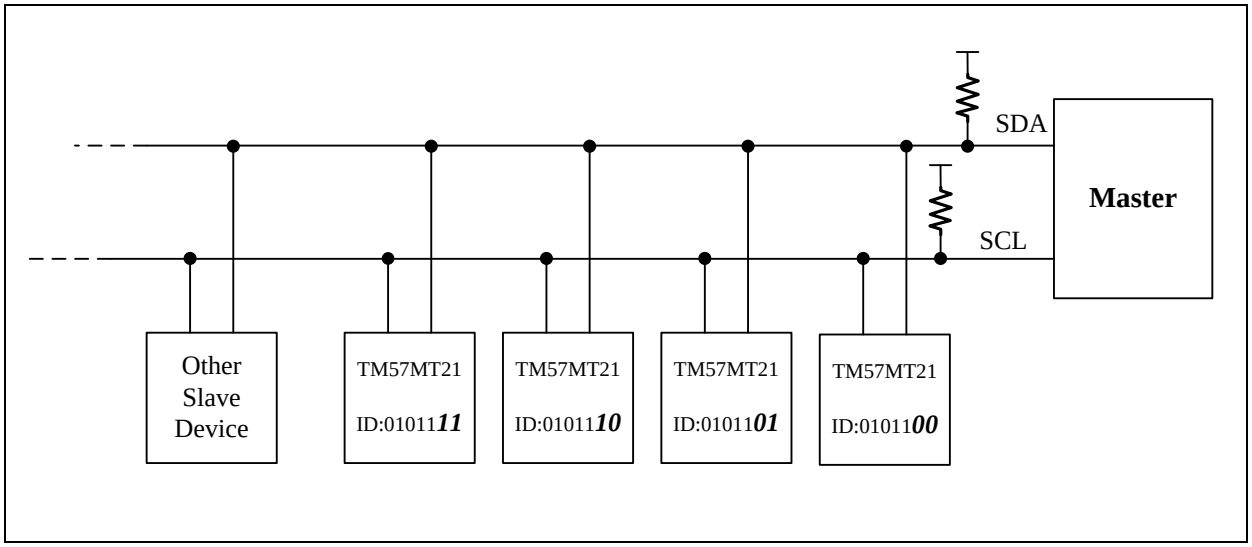



**Slave I2C Interface Block Diagram**

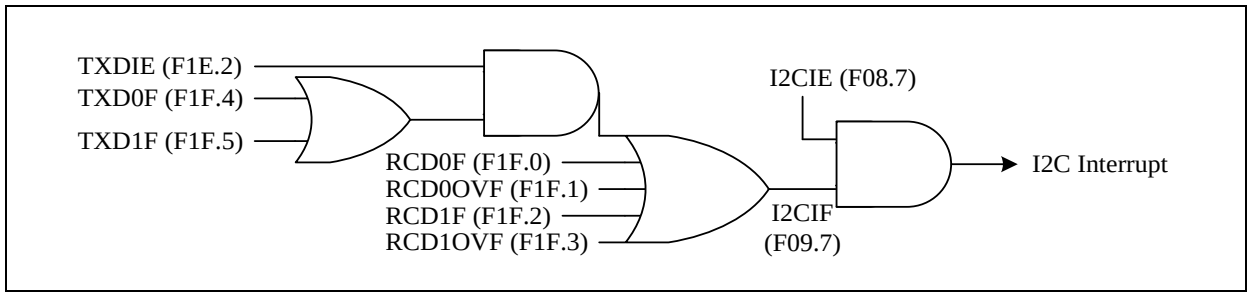



**I2C Protocol**

To use the slave I2C interface, the I2CEN (F1E.3) bit has to be set. TM57MT21 supports 4 slave device IDs by setting I2CID (F1E.1~0). TM57MT21 can generate the transmitting flag TXD0F (F1F.4) and TXD1F (F1F.5) when data transmitting finished. It generates the receiving flag RCD0F (F1F.0) and RCD1F (F1F.2) when data receiving finished. It can also generate the receiving overflow flag RCD0OVF (F1F.1) and RCD1OVF (F1F.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 (F09.7) will be generated. It generates I2C interrupt if the I2CIE (F08.7) bit is set. The transmitting interrupt can be disabled by setting TXDIE (F1E.2). 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 |

| TXDIE | TXDxF | I2CIF | STATE |
|---|---|---|---|
| 0 | x | x | Disable the transmitting interrupt |
| 1 | 0 | 0 | IDLE |
| 1 | 1 | 1 | Data in I2CTXDx is transmitting finish |

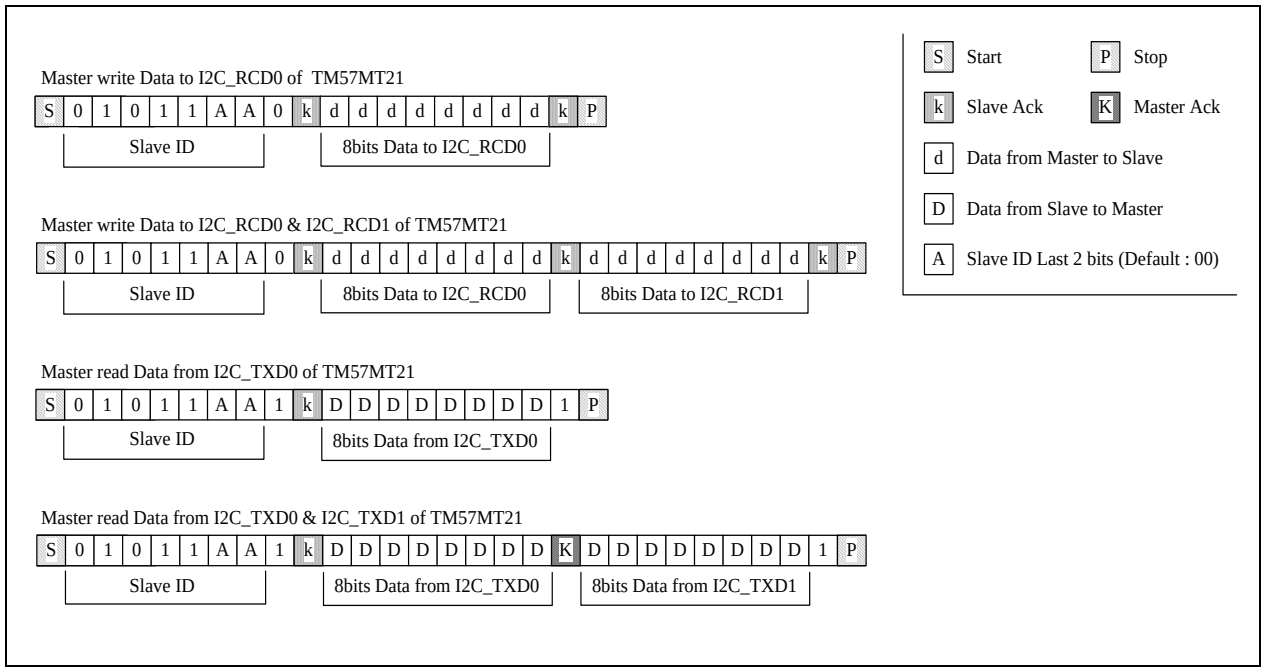



**Table of TM57MT21 I2C Commands**

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F08.7 **I2CIE**: I2C Receive/Transmit Data finished Interrupt Enable
0: disable
1: enable

| F09 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIF | I2CIF | ATKIF | TM1IF | TM0IF | WKTIF | INT2IF | INT1IF | INT0IF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F09.7 **I2CIF**: 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

| F1A | 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 |

F1A.7~0 **I2CRCD0:** The receiving register 0 of slave I2C

| F1B | 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 |

F1B.7~0 **I2CRCD1:** The receiving register 1 of slave I2C

| F1C | 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 |

F1C.7~0 **I2CTXD0:** The transmitting register 0 of slave I2C

| F1D | 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 |

F1D.7~0 **I2CTXD1:** The transmitting register 1 of slave I2C

| F1E | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| I2CCTL | – | – | – | – | I2CEN | TXDIE | I2CID | |
| R/W | – | – | – | – | R/W | R/W | R/W | |
| Reset | – | – | – | – | 0 | 0 | 0 | 0 |

F1E.3 **I2CEN:** Slave I2C interface enable
0: disable
1: enable

F1E.2 **TXDIE**: Slave I2C transmitting interrupt enable
0: disable
1: enable

F1E.1~0 **I2CID**: Slave I2C ID last 2 bits

| F1F | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| I2CFLAG | – | – | TXD1F | TXD0F | RCD1OVF | RCD1F | RCD0OVF | RCD0F |
| R/W | – | – | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | – | – | 0 | 0 | 0 | 0 | 0 | 0 |

F1F.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

F1F.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

F1F.3 **RCD1OVF**: Slave I2C receiving data register 1 overflow
This bit is set by H/W while receiving data to I2CRCD1 overflow, write 0 to this bit will clear this flag

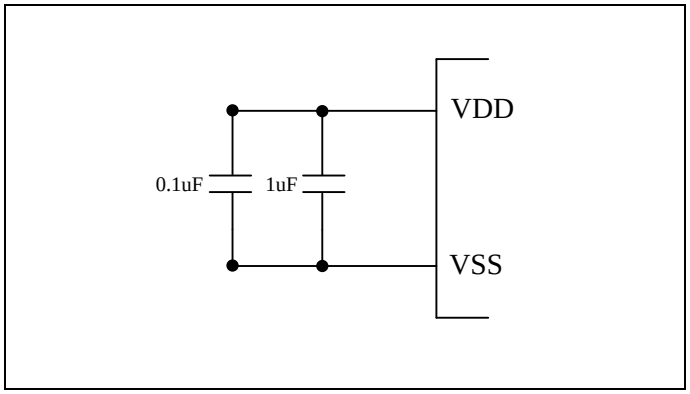
F1F.2 **RCD1F**: Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, write 0 to this bit will clear this flag

F1F.1 **RCD0OVF**: Slave I2C receiving data register 0 overflow
This bit is set by H/W while receiving data to I2CRCD0 overflow, write 0 to this bit will clear this flag

F1F.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

### 3.8 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 8 MHz system clock. For the operation voltage can be protected by LVR, we suggest setting the FIRC/2=4 MHz as maximum system clock. In the Slow Internal RC mode (SIRC), the on-chip oscillator generates 17 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 VDD/VSS pins to improve the stability of clock and the overall system.




**Internal RC Mode**

# 4. I/O Port

## 4.1 PA0-6, PB0-7, PD7

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.

| Mode | PA0-6, PB0-7, PD7 pin function | | PXn SFR data | Pin State | Resistor Pull-up | Digital Input |
|---|---|---|---|---|---|---|
| **Mode 0** | Open Drain | | 0 | Drive Low | N | N |
| | | | 1 | Pull-up | Y | Y |
| **Mode 1** | Open Drain | | 0 | Drive Low | N | N |
| | | | 1 | Hi-Z | N | Y |
| **Mode 2** | Alternative Function | Wake-up | 0 | Drive Low | N | N |
| | | | 1 | Pull-up | Y | Y |
| | | LCD COM | X | – | N | N |
| | | Touch Key Clock output | X | TRCOUT | N | N |
| | | PWM output | 0 | PWM0 | N | N |
| | | | 1 | | by PWM0 | N |
| **Mode 3** | CMOS Output | | 0 | Drive Low | N | N |
| | | | 1 | Drive High | N | N |

**PA0-6, PB0-7, PD7 I/O Pin Function Table**

If a PA0-6, PB0-7, PD7 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 PA0-6, PB0-7, PD7 pin has one or more alternative functions, Wake-up, LCD COM, Touch Key clock output or PWM output. Most of the functions are activated by setting the individual pin mode control SFR to Mode2. Reading the pin data (PA0-6, PB0-7, PD7) 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.

| Pin Name | Wake-up | INT | TK | LCD | others | Mode2 |
|---|---|---|---|---|---|---|
| PA0 | Y | | TK0 | | | Wake-up |
| PA1 | Y | | TK1 | | | Wake-up |
| PA2 | Y | | TK2 | | TM0CKI | Wake-up |
| PA3 | Y | | TK3 | | | Wake-up |
| PA4 | Y | | TK4 | | | Wake-up |
| PA5 | Y | | TK5 | | | Wake-up |
| PA6 | Y | | TK6 | | | Wake-up |
| PB0 | | | TK8 | COM0 | | COM0 |
| PB1 | | | TK9 | COM1 | | COM1 |
| PB2 | | | TK10 | COM2 | | COM2 |
| PB3 | | | TK11 | COM3 | | COM3 |
| PB4 | | INT0 | TK12 | | TRCOUT | TRCOUT |
| PB5 | | | TK13 | | PWM0 | PWM0 |
| PB6 | Y | INT2 | | | I2CSDA | Wake-up |
| PB7 | Y | | | | I2CSCL | Wake-up |
| PD7 | Y | | TK7 | | | Wake-up |

**PA0-6, PB0-7, PD7 multi-function Table**

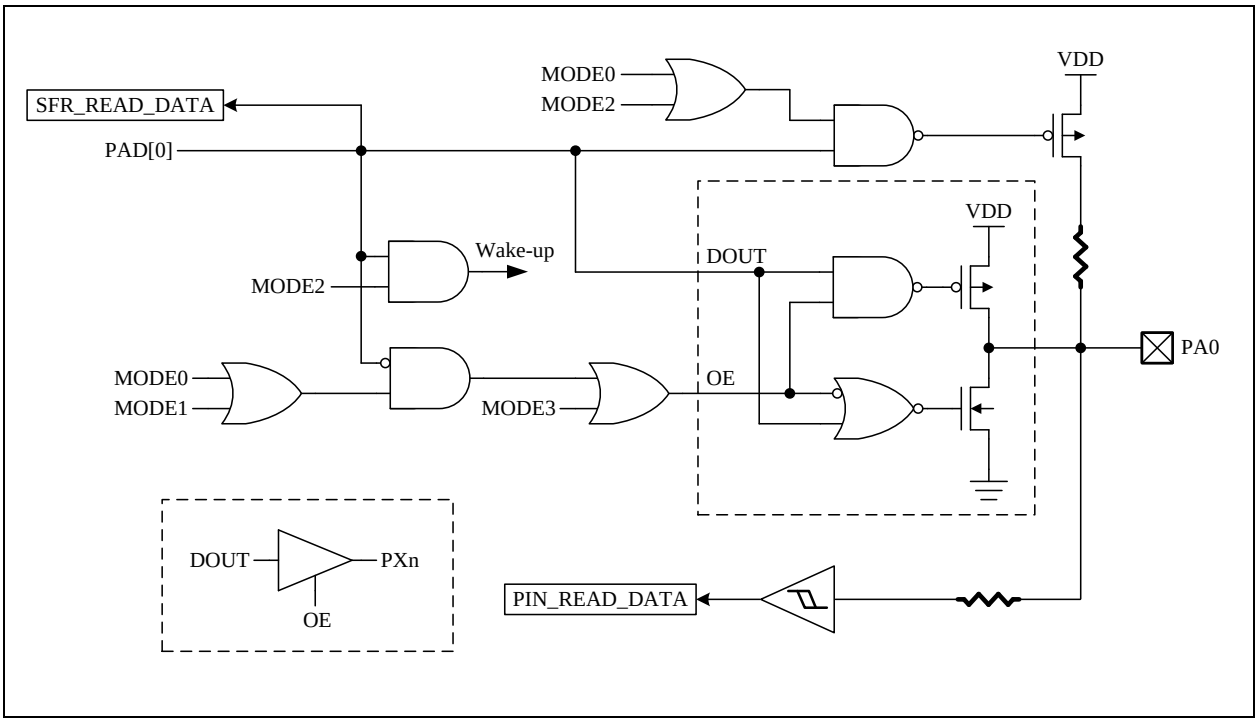
The necessary SFR setting for PA0-6, PB0-7, PD7 pin's alternative function is list below.

| Alternative Function | Mode | PXn SFR data | Pin State | Other necessary SFR setting |
|---|---|---|---|---|
| TM0CKI, INT0, INT2 | **0** | 1 | Input with Pull-up | TM0CTL, INTIE |
| | **1** | 1 | Input | |
| I2CSCL | **0** | 1 | Input with Pull-up | I2CCTL |
| | **1** | 1 | `Input | |
| I2CSDA | **0** | X | Input with Pull-up / Open Drain Output | |
| | **1** | X | Input / Open Drain Output | |
| TRCOUT | **2** | X | Touch Key Clock Output (CMOS Push-Pull) | |
| Wake-up | **2** | 1 | Input with Pull-up | |
| TK0~TK13 | **0** | 1 | Touch Key Idling with Pull-up | TKCTL1 |
| | | | Touch Key Scanning without Pull-up (automatically) | |
| COM0~COM3 | **2** | X | 1/2 $V_{DD}$ output | |
| PWM0 | **2** | 0 | PWM CMOS Push-Pull Output | |
| | | 1 | PWM Open Drain Output | |

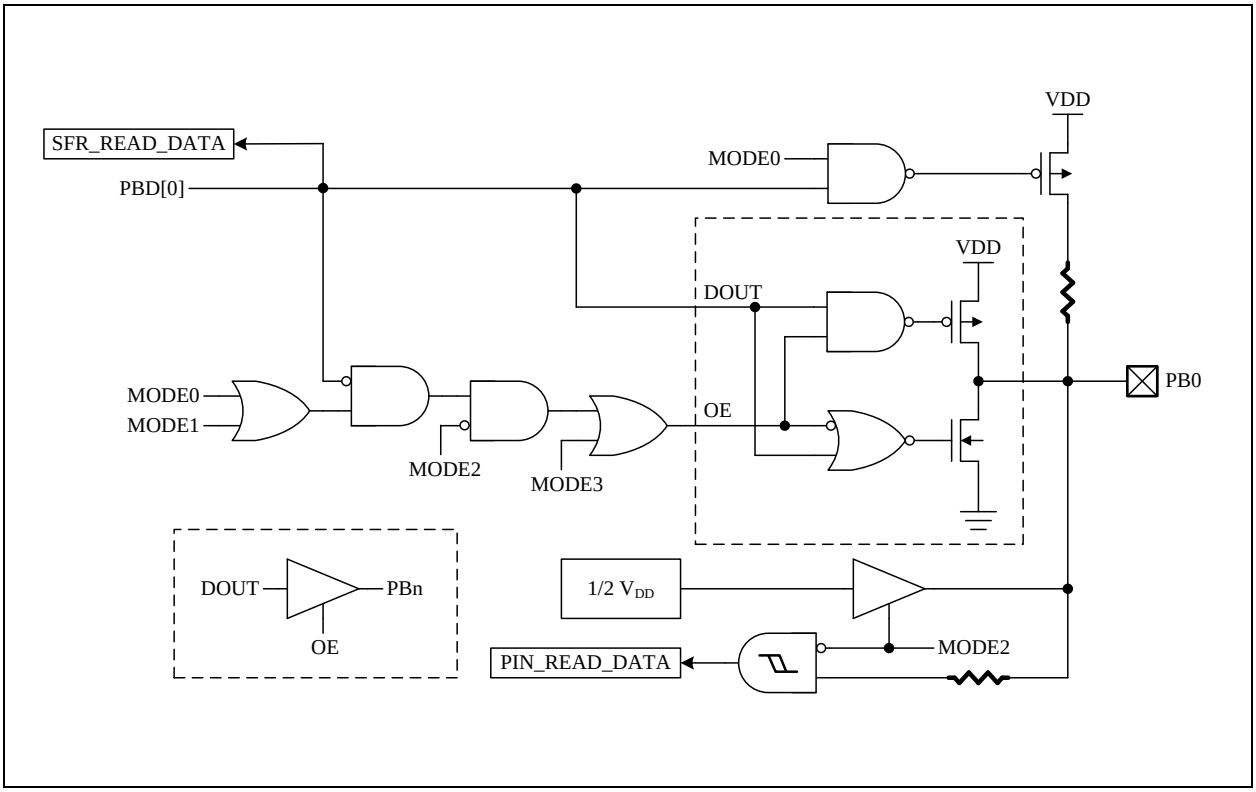
**Mode Setting for PA0-6, PB0-7, PD7 Alternative Function**

For tables above, a **"** CMOS Output " pin means it can sink and drive at least 4mA current. It is not recommended to use such pin as input function.

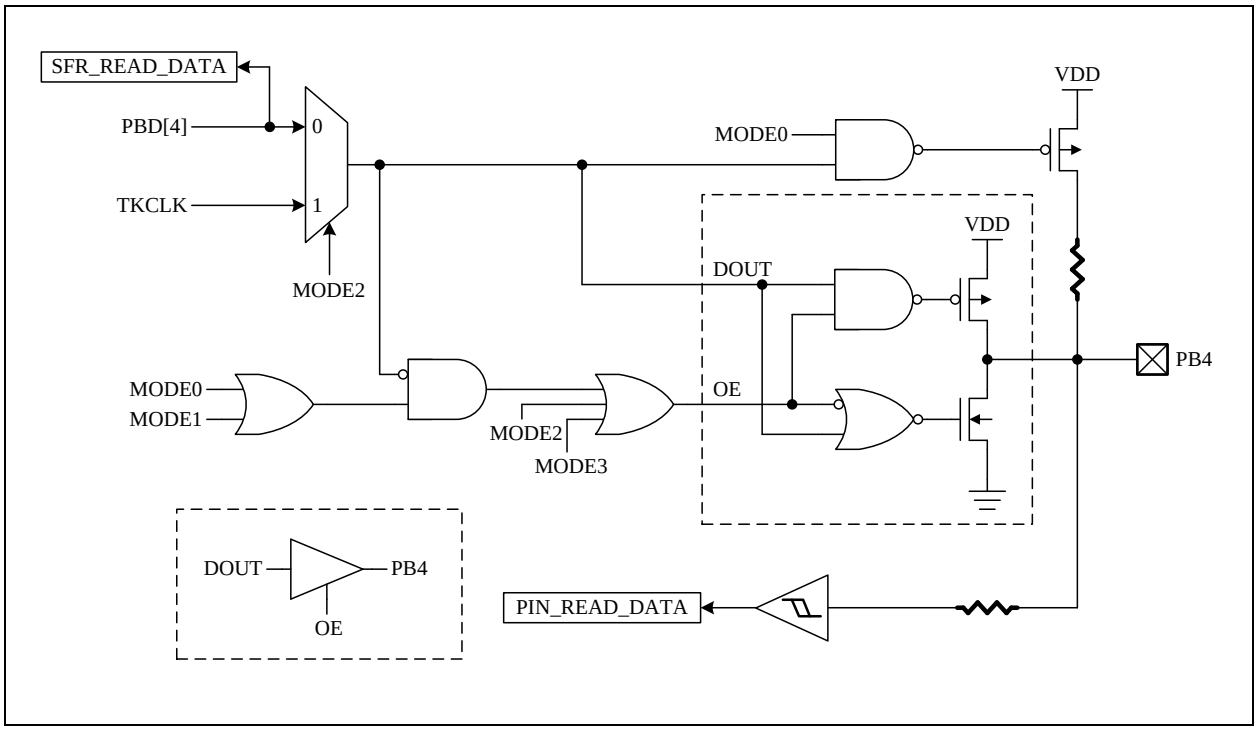
An " Open Drain " pin means it can sink at least 4 mA current but only drive a small current (<20 μA). It can be used as input or output function and typically needs an external pull up resistor.



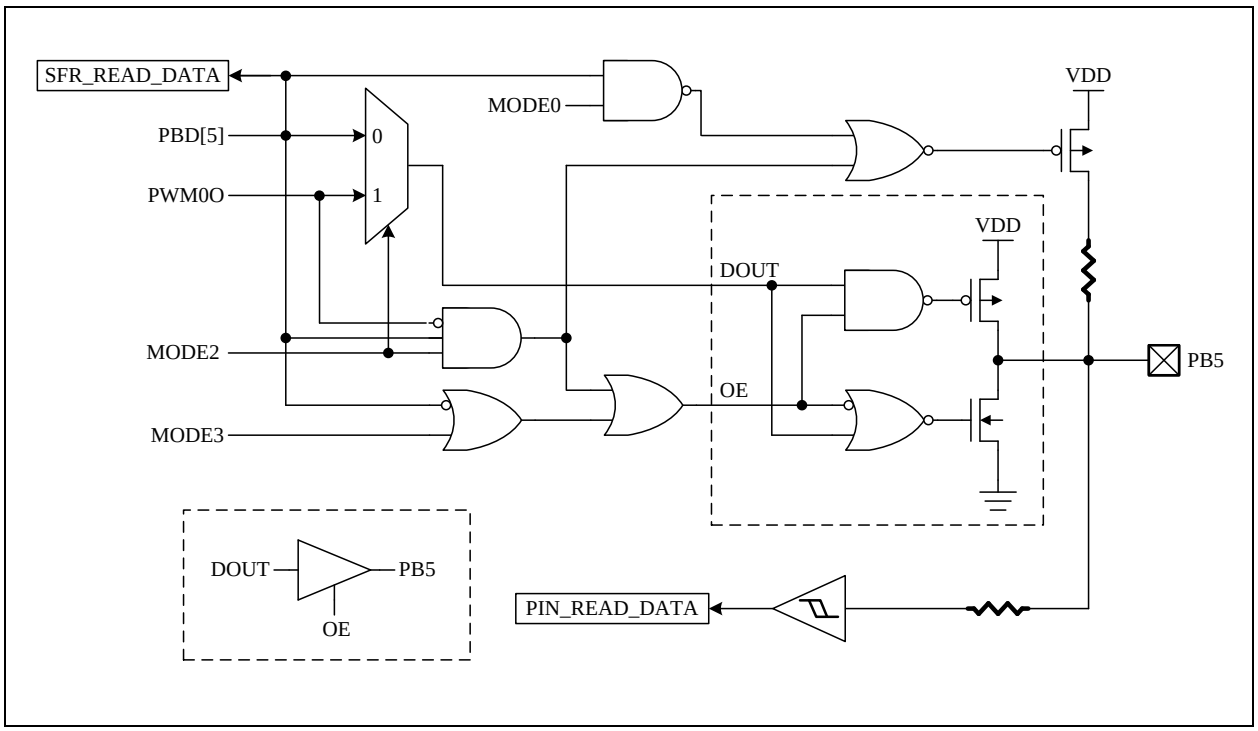

**PA0 Pin Structure**




**PB0 Pin Structure**




**PB4 Pin Structure**




**PB5 Pin Structure**

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

```
MOVLW   xxxxxxx1B
MOVWF   PAD
MOVLW   xxxxxx00B
MOVWR   PAMODL          ; Set PA0 as Schmitt-trigger input with pull-up
```

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

```
MOVLW   xxxxxxx1B
MOVWF   PAD
MOVLW   xxxxxx01B
MOVWR   PAMODL          ; Set PA0 as Schmitt-trigger input without pull-up
```

◇Example: Set PA0 as Schmitt-trigger input with pull-high and low level wake up control (Mode2)

```
MOVLW   xxxxxxx1B
MOVWF   PAD
MOVLW   xxxxxx10B       ; Set PA0 as Schmitt-trigger input with pull-high
MOVWR   PAMODL          ; and enable low level wake up
```

◇Example: Set PA0 as CMOS push-pull output mode and drive Low (Mode3)

```
MOVLW   xxxxxxx0B       ; PAD[0] =0
MOVWF   PAD             ; Set PA0 as CMOS push-pull output Low
MOVLW   xxxxxx11B
MOVWR   PAMODL
```

◇Example: Set PB0-3 as LCD COM with 1/2 $V_{DD}$ bias output (Mode2)

```
MOVLW   xxxxxxxxB       ; PBD[3:0] don't care
MOVWF   PBD
MOVLW   10 10 10 10B
MOVWR   PBMODL          ; Set PB0-3 as LCD COM
```

◇Example: Set PB5 as PWM0 open-drain output mode (Mode2)

```
MOVLW   xx0xxxxxB       ; PBD[5] =0
MOVWF   PBD             ; Set PB5 as PWM0 open-drain output mode
MOVLW   xxxx10xxB
MOVWR   PBMODH
```

| 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 | 1 | 1 |

F05.7~0 **PAD:** PA7~PA0 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:** PB7~PB0 data

| F07 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PDD | PDD7 | – | – | – | – | – | – | – |
| R/W | R/W | – | – | – | – | – | – | – |
| Reset | 1 | – | – | – | – | – | – | – |

F07.7 **PDD7:** PD7 data

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | I2CIE | ATKIE | TM1IE | TM0IE | WKTIE | INT2IE | INT1IE | INT0IE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F08.2 **INT2IE**: INT2 (PB6) pin interrupt enable
0: disable
1: enable

F08.0 **INT0IE**: INT0 (PB4) pin interrupt enable
0: disable
1: enable

| F10 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| TKCTL1 | TKSOC | TKTMR | | | TKCHS | | | |
| R/W | R/W | R/W | | | R/W | | | |
| Reset | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |

F10.3~0 **TKCHS**: Touch key channel select
0000: TK0 (PA0)
0001: TK1 (PA1)
0010: TK2 (PA2)
0011: TK3 (PA3)
0100: TK4 (PA4)
0101: TK5 (PA5)
0110: TK6 (PA6)
0111: TK7 (PD7)
1000: TK8 (PB0)
1001: TK9 (PB1)
1010: TK10 (PB2)
1011: TK11 (PB3)
1100: TK12 (PB4)
1101: TK13 (PB5)
1110: Undefined
1111: Internal reference capacitor

| F1E | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| I2CCTL | – | – | – | – | I2CEN | TXDIE | I2CID | |
| R/W | – | – | – | – | R/W | R/W | R/W | |
| Reset | – | – | – | – | 0 | 0 | 0 | 0 |

F1E.3 **I2CEN:** Slave I2C interface enable
0: disable
1: enable

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

R02.4 **TM0CKS:** Timer0 clock source select
0: Instruction Cycle (Fsys/2) as Timer0 prescaler clock
1: TM0CKI (PA2) as Timer0 prescaler clock

| R05 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PAMODH | – | – | PA6MOD | | PA5MOD | | PA4MOD | |
| R/W | – | – | W | | W | | W | |
| Reset | – | – | 0 | 1 | 0 | 1 | 0 | 1 |

R05.5~4 **PA6MOD**: PA6 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R05.3~2 **PA5MOD**: PA5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R05.1~0 **PA4MOD**: PA4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
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 | W | | W | | W | | W | |
| Reset | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

R06.7~6 **PA3MOD**: PA3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R06.5~4 **PA2MOD**: PA2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R06.3~2 **PA1MOD**: PA1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R06.1~0 **PA0MOD**: PA0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
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 | W | | W | | W | | W | |
| Reset | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

R07.7~6 **PB7MOD**: PB7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R07.5~4 **PB6MOD**: PB6 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

R07.3~2 **PB5MOD**: PB5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, PWM0 Output (PBD[5] =1, Open Drain output; PBD[5] =0, CMOS push-pull output)
11: Mode3

R07.1~0 **PB4MOD**: PB4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, Touch key clock output (CMOS push-pull output)
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 | W | | W | | W | | W | |
| Reset | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

R08.7~6 **PB3MOD**: PB3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

R08.5~4 **PB2MOD**: PB2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

R08.3~2 **PB1MOD**: PB1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

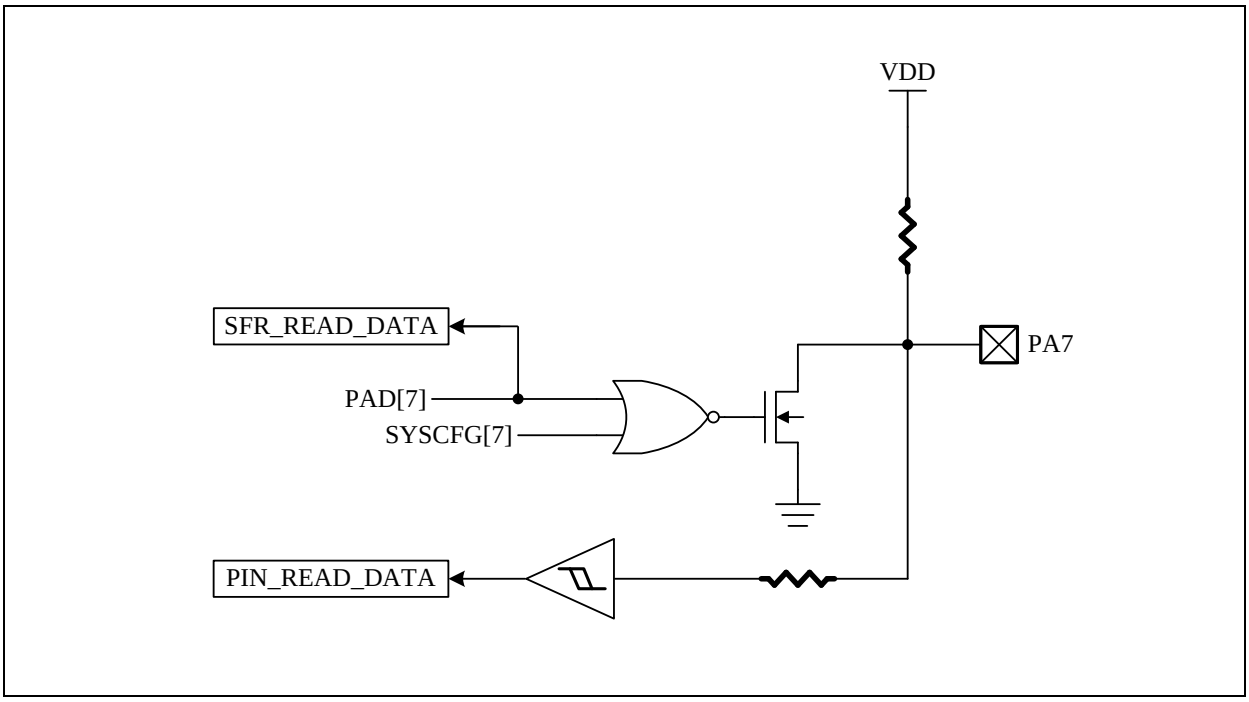
R08.1~0 **PB0MOD**: PB0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, LCD COM 1/2 $V_{DD}$ output
11: Mode3

| R0A | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PDMOD | PD7MOD | | – | – | – | – | – | – |
| R/W | W | | – | – | – | – | – | – |
| Reset | 0 | 1 | – | – | – | – | – | – |

R0A.7~6 **PD7MOD**: PD7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2, input with pull-up/wake-up
11: Mode3

### 4.2 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 always connected to this pin. When SYSCFG[7] is set, PA7 is only used in Schmitt-trigger input for external active low reset.




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

| SYSCFG[7] | PAD[7] | PIN STATE | Pull-up | Mode |
|---|---|---|---|---|
| 0 | 0 | Low | No | Open-drain output low |
| | 1 | High | Yes | Input with pull-up |
| 1 | x | High | Yes | Reset input with pull-up |

# MEMORY MAP

## F-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F00) INDF** | | | | **Function related to: RAM W/R** |
| INDF | 00.7~0 | R/W | - | Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register |
| **(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: RAM W/R/Table Read** |
| 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 | R | - | PA7 pin or “data register” state |
| | | W | 1 | 0: PA7 is open-drain output mode<br>1: PA7 is Schmitt-trigger input mode |
| | 05.6~0 | R | - | Port A pin or “data register” state |
| | | W | 7F | 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** |
| PDD7 | 07.7 | R | - | Port D pin or “data register” state |
| | | W | 1 | Port D output data register |
| **(F08) INTIE** | | | | **Function related to: Interrupt Enable** |
| I2CIE | 08.7 | R/W | 0 | I2C receive/transmit data finished interrupt enable<br>0: disable<br>1: enable |
| ATKIE | 08.6 | R/W | 0 | Auto Touch Key interrupt enable<br>0: disable<br>1: enable |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| TM1IE | 08.5 | R/W | 0 | Timer1 interrupt enable<br>0: disable<br>1: enable |
| TM0IE | 08.4 | R/W | 0 | Timer0 interrupt enable<br>0: disable<br>1: enable |
| WKTIE | 08.3 | R/W | 0 | Wakeup Timer interrupt enable<br>0: disable<br>1: enable |
| INT2IE | 08.2 | R/W | 0 | INT2 (PB6) pin interrupt enable<br>0: disable<br>1: enable |
| INT1IE | 08.1 | R/W | 0 | INT1 (PA7) pin interrupt enable<br>0: disable<br>1: enable |
| INT0IE | 08.0 | R/W | 0 | INT0 (PB4) pin interrupt enable<br>0: disable<br>1: enable |
| **(F09) INTIF** | | | | **Function related to: Interrupt Flag** |
| I2CIF | 09.7 | R | - | I2C interrupt event pending flag, set by H/W while<br>a. I2CRCD0 or I2CRCD1 receive new data finished<br>b. I2CRCD0 or I2CRCD1 data overflow occurred<br>c. I2CTXD0 or I2CTXD1 data transmit finished |
| | | W | 0 | 0: clear this flag and slave I2C related flags<br>1: no action |
| ATKIF | 09.6 | R | - | Auto Touch Key interrupt event pending flag, set by H/W while Key's TK Data Count is over the pre-set compare threshold range |
| | | W | 0 | 0: clear this flag<br>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<br>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<br>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<br>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<br>1: no action |
| INT1IF | 09.1 | R | - | INT1 interrupt event pending flag, set by H/W at INT1 pin's falling edge |
| | | W | 0 | 0: clear this flag<br>1: no action |
| INT0IF | 09.0 | R | - | INT0 interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge |
| | | W | 0 | 0: clear this flag<br>1: no action |
| **(F0A) TM1** | | | | **Function related to: Timer1** |
| TM1 | 0a.7~0 | R/W | 0 | Timer1 content |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F0C) PWM0D** | | | | **Function related to: PWM0** |
| PWM0D | 0c.7~0 | R/W | 0 | PWM0 duty |
| **(F0D) MF0D** | | | | **Function related to: VDDFLT/EMI Improve/PWM0/Timer1/Timer0** |
| VDDFLT | 0d.4 | R/W | 0 | Power noise filter<br>0: disable<br>1: enable |
| EMIIMPV | 0d.3 | R/W | 1 | EMI improve<br>0: disable<br>1: enable |
| PWM0CLR | 0d.2 | R/W | 1 | PWM0 clear and hold<br>0: PWM0 is running<br>1: PWM0 is clear and hold |
| TM1STP | 0d.1 | R/W | 0 | Timer1 counter stop<br>0: Timer1 is counting<br>1: Timer1 stops counting |
| TM0STP | 0d.0 | R/W | 0 | Timer0 counter stop<br>0: Timer0 is counting<br>1: Timer0 stops counting |
| **(F0E) IRCF** | | | | **Function relate to: Trim FIRC** |
| IRCF | 0e.5~0 | R/W | XX | FIRC frequency adjustment |
| **(F0F) CLKCTL** | | | | **Function related to: System Clock** |
| SLOWSTP | 0f.4 | R/W | 0 | Slow-clock stop<br>0: Slow-clock is running<br>1: Slow-clock stops running in Power-down mode |
| FASTSTP | 0f.3 | R/W | 0 | Fast-clock stop<br>0: Fast-clock is running<br>1: Fast-clock stops running |
| CPUCKS | 0f.2 | R/W | 0 | System clock source select<br>0: Slow-clock<br>1: Fast-clock (forbid using, when CPUPSC=3) |
| CPUPSC | 0f.1~0 | R/W | 3 | System clock source prescaler. System clock source<br>00: divided by 8<br>01: divided by 4<br>10: divided by 2<br>11: divided by 1 (forbid using, when CPUCKS=1) |
| **(F10) TKCTL1** | | | | **Function related to: Touch Key** |
| TKSOC | 10.7 | R/W | 0 | Touch key start of conversion, rising edge to start H/W auto cleared while end of conversion |
| TKTMR | 10.6~4 | R/W | 4 | Touch key conversion time<br>000: shortest<br>…<br>111: longest |
| TKCHS | 10.3~0 | R/W | F | Touch key channel select<br>0000: TK0 (PA0) 1000: TK8 (PB0)<br>0001: TK1 (PA1) 1001: TK9 (PB1)<br>0010: TK2 (PA2) 1010: TK10 (PB2)<br>0011: TK3 (PA3) 1011: TK11 (PB3)<br>0100: TK4 (PA4) 1100: TK12 (PB4)<br>0101: TK5 (PA5) 1101: TK13 (PB5)<br>0110: TK6 (PA6) 1110: Undefined<br>0111: TK7 (PD7) 1111: Internal reference capacitor |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F11) TKCTL2** | | | | **Function related to: Touch Key** |
| TKPD | 11.4 | R/W | 1 | Touch Key Power Down<br>0: Touch key running<br>1: Touch key power down |
| TKEOC | 11.3 | R | 1 | Touch key end of conversion<br>0: conversion is in process<br>1: end of conversion |
| TKOVF | 11.2 | R | 0 | Touch key counter overflow flag<br>0: not overflow<br>1: overflow |
| TKDH | 11.1~0 | R | 0 | Touch key data MSB [9~8] |
| **(F12) TKDL** | | | | **Function related to: Touch Key** |
| TKDL | 12.7~0 | R | 0 | Touch key data LSB [7~0] |
| **(F13) ATKCTL** | | | | **Function related to: Touch Key** |
| TKAUTO | 13.4~3 | R/W | 0 | Touch key auto scan mode enable<br>00: disable H/W Auto Mode<br>01: enable H/W Auto Mode (1 time scanning)<br>10: enable H/W Auto Mode (2 times scanning)<br>11: enable H/W Auto Mode (4 times scanning) |
| TKSCNUM | 13.2~0 | R/W | 7 | Touch key auto scan channel number<br>000: only scan 1 channel (TK0)<br>001: scan 2 channels (TK0~TK1)<br>010: scan 3 channels (TK0~TK2)<br>011: scan 4 channels (TK0~TK3)<br>100: scan 5 channels (TK0~TK4)<br>101: scan 6 channels (TK0~TK5)<br>110: scan 7 channels (TK0~TK6)<br>111: scan 8 channels (TK0~TK7) |
| **(F14) ATKDT** | | | | **Function related to: Touch Key** |
| ATKDT | 14.7~0 | R | 0 | Touch key auto scan result<br>xxxx_xxx1: TK0 has a touch event<br>xxxx_xx1x: TK1 has a touch event<br>xxxx_x1xx: TK2 has a touch event<br>xxxx_1xxx: TK3 has a touch event<br>xxx1_xxxx: TK4 has a touch event<br>xx1x_xxxx: TK5 has a touch event<br>x1xx_xxxx: TK6 has a touch event<br>1xxx_xxxx: TK7 has a touch event |
| **(F1A) I2CRCD0** | | | | **Function related to: Slave I2C** |
| I2CRCD0 | 1a.7~0 | R | 0 | The receiving register 0 of slave I2C |
| **(F1B) I2CRCD1** | | | | **Function related to: Slave I2C** |
| I2CRCD1 | 1b.7~0 | R | 0 | The receiving register 1 of slave I2C |
| **(F1C) I2CTXD0** | | | | **Function related to: Slave I2C** |
| I2CTXD0 | 1c.7~0 | R/W | 0 | The transmitting register 0 of slave I2C |
| **(F1D) I2CTXD1** | | | | **Function related to: Slave I2C** |
| I2CTXD1 | 1d.7~0 | R/W | 0 | The transmitting register 1 of slave I2C |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F1E) I2CCTL** | | | | **Function related to: Slave I2C** |
| I2CEN | 1e.3 | R/W | 0 | Slave I2C interface enable<br>0: disable<br>1: enable |
| TXDIE | 1e.2 | R/W | 0 | Slave I2C transmitting interrupt enable<br>0: disable<br>1: enable |
| I2CID | 1e.1~0 | R/W | 0 | Slave I2C ID last 2 bits |
| **(F1F) I2CFLAG** | | | | **Function related to: Slave I2C** |
| TXD1F | 1f.5 | R | - | Slave I2C transmitting data register 1 flag, set by H/W while I2CTXD1 data transmitting finished |
| | | W | 0 | 0: clear this flag<br>1: no action |
| TXD0F | 1f.4 | R | - | Slave I2C transmitting data register 0 flag, set by H/W while I2CTXD0 data transmitting finished |
| | | W | 0 | 0: clear this flag<br>1: no action |
| RCD1OVF | 1f.3 | R | - | Slave I2C receiving data register 1 overflow, set by H/W while receiving data to I2CRCD1 overflow |
| | | W | 0 | 0: clear this flag<br>1: no action |
| RCD1F | 1f.2 | R | - | Slave I2C receiving data register 1 flag, set by H/W while data receiving to I2CRCD1 finished |
| | | W | 0 | 0: clear this flag<br>1: no action |
| RCD0OVF | 1f.1 | R | - | Slave I2C receiving data register 0 overflow, set by H/W while receiving data to I2CRCD0 overflow |
| | | W | 0 | 0: clear this flag<br>1: no action |
| RCD0F | 1f.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<br>1: no action |
| **User Data Memory** | | | | |
| SRAM | 20~27 | R/W | - | SRAM common area (8 bytes) |
| | 28~7f | R/W | - | SRAM Bank0 area (RAMBK=0, 88 bytes) |
| | 28~7f | R/W | - | SRAM Bank1 area (RAMBK=1, 88 bytes) |

## R-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R01) TM0RLD** | | | | **Function related to: Timer0** |
| TM0RLD | 01.7~0 | W | 0 | Timer0 reload Data |
| **(R02) TM0CTL** | | | | **Function related to: Timer0** |
| TM0EDG | 02.5 | W | 0 | TM0CKI (PA2) edge selection for Timer0 prescaler count<br>0: TM0CKI rising edge for Timer0 prescaler count<br>1: TM0CKI falling edge for Timer0 prescaler count |
| TM0CKS | 02.4 | W | 0 | Timer0 clock source select<br>0: Instruction Cycle (Fsys/2) as Timer0 prescaler clock<br>1: TM0CKI (PA2) as Timer0 prescaler clock |
| TM0PSC | 02.3~0 | W | 0 | Timer0 prescaler. Timer0 clock source<br>0000: divided by 1<br>0001: divided by 2<br>0010: divided by 4<br>0011: divided by 8<br>0100: divided by 16<br>0101: divided by 32<br>0110: divided by 64<br>0111: divided by 128<br>1xxx: divided by 256 |
| **(R03) PWRDN** | | | | **Function related to: POWER DOWN** |
| PWRDN | 03 | W | - | Write this register to enter Power-down (STOP/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** |
| PA6MOD | 05.5~4 | W | 1 | PA6~PA4 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, input with pull-up/wake-up<br>11: Mode3 |
| PA5MOD | 05.3~2 | W | 1 | |
| PA4MOD | 05.1~0 | W | 1 | |
| **(R06) PAMODL** | | | | **Function related to: Port A** |
| PA3MOD | 06.7~6 | W | 1 | PA3~PA0 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, input with pull-up/wake-up<br>11: Mode3 |
| PA2MOD | 06.5~4 | W | 1 | |
| PA1MOD | 06.3~2 | W | 1 | |
| PA0MOD | 06.1~0 | W | 1 | |
| **(R07) PBMODH** | | | | **Function related to: Port B** |
| PB7MOD | 07.7~6 | W | 1 | PB7~PB6 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, input with pull-up/wake-up<br>11: Mode3 |
| PB6MOD | 07.5~4 | W | 1 | |
| PB5MOD | 07.3~2 | W | 1 | PB5 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, PWM0 Output<br>(PBD[5] =1, Open Drain output; PBD[5] =0, CMOS output)<br>11: Mode3 |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| PB4MOD | 07.1~0 | W | 1 | PB4 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, Touch key clock output<br>11: Mode3 |
| **(R08) PBMODL** | | | | **Function related to: Port B** |
| PB3MOD | 08.7~6 | W | 1 | PB3~PB0 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, LCD COM 1/2 $V_{DD}$ output<br>11: Mode3 |
| PB2MOD | 08.5~4 | W | 1 | |
| PB1MOD | 08.3~2 | W | 1 | |
| PB0MOD | 08.1~0 | W | 1 | |
| **(R09) PDMOD** | | | | **Function related to: Port D** |
| PD7MOD | 09.7~6 | W | 1 | PD7 I/O mode control<br>00: Mode0<br>01: Mode1<br>10: Mode2, input with pull-up / wake-up<br>11: Mode3 |
| **(R0B) MR0B** | | | | **Function related to: INT0/TRCOUT/WDT** |
| INT0EDG | 0b.4 | W | 0 | INT0 pin (PB4) edge interrupt event<br>0: falling edge to trigger<br>1: rising edge to trigger |
| TRCNOE | 0b.2 | W | 0 | Touch key clock output select<br>0: positive output<br>1: negative output |
| WKTPSC | 0b.1~0 | W | 3 | WDT / WKT pre-scale selections:<br>00: 30mS<br>01: 60mS<br>10: 120mS<br>11: 240mS |
| **(R0C) PWM0CTL** | | | | **Function related to: PWM0** |
| PWM0NOE | 0c.2 | W | 0 | PWM0 output select<br>0: positive output<br>1: negative output |
| PWM0PSC | 0c.1~0 | W | 0 | PWM0 prescaler, PWM0 clock source (Fsys)<br>00: divided by 1<br>01: divided by 2<br>10: divided by 4<br>11: divided by 8 |
| **(R0D) PWM0PRD** | | | | **Function related to: PWM0** |
| PWM0PRD | 0d.7~0 | W | FF | PWM0 period data |
| **(R11) TM1RLD** | | | | **Function related to: Timer1** |
| TM1RLD | 11.7~0 | W | 0 | Timer1 reload Data |
| **(R12) TM1CTL** | | | | **Function related to: Timer1** |
| TM1PSC | 12.3~0 | W | 0 | Timer1 prescaler. Timer1 clock source (Fsys/2)<br>0000: divided by 1<br>0001: divided by 2<br>0010: divided by 4<br>0011: divided by 8<br>0100: divided by 16<br>0101: divided by 32<br>0110: divided by 64<br>0111: divided by 128<br>1xxx: divided by 256 |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R14) ATKUB0H** | | | | **Function related to: Touch Key** |
| ATKUB0H | 14.1~0 | W | 0 | Auto Touch Key TK0 upper boundary MSB[9:8] |
| **(R15) ATKUB0L** | | | | **Function related to: Touch Key** |
| ATKUB0L | 15.7~0 | W | FF | Auto Touch Key TK0 upper boundary LSB[7:0] |
| **(R16) ATKLB0H** | | | | **Function related to: Touch Key** |
| ATKLB0H | 16.1~0 | W | 0 | Auto Touch Key TK0 lower boundary MSB[9:8] |
| **(R17) ATKLB0L** | | | | **Function related to: Touch Key** |
| ATKLB0L | 17.7~0 | W | 0 | Auto Touch Key TK0 lower boundary LSB[7:0] |
| **(R18) ATKUB1H** | | | | **Function related to: Touch Key** |
| ATKUB1H | 18.1~0 | W | 0 | Auto Touch Key TK1 upper boundary MSB[9:8] |
| **(R19) ATKUB1L** | | | | **Function related to: Touch Key** |
| ATKUB1L | 19.7~0 | W | FF | Auto Touch Key TK1 upper boundary LSB[7:0] |
| **(R1A) ATKLB1H** | | | | **Function related to: Touch Key** |
| ATKLB1H | 1a.1~0 | W | 0 | Auto Touch Key TK1 lower boundary MSB[9:8] |
| **(R1B) ATKLB1L** | | | | **Function related to: Touch Key** |
| ATKLB1L | 1b.7~0 | W | 0 | Auto Touch Key TK1 lower boundary LSB[7:0] |
| **(R1C) ATKUB2H** | | | | **Function related to: Touch Key** |
| ATKUB2H | 1c.1~0 | W | 0 | Auto Touch Key TK2 upper boundary MSB[9:8] |
| **(R1D) ATKUB2L** | | | | **Function related to: Touch Key** |
| ATKUB2L | 1d.7~0 | W | FF | Auto Touch Key TK2 upper boundary LSB[7:0] |
| **(R1E) ATKLB2H** | | | | **Function related to: Touch Key** |
| ATKLB2H | 1e.1~0 | W | 0 | Auto Touch Key TK2 lower boundary MSB[9:8] |
| **(R1F) ATKLB2L** | | | | **Function related to: Touch Key** |
| ATKLB2L | 1f.7~0 | W | 0 | Auto Touch Key TK2 lower boundary LSB[7:0] |
| **(R20) ATKUB3H** | | | | **Function related to: Touch Key** |
| ATKUB3H | 20.1~0 | W | 0 | Auto Touch Key TK3 upper boundary MSB[9:8] |
| **(R21) ATKUB3L** | | | | **Function related to: Touch Key** |
| ATKUB3L | 21.7~0 | W | FF | Auto Touch Key TK3 upper boundary LSB[7:0] |
| **(R22) ATKLB3H** | | | | **Function related to: Touch Key** |
| ATKLB3H | 22.1~0 | W | 0 | Auto Touch Key TK3 lower boundary MSB[9:8] |
| **(R23) ATKLB3L** | | | | **Function related to: Touch Key** |
| ATKLB3L | 23.7~0 | W | 0 | Auto Touch Key TK3 lower boundary LSB[7:0] |
| **(R24) ATKUB4H** | | | | **Function related to: Touch Key** |
| ATKUB4H | 24.1~0 | W | 0 | Auto Touch Key TK4 upper boundary MSB[9:8] |
| **(R25) ATKUB4L** | | | | **Function related to: Touch Key** |
| ATKUB4L | 25.7~0 | W | FF | Auto Touch Key TK4 upper boundary LSB[7:0] |
| **(R26) ATKLB4H** | | | | **Function related to: Touch Key** |
| ATKLB4H | 26.1~0 | W | 0 | Auto Touch Key TK4 lower boundary MSB[9:8] |
| **(R27) ATKLB4L** | | | | **Function related to: Touch Key** |
| ATKLB4L | 27.7~0 | W | 0 | Auto Touch Key TK4 lower boundary LSB[7:0] |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R28) ATKUB5H** | | | | **Function related to: Touch Key** |
| ATKUB5H | 28.1~0 | W | 0 | Auto Touch Key TK5 upper boundary MSB[9:8] |
| **(R29) ATKUB5L** | | | | **Function related to: Touch Key** |
| ATKUB5L | 29.7~0 | W | FF | Auto Touch Key TK5 upper boundary LSB[7:0] |
| **(R2A) ATKLB5H** | | | | **Function related to: Touch Key** |
| ATKLB5H | 2a.1~0 | W | 0 | Auto Touch Key TK5 lower boundary MSB[9:8] |
| **(R2B) ATKLB5L** | | | | **Function related to: Touch Key** |
| ATKLB5L | 2b.7~0 | W | 0 | Auto Touch Key TK5 lower boundary LSB[7:0] |
| **(R2C) ATKUB6H** | | | | **Function related to: Touch Key** |
| ATKUB6H | 2c.1~0 | W | 0 | Auto Touch Key TK6 upper boundary MSB[9:8] |
| **(R2D) ATKUB6L** | | | | **Function related to: Touch Key** |
| ATKUB6L | 2d.7~0 | W | FF | Auto Touch Key TK6 upper boundary LSB[7:0] |
| **(R2E) ATKLB6H** | | | | **Function related to: Touch Key** |
| ATKLB6H | 2e.1~0 | W | 0 | Auto Touch Key TK6 lower boundary MSB[9:8] |
| **(R2F) ATKLB6L** | | | | **Function related to: Touch Key** |
| ATKLB6L | 2f.7~0 | W | 0 | Auto Touch Key TK6 lower boundary LSB[7:0] |
| **(R30) ATKUB7H** | | | | **Function related to: Touch Key** |
| ATKUB7H | 30.1~0 | W | 0 | Auto Touch Key TK7 upper boundary MSB[9:8] |
| **(R31) ATKUB7L** | | | | **Function related to: Touch Key** |
| ATKUB7L | 31.7~0 | W | FF | Auto Touch Key TK7 upper boundary LSB[7:0] |
| **(R32) ATKLB7H** | | | | **Function related to: Touch Key** |
| ATKLB7H | 32.1~0 | W | 0 | Auto Touch Key TK7 lower boundary MSB[9:8] |
| **(R33) ATKLB7L** | | | | **Function related to: Touch Key** |
| ATKLB7L | 33.7~0 | W | 0 | Auto Touch Key TK7 lower boundary LSB[7:0] |

## Instruction Set

Each instruction is a 14-bit word divided into an OPCODE, 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 listed 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 |
| 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 |
| MOVWF | f | **00 0000 1fff ffff** | 1 | - | Move W to "f" |
| MOVWR | r | **00 0000 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 dfff 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 00kk kkkk kkkk** | 2 | - | Call subroutine "k" |
| CLRWDT | | **00 0000 0000 0100** | 1 | TO, PD | Clear WDT/WKT Timer |
| GOTO | k | **11 00kk 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 |
| SLEEP | | **00 0000 0000 0011** | 1 | TO, PD | Go into standby mode, Clock oscillation stops |
| XORLW | k | **01 1111 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<br>A : W = 0x25 |

| ADDWF | Add W and "f" | |
|---|---|---|
| Syntax | ADDWF f [,d] | |
| Operands | f : 00h ~ 5Fh 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<br>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<br>A : W = 0x03 |

| ANDWF | AND W with "f" | |
|---|---|---|
| Syntax | ANDWF f [,d] | |
| Operands | f : 00h ~ 5Fh 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<br>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<br>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<br>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 ‘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 BTFSC FLAG, 1<br>TRUE GOTO SUB1<br>FALSE ... | B : PC = LABEL1<br>A : if FLAG.1 = 0, PC = FALSE<br>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<br>TRUE GOTO SUB1<br>FALSE ... | B : PC = LABEL1<br>A : if FLAG.1 = 0, PC = TRUE<br>if FLAG.1 = 1, PC = FALSE |

| CALL | Call subroutine "k" | |
|---|---|---|
| Syntax | CALL k | |
| Operands | K : 00h ~ 3FFh | |
| Operation | Operation: TOS ← (PC)+ 1, PC.9~0 ← k | |
| Status Affected | - | |
| OP-Code | 10 00kk kkkk kkkk | |
| Description | Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 10-bit immediate address is loaded into PC bits <9:0>. CALL is a two-cycle instruction. | |
| Cycle | 2 | |
| Example | LABEL1 CALL SUB1 | B : PC = LABEL1<br>A : PC = SUB1, TOS = LABEL1+1 |

| CLRF | Clear “f” | |
|---|---|---|
| Syntax | CLRF f | |
| Operands | f : 00h ~ 5Fh | |
| 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<br>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 Zero bit (Z) is set. | |
| Cycle | 1 | |
| Example | CLRW | B : W = 0x5A<br>A : W = 0x00, Z = 1 |

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

| COMF | Complement "f" | |
|---|---|---|
| Syntax | COMF f [,d] | |
| Operands | f : 00h ~ 5Fh, 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<br>A : REG1 = 0x13, W = 0xEC |

| DECF | Decrement "f" | |
|---|---|---|
| Syntax | DECF f [,d] | |
| Operands | f : 00h ~ 5Fh, 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<br>A : CNT = 0x00, Z = 1 |

| DECFSZ | Decrement "f", Skip if 0 | |
|---|---|---|
| Syntax | DECFSZ f [,d] | |
| Operands | f : 00h ~ 5Fh, 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<br>GOTO LOOP<br>CONTINUE | B : PC = LABEL1<br>A : CNT = CNT – 1<br>if CNT=0, PC = CONTINUE<br>if CNT≠0, PC = LABEL1+1 |

## GOTO Unconditional Branch

| | |
|---|---|
| Syntax | GOTO k |
| Operands | k : 00h ~ 3FFh |
| Operation | PC.9~0 ← k |
| Status Affected | - |
| OP-Code | 11 00kk kkkk kkkk |
| Description | GOTO is an unconditional branch. The 10-bit immediate value is loaded into PC bits <9:0>. GOTO is a two-cycle instruction. |
| Cycle | 2 |
| Example | LABEL1 GOTO SUB1 B : PC = LABEL1<br>A : PC = SUB1 |

## INCF Increment "f"

| | |
|---|---|
| Syntax | INCF f [,d] |
| Operands | f : 00h ~ 5Fh |
| 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<br>A : CNT = 0x00, Z = 1 |

## INCFSZ Increment "f", Skip if 0

| | |
|---|---|
| Syntax | INCFSZ f [,d] |
| Operands | f : 00h ~ 5Fh, 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 B : PC = LABEL1<br>GOTO LOOP A : CNT = CNT + 1<br>CONTINUE if CNT=0, PC = CONTINUE<br>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 is OR'ed with the eight-bit literal 'k'. The result is placed in the W register. |
| Cycle | 1 |
| Example | IORLW 0x35 B : W = 0x9A<br>A : W = 0xBF, Z = 0 |

| IORWF | Inclusive OR W with "f" |
|---|---|
| Syntax | IORWF f [,d] |
| Operands | f : 00h ~ 5Fh, 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<br>A : RESULT = 0x13, W = 0x93, Z = 0 |

| MOVFW | Move "f" to W |
|---|---|
| Syntax | MOVFW f |
| Operands | f : 00h ~ 5Fh |
| 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 | MOVF FSR, 0 B : W = ?<br>A : W ← f, if W = 0 Z = 1 |

| 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 = ?<br>A : W = 0x5A |

| MOVWF | Move W to "f" |
|---|---|
| Syntax | MOVWF f |
| Operands | f : 00h ~ 5Fh |
| 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<br>A : REG1 = 0x4F, W = 0x4F |

| MOVWR | Move W to "r" | |
|---|---|---|
| Syntax | MOVWR r | |
| Operands | r : 00h ~ 12h | |
| Operation | (r) ← (W) | |
| Status Affected | - | |
| OP-Code | 00 0000 00rr rrrr | |
| Description | Move data from W register to register 'r'. | |
| Cycle | 1 | |
| Example | MOVWR REG1 | B : REG1 = 0xFF, W = 0x4F<br>A : REG1 = 0x4F, W = 0x4F |

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

| 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<br>:<br>TABLE ADDWF PCL,1<br>RETLW k1<br>RETLW k2<br>:<br>RETLW kn | B : W = 0x07<br>A : W = value of k8 |

## 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 |

## 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<br>A : REG1 = 1110 0110<br>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<br>A : REG1 = 1110 0110<br>W = 0111 0011, C = 0 |

| SLEEP | Go into standby mode, Clock oscillation stops | |
|---|---|---|
| Syntax | SLEEP | |
| Operands | - | |
| Operation | - | |
| Status Affected | TO,PD | |
| OP-Code | 00 0000 0000 0011 | |
| Description | Go into SLEEP 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 = 3, W = 2, C = ?, Z = ?<br>A : REG1 = 1, W = 2, C = 1, Z = 0 |
| | SUBWF REG1,1 | B : REG1 = 2, W = 2, C = ?, Z = ?<br>A : REG1 = 0, W = 2, C = 1, Z = 1 |
| | SUBWF REG1,1 | B : REG1 = 1, W = 2, C = ?, Z = ?<br>A : REG1 = FFh, W = 2, 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 REG1, 0 | B : REG1 = 0xA5<br>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 = ?<br>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 1111 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<br>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<br>A : REG = 0x1A, W = 0xB5 |

# Electrical Characteristics

## 1. Absolute Maximum Ratings ($T_A$ = 25℃)

| Parameter | Rating | Unit |
|---|---|---|
| Supply voltage | $V_{SS}$ -0.3 to $V_{SS}$ +4.0 | 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 | 4.0 | V |
| Operating temperature | -40 to +85 | ℃ |
| Storage temperature | -65 to +150 | |

## 2. DC Characteristics ($T_A$=25℃, $V_{DD}$=1.3V to 4.0V)

| Parameter | Symbol | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Operating Voltage | $V_{DD}$ | FAST mode, 25°C, Fsys = 4 MHz | | 1.5 | – | 4.0 | V |
| | | FAST mode, 25°C, Fsys = 2 MHz | | 1.3 | – | 4.0 | |
| | | SLOW mode, 25°C, Fsys = 17 KHz | | 1.3 | – | 4.0 | |
| Input High Voltage | $V_{IH}$ | All Input, except PA7 | $V_{DD}$ = 3V | $0.6V_{DD}$ | – | – | V |
| | | PA7 | | $0.7V_{DD}$ | – | – | |
| Input Low Voltage | $V_{IL}$ | All Input | $V_{DD}$ = 3V | – | – | $0.2V_{DD}$ | V |
| I/O Port Source Current | $I_{OH}$ | All Output | $V_{DD}$ = 3V, $V_{OH}$ = $0.9V_{DD}$ | 2 | 4 | – | mA |
| I/O Port Sink Current | $I_{OL}$ | All Output | $V_{DD}$ = 3V, $V_{OL}$ = $0.1V_{DD}$ | 5 | 10 | – | mA |
| Input Leakage Current (pin high) | $I_{ILH}$ | All Input | $V_{IN}$ = $V_{DD}$ | – | – | 1 | µA |
| Input Leakage Current (pin low) | $I_{ILL}$ | All Input | $V_{IN}$ = 0V | – | – | -1 | |
| Supply Current | $I_{DD}$ | FAST mode, LVR enable, WDT enable | $V_{DD}$ = 3V, FIRC = 4 MHz | – | 0.7 | – | mA |
| | | | $V_{DD}$ = 3V, FIRC = 2 MHz | – | 0.5 | – | |
| | | SLOW mode, LVR enable | $V_{DD}$ = 3V, SIRC = 17 KHz | – | 3.5 | – | µA |
| | | IDLE mode, LVR enable | $V_{DD}$ = 3V, SIRC = 17 KHz | – | 1.5 | – | |
| | | IDLE mode, LVR disable | $V_{DD}$ = 3V, SIRC = 17 KHz | – | 1.1 | – | |
| | | STOP mode, LVR enable | $V_{DD}$ = 3V | – | 0.6 | – | |
| | | STOP mode, LVR disable | $V_{DD}$ = 3V | – | – | 0.1 | |

| Parameter | Symbol | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| System Clock Frequency | Fsys | $V_{DD}$ = 1.5V | – | – | 4 | MHz |
| | | $V_{DD}$ = 1.3V | – | – | 2 | |
| LVR Reference Voltage | $V_{LVR}$ | $T_A$ = 25°C | – | 1.9 | – | 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, $V_{DD}$ = 3V, All Pins except PA7 | – | 235 | – | KΩ |
| | | $V_{IN}$ = 0 V, $V_{DD}$ = 3V, PA7 | – | 160 | – | |

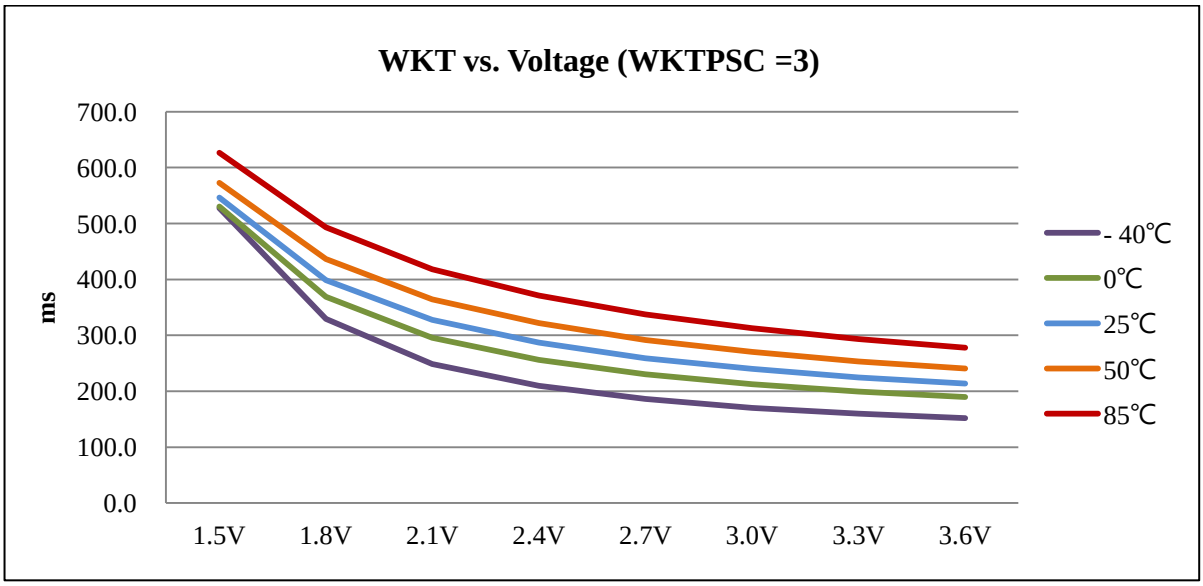
## 3. Clock Timing ($T_A$ = -40°C to +85°C)

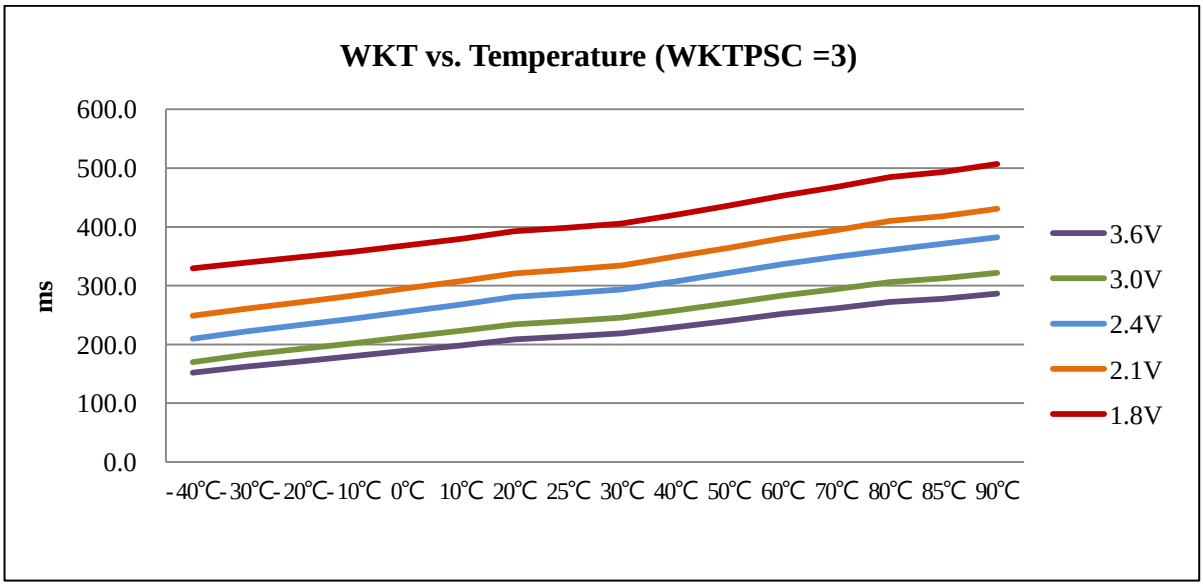
| Parameter | Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Fast Internal RC Frequency | 25℃, $V_{DD}$ = 2.1 ~ 4.0V | 7.8 | 8 | 8.2 | MHz |
| | -40℃ ~ 85℃, $V_{DD}$ = 2.1 ~ 4.0V | 7.6 | 8 | 8.4 | |

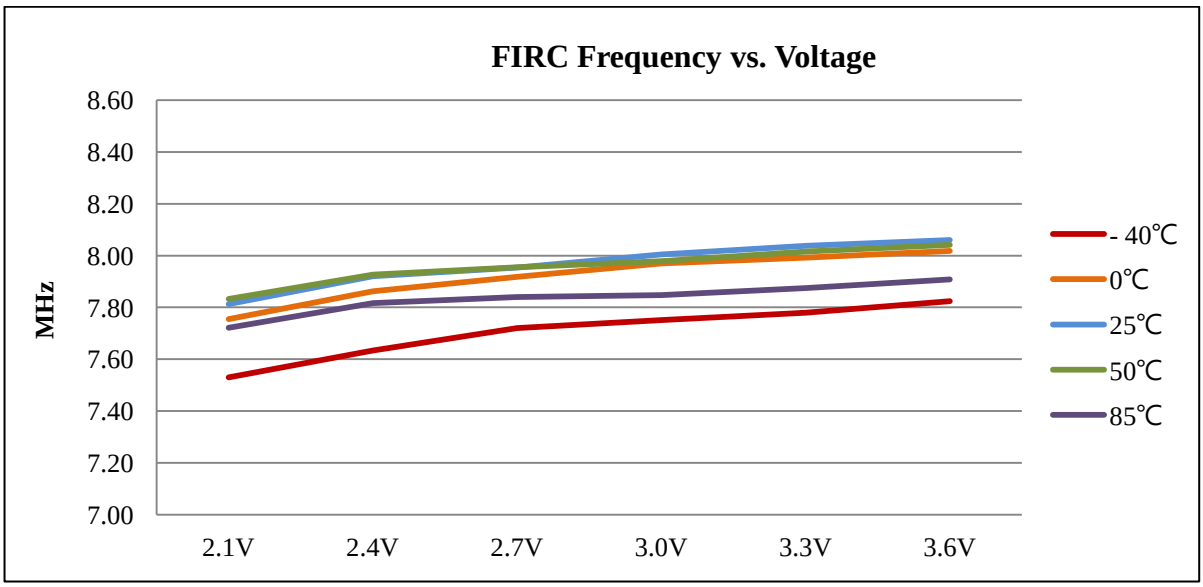
## 4. Reset Timing Characteristics ($T_A$ = -40°C to +85°C, $V_{DD}$ = 3V)

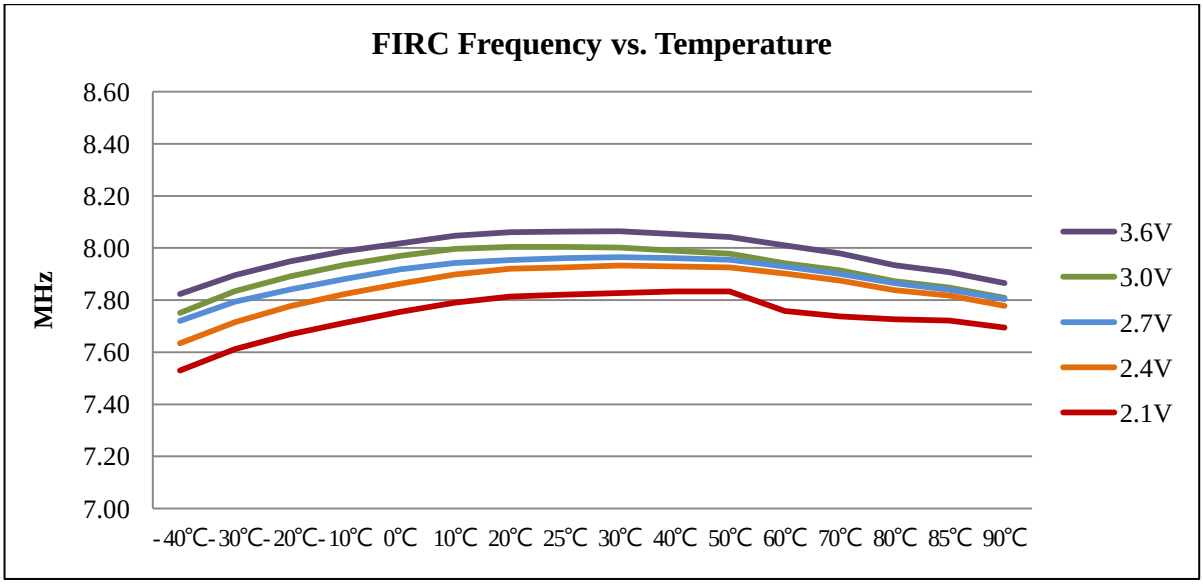
| Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| RESET Input Low width | Input $V_{DD}$ = 3 V ± 10 % | 3 | – | – | μs |
| WKT/WDT time | $V_{DD}$ = 3V, WKTPSC = 3 | – | 240 | – | ms |
| CPU start up time | $V_{DD}$ = 3V | – | 35 | – | ms |

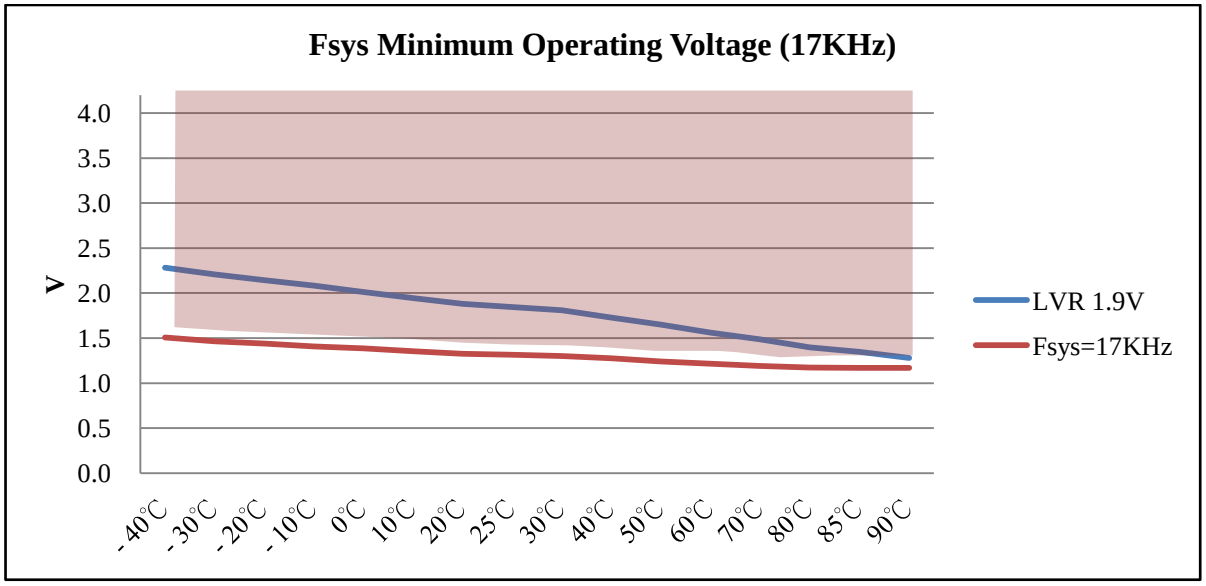
## 5. Characteristic Graphs



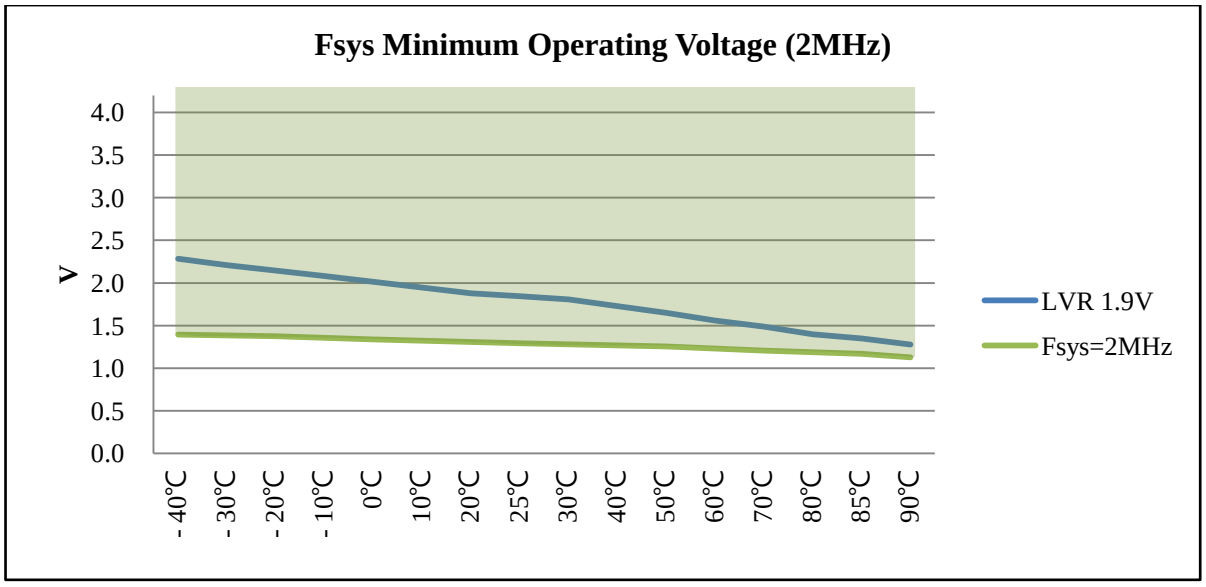




FIRC Frequency vs. Voltage
MHz
8.60
8.40
8.20
8.00
7.80
7.60
7.40
7.20
7.00
2.1V
2.4V
2.7V
3.0V
3.3V
3.6V
- 40℃
0℃
25℃
50℃
85℃

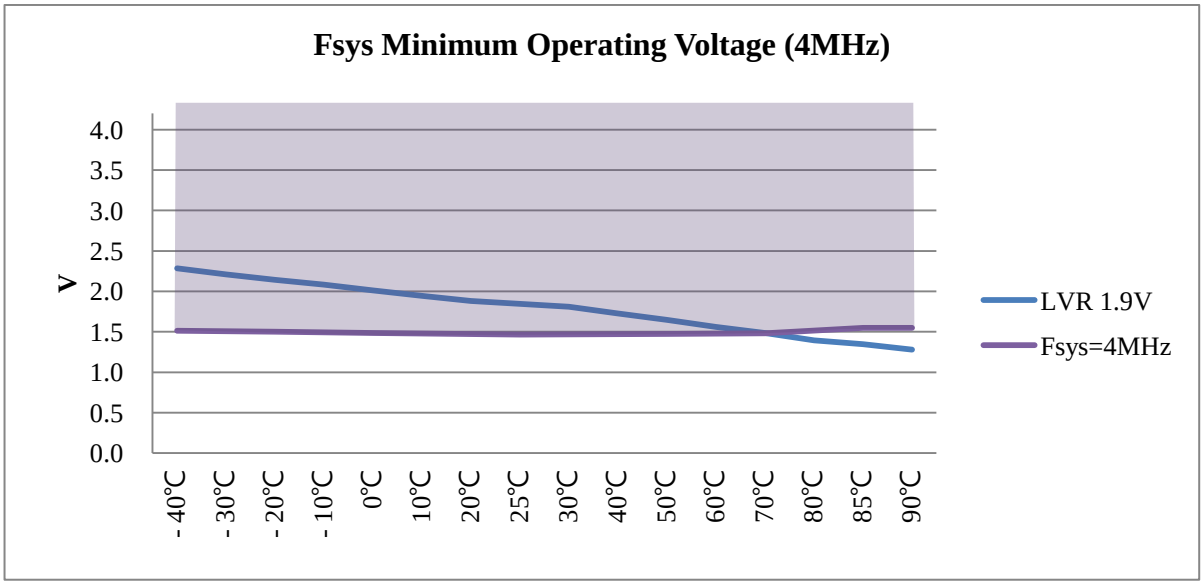

FIRC Frequency vs. Temperature
MHz
8.60
8.40
8.20
8.00
7.80
7.60
7.40
7.20
7.00
-40℃ -30℃ -20℃ -10℃ 0℃ 10℃ 20℃ 25℃ 30℃ 40℃ 50℃ 60℃ 70℃ 80℃ 85℃ 90℃
3.6V
3.0V
2.7V
2.4V
2.1V




**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.**




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

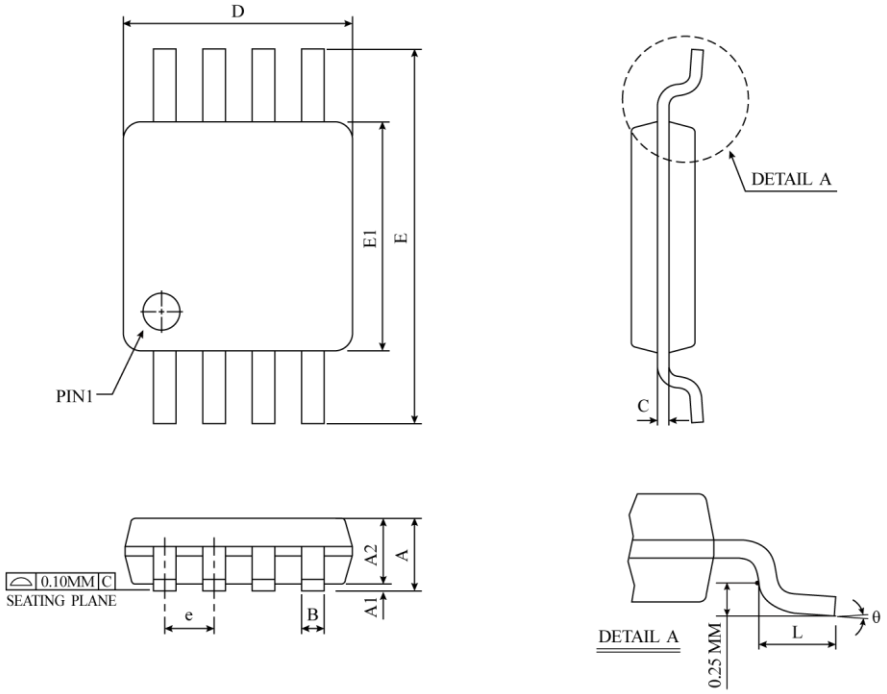
## Packaging information

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

The ordering information:

| Ordering number | Package |
|---|---|
| TM57MT21-MTP | Wafer/Dice blank chip |
| TM57MT21-COD | Wafer/Dice with code |
| TM57MT21-MTP-05 | DIP 20-pin (300 mil) |
| TM57MT21-MTP-21 | SOP 20-pin (300 mil) |
| TM57MT21AQ44-MTP-97 | QFN-20 (4x4x0.75-0.5mm) |
| TM57MT21-MTP-03 | DIP 16-pin (300 mil) |
| TM57MT21-MTP-16 | SOP 16-pin (150 mil) |
| TM57MT21-MTP-53 | MSOP 10-pin (118 mil) |
| TM57MT21-MTP-52 | MSOP 8-pin (118 mil) |

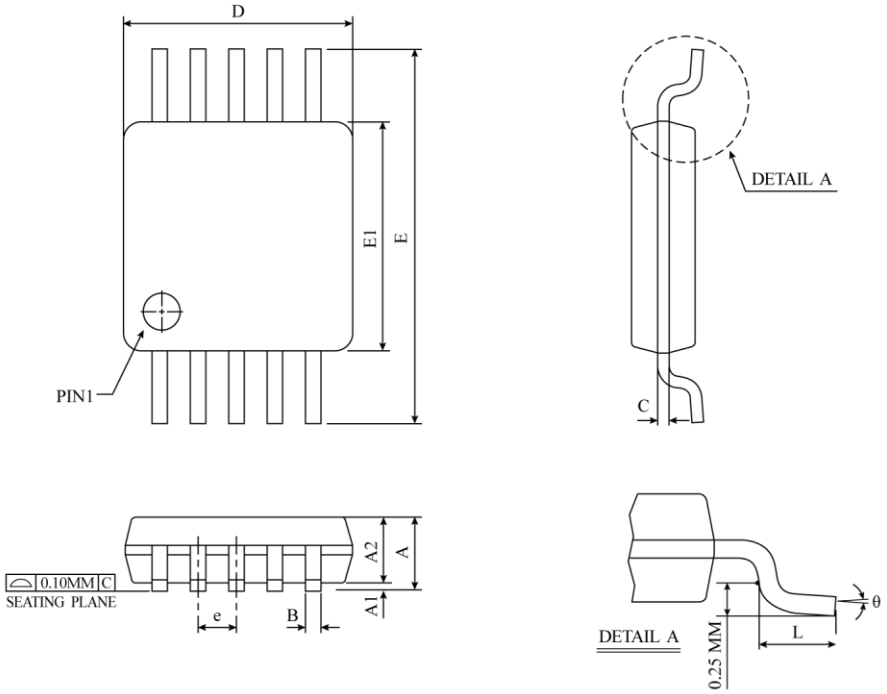
## 8-MSOP Package Dimension




| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | 0.81 | 0.96 | 1.10 | 0.032 | 0.038 | 0.043 |
| A1 | 0.05 | 0.10 | 0.15 | 0.002 | 0.004 | 0.006 |
| A2 | 0.76 | 0.86 | 0.95 | 0.030 | 0.034 | 0.037 |
| B | 0.28 | 0.33 | 0.38 | 0.011 | 0.013 | 0.015 |
| C | 0.13 | 0.18 | 0.23 | 0.005 | 0.007 | 0.009 |
| D | 2.90 | 3.00 | 3.10 | 0.114 | 0.118 | 0.122 |
| E | 4.75 | 4.90 | 5.05 | 0.187 | 0.193 | 0.199 |
| E1 | 2.90 | 3.00 | 3.10 | 0.114 | 0.118 | 0.122 |
| e | 0.65 BSC | | | 0.026 BSC | | |
| L | 0.40 | 0.55 | 0.70 | 0.016 | 0.022 | 0.028 |
| θ | 0° | 3° | 6° | 0° | 3° | 6° |
| JEDEC | | | | | | |

△ * NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS.
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.15 MM ( 0.006 INCH ) PER SIDE.
DIMENSION " E1 " DOES NOT INCLUDE MOLD PROTRUSIONS
MOLD PROTRUSIONS SHALL NOT EXCEED 0.25 MM ( 0.010 INCH ) PER SIDE.

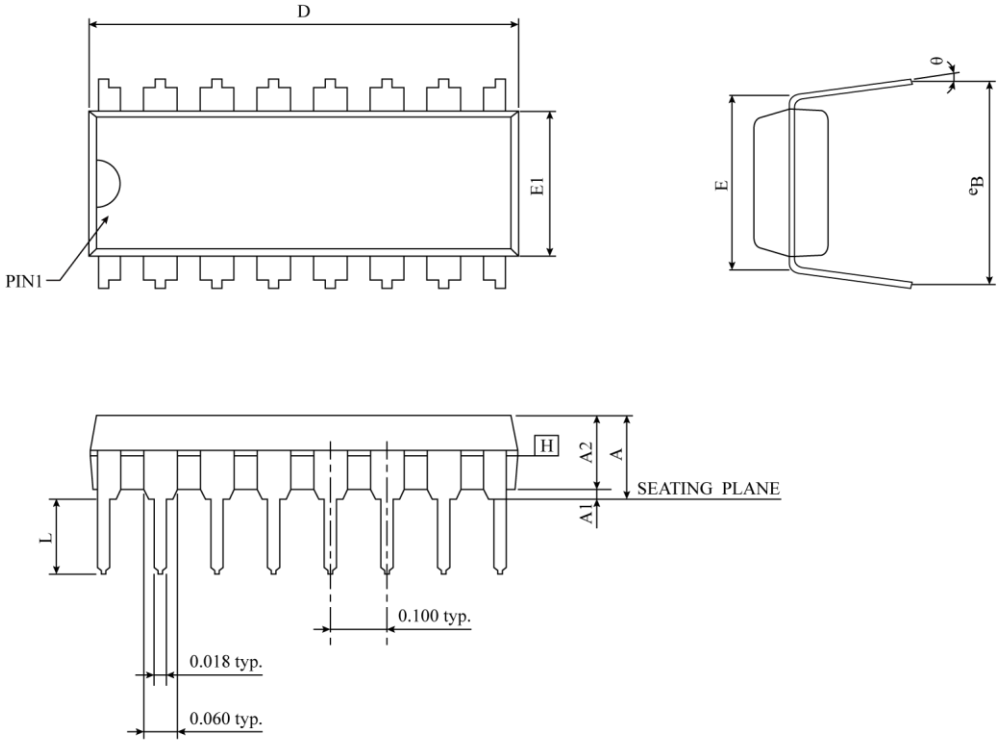
## 10-MSOP Package Dimension




| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | 0.81 | 0.96 | 1.10 | 0.032 | 0.038 | 0.043 |
| A1 | 0.05 | 0.10 | 0.15 | 0.002 | 0.004 | 0.006 |
| A2 | 0.75 | 0.85 | 0.95 | 0.030 | 0.034 | 0.037 |
| B | 0.17 | 0.22 | 0.27 | 0.007 | 0.009 | 0.011 |
| C | 0.13 | 0.18 | 0.23 | 0.005 | 0.007 | 0.009 |
| D | 2.90 | 3.00 | 3.10 | 0.114 | 0.118 | 0.122 |
| E | 4.75 | 4.90 | 5.05 | 0.187 | 0.193 | 0.199 |
| E1 | 2.90 | 3.00 | 3.10 | 0.114 | 0.118 | 0.122 |
| e | 0.50 BSC | | | 0.020 BSC | | |
| L | 0.40 | 0.55 | 0.70 | 0.016 | 0.022 | 0.028 |
| θ | 0° | 3° | 6° | 0° | 3° | 6° |
| JEDEC | | | | | | |

△ * NOTES：DIMENSION〝D〞DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS.
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.12 MM ( 0.005 INCH ) PER SIDE.
DIMENSION〝E1〞DOES NOT INCLUDE MOLD PROTRUSIONS
MOLD PROTRUSIONS SHALL NOT EXCEED 0.25 MM ( 0.010 INCH ) PER SIDE.

## 16-DIP Package Dimension

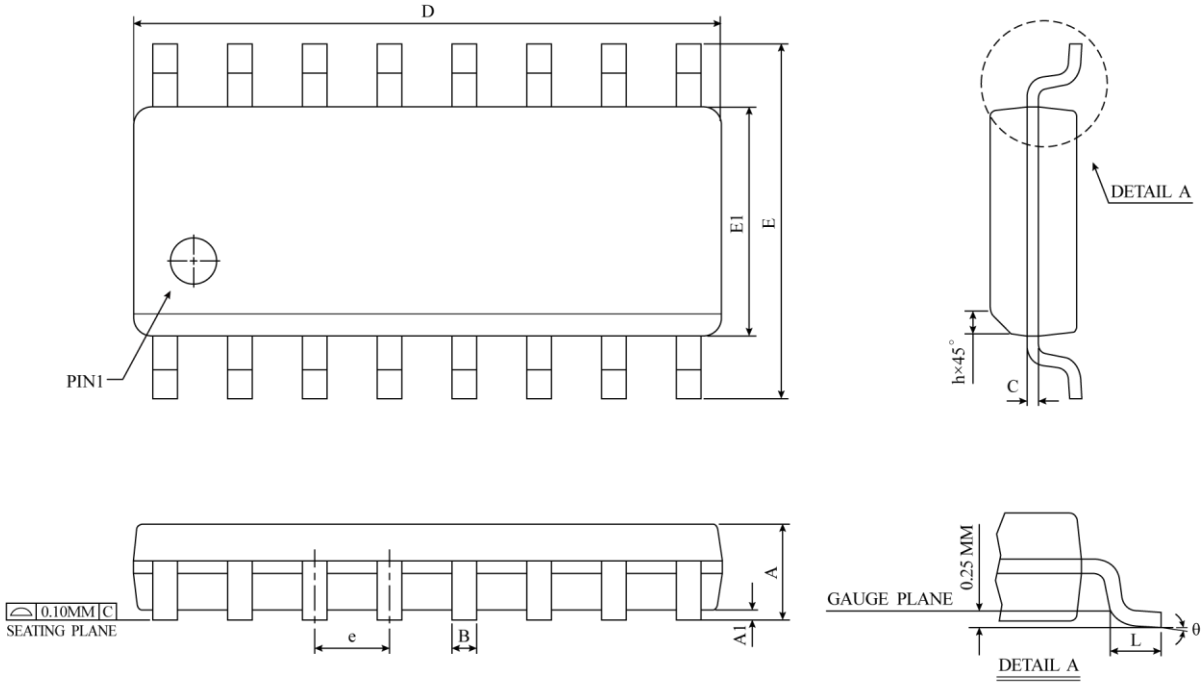



| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | - | - | 4.369 | - | - | 0.172 |
| A1 | 0.381 | 0.673 | 0.965 | 0.015 | 0.027 | 0.038 |
| A2 | 3.175 | 3.302 | 3.429 | 0.125 | 0.130 | 0.135 |
| D | 18.669 | 19.177 | 19.685 | 0.735 | 0.755 | 0.775 |
| E | | 7.620 BSC | | | 0.300 BSC | |
| E1 | 6.223 | 6.350 | 6.477 | 0.245 | 0.250 | 0.255 |
| L | 2.921 | 3.366 | 3.810 | 0.115 | 0.133 | 0.150 |
| $^{e}$B | 8.509 | 9.017 | 9.525 | 0.335 | 0.355 | 0.375 |
| θ | 0° | 7.5° | 15° | 0° | 7.5° | 15° |
| JEDEC | MS-001 (BB) | | | | | |

NOTES：

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

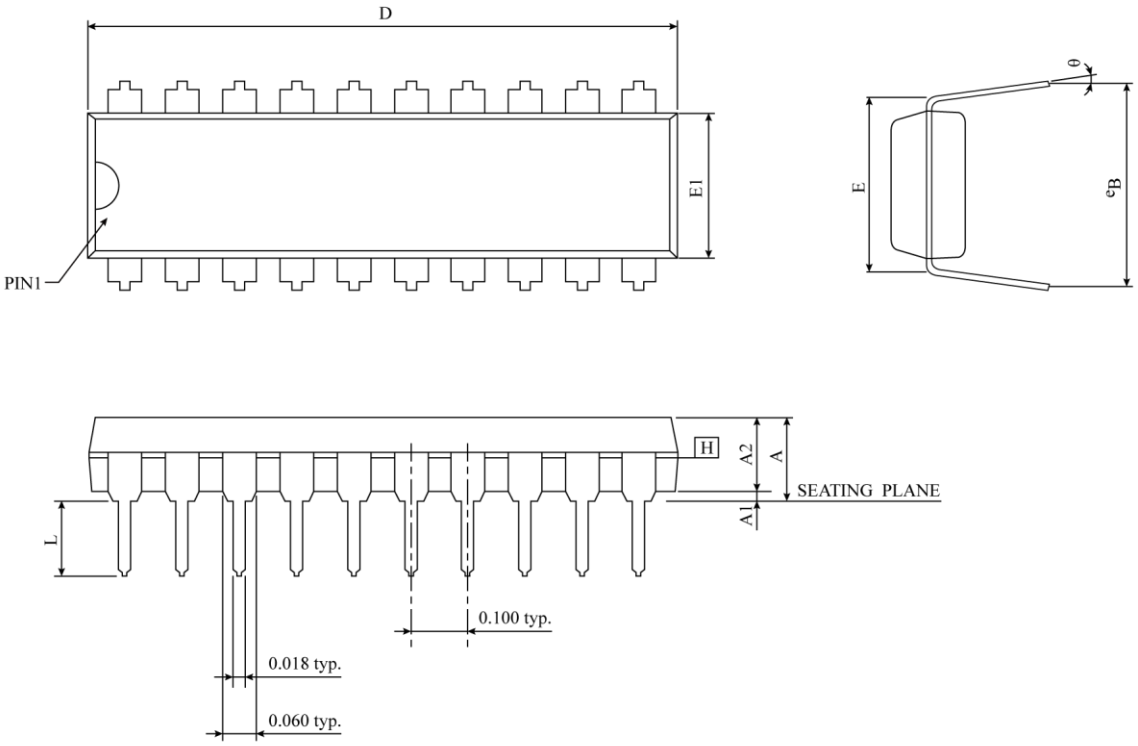
## 16-SOP Package Dimension




| 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.

## 20-DIP Package Dimension

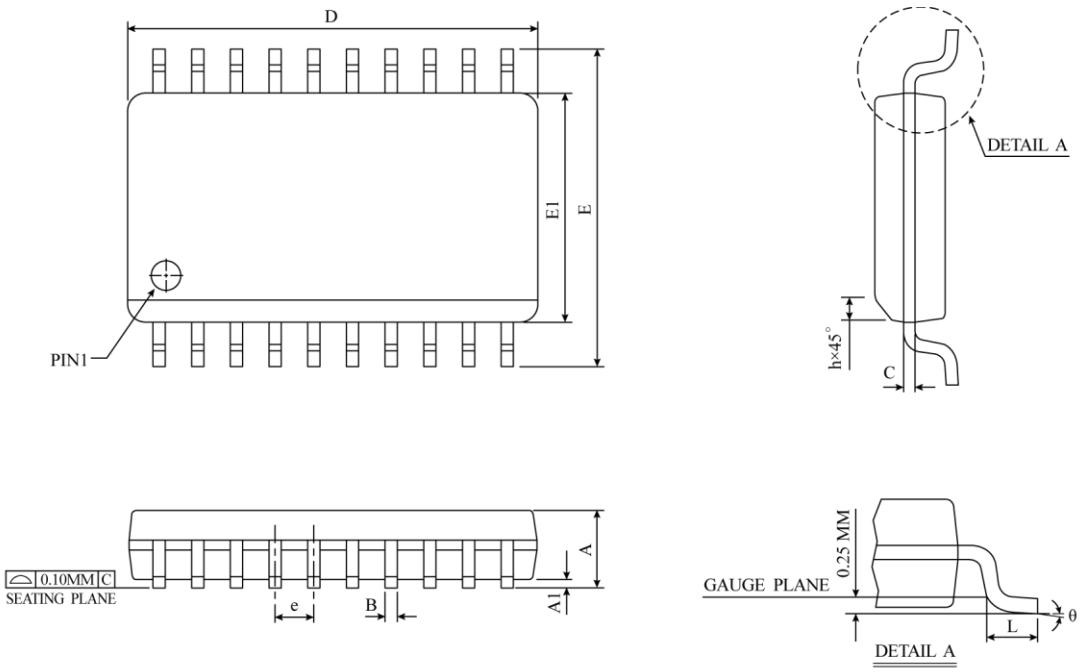



| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | - | - | 4.445 | - | - | 0.175 |
| A1 | 0.381 | - | - | 0.015 | - | - |
| A2 | 3.175 | 3.302 | 3.429 | 0.125 | 0.130 | 0.135 |
| D | 25.705 | 26.061 | 26.416 | 1.012 | 1.026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

NOTES：

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

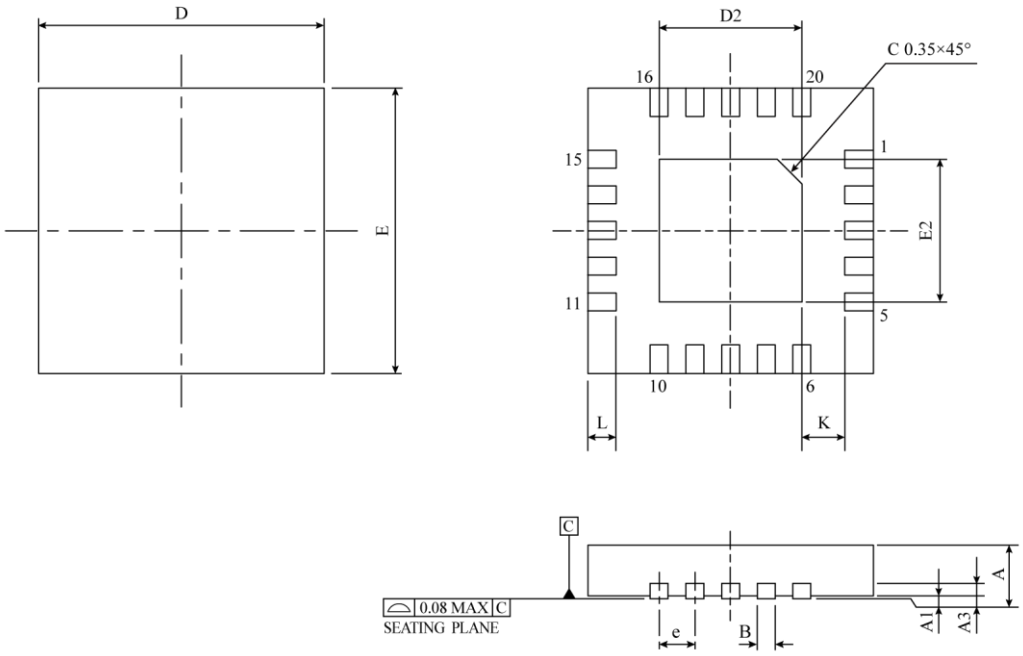
## 20-SOP Package Dimension




| 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.

## 20-QFN 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.