十速

TM57MT20

DATA SHEET

Rev 0.91

AMENDMENT HISTORY

Version	Date	Description
V0.90	Jun, 2014	New release
V0.91	Jan, 2015	Add LVR relevant information

CONTENTS

FEATURES

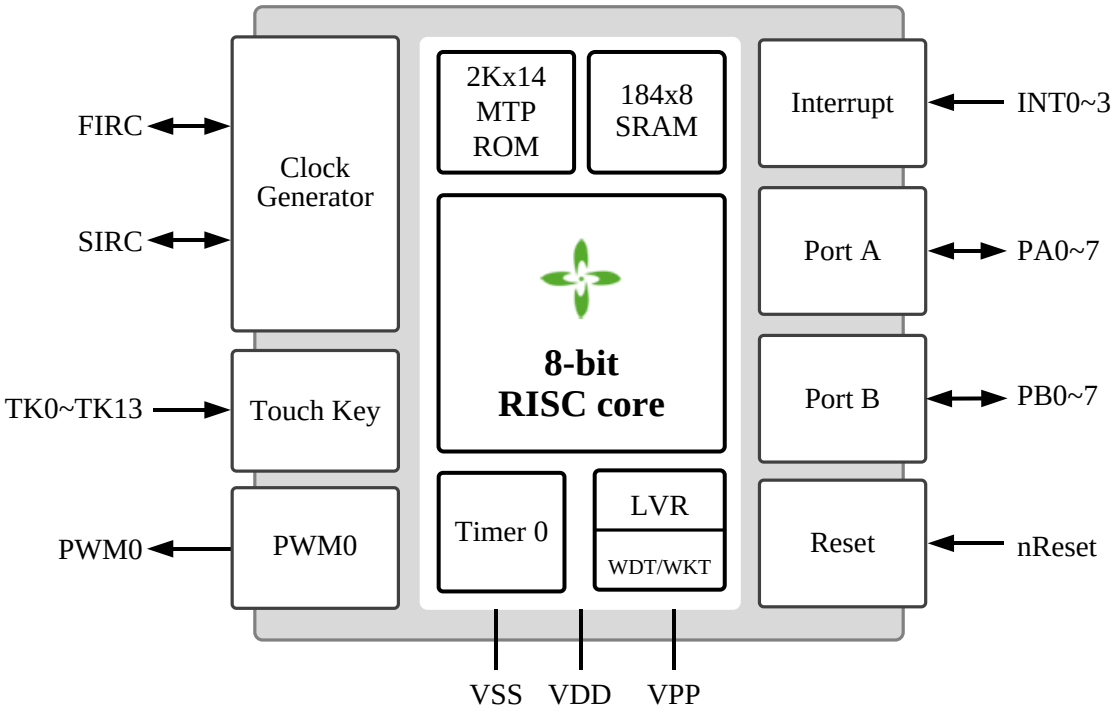
1. **36 instructions, two clock cycles execution**
2. **2Kx14 MTP (Multi-Time Programmable) ROM**
3. **184-Byte SRAM**
4. **6-level Stack**
5. **Independent RC Oscillating Watchdog Timer (or Wake-up Timer) with 4 adjustable Reset/Interrupt Times (960 ms/240 ms/120 ms/30 ms)**
6. **Independent Timers**

   Timer0 is 8-bit with 8-bit prescaler, Counter/Reload/Read/Write/Capture/Interrupt function
7. **Max. 16 Programmable I/O pins**
   - CMOS Output
   - Open-Drain Output
   - Schmitt Trigger Input
8. **Dual-clock selections**
   - FIRC (Fast Internal RC): 3M Hz
   - SIRC (Slow Internal RC) 32K/16K/8K/4KHz
9. **Power Saving Operation Modes**
   - Fast Mode: Slow Clock can be disabled or enabled
   - Slow Mode: Fast Clock stop, CPU running
   - Idle Mode: Slow Clock running, CPU off, Low power auto touch key running
   - Stop Mode: All Clock stop, wake-up Timer can be disabled or enabled
10. **7 Maskable Interrupt Sources**
    - 4 External Interrupt pins: 3-pin negative edge triggers, 1-pin positive or negative edge trigger
    - Timer0, Wake-up Timer Interrupt
    - I2C Interrupt
    - Auto Touch Key scan interrupt
11. **Independent 8-bit PWM**
    - PWM0 with prescaler/period-adjustment/buffer-reload/pos-neg-output
12. **14-channel Touch Key**

    1 ~ 4 Key H/W auto scan, threshold adjustable for each key

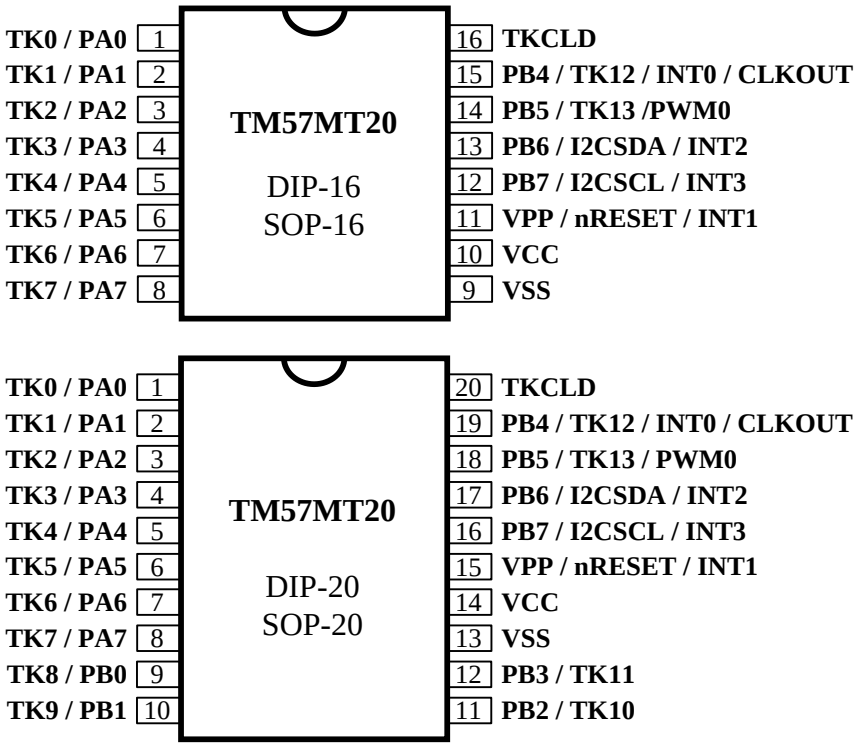
    Interrupt / Wake-up CPU while key is pressed
13. **Specific purpose slave I2C interface**
14. **Operation voltage: 1.2V to 3.6V**

BLOCK DIAGRAM



TM57MT20 Block Diagram

PIN ASSIGNMENT DIAGRAM



Pin Description

Name	In/Out	Pin Description
PA7-PA0 PB7-PB0	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.
VPP	I	PROM programming high voltage input
VCC	I	MTP programming high voltage input
nRESET	I	External active low reset
VCC, VSS	P	Power input pin and ground
INT3~INT2	I	External interrupt input
PWM0	O	PWM0 output
TK0-TK13	I	Touch key input
TKCLD	I	Touch key capacitor input
I2CSDA, I2CSCL	-	Inter-Integrated Circuit Pin

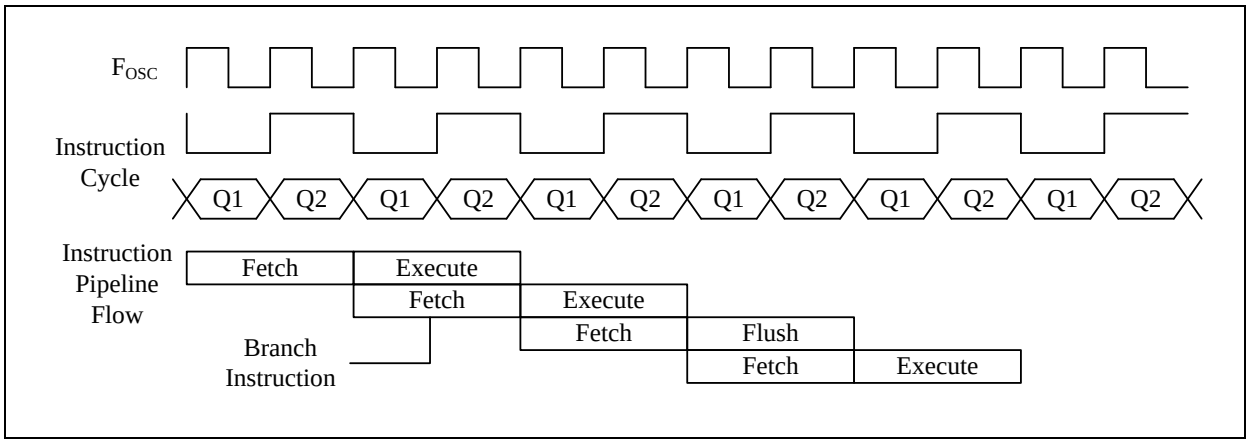
Programming pins : VCC / VSS / PA0 / PA1 / PA2 / VPP

FUNCTION DESCRIPTION

1. CPU Core

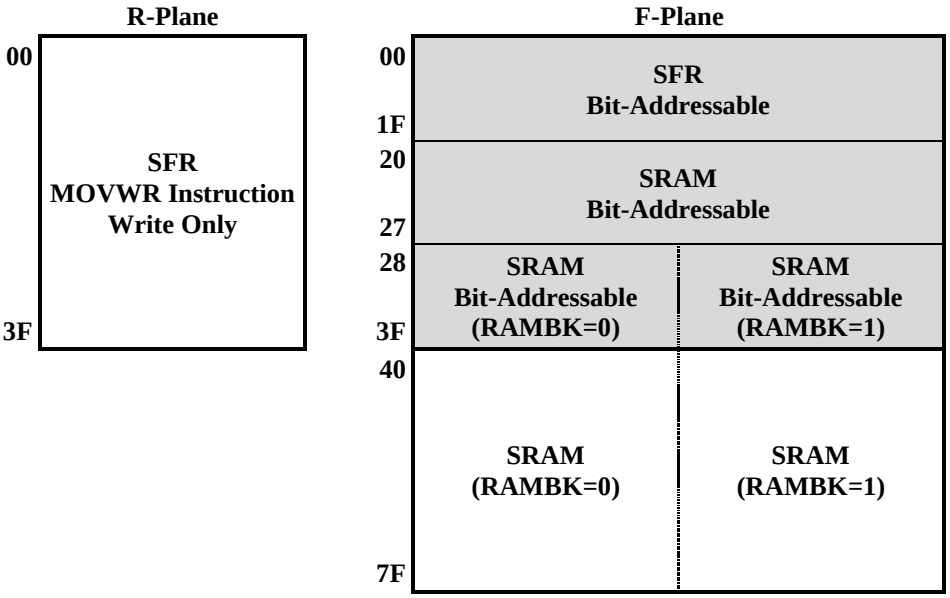
1.1 Clock Scheme and Instruction Cycle

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



1.2 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).



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 [9:8] keeps unchanged. 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.

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

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

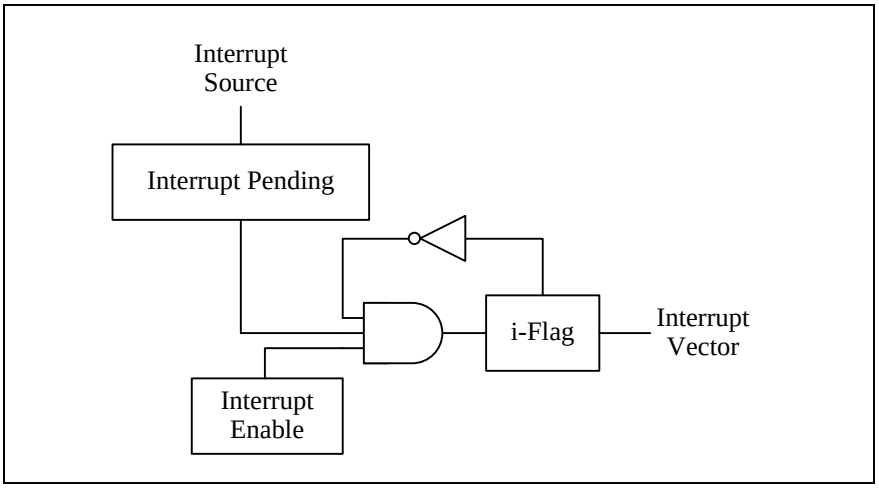
STATUS	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset Value	–	ϕ	–	–	–	**0**	**0**	**0**
R/W	–	**R/W**	–	**R**	**R**	**R/W**	**R/W**	**R/W**

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

1.6 Interrupt

The TM57MT20 has 1 level, 1 vector and six 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 TM57MT20 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 will 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.



2. Chip Operation Mode

2.1 Reset

The TM57MT20 can be RESET in four ways.

- Power-On-Reset
- External Pin Reset
- Watchdog Reset (WDT)

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

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)

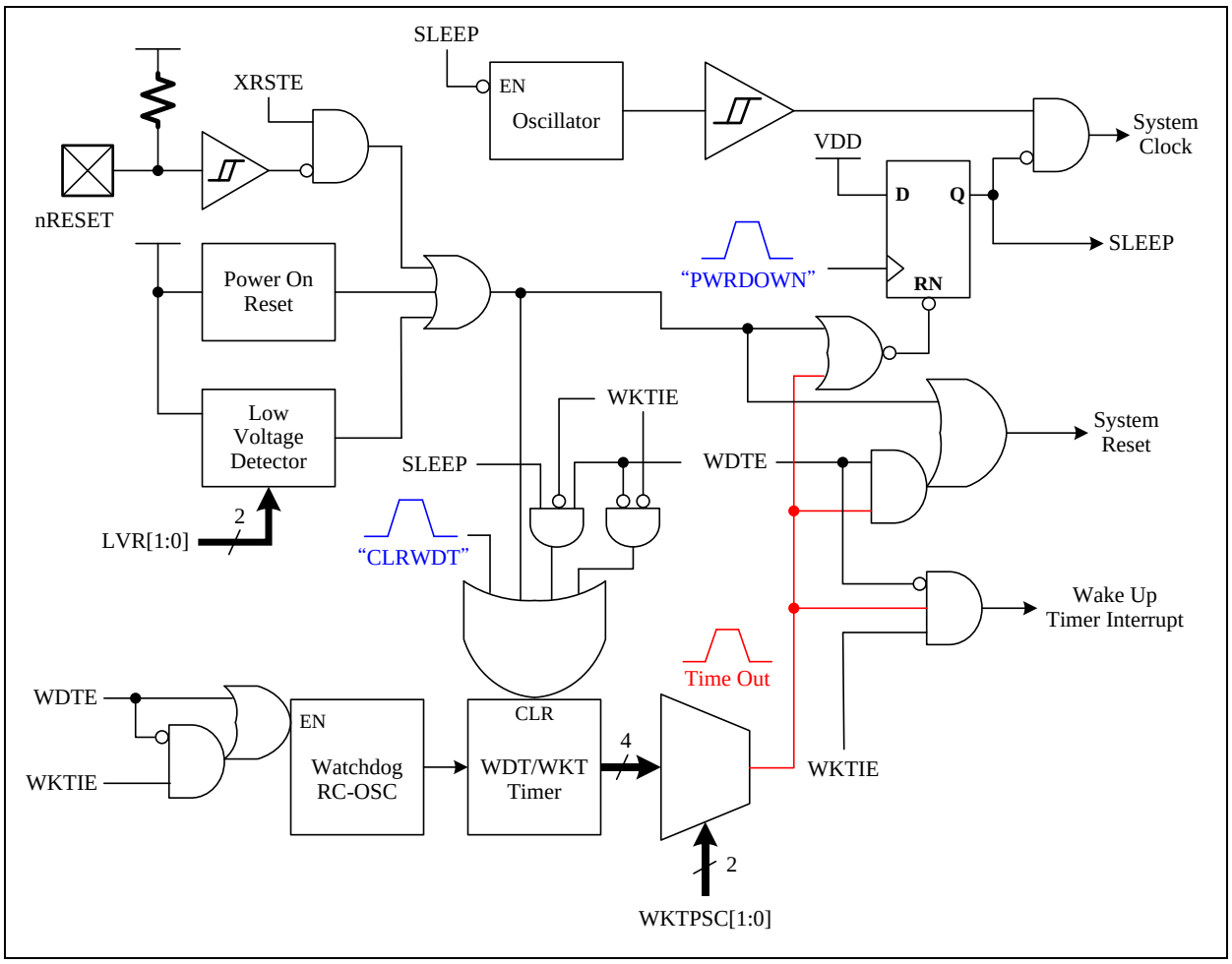
The System Configuration Register (SYSCFG) is located at MTP INFO area. The SYSCFG determines the option for initial condition of MCU. It is written by MTP Writer only. User can select chip operation mode by SYSCFG register. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in MTP will be protected, when user reads MTP.

Bit	**13~0**	
Default Value	**00_0000_00X_XXXX**	
Bit	Description	
13	**PROTECT**: Code Protection Selection	
	1	Enable
	0	Disable
12	Not used	
11-10	**LVR**: Low Voltage Reset Mode	
	11	LVR = 1.1V, sleep disable
	10	LVR = 1.1V, always enable
	01	LVR disable
	00	LVR = 1.1V, always enable
12-8	Not used	
7	**XRSTE**: External pin Reset Enable	
	1	Enable
	0	Disable, VPP as Input
6	**WDTE**: WDT Reset Enable	
	1	Enable WDT Reset, Disable WKT Timer
	0	Disable WDT Reset, Enable WKT Timer
5-0	Not used	

3. Peripheral Functional Block

3.1 Watchdog (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 960 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT/WKT works in both normal mode and STOP mode. During STOP mode, user can further choose to enable or disable the WDT/WKT by "WKTIE". If WKTIE=0 in sleep mode (no matter WDTE is 1 or 0), the internal RC Timer stops for power saving. In other words, user keeps the WDT/WKT alive in Stop Mode by setting WKTIE=1.



Watchdog clearance is controlled by CLRWDT instruction and moving any value into WDTCLR is to clear watchdog timer.

Example: Clear watchdog timer by CLRWDT instruction.

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

Example: Clear watchdog timer by writing WDTCLR register.

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

Example: Setup WDT time and enable after executing SLEEP instruction.

```
    MOVLW       011000000B
    MOVWR       R0E         ; Select WDT Time out = 960 ms @3V
    BSF         WKTIE       ; Setup WDT enable in STOP mode (default is disabled).
    SLEEP
```

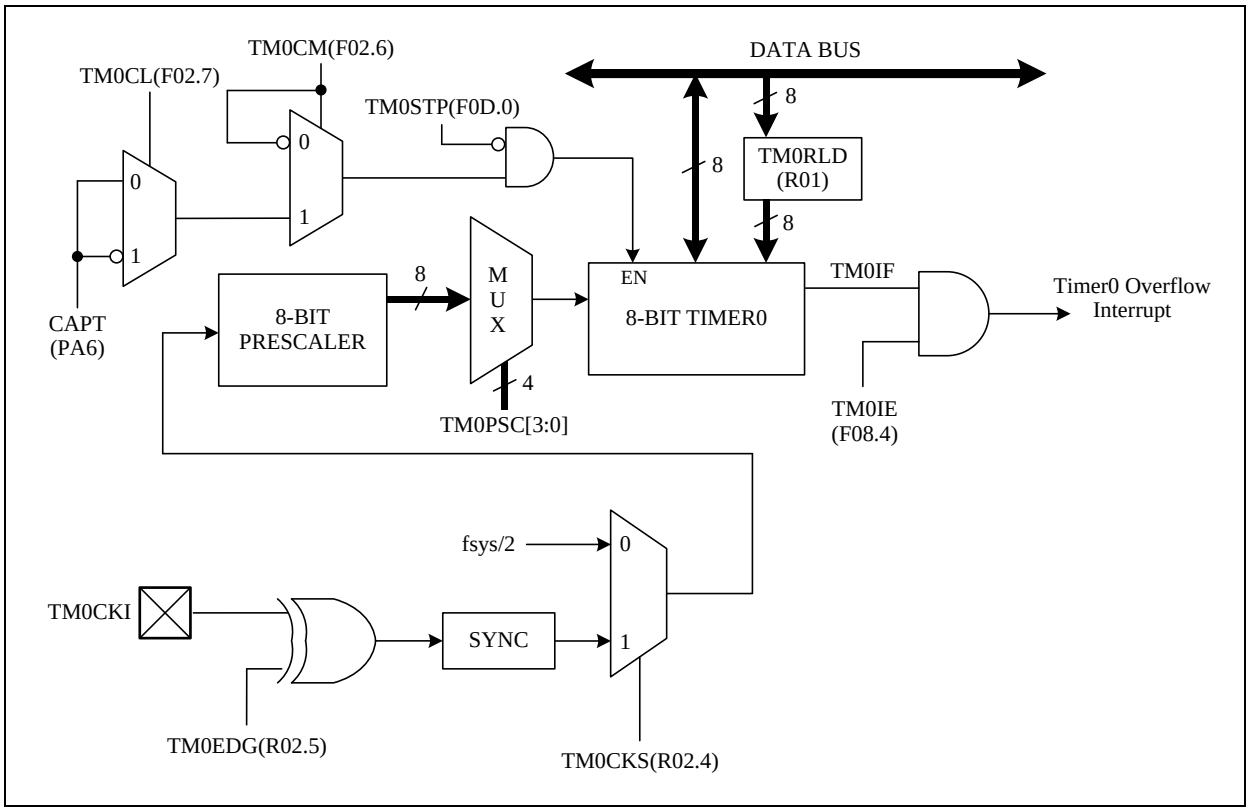
Example: Set WKT period and interrupt function.

```
    MOVLW       010000000B
    MOVWR       R0E         ; Select WKT Time out = 240 ms @3V
    MOVLW       11110111B   ; Clear WKT interrupt request flag by using byte operation
                            ; Don't use bit operation "BCF  WKTIF" clear interrupt flag
    MOVWF       INTIF       ; F-Plane 09H

    MOVLW       00001000B   ; Enable WKT interrupt function
    MOVWF       INTIE       ; F-Plane 08H
```

3.2 8-bit Timer/Counter/Capture (Timer0) with Pre-scaler (PSC)

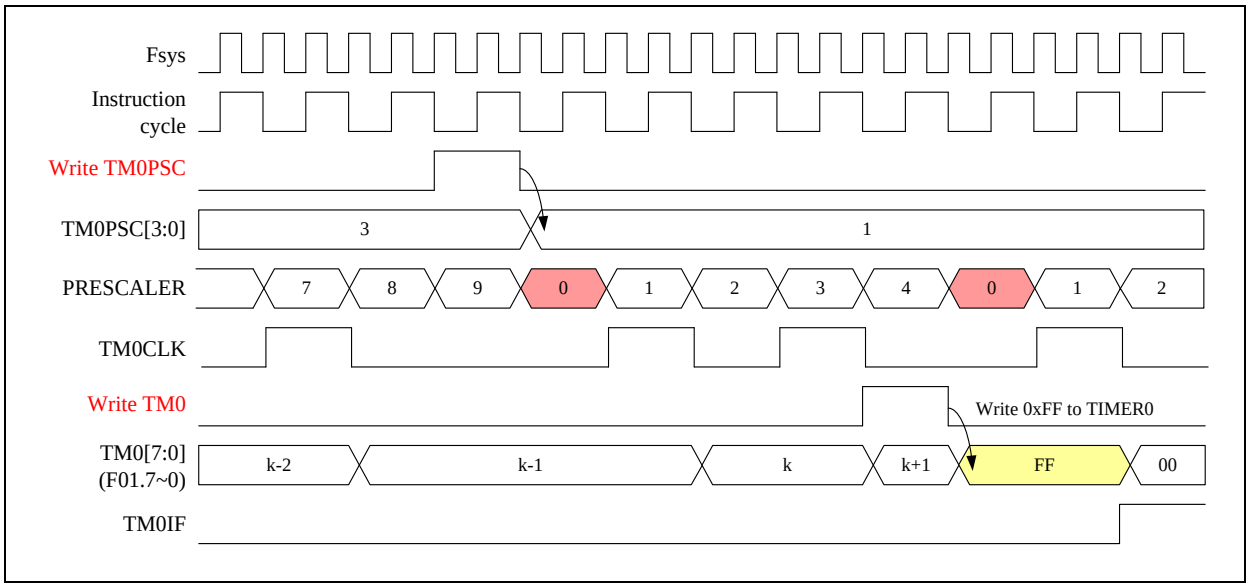
The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TMCKInput. The Timer0 increase rate is determined by "Timer0 Pre-Scale" (TM0PSC) register in R-Plane. The Timer0 can generate interrupt (TM0IF) when it rolls over.

Timer0 can be stopped counting if the TM0STP is set. Timer0 can be configured as capture mode. If TM0CM is set to "1", Timer0 will not count until the CAPT pin is active. TM0CL can select CAPT pin high or low active.



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

When the Timer0 prescaler (TM0PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer0 count. TM0CLK is the internal signal that causes the Timer0 to increase by 1 at the end of TM0CLK. TM0WR is also the internal signal that indicates the Timer0 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer0 counts from FFh to 00h, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set.



Timer0 works in Timer mode (TM0CKS=0, TM0CM=0)

The equation of TM0 interrupt time value is as following:

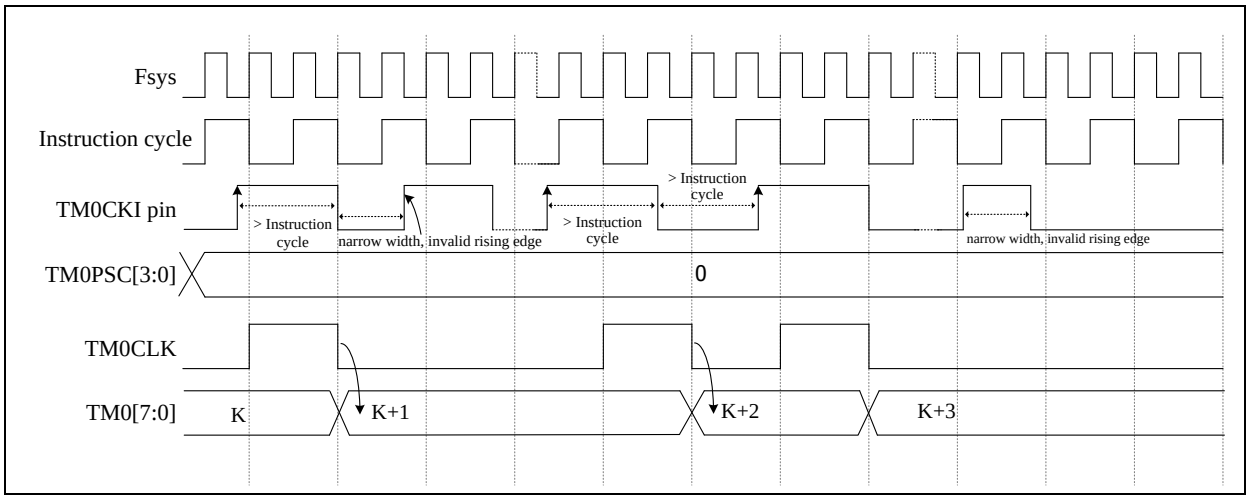
TM0 interrupt interval time = Instruction cycle / TM0PSC / (256-TM0)

Example: Setup TM0 work in Timer mode

```
; Setup TM0 clock source and divider
    MOVLW     00000101B     ; R02.4=0, Setup TM0 clock = Instruction cycle
    MOVWR     R02           ; R02.3~0=5 (TM0PSC)
                            ;TM0 clock prescaler=Instruction cycle divided by 32

; Set TM0 timer.
    BSF       TM0STP        ; Disable TM0 counting (Default "0")
    MOVLW     156
    MOVWF     TM0           ; Write 156 into TM0 register of F-Plane (F01)

    ; Enable TM0 timer and interrupt function.
    MOVLW     11101111B     ; Clear TM0 request interrupt flag by byte operation
    MOVWF     INTIF         ; F-Plane 09H
    MOVLW     00010000B     ; Enable TM0 interrupt function
    MOVWF     INTIE         ; F-Plane 08H
    BCF       TM0STP        ; Enable TM0 counting (Default “0”)
```

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

If TM0CKS=1, then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle 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.

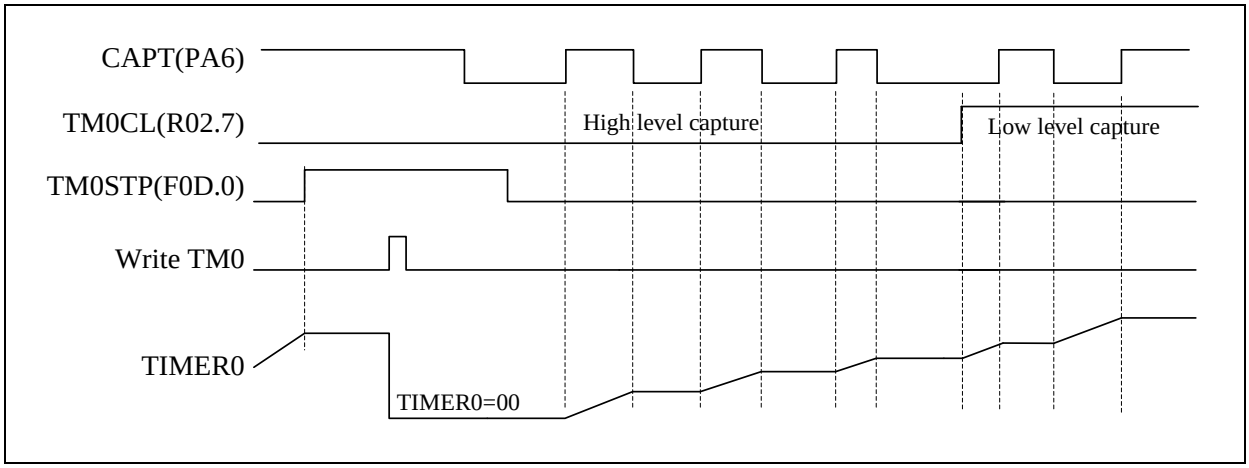


Timer0 works in Counter mode for TM0CKI (TM0EDG=0), TM0CKS=1, TM0CM=0

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

```
; Setup TM0 clock source from TM0CKI pin (PA2) and divider.
    MOVLW     00110000B
    MOVWR     R02           ; R02.5=1, Select TM0 prescaler counting edge = falling edge
                            ; R02.4=1, Setup TM0 clock = TM0CKI pin (PA2)
                            ; R02.3~0=0 (TM0PSC)
                            ; TM0 clock prescaler=Instruction cycle divided by 1

; Set TM0 timer and stop TM0 counting.
    BSF       TM0STP        ; Disable TM0 counting (Default "0")
    MOVLW     00H
    MOVWF     TM0           ; Write 0 into TM0 register of F-Plane

; Start TM0 count and read TM0 counter.
    BCF       TM0STP        ; Enable TM0 counting.
    NOP
    NOP
    NOP
    BSF       TM0STP        ; Disable TM0 counting (Default "0")
    MOVFW     TM0
```

The following timing diagram describes the Timer0 works in Capture mode. As shown in Figure, Timer0 is counting up only when PA6 (CAPT pin) is active (high or low). Note that the internal prescaler will be kept to next Timer0 count, so it will not lose the counting accuracy.



Timer0 works in Counter mode (TM0CM=1)

Example: Setup TM0 work in Capture mode

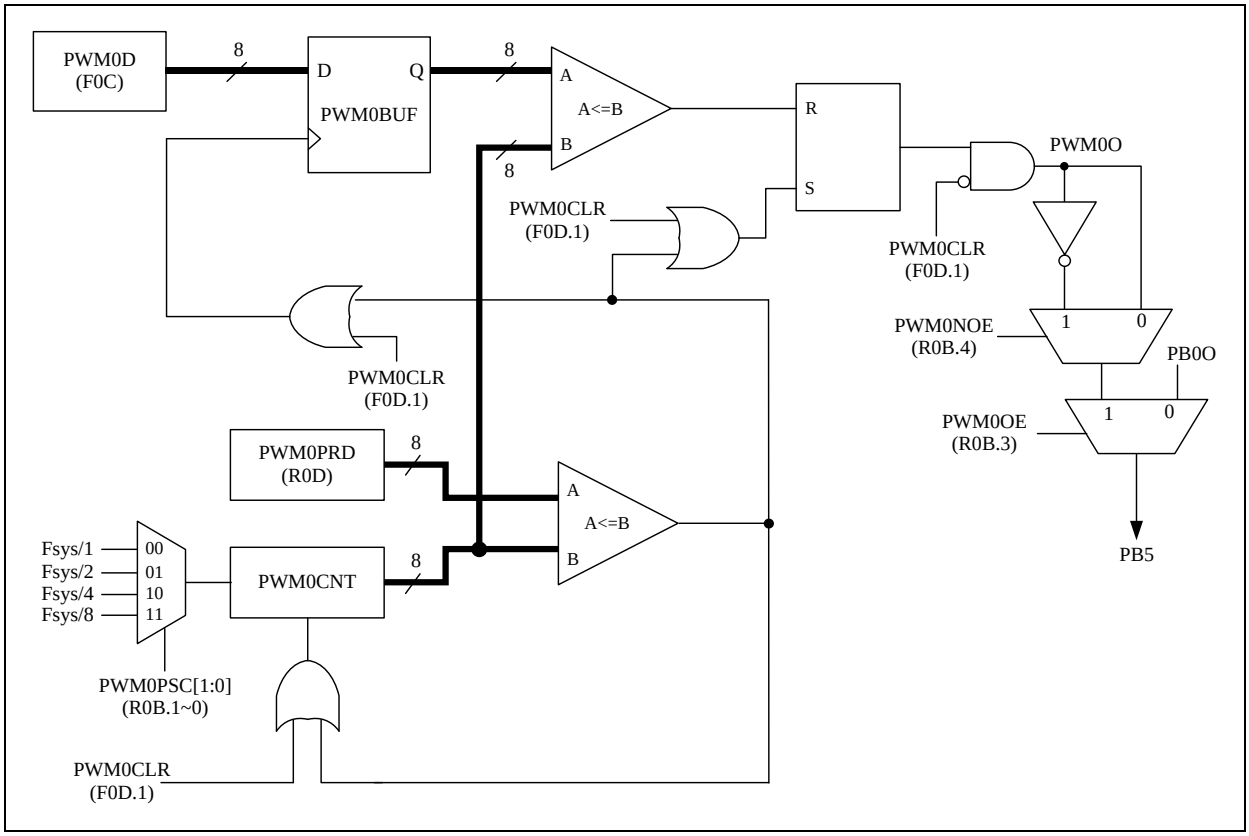
```
; Setup TM0 in low level capture.
    MOVLW     11000000B
    MOVWR     R02          ; R02.7~6=11, Select TM0 = Low level capture mode
                           ; R02.3~0=0 (TM0PSC)
                           ; TM0 clock prescaler=Instruction cycle divided by 1

; Set TM0 timer and stop TM0 counting.
    BSF       TM0STP       ; Disable TM0 counting (Default "0").
    MOVLW     00H
    MOVWF     TM0          ; Write 0 into TM0 register of F-Plane

; Start TM0 count and read TM0 counter.
    BCF       TM0STP       ; Enable TM0 counting
    NOP
    NOP
    NOP
    BSF       TM0STP       ; Disable TM0 counting (Default "0")

    MOVFW     TM0
```

3.3 PWM0

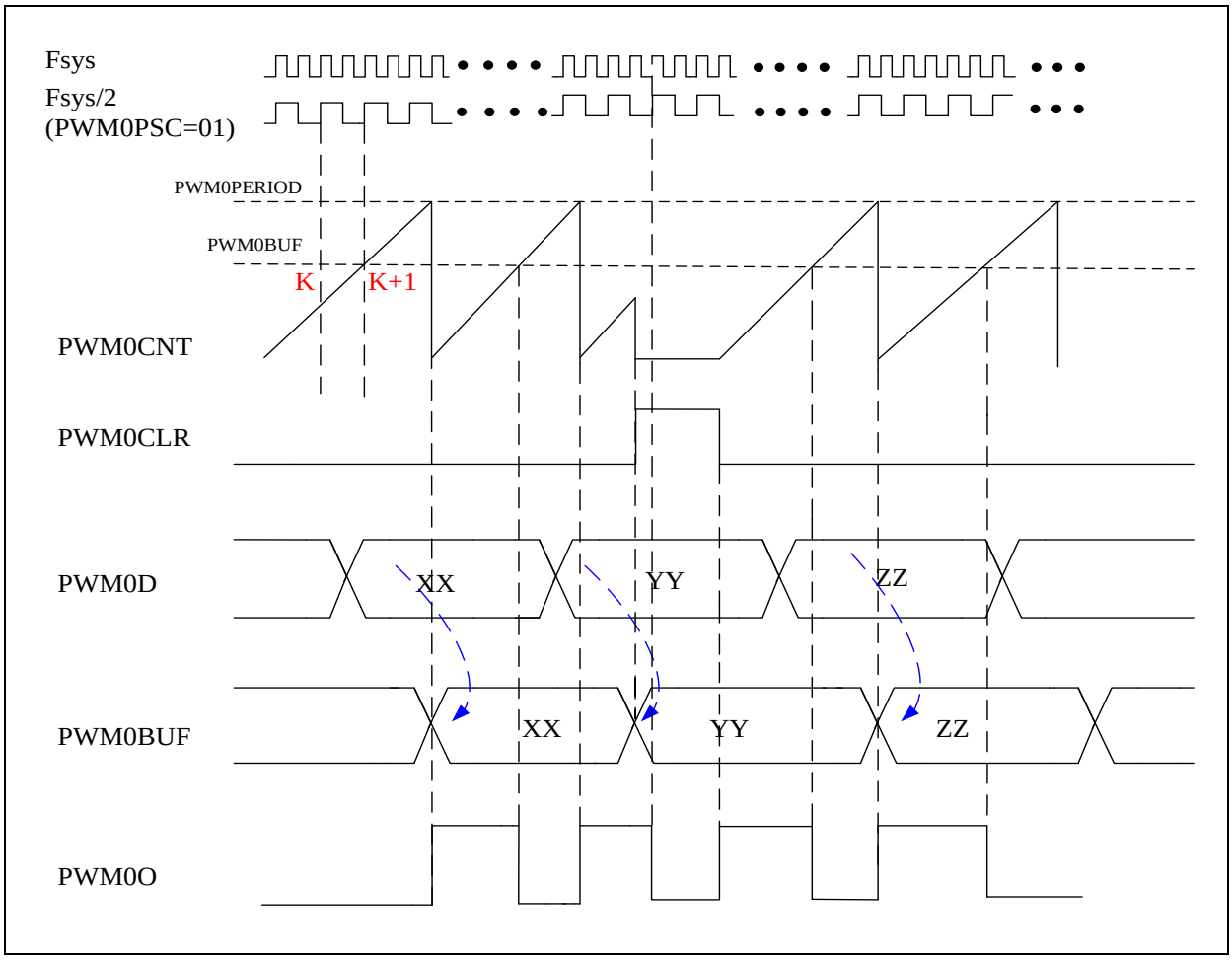
The chip has a built-in 8-bit PWM generator. The source clock comes from Fsys divided by 1, 2, 4, and 8. The PWM0 duty cycle can be changed with writing to PWM0D, writing to PWM0D will not change the current PWM duty until the current PWM period complete. When current PWM period is finished, the new value of PWM0D will be updated to the PWM0BUF.

The PWM0 will be output to PB5 if PWM0OE is set to 1 and PWM0NOE = 0. The complement of PWM0, PWM0N, will be output to PB5 if PWM0OE is set to 1 and PWN0NOE = 1. Setting the PWM0CLR bit will clear the PWM0 counter and load the PWM0D to PWM0BUF, PWM0CLR bit must be cleared so that the PWM0 counter can count.

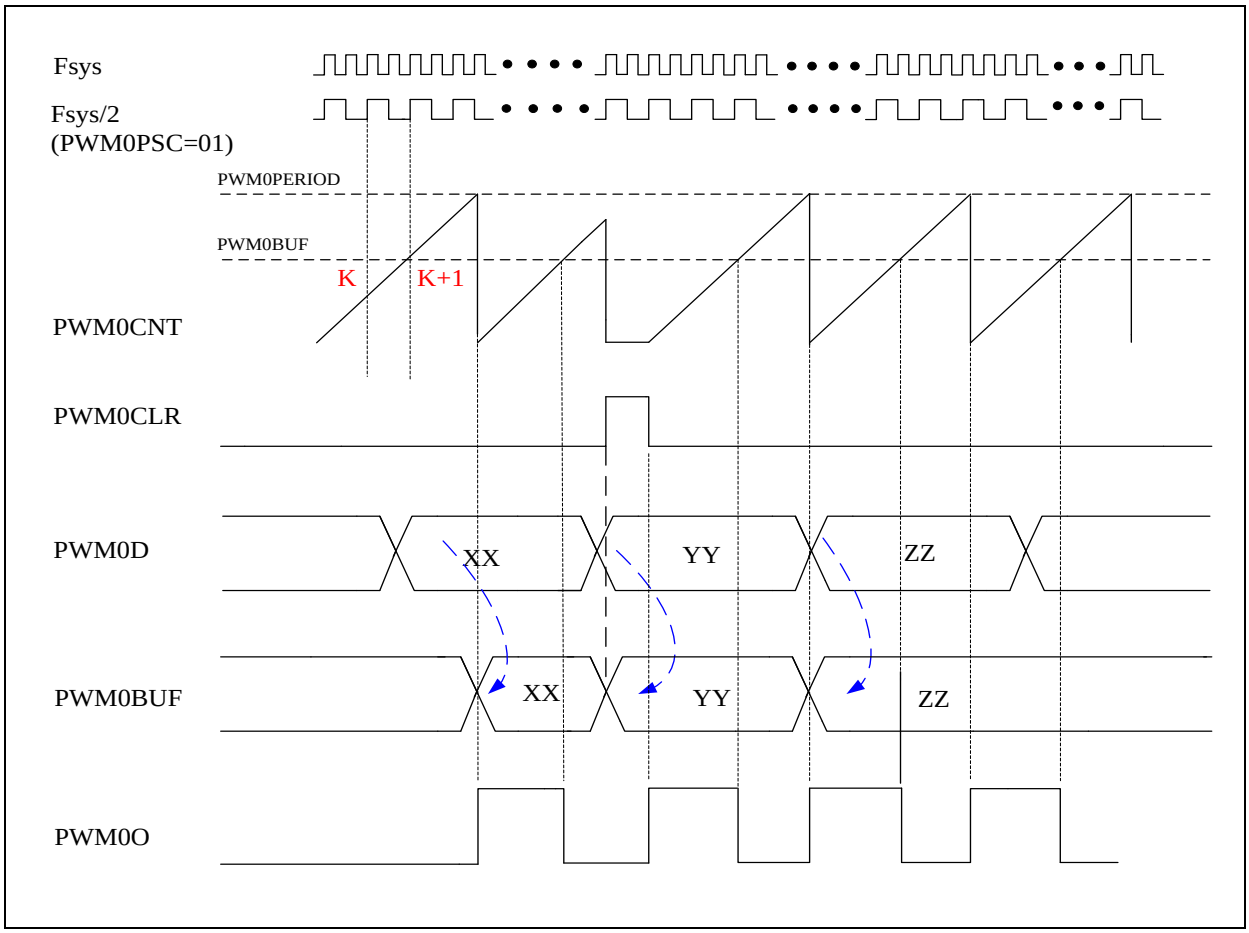
Note that the default value of PWM0CLR bit is '1'.



The next two Figures show the PWM0 waveforms. When PWM0CLR bit is set to '1', the PWM0 output is cleared to '0' no matter what its current status is. Once the PWM0CLR bit is cleared to '0', 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, the PWM0 output is set to '1' again.



PWM0 Timing (PWM0CLR before PWM0CNT reaches PWM0BUF)



PWM0 Timing (PWM0CLR after PWM0CNT over PWM0BUF)

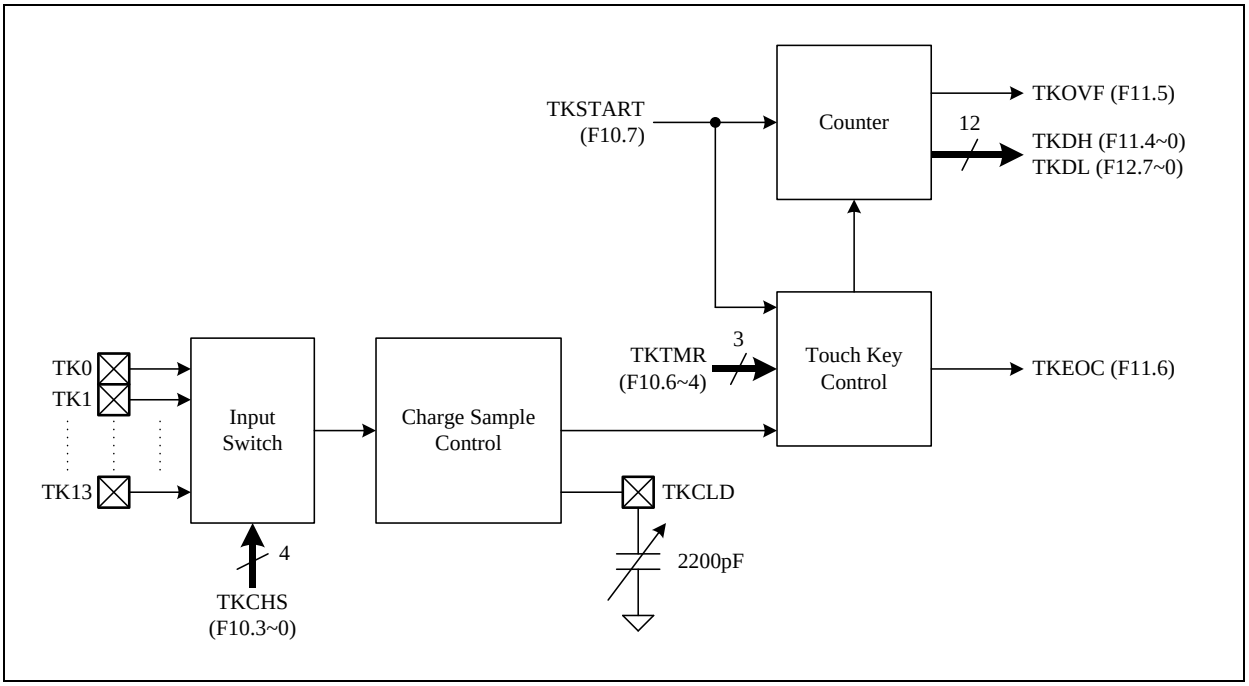
3.4 Touch Key

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

3.4.1 S/W Manual Mode Touch Key Detection

All Touch Key (TK0~TK13) can be used for S/W mode. To start the S/W mode, user assigns TKAUTO=0 and TKPD=0, then set the TKSTART (F10.7) bit to start touch key conversion, the TKSTART bit will be automatically cleared by hardware. "TKEOC=0" means conversion is in process, while "TKEOC=1" means the conversion is finish. After TKEOC's (F11.6) edge rising, user must wait at least 10 us for next conversion. The touch key counting value is stored into TKDATA[9:0] (TKDH, TKDL). If TKOVF=1, it means the conversion has exceeded in period time, reduce TKTMR (F10.6~4) to fit the range of TKDATA[11:0]. On the other hand, if TKOVF=0, but TKDATA[11:0] is too small, increase TKTMR to adapting the system board circumstances. The more detailed information, refer to touch key application note.

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.



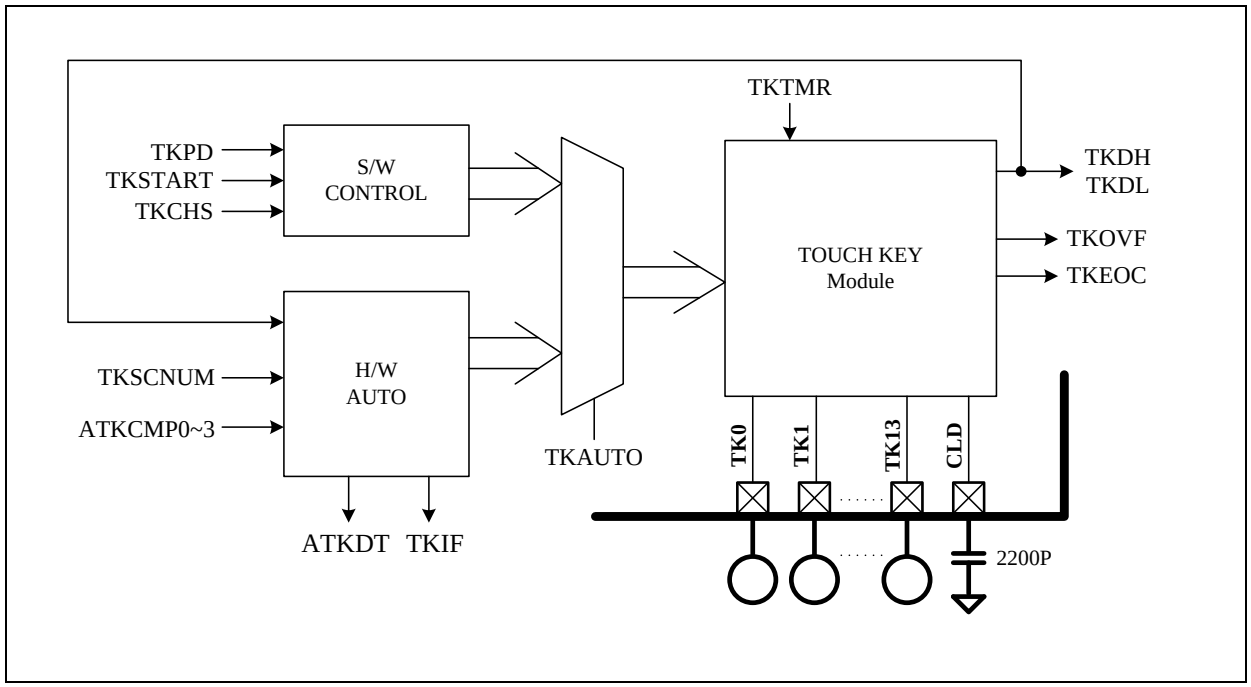
Touch Key Block Diagram

◇Example: Touch key channel = TK5 (PB3).

```
        MOVLW   00xxxxxxB       ; PBMODL[7:6] = 00B
        MOVWR   PAMODL          ; Set PA3 for touch key input


        MOVLW   0100 0011B
        MOVWF   TKCON0          ; TKTMR=4, TKCHS=3 (TK3)

        BCF     TKPD            ; TKPD=0
        :
        BSF     TKSTART         ; touch key start conversion
        NOP

WAIT_TK:
        BTFSS   TKEOC           ; wait touch key conversion is finish
        GOTO    WAIT_TK

        MOVFW   TKDH            ; read TKDATA[11:8]
        MOVFW   TKDL            ; read TKDATA[7:0]
```

3.4.2 H/W Auto Mode Touch Key Detection

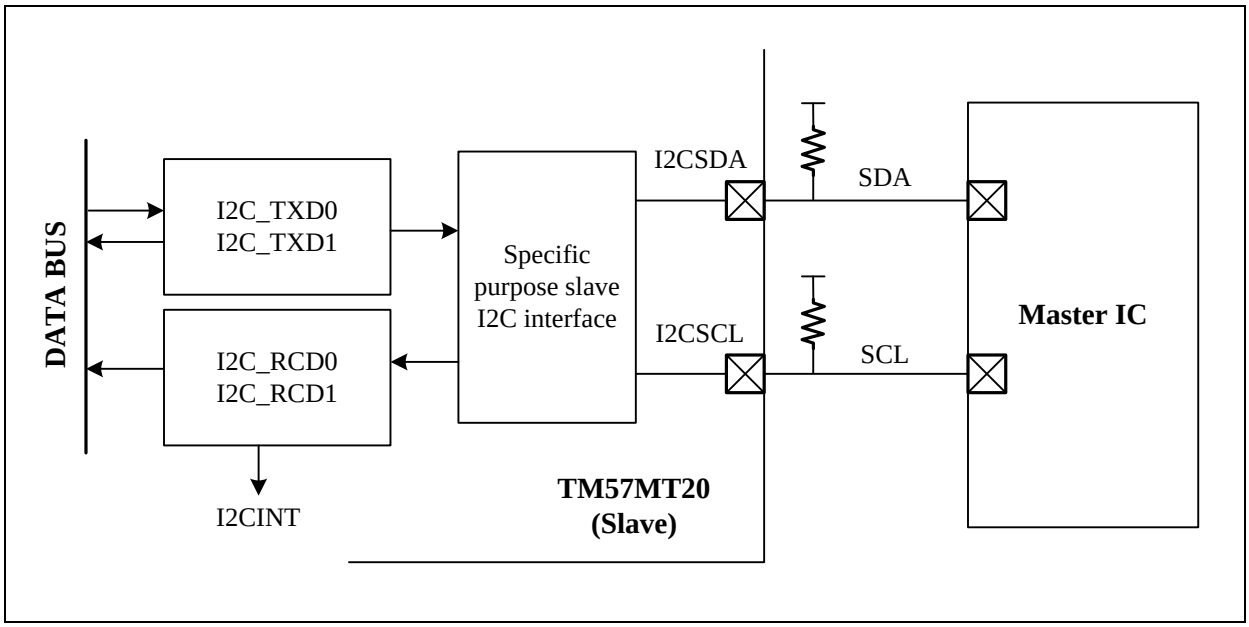
Only TK0~TK3 are eligible for H/W auto mode. This function can work in Fast/Slow/Idle mode and save the S/W effort as well as minimize the chip current consumption. To use this function, user set TKAUTO=1 to enable H/W fully control the TK module. H/W then automatically detects the TK0~TK3's TK Data Count at every 30/120/240/960 ms(WDTPCS) rate. If a Key's TK Data Count is less than the pre-set compare threshold (ATKCMP0~3), H/W generates interrupt and wake up CPU. User can switch the TK module to S/W Manual Mode after the TK interrupt and identify/confirm the Key touch event.

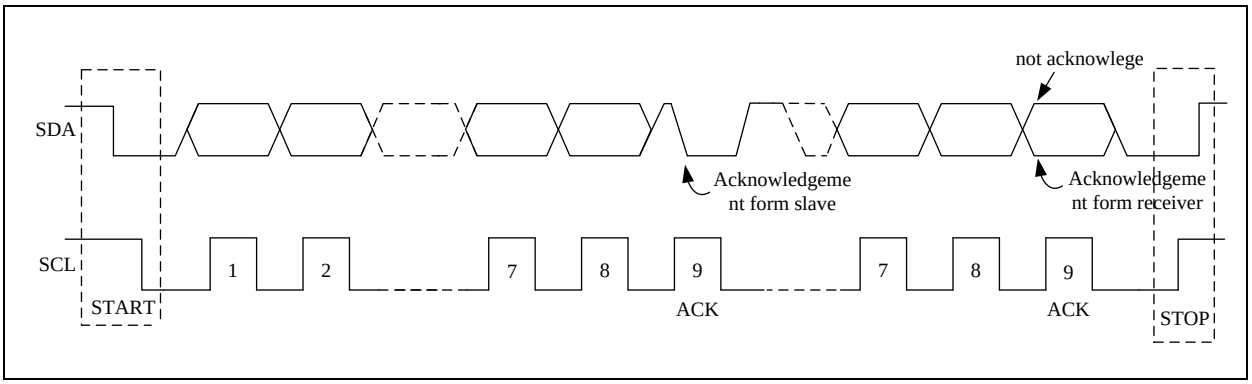


H/W Auto Mode Touch Key

3.5 Specific Purpose Slave I2C Interface

Specific purpose slave I2C interface in TM57MT20 could be used for data transmission. This interface is based on a standard I2C (Inter-Integrated Circuit), and TM57MT20 is always as a slave node. When the master node (another IC or device) sends the correct ID and Address through I2C, it can read data from the register (I2C_TXD0/ I2C_TXD1) of TM57MT20 or write data to the register (I2C_RCD0/ I2C_RCD0) of TM57MT20.





I2C Protocol

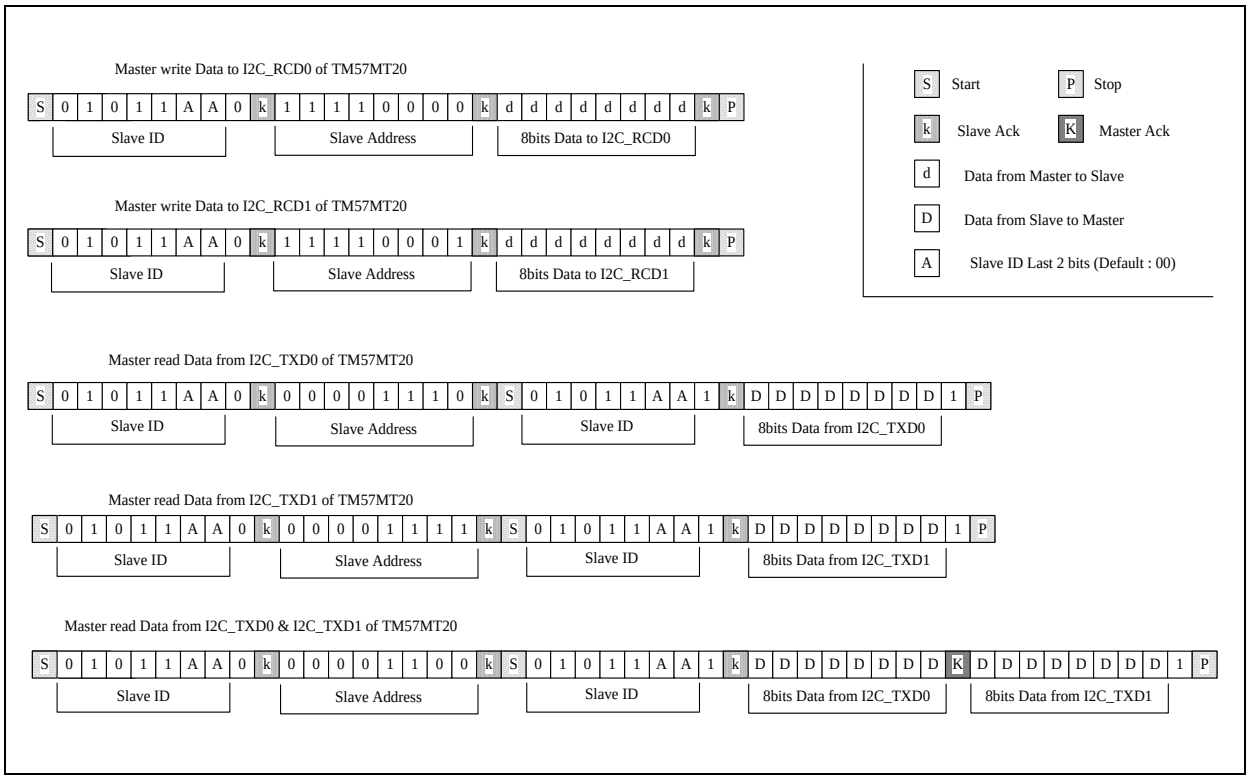
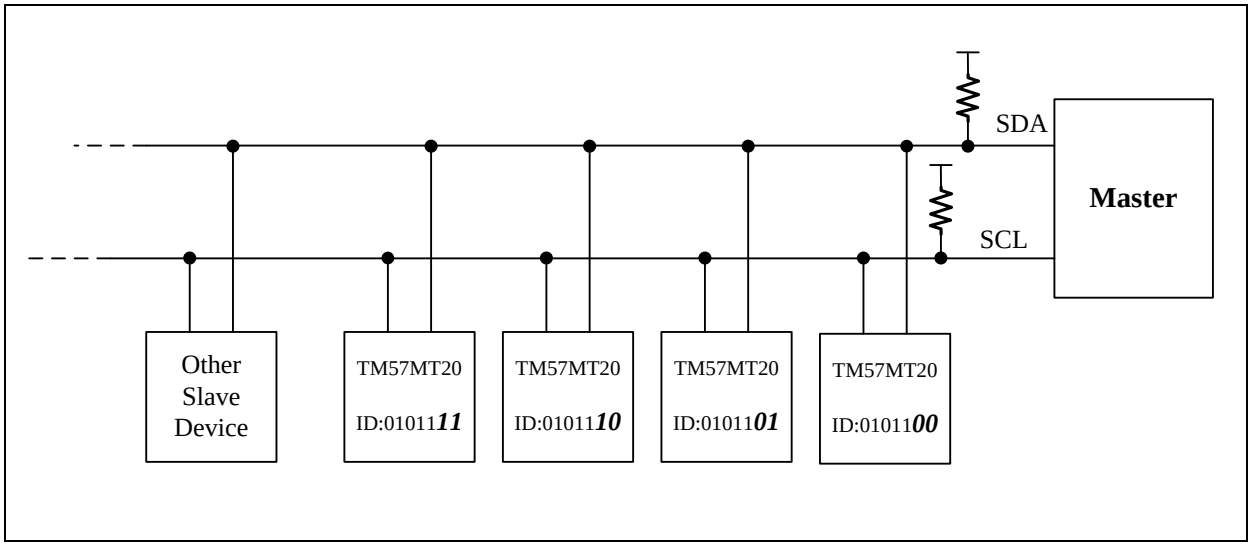


Table of TM57MT20 I2C Commands

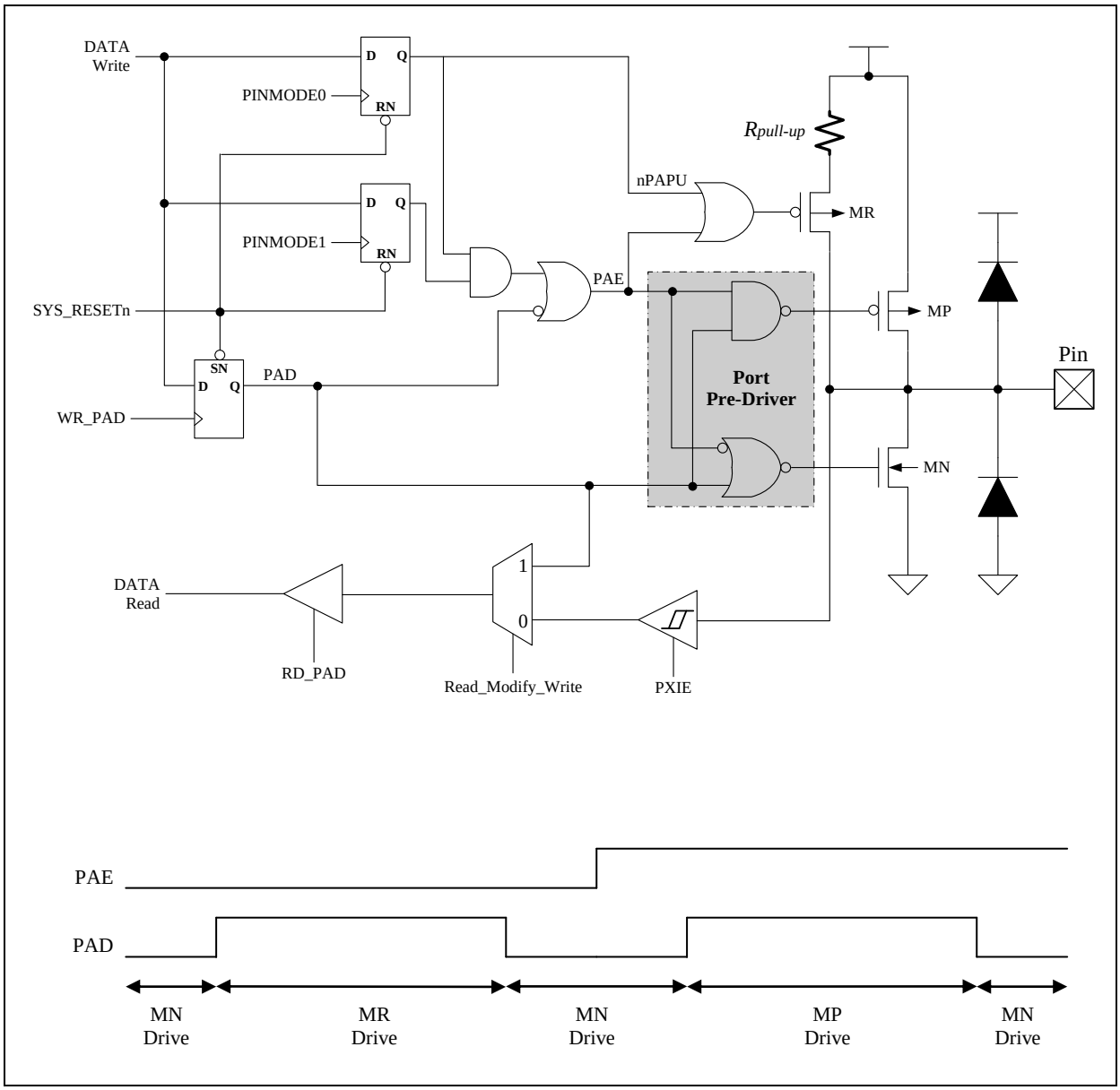


I2C Parallel Connection Application Diagram

4. I/O Port

4.1 PA0-7 & PB0-7

These pins can be used as Schmitt-trigger input or CMOS push-pull output. The pull-up resistor is assignable to each pin. User can set each pin by PINMODE0 & PINMODE1 (*Note1). Reading the pin data (PAD) 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.



Note 1: Pin Mode Setting

Mode1	Mode0	PxD	PxE	Pull-up	Wake-up	
0	0	0	Y	N	N	Open Drain output Low
0	0	1	N	Y	N	Input with Pull-up
0	1	0	Y	N	N	Open Drain output Low
0	1	1	N	N	N	Input (Reset Default)
1	0	0	Y	N	N	Open Drain output Low
1	0	1	N	Y	Y	Input with Pull-up/Wake-up
1	1	0	Y	N	N	CMOS Output Low
1	1	1	Y	N	N	CMOS Output High

(R05) PAMODH				
PAMODH	05	W	55	R05.7~6 : PA7 PIN mode settings R05.5~4 : PA6 PIN mode settings R05.3~2 : PA5 PIN mode settings R05.1~0 : PA4 PIN mode settings
(R06) PAMODL				
PAMODL	06	W	55	R06.7~6 : PA3 PIN mode settings R06.5~4 : PA2 PIN mode settings R06.3~2 : PA1 PIN mode settings R06.1~0 : PA0 PIN mode settings
(R07) PBMODH				
PBMODH	07	W	55	R07.7~6 : PB7 PIN mode settings R07.5~4 : PB6 PIN mode settings R07.3~2 : PB5 PIN mode settings R07.1~0 : PB4 PIN mode settings
(R08) PBMODL				
PBMODL	08	W	55	R08.7~6 : PB3 PIN mode settings R08.5~4 : PB2 PIN mode settings R08.3~2 : PB1 PIN mode settings R08.1~0 : PB0 PIN mode settings
(R09) PAIE				
PAIE	09	W	FF	PA7~PA0 PIN Digital input Enable 0:Disable 1:Enable
(R0A) PBIE				
PBIE	0a	W	FF	PB7~PB0 PIN Digital input Enable 0:Disable 1:Enable

Example: Setup PA0 = CMOS Output pin

```
MOVLW   01010111B
MOVWR   PAMODL      ; R06.1~0=11 , Mode1 of PA0=1, Mode0 of PA0=1,
                      PxD don't care
                    ; Select PA0= CMOS Output mode.
```

Example: Setup PB5 = Input pin with pull-up resistance and enable Low level Wake-up function of PB5

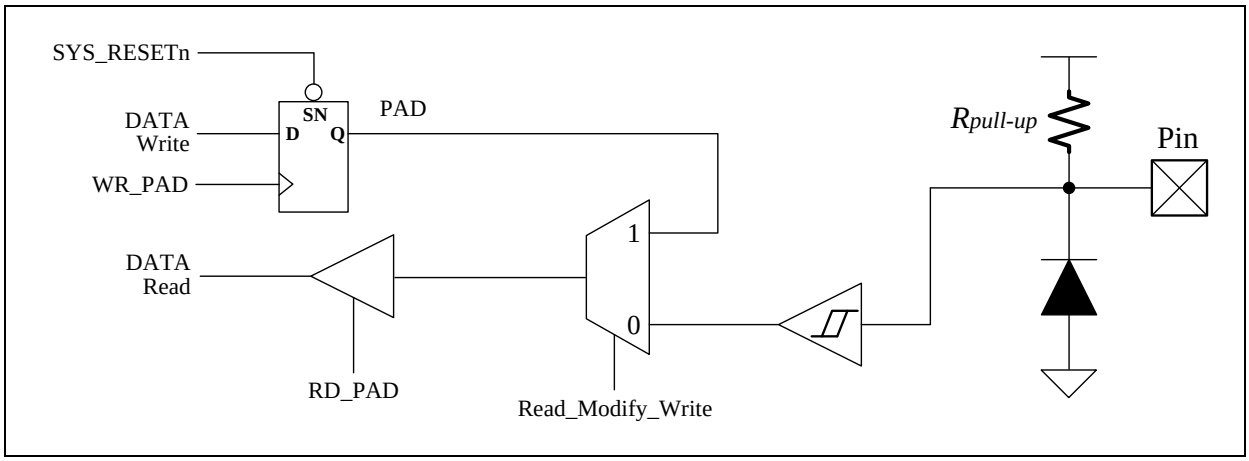
```
MOVLW   11111111B
MOVWF   PBD         ; F06.5=1, set PxD of PB5 = 1;
MOVLW   01011001B
MOVWR   PBMODH      ; R07.3~2=10, Mode1 of PB5=1, Mode0 of PB5=0
                    ; Select PB5= Input with Pull-up/Wake-up
```

Example: Setup PA0 = Touch Key pin

```
MOVLW   01010101B
MOVWR   PAMODL      ; R06.1~0=01 , Mode1 of PA0=0, Mode0 of PA0=1, PAD[0]=1
MOVLW   11111110B
MOVWR   PAIE        ;  Disable PA0 Digital input, for Touch Key.
```

4.2 VPP

VPP can be only used in Schmitt-trigger input mode. The pull-up resistor is always connected to this pin.



MEMORY MAP

F-Plane

Name	Address	R/W	Rst	Description
(F00) INDF				
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address contains in the FSR register.
(F01) TM0				
TM0	01.7~0	R/W	0	Timer 0
(F02) PCL				
PCL	02.7~0	R/W	0	Program Counter LSB[7~0]
(F03) STATUS				
GB0	03.6	R/W	0	General purpose bit
RAMBK	03.5	R/W	0	RAM bank select
TO	03.4	R	0	WDT time out flag
PD	03.3	R	0	STOP mode flag
Z	03.2	R/W	0	Zero Flag
DC	03.1	R/W	0	Decimal Carry Flag
C	03.0	R/W	0	Carry Flag
(F04) FSR				
FSR	04.7~0	R/W	0	F-Plane File Select Register
(F05) PAD				
PAD	05.7~0	R	-	Port A pin or "data register" state
		W	ff	Port A data output register
(F06) PBD				
PBD	06.7~0	R	-	Port B pin or "data register" state
		W	ff	Port B data output register
(F08) INTIE				
I2CIE	08.7	R/W	0	PWM0 Interrupt Enable, 1=Enable, 0=Disable
ATKIE	08.6	R/W	0	Auto Touch Key Interrupt Enable , 1=Enable, 0=Disable
INT3IE	08.5	R/W	0	INT2 (PB7) falling Interrupt Enable, 1=Enable, 0=Disable
TM0IE	08.4	R/W	0	Timer0 Interrupt Enable, 1=Enable, 0=Disable
WKTIE	08.3	R/W	0	Wake-up Timer Interrupt Enable, 1=Enable, 0=Disable
INT2IE	08.2	R/W	0	INT2 (PB6) falling Interrupt Enable, 1=Enable, 0=Disable
INT1IE	08.1	R/W	0	INT1 (VPP) falling Interrupt Enable, 1=Enable, 0=Disable
INT0IE	08.0	R/W	0	INT0 (PB4) falling/rising Interrupt Enable, 1=Enable, 0=Disable

Name	Address	R/W	Rst	Description
(F09) INTIF				
I2CIF	09.7	R	-	I2C interrupt event pending flag, set by H/W while I2C_RCD0 or I2C_RCD1 finished receiving new data.
		W	0	write 0: clear this flag; write 1: no action.
ATKINT	0.9.6	R	-	Auto Touch Key interrupt event pending flag, set by H/W while Key's TK Data Count is less than the pre-set compare threshold
		W	0	write 0: clear this flag; write 1: no action
INT3IF	0.9.6	R	-	INT3 pin falling interrupt pending flag, set by H/W at INT3 pin's falling edge.
		W	0	write 0: clear this flag; write 1: no action
TM0IF	09.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows.
		W	0	write 0: clear this flag; write 1: no action.
WKTIF	09.3	R	-	WKT interrupt event pending flag, set by H/W while WKT is timeout.
		W	0	write 0: clear this flag; write 1: no action.
INT2IF	09.2	R	-	INT2 pin falling interrupt pending flag, set by H/W at INT2 pin's falling edge.
		W	0	write 0: clear this flag; write 1: no action.
INT1IF	09.1	R	-	INT1 pin falling interrupt pending flag, set by H/W at INT1 pin's falling edge.
		W	0	write 0: clear this flag; write 1: no action
INT0IF	09.0	R	-	INT0 pin falling/rising interrupt pending flag, set by H/W at INT0 pin's falling/rising edge..
		W	0	write 0: clear this flag; write 1: no action
(F0C) PWM0D				
PWM0D	0c.7~0	R/W	0	PWM0 duty 8-bit
(F0D) MF0D				
EMI_IMPRV	0d.2	R/W	2	EMI improve 0:Off 1:On
PWM0CLR	0d.1	R/W	1	PWM0 clear and hold 0:Release 1:Clear and hold
TM0STP	0d.0	R/W	0	Timer0 counter stop 0:Release 1:Stop counting
(F0F) CLKCON				
FASTSTP	0f.4	R/W	0	Fast-clock Enable / Disable 0:Enable 1:Disable
CPUCKS	0f.3	R/W	0	System clock (Fsys) selection 0:Fast-clock 1:Slow-clock
SLOWEN	0f.2	R/W	0	When CPUCKS=1,this bit don't care; When CPUCKS=0, then 1:Enable 'Slow-clock' oscillating 0: Stop 'Slow-clock' oscillating
SIRCKS	0f.1~0	R/W	11	SIRC select. 00:32 KHz 01:16 KHz 10:8 KHz 11:4 KHz
(F10) TKCON0				
TKSTART	10.7	R/W	0	Touch key START of conversion
TKTMR	10.6~4	R/W	4	Touch Key Conversion Time. 0=shortest, 7=longest
TKCHS	10.3~0	R/W	f	Touch Key Channel Select

Name	Address	R/W	Rst	Description
(F11) TKCON1				
TKPD	11.7	R/W	1	Touch Key Power Down, No Effect in Auto Scan Mode
TKEOC	11.6	R	-	Touch Key End of Conversion
TKOVF	11.5	R	-	Touch Key Counter Overflow
TKDH	11.4~0	R	-	Touch Key Counter Data [12:8]
(F12) TKDL				
TKDL	12.7~0	R	-	Touch Key Counter Data [7:0]
(F13) MF13				
I2CEN	13.7	R/W	0	Slave I2C interface Enable 0:Disable 1:Enable
TKAUTO	13.6	R/W	0	Touch Key Auto Scan Mode Enable 0:Disable 1:Enable
TKSCNUM	13.5~4	R/W	11	Touch Key Auto Scan Number. 00=1 channel, 01=2 channel, 10=3 channel, 11=4 channel
ATKDT	13.3~0	R		Auto Touch Key Scan Result [TK3~0] 0: No touch 1: Touch
(F15) ATKCMP0				
ATKCMP0	15.7~0	R/W	0	Auto Touch Key CH0 Comparison Value
(F16) ATKCMP1				
ATKCMP1	16.7~0	R/W	0	Auto Touch Key CH1 Comparison Value
(F17) ATKCMP2				
ATKCMP2	17.7~0	R/W	0	Auto Touch Key CH2 Comparison Value
(F18) ATKCMP3				
ATKCMP3	18.7~0	R/W	0	Auto Touch Key CH3 Comparison Value
(F1A)I2C_RCD0				
I2C_RCD0	1a.7~0	R	0	The Receiving register 0 of Slave I2C
(F1B)I2C_RCD1				
I2C_RCD1	1b.7~0	R	0	The Receiving register 1 of Slave I2C
(F1C)I2C_TXD0				
I2C_TXD0	1c.7~0	R/W	0	The Transmission register 0 of Slave I2C
(F1D)I2C_TXD1				
I2C_TXD1	1d.7~0	R/W	0	The Transmission register 1 of Slave I2C
(F1E)I2C_ID				
I2C_ID	1e.7~0	R/W	0	Slave I2C ID last 2 bits
SRAM	20~ff	R/W	-	SRAM

R-Plane

Name	Address	R/W	Rst	Description
(R01) TM0RLD				
TM0RLD	01.7~0	W	0	Timer0 reload Data
(R02) TM0CTL				
TM0CL	02.7	W	0	Timer0 Capture polarity. 0: High level capture, 1: Low level capture
TM0CM	02.6	W	0	1: Timer0 works in CAPTURE Mode; 0: Timer0 works in COUNTER Mode
TM0EDG	02.5	W	0	1: TM0CKI falling edge; 0: TM0CKI rising edge for Timer0 Prescaler count
TM0CKS	02.4	W	0	1: TM0CKI as Timer0 Prescaler clock; 0: Instruction Cycle as Timer0 Prescaler clock
TM0PSC	02.3~0	W	0	Timer0 Pre-Scale 0000: div1 0001: div2 0010: div4 0011: div8 0100: div16 0101: div32 0110: div64 0111: div128 1xxx: div256
(R03) PWRDN				
PWRDN	03	W		Write this register to enter STOP Mode.
(R04) WDTCLR				
WDTCLR	04	W		Write this register to clear WDT.
(R05) PAMODH				
PAMODH	05	W	55	05.7~6 : PA7 PIN mode settings 05.5~4 : PA6 PIN mode settings 05.3~2 : PA5 PIN mode settings 05.1~0 : PA4 PIN mode settings * Note 1 (Refer section 4-1)
(R06) PAMODL				
PAMODL	06	W	55	PA3~PA0 PIN mode settings * Note 1
(R07) PBMODH				
PBMODH	07	W	55	PB7~PB4 PIN mode settings * Note 1
(R08) PBMODL				
PBMODL	08	W	55	PB3~PB0 PIN mode settings * Note 1
(R09) PAIE				
PAIE	09	W	FF	PA7~PA0 PIN Digital input Enable 0:Disable 1:Enable
(R0A) PBIE				
PBIE	0a	W	FF	PB7~PB0 PIN Digital input Enable 0:Disable 1:Enable

Name	Address	R/W	Rst	Description
(R0B) PWM0CON				
PWM0OE	0b.4	W	0	PWM0 output to PB5 pin enable
PWM0NOE	0b.3	W	0	PWM0 negative output to PB5 pin. 1:negative 0:positive
	0b.2	-	-	Reserved
PWM0PSC	0b.1~0	W	0	PWM0 prescaler 0:div1(TC/2), 1:div2(TC), 2:div4(2TC), 3:div8 (4TC)
(R0C) MR0C				
INT0EDG	0c.2	W	0	0=INT0(PA0) pin falling generate Interrupt; 1=rising
CLKNOE	0c.1	W	0	CLK negative output select. 1:negative 0:positive
CLKOE	0c.0	W	0	CLK Output to PB4. 0:Disable 1:Enable
(R0D) PWM0PRD				
PWM0PRD	0d	W	ff	PWM0 Period. ff=256
(R0E) MR0E				
WDTPSC	0e.6~5	W	01	WDT/WKT timeout, ATK Scanning Frequency 00:30 ms 01:120 ms 10:240 ms 11:960 ms
(R0F) TESTREG				
TESTREG	0f.1~0	W	0	TEST Reg

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 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 A : W = 0xD9, FSR = 0xC2

ANDLW	Logical AND Literal "k" with W	
Syntax	ANDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) ‘AND’ (k)	
Status Affected	Z	
OP-Code	01 1011 kkkk kkkk	
Description	The contents of W register are AND’ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	ANDLW 0x5F	B : W = 0xA3 A : W = 0x03

ANDWF	AND W with “f”	
Syntax	ANDWF f [,d]	
Operands	f : 00h ~ 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 A : W = 0x17, FSR = 0x02

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

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

BTFSC	Test "b" bit of "f", skip if clear(0)	
Syntax	BTFSC f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 0	
Status Affected	-	
OP-Code	01 010b bbff ffff	
Description	If bit 'b' in register 'f' is '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 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = FALSE if FLAG.1 = 1, PC = TRUE

BTFSS	Test "b" bit of "f", skip if set(1)	
Syntax	BTFSS f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 1	
Status Affected	-	
OP-Code	01 011b bbff ffff	
Description	If bit 'b' in register 'f' is '0', then the next instruction is executed. If bit 'b' in register 'f' is '1', then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTFSS FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = TRUE if FLAG.1 = 1, PC = FALSE

CALL	Call subroutine "k"	
Syntax	CALL k	
Operands	K : 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 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 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 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 = ? 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 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 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 GOTO LOOP CONTINUE	B : PC = LABEL1 A : CNT = CNT – 1 if CNT=0, PC = CONTINUE if CNT≠0, PC = LABEL1+1

GOTO Unconditional Branch

Syntax	GOTO k
Operands	k : 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 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 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 GOTO LOOP A : CNT = CNT + 1 CONTINUE if CNT=0, PC = CONTINUE if CNT≠0, PC = LABEL1+1

IORLW Inclusive OR Literal with W

Syntax	IORLW k
Operands	k : 00h ~ FFh
Operation	(W) ← (W) OR k
Status Affected	Z
OP-Code	01 1010 kkkk kkkk
Description	The contents of the W register 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 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 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 = ? 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 = ? 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 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 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	-

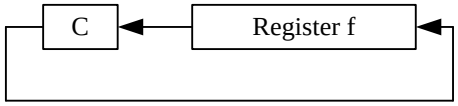
RETI	Return from Interrupt	
Syntax	RETI	
Operands	-	
Operation	PC ← TOS, GIE ← 1	
Status Affected	-	
OP-Code	00 0000 0110 0000	
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is a two-cycle instruction.	
Cycle	2	
Example	RETI	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 : TABLE ADDWF PCL,1 RETLW k1 RETLW k2 : RETLW kn	B : W = 0x07 A : W = value of k8

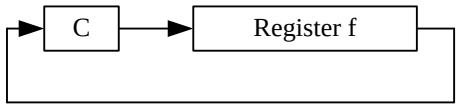
RET Return from Subroutine

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 A : REG1 = 1110 0110 W = 1100 1100, C = 1

RRF Rotate Right “f” through Carry

Syntax	RRF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 1100 dfff ffff
Description	The contents of register ‘f’ are rotated one bit to the right through the Carry Flag. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.
Cycle	1
Example	RRF REG1,0 B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 0111 0011, C = 0

SLEEP	Go into 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 = ? A : REG1 = 1, W = 2, C = 1, Z = 0
	SUBWF REG1,1	B : REG1 = 2, W = 2, C = ?, Z = ? A : REG1 = 0, W = 2, C = 1, Z = 1
	SUBWF REG1,1	B : REG1 = 1, W = 2, C = ?, Z = ? 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 A : REG1 = 0xA5, W = 0x5A

TESTZ	Test if “f” is zero	
Syntax	TESTZ f	
Operands	f : 00h ~ 7Fh	
Operation	Set Z flag if (f) is 0	
Status Affected	Z	
OP-Code	00 1000 1fff ffff	
Description	If the content of register ‘f’ is 0, Zero flag is set to 1.	
Cycle	1	
Example	TESTZ REG1	B : REG1 = 0, Z = ? A : REG1 = 0, Z = 1

XORLW	Exclusive OR Literal with W	
Syntax	XORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) XOR k	
Status Affected	Z	
OP-Code	01 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 A : W = 0x1A

XORWF	Exclusive OR W with “f”	
Syntax	XORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	00 0110 dfff ffff	
Description	Exclusive OR the contents of the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	XORWF REG, 1	B : REG = 0xAF, W = 0xB5 A : REG = 0x1A, W = 0xB5

Electrical Characteristics

1. Absolute Maximum Ratings (T_A = 25℃)

Parameter	Rating	Unit
Supply voltage	V_{SS} - 0.3 to V_{SS} + 3.6	V
Input voltage	V_{SS} – 0.3 to V_{CC} + 0.3	
Output voltage	V_{SS} – 0.3 to V_{CC} + 0.3	
Output current high per 1 PIN	-25	mA
Output current high per all PIN	-80	
Output current low per 1 PIN	+30	
Maximum Operating Voltage	3.6	V
Operating temperature	-40 to +85	℃
Storage temperature	-65 to +150	

2. DC Characteristics (TA = 25℃, VCC = 2.0V to 3.6V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Input High Voltage	V_{IH}	All Input Except PA7	V_{CC} = 3.0V	0.55 V_{CC}	0.6 V_{CC}		V
Input High Voltage	V_{IH}	PA7	V_{CC} = 3.0V	0.55 V_{CC}	0.6 V_{CC}		V
Input Low Voltage	V_{IL}	All Input Except PA7	V_{CC} = 3.0V		0.33 V_{CC}	0.4V_{CC}	V
Input Low Voltage	V_{IL}	PA7	V_{CC} = 3.0V		0.45 V_{CC}	0.5 V_{CC}	
Output High Current	I_{OH}	All Output	V_{CC} = 3.0V V_{OH} = 2.7V		6		mA
Output Low Current	I_{OL}	All Output	V_{CC} = 3.0V V_{OL} = 0.3V		18		mA
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}	–	–	1	uA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0 V	–	–	-1	uA
Output Leakage Current (pin high)	I_{OLH}	All Output	V_{OUT} = V_{CC}	–	–	2	uA
Output Leakage Current (pin low)	I_{OLL}	All Output	V_{OUT} = 0 V	–	–	-2	uA
Operating Supply Current	I_{CC}	Run 3 MHz	V_{CC} = 3.0 V	–	2.0	–	mA
		Auto Touch mode (LVR enable)	V_{CC} = 3.0V	–	3.5	–	uA
		Auto Touch mode (LVR disable)	V_{CC} = 3.0V	–	2	–	uA
Stop Current	I_{CC}	Stop mode	V_{CC} = 3.0V	–	0.1	–	uA
Pull-Up Resistor	R_P	V_{IN} = 0 V Ports A	V_{CC} = 3.0V	65	130	200	kΩ
		PA7	V_{CC} = 3.0V	40	72	120	kΩ

3. Timing Characteristics (T_A = -40℃ to +85℃)

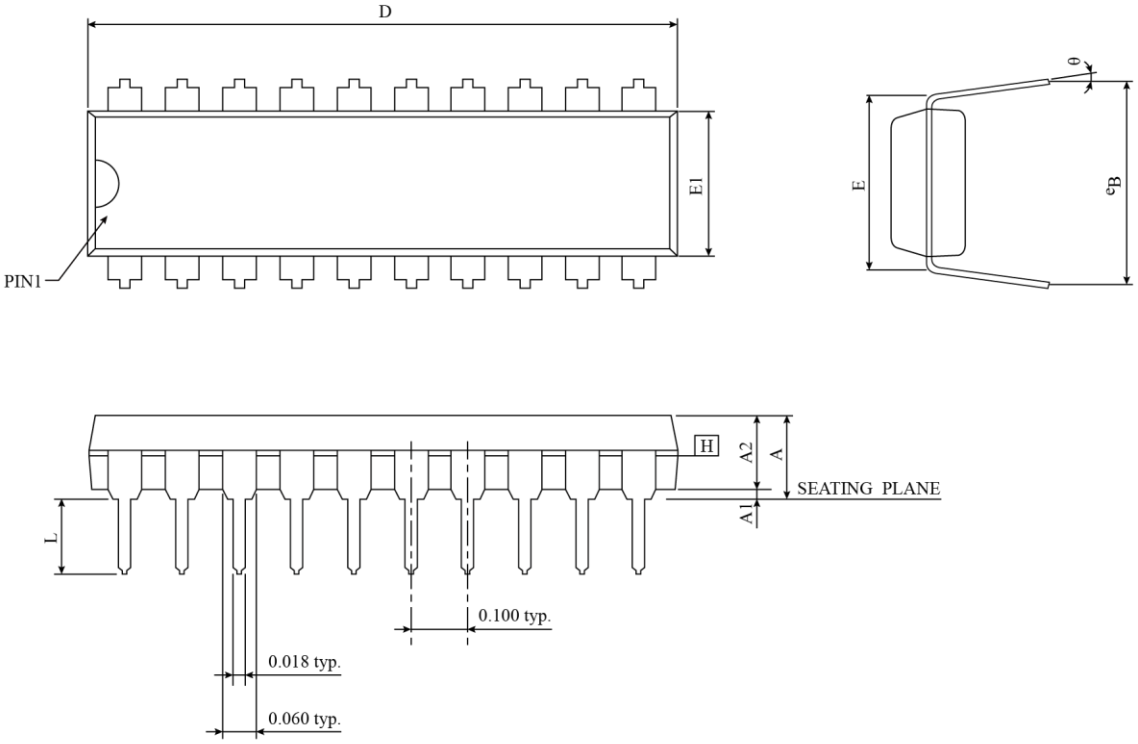
Parameter	Condition	Min	Typ	Max	Unit
FIRC Frequency	-40℃ ~ 85℃, V_{CC} = 3.0 V	-30%	3	+30%	MHz
	25℃, V_{CC} = 3.0 V	-30%	3	+30%	
	25℃, V_{CC} = 1.8 ~ 3.6 V	-50%	3	+50%	
RESET Input Low width	Input V_{CC} = 3 V ± 10 %	3	–	–	μs
WDT time	V_{CC} = 3 V, WDTPSC = 01	-30%	120	+30%	ms
CPU start up time	V_{CC} = 3 V	–	17	–	ms

Packaging information

The ordering information:

Ordering number	Package
TM57MT20-MTP-05	DIP 20-pin (300 mil)

20-DIP Package Dimension

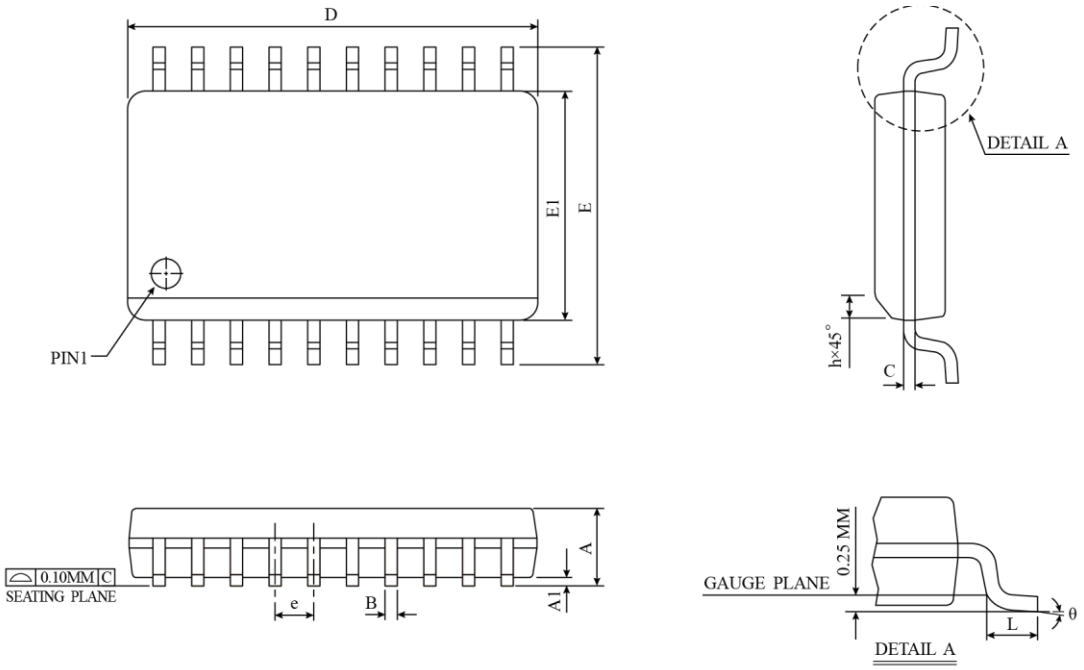


SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN	MAX	MIN	MAX
A	-	4.445	-	0.175
A1	0.381	-	0.015	-
A2	3.175	3.429	0.125	0.135
D	25.705	26.416	1.012	1.040
E	7.620	7.874	0.300	0.310
E1	6.223	6.477	0.245	0.255
L	3.048	3.556	0.120	0.140
eB	8.509	9.525	0.335	0.375
θ	0°	15°	0°	15°
JEDEC	MS-001 (AD)			

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	MAX	MIN	MAX
A	2.35	2.65	0.0926	0.1043
A1	0.10	0.30	0.0040	0.0118
B	0.33	0.51	0.013	0.020
C	0.23	0.32	0.0091	0.0125
D	12.60	13.00	0.4961	0.5118
E	10.00	10.65	0.394	0.491
E1	7.40	7.60	0.2914	0.2992
e	1.27 BSC		0.050 BSC	
h	0.25	0.75	0.010	0.029
L	0.40	1.27	0.016	0.050
θ	0°	8°	0°	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.