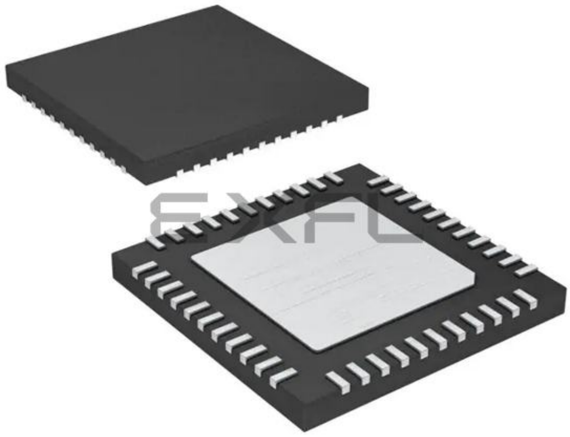


Welcome to E-XFL.COM

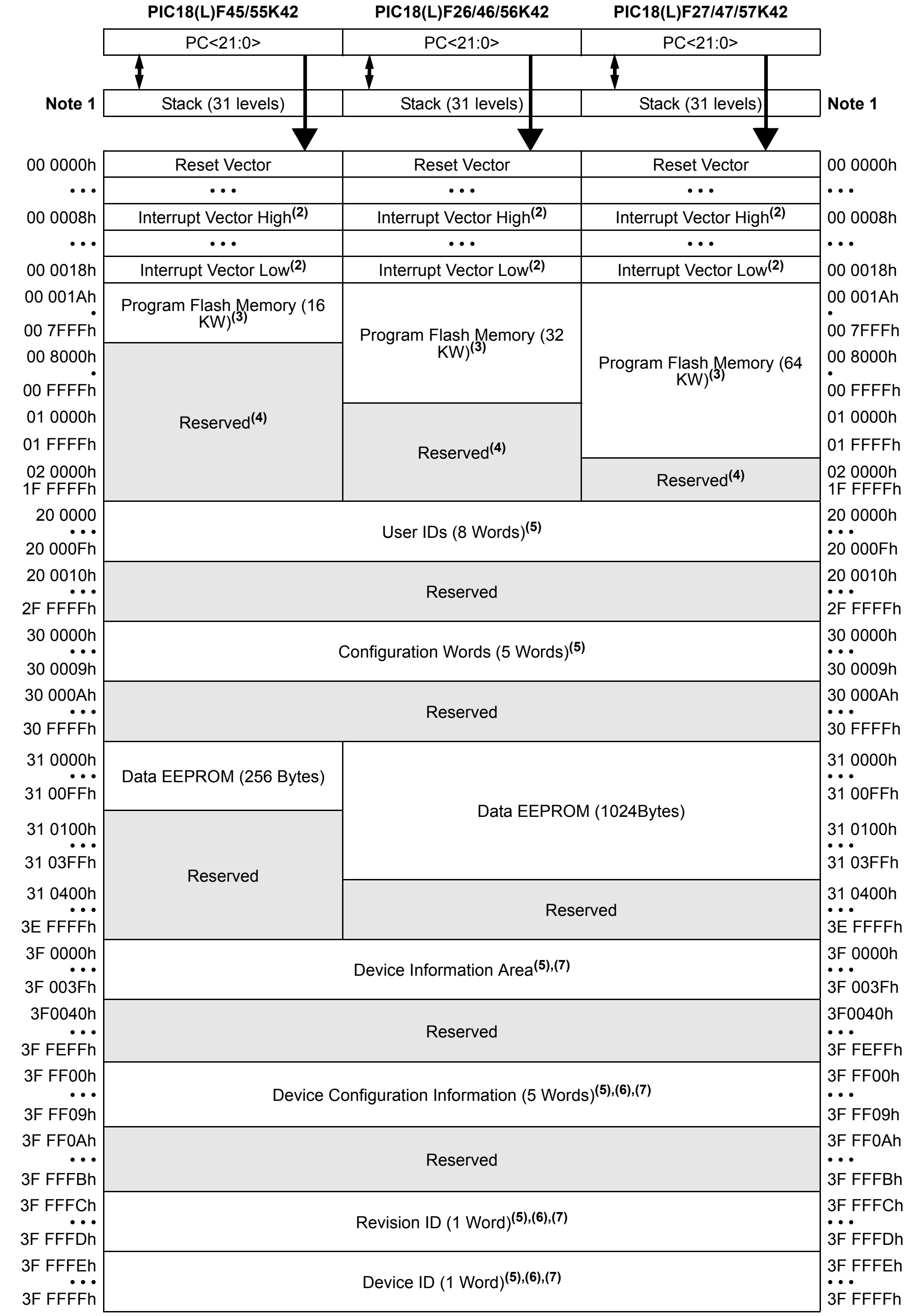
## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 64MHz |
| Connectivity | I²C, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT |
| Number of I/O | 36 |
| Program Memory Size | 128KB (64K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 1K x 8 |
| RAM Size | 8K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.3V ~ 5.5V |
| Data Converters | A/D 35x12b; D/A 1x5b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-VQFN Exposed Pad |
| Supplier Device Package | 44-QFN (8x8) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18f47k42t-i-ml |

## TABLE 4-1: PROGRAM AND DATA EEPROM MEMORY MAP



| | PIC18(L)F45/55K42 | PIC18(L)F26/46/56K42 | PIC18(L)F27/47/57K42 | |
|---|---|---|---|---|
| | PC<21:0> | PC<21:0> | PC<21:0> | |
| Note 1 | Stack (31 levels) | Stack (31 levels) | Stack (31 levels) | Note 1 |
| 00 0000h | Reset Vector | Reset Vector | Reset Vector | 00 0000h |
| • • • | • • • | • • • | • • • | • • • |
| 00 0008h | Interrupt Vector High(2) | Interrupt Vector High(2) | Interrupt Vector High(2) | 00 0008h |
| • • • | • • • | • • • | • • • | • • • |
| 00 0018h | Interrupt Vector Low(2) | Interrupt Vector Low(2) | Interrupt Vector Low(2) | 00 0018h |
| 00 001Ah<br>•<br>00 7FFFh | Program Flash Memory (16 KW)(3) | Program Flash Memory (32 KW)(3) | Program Flash Memory (64 KW)(3) | 00 001Ah<br>•<br>00 7FFFh |
| 00 8000h<br>•<br>00 FFFFh | Reserved(4) | | | 00 8000h<br>•<br>00 FFFFh |
| 01 0000h<br>01 FFFFh | | Reserved(4) | | 01 0000h<br>01 FFFFh |
| 02 0000h<br>1F FFFFh | | | Reserved(4) | 02 0000h<br>1F FFFFh |
| 20 0000<br>• • •<br>20 000Fh | User IDs (8 Words)(5) | | | 20 0000h<br>• • •<br>20 000Fh |
| 20 0010h<br>• • •<br>2F FFFFh | Reserved | | | 20 0010h<br>• • •<br>2F FFFFh |
| 30 0000h<br>• • •<br>30 0009h | Configuration Words (5 Words)(5) | | | 30 0000h<br>• • •<br>30 0009h |
| 30 000Ah<br>• • •<br>30 FFFFh | Reserved | | | 30 000Ah<br>• • •<br>30 FFFFh |
| 31 0000h<br>• • •<br>31 00FFh | Data EEPROM (256 Bytes) | Data EEPROM (1024Bytes) | | 31 0000h<br>• • •<br>31 00FFh |
| 31 0100h<br>• • •<br>31 03FFh | Reserved | | | 31 0100h<br>• • •<br>31 03FFh |
| 31 0400h<br>• • •<br>3E FFFFh | | Reserved | | 31 0400h<br>• • •<br>3E FFFFh |
| 3F 0000h<br>• • •<br>3F 003Fh | Device Information Area(5),(7) | | | 3F 0000h<br>• • •<br>3F 003Fh |
| 3F0040h<br>• • •<br>3F FEFFh | Reserved | | | 3F0040h<br>• • •<br>3F FEFFh |
| 3F FF00h<br>• • •<br>3F FF09h | Device Configuration Information (5 Words)(5),(6),(7) | | | 3F FF00h<br>• • •<br>3F FF09h |
| 3F FF0Ah<br>• • •<br>3F FFFBh | Reserved | | | 3F FF0Ah<br>• • •<br>3F FFFBh |
| 3F FFFCh<br>• • •<br>3F FFFDh | Revision ID (1 Word)(5),(6),(7) | | | 3F FFFCh<br>• • •<br>3F FFFDh |
| 3F FFFEh<br>• • •<br>3F FFFFh | Device ID (1 Word)(5),(6),(7) | | | 3F FFFEh<br>• • •<br>3F FFFFh |

**Note**
1: The stack is a separate SRAM panel, apart from all user memory panels.
2: 00 0008h location is used as the reset default for the IVTBASE register, the vector table can be relocated in the memory by programming the IVTBASE register.
3: Storage area Flash is implemented as the last 128 Words of user Flash.
4: The addresses do not roll over. The region is read as ‘0’.
5: Not code-protected.
6: Hard-coded in silicon.
7: This region cannot be written by the user and it’s not affected by a Bulk Erase.

### REGISTER 5-4: CONFIGURATION WORD 2H (30 0003h)

| R/W-1 | U-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{XINST}}$ | — | $\overline{\text{DEBUG}}$ | STVREN | PPS1WAY | $\overline{\text{ZCD}}$ | BORV<1:0>[1] | |
| bit 7 | | | | | | | bit 0 |

| | | | |
|---|---|---|---|
| **Legend:** | | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘1’ | |
| -n = Value for blank device | ‘1’ = Bit is set | ‘0’ = Bit is cleared | x = Bit is unknown |

bit 7 **$\overline{\text{XINST}}$:** Extended Instruction Set Enable bit
`1` = Extended instruction set and Indexed Addressing mode are disabled (Legacy mode)
`0` = Extended instruction set and Indexed Addressing mode are enabled

bit 6 **Unimplemented:** Read as ‘`1`’

bit 5 **$\overline{\text{DEBUG}}$:** Debugger Enable bit
`1` = Background debugger is disabled
`0` = Background debugger is enabled

bit 4 **STVREN:** Stack Overflow/Underflow Reset Enable bit
`1` = Stack Overflow or Underflow will cause a Reset
`0` = Stack Overflow or Underflow will not cause a Reset

bit 3 **PPS1WAY:** PPSLOCKED One-Way Set Enable bit
`1` = PPSLOCKED bit can be cleared and set only once; PPS registers remain locked after one clear/set cycle
`0` = PPSLOCKED bit can be set and cleared multiple times (subject to the unlock sequence)

bit 2 **$\overline{\text{ZCD}}$:** Zero-Cross Detect Enable bit
`1` = ZCD is disabled; ZCD can be enabled by setting the bit SEN of the ZCDCON register
`0` = ZCD is always enabled

bit 1-0 **BORV<1:0>:** Brown-out Reset Voltage Selection bits[1]
PIC18FXXK42 Devices:
`11` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.45V
`10` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.45V
`01` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.7V
`00` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.85V
PIC18LFXXK42 Device:
`11` = Brown-out Reset Voltage ($V_{BOR}$) is set to 1.90V
`10` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.45V
`01` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.7V
`00` = Brown-out Reset Voltage ($V_{BOR}$) is set to 2.85V

**Note 1:** The higher voltage setting is recommended for operation at or above 16 MHz.

## 9.0 INTERRUPT CONTROLLER

The Vectored Interrupt Controller module reduces the numerous peripheral interrupt request signals to a single interrupt request signal to the CPU. This module includes the following major features:

- Interrupt Vector Table (IVT) with a unique vector for each interrupt source
- Fixed and ensured interrupt latency
- Programmable base address for Interrupt Vector Table (IVT) with lock
- Two user-selectable priority levels – High priority and Low priority
- Two levels of context saving
- Interrupt state status bits to indicate the current execution status of the CPU

The Interrupt Controller module assembles all of the interrupt request signals and resolves the interrupts based on both a fixed natural order priority (i.e., determined by the Interrupt Vector Table), and a user-assigned priority (i.e., determined by the IPRx registers), thereby eliminating scanning of interrupt sources.

### 9.1 Interrupt Control and Status Registers

The devices in this family implement the following registers for the interrupt controller:

- INTCON0, INTCON1 Control Registers
- PIRx – Peripheral Interrupt Status Registers
- PIEx – Peripheral Interrupt Enable Registers
- IPRx – Peripheral Interrupt Priority Registers
- IVTBASE<20:0> Address Registers
- IVTLOCK Register

Global interrupt control functions and external interrupts are controlled from the INTCON0 register. The INTCON1 register contains the status flags for the Interrupt controller.

The PIRx registers contain all of the interrupt request flags. Each source of interrupt has a status bit, which is set by the respective peripherals or an external signal and is cleared via software.

The PIEx registers contain all of the interrupt enable bits. These control bits are used to individually enable interrupts from the peripherals or external signals.

The IPRx registers are used to set the Interrupt Priority Level for each source of interrupt. Each user interrupt source can be assigned to either a high or low priority.

The IVTBASE register is user programmable and is used to determine the start address of the Interrupt Vector Table and the IVTLOCK register is used to prevent any unintended writes to the IVTBASE register.

There are two other configuration bits that control the way the interrupt controller can be configured.

- CONFIG2L<3>, MVECEN bit
- CONFIG2L<4>, IVT1WAY bit

The MVECEN bit in CONFIG2L determines whether the Vector table is used to determine the interrupt priorities.

- When the IVT1WAY determines the number of times the IVTLOCKED bit can be cleared and set after a device Reset. See **Section 9.2.3 “Interrupt Vector Table (IVT) address calculation”** for details.

## 9.6 Returning from Interrupt Service Routine (ISR)

The “Return from Interrupt” instruction (`RETFIE)` is used to mark the end of an ISR.

When `RETFIE 1` instruction is executed, the PC is loaded with the saved PC value from the top of the PC stack. Saved context is also restored with the execution of this instruction. Thus, execution returns to the previous state of operation that existed before the interrupt occurred.

When `RETFIE 0` instruction is executed, the saved context is not restored back to the registers.

## 9.7 Interrupt Latency

By assigning each interrupt with a vector address/ number (MVECEN = `1`), scanning of all interrupts is not necessary to determine the source of the interrupt.

When MVECEN = `1`, Vectored interrupt controller requires three clock cycles to vector to the ISR from main routine, thereby removing dependency of interrupt timing on compiled code.

There is a fixed latency of three instruction cycles between the completion of the instruction active when the interrupt occurred and the first instruction of the Interrupt Service Routine. Figure 9-7, Figure 9-8 and Figure 9-9 illustrate the sequence of events when a peripheral interrupt is asserted when the last executed instruction is one-cycle, two-cycle and three-cycle respectively, when MVECEN = `1`.

After the Interrupt Flag Status bit is set, the current instruction completes executing. In the first latency cycle, the contents of the PC, STATUS, WREG, BSR, FSR0/1/2, PRODL/H and PCLATH/U registers are context saved and the IVTBASE+ Vector number is calculated. In the second latency cycle, the PC is loaded with the calculated vector table address for the interrupt source and the starting address of the ISR is fetched. In the third latency cycle, the PC is loaded with the ISR address. All the latency cycles are executed as a `FNOP` instruction.

When MVECEN = `0`, Vectored interrupt controller requires two clock cycles to vector to the ISR from main routine. There is a latency of two instruction cycles plus the software latency between the completion of the instruction active when the interrupt occurred and the first instruction of the Interrupt Service Routine.

### REGISTER 9-13: PIR10: PERIPHERAL INTERRUPT REGISTER 10[1]

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | CLC4IF | CCP4IF |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | HS = Bit is set in hardware |

bit 7-2 **Unimplemented:** Read as ‘0’

bit 1 **CLC4IF:** CLC4 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 0 **CCP4IF:** CCP4 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

**Note 1:** Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

## 15.10 Reset

The DMA registers are set to the default state on any Reset. The registers are also reset to the default state when the enable bit is cleared (DMA1CON1bits.EN=0).

## 15.11 Power Saving Mode Operation

The DMA utilizes system clocks and it is treated as a peripheral when it comes to power saving operations. Like other peripherals, the DMA also uses Peripheral Module Disable bits to further tailor its operation in low-power states.

### 15.11.1 SLEEP MODE

When the device enters Sleep mode, the system clock to the module is shut down, therefore no DMA operation is supported in Sleep. Once the system clock is disabled, the requisite read and write clocks are also disabled, without which the DMA cannot perform any of its tasks.

Any transfers that may be in progress are resumed on exiting from Sleep mode. Register contents are not affected by the device entering or leaving Sleep mode. It is recommended that DMA transactions be allowed to finish before entering Sleep mode.

### 15.11.2 IDLE MODE

In IDLE mode, all of the system clocks (including the read and write clocks) are still operating but the CPU is not using them to save power.

Therefore, every instruction cycle is available to the system arbiter and if the bubble is granted to the DMA, it may be utilized to move data.

### 15.11.3 DOZE MODE

Similar to the Idle mode, the CPU does not utilize all of the available instruction cycles slots that are available to it in order to save power. It only executes instructions based on its settings from the Doze settings.

Therefore, every instruction not used by the CPU is available for system arbitration and may be utilized by the DMA if granted by the arbiter.

### 15.11.4 PERIPHERAL MODULE DISABLE

The Peripheral Module Disable (PMD) registers provide a method to disable DMA by gating all clock sources supplied to it. The respective DMAxMD bit needs to be set in order to disable the DMA.

## 15.12 DMA Register Interfaces

The DMA can transfer data to any GPR or SFR location. For better user accessibility, some of the more commonly used SFR spaces have their Mirror registers placed in Bank 64 (0x4000-0x40FF). These Mirror registers can be only accessed through the DMA Source and Destination Address registers. Refer to Table 4-3 for details about Bank 64 Registers.

## 17.8 Register Definitions: PPS Input Selection

### REGISTER 17-1: xxxPPS: PERIPHERAL xxx INPUT SELECTION

| U-0 | U-0 | R/W-m/u(1,3) | R/W-m/u(1) | R/W-m/u(1) | R/W-m/u(1) | R/W-m/u(1) | R/W-m/u(1) |
|---|---|---|---|---|---|---|---|
| — | — | xxxPPS<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | |
|---|---|---|
| R = Readable bit | W = Writable bit | -n/n = Value at POR and BOR/Value at all other Resets |
| u = Bit is unchanged | x = Bit is unknown | q = value depends on peripheral |
| ‘1’ = Bit is set | U = Unimplemented bit, read as ‘0’ | m = value depends on default location for that input |
| ‘0’ = Bit is cleared | | |

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-3 **xxxPPS<5:3>:** Peripheral xxx Input PORTx Pin Selection bits

See Table 17-1 for the list of available ports and default pin locations.
101 = PORTF(2)
100 = PORTE(3)
011 = PORTD(3)
010 = PORTC
001 = PORTB
000 = PORTA

bit 2-0 **xxxPPS<2:0>:** Peripheral xxx Input PORTx Pin Selection bits

111 = Peripheral input is from PORTx Pin 7 (Rx7)
110 = Peripheral input is from PORTx Pin 6 (Rx6)
101 = Peripheral input is from PORTx Pin 5 (Rx5)
100 = Peripheral input is from PORTx Pin 4 (Rx4)
011 = Peripheral input is from PORTx Pin 3 (Rx3)
010 = Peripheral input is from PORTx Pin 2 (Rx2)
001 = Peripheral input is from PORTx Pin 1 (Rx1)
000 = Peripheral input is from PORTx Pin 0 (Rx0)

**Note 1:** The Reset value ‘m’ of this register is determined by device default locations for that input.
**2:** Reserved on PIC18LF26/27/45/46/57K42 parts.
**3:** Reserved on PIC18LF26/27K42 parts.

## 24.1 PWMx Pin Configuration

All PWM outputs are multiplexed with the PORT data latch. The user must configure the pins as outputs by clearing the associated TRIS bits.

### 24.1.1 FUNDAMENTAL OPERATION

The PWM module produces a 10-bit resolution output. The PWM timer can be selected using the PxTSEL bits in the CCPTMRS1 register. The default selection for PWMx is T2TMR. Please note that the PWM module operation in the following sections is described with respect to T2TMR. Timer2 and T2PR set the period of the PWM. The PWMxDCL and PWMxDCH registers configure the duty cycle. The period is common to all PWM modules, whereas the duty cycle is independently controlled.

> **Note:** The Timer2 postscaler is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

All PWM outputs associated with Timer2 are set when T2TMR is cleared. Each PWMx is cleared when T2TMR is equal to the value specified in the corresponding PWMxDCH (8 MSb) and PWMxDCL<7:6> (2 LSb) registers. When the value is greater than or equal to T2PR, the PWM output is never cleared (100% duty cycle).

> **Note:** The PWMxDCH and PWMxDCL registers are double buffered. The buffers are updated when Timer2 matches T2PR. Care should be taken to update both registers before the timer match occurs.

### 24.1.2 PWM OUTPUT POLARITY

The output polarity is inverted by setting the PWMxPOL bit of the PWMxCON register.

### 24.1.3 PWM PERIOD

The PWM period is specified by the T2PR register of Timer2. The PWM period can be calculated using the formula of Equation 24-1. It is required to have FOSC/4 as clock input to Timer2/4/6 for correct PWM operation.

**EQUATION 24-1: PWM PERIOD**

$$PWM\ Period = [(T2PR) + 1] \bullet 4 \bullet TOSC \bullet (TMR2\ Prescale\ Value)$$

**Note:** TOSC = 1/FOSC

When T2TMR is equal to T2PR, the following three events occur on the next increment cycle:

- T2TMR is cleared
- The PWM output is active. (Exception: When the PWM duty cycle = 0%, the PWM output will remain inactive.)
- The PWMxDCH and PWMxDCL register values are latched into the buffers.

> **Note:** The Timer2 postscaler has no effect on the PWM operation.

### 24.1.4 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the PWMxDCH and PWMxDCL register pair. The PWMxDCH register contains the eight MSbs and the PWMxDCL<7:6>, the two LSbs. The PWMxDCH and PWMxDCL registers can be written to at any time.

Equation 24-2 is used to calculate the PWM pulse width.

Equation 24-3 is used to calculate the PWM duty cycle ratio.

**EQUATION 24-2: PULSE WIDTH**

$$Pulse\ Width = (PWMxDCH{:}PWMxDCL{<}7{:}6{>}) \bullet TOSC \bullet (TMR2\ Prescale\ Value)$$

**Note:** TOSC = 1/FOSC

**EQUATION 24-3: DUTY CYCLE RATIO**

$$Duty\ Cycle\ Ratio = \frac{(PWMxDCH{:}PWMxDCL{<}7{:}6{>})}{4(T2PR + 1)}$$

The 8-bit timer T2TMR register is concatenated with the two Least Significant bits of 1/FOSC, adjusted by the Timer2 prescaler to create the 10-bit time base. The system clock is used if the Timer2 prescaler is set to 1:1.

### 24.1.9 SETUP FOR PWM OPERATION USING PWMx PINS

The following steps should be taken when configuring the module for PWM operation using the PWMx pins:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note 1 below.
   - Select the timer clock source to be as FOSC/4 using the TxCLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the respective PIR register is set. See note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the desired pin PPS control bits.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

**Note 1:** In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.

**2:** For operation with other peripherals only, disable PWMx pin outputs.

### 24.1.10 SETUP FOR PWM OPERATION TO OTHER DEVICE PERIPHERALS

The following steps should be taken when configuring the module for PWM operation to be used by other device peripherals:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note 1 below.
   - Select the timer clock source to be as FOSC/4 using the TxCLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin:
   - Wait until Timer2 overflows, TMR2IF bit of the respective PIR register is set. See Note 1 below.
7. Configure the PWM module by loading the PWMxCON register with the appropriate values.

**Note 1:** In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored.

**REGISTER 24-2: CCPTMRS1: CCP TIMERS CONTROL REGISTER 1**

| R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| P8TSEL<1:0> | | P7TSEL<1:0> | | P6TSEL<1:0> | | P5TSEL<1:0> | |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ | |
| -n = Value at POR | ‘1’ = Bit is set | ‘0’ = Bit is cleared | x = Bit is unknown |

bit 7-6 **P8TSEL<1:0>:** PWM8 Timer Selection bits
11 = PWM8 based on TMR6
10 = PWM8 based on TMR4
01 = PWM8 based on TMR2
00 = Reserved

bit 5-4 **P7TSEL<1:0>:** PWM7 Timer Selection bits
11 = PWM7 based on TMR6
10 = PWM7 based on TMR4
01 = PWM7 based on TMR2
00 = Reserved

bit 3-2 **P6TSEL<1:0>:** PWM6 Timer Selection bits
11 = PWM6 based on TMR6
10 = PWM6 based on TMR4
01 = PWM6 based on TMR2
00 = Reserved

bit 1-0 **P5TSEL<1:0>:** PWM5 Timer Selection bits
11 = PWM5 based on TMR6
10 = PWM5 based on TMR4
01 = PWM5 based on TMR2
00 = Reserved

### 25.6.3 PERIOD AND DUTY CYCLE MODE

In Duty Cycle mode, either the duty cycle or period (depending on polarity) of the SMT1_signal can be acquired relative to the SMT clock. The CPW register is updated on a falling edge of the signal, and the CPR register is updated on a rising edge of the signal, along with the SMT1TMR resetting to 0x0001. In addition, the GO bit is reset on a rising edge when the SMT is in Single Acquisition mode. See Figure 25-6 and Figure 25-7.

### REGISTER 27-3: CLCxSEL0: GENERIC CLCx DATA 0 SELECT REGISTER

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | D1S<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

| | | |
|---|---|---|
| **Legend:** | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **D1S<5:0>:** CLCx Data1 Input Selection bits
See Table 27-1.

### REGISTER 27-4: CLCxSEL1: GENERIC CLCx DATA 1 SELECT REGISTER

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | D2S<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

| | | |
|---|---|---|
| **Legend:** | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **D2S<5:0>:** CLCx Data 2 Input Selection bits
See Table 27-1.

### REGISTER 27-5: CLCxSEL2: GENERIC CLCx DATA 2 SELECT REGISTER

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | D3S<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

| | | |
|---|---|---|
| **Legend:** | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **D3S<5:0>:** CLCx Data 3 Input Selection bits
See Table 27-1.

### REGISTER 27-6: CLCxSEL3: GENERIC CLCx DATA 3 SELECT REGISTER

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | D4S<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

| | | |
|---|---|---|
| **Legend:** | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **D4S<5:0>:** CLCx Data 4 Input Selection bits
See Table 27-1.

### REGISTER 28-3: NCO1ACCL: NCO1 ACCUMULATOR REGISTER – LOW BYTE

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| ACC<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-0 **ACC<7:0>:** NCO1 Accumulator, Low Byte

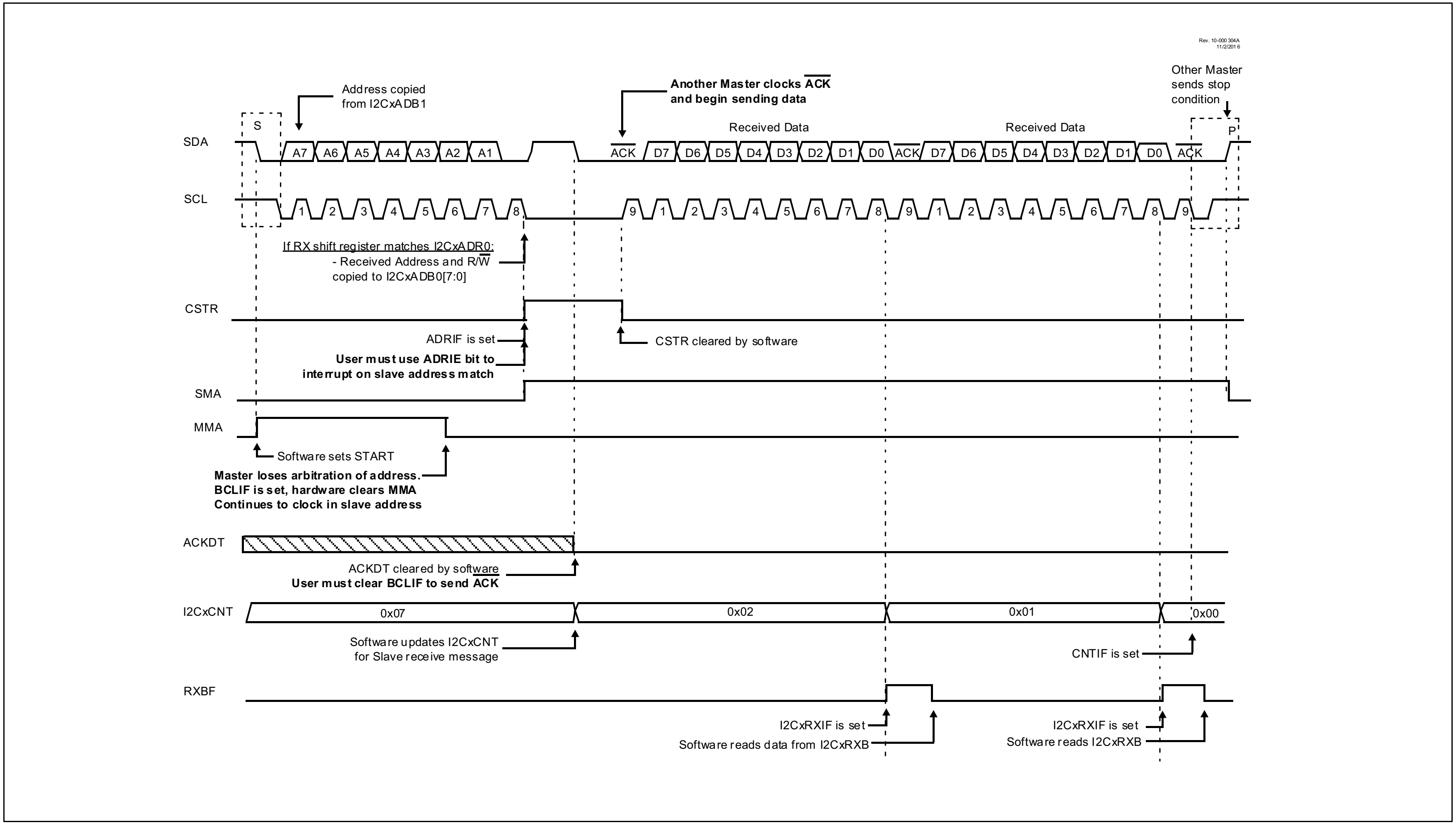
### REGISTER 28-4: NCO1ACCH: NCO1 ACCUMULATOR REGISTER – HIGH BYTE

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| ACC<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | |

bit 7-0 **ACC<15:8>:** NCO1 Accumulator, High Byte

### FIGURE 33-24: I$^2$C MULTI-MASTER, 7-BIT ADDRESS, WRITE (ADRIE = 1, WRIE = 0)



### REGISTER 33-5: I2CxBTO: I$^2$C BUS TIMEOUT SELECTION REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | BTO<2:0> | | |
| bit 7 | | | | | | | bit 0 |

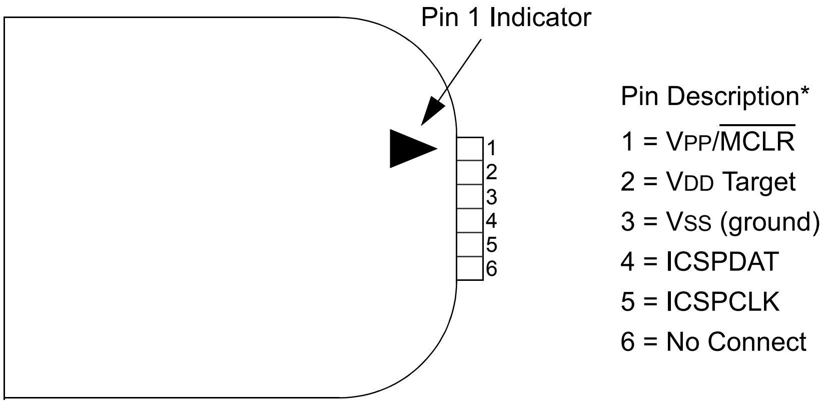
| Legend: | | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| ‘1’ = Bit is set | ‘0’ = Bit is cleared | HS = Hardware set HC = Hardware clear |

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **BTO<2:0>:** I$^2$C Bus Timeout Selection bits

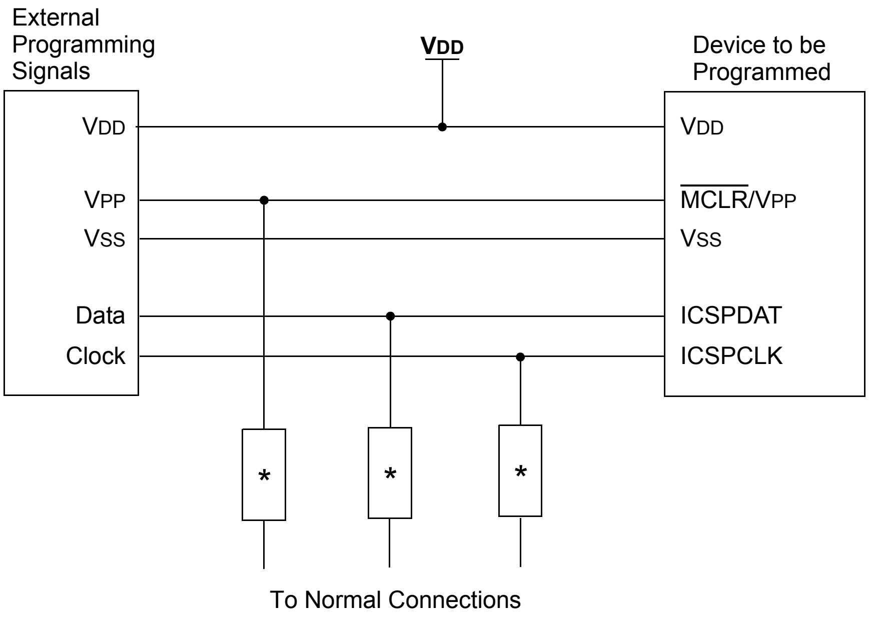
| BTO<2:0> | I$^2$Cx Bus Timeout Selection |
|---|---|
| 111 | CLC4OUT |
| 110 | CLC3OUT |
| 101 | CLC2OUT |
| 100 | CLC1OUT |
| 011 | TMR6 post scaled output |
| 010 | TMR4 post scaled output |
| 001 | TMR2 post scaled output |
| 000 | Reserved |

**FIGURE 40-2: PICkit™ PROGRAMMER STYLE CONNECTOR INTERFACE**



* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

**FIGURE 40-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING**



* Isolation devices (as required).

## CALLW Subroutine Call Using WREG

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0001 | 0100 |

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a `NOP` instruction while the new next instruction is fetched.
Unlike `CALL`, there is no option to update W, Status or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read WREG | PUSH PC to stack | No operation |
| No operation | No operation | No operation | No operation |

Example: `HERE CALLW`

Before Instruction

| | | |
|---|---|---|
| PC | = | address `(HERE)` |
| PCLATH | = | 10h |
| PCLATU | = | 00h |
| W | = | 06h |

After Instruction

| | | |
|---|---|---|
| PC | = | 001006h |
| TOS | = | address `(HERE + 2)` |
| PCLATH | = | 10h |
| PCLATU | = | 00h |
| W | = | 06h |

## CLRF Clear f

Syntax: CLRF f {,a}

Operands: $0 \leq f \leq 255$
$a \in [0,1]$

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 101a | ffff | ffff |

Description: Clears the contents of the specified register.
If ‘a’ is ‘`0`’, the Access Bank is selected.
If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank.
If ‘a’ is ‘`0`’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 41.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register ‘f’ | Process Data | Write register ‘f’ |

Example: `CLRF FLAG_REG, 1`

Before Instruction

| | | |
|---|---|---|
| FLAG_REG | = | 5Ah |

After Instruction

| | | |
|---|---|---|
| FLAG_REG | = | 00h |

### TABLE 44-27: TEMPERATURE INDICATOR REQUIREMENTS

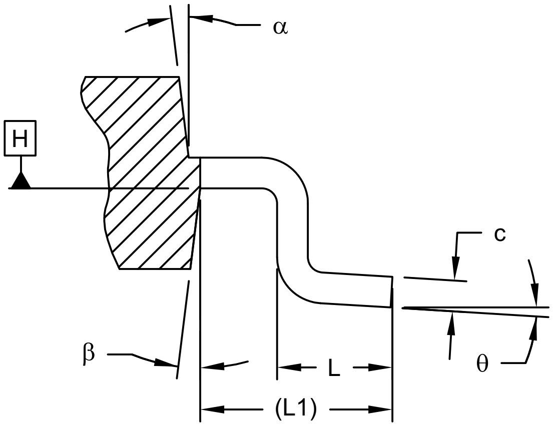
| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Param No. | Symbol | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
| TS01* | TACQMIN | Minimum ADC Acquisition Time Delay | | — | 25 | — | µs | |
| TS02* | MV | Voltage Sensitivity | High Range | — | -3.684 | — | mV/°C | TSRNG = 1 |
| | | | Low Range | — | -2.456 | — | mV/°C | TSRNG = 0 |

* These parameters are characterized but not tested.

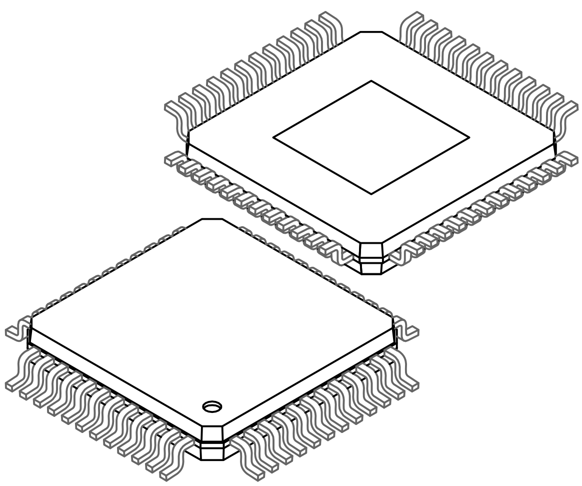
† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

## 48-Lead Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP] With Exposed Pad

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



SECTION A-A



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Leads | N | | 48 | |
| Lead Pitch | e | | 0.50 BSC | |
| Overall Height | A | - | - | 1.20 |
| Standoff | A1 | 0.05 | - | 0.15 |
| Molded Package Thickness | A2 | 0.95 | 1.00 | 1.05 |
| Foot Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Foot Angle | φ | 0° | 3.5° | 7° |
| Overall Width | E | | 9.00 BSC | |
| Overall Length | D | | 9.00 BSC | |
| Molded Package Width | E1 | | 7.00 BSC | |
| Molded Package Length | D1 | | 7.00 BSC | |
| Exposed Pad Width | E2 | | 3.50 BSC | |
| Exposed Pad Length | D2 | | 3.50 BSC | |
| Lead Thickness | c | 0.09 | - | 0.16 |
| Lead Width | b | 0.17 | 0.22 | 0.27 |
| Mold Draft Angle Top | α | 11° | 12° | 13° |
| Mold Draft Angle Bottom | β | 11° | 12° | 13° |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-183A Sheet 2 of 2

## THE MICROCHIP WEBSITE

Microchip provides online support via our website at www.microchip.com. This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip website at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the website at: http://www.microchip.com/support**