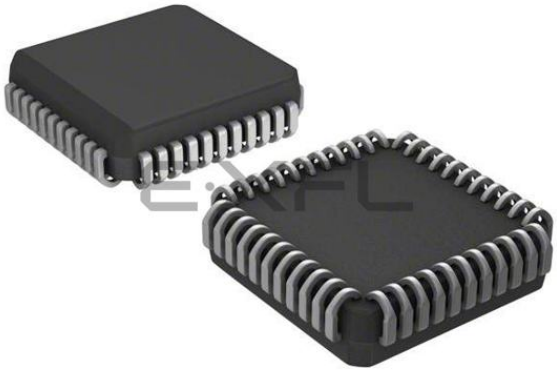


**Welcome to E-XFL.COM**

## 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 | 8MHz |
| Connectivity | UART/USART |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 33 |
| Program Memory Size | 16KB (8K x 16) |
| Program Memory Type | OTP |
| EEPROM Size | - |
| RAM Size | 454 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.5V ~ 6V |
| Data Converters | - |
| Oscillator Type | External |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-LCC (J-Lead) |
| Supplier Device Package | 44-PLCC (16.59x16.59) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic17lc44-08i-l |

### 5.5 RA0/INT Interrupt

The external interrupt on the RA0/INT pin is edge triggered. Either the rising edge, if INTEDG bit (T0STA<7>) is set, or the falling edge, if INTEDG bit is clear. When a valid edge appears on the RA0/INT pin, the INTF bit (INTSTA<4>) is set. This interrupt can be disabled by clearing the INTE control bit (INTSTA<0>). The INT interrupt can wake the processor from SLEEP. See Section 14.4 for details on SLEEP operation.

### 5.6 TMR0 Interrupt

An overflow (FFFFh $\rightarrow$ 0000h) in TMR0 will set the T0IF (INTSTA<5>) bit. The interrupt can be enabled/ disabled by setting/clearing the T0IE control bit (INTSTA<1>). For operation of the Timer0 module, see Section 11.0.

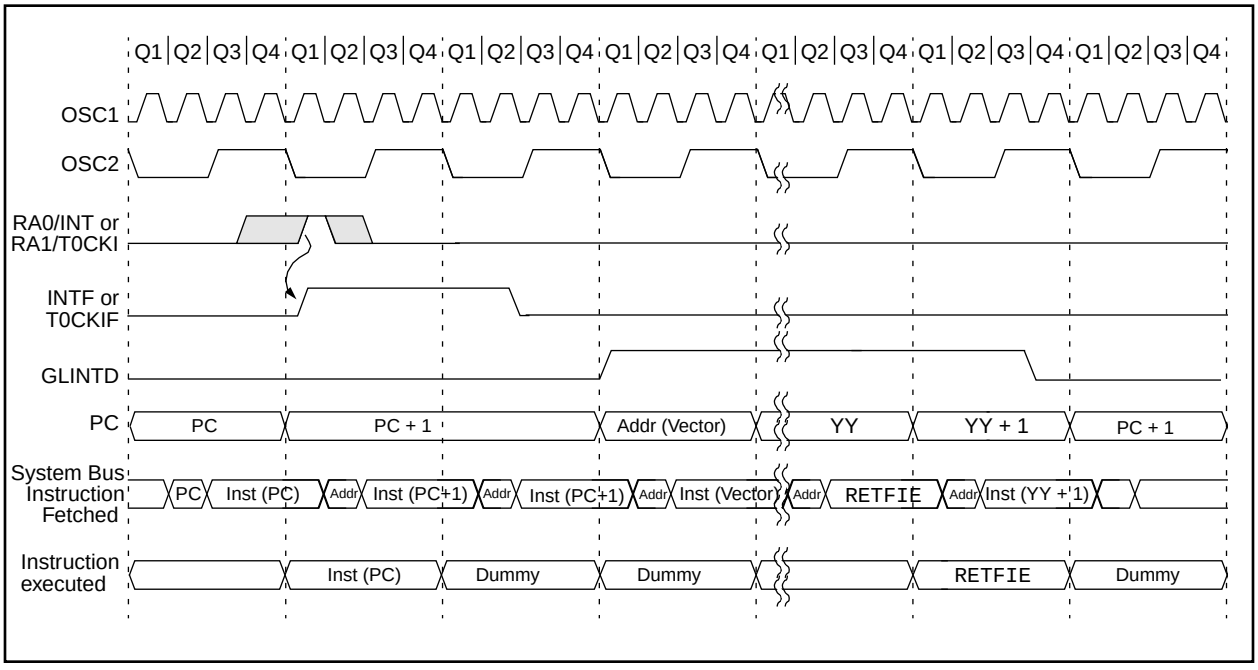
### 5.7 T0CKI Interrupt

The external interrupt on the RA1/T0CKI pin is edge triggered. Either the rising edge, if the T0SE bit (T0STA<6>) is set, or the falling edge, if the T0SE bit is clear. When a valid edge appears on the RA1/T0CKI pin, the T0CKIF bit (INTSTA<6>) is set. This interrupt can be disabled by clearing the T0CKIE control bit (INTSTA<2>). The T0CKI interrupt can wake up the processor from SLEEP. See Section 14.4 for details on SLEEP operation.

### 5.8 Peripheral Interrupt

The peripheral interrupt flag indicates that at least one of the peripheral interrupts occurred (PEIF is set). The PEIF bit is a read only bit, and is a bit wise OR of all the flag bits in the PIR register AND'ed with the corresponding enable bits in the PIE register. Some of the peripheral interrupts can wake the processor from SLEEP. See Section 14.4 for details on SLEEP operation.

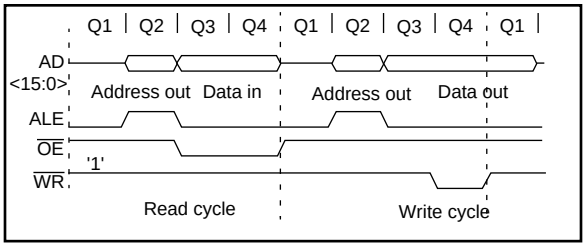
**FIGURE 5-5: INT PIN / T0CKI PIN INTERRUPT TIMING**



### 6.1.2 EXTERNAL MEMORY INTERFACE

When either microprocessor or extended microcontroller mode is selected, PORTC, PORTD and PORTE are configured as the system bus. PORTC and PORTD are the multiplexed address/data bus and PORTE is for the control signals. External components are needed to demultiplex the address and data. This can be done as shown in Figure 6-4. The waveforms of address and data are shown in Figure 6-3. For complete timings, please refer to the electrical specification section.

**FIGURE 6-3: EXTERNAL PROGRAM MEMORY ACCESS WAVEFORMS**




The system bus requires that there is no bus conflict (minimal leakage), so the output value (address) will be capacitively held at the desired value.

As the speed of the processor increases, external EPROM memory with faster access time must be used. Table 6-2 lists external memory speed requirements for a given PIC17C4X device frequency.

In extended microcontroller mode, when the device is executing out of internal memory, the control signals will continue to be active. That is, they indicate the action that is occurring in the internal memory. The external memory access is ignored.

This following selection is for use with Microchip EPROMs. For interfacing to other manufacturers memory, please refer to the electrical specifications of the desired PIC17C4X device, as well as the desired memory device to ensure compatibility.

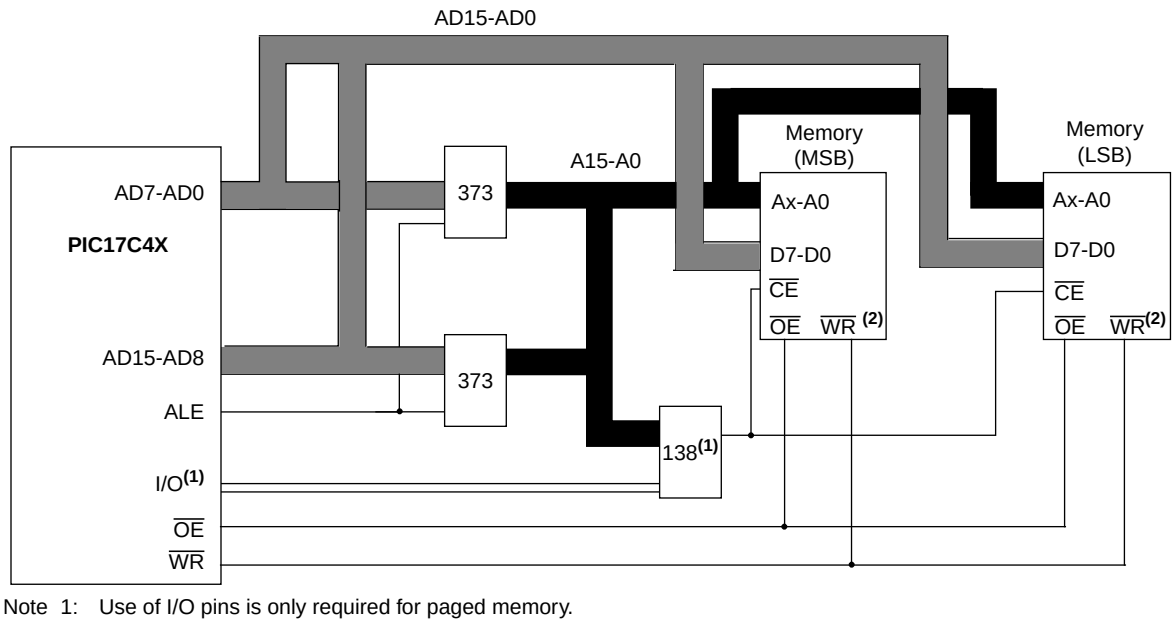
**TABLE 6-2: EPROM MEMORY ACCESS TIME ORDERING SUFFIX**

| PIC17C4X Oscillator Frequency | Instruction Cycle Time (TCY) | EPROM Suffix | |
|---|---|---|---|
| | | PIC17C42 | PIC17C43 PIC17C44 |
| 8 MHz | 500 ns | -25 | -25 |
| 16 MHz | 250 ns | -12 | -15 |
| 20 MHz | 200 ns | -90 | -10 |
| 25 MHz | 160 ns | N.A. | -70 |
| 33 MHz | 121 ns | N.A. | (1) |

Note 1: The access times for this requires the use of fast SRAMS.

**Note:** The external memory interface is not supported for the LC devices.

**FIGURE 6-4: TYPICAL EXTERNAL PROGRAM MEMORY CONNECTION DIAGRAM**




Note 1: Use of I/O pins is only required for paged memory.
2: This signal is unused for ROM and EPROM devices.

### 6.4.1 INDIRECT ADDRESSING REGISTERS

The PIC17C4X has four registers for indirect addressing. These registers are:

- INDF0 and FSR0
- INDF1 and FSR1

Registers INDF0 and INDF1 are not physically implemented. Reading or writing to these registers activates indirect addressing, with the value in the corresponding FSR register being the address of the data. The FSR is an 8-bit register and allows addressing anywhere in the 256-byte data memory address range. For banked memory, the bank of memory accessed is specified by the value in the BSR.

If file INDF0 (or INDF1) itself is read indirectly via an FSR, all '0's are read (Zero bit is set). Similarly, if INDF0 (or INDF1) is written to indirectly, the operation will be equivalent to a NOP, and the status bits are not affected.

### 6.4.2 INDIRECT ADDRESSING OPERATION

The indirect addressing capability has been enhanced over that of the PIC16CXX family. There are two control bits associated with each FSR register. These two bits configure the FSR register to:

- Auto-decrement the value (address) in the FSR after an indirect access
- Auto-increment the value (address) in the FSR after an indirect access
- No change to the value (address) in the FSR after an indirect access

These control bits are located in the ALUSTA register. The FSR1 register is controlled by the FS3:FS2 bits and FSR0 is controlled by the FS1:FS0 bits.

When using the auto-increment or auto-decrement features, the effect on the FSR is not reflected in the ALUSTA register. For example, if the indirect address causes the FSR to equal '0', the Z bit will not be set.

If the FSR register contains a value of 0h, an indirect read will read 0h (Zero bit is set) while an indirect write will be equivalent to a NOP (status bits are not affected).

Indirect addressing allows single cycle data transfers within the entire data space. This is possible with the use of the MOVPF and MOVFP instructions, where either 'p' or 'f' is specified as INDF0 (or INDF1).

If the source or destination of the indirect address is in banked memory, the location accessed will be determined by the value in the BSR.

A simple program to clear RAM from 20h - FFh is shown in Example 6-1.

**EXAMPLE 6-1: INDIRECT ADDRESSING**

```
    MOVLW    0x20          ;
    MOVWF    FSR0          ; FSR0 = 20h
    BCF      ALUSTA, FS1   ; Increment FSR
    BSF      ALUSTA, FS0   ; after access
    BCF      ALUSTA, C     ; C = 0
    MOVLW    END_RAM + 1   ;
LP  CLRF     INDF0         ; Addr(FSR) = 0
    CPFSEQ   FSR0          ; FSR0 = END_RAM+1?
    GOTO     LP            ; NO, clear next
    :                      ; YES, All RAM is
    :                      ; cleared
```

## 6.5 Table Pointer (TBLPTRL and TBLPTRH)

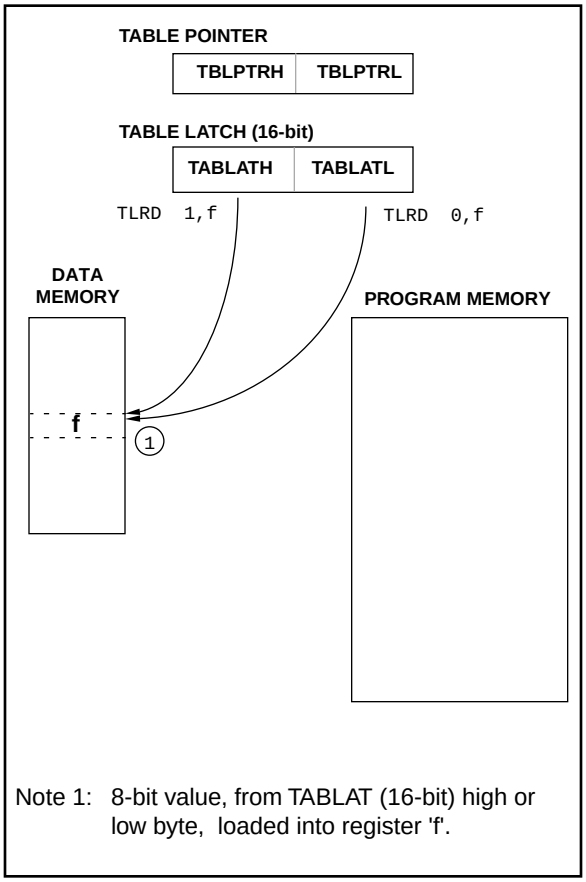
File registers TBLPTRL and TBLPTRH form a 16-bit pointer to address the 64K program memory space. The table pointer is used by instructions TABLWT and TABLRD.

The TABLRD and the TABLWT instructions allow transfer of data between program and data space. The table pointer serves as the 16-bit address of the data word within the program memory. For a more complete description of these registers and the operation of Table Reads and Table Writes, see Section 7.0.

## 6.6 Table Latch (TBLATH, TBLATL)

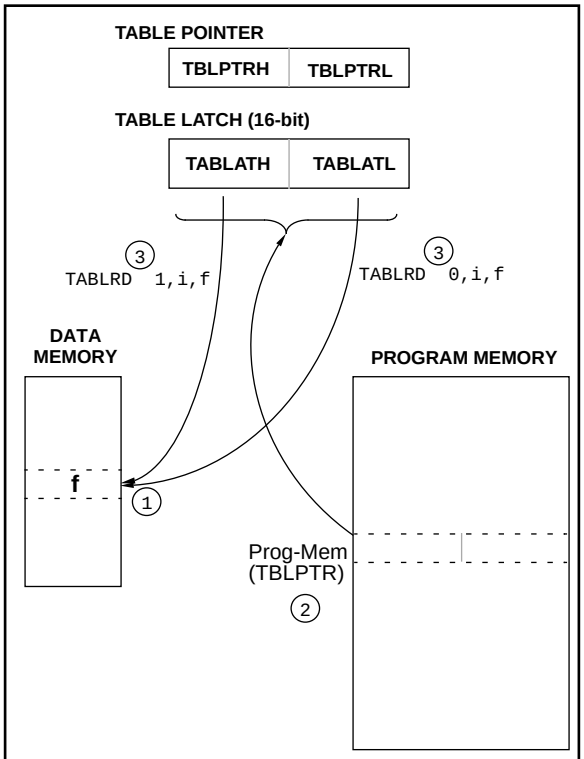
The table latch (TBLAT) is a 16-bit register, with TBLATH and TBLATL referring to the high and low bytes of the register. It is not mapped into data or program memory. The table latch is used as a temporary holding latch during data transfer between program and data memory (see descriptions of instructions TABLRD, TABLWT, TLRD and TLWT). For a more complete description of these registers and the operation of Table Reads and Table Writes, see Section 7.0.

**FIGURE 7-3: TLRD INSTRUCTION OPERATION**




Note 1: 8-bit value, from TABLAT (16-bit) high or low byte, loaded into register 'f'.

**FIGURE 7-4: TABLRD INSTRUCTION OPERATION**




Note 1: 8-bit value, from TABLAT (16-bit) high or low byte, loaded into register 'f'.
2: 16-bit value at Program Memory (TBLPTR) loaded into TABLAT register.
3: If "i" = 1, then TBLPTR = TBLPTR + 1, If "i" = 0, then TBLPTR is unchanged.

## 7.2 Table Writes to External Memory

Table writes to external memory are always two-cycle instructions. The second cycle writes the data to the external memory location. The sequence of events for an external memory write are the same for an internal write.

**Note:** If an interrupt is pending or occurs during the TABLWT, the two cycle table write completes. The RA0/INT, TMR0, or T0CKI interrupt flag is automatically cleared or the pending peripheral interrupt is acknowledged.

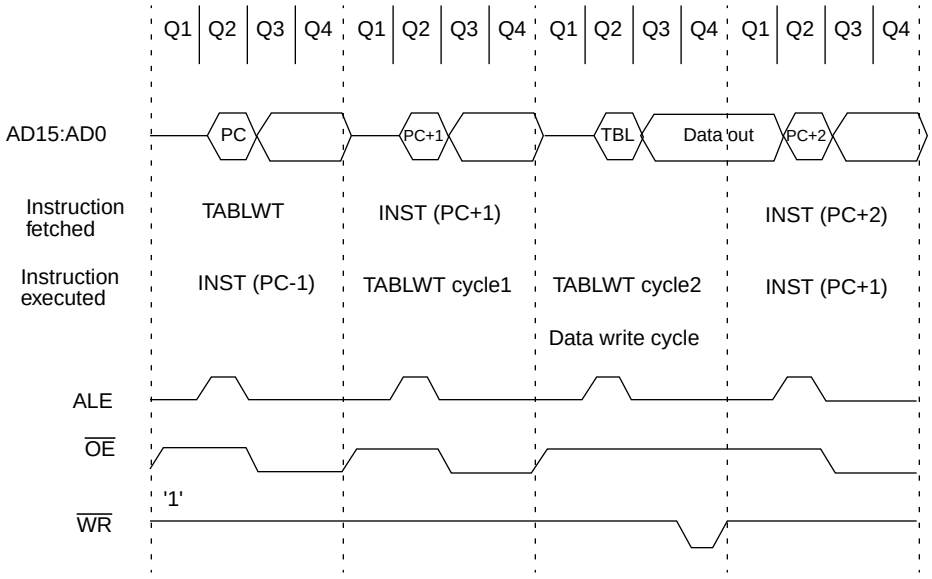
### 7.2.2 TABLE WRITE CODE

The "i" operand of the TABLWT instruction can specify that the value in the 16-bit TBLPTR register is automatically incremented for the next write. In Example 7-1, the TBLPTR register is not automatically incremented.

**EXAMPLE 7-1: TABLE WRITE**

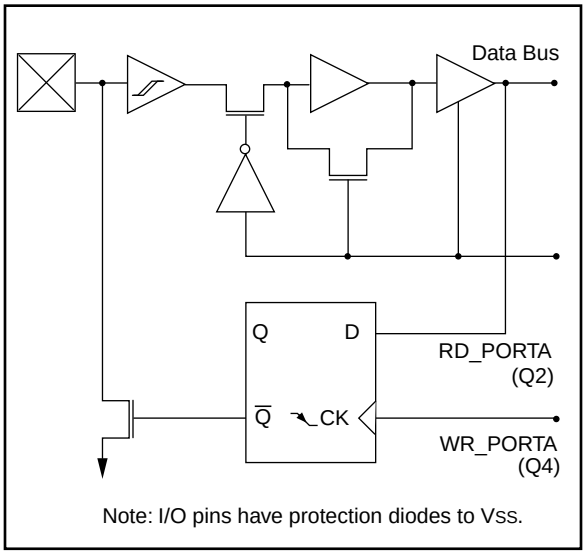
```
CLRWDT                  ; Clear WDT
MOVLW   HIGH (TBL_ADDR) ; Load the Table
MOVWF   TBLPTRH         ;   address
MOVLW   LOW (TBL_ADDR)  ;
MOVWF   TBLPTRL         ;
MOVLW   HIGH (DATA)     ; Load HI byte
TLWT    1, WREG         ;   in TABLATCH
MOVLW   LOW (DATA)      ; Load LO byte
TABLWT  0,0,WREG        ;   in TABLATCH
                        ;   and write to
                        ;   program memory
                        ;   (Ext. SRAM)
```

**FIGURE 7-5: TABLWT WRITE TIMING (EXTERNAL MEMORY)**



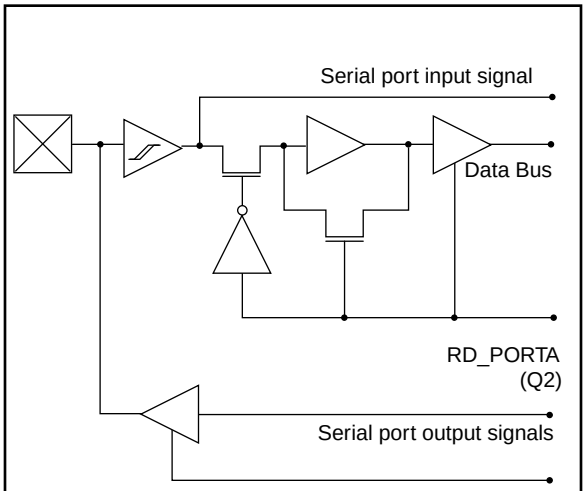

Note: If external write GLINTD = '1', Enable bit = '1', '1' → Flag bit, Do table write. The highest pending interrupt is cleared.

**FIGURE 9-2: RA2 AND RA3 BLOCK DIAGRAM**




Note: I/O pins have protection diodes to VSS.

**FIGURE 9-3: RA4 AND RA5 BLOCK DIAGRAM**




$\overline{OE}$ = SPEN,SYNC,TXEN, $\overline{CREN}$, $\overline{SREN}$ for RA4

$\overline{OE}$ = SPEN ($\overline{SYNC}$+SYNC,$\overline{CSRC}$) for RA5

Note: I/O pins have protection diodes to VDD and VSS.

**TABLE 9-1: PORTA FUNCTIONS**

| Name | Bit0 | Buffer Type | Function |
|---|---|---|---|
| RA0/INT | bit0 | ST | Input or external interrupt input. |
| RA1/T0CKI | bit1 | ST | Input or clock input to the TMR0 timer/counter, and/or an external interrupt input. |
| RA2 | bit2 | ST | Input/Output. Output is open drain type. |
| RA3 | bit3 | ST | Input/Output. Output is open drain type. |
| RA4/RX/DT | bit4 | ST | Input or USART Asynchronous Receive or USART Synchronous Data. |
| RA5/TX/CK | bit5 | ST | Input or USART Asynchronous Transmit or USART Synchronous Clock. |
| $\overline{RBPU}$ | bit7 | — | Control bit for PORTB weak pull-ups. |

Legend: ST = Schmitt Trigger input.

**TABLE 9-2: REGISTERS/BITS ASSOCIATED WITH PORTA**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other resets (Note1) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10h, Bank 0 | PORTA | $\overline{RBPU}$ | — | RA5 | RA4 | RA3 | RA2 | RA1/T0CKI | RA0/INT | 0-xx xxxx | 0-uu uuuu |
| 05h, Unbanked | T0STA | INTEDG | T0SE | T0CS | PS3 | PS2 | PS1 | PS0 | — | 0000 000- | 0000 000- |
| 13h, Bank 0 | RCSTA | SPEN | RC9 | SREN | CREN | — | FERR | OERR | RC9D | 0000 -00x | 0000 -00u |
| 15h, Bank 0 | TXSTA | CSRC | TX9 | TXEN | SYNC | — | — | TRMT | TX9D | 0000 --1x | 0000 --1u |

Legend: x = unknown, u = unchanged, - = unimplemented reads as '0'. Shaded cells are not used by PORTA.

Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and the Watchdog Timer Reset.

## 9.4 PORTD and DDRD Registers

PORTD is an 8-bit bi-directional port. The corresponding data direction register is DDRD. A '1' in DDRD configures the corresponding port pin as an input. A '0' in the DDRC register configures the corresponding port pin as an output. Reading PORTD reads the status of the pins, whereas writing to it will write to the port latch. PORTD is multiplexed with the system bus. When operating as the system bus, PORTD is the high order byte of the address/data bus (AD15:AD8). The timing for the system bus is shown in the Electrical Characteristics section.

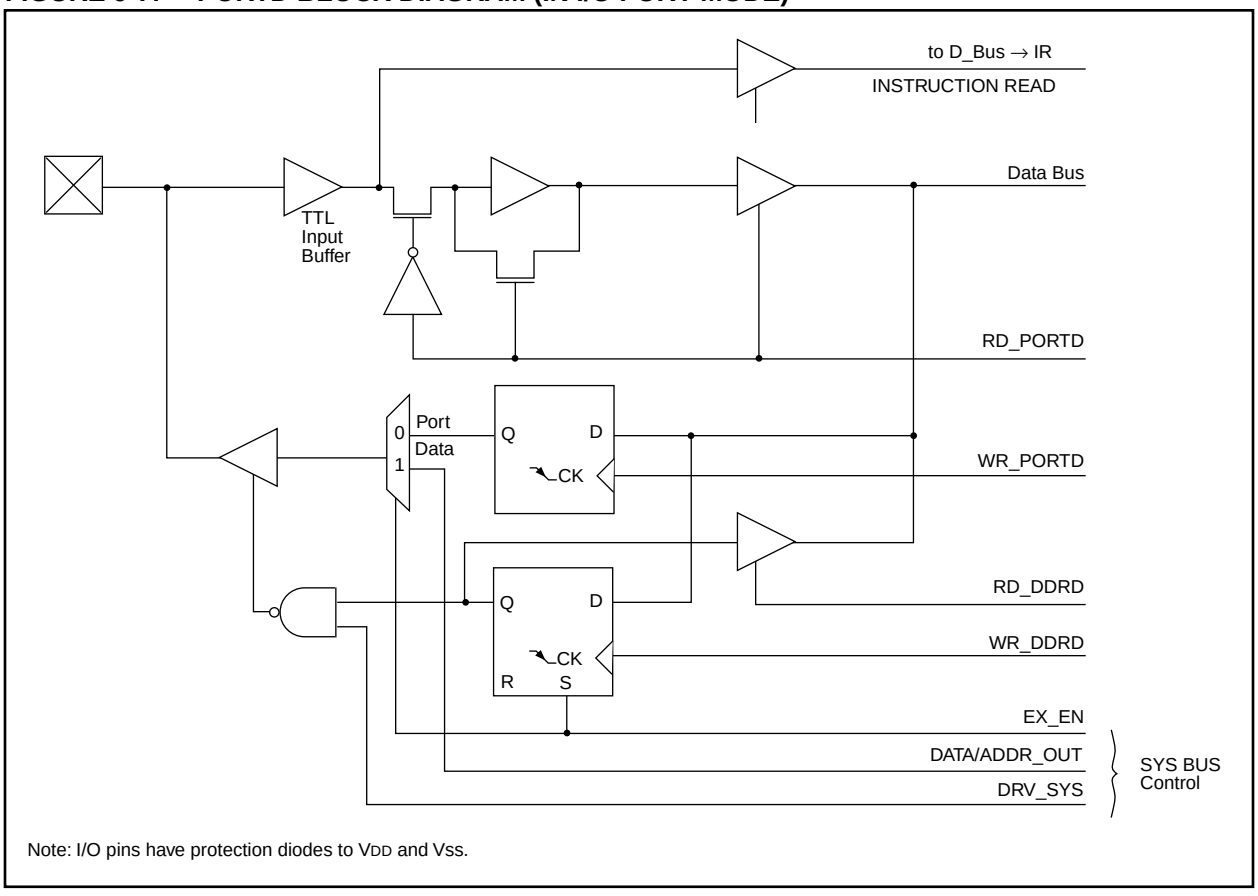
| **Note:** | This port is configured as the system bus when the device's configuration bits are selected to Microprocessor or Extended Microcontroller modes. In the two other microcontroller modes, this port is a general purpose I/O. |
|---|---|

Example 9-3 shows the instruction sequence to initialize PORTD. The Bank Select Register (BSR) must be selected to Bank 1 for the port to be initialized.

**EXAMPLE 9-3: INITIALIZING PORTD**

```
MOVLB  1          ;  Select Bank 1
CLRF   PORTD      ;  Initialize PORTD data
                  ;   latches before setting
                  ;   the data direction
                  ;   register
MOVLW  0xCF       ;  Value used to initialize
                  ;   data direction
MOVWF  DDRD       ;  Set RD<3:0> as inputs
                  ;   RD<5:4> as outputs
                  ;   RD<7:6> as inputs
```

**FIGURE 9-7: PORTD BLOCK DIAGRAM (IN I/O PORT MODE)**




Note: I/O pins have protection diodes to VDD and Vss.

**TABLE 9-7: PORTD FUNCTIONS**

| Name | Bit | Buffer Type | Function |
|---|---|---|---|
| RD0/AD8 | bit0 | TTL | Input/Output or system bus address/data pin. |
| RD1/AD9 | bit1 | TTL | Input/Output or system bus address/data pin. |
| RD2/AD10 | bit2 | TTL | Input/Output or system bus address/data pin. |
| RD3/AD11 | bit3 | TTL | Input/Output or system bus address/data pin. |
| RD4/AD12 | bit4 | TTL | Input/Output or system bus address/data pin. |
| RD5/AD13 | bit5 | TTL | Input/Output or system bus address/data pin. |
| RD6/AD14 | bit6 | TTL | Input/Output or system bus address/data pin. |
| RD7/AD15 | bit7 | TTL | Input/Output or system bus address/data pin. |

Legend: TTL = TTL input.

**TABLE 9-8: REGISTERS/BITS ASSOCIATED WITH PORTD**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other resets (Note1) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 13h, Bank 1 | PORTD | RD7/ AD15 | RD6/ AD14 | RD5/ AD13 | RD4/ AD12 | RD3/ AD11 | RD2/ AD10 | RD1/ AD9 | RD0/ AD8 | xxxx xxxx | uuuu uuuu |
| 12h, Bank 1 | DDRD | Data direction register for PORTD | | | | | | | | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged.

Note 1: Other (non power-up) resets include: external reset through $\overline{\text{MCLR}}$ and the Watchdog Timer Reset.

## 11.3 Read/Write Consideration for TMR0

Although TMR0 is a 16-bit timer/counter, only 8-bits at a time can be read or written during a single instruction cycle. Care must be taken during any read or write.

### 11.3.1 READING 16-BIT VALUE

The problem in reading the entire 16-bit value is that after reading the low (or high) byte, its value may change from FFh to 00h.

Example 11-1 shows a 16-bit read. To ensure a proper read, interrupts must be disabled during this routine.

**EXAMPLE 11-1: 16-BIT READ**

```
MOVPF   TMR0L, TMPLO    ;read low tmr0
MOVPF   TMR0H, TMPHI    ;read high tmr0
MOVFP   TMPLO, WREG     ;tmplo –> wreg
CPFSLT  TMR0L           ;tmr0l < wreg?
RETURN                  ;no then return
MOVPF   TMR0L, TMPLO    ;read low tmr0
MOVPF   TMR0H, TMPHI    ;read high tmr0
RETURN                  ;return
```

### 11.3.2 WRITING A 16-BIT VALUE TO TMR0

Since writing to either TMR0L or TMR0H will effectively inhibit increment of that half of the TMR0 in the next cycle (following write), but not inhibit increment of the other half, the user must write to TMR0L first and TMR0H next in two consecutive instructions, as shown in Example 11-2. The interrupt must be disabled. Any write to either TMR0L or TMR0H clears the prescaler.

**EXAMPLE 11-2: 16-BIT WRITE**

```
BSF   CPUSTA, GLINTD ; Disable interrupt
MOVFP RAM_L, TMR0L   ;
MOVFP RAM_H, TMR0H   ;
BCF   CPUSTA, GLINTD ; Done, enable interrupt
```

## 11.4 Prescaler Assignments

Timer0 has an 8-bit prescaler. The prescaler assignment is fully under software control; i.e., it can be changed “on the fly” during program execution. When changing the prescaler assignment, clearing the prescaler is recommended before changing assignment. The value of the prescaler is “unknown,” and assigning a value that is less then the present value makes it difficult to take this unknown time into account.

**FIGURE 11-4: TMR0 TIMING: WRITE HIGH OR LOW BYTE**

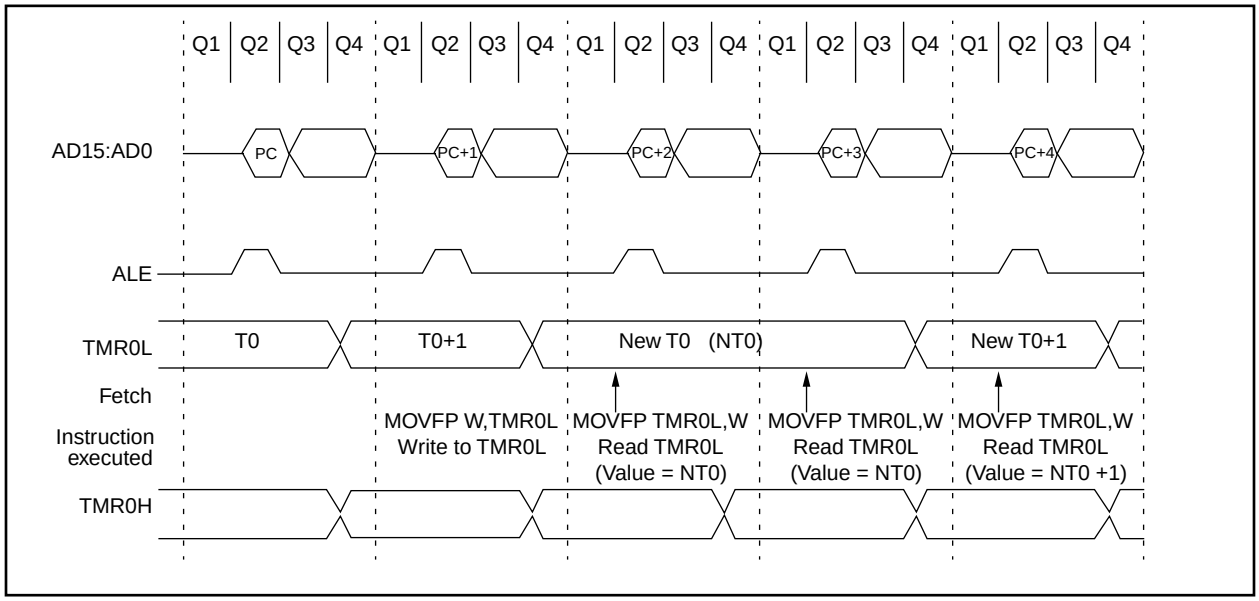
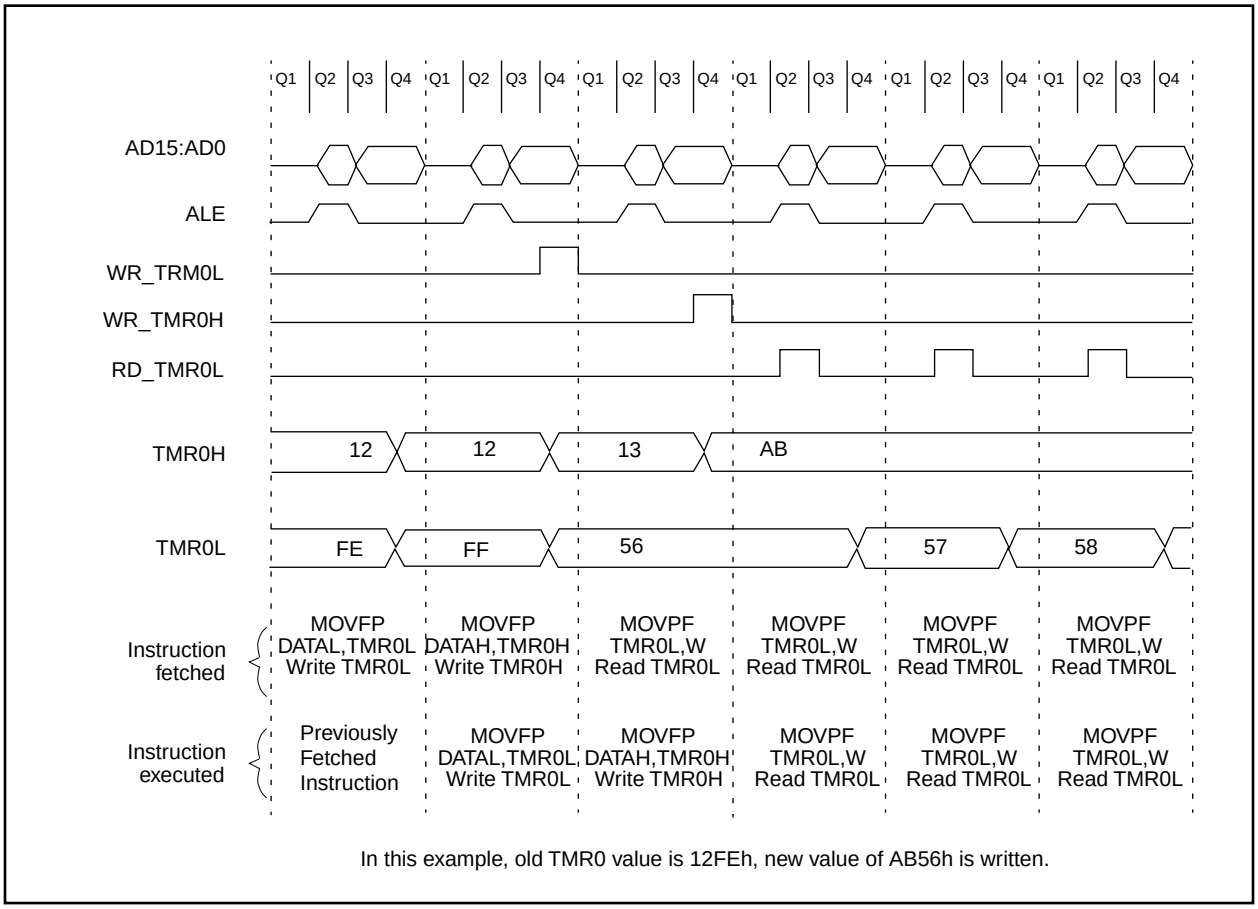


**FIGURE 11-5: TMR0 READ/WRITE IN TIMER MODE**




In this example, old TMR0 value is 12FEh, new value of AB56h is written.

**TABLE 11-1: REGISTERS/BITS ASSOCIATED WITH TIMER0**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other resets (Note1) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 05h, Unbanked | T0STA | INTEDG | T0SE | T0CS | PS3 | PS2 | PS1 | PS0 | — | 0000 000- | 0000 000- |
| 06h, Unbanked | CPUSTA | — | — | STKAV | GLINTD | $\overline{TO}$ | $\overline{PD}$ | — | — | --11 11-- | --11 qq-- |
| 07h, Unbanked | INTSTA | PEIF | T0CKIF | T0IF | INTF | PEIE | T0CKIE | T0IE | INTE | 0000 0000 | 0000 0000 |
| 0Bh, Unbanked | TMR0L | TMR0 register; low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Ch, Unbanked | TMR0H | TMR0 register; high byte | | | | | | | | xxxx xxxx | uuuu uuuu |

Legend: x = unknown, u = unchanged, - = unimplemented read as a '0', q - value depends on condition, Shaded cells are not used by Timer0.
Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and the Watchdog Timer Reset.

**FIGURE 12-10: TMR1, TMR2, AND TMR3 OPERATION IN TIMER MODE**

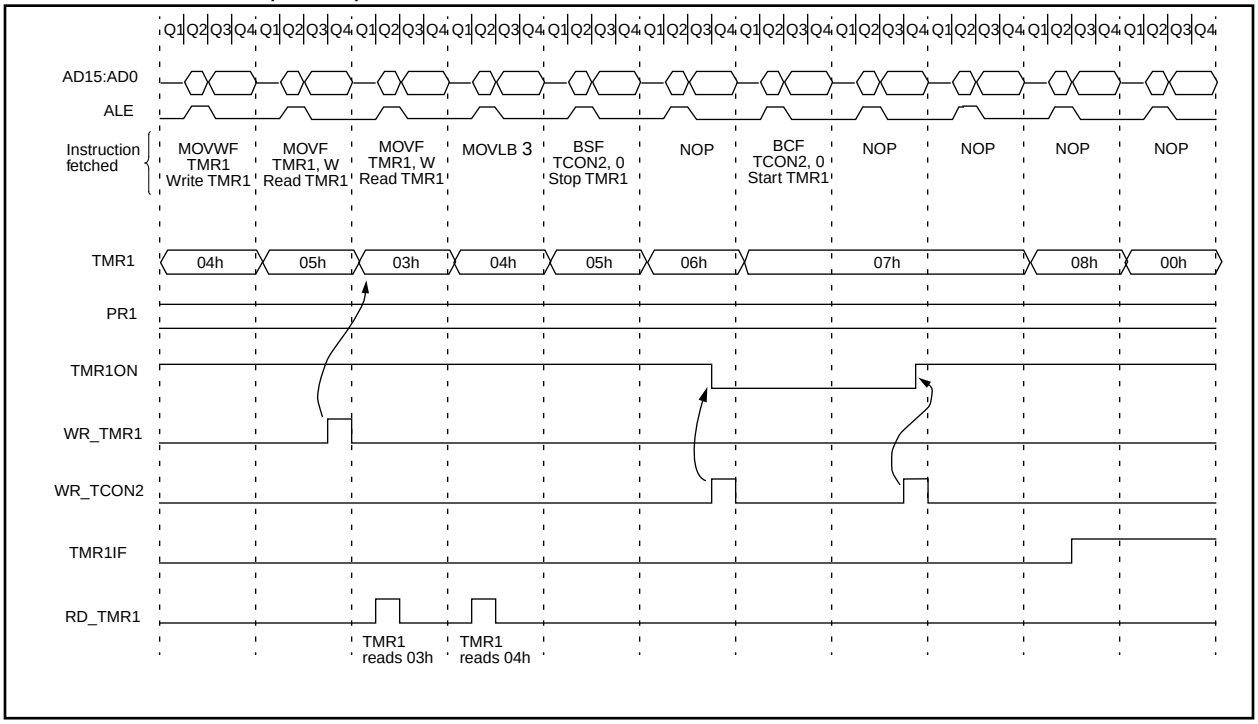



**TABLE 12-6: SUMMARY OF TMR1, TMR2, AND TMR3 REGISTERS**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other resets (Note1) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 16h, Bank 3 | TCON1 | CA2ED1 | CA2ED0 | CA1ED1 | CA1ED0 | T16 | TMR3CS | TMR2CS | TMR1CS | 0000 0000 | 0000 0000 |
| 17h, Bank 3 | TCON2 | CA2OVF | CA1OVF | PWM2ON | PWM1ON | CA1/$\overline{PR3}$ | TMR3ON | TMR2ON | TMR1ON | 0000 0000 | 0000 0000 |
| 10h, Bank 2 | TMR1 | Timer1 register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 11h, Bank 2 | TMR2 | Timer2 register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 12h, Bank 2 | TMR3L | TMR3 register; low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 13h, Bank 2 | TMR3H | TMR3 register; high byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 16h, Bank 1 | PIR | RBIF | TMR3IF | TMR2IF | TMR1IF | CA2IF | CA1IF | TXIF | RCIF | 0000 0010 | 0000 0010 |
| 17h, Bank 1 | PIE | RBIE | TMR3IE | TMR2IE | TMR1IE | CA2IE | CA1IE | TXIE | RCIE | 0000 0000 | 0000 0000 |
| 07h, Unbanked | INTSTA | PEIF | T0CKIF | T0IF | INTF | PEIE | T0CKIE | T0IE | INTE | 0000 0000 | 0000 0000 |
| 06h, Unbanked | CPUSTA | — | — | STKAV | GLINTD | $\overline{TO}$ | $\overline{PD}$ | — | — | --11 11-- | --11 qq-- |
| 14h, Bank 2 | PR1 | Timer1 period register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 15h, Bank 2 | PR2 | Timer2 period register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 16h, Bank 2 | PR3L/CA1L | Timer3 period/capture1 register; low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 17h, Bank 2 | PR3H/CA1H | Timer3 period/capture1 register; high byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10h, Bank 3 | PW1DCL | DC1 | DC0 | — | — | — | — | — | — | xx-- ---- | uu-- ---- |
| 11h, Bank 3 | PW2DCL | DC1 | DC0 | TM2PW2 | — | — | — | — | — | xx0- ---- | uu0- ---- |
| 12h, Bank 3 | PW1DCH | DC9 | DC8 | DC7 | DC6 | DC5 | DC4 | DC3 | DC2 | xxxx xxxx | uuuu uuuu |
| 13h, Bank 3 | PW2DCH | DC9 | DC8 | DC7 | DC6 | DC5 | DC4 | DC3 | DC2 | xxxx xxxx | uuuu uuuu |
| 14h, Bank 3 | CA2L | Capture2 low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 15h, Bank 3 | CA2H | Capture2 high byte | | | | | | | | xxxx xxxx | uuuu uuuu |

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', q - value depends on condition, shaded cells are not used by TMR1, TMR2 or TMR3.

Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and WDT Timer Reset.

### 13.2.2 USART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 13-4. The data comes in the RA4/RX/DT pin and drives the data recovery block. The data recovery block is actually a high speed shifter operating at 16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC.

Once asynchronous mode is selected, reception is enabled by setting bit CREN (RCSTA<4>).

The heart of the receiver is the receive (serial) shift register (RSR). After sampling the stop bit, the received data in the RSR is transferred to the RCREG (if it is empty). If the transfer is complete, the interrupt bit RCIF (PIR<0>) is set. The actual interrupt can be enabled/disabled by setting/clearing the RCIE (PIE<0>) bit. RCIF is a read only bit which is cleared by the hardware. It is cleared when RCREG has been read and is empty. RCREG is a double buffered register; (i.e. it is a two deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte begin shifting to the RSR. On detection of the stop bit of the third byte, if the RCREG is still full, then the overrun error bit, OERR (RCSTA<1>) will be set. The word in the RSR will be lost. RCREG can be read twice to retrieve the two bytes in the FIFO. The OERR bit has to be cleared in software which is done by resetting the receive logic (CREN is set). If the OERR bit is set, transfers from the RSR to RCREG are inhibited, so it is essential to clear the OERR bit if it is set. The framing error bit FERR (RCSTA<2>) is set if a stop bit is not detected.

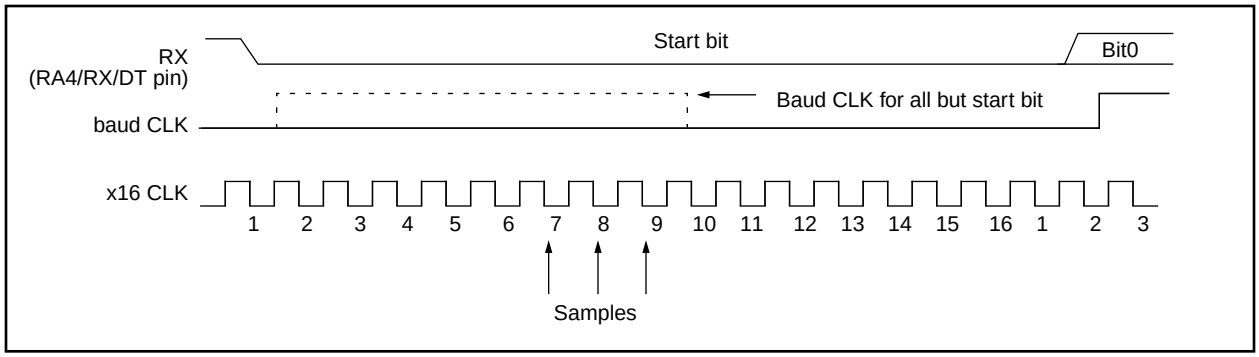
| **Note:** | The FERR and the 9th receive bit are buffered the same way as the receive data. Reading the RCREG register will allow the RX9D and FERR bits to be loaded with values for the next received Received data; therefore, it is essential for the user to read the RCSTA register before reading RCREG in order not to lose the old FERR and RX9D information. |
|---|---|

### 13.2.3 SAMPLING

The data on the RA4/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RA4/RX/DT pin. The sampling is done on the seventh, eighth and ninth falling edges of a x16 clock (Figure 11-3).

The x16 clock is a free running clock, and the three sample points occur at a frequency of every 16 falling edges.

**FIGURE 13-7: RX PIN SAMPLING SCHEME**



### 13.3.2 USART SYNCHRONOUS MASTER RECEPTION

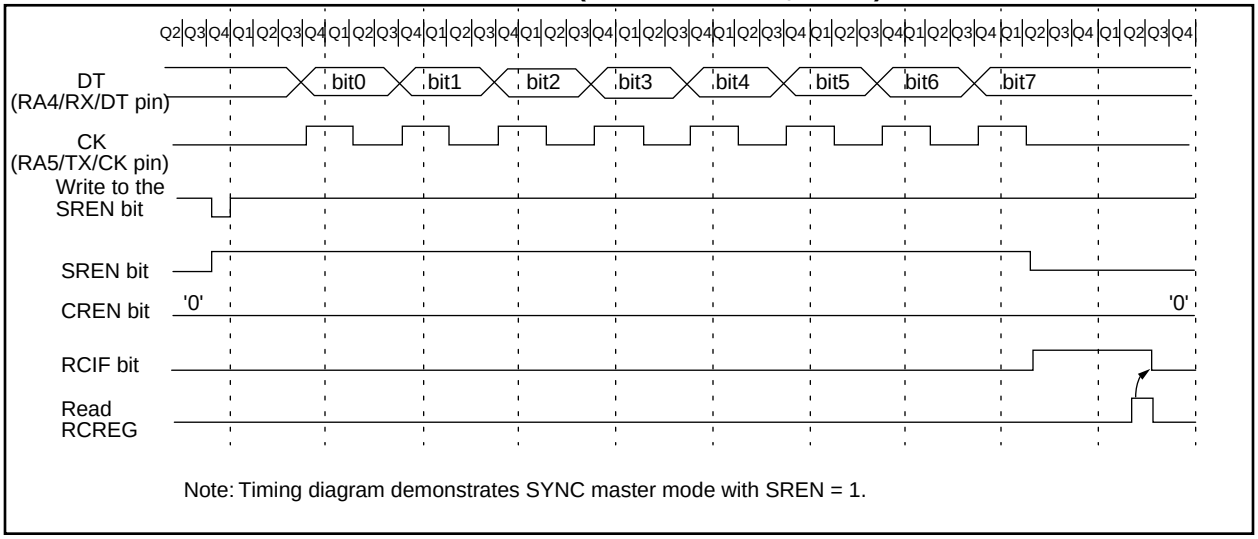
Once synchronous mode is selected, reception is enabled by setting either the SREN (RCSTA<5>) bit or the CREN (RCSTA<4>) bit. Data is sampled on the RA4/RX/DT pin on the falling edge of the clock. If SREN is set, then only a single word is received. If CREN is set, the reception is continuous until CREN is reset. If both bits are set, then CREN takes precedence. After clocking the last bit, the received data in the Receive Shift Register (RSR) is transferred to RCREG (if it is empty). If the transfer is complete, the interrupt bit RCIF (PIR<0>) is set. The actual interrupt can be enabled/disabled by setting/clearing the RCIE (PIE<0>) bit. RCIF is a read only bit which is RESET by the hardware. In this case it is reset when RCREG has been read and is empty. RCREG is a double buffered register; i.e., it is a two deep FIFO. It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting into the RSR. On the clocking of the last bit of the third byte, if RCREG is still full, then the overrun error bit OERR (RCSTA<1>) is set. The word in the RSR will be lost. RCREG can be read twice to retrieve the two bytes in the FIFO. The OERR bit has to be cleared in software. This is done by clearing the CREN bit. If OERR bit is set, transfers from RSR to RCREG are inhibited, so it is essential to clear OERR bit if it is set. The 9th receive bit is buffered the same way as the receive data. Reading the RCREG register will allow the RX9D and FERR bits to be loaded with values for the next received data; therefore, it is essential for the user to read the RCSTA register before reading RCREG in order not to lose the old FERR and RX9D information.

Steps to follow when setting up a Synchronous Master Reception:

1. Initialize the SPBRG register for the appropriate baud rate. See Section 13.1 for details.
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. If interrupts are desired, then set the RCIE bit.
4. If 9-bit reception is desired, then set the RX9 bit.
5. If a single reception is required, set bit SREN. For continuous reception set bit CREN.
6. The RCIF bit will be set when reception is complete and an interrupt will be generated if the RCIE bit was set.
7. Read RCSTA to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading RCREG.
9. If any error occurred, clear the error by clearing CREN.

| **Note:** | To terminate a reception, either clear the SREN and CREN bits, or the SPEN bit. This will reset the receive logic, so that it will be in the proper state when receive is re-enabled. |
|---|---|

**FIGURE 13-11: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**




Note: Timing diagram demonstrates SYNC master mode with SREN = 1.

**CLRWDT** **Clear Watchdog Timer**

Syntax: [ *label* ] CLRWDT

Operands: None

Operation: 00h → WDT
0 → WDT postscaler,
1 → $\overline{TO}$
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

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

Description: CLRWDT instruction resets the watchdog timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register ALUSTA | Execute | NOP |

Example: CLRWDT

Before Instruction
WDT counter = ?

After Instruction
WDT counter = 0x00
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

**COMF** **Complement f**

Syntax: [ *label* ] COMF f,d

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]

Operation: $(\bar{f})$ → (dest)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 001d | ffff | ffff |

Description: The contents of register 'f' are complemented. If 'd' is 0 the result is stored in WREG. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Execute | Write register 'f' |

Example: COMF REG1,0

Before Instruction
REG1 = 0x13

After Instruction
REG1 = 0x13
WREG = 0xEC

**RETFIE** **Return from Interrupt**

Syntax: [ *label* ] RETFIE

Operands: None

Operation: TOS → (PC);
0 → GLINTD;
PCLATH is unchanged.

Status Affected: GLINTD

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0101 |

Description: Return from Interrupt. Stack is POP'ed and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by clearing the GLINTD bit. GLINTD is the global interrupt disable bit (CPUSTA<4>).

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register T0STA | Execute | NOP |
| Forced NOP | NOP | Execute | NOP |

Example:

```
RETFIE
```

After Interrupt
PC = TOS
GLINTD = 0

**RETLW** **Return Literal to WREG**

Syntax: [ *label* ] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (WREG); TOS → (PC);
PCLATH is unchanged

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1011 | 0110 | kkkk | kkkk |

Description: WREG is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Execute | Write to WREG |
| Forced NOP | NOP | Execute | NOP |

Example:

```
      CALL TABLE ; WREG contains table
                 ;  offset value
                 ;  WREG now has
                 ;  table value
        :
TABLE
        ADDWF PC   ; WREG = offset
        RETLW k0   ; Begin table
        RETLW k1   ;
        :
        :
        RETLW kn   ; End of table
```

Before Instruction
WREG = 0x07

After Instruction
WREG = value of k7

| Applicable Devices | 42 | R42 | 42A | 43 | R43 | 44 |
|---|---|---|---|---|---|---|

## 17.0 PIC17C42 ELECTRICAL CHARACTERISTICS

**Absolute Maximum Ratings †**

Ambient temperature under bias ..... -55 to +125˚C

Storage temperature ..... -65˚C to +150˚C

Voltage on $V_{DD}$ with respect to $V_{SS}$ ..... 0 to +7.5V

Voltage on $\overline{MCLR}$ with respect to $V_{SS}$ (Note 2) ..... -0.6V to +14V

Voltage on RA2 and RA3 with respect to $V_{SS}$ ..... -0.6V to +12V

Voltage on all other pins with respect to $V_{SS}$ ..... -0.6V to $V_{DD}$ + 0.6V

Total power dissipation (Note 1) ..... 1.0W

Maximum current out of $V_{SS}$ pin(s) - Total ..... 250 mA

Maximum current into $V_{DD}$ pin(s) - Total ..... 200 mA

Input clamp current, $I_{IK}$ ($V_I$ < 0 or $V_I$ > $V_{DD}$) ..... ±20 mA

Output clamp current, $I_{OK}$ ($V_O$ < 0 or $V_O$ > $V_{DD}$) ..... ±20 mA

Maximum output current sunk by any I/O pin (except RA2 and RA3) ..... 35 mA

Maximum output current sunk by RA2 or RA3 pins ..... 60 mA

Maximum output current sourced by any I/O pin ..... 20 mA

Maximum current sunk by PORTA and PORTB (combined) ..... 150 mA

Maximum current sourced by PORTA and PORTB (combined) ..... 100 mA

Maximum current sunk by PORTC, PORTD and PORTE (combined) ..... 150 mA

Maximum current sourced by PORTC, PORTD and PORTE (combined) ..... 100 mA

**Note 1:** Power dissipation is calculated as follows: $Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD}-V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

**Note 2:** Voltage spikes below $V_{SS}$ at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin rather than pulling this pin directly to $V_{SS}$.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

| Applicable Devices | 42 | R42 | 42A | 43 | R43 | 44 |
|---|---|---|---|---|---|---|

## 18.0 PIC17C42 DC AND AC CHARACTERISTICS

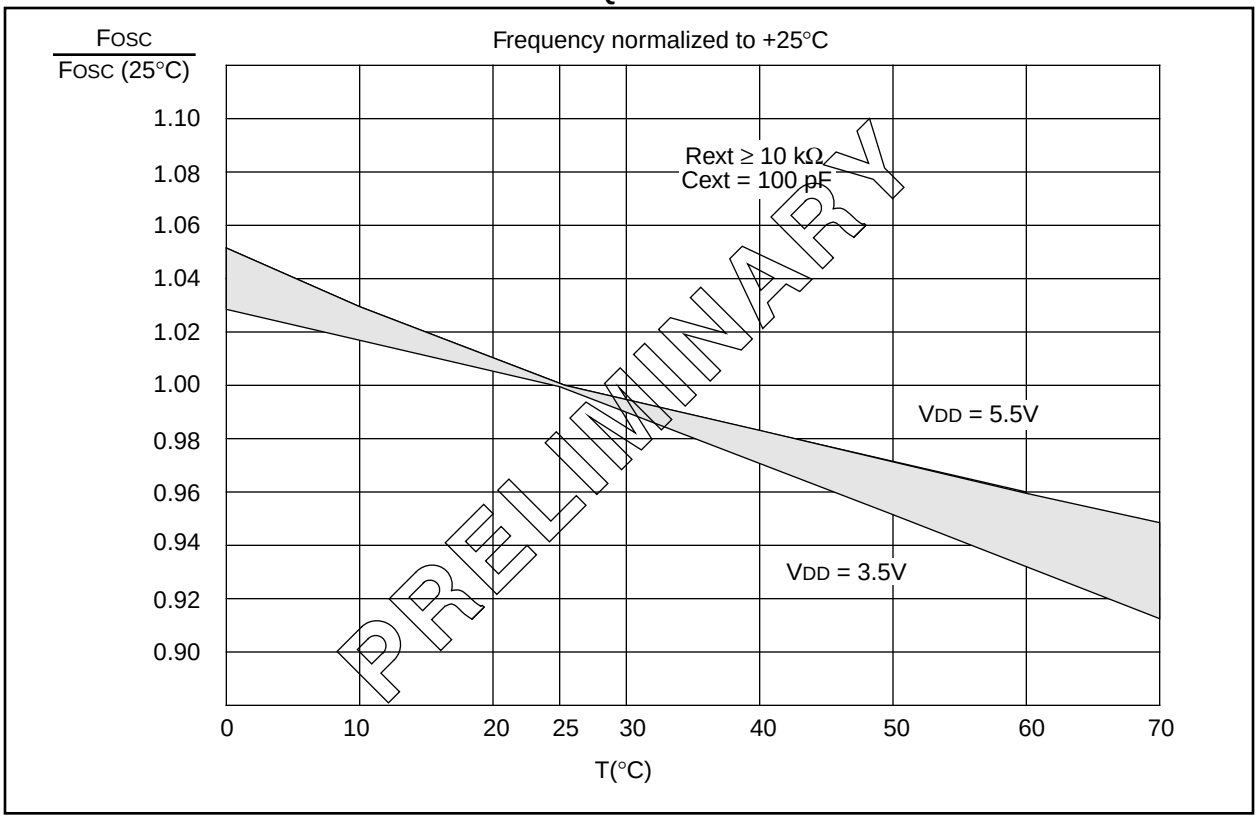
The graphs and tables provided in this section are for design guidance and are not tested or guaranteed. In some graphs or tables the data presented are outside specified operating range (e.g. outside specified VDD range). This is for information only and devices are ensured to operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. "Typical" represents the mean of the distribution while "max" or "min" represents (mean + 3σ) and (mean - 3σ) respectively where σ is standard deviation.

**TABLE 18-1: PIN CAPACITANCE PER PACKAGE TYPE**

| Pin Name | Typical Capacitance (pF) | | | |
|---|---|---|---|---|
| | 40-pin DIP | 44-pin PLCC | 44-pin MQFP | 44-pin TQFP |
| All pins, except $\overline{MCLR}$, VDD, and VSS | 10 | 10 | 10 | 10 |
| $\overline{MCLR}$ pin | 20 | 20 | 20 | 20 |

**FIGURE 18-1: TYPICAL RC OSCILLATOR FREQUENCY vs. TEMPERATURE**



| Applicable Devices | 42 | R42 | 42A | 43 | R43 | 44 |
|---|---|---|---|---|---|---|

**FIGURE 19-9: USART MODULE: SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**

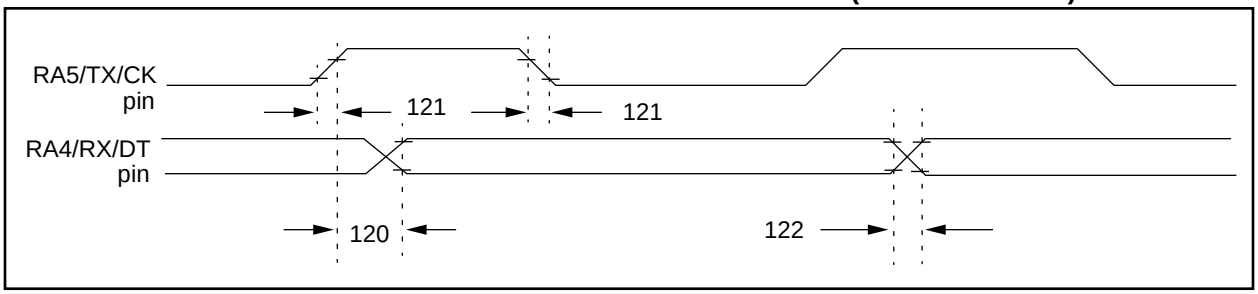



**TABLE 19-9: SYNCHRONOUS TRANSMISSION REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 120 | TckH2dtV | SYNC XMIT (MASTER & SLAVE) Clock high to data out valid | PIC17CR42/42A/43/R43/44 | — | — | 50 | ns | |
| | | | PIC17LCR42/42A/43/R43/44 | — | — | 75 | ns | |
| 121 | TckRF | Clock out rise time and fall time (Master Mode) | PIC17CR42/42A/43/R43/44 | — | — | 25 | ns | |
| | | | PIC17LCR42/42A/43/R43/44 | — | — | 40 | ns | |
| 122 | TdtRF | Data out rise time and fall time | PIC17CR42/42A/43/R43/44 | — | — | 25 | ns | |
| | | | PIC17LCR42/42A/43/R43/44 | — | — | 40 | ns | |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 19-10: USART MODULE: SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**

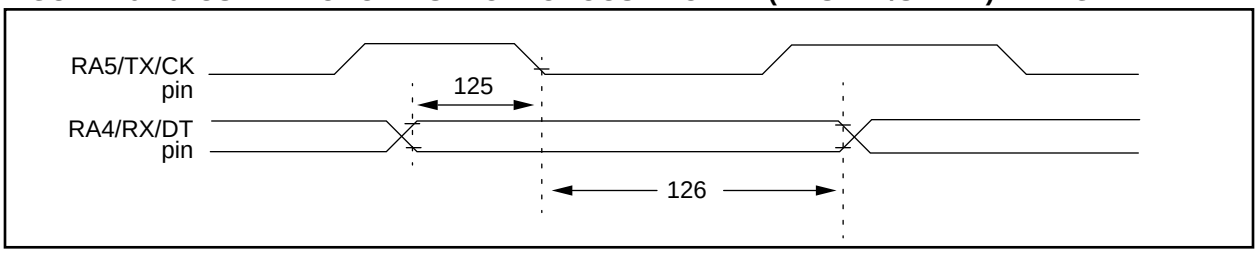



**TABLE 19-10: SYNCHRONOUS RECEIVE REQUIREMENTS**

| Parameter No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 125 | TdtV2ckL | SYNC RCV (MASTER & SLAVE) Data hold before CK↓ (DT hold time) | 15 | — | — | ns | |
| 126 | TckL2dtl | Data hold after CK↓ (DT hold time) | 15 | — | — | ns | |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

| Applicable Devices | 42 | R42 | 42A | 43 | R43 | 44 |
|---|---|---|---|---|---|---|

**FIGURE 20-11: TYPICAL IPD vs. VDD WATCHDOG ENABLED 25°C**

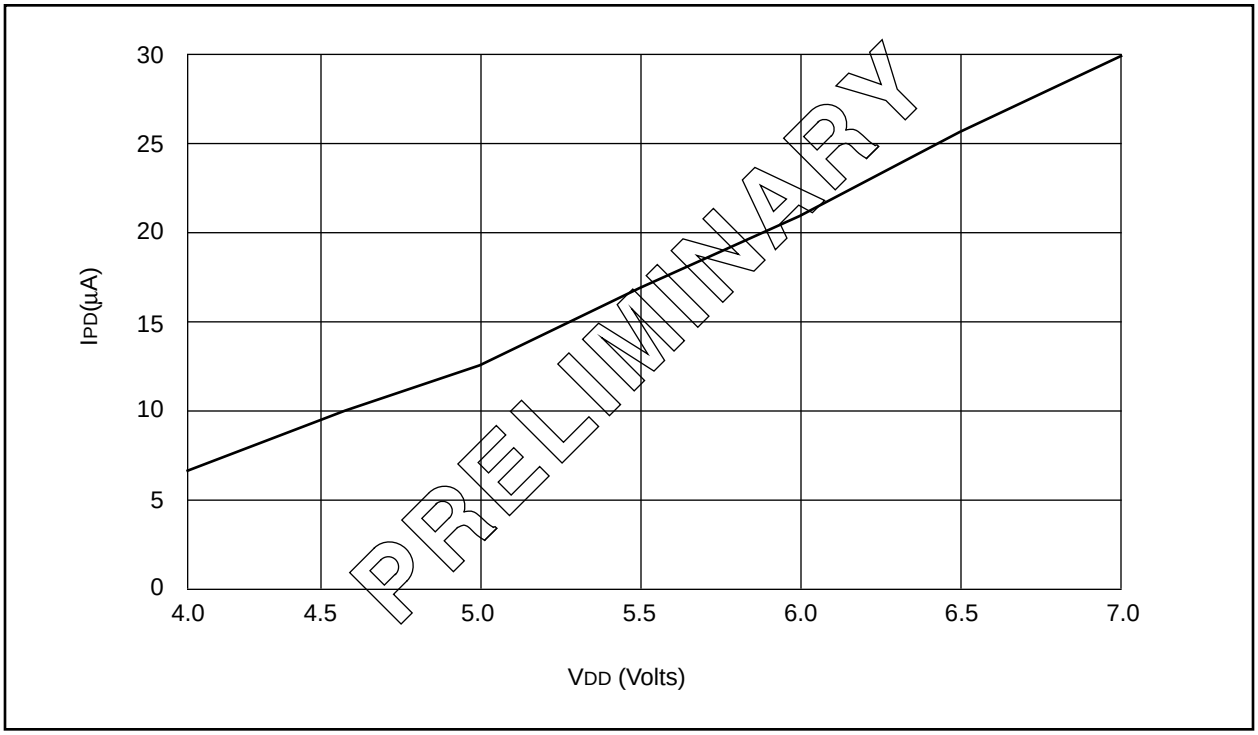
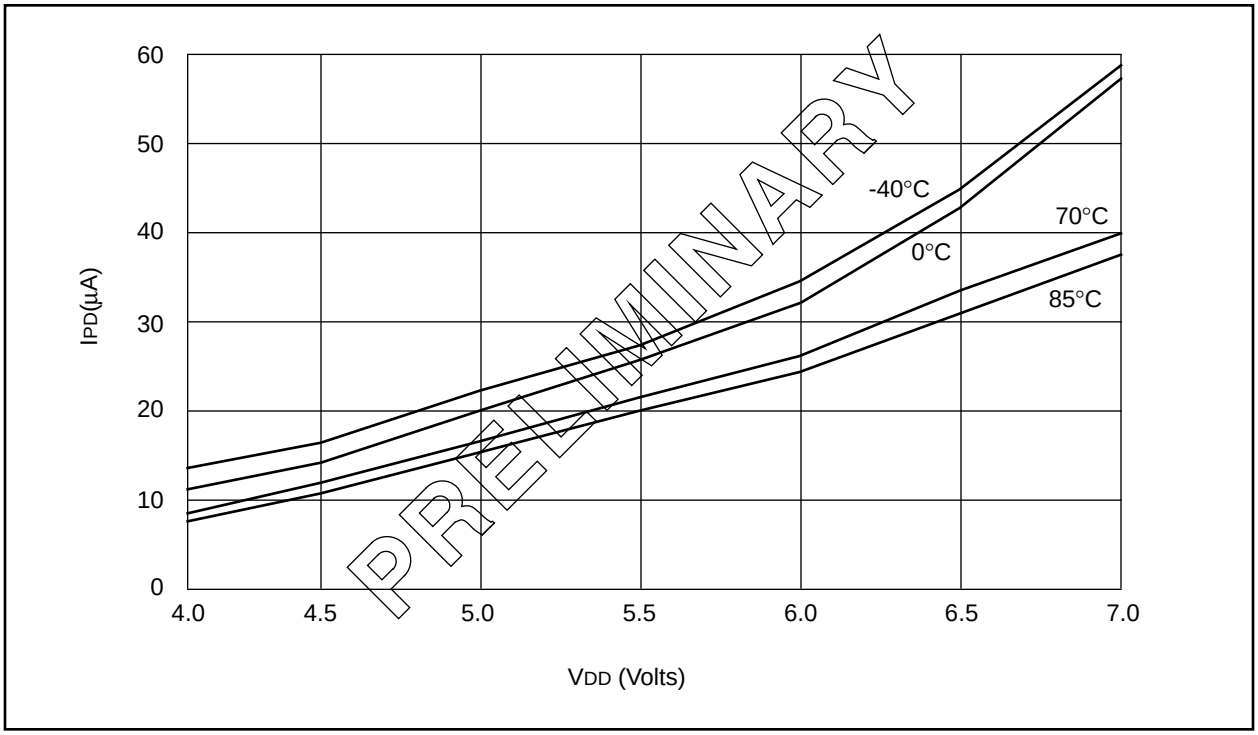



**FIGURE 20-12: MAXIMUM IPD vs. VDD WATCHDOG ENABLED**



## N



## O



## P



## R