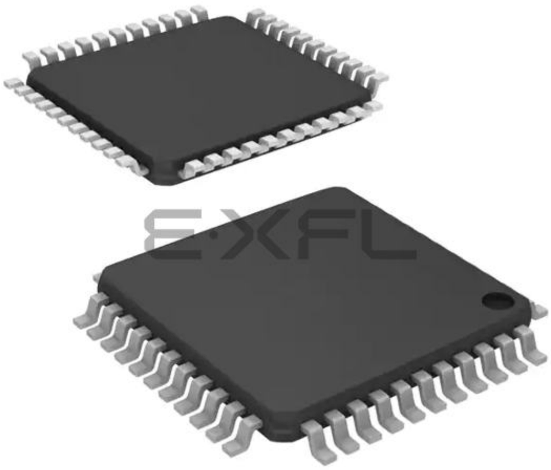


Welcome to E-XFL.COM

### 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 | 20MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 36 |
| Program Memory Size | 14KB (8K x 14) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 363 x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 14x8b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 125°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-TQFP |
| Supplier Device Package | 44-TQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic16lf707-e-pt |

# 1.0 DEVICE OVERVIEW

The PIC16(L)F707 devices are covered by this data sheet. They are available in 40/44-pin packages. Figure 1-1 shows a block diagram of the PIC16(L)F707 devices. Table 1-1 shows the pinout descriptions.

**TABLE 1-1: PIC16(L)F707 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RC4/SDI/SDA | RC4 | ST | CMOS | General purpose I/O. |
| | SDI | ST | — | SPI data input. |
| | SDA | $I^2C$ | OD | $I^2C$ data input/output. |
| RC5/SDO/CPSA9 | RC5 | ST | CMOS | General purpose I/O. |
| | SDO | — | CMOS | SPI data output. |
| | CPSA9 | AN | — | Capacitive sensing A input 9. |
| RC6/TX/CK/CPSA10 | RC6 | ST | CMOS | General purpose I/O. |
| | TX | — | CMOS | USART asynchronous transmit. |
| | CK | ST | CMOS | USART synchronous clock. |
| | CPSA10 | AN | — | Capacitive sensing A input 10. |
| RC7/RX/DT/CPSA11 | RC7 | ST | CMOS | General purpose I/O. |
| | RX | ST | — | USART asynchronous input. |
| | DT | ST | CMOS | USART synchronous data. |
| | CPSA11 | AN | — | Capacitive sensing A input 11. |
| RD0/CPSB5/T3G | RD0 | ST | CMOS | General purpose I/O. |
| | CPSB5 | AN | — | Capacitive sensing B input 5. |
| | T3G | ST | — | Timer3 Gate input. |
| RD1/CPSB6 | RD1 | ST | CMOS | General purpose I/O. |
| | CPSB6 | AN | — | Capacitive sensing B input 6. |
| RD2/CPSB7 | RD2 | ST | CMOS | General purpose I/O. |
| | CPSB7 | AN | — | Capacitive sensing B input 7. |
| RD3/CPSA8 | RD3 | ST | CMOS | General purpose I/O. |
| | CPSA8 | AN | — | Capacitive sensing A input 8. |
| RD4/CPSA12 | RD4 | ST | CMOS | General purpose I/O. |
| | CPSA12 | AN | — | Capacitive sensing A input 12. |
| RD5/CPSA13 | RD5 | ST | CMOS | General purpose I/O. |
| | CPSA13 | AN | — | Capacitive sensing A input 13. |
| RD6/CPSA14 | RD6 | ST | CMOS | General purpose I/O. |
| | CPSA14 | AN | — | Capacitive sensing A input 14. |
| RD7/CPSA15 | RD7 | ST | CMOS | General purpose I/O. |
| | CPSA15 | AN | — | Capacitive sensing A input 15. |
| RE0/AN5/CPSA5 | RE0 | ST | CMOS | General purpose I/O. |
| | AN5 | AN | — | A/D Channel 5 input. |
| | CPSA5 | AN | — | Capacitive sensing A input 5. |
| RE1/AN6/CPSA6 | RE1 | ST | CMOS | General purpose I/O. |
| | AN6 | AN | — | A/D Channel 6 input. |
| | CPSA6 | AN | — | Capacitive sensing A input 6. |
| RE2/AN7/CPSA7 | RE2 | ST | CMOS | General purpose I/O. |
| | AN7 | AN | — | A/D Channel 7 input. |
| | CPSA7 | AN | — | Capacitive sensing A input 7. |
| RE3/$\overline{\text{MCLR}}$/VPP | RE3 | TTL | — | General purpose input. |
| | $\overline{\text{MCLR}}$ | ST | — | Master Clear with internal pull-up. |
| | VPP | HV | — | Programming voltage. |
| VDD | VDD | Power | — | Positive supply. |

**Legend:** AN = Analog input or output; CMOS = CMOS compatible input or output; OD = Open Drain; TTL = TTL compatible input; ST = Schmitt Trigger input with CMOS levels; $I^2C$ = Schmitt Trigger input with $I^2C$ levels; HV = High Voltage; XTAL = Crystal

**TABLE 3-5: INITIALIZATION CONDITION FOR REGISTERS (CONTINUED)**

| Register | Address | Power-on Reset/ Brown-out Reset(1) | $\overline{MCLR}$ Reset/ WDT Reset | Wake-up from Sleep through Interrupt/Time-out |
|---|---|---|---|---|
| ADCON0 | 1Fh | --00 0000 | --00 0000 | --uu uuuu |
| OPTION_REG | 81h/181h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA | 85h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISB | 86h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISC | 87h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISD | 88h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISE | 89h | ---- 1111 | ---- 1111 | ---- uuuu |
| PIE1 | 8Ch | 0000 0000 | 0000 0000 | uuuu uuuu |
| PIE2 | 8Dh | 0000 ---0 | 0000 ---0 | uuuu ---u |
| PCON | 8Eh | ---- --qq | ---- --uu(1,5) | ---- --uu |
| T1GCON | 8Fh | 0000 0x00 | uuuu uxuu | uuuu uxuu |
| OSCCON | 90h | --10 qq-- | --10 qq-- | --uu qq-- |
| OSCTUNE | 91h | --00 0000 | --uu uuuu | --uu uuuu |
| PR2 | 92h | 1111 1111 | 1111 1111 | uuuu uuuu |
| SSPADD | 93h | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSPMSK | 93h | 1111 1111 | 1111 1111 | uuuu uuuu |
| SSPSTAT | 94h | 0000 0000 | 0000 0000 | uuuu uuuu |
| WPUB | 95h | 1111 1111 | 1111 1111 | uuuu uuuu |
| IOCB | 96h | 0000 0000 | 0000 0000 | uuuu uuuu |
| T3CON | 97h | 0000 -0-0 | 0000 -0-0 | uuuu -u-u |
| TXSTA | 98h | 0000 -010 | 0000 -010 | uuuu -uuu |
| SPBRG | 99h | 0000 0000 | 0000 0000 | uuuu uuuu |
| TMR3L | 9Ah | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR3H | 9Bh | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| APFCON | 9Ch | ---- --00 | ---- --00 | ---- --uu |
| FVRCON | 9Dh | q000 0000 | q000 0000 | q000 0000 |
| ADCON1 | 9Fh | -000 --00 | -000 --00 | -uuu --uu |
| TACON | 105h | 0-00 0000 | 0-00 0000 | u-uu uuuu |
| CPSBCON0 | 106h | 00-- 0000 | 00-- 0000 | uu-- uuuu |
| CPSBCON1 | 107h | ---- 0000 | ---- 0000 | ---- uuuu |
| CPSACON0 | 108h | 00-- 0000 | 00-- 0000 | uu-- uuuu |
| CPSACON1 | 109h | ---- 0000 | ---- 0000 | ---- uuuu |
| PMDATL | 10Ch | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| PMADRL | 10Dh | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| PMDATH | 10Eh | --xx xxxx | --xx xxxx | --uu uuuu |
| PMADRH | 10Fh | ---x xxxx | ---x xxxx | ---u uuuu |
| TMRA | 110h | 0000 0000 | 0000 0000 | uuuu uuuu |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, reads as '0', q = value depends on condition.

**Note 1:** If $V_{DD}$ goes too low, Power-on Reset will be activated and registers will be affected differently.

**2:** One or more bits in INTCON and/or PIR1 and PIR2 will be affected (to cause wake-up).

**3:** When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

**4:** See Table 3-2 for Reset value for specific condition.

**5:** If Reset was due to brown-out, then bit 0 = 0. All other Resets will cause bit 0 = u.

## 4.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt enable bit(s) for the specific interrupt event(s)
- PEIE bit of the INTCON register (if the interrupt enable bit of the interrupt event is contained in the PIE1 and PIE2 registers)

The INTCON, PIR1 and PIR2 registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual Interrupt Enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- PC is loaded with the interrupt vector 0004h

The ISR determines the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack and setting the GIE bit.

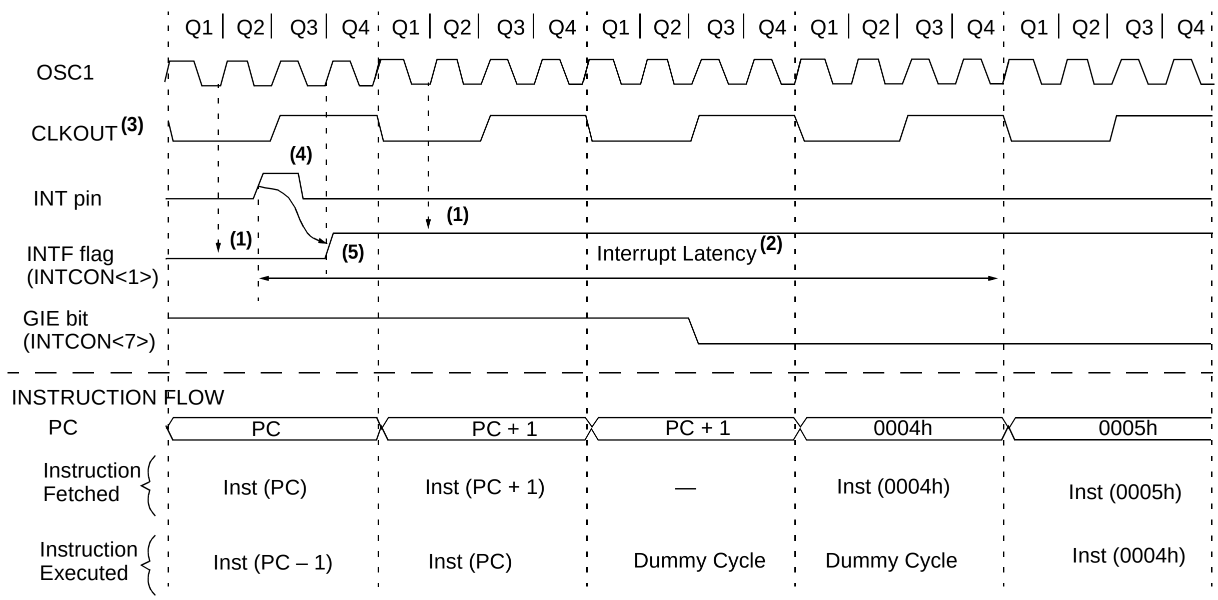
For additional information on a specific interrupt's operation, refer to its peripheral chapter.

> **Note 1:** Individual interrupt flag bits are set, regardless of the state of any other enable bits.
>
> **2:** All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

## 4.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three instruction cycles. For asynchronous interrupts, the latency is three to four instruction cycles, depending on when the interrupt occurs. See Figure 4-2 for timing details.

**FIGURE 4-2: INT PIN INTERRUPT TIMING**




**Note 1:** INTF flag is sampled here (every Q1).

**2:** Asynchronous interrupt latency = 3-4 TCY. Synchronous latency = 3 TCY, where TCY = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.

**3:** CLKOUT is available only in INTOSC and RC Oscillator modes.

**4:** For minimum width of INT pulse, refer to AC specifications in **Section 25.0 "Electrical Specifications"**.

**5:** INTF is enabled to be set any time during the Q4-Q1 cycles.

**TABLE 4-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPTS**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000x |
| OPTION_REG | $\overline{\text{RBPU}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| PIE2 | TMR3GIE | TMR3IE | TMRBIE | TMRAIE | — | — | — | CCP2IE | 0000 ---0 | 0000 ---0 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIR2 | TMR3GIF | TMR3IF | TMRBIF | TMRAIF | — | — | — | CCP2IF | 0000 ---0 | 0000 ---0 |

**Legend:** - = Unimplemented locations, read as '0', u = unchanged, x = unknown. Shaded cells are not used by interrupts.

#### 6.4.2.3 RC2/CCP1/CPSB4/TBCKI

These pins are configurable to function as one of the following:

- General purpose I/O
- Capture 1 input, Compare 1 output, and PWM1 output
- Capacitive sensing input
- TimerB Clock input

#### 6.4.2.4 RC3/SCK/SCL

These pins are configurable to function as one of the following:

- General purpose I/O
- SPI clock
- I²C clock

#### 6.4.2.5 RC4/SDI/SDA

These pins are configurable to function as one of the following:

- General purpose I/O
- SPI data input
- I²C data I/O

#### 6.4.2.6 RC5/SDO/CPSA9

These pins are configurable to function as one of the following:

- General purpose I/O
- SPI data output
- Capacitive sensing input

#### 6.4.2.7 RC6/TX/CK/CPSA10

These pins are configurable to function as one of the following:

- General purpose I/O
- Asynchronous serial output
- Synchronous clock I/O
- Capacitive sensing input

#### 6.4.2.8 RC7/RX/DT/CPSA11

These pins are configurable to function as one of the following:

- General purpose I/O
- Asynchronous serial input
- Synchronous serial data I/O
- Capacitive sensing input

**TABLE 6-3: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| ANSELC | ANSC7 | ANSC6 | ANSC5 | — | — | ANSC2 | ANSC1 | ANSC0 | 111- -111 | 111- -111 |
| APFCON | — | — | — | — | — | — | SSSEL | CCP2SEL | ---- --00 | ---- --00 |
| CCP1CON | — | — | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | --00 0000 |
| CPSACON0 | CPSAON | CPSARM | — | — | CPSARNG1 | CPSARNG0 | CPSAOUT | TAXCS | 00-- 0000 | 00-- 0000 |
| CPSACON1 | — | — | — | — | CPSACH3 | CPSACH2 | CPSACH1 | CPSACH0 | ---- 0000 | ---- 0000 |
| CPSBCON0 | CPSBON | CPSBRM | — | — | CPSBRNG1 | CPSBRNG0 | CPSBOUT | TBXCS | 00-- 0000 | 00-- 0000 |
| CPSBCON1 | — | — | — | — | CPSBCH3 | CPSBCH2 | CPSBCH1 | CPSBCH0 | ---- 0000 | ---- 0000 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | xxxx xxxx |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| SSPCON | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 0000 0000 |
| SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |
| T1CON | TMR1CS1 | TMR1CS0 | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | — | TMR1ON | 0000 00-0 | uuuu uu-u |
| TBCON | TMRBON | — | TBCS | TBSE | TBPSA | TBPS2 | TBPS1 | TBPS0 | 0-00 0000 | 0-00 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 |

**Legend:** x = unknown, u = unchanged, - = unimplemented locations read as ‘0’. Shaded cells are not used by PORTC.

## 11.0 DIGITAL-TO-ANALOG CONVERTER (DAC) MODULE

The Digital-to-Analog Converter supplies a variable voltage reference, ratiometric with $V_{DD}$, with 32 selectable output levels. The output of the DAC can be configured to supply a reference voltage to the following:

- DACOUT device pin
- Capacitive sensing modules

The Digital-to-Analog Converter (DAC) can be enabled by setting the DACEN bit of the DACCON0 register.

### 11.1 Output Voltage Selection

The DAC has 32 voltage level ranges. The 32 levels are set with the DACR<4:0> bits of the DACCON1 register.

The DAC output voltage is determined by the following equation:

**EQUATION 11-1:**

***IF DACEN = 1***

$$V_{OUT} = \left((V_{SOURCE+} - V_{SOURCE-}) \times \frac{DACR[4:0]}{2^5}\right) + V_{SOURCE-}$$

***IF DACEN = 0 & DACLPS = 1 & DACR[4:0] = 11111***

$$V_{OUT} = V_{SOURCE+}$$

***IF DACEN = 0 & DACLPS = 0 & DACR[4:0] = 00000***

$$V_{OUT} = V_{SOURCE-}$$

$V_{SOURCE+} = V_{DD}$, $V_{REF}$, or FVR BUFFER 2

$V_{SOURCE-} = V_{SS}$

### 11.2 Output Clamped to VSS

The DAC output voltage can be set to $V_{SS}$ with no power consumption by setting the DACEN bit of the DACCON0 register to '0'.

### 11.3 Output Ratiometric to VDD

The DAC is $V_{DD}$ derived and therefore, the DAC output changes with fluctuations in $V_{DD}$. The tested absolute accuracy of the DAC can be found in **Section 25.0 "Electrical Specifications"**.

### 11.4 Voltage Reference Output

The DAC can be output to the device DACOUT pin by setting the DACOE bit of the DACCON0 register to '1'. Selecting the reference voltage for output on the DACOUT pin automatically overrides the digital output buffer and digital input threshold detector functions of that pin. Reading the DACOUT pin when it has been configured for reference voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the voltage reference output for external connections to DACOUT. Example 11-1 shows an example buffering technique.

### 11.5 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DACCON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

### 11.6 Effects of a Reset

A device Reset affects the following:

- Voltage reference is disabled
- Fixed Voltage Reference is disabled
- DAC is removed from the DACOUT pin
- The DACR<4:0> range select bits are cleared

## 12.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

### 12.1.1 8-BIT TIMER MODE

The Timer0 module will increment every instruction cycle, if used without a prescaler. 8-bit Timer mode is selected by clearing the TMR0CS bit of the OPTION register.

When TMR0 is written, the increment is inhibited for two instruction cycles immediately following the write.

| **Note:** | The value written to the TMR0 register can be adjusted, in order to account for the two instruction cycle delay when TMR0 is written. |
|---|---|

### 12.1.2 8-BIT COUNTER MODE

In 8-bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin. 8-bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit of the OPTION register to '1'.

The rising or falling transition of the incrementing edge for either input source is determined by the TMR0SE bit in the OPTION register.

### 12.1.3 SOFTWARE PROGRAMMABLE PRESCALER

A single software programmable prescaler is available for use with either Timer0 or the Watchdog Timer (WDT), but not both simultaneously. The prescaler assignment is controlled by the PSA bit of the OPTION register. To assign the prescaler to Timer0, the PSA bit must be cleared to a '0'.

There are eight prescaler options for the Timer0 module ranging from 1:2 to 1:256. The prescale values are selectable via the PS<2:0> bits of the OPTION register. In order to have a 1:1 prescaler value for the Timer0 module, the prescaler must be assigned to the WDT module.

The prescaler is not readable or writable. When the prescaler is enabled or assigned to the Timer0 module, all instructions writing to the TMR0 register will clear the prescaler.

| **Note:** | When the prescaler is assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the WDT. |
|---|---|

### 12.1.4 TIMER0 INTERRUPT

Timer0 will generate an interrupt when the TMR0 register overflows from FFh to 00h. The TMR0IF interrupt flag bit of the INTCON register is set every time the TMR0 register overflows, regardless of whether or not the Timer0 interrupt is enabled. The TMR0IF bit can only be cleared in software. The Timer0 interrupt enable is the TMR0IE bit of the INTCON register.

| **Note:** | The Timer0 interrupt cannot wake the processor from Sleep since the timer is frozen during Sleep. |
|---|---|

### 12.1.5 USING TIMER0 WITH AN EXTERNAL CLOCK

When Timer0 is in Counter mode, the synchronization of the T0CKI input and the Timer0 register is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks. Therefore, the high and low periods of the external clock source must meet the timing requirements as shown in **Section 25.0 "Electrical Specifications"**.

### 12.1.6 TIMER ENABLE

Operation of Timer0 is always enabled and the module will operate according to the settings of the OPTION register.

### 12.1.7 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.

### REGISTER 16-1: CPSxCON0: CAPACITIVE SENSING CONTROL REGISTER 0

| R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| CPSxON | CPSxRM | — | — | CPSxRNG1 | CPSxRNG0 | CPSxOUT | TxXCS |
| 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 **CPSxON:** Capacitive Sensing Module Enable bit
1 = Capacitive sensing module is enabled
0 = Capacitive sensing module is disabled

bit 6 **CPSxRM:** Capacitive Sensing Reference Mode bit
1 = Capacitive sensing module is in high range. DAC and FVR provide oscillator voltage references.
0 = Capacitive sensing module is in low range. Internal oscillator voltage references are used.

bit 5-4 **Unimplemented:** Read as ‘0’

bit 3-2 **CPSxRNG<1:0>:** Capacitive Sensing Current Range bits
If CPSxRM = 0 (low range):
11 = Oscillator is in high range: Charge/discharge current is nominally 18 µA.
10 = Oscillator is in medium range. Charge/discharge current is nominally 1.2 µA.
01 = Oscillator is in low range. Charge/discharge current is nominally 0.1 µA.
00 = Oscillator is off.
If CPSxRM = 1 (high range):
11 = Oscillator is in high range: Charge/discharge current is nominally 100 µA.
10 = Oscillator is in medium range. Charge/discharge current is nominally 30 µA.
01 = Oscillator is in low range. Charge/discharge current is nominally 9 µA.
00 =Oscillator is on; Noise Detection mode; No charge/discharge current is supplied.

bit 1 **CPSxOUT:** Capacitive Sensing Oscillator Status bit
1 = Oscillator is sourcing current (Current flowing out of the pin)
0 = Oscillator is sinking current (Current flowing into the pin)

bit 0 **TxXCS:** TimerA/B External Clock Source Select bit
If TMRxCS = 1:
The TxXCS bit controls which clock external to the core/TimerA/B module supplies TimerA/B:
1 = TimerA/B clock source is the capacitive sensing oscillator
0 = TimerA/B clock source is the TxCKI pin
If TMRxCS = 0:
TimerA/B clock source is controlled by the core/TimerA/B module and is FOSC/4.

**REGISTER 17-1: CCPxCON: CCPx CONTROL REGISTER**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | DCxB1 | DCxB0 | CCPxM3 | CCPxM2 | CCPxM1 | CCPxM0 |
| 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 **Unimplemented:** Read as ‘0’

bit 5-4 **DCxB<1:0>**: PWM Duty Cycle Least Significant bits
Capture mode:
Unused
Compare mode:
Unused
PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0 **CCPxM<3:0>:** CCP Mode Select bits
`0000` = Capture/Compare/PWM off (resets CCP module)
`0001` = Unused (reserved)
`0010` = Compare mode, toggle output on match (CCPxIF bit of the PIRx register is set)
`0011` = Unused (reserved)
`0100` = Capture mode, every falling edge
`0101` = Capture mode, every rising edge
`0110` = Capture mode, every 4th rising edge
`0111` = Capture mode, every 16th rising edge
`1000` = Compare mode, set output on match (CCPxIF bit of the PIRx register is set)
`1001` = Compare mode, clear output on match (CCPxIF bit of the PIRx register is set)
`1010` = Compare mode, generate software interrupt on match (CCPxIF bit is set of the PIRx register, CCPx pin is unaffected)
`1011` = Compare mode, trigger special event (CCPxIF bit of the PIRx register is set, TMR1 is reset and A/D conversion(1) is started if the ADC module is enabled. CCPx pin is unaffected.)
`11xx` = PWM mode.

**Note 1:** A/D conversion start feature is available only on CCP2.

## 17.1 Capture Mode

In Capture mode, CCPRxH:CCPRxL captures the 16-bit value of the TMR1 register when an event occurs on pin CCPx. An event is defined as one of the following and is configured by the CCPxM<3:0> bits of the CCPxCON register:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

When a capture is made, the interrupt request flag bit CCPxIF of the PIRx register is set. The interrupt flag must be cleared in software. If another capture occurs before the value in the CCPRxH, CCPRxL register pair is read, the old captured value is overwritten by the new captured value (refer to Figure 17-1).

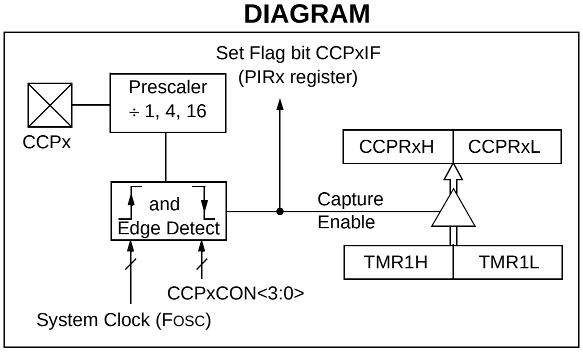
### 17.1.1 CCPx PIN CONFIGURATION

In Capture mode, the CCPx pin should be configured as an input by setting the associated TRIS control bit.

Either RC1 or RB3 can be selected as the CCP2 pin. Refer to **Section 6.1 “Alternate Pin Function”** for more information.

> **Note:** If the CCPx pin is configured as an output, a write to the port can cause a capture condition.

**FIGURE 17-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**




### 17.1.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode for the CCP module to use the capture feature. In Asynchronous Counter mode or when Timer1 is clocked at FOSC, the capture operation may not work.

### 17.1.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit of the PIEx register clear to avoid false interrupts. Additionally, the user should clear the CCPxIF interrupt flag bit of the PIRx register following any change in operating mode.

> **Note:** Clocking Timer1 from the system clock (FOSC) should not be used in Capture mode. In order for Capture mode to recognize the trigger event on the CCPx pin, Timer1 must be clocked from the instruction clock (FOSC/4) or from an external clock source.

### 17.1.4 CCP PRESCALER

There are four prescaler settings specified by the CCPxM<3:0> bits of the CCPxCON register. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. Any Reset will clear the prescaler counter.

Switching from one capture prescaler to another does not clear the prescaler and may generate a false interrupt. To avoid this unexpected operation, turn the module off by clearing the CCPxCON register before changing the prescaler (refer to Example 17-1).

**EXAMPLE 17-1: CHANGING BETWEEN CAPTURE PRESCALERS**

```
BANKSEL CCP1CON     ;Set Bank bits to point
                    ;to CCP1CON
CLRF    CCP1CON     ;Turn CCP module off
MOVLW   NEW_CAPT_PS ;Load the W reg with
                    ; the new prescaler
                    ; move value and CCP ON
MOVWF   CCP1CON     ;Load CCP1CON with this
                    ; value
```

### 17.1.5 CAPTURE DURING SLEEP

Capture mode depends upon the Timer1 module for proper operation. There are two options for driving the Timer1 module in Capture mode. It can be driven by the instruction clock (FOSC/4), or by an external clock source.

If Timer1 is clocked by FOSC/4, then Timer1 will not increment during Sleep. When the device wakes from Sleep, Timer1 will continue from its previous state.

If Timer1 is clocked by an external clock source, then Capture mode will operate as defined in **Section 17.1 “Capture Mode”**.

### TABLE 17-3: SUMMARY OF REGISTERS ASSOCIATED WITH CAPTURE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| ANSELB | ANSB7 | ANSB6 | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 1111 1111 | 1111 1111 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | — | — | ANSC2 | ANSC1 | ANSC0 | 111- -111 | 111- -111 |
| APFCON | — | — | — | — | — | — | SSSEL | CCP2SEL | ---- --00 | ---- --00 |
| CCP1CON | — | — | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | --00 0000 |
| CCPRxL | Capture/Compare/PWM Register X Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPRxH | Capture/Compare/PWM Register X High Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000x |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| PIE2 | TMR3GIE | TMR3IE | TMRBIE | TMRAIE | — | — | — | CCP2IE | 0000 ---0 | 0000 ---0 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIR2 | TMR3GIF | TMR3IF | TMRBIF | TMRAIF | — | — | — | CCP2IF | 0000 ---0 | 0000 ---0 |
| T1CON | TMR1CS1 | TMR1CS0 | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 0000 00-0 | uuuu uu-u |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/$\overline{\text{DONE}}$ | T1GVAL | T1GSS1 | T1GSS0 | 0000 0x00 | 0000 0x00 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 | 1111 1111 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 |

**Legend:** - = Unimplemented locations, read as '0', u = unchanged, x = unknown. Shaded cells are not used by the capture.

## 19.2 I²C Mode

The SSP module, in I²C mode, implements all slave functions, except general call support. It provides interrupts on Start and Stop bits in hardware to facilitate firmware implementations of the master functions. The SSP module implements the I²C Standard mode specifications:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- Start and Stop bit interrupts enabled to support firmware Master mode
- Address masking

Two pins are used for data transfer; the SCL pin (clock line) and the SDA pin (data line). The user must configure the two pin's data direction bits as inputs in the appropriate TRIS register. Upon enabling I²C mode, the I²C slew rate limiters in the I/O pads are controlled by the SMP bit of the SSPSTAT register. The SSP module functions are enabled by setting the SSPEN bit of the SSPCON register.

Data is sampled on the rising edge and shifted out on the falling edge of the clock. This ensures that the SDA signal is valid during the SCL high time. The SCL clock input must have minimum high and low times for proper operation. Refer to **Section 25.0 "Electrical Specifications"**.

**FIGURE 19-7: I²C MODE BLOCK DIAGRAM**

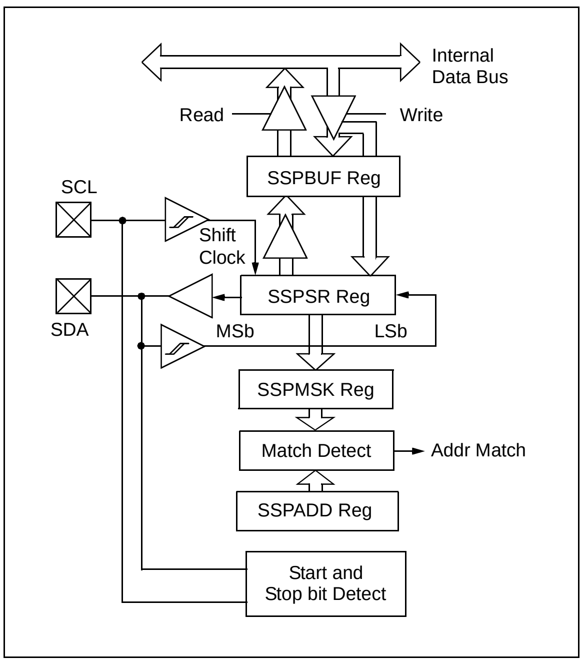
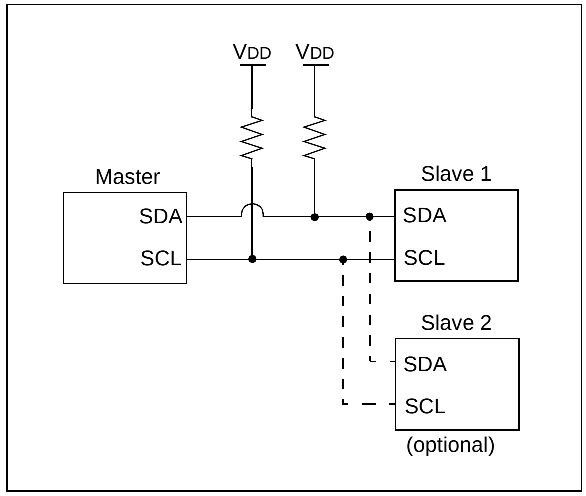



**FIGURE 19-8: TYPICAL I²C CONNECTIONS**




The SSP module has six registers for I²C operation. They are:

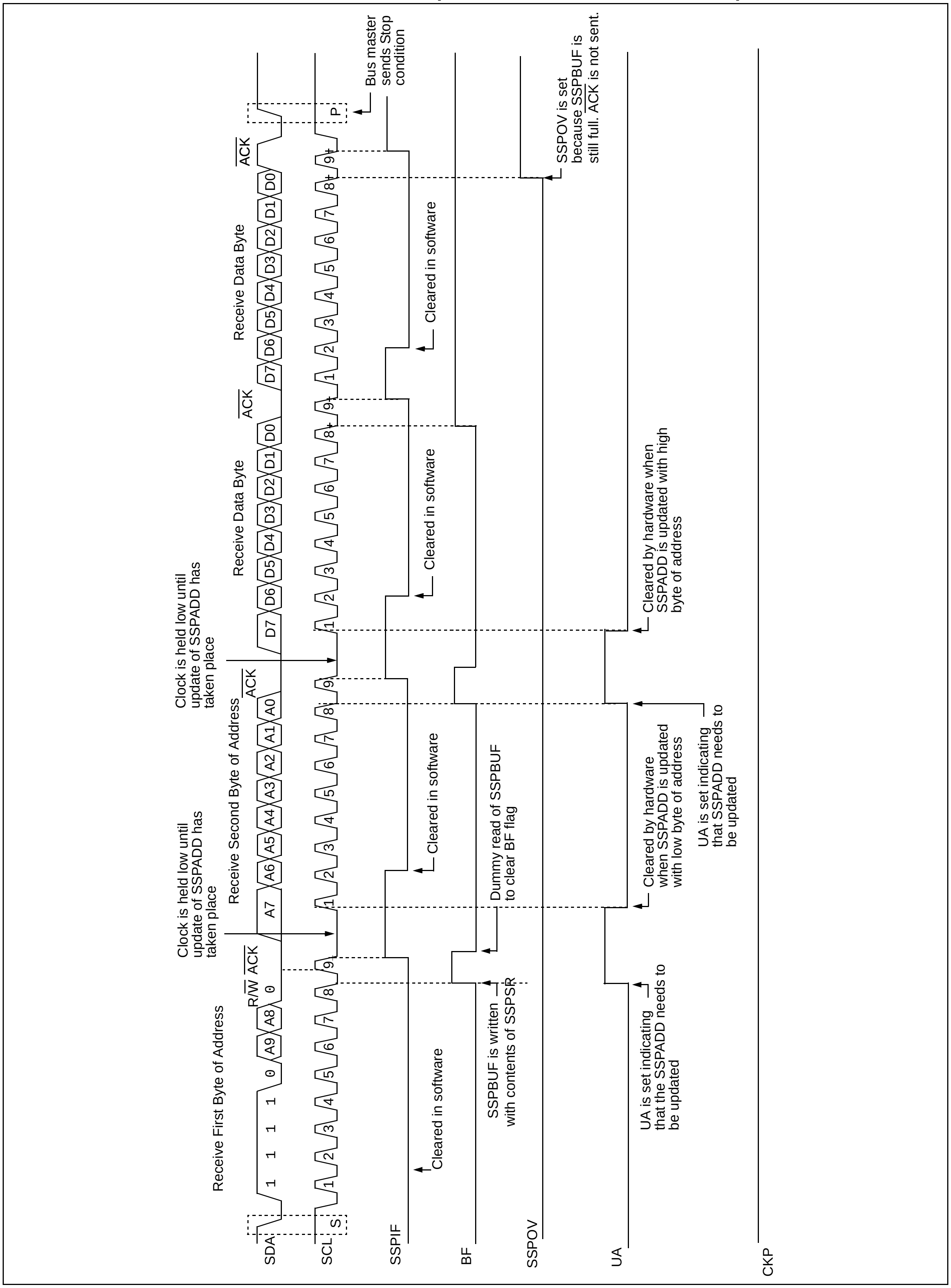
- SSP Control (SSPCON) register
- SSP Status (SSPSTAT) register
- Serial Receive/Transmit Buffer (SSPBUF) register
- SSP Shift Register (SSPSR), not directly accessible
- SSP Address (SSPADD) register
- SSP Address Mask (SSPMSK) register

### 19.2.1 HARDWARE SETUP

Selection of I²C mode, with the SSPEN bit of the SSPCON register set, forces the SCL and SDA pins to be open drain, provided these pins are programmed as inputs by setting the appropriate TRISC bits. The SSP module will override the input state with the output data, when required, such as for Acknowledge and slave-transmitter sequences.

| | |
|---|---|
| **Note:** | Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I²C module. |

**FIGURE 19-11: I²C SLAVE MODE TIMING (RECEPTION, 10-BIT ADDRESS)**



### 19.2.7 CLOCK STRETCHING

During any SCL low phase, any device on the $I^2C$ bus may hold the SCL line low and delay, or pause, the transmission of data. This "stretching" of a transmission allows devices to slow down communication on the bus. The SCL line must be constantly sampled by the master to ensure that all devices on the bus have released SCL for more data.

Stretching usually occurs after an $\overline{\text{ACK}}$ bit of a transmission, delaying the first bit of the next byte. The SSP module hardware automatically stretches for two conditions:

- After a 10-bit address byte is received (update SSPADD register)
- Anytime the CKP bit of the SSPCON register is cleared by hardware

The module will hold SCL low until the CKP bit is set. This allows the user slave software to update SSPBUF with data that may not be readily available. In 10-bit addressing modes, the SSPADD register must be updated after receiving the first and second address bytes. The SSP module will hold the SCL line low until the SSPADD has a byte written to it. The UA bit of the SSPSTAT register will be set, along with SSPIF, indicating an address update is needed.

### 19.2.8 FIRMWARE MASTER MODE

Master mode of operation is supported in firmware using interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits of the SSPSTAT register are cleared from a Reset or when the SSP module is disabled (SSPEN cleared). The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the $I^2C$ bus may be taken when the P bit is set or the bus is Idle and both the S and P bits are clear.

In Firmware Master mode, the SCL and SDA lines are manipulated by setting/clearing the corresponding TRIS bit(s). The output level is always low, irrespective of the value(s) in the corresponding PORT register bit(s). When transmitting a '1', the TRIS bit must be set (input) and a '0', the TRIS bit must be clear (output).

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP Interrupt will occur if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received

Firmware Master mode of operation can be done with either the Slave mode Idle (SSPM<3:0> = 1011), or with either of the Slave modes in which interrupts are enabled. When both master and slave functionality is enabled, the software needs to differentiate the source(s) of the interrupt.

Refer to Application Note AN554, "*Software Implementation of $I^2C$™ Bus Master*" (DS00554) for more information.

### 19.2.9 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allow the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the $I^2C$ bus may be taken when the P bit of the SSPSTAT register is set or when the bus is Idle, and both the S and P bits are clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In Multi-Master operation, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRIS bits). There are two stages where this arbitration of the bus can be lost. They are the Address Transfer and Data Transfer stages.

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{\text{ACK}}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to re-transfer the data at a later time.

Refer to Application Note AN578, "*Use of the SSP Module in the $I^2C$™ Multi-Master Environment*" (DS00578) for more information.

## 25.7 AC Characteristics: PIC16F707-I/E

**FIGURE 25-3: CLOCK TIMING**

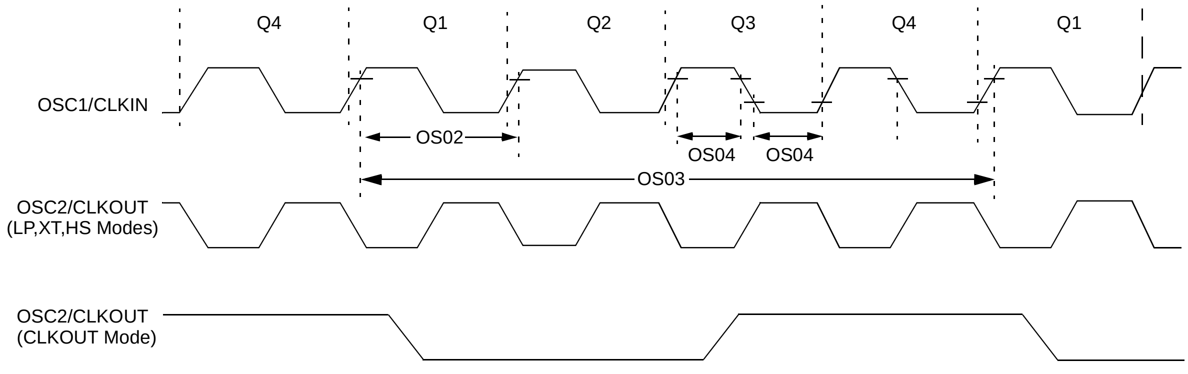
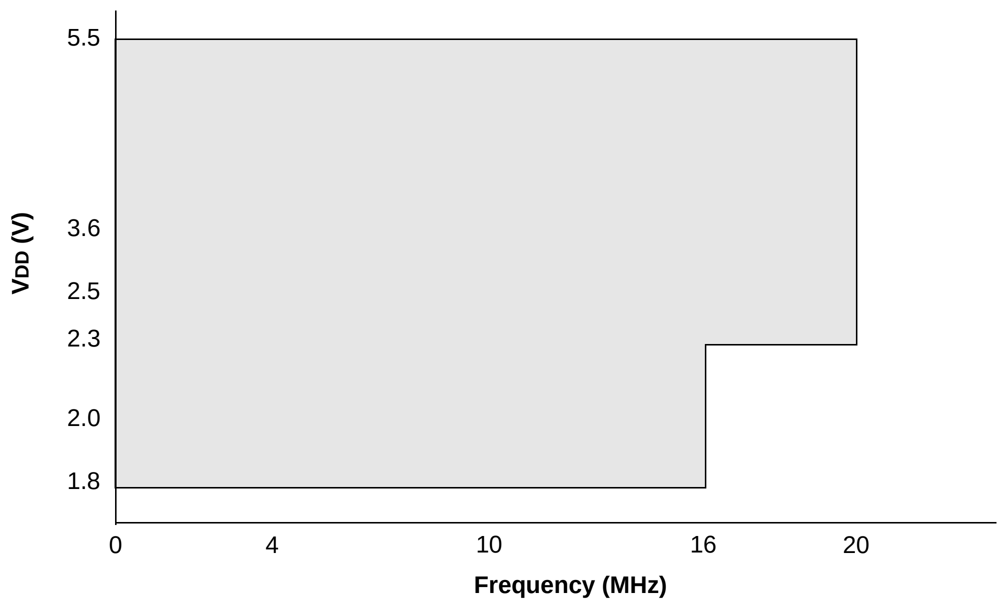



**FIGURE 25-4: PIC16F707 VOLTAGE FREQUENCY GRAPH, -40°C ≤ TA ≤ +125°C**




**Note 1:** The shaded region indicates the permissible combinations of voltage and frequency.
**2:** Refer to Table 25-1 for each Oscillator mode's supported frequencies.

**TABLE 25-14: CAP SENSE OSCILLATOR SPECIFICATIONS**

| Param. No. | Symbol | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| CS01 | ISRC | Current Source | High | — | -5.8 | -6 | μA | -40, -85°C |
| | | | Medium | — | -1.1 | -3.2 | μA | |
| | | | Low | — | -0.2 | -0.9 | μA | |
| CS02 | ISNK | Current Sink | High | — | 6.6 | 6 | μA | -40, -85°C |
| | | | Medium | — | 1.3 | 3.2 | μA | |
| | | | Low | — | 0.24 | 0.9 | μA | |
| CS03 | VCHYST | Cap Hysteresis | High | — | 525 | — | mV | VCTH-VCTL |
| | | | Medium | — | 375 | — | mV | |
| | | | Low | — | 280 | — | mV | |

* 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.

**FIGURE 25-22: CAP SENSE OSCILLATOR**

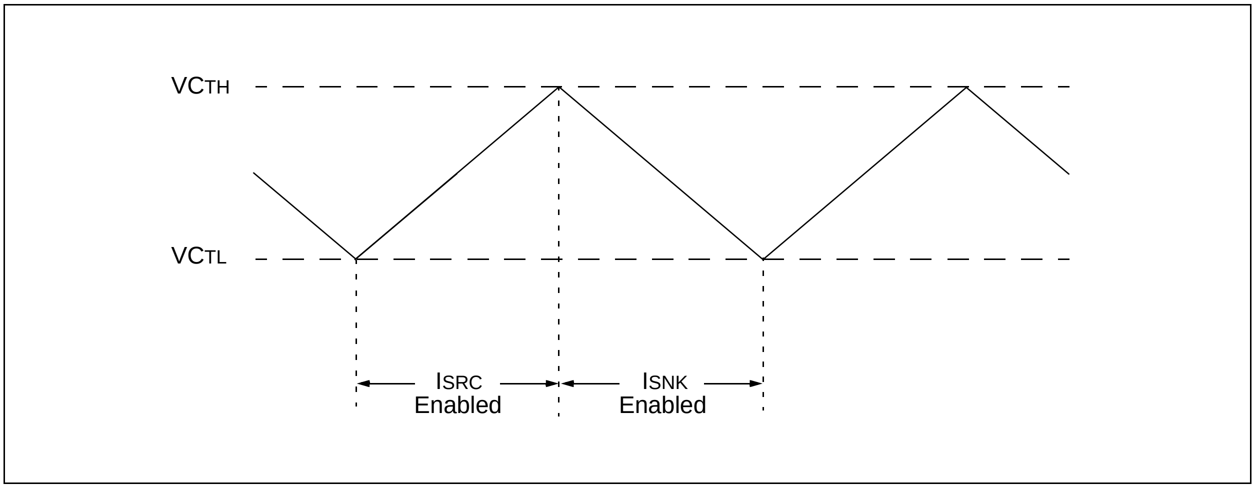


**FIGURE 26-14: PIC16LF707 MAXIMUM IDD vs. VDD OVER FOSC, XT MODE**

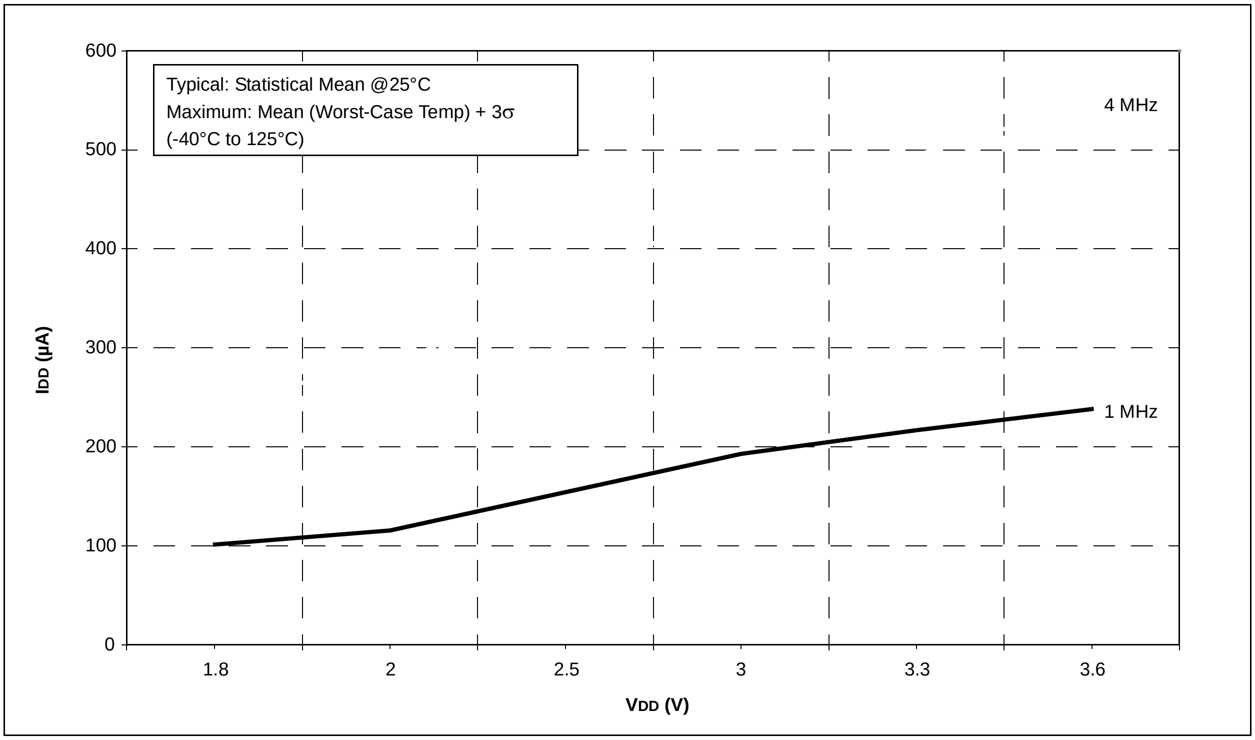



**FIGURE 26-15: PIC16F707 TYPICAL IDD vs. VDD OVER FOSC, XT MODE, VCAP = 0.1µF**

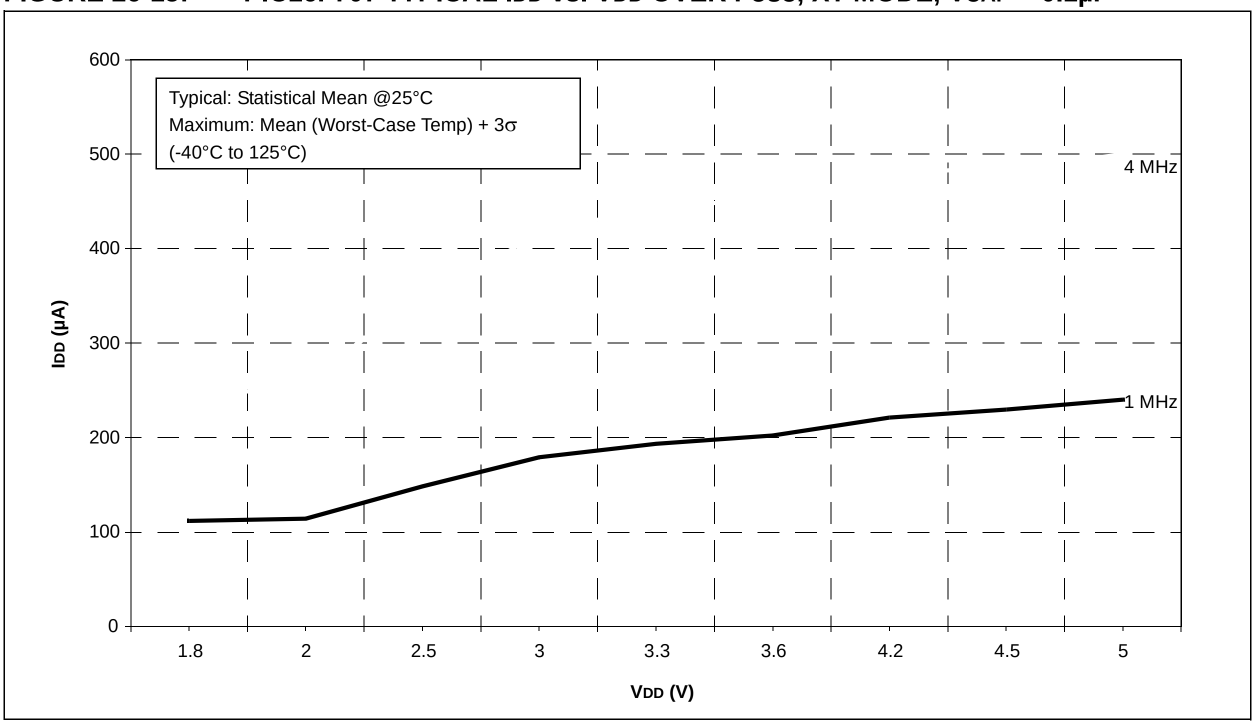


**FIGURE 26-44: PIC16LF707 TYPICAL ADC IPD vs. VDD**

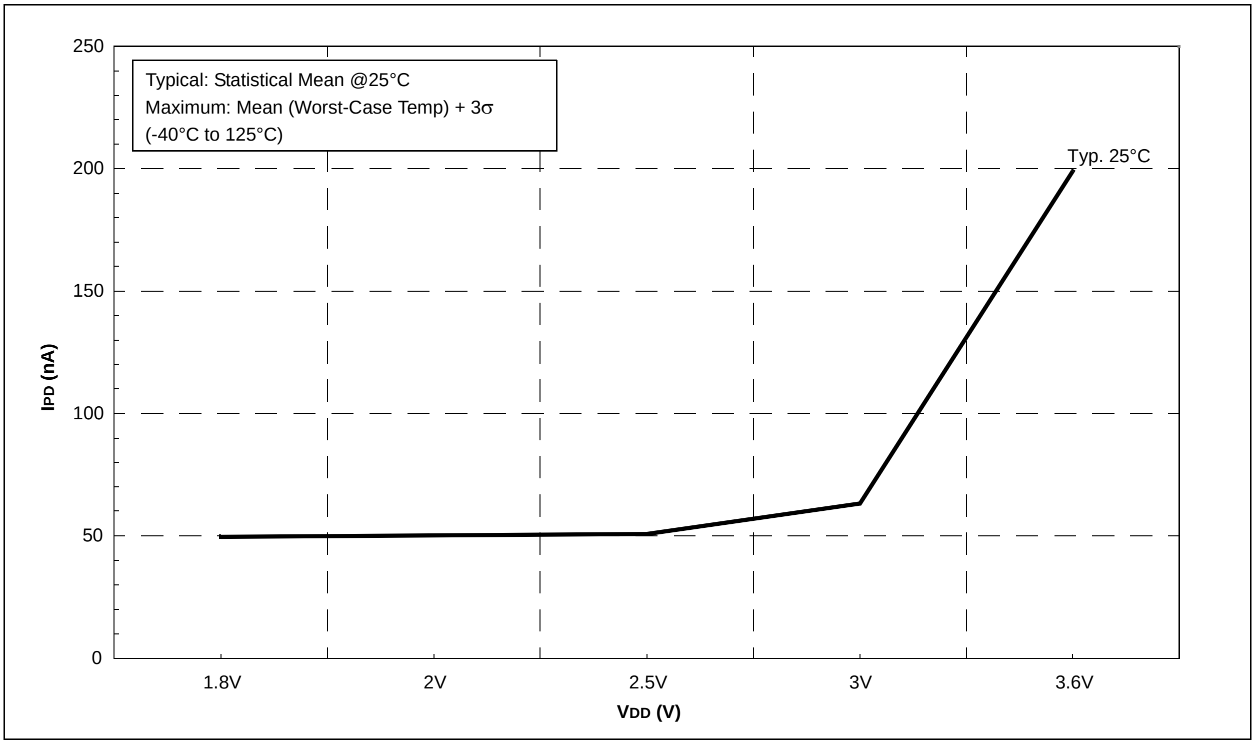



**FIGURE 26-45: PIC16F707 ADC IPD vs. VDD, VCAP = 0.1μF**

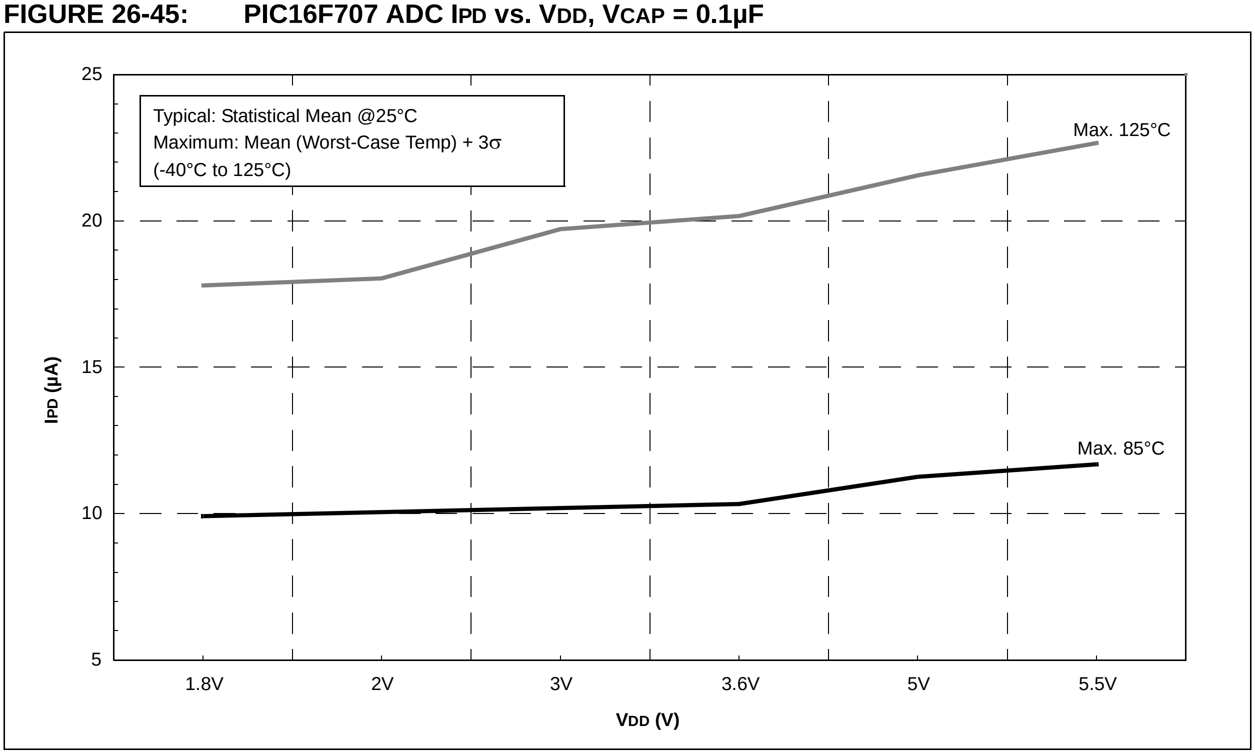


**FIGURE 26-46: PIC16LF707 ADC IPD vs. VDD**

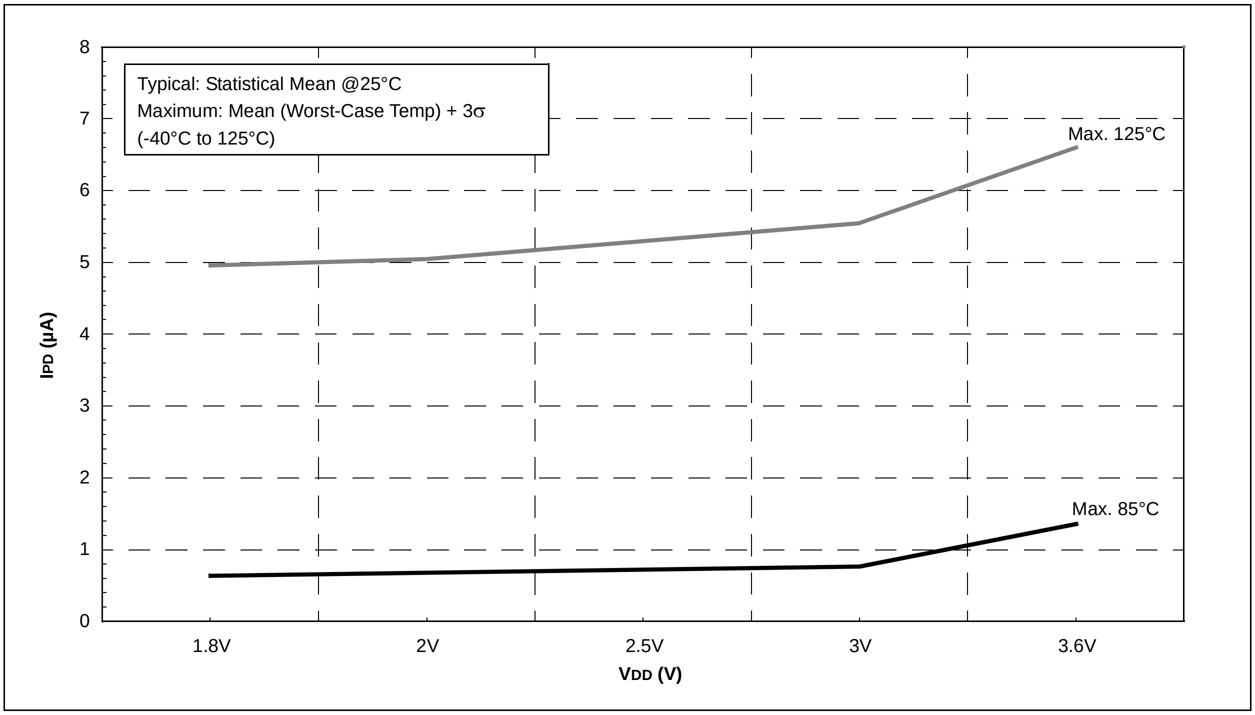



**FIGURE 26-47: PIC16F707 WDT IPD vs. VDD, VCAP = 0.1µF**