



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	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	25
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 11x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf722-i-ss

FIGURE 2-4: PIC16F722/LF722 SPECIAL FUNCTION REGISTERS

						File Address	
Indirect addr. ^(*)	00h	Indirect addr. ^(*)	80h	Indirect addr. ^(*)	100h	Indirect addr. ^(*)	180h
TMR0	01h	OPTION	81h	TMR0	101h	OPTION	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h	ANSELA	185h
PORTB	06h	TRISB	86h		106h	ANSELB	186h
PORTC	07h	TRISC	87h		107h		187h
	08h		88h	CPSCON0	108h		188h
PORTE	09h	TRISE	89h	CPSCON1	109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	PMDATL	10Ch	PMCON1	18Ch
PIR2	0Dh	PIE2	8Dh	PMADRL	10Dh	Reserved	18Dh
TMR1L	0Eh	PCON	8Eh	PMDATH	10Eh	Reserved	18Eh
TMR1H	0Fh	T1GCON	8Fh	PMADRH	10Fh	Reserved	18Fh
T1CON	10h	OSCCON	90h		110h		190h
TMR2	11h	OSCTUNE	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD/SSPMASK	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h	WPUB	95h		115h		195h
CCPR1H	16h	IOCB	96h		116h		196h
CCP1CON	17h		97h		117h		197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	APFCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	FVRCON	9Dh		11Dh		19Dh
ADRES	1Eh		9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
	20h	A0h	A0h		120h		1A0h
General Purpose Register 96 Bytes	7Fh	General Purpose Register 32 Bytes	BFh		16Fh		1EFh
	7Fh		C0h		170h		1F0h
	7Fh	Accesses 70h-7Fh	EFh	Accesses 70h-7Fh	17Fh	Accesses 70h-7Fh	1FFh
Bank 0	7Fh	Bank 1	FFh	Bank 2	17Fh	Bank 3	1FFh

Legend: = Unimplemented data memory locations, read as '0'.
* = Not a physical register.

2.5 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the File Select Register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no operation (although Status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit of the STATUS register, as shown in Figure 2-8.

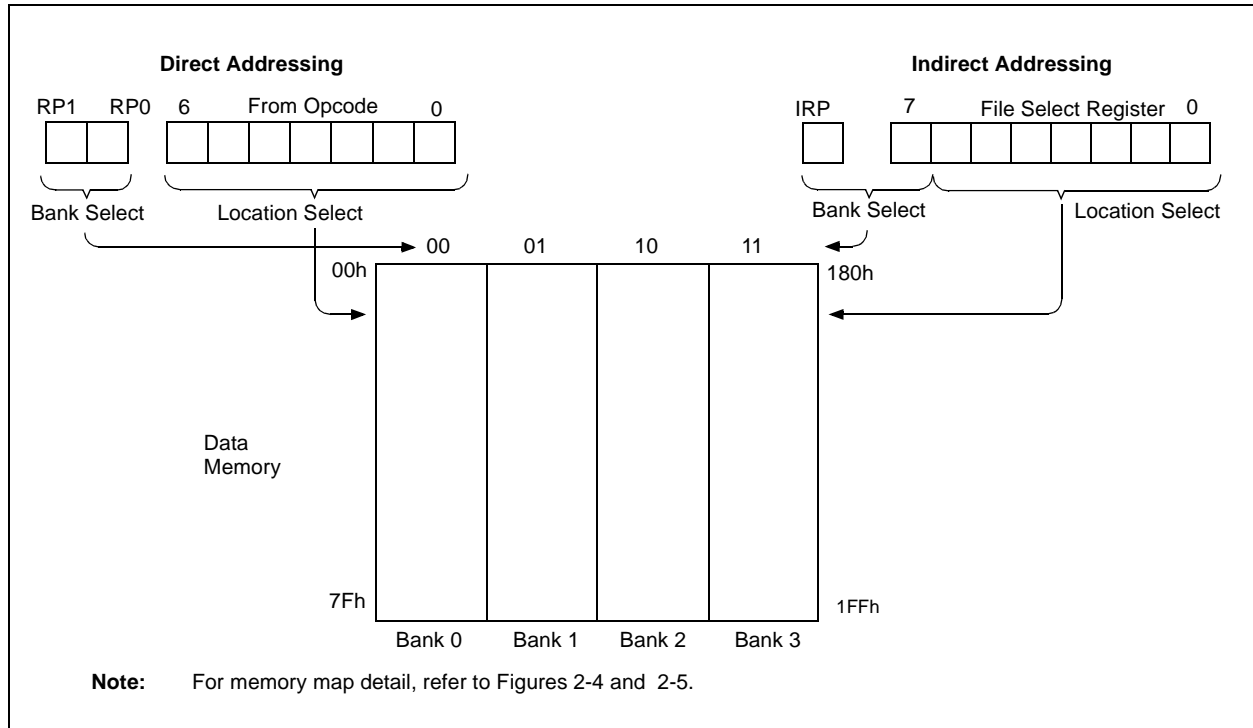
A simple program to clear RAM location 020h-02Fh using indirect addressing is shown in Example 2-2.

EXAMPLE 2-2: INDIRECT ADDRESSING

```

MOVLW    020h    ;initialize pointer
MOVWF    FSR     ;to RAM
BANKISEL 020h
NEXT CLRF INDF   ;clear INDF register
INCF     FSR     ;inc pointer
BTSS    FSR,4   ;all done?
GOTO    NEXT    ;no clear next
CONTINUE ;yes continue
    
```

FIGURE 2-8: DIRECT/INDIRECT ADDRESSING



PIC16(L)F722/3/4/6/7

6.0 I/O PORTS

There are as many as 35 general purpose I/O pins available. Depending on which peripherals are enabled, some or all of the pins may not be available as general purpose I/O. In general, when a peripheral is enabled, the associated pin may not be used as a general purpose I/O pin.

6.1 Alternate Pin Function

The Alternate Pin Function Control (APFCON) register is used to steer specific peripheral input and output functions between different pins. The APFCON register is shown in Register 6-1. For this device family, the following functions can be moved between different pins.

- \overline{SS} (Slave Select)
- CCP2

REGISTER 6-1: APFCON: ALTERNATE PIN FUNCTION CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SSSEL	CCP2SEL
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

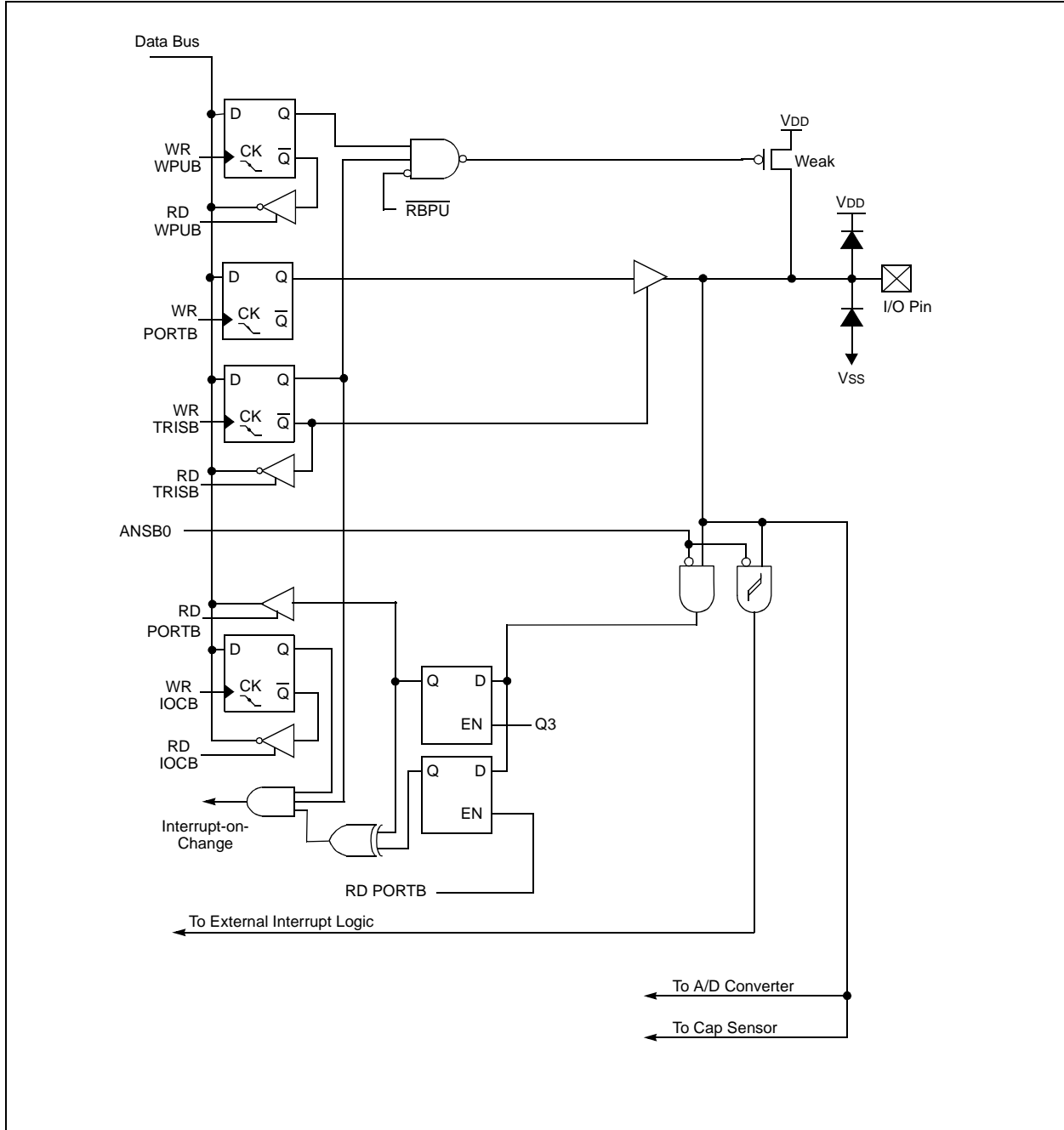
x = Bit is unknown

bit 7-2 **Unimplemented:** Read as '0'.

bit 1 **SSSEL:** \overline{SS} Input Pin Selection bit
0 = \overline{SS} function is on RA5/AN4/CPS7/ \overline{SS} /VCAP
1 = \overline{SS} function is on RA0/AN0/ \overline{SS} /VCAP

bit 0 **CCP2SEL:** CCP2 Input/Output Pin Selection bit
0 = CCP2 function is on RC1/T1OSI/CCP2
1 = CCP2 function is on RB3/CCP2

FIGURE 6-7: BLOCK DIAGRAM OF RB0



PIC16(L)F722/3/4/6/7

FIGURE 6-22: BLOCK DIAGRAM OF RE<2:0>

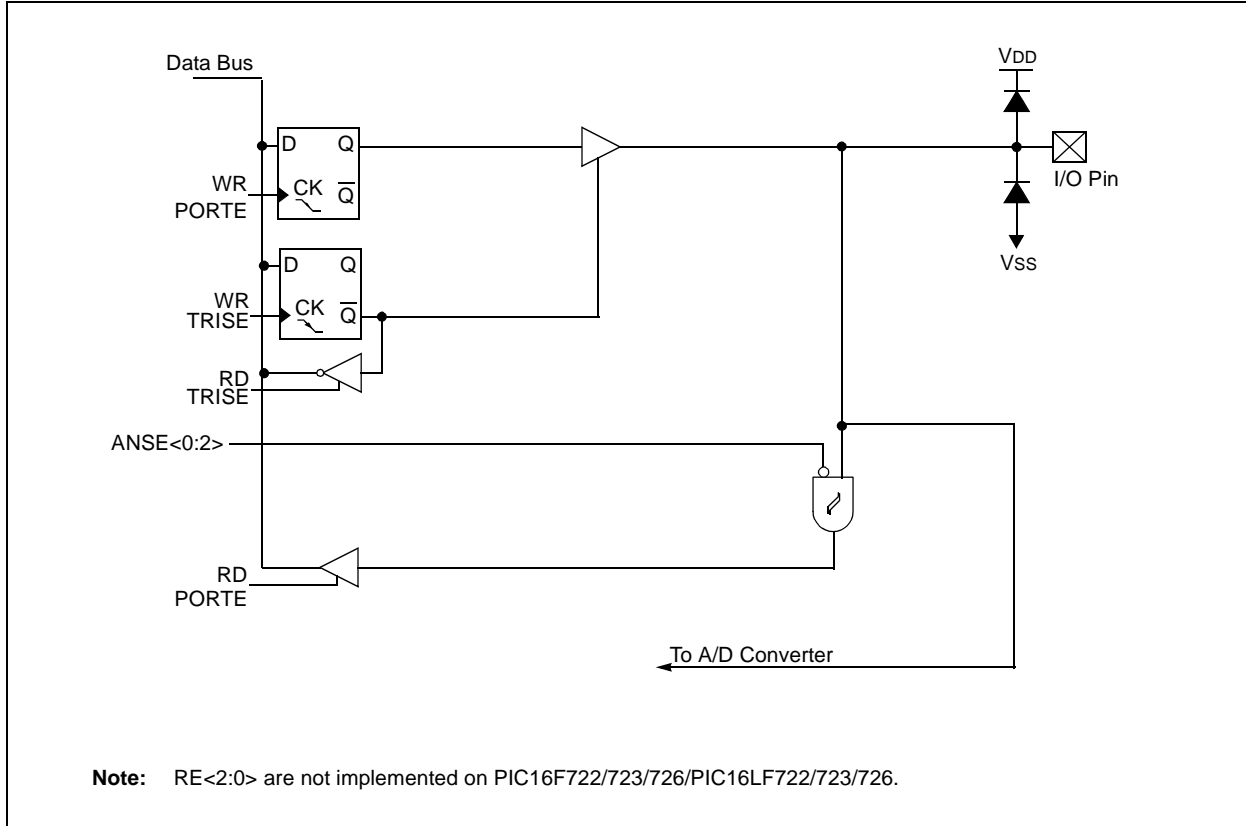
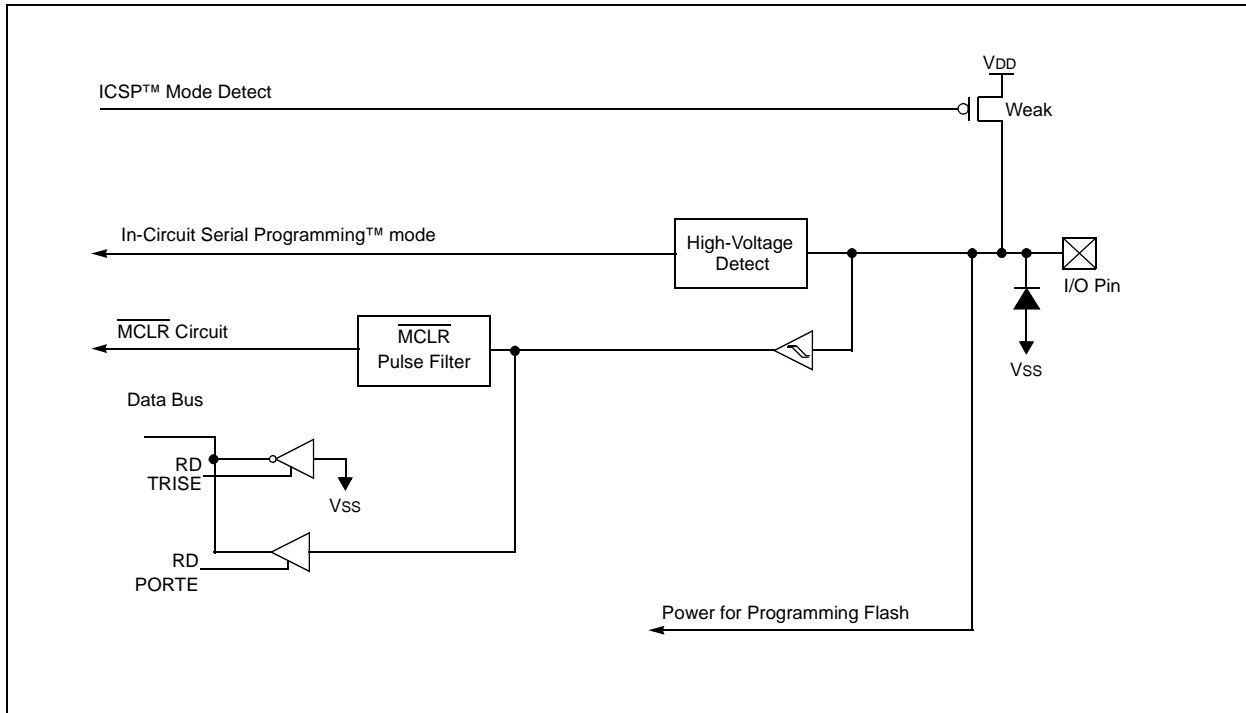


FIGURE 6-23: BLOCK DIAGRAM OF RE3



14.6 Operation during Sleep

The capacitive sensing oscillator will continue to run as long as the module is enabled, independent of the part being in Sleep. In order for the software to determine if a frequency change has occurred, the part must be awake. However, the part does not have to be awake when the timer resource is acquiring counts. One way to acquire the Timer1 counts while in Sleep is to have Timer1 gated with the overflow of the Watchdog Timer. This can be accomplished using the following steps:

1. Configure the Watchdog Time-out overflow as the Timer1's gate source $T1GSS<1:0> = 11$.
2. Set Timer1 Gate to toggle mode by setting the T1GTM bit of the T1GCON register.
3. Set the TMR1GE bit of the T1GCON register.
4. Set TMR1ON bit of the T1CON register.
5. Enable capacitive sensing module with the appropriate current settings and pin selection.
6. Clear Timer1.
7. Put the part to Sleep.
8. On the first WDT overflow, the capacitive sensing oscillator will begin to increment Timer1. Then put the part to Sleep.
9. On the second WDT overflow Timer1 will stop incrementing. Then run the software routine to determine if a frequency change has occurred.

Refer to **Section 12.0 "Timer1 Module with Gate Control"** for additional information.

Note 1: When using the WDT to set the interval on Timer1, any other source that wakes the part up early will cause the WDT overflow to be delayed, affecting the value captured by Timer1.

2: Timer0 does not operate when in Sleep, and therefore cannot be used for capacitive sense measurements in Sleep.

PIC16(L)F722/3/4/6/7

15.0 CAPTURE/COMPARE/PWM (CCP) MODULE

The Capture/Compare/PWM module is a peripheral which allows the user to time and control different events. In Capture mode, the peripheral allows the timing of the duration of an event. The Compare mode allows the user to trigger an external event when a predetermined amount of time has expired. The PWM mode can generate a pulse-width modulated signal of varying frequency and duty cycle.

The timer resources used by the module are shown in Table 15-1.

Additional information on CCP modules is available in the Application Note AN594, *Using the CCP Modules* (DS00594).

TABLE 15-1: CCP MODE – TIMER RESOURCES REQUIRED

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

TABLE 15-2: INTERACTION OF TWO CCP MODULES

CCP1 Mode	CCP2 Mode	Interaction
Capture	Capture	Same TMR1 time base
Capture	Compare	Same TMR1 time base ^(1, 2)
Compare	Compare	Same TMR1 time base ^(1, 2)
PWM	PWM	The PWMs will have the same frequency and update rate (TMR2 interrupt). The rising edges will be aligned.
PWM	Capture	None
PWM	Compare	None

- Note 1:** If CCP2 is configured as a Special Event Trigger, CCP1 will clear Timer1, affecting the value captured on the CCP2 pin.
- 2:** If CCP1 is in Capture mode and CCP2 is configured as a Special Event Trigger, CCP2 will clear Timer1, affecting the value captured on the CCP1 pin.

Note: CCPRx and CCPx throughout this document refer to CCPR1 or CCPR2 and CCP1 or CCP2, respectively

PIC16(L)F722/3/4/6/7

15.1 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 15-1).

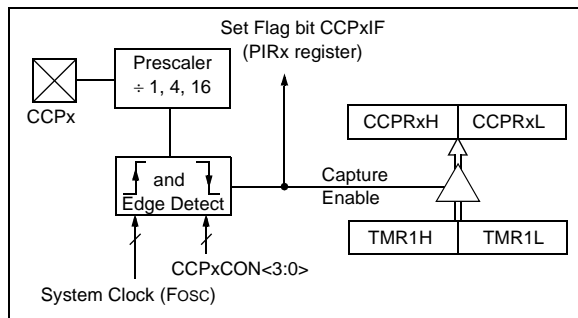
15.1.1 CCPx PIN CONFIGURATION

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 15-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



15.1.2 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.

15.1.3 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.

15.1.4 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 15-1).

EXAMPLE 15-1: 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
MOVWF  CCP1CON    ; move value and CCP ON
```

15.1.5 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 15.1 “Capture Mode”**.

PIC16(L)F722/3/4/6/7

TABLE 15-3: 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	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	--11 1111	--11 1111
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	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000x
PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
PIE2	—	—	—	—	—	—	—	CCP2IE	---- ---0	---- ---0
PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIR2	—	—	—	—	—	—	—	CCP2IF	---- ---0	---- ---0
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	T1SYN \bar{C}	—	TMR1ON	0000 00-0	uuuu uu-u
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ 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.

PIC16(L)F722/3/4/6/7

16.4 AUSART Operation During Sleep

The AUSART will remain active during Sleep only in the Synchronous Slave mode. All other modes require the system clock and therefore cannot generate the necessary signals to run the Transmit or Receive Shift registers during Sleep.

Synchronous Slave mode uses an externally generated clock to run the Transmit and Receive Shift registers.

16.4.1 SYNCHRONOUS RECEIVE DURING SLEEP

To receive during Sleep, all the following conditions must be met before entering Sleep mode:

- RCSTA and TXSTA Control registers must be configured for Synchronous Slave Reception (refer to **Section 16.3.2.4 “Synchronous Slave Reception Setup:”**).
- If interrupts are desired, set the RCIE bit of the PIE1 register and the PEIE bit of the INTCON register.
- The RCIF interrupt flag must be cleared by reading RCREG to unload any pending characters in the receive buffer.

Upon entering Sleep mode, the device will be ready to accept data and clocks on the RX/DT and TX/CK pins, respectively. When the data word has been completely clocked in by the external device, the RCIF interrupt flag bit of the PIR1 register will be set. Thereby, waking the processor from Sleep.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the GIE global interrupt enable bit of the INTCON register is also set, then the Interrupt Service Routine at address 0004h will be called.

16.4.2 SYNCHRONOUS TRANSMIT DURING SLEEP

To transmit during Sleep, all the following conditions must be met before entering Sleep mode:

- RCSTA and TXSTA Control registers must be configured for Synchronous Slave Transmission (refer to **Section 16.3.2.2 “Synchronous Slave Transmission Setup:”**).
- The TXIF interrupt flag must be cleared by writing the output data to the TXREG, thereby filling the TSR and transmit buffer.
- If interrupts are desired, set the TXIE bit of the PIE1 register and the PEIE bit of the INTCON register.

Upon entering Sleep mode, the device will be ready to accept clocks on TX/CK pin and transmit data on the RX/DT pin. When the data word in the TSR has been completely clocked out by the external device, the pending byte in the TXREG will transfer to the TSR and the TXIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TXREG is available to accept another character for transmission, which will clear the TXIF flag.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the GIE global interrupt enable bit is also set then the Interrupt Service Routine at address 0004h will be called.

17.0 SSP MODULE OVERVIEW

The Synchronous Serial Port (SSP) module is a serial interface useful for communicating with other peripherals or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I²C)

17.1 SPI Mode

The SPI mode allows eight bits of data to be synchronously transmitted and received, simultaneously. The SSP module can be operated in one of two SPI modes:

- Master mode
- Slave mode

SPI is a full-duplex protocol, with all communication being bidirectional and initiated by a master device. All clocking is provided by the master device and all bits are transmitted, MSb first. Care must be taken to ensure that all devices on the SPI bus are setup to allow all controllers to send and receive data at the same time.

A typical SPI connection between microcontroller devices is shown in Figure 17-1. Addressing of more than one slave device is accomplished via multiple hardware slave select lines. External hardware and additional I/O pins must be used to support multiple slave select addressing. This prevents extra overhead in software for communication.

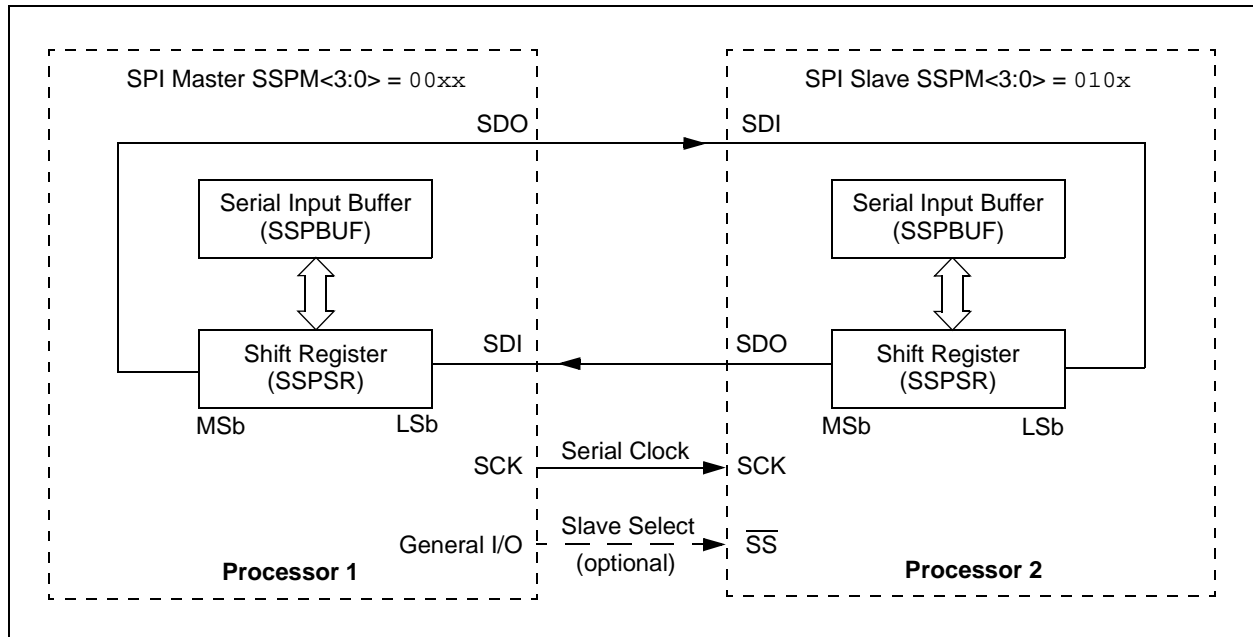
For SPI communication, typically three pins are used:

- Serial Data Out (SDO)
- Serial Data In (SDI)
- Serial Clock (SCK)

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select (\overline{SS})

FIGURE 17-1: TYPICAL SPI MASTER/SLAVE CONNECTION



17.1.1 MASTER MODE

In Master mode, data transfer can be initiated at any time because the master controls the SCK line. Master mode determines when the slave (Figure 17-1, Processor 2) transmits data via control of the SCK line.

17.1.1.1 Master Mode Operation

The SSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPBUF). The SSPSR register shifts the data in and out of the device, MSb first. The SSPBUF register holds the data that is written out of the master until the received data is ready. Once the eight bits of data have been received, the byte is moved to the SSPBUF register. The Buffer Full Status bit, BF of the SSPSTAT register, and the SSP Interrupt Flag bit, SSPIF of the PIR1 register, are then set.

Any write to the SSPBUF register during transmission/reception of data will be ignored and the Write Collision Detect bit, WCOL of the SSPCON register, will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data is written to the SSPBUF. The BF bit of the SSPSTAT register is set when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. The SSP interrupt may be used to determine when the transmission/reception is complete and the SSPBUF must be read and/or written. If interrupts are not used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

Note: The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register.

17.1.1.2 Enabling Master I/O

To enable the serial port, the SSPEN bit of the SSPCON register, must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPCON register and then set the SSPEN bit. If a Master mode of operation is selected in the SSPM bits of the SSPCON register, the SDI, SDO and SCK pins will be assigned as serial port pins.

For these pins to function as serial port pins, they must have their corresponding data direction bits set or cleared in the associated TRIS register as follows:

- SDI configured as input
- SDO configured as output
- SCK configured as output

17.1.1.3 Master Mode Setup

In Master mode, the data is transmitted/received as soon as the SSPBUF register is loaded with a byte value. If the master is only going to receive, SDO output could be disabled (programmed and used as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate.

When initializing SPI Master mode operation, several options need to be specified. This is accomplished by programming the appropriate control bits in the SSPCON and SSPSTAT registers. These control bits allow the following to be specified:

- SCK as clock output
- Idle state of SCK (CKP bit)
- Data input sample phase (SMP bit)
- Output data on rising/falling edge of SCK (CKE bit)
- Clock bit rate

In Master mode, the SPI clock rate (bit rate) is user selectable to be one of the following:

- $F_{osc}/4$ (or TCY)
- $F_{osc}/16$ (or $4 \cdot TCY$)
- $F_{osc}/64$ (or $16 \cdot TCY$)
- (Timer2 output)/2

This allows a maximum data rate of 5 Mbps (at $F_{osc} = 20$ MHz).

Figure 17-3 shows the waveforms for Master mode. The clock polarity is selected by appropriately programming the CKP bit of the SSPCON register. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The sample time of the input data is shown based on the state of the SMP bit and can occur at the middle or end of the data output time. The time when the SSPBUF is loaded with the received data is shown.

17.1.1.4 Sleep in Master Mode

In Master mode, all module clocks are halted and the transmission/reception will remain in their current state, paused, until the device wakes from Sleep. After the device wakes up from Sleep, the module will continue to transmit/receive data.

PIC16(L)F722/3/4/6/7

BTFS Bit Test f, Skip if Set

Syntax: [*label*] BTFS f,b
Operands: $0 \leq f \leq 127$
 $0 \leq b < 7$
Operation: skip if (f) = 1
Status Affected: None
Description: If bit 'b' in register 'f' is '0', the next instruction is executed.
If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

CLRWD Clear Watchdog Timer

Syntax: [*label*] CLRWD
Operands: None
Operation: 00h → WDT
0 → WDT prescaler,
1 → \overline{TO}
1 → \overline{PD}
Status Affected: \overline{TO} , \overline{PD}
Description: CLRWD instruction resets the Watchdog Timer. It also resets the prescaler of the WDT.
Status bits \overline{TO} and \overline{PD} are set.

CALL Call Subroutine

Syntax: [*label*] CALL k
Operands: $0 \leq k \leq 2047$
Operation: (PC)+ 1 → TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>
Status Affected: None
Description: Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

COMF Complement f

Syntax: [*label*] COMF f,d
Operands: $0 \leq f \leq 127$
d ∈ [0,1]
Operation: (\bar{f}) → (destination)
Status Affected: Z
Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

CLRF Clear f

Syntax: [*label*] CLRF f
Operands: $0 \leq f \leq 127$
Operation: 00h → (f)
1 → Z
Status Affected: Z
Description: The contents of register 'f' are cleared and the Z bit is set.

DECF Decrement f

Syntax: [*label*] DECF f,d
Operands: $0 \leq f \leq 127$
d ∈ [0,1]
Operation: (f) - 1 → (destination)
Status Affected: Z
Description: Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

CLRW Clear W

Syntax: [*label*] CLRW
Operands: None
Operation: 00h → (W)
1 → Z
Status Affected: Z
Description: W register is cleared. Zero bit (Z) is set.

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) - (W) \rightarrow (\text{destination})$

Status Affected: C, DC, Z

Description: Subtract (2's complement method) W register from register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

C = 0	$W > f$
C = 1	$W \leq f$
DC = 0	$W\langle 3:0 \rangle > f\langle 3:0 \rangle$
DC = 1	$W\langle 3:0 \rangle \leq f\langle 3:0 \rangle$

SWAPF **Swap Nibbles in f**

Syntax: [*label*] SWAPF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f\langle 3:0 \rangle) \rightarrow (\text{destination}\langle 7:4 \rangle)$,
 $(f\langle 7:4 \rangle) \rightarrow (\text{destination}\langle 3:0 \rangle)$

Status Affected: None

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed in register 'f'.

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) .XOR. k \rightarrow (W)$

Status Affected: Z

Description: The contents of the W register are XOR'ed with the 8-bit literal 'k'. The result is placed in the W register.

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(W) .XOR. (f) \rightarrow (\text{destination})$

Status Affected: Z

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

FIGURE 24-7: PIC16F722/3/4/6/7 TYPICAL I_{DD} vs. V_{DD} OVER F_{osc} , EXTRC MODE, $V_{CAP} = 0.1 \mu F$

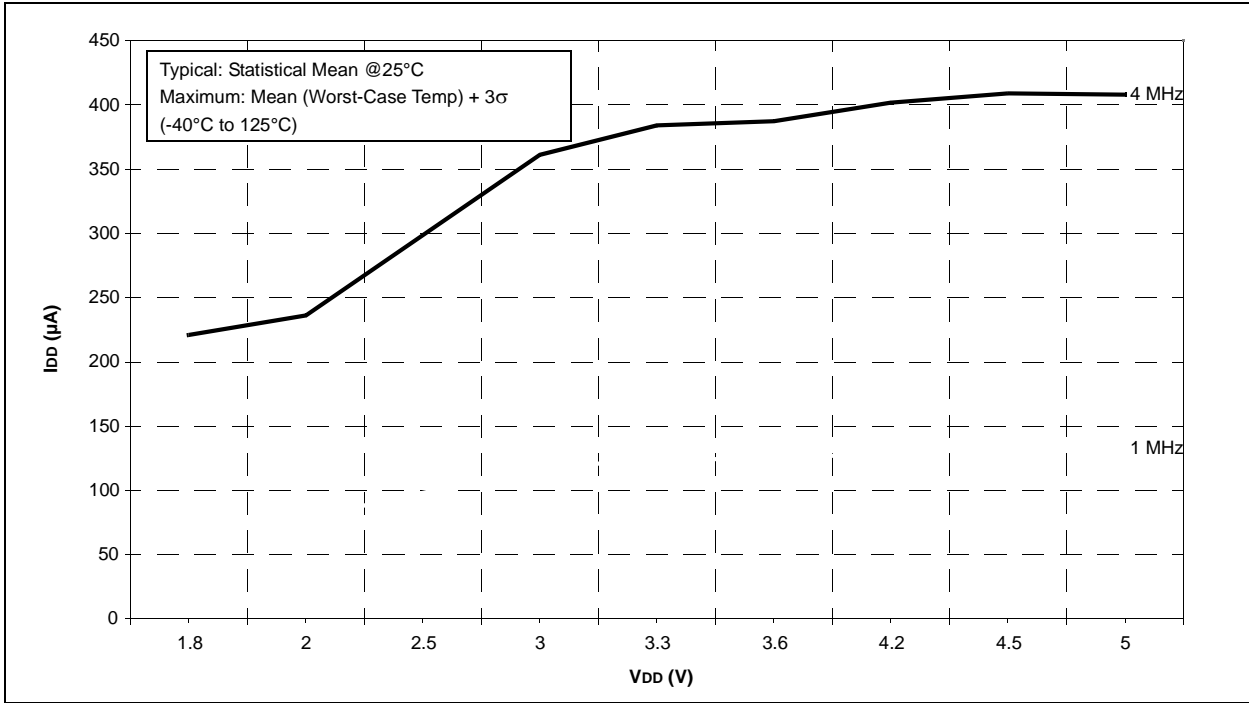
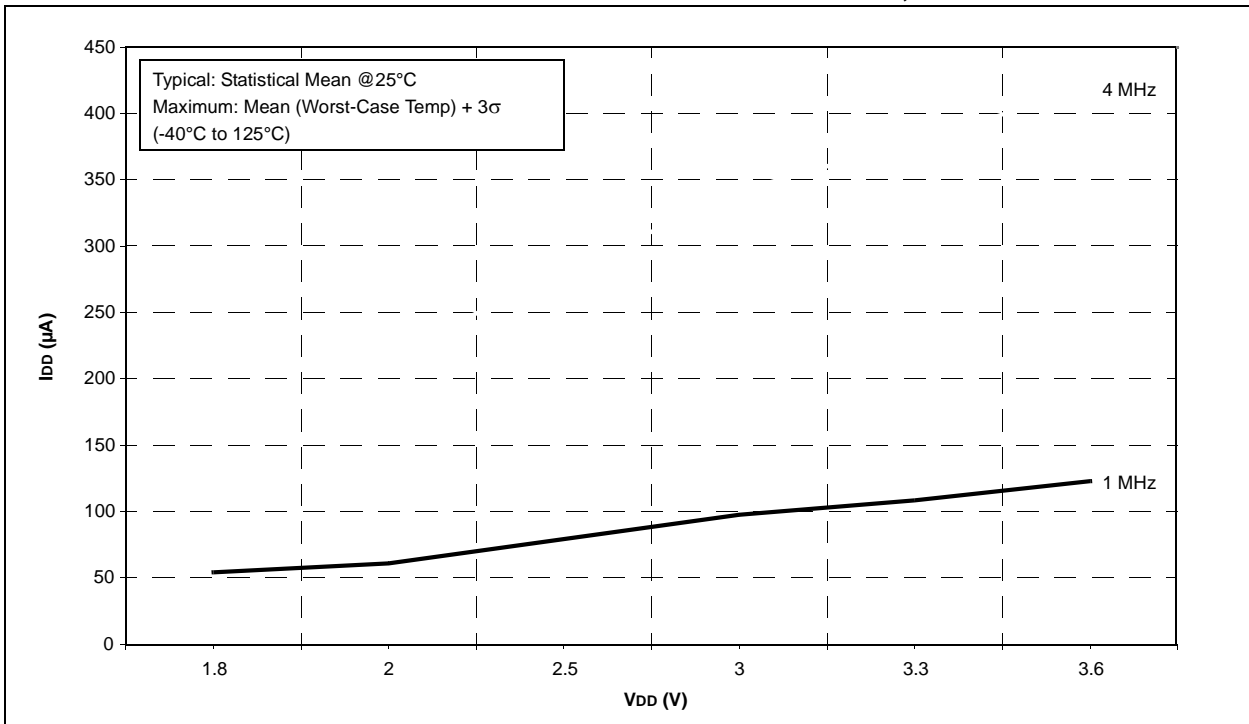


FIGURE 24-8: PIC16LF722/3/4/6/7 TYPICAL I_{DD} vs. V_{DD} OVER F_{osc} , EXTRC MODE



PIC16(L)F722/3/4/6/7

FIGURE 24-19: PIC16F722/3/4/6/7 MAXIMUM I_{DD} vs. F_{OSC} OVER V_{DD}, INTOSC MODE, V_{CAP} = 0.1 μF

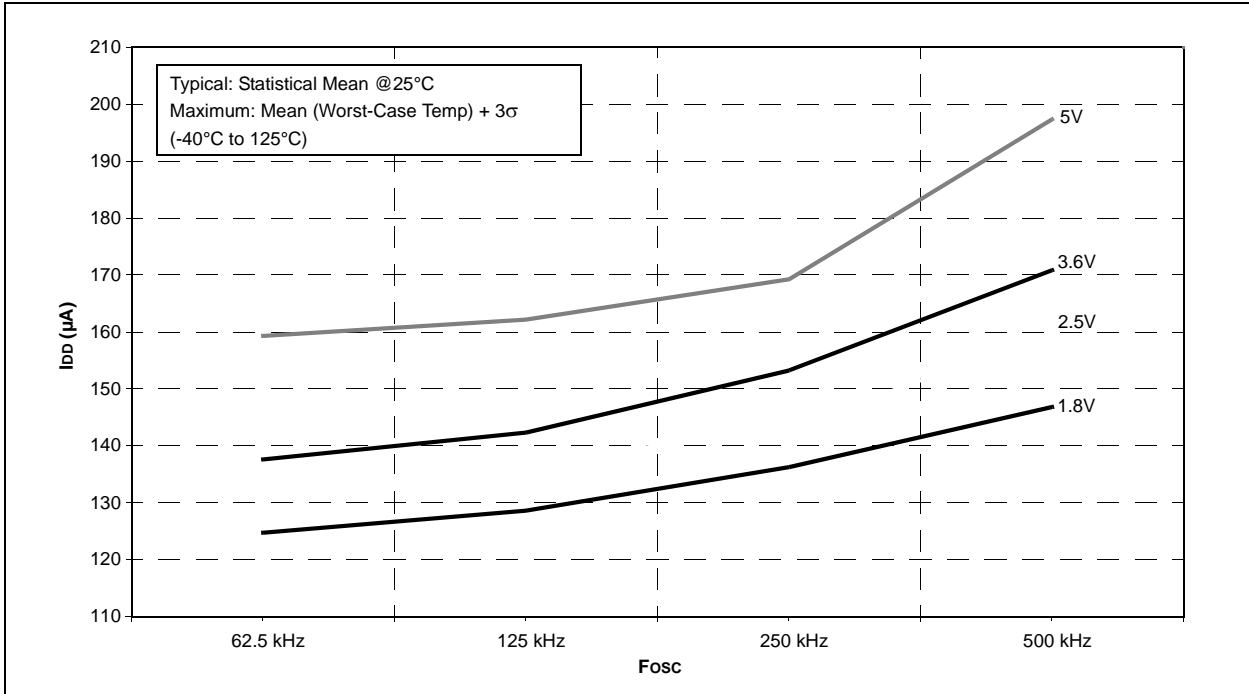
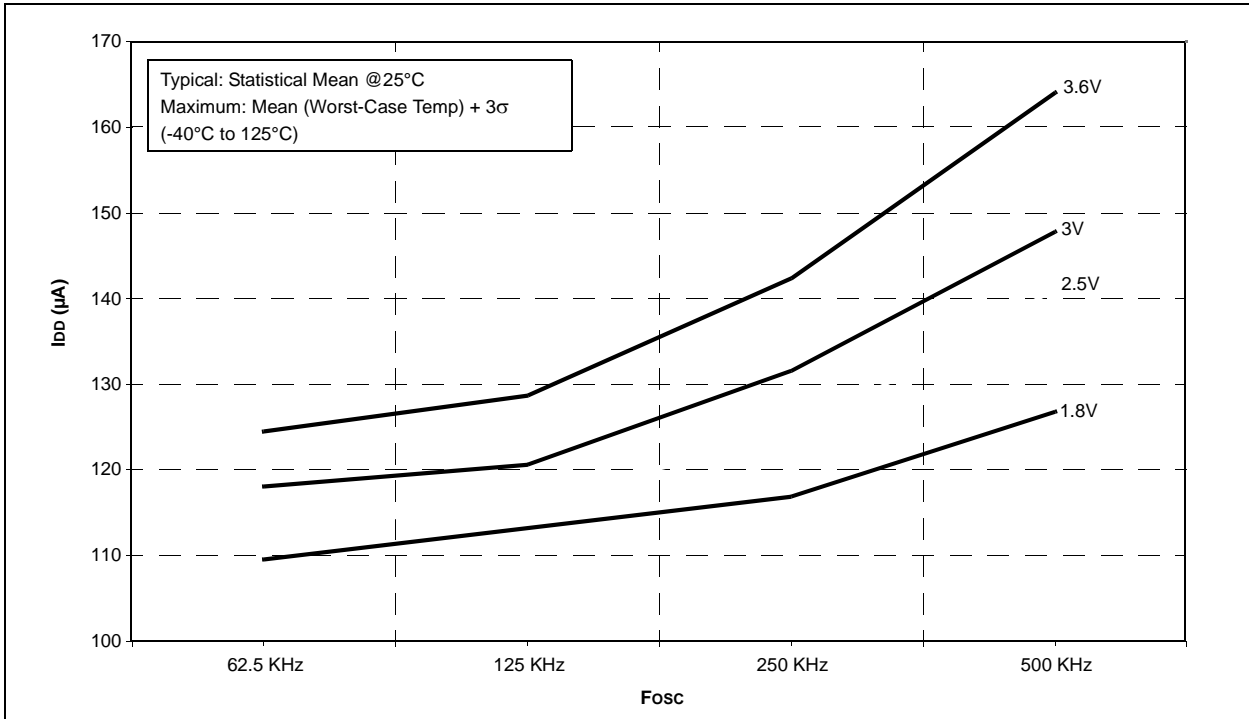


FIGURE 24-20: PIC16LF722/3/4/6/7 MAXIMUM I_{DD} vs. F_{OSC} OVER V_{DD}, INTOSC MODE



PIC16(L)F722/3/4/6/7

FIGURE 24-25: PIC16F722/3/4/6/7 TYPICAL I_{DD} vs. F_{OSC} OVER V_{DD} , INTOSC MODE, $V_{CAP} = 0.1 \mu F$

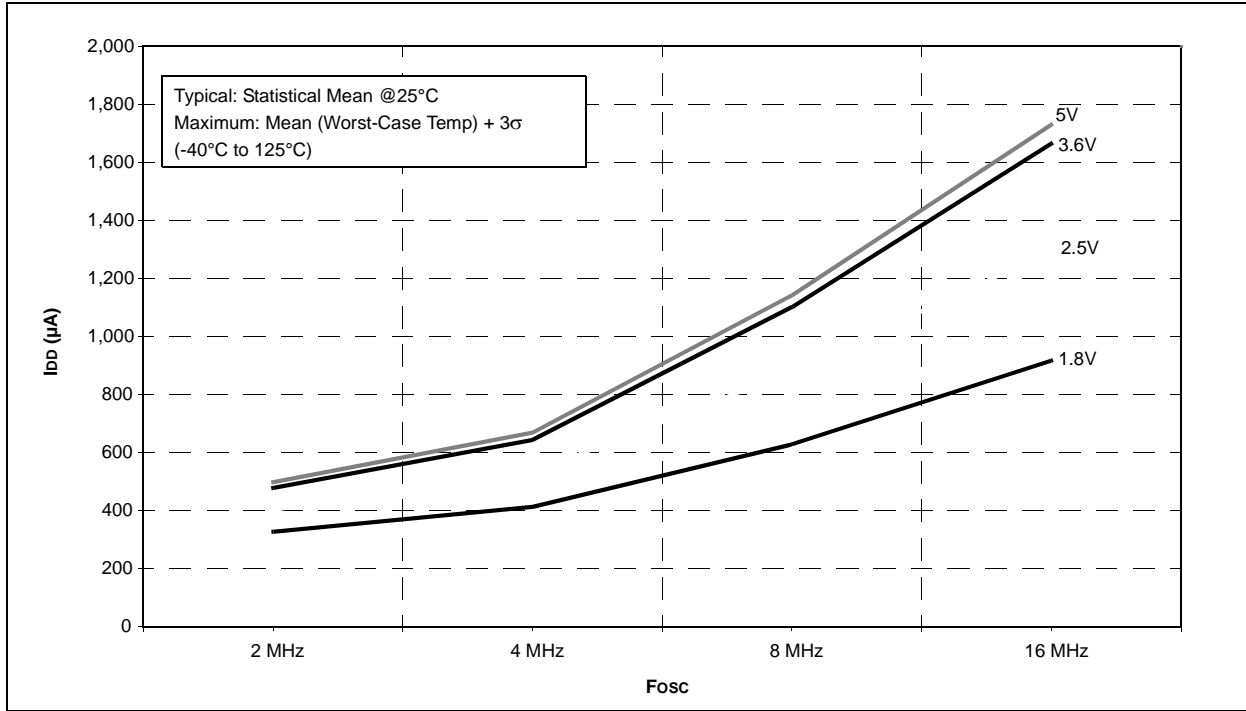
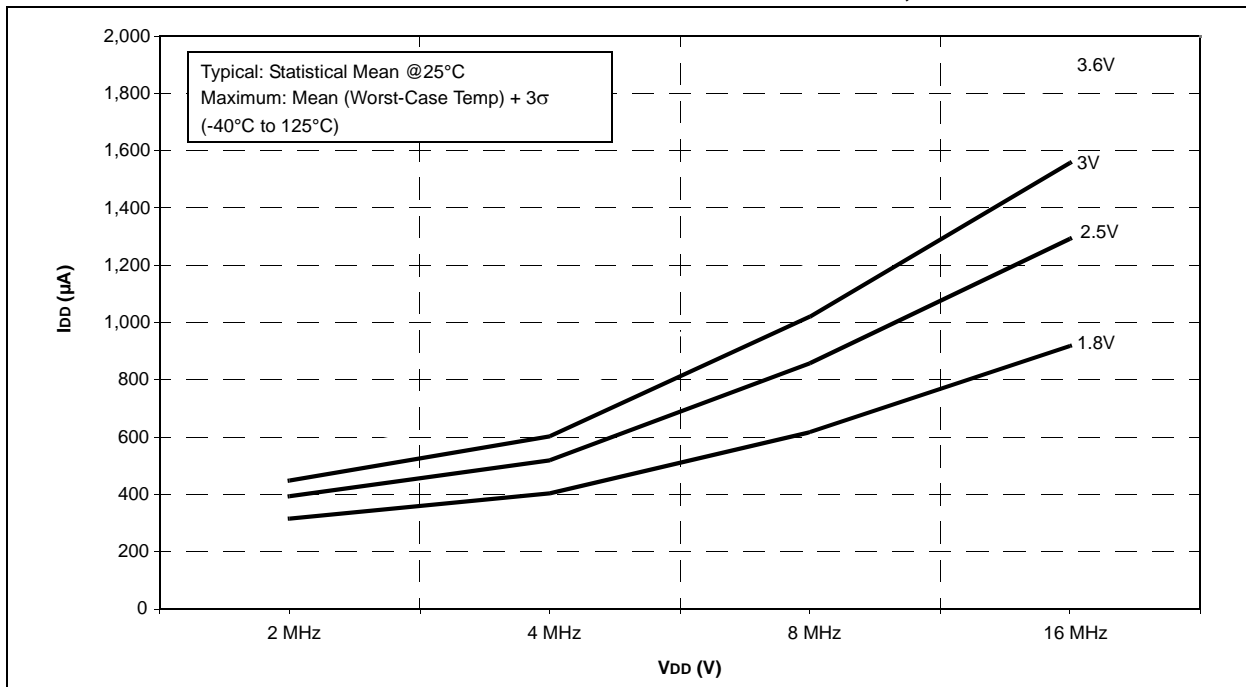


FIGURE 24-26: PIC16LF722/3/4/6/7 TYPICAL I_{DD} vs. F_{OSC} OVER V_{DD} , INTOSC MODE



PIC16(L)F722/3/4/6/7

FIGURE 24-43: PIC16F722/3/4/6/7 TYPICAL ADC IPD vs. VDD, VCAP = 0.1 μ F

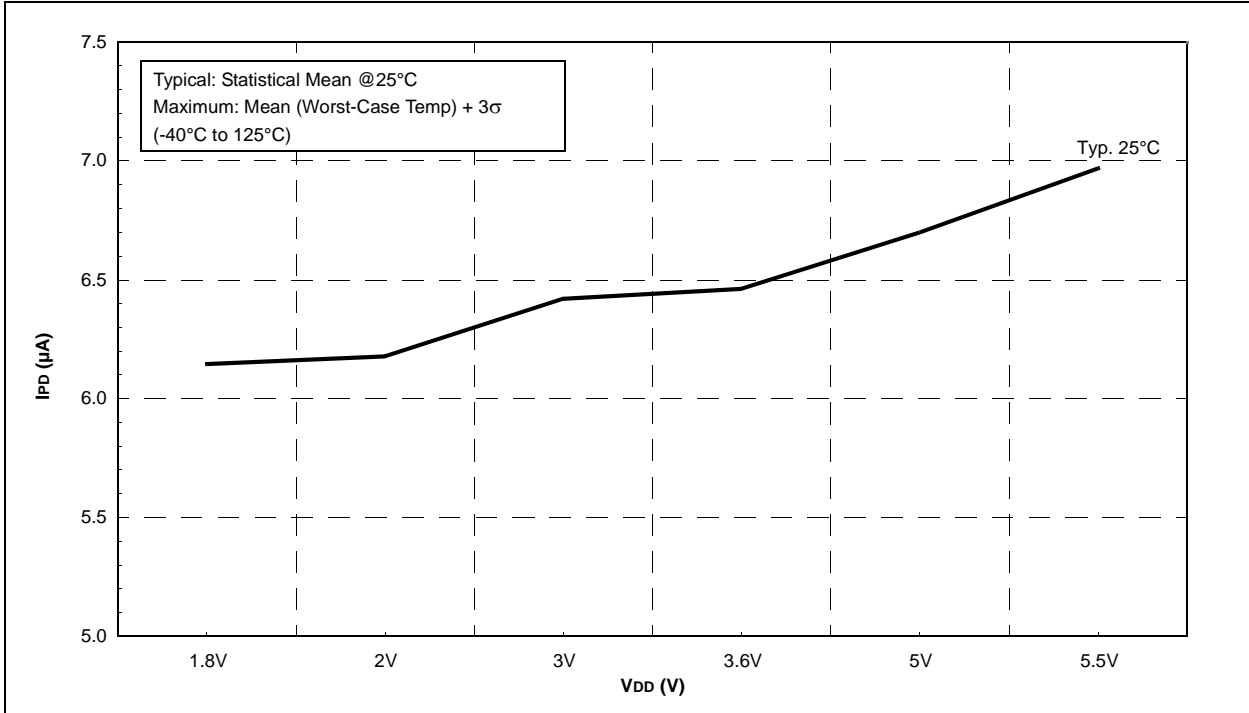
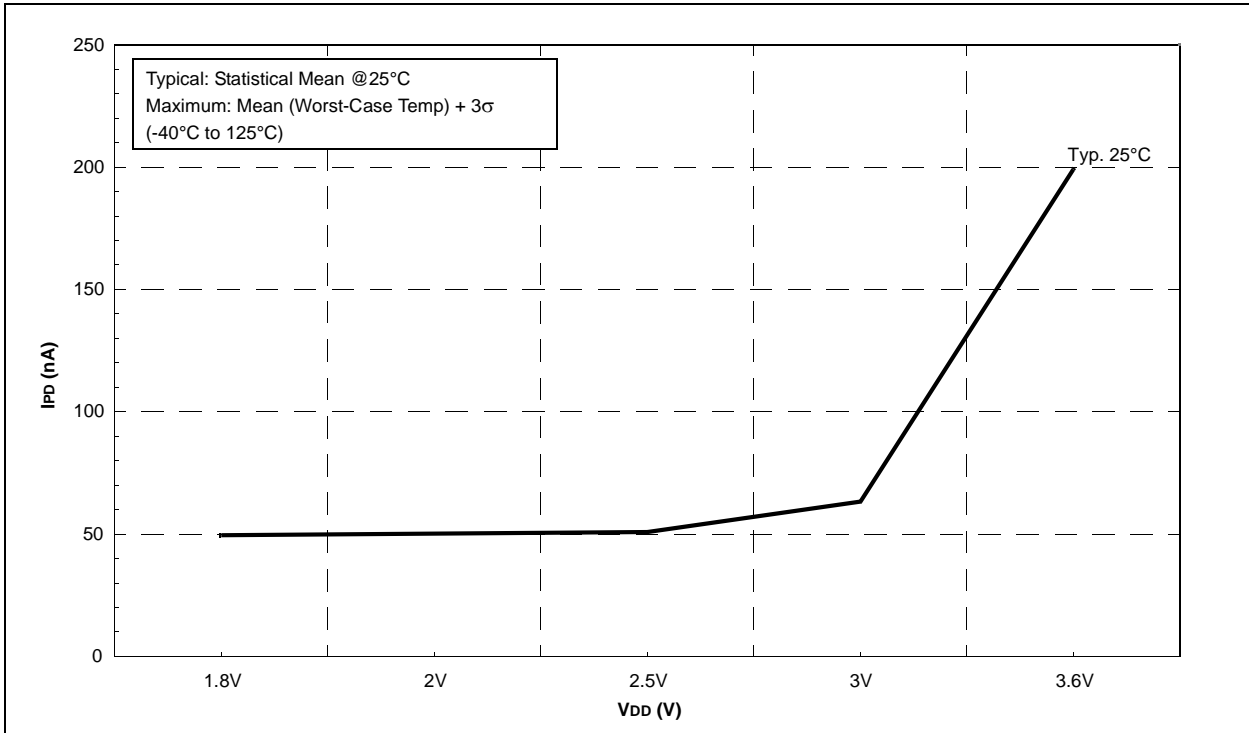


FIGURE 24-44: PIC16LF722/3/4/6/7 TYPICAL ADC IPD vs. VDD



PIC16(L)F722/3/4/6/7

FIGURE 24-61: PIC16F722/3/4/6/7 A/D INTERNAL RC OSCILLATOR PERIOD

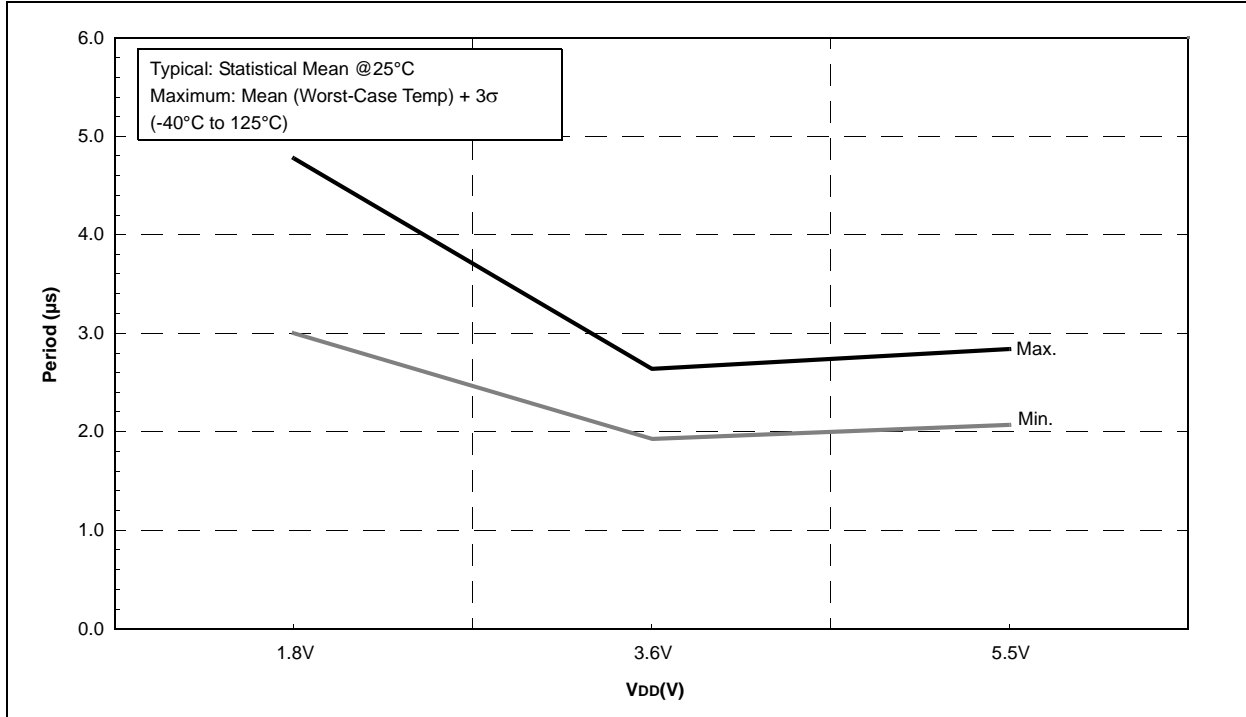
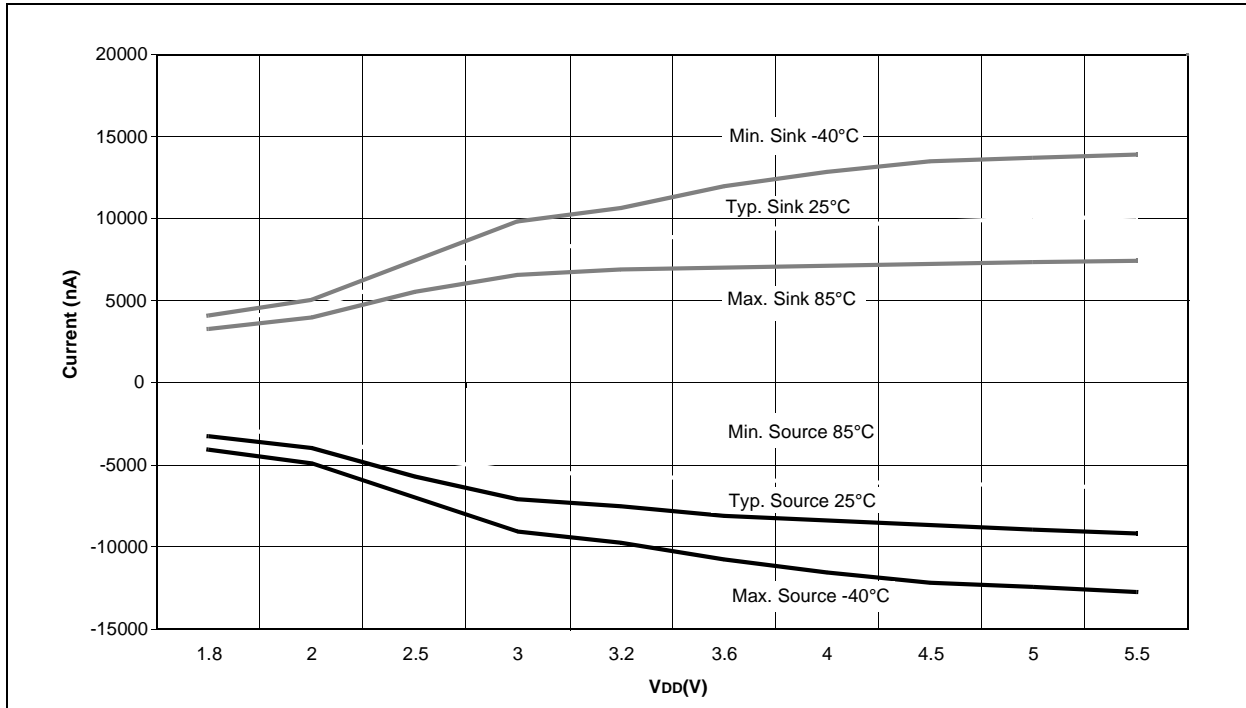


FIGURE 24-62: PIC16F722/3/4/6/7 CAP SENSE OUTPUT CURRENT, POWER MODE = HIGH



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>