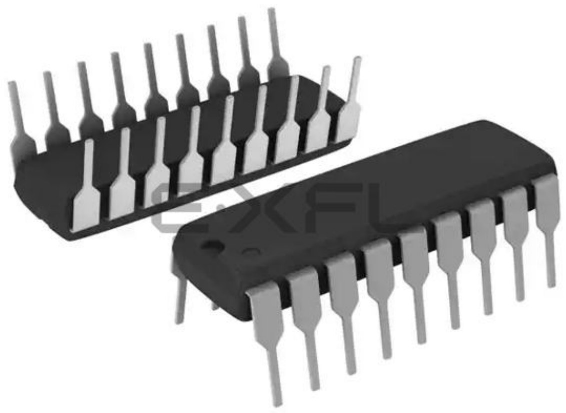
Microchip Technology - PIC16F648A-I/P Datasheet



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	UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	18-DIP (0.300", 7.62mm)
Supplier Device Package	18-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f648a-i-p

TABLE 4-5: SPECIAL FUNCTION REGISTERS SUMMARY BANK2

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset(1)	Details on Page
Bank 2											
100h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								`xxxx xxxx`	30
101h	TMR0	Timer0 Module’s Register								`xxxx xxxx`	47
102h	PCL	Program Counter’s (PC) Least Significant Byte								`0000 0000`	30
103h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	`0001 1xxx`	24
104h	FSR	Indirect Data Memory Address Pointer								`xxxx xxxx`	30
105h	—	Unimplemented								—	—
106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	`xxxx xxxx`	38
107h	—	Unimplemented								—	—
108h	—	Unimplemented								—	—
109h	—	Unimplemented								—	—
10Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of Program Counter					`---0 0000`	30
10Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	`0000 000x`	26
10Ch	—	Unimplemented								—	—
10Dh	—	Unimplemented								—	—
10Eh	—	Unimplemented								—	—
10Fh	—	Unimplemented								—	—
110h	—	Unimplemented								—	—
111h	—	Unimplemented								—	—
112h	—	Unimplemented								—	—
113h	—	Unimplemented								—	—
114h	—	Unimplemented								—	—
115h	—	Unimplemented								—	—
116h	—	Unimplemented								—	—
117h	—	Unimplemented								—	—
118h	—	Unimplemented								—	—
119h	—	Unimplemented								—	—
11Ah	—	Unimplemented								—	—
11Bh	—	Unimplemented								—	—
11Ch	—	Unimplemented								—	—
11Dh	—	Unimplemented								—	—
11Eh	—	Unimplemented								—	—
11Fh	—	Unimplemented								—	—

Legend: `-` = Unimplemented locations read as ‘`0`’, `u` = unchanged, `x` = unknown, `q` = value depends on condition, shaded = unimplemented.
Note 1: For the initialization condition for registers tables, refer to Table 14-6 and Table 14-7.

4.2.2.1 Status Register

The Status register, shown in Register 4-1, contains the arithmetic status of the ALU; the Reset status and the bank select bits for data memory (SRAM).

The Status register can be the destination for any instruction, like any other register. If the Status register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits are non-writable. Therefore, the result of an instruction with the Status register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the Status register as "`000uu1uu`" (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the Status register because these instructions do not affect any Status bit. For other instructions, not affecting any Status bits, see the "Instruction Set Summary".

Note: The C and DC bits operate as a $\overline{\text{Borrow}}$ and $\overline{\text{Digit Borrow}}$ out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

REGISTER 4-1: STATUS – STATUS REGISTER (ADDRESS: 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP**: Register Bank Select bit (used for indirect addressing)
`1` = Bank 2, 3 (100h-1FFh)
`0` = Bank 0, 1 (00h-FFh)

bit 6-5 **RP<1:0>**: Register Bank Select bits (used for direct addressing)
`00` = Bank 0 (00h-7Fh)
`01` = Bank 1 (80h-FFh)
`10` = Bank 2 (100h-17Fh)
`11` = Bank 3 (180h-1FFh)

bit 4 **$\overline{\text{TO}}$**: Time Out bit
`1` = After power-up, `CLRWDT` instruction or `SLEEP` instruction
`0` = A WDT time out occurred

bit 3 **$\overline{\text{PD}}$**: Power-down bit
`1` = After power-up or by the `CLRWDT` instruction
`0` = By execution of the `SLEEP` instruction

bit 2 **Z**: Zero bit
`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC**: Digit Carry/$\overline{\text{Borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions) (for $\overline{\text{Borrow}}$ the polarity is reversed)
`1` = A carry-out from the 4th low order bit of the result occurred
`0` = No carry-out from the 4th low order bit of the result

bit 0 **C**: Carry/$\overline{\text{Borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

Note: For $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

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

4.2.2.4 PIE1 Register

This register contains interrupt enable bits.

REGISTER 4-4: PIE1 – PERIPHERAL INTERRUPT ENABLE REGISTER 1 (ADDRESS: 8Ch)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7	**EEIE:** EE Write Complete Interrupt Enable Bit `1` = Enables the EE write complete interrupt `0` = Disables the EE write complete interrupt
bit 6	**CMIE**: Comparator Interrupt Enable bit `1` = Enables the comparator interrupt `0` = Disables the comparator interrupt
bit 5	**RCIE**: USART Receive Interrupt Enable bit `1` = Enables the USART receive interrupt `0` = Disables the USART receive interrupt
bit 4	**TXIE**: USART Transmit Interrupt Enable bit `1` = Enables the USART transmit interrupt `0` = Disables the USART transmit interrupt
bit 3	**Unimplemented**: Read as ‘`0`’
bit 2	**CCP1IE**: CCP1 Interrupt Enable bit `1` = Enables the CCP1 interrupt `0` = Disables the CCP1 interrupt
bit 1	**TMR2IE**: TMR2 to PR2 Match Interrupt Enable bit `1` = Enables the TMR2 to PR2 match interrupt `0` = Disables the TMR2 to PR2 match interrupt
bit 0	**TMR1IE**: TMR1 Overflow Interrupt Enable bit `1` = Enables the TMR1 overflow interrupt `0` = Disables the TMR1 overflow interrupt

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

8.0 TIMER2 MODULE

Timer2 is an 8-bit timer with a prescaler and a postscaler. It can be used as the PWM time base for PWM mode of the CCP module. The TMR2 register is readable and writable, and is cleared on any device Reset.

The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits T2CKPS<1:0> (T2CON<1:0>).

The Timer2 module has an 8-bit period register PR2. The TMR2 register value increments from 00h until it matches the PR2 register value and then resets to 00h on the next increment cycle. The PR2 register is a readable and writable register. The PR2 register is initialized to FFh upon Reset.

The match output of Timer2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a Timer2 interrupt (latched in flag bit TMR2IF, (PIR1<1>)).

Timer2 can be shut off by clearing control bit TMR2ON (T2CON<2>) to minimize power consumption.

Register 8-1 shows the Timer2 control register.

8.1 Timer2 Prescaler and Postscaler

The prescaler and postscaler counters are cleared when any of the following occurs:

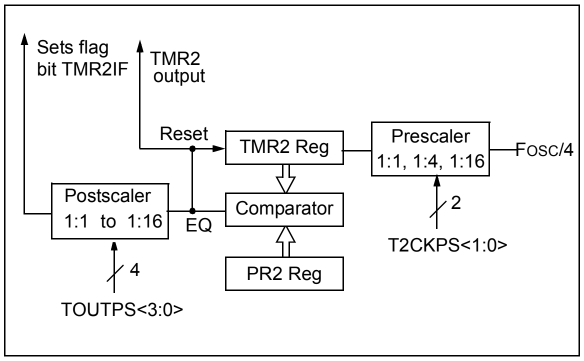
- a write to the TMR2 register
- a write to the T2CON register
- any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

The TMR2 register is not cleared when T2CON is written.

8.2 TMR2 Output

The TMR2 output (before the postscaler) is fed to the Synchronous Serial Port module which optionally uses it to generate shift clock.

FIGURE 8-1: TIMER2 BLOCK DIAGRAM



REGISTER 8-1: T2CON – TIMER2 CONTROL REGISTER (ADDRESS: 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented**: Read as '0'

bit 6-3 **TOUTPS<3:0>**: Timer2 Output Postscale Select bits

`0000` = 1:1 Postscale Value
`0001` = 1:2 Postscale Value
•
•
•
`1111` = 1:16 Postscale

bit 2 **TMR2ON**: Timer2 On bit

`1` = Timer2 is on
`0` = Timer2 is off

bit 1-0 **T2CKPS<1:0>**: Timer2 Clock Prescale Select bits

`00` = 1:1 Prescaler Value
`01` = 1:4 Prescaler Value
`1x` = 1:16 Prescaler Value

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

TABLE 8-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
11h	TMR2	Timer2 Module's Register								0000 0000	0000 0000
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
92h	PR2	Timer2 Period Register								1111 1111	1111 1111

Legend: `x` = unknown, `u` = unchanged, `-` = unimplemented read as '0'. Shaded cells are not used by the Timer2 module.

NOTES:

9.1 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on pin RB3/CCP1. An event is defined as:

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

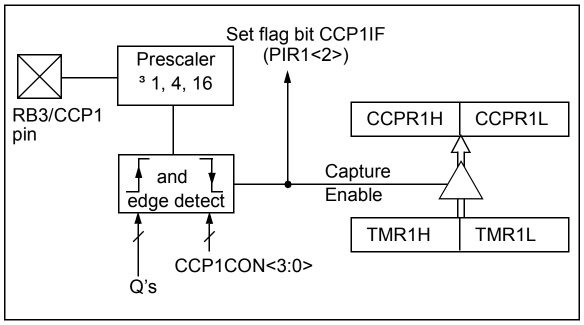
An event is selected by control bits CCP1M<3:0> (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit CCP1IF (PIR1<2>) is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value will be lost.

9.1.1 CCP PIN CONFIGURATION

In Capture mode, the RB3/CCP1 pin should be configured as an input by setting the TRISB<3> bit.

Note: If the RB3/CCP1 is configured as an output, a write to the port can cause a capture condition.

FIGURE 9-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



9.1.2 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, the capture operation may not work.

9.1.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP1IE (PIE1<2>) clear to avoid false interrupts and should clear the flag bit CCP1IF following any such change in Operating mode.

9.1.4 CCP PRESCALER

There are four prescaler settings, specified by bits CCP1M<3:0>. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore the first capture may be from a non-zero prescaler. Example 9-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

EXAMPLE 9-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF    CCP1CON     ;Turn CCP module off
MOVLW   NEW_CAPT_PS ;Load the W reg with
                    ; the new prescaler
                    ; mode value and CCP ON
MOVWF   CCP1CON     ;Load CCP1CON with this
                    ; value
```

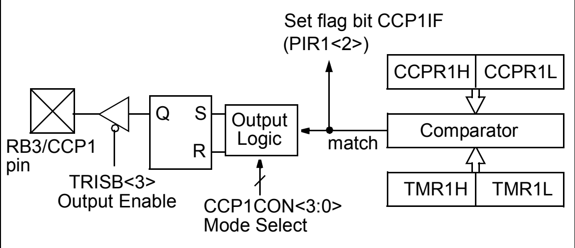
9.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the RB3/CCP1 pin is:

- Driven high
- Driven low
- Remains unchanged

The action on the pin is based on the value of control bits CCP1M<3:0> (CCP1CON<3:0>). At the same time, interrupt flag bit CCP1IF is set.

FIGURE 9-2: COMPARE MODE OPERATION BLOCK DIAGRAM



Note: Special event trigger will reset Timer1, but not set interrupt flag bit TMR1IF (PIR1<0>).

11.0 VOLTAGE REFERENCE MODULE

The Voltage Reference module consists of a 16-tap resistor ladder network that provides a selectable voltage reference. The resistor ladder is segmented to provide two ranges of VREF values and has a power-down function to conserve power when the reference is not being used. The VRCON register controls the operation of the reference as shown in Figure 11-1. The block diagram is given in Figure 11-1.

11.1 Voltage Reference Configuration

The Voltage Reference module can output 16 distinct voltage levels for each range.

The equations used to calculate the output of the Voltage Reference module are as follows:

if VRR = 1:

$$V_{REF} = \frac{VR\langle 3:0\rangle}{24} \times V_{DD}$$

if VRR = 0:

$$V_{REF} = \left(V_{DD} \times \frac{1}{4}\right) + \frac{VR\langle 3:0\rangle}{32} \times V_{DD}$$

The setting time of the Voltage Reference module must be considered when changing the VREF output (Table 17-3). Example 11-1 demonstrates how voltage reference is configured for an output voltage of 1.25V with VDD = 5.0V.

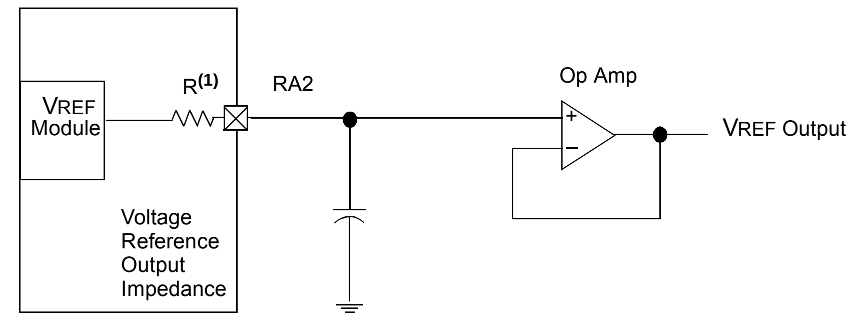
REGISTER 11-1: VRCON – VOLTAGE REFERENCE CONTROL REGISTER (ADDRESS: 9Fh)

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
VREN	VROE	VRR	—	VR3	VR2	VR1	VR0
bit 7							bit 0

bit 7	**VREN**: VREF Enable bit 1 = VREF circuit powered on 0 = VREF circuit powered down, no IDD drain
bit 6	**VROE**: VREF Output Enable bit 1 = VREF is output on RA2 pin 0 = VREF is disconnected from RA2 pin
bit 5	**VRR**: VREF Range Selection bit 1 = Low range 0 = High range
bit 4	**Unimplemented**: Read as '0'
bit 3-0	**VR<3:0>**: VREF Value Selection bits 0 ≤ VR <3:0> ≤ 15 When VRR = 1: VREF = (VR<3:0>/ 24) * VDD When VRR = 0: VREF = 1/4 * VDD + (VR<3:0>/ 32) * VDD

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

FIGURE 11-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the voltage reference configuration VRCON<3:0> and VRCON<5>.

TABLE 11-1: REGISTERS ASSOCIATED WITH VOLTAGE REFERENCE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value On POR	Value On All Other Resets
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	`000- 0000`	`000- 0000`
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	`0000 0000`	`0000 0000`
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	`1111 1111`	`1111 1111`

Legend: - = Unimplemented, read as '0'.

REGISTER 12-2: RCSTA – RECEIVE STATUS AND CONTROL REGISTER (ADDRESS: 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**: Serial Port Enable bit
(Configures RB1/RX/DT and RB2/TX/CK pins as serial port pins when bits TRISB<2:1> are set)
`1` = Serial port enabled
`0` = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
`1` = Selects 9-bit reception
`0` = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care
Synchronous mode - master:
`1` = Enables single receive
`0` = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode - slave:
Unused in this mode

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
`1` = Enables continuous receive
`0` = Disables continuous receive
Synchronous mode:
`1` = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
`0` = Disables continuous receive

bit 3 **ADEN**: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = `1`):
`1` = Enables address detection, enable interrupt and load of the receive buffer when RSR<8> is set
`0` = Disables address detection, all bytes are received, and ninth bit can be used as parity bit
Asynchronous mode 8-bit (RX9 = `0`):
Unused in this mode
Synchronous mode
Unused in this mode

bit 2 **FERR**: Framing Error bit
`1` = Framing error (Can be updated by reading RCREG register and receive next valid byte)
`0` = No framing error

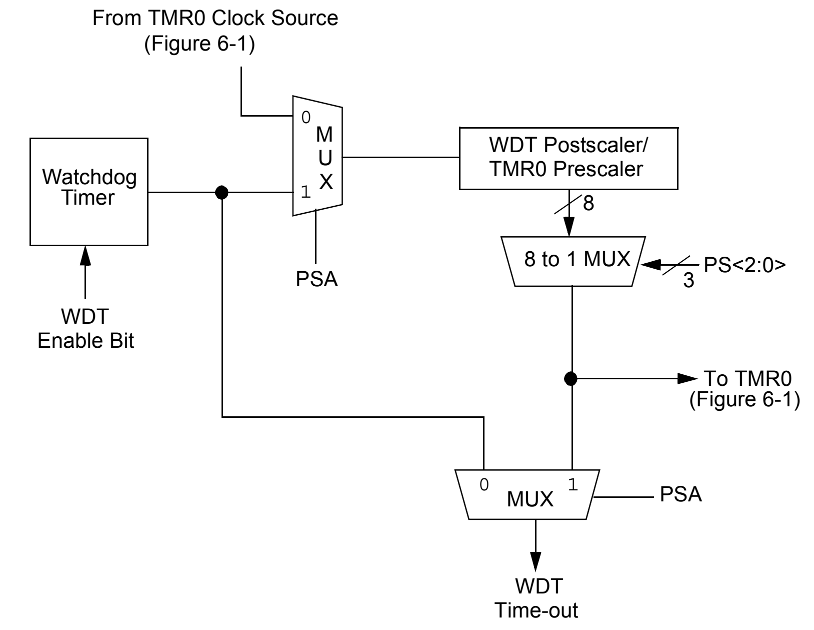
bit 1 **OERR**: Overrun Error bit
`1` = Overrun error (Can be cleared by clearing bit CREN)
`0` = No overrun error

bit 0 **RX9D**: 9th bit of received data (Can be parity bit)

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

FIGURE 14-16: WATCHDOG TIMER BLOCK DIAGRAM



Note: T0SE, T0CS, PSA, PS0-PS2 are bits in the OPTION register.

TABLE 14-9: SUMMARY OF WATCHDOG TIMER REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other Resets
2007h	CONFIG	LVP	BOREN	MCLRE	FOSC2	$\overline{\text{PWRTE}}$	WDTE	FOSC1	FOSC0	`uuuu uuuu`	`uuuu uuuu`
81h, 181h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	`1111 1111`	`1111 1111`

Legend: `x` = unknown, `u` = unchanged, `-` = unimplemented read as '`0`', `q` = value depends upon condition.
Note: Shaded cells are not used by the Watchdog Timer.

14.8 Power-Down Mode (Sleep)

The Power-down mode is entered by executing a `SLEEP` instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{\text{PD}}$ bit in the Status register is cleared, the $\overline{\text{TO}}$ bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had, before `SLEEP` was executed (driving high, low or high-impedance).

For lowest current consumption in this mode, all I/O pins should be either at VDD or VSS with no external circuitry drawing current from the I/O pin and the comparators, and VREF should be disabled. I/O pins that are high-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at VDD or VSS for lowest current consumption. The contribution from on chip pull-ups on PORTB should be considered.

The $\overline{\text{MCLR}}$ pin must be at a logic high level (VIHMC).

Note:	It should be noted that a Reset generated by a WDT time-out does not drive $\overline{\text{MCLR}}$ pin low.

14.11 In-Circuit Serial Programming™ (ICSP™)

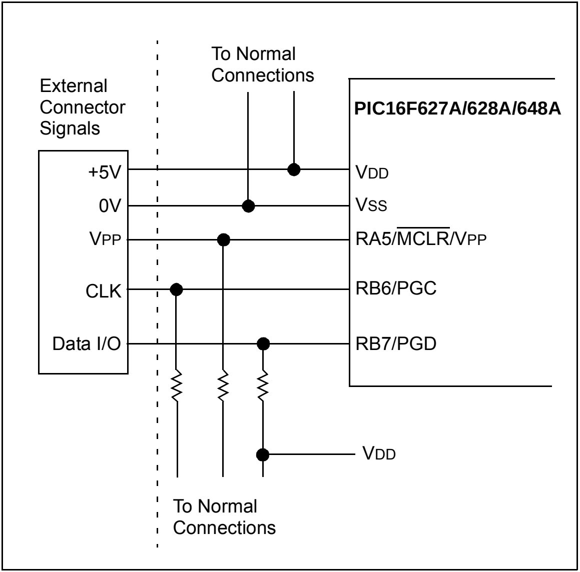
The PIC16F627A/628A/648A microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or custom firmware to be programmed.

The device is placed into a Program/Verify mode by holding the RB6 and RB7 pins low while raising the $\overline{\text{MCLR}}$ (VPP) pin from VIL to VIHH. See "*PIC16F627A/628A/648A EEPROM Memory Programming Specification*" (DS41196) for details. RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode.

After Reset, to place the device into Programming/Verify mode, the Program Counter (PC) is at location 00h. A 6-bit command is then supplied to the device. Depending on the command, 14 bits of program data are then supplied to or from the device, depending if the command was a load or a read. For complete details of serial programming, please refer to "*PIC16F627A/628A/648A EEPROM Memory Programming Specification*" (DS41196).

A typical In-Circuit Serial Programming connection is shown in Figure 14-18.

FIGURE 14-18: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION



14.12 Low-Voltage Programming

The LVP bit of the Configuration Word, enables the low-voltage programming. This mode allows the microcontroller to be programmed via ICSP using only a 5V source. This mode removes the requirement of VIHH to be placed on the $\overline{\text{MCLR}}$ pin. The LVP bit is normally erased to '1' which enables the low-voltage programming. In this mode, the RB4/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. The device will enter Programming mode when a '1' is placed on the RB4/PGM pin. The High-Voltage Programming mode is still available by placing VIHH on the $\overline{\text{MCLR}}$ pin.

Note 1: While in this mode, the RB4 pin can no longer be used as a general purpose I/O pin.

2: VDD must be 5.0V $\pm$10% during erase operations.

If Low-Voltage Programming mode is not used, the LVP bit should be programmed to a '0' so that RB4/PGM becomes a digital I/O pin. To program the device, VIHH must be placed onto $\overline{\text{MCLR}}$ during programming. The LVP bit may only be programmed when programming is entered with VIHH on $\overline{\text{MCLR}}$. The LVP bit cannot be programmed when programming is entered with RB4/PGM.

It should be noted, that once the LVP bit is programmed to '0', only High-Voltage Programming mode can be used to program the device.

NOTES:

15.2 Instruction Descriptions

ADDLW	**Add Literal and W**
Syntax:	[*label*] ADDLW k
Operands:	$0 \le k \le 255$
Operation:	(W) + k $\rightarrow$ (W)
Status Affected:	C, DC, Z
Encoding:	`11` | `111x` | `kkkk` | `kkkk`
Description:	The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.
Words:	1
Cycles:	1

Example	`ADDLW 0x15`
	Before Instruction
	W = 0x10
	After Instruction
	W = 0x25

ADDWF	**Add W and f**
Syntax:	[*label*] ADDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) + (f) $\rightarrow$ (dest)
Status Affected:	C, DC, Z
Encoding:	`00` | `0111` | `dfff` | `ffff`
Description:	Add the contents of the W register with register 'f'. If 'd' is '`0`', the result is stored in the W register. If 'd' is '`1`', the result is stored back in register 'f'.
Words:	1
Cycles:	1

Example	`ADDWF REG1, 0`
	Before Instruction
	W = 0x17
	REG1 = 0xC2
	After Instruction
	W = 0xD9
	REG1 = 0xC2
	Z = 0
	C = 0
	DC = 0

ANDLW	**AND Literal with W**
Syntax:	[*label*] ANDLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .AND. (k) $\rightarrow$ (W)
Status Affected:	Z
Encoding:	`11` | `1001` | `kkkk` | `kkkk`
Description:	The contents of W register are AND'ed with the eight bit literal 'k'. The result is placed in the W register.
Words:	1
Cycles:	1

Example	`ANDLW 0x5F`
	Before Instruction
	W = 0xA3
	After Instruction
	W = 0x03

ANDWF	**AND W with f**
Syntax:	[*label*] ANDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .AND. (f) $\rightarrow$ (dest)
Status Affected:	Z
Encoding:	`00` | `0101` | `dfff` | `ffff`
Description:	AND the W register with register 'f'. If 'd' is '`0`', the result is stored in the W register. If 'd' is '`1`', the result is stored back in register 'f'.
Words:	1
Cycles:	1

Example	`ANDWF REG1, 1`
	Before Instruction
	W = 0x17
	REG1 = 0xC2
	After Instruction
	W = 0x17
	REG1 = 0x02

SUBWF	Subtract W from f			
Syntax:	[*label*] SUBWF f,d			
Operands:	$0 \le f \le 127$ $d \in [0,1]$			
Operation:	(f) - (W) $\rightarrow$ (dest)			
Status Affected:	C, DC, Z			
Encoding:	`00`	`0010`	`dfff`	`ffff`
Description:	Subtract (2's complement method) W register from register 'f'. If 'd' is '`0`', the result is stored in the W register. If 'd' is '`1`', the result is stored back in register 'f'.			
Words:	1			
Cycles:	1			

<u>Example 1</u>: `SUBWF REG1, 1`

Before Instruction

REG1 = 3
W = 2
C = ?

After Instruction

REG1 = `1`
W = 2
C = `1`; result is positive
DC = `1`
Z = `0`

<u>Example 2</u>: Before Instruction

REG1 = 2
W = 2
C = ?

After Instruction

REG1 = `0`
W = 2
C = `1`; result is zero
Z = DC = `1`

<u>Example 3</u>: Before Instruction

REG1 = `1`
W = 2
C = ?

After Instruction

REG1 = 0xFF
W = 2
C = `0`; result is negative
Z = DC = `0`

SWAPF	Swap Nibbles in f			
Syntax:	[*label*] SWAPF f,d			
Operands:	$0 \le f \le 127$ $d \in [0,1]$			
Operation:	(f<3:0>) $\rightarrow$ (dest<7:4>), (f<7:4>) $\rightarrow$ (dest<3:0>)			
Status Affected:	None			
Encoding:	`00`	`1110`	`dfff`	`ffff`
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' is '`0`', the result is placed in W register. If 'd' is '`1`', the result is placed in register 'f'.			
Words:	1			
Cycles:	1			

<u>Example</u> `SWAPF REG1, 0`

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5
W = 0x5A

TRIS	Load TRIS Register			
Syntax:	[*label*] TRIS f			
Operands:	$5 \le f \le 7$			
Operation:	(W) $\rightarrow$ TRIS register f;			
Status Affected:	None			
Encoding:	`00`	`0000`	`0110`	`0fff`
Description:	The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.			
Words:	1			
Cycles:	1			
<u>Example</u>				

To maintain upward compatibility with future PIC® MCU products, do not use this instruction.

TABLE 17-8: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Sym	Characteristic			Min	Typ†	Max	Units	Conditions
40	TT0H	T0CKI High Pulse Width	No Prescaler		0.5TCY + 20*	—	—	ns	
			With Prescaler		10*	—	—	ns	
41	TT0L	T0CKI Low Pulse Width	No Prescaler		0.5TCY + 20*	—	—	ns	
			With Prescaler		10*	—	—	ns	
42	TT0P	T0CKI Period			Greater of: 20 or $\frac{TCY + 40^*}{N}$	—	—	ns	N = prescale value (2, 4, ..., 256)
45	TT1H	T1CKI High Time	Synchronous, No Prescaler		0.5TCY + 20*	—	—	ns	
			Synchronous, with Prescaler	PIC16F62XA	15*	—	—	ns	
				PIC16LF62XA	25*	—	—	ns	
			Asynchronous	PIC16F62XA	30*	—	—	ns	
				PIC16LF62XA	50*	—	—	ns	
46	TT1L	T1CKI Low Time	Synchronous, No Prescaler		0.5TCY + 20*	—	—	ns	
			Synchronous, with Prescaler	PIC16F62XA	15*	—	—	ns	
				PIC16LF62XA	25*	—	—	ns	
			Asynchronous	PIC16F62XA	30*	—	—	ns	
				PIC16LF62XA	50*	—	—	ns	
47	TT1P	T1CKI input period	Synchronous	PIC16F62XA	Greater of: 20 or $\frac{TCY + 40^*}{N}$	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16LF62XA	Greater of: 20 or $\frac{TCY + 40^*}{N}$	—	—	—	
			Asynchronous	PIC16F62XA	60*	—	—	ns	
				PIC16LF62XA	100*	—	—	ns	
	FT1	Timer1 oscillator input frequency range (oscillator enabled by setting bit T1OSCEN)			—	32.7(1)	—	kHz	
48	TCKEZTMR1	Delay from external clock edge to timer increment			2TOSC	—	7TOSC	—	

* These parameters are characterized but not tested.

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

Note 1: This oscillator is intended to work only with 32.768 kHz watch crystals and their manufactured tolerances. Higher value crystal frequencies may not be compatible with this crystal driver.

FIGURE 17-9: CAPTURE/COMPARE/PWM TIMINGS

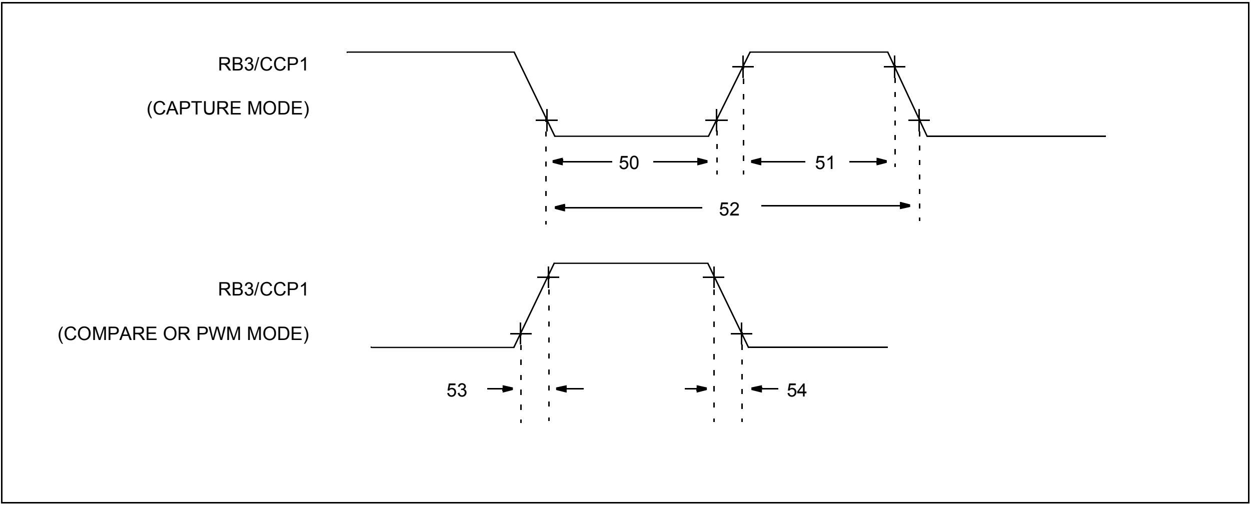


TABLE 17-9: CAPTURE/COMPARE/PWM REQUIREMENTS

Param No.	Sym	Characteristic			Min	Typ†	Max	Units	Conditions
50	TCCL	CCP input low time	No Prescaler		0.5TCY + 20*	—	—	ns	
			With Prescaler	PIC16F62XA	10*	—	—	ns	
				PIC16LF62XA	20*	—	—	ns	
51	TCCH	CCP input high time	No Prescaler		0.5TCY + 20*	—	—	ns	
			With Prescaler	PIC16F62XA	10*	—	—	ns	
				PIC16LF62XA	20*	—	—	ns	
52	TCCP	CCP input period			$\frac{3T_{CY} + 40^*}{N}$	—	—	ns	N = prescale value (1,4 or 16)
53	TCCR	CCP output rise time		PIC16F62XA		10	25*	ns	
				PIC16LF62XA		25	45*	ns	
54	TCCF	CCP output fall time		PIC16F62XA		10	25*	ns	
				PIC16LF62XA		25	45*	ns	

* These parameters are characterized but not tested.

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

FIGURE 18-4: TYPICAL BOR IPD VS. VDD

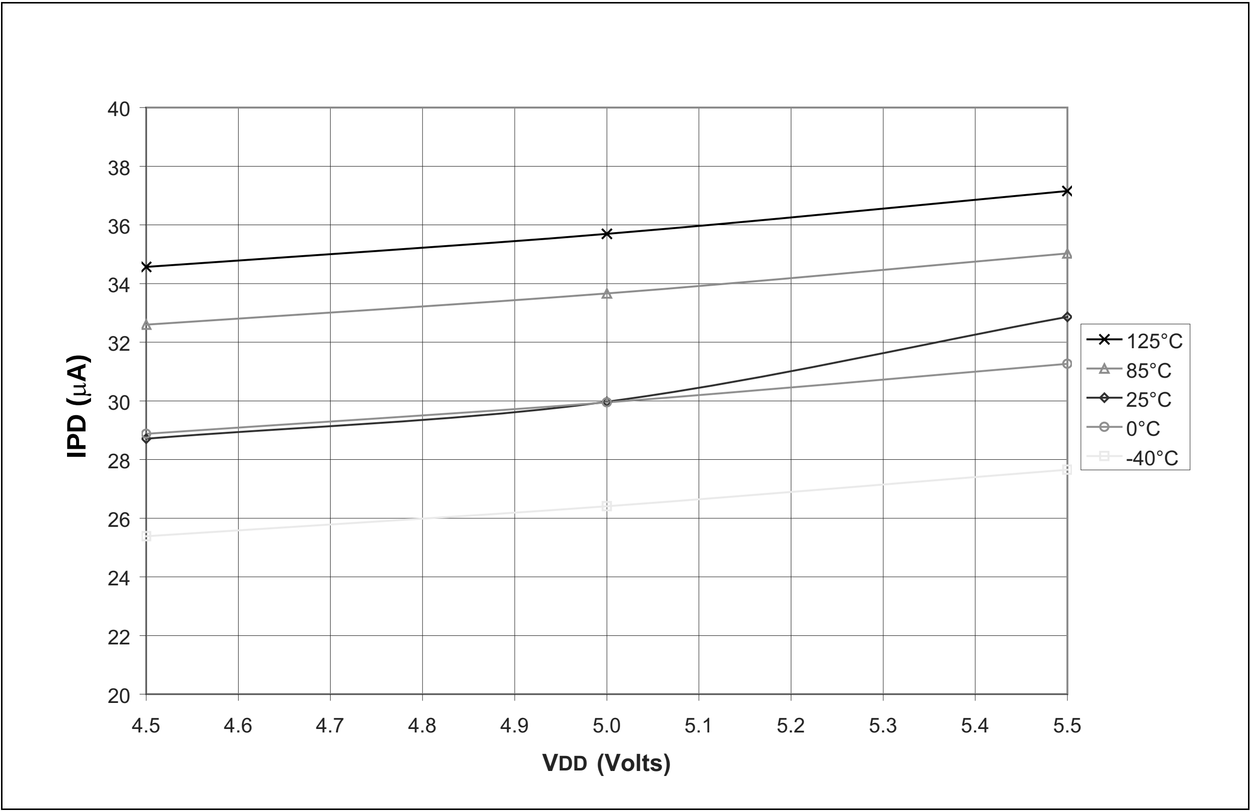


FIGURE 18-5: TYPICAL SINGLE COMPARATOR IPD VS. VDD

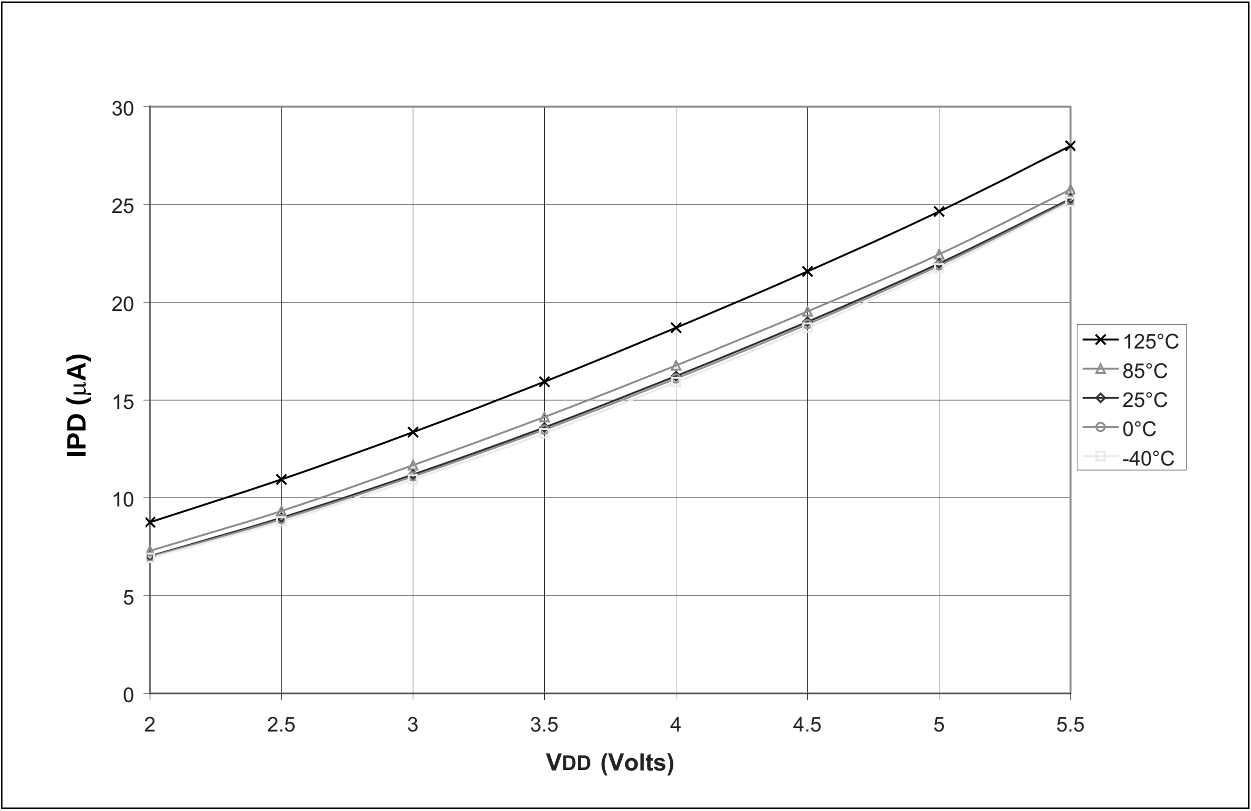


FIGURE 18-12: TYPICAL INTERNAL OSCILLATOR DEVIATION vs. VDD AT 25°C – 4 MHz MODE

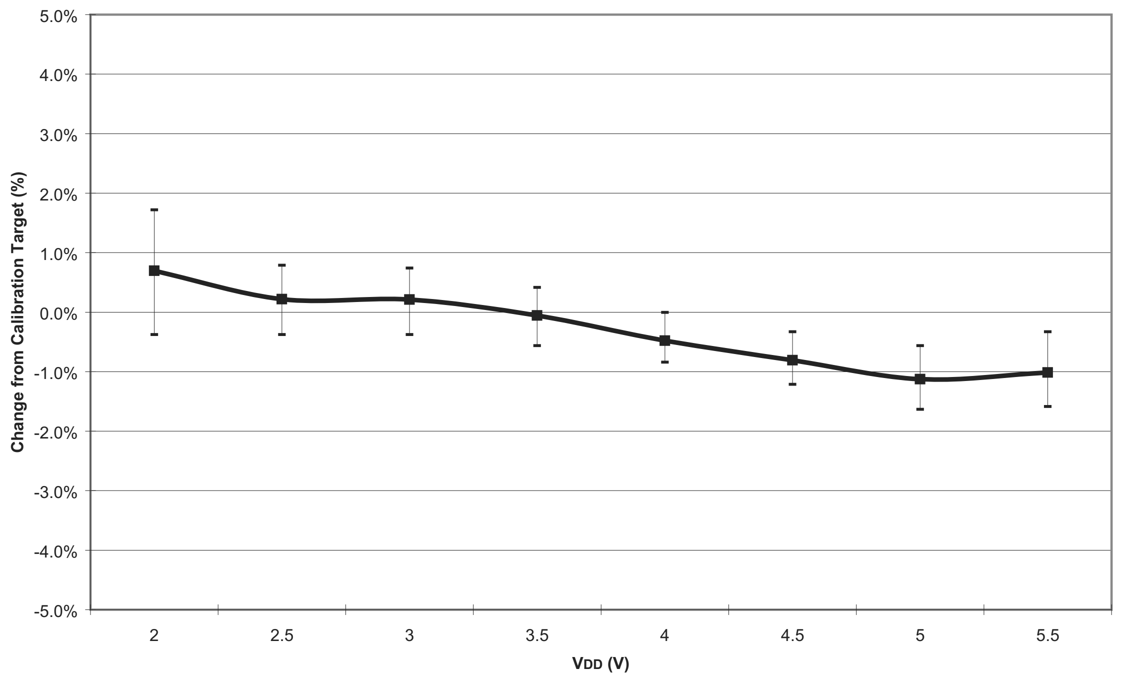


FIGURE 18-13: TYPICAL INTERNAL OSCILLATOR FREQUENCY vs. VDD TEMPERATURE = -40°C TO 85°C

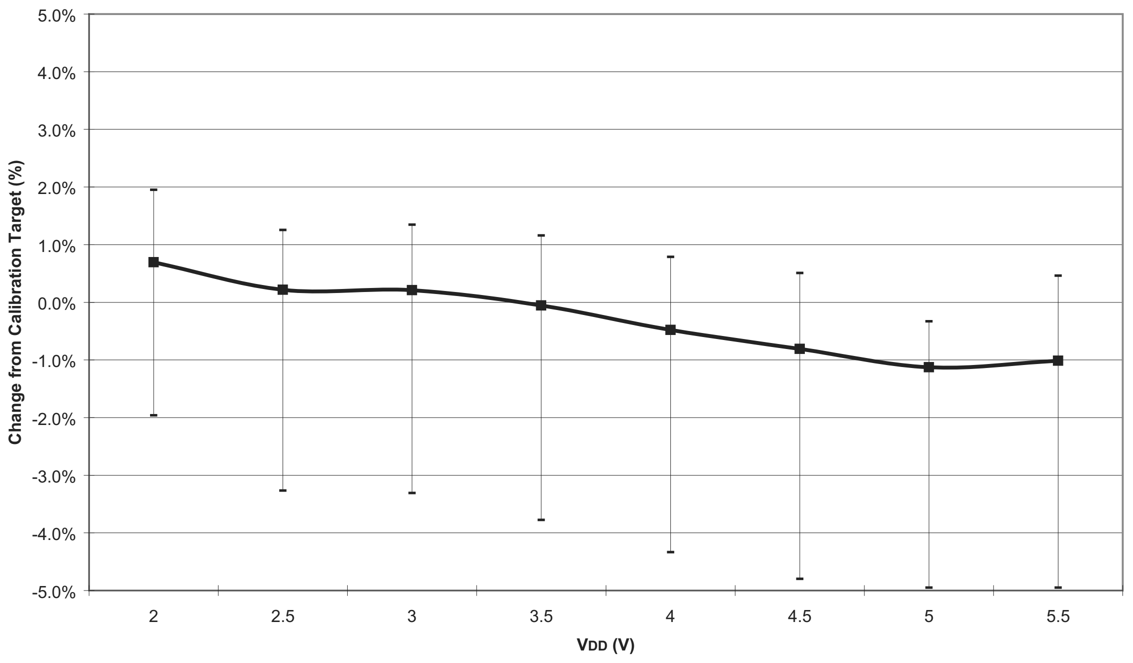


FIGURE 18-18: SUPPLY CURRENT (IDD vs. VDD, FOSC = 4 MHz (XT OSCILLATOR MODE)

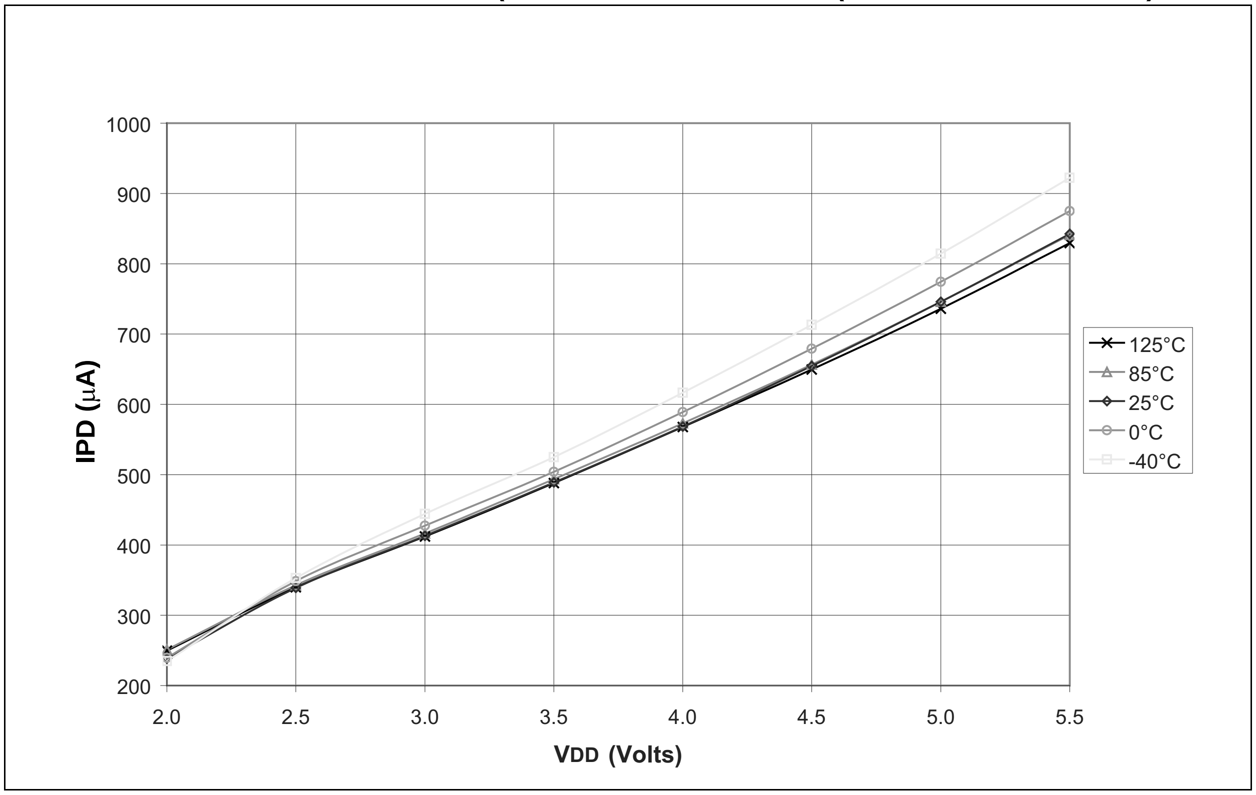
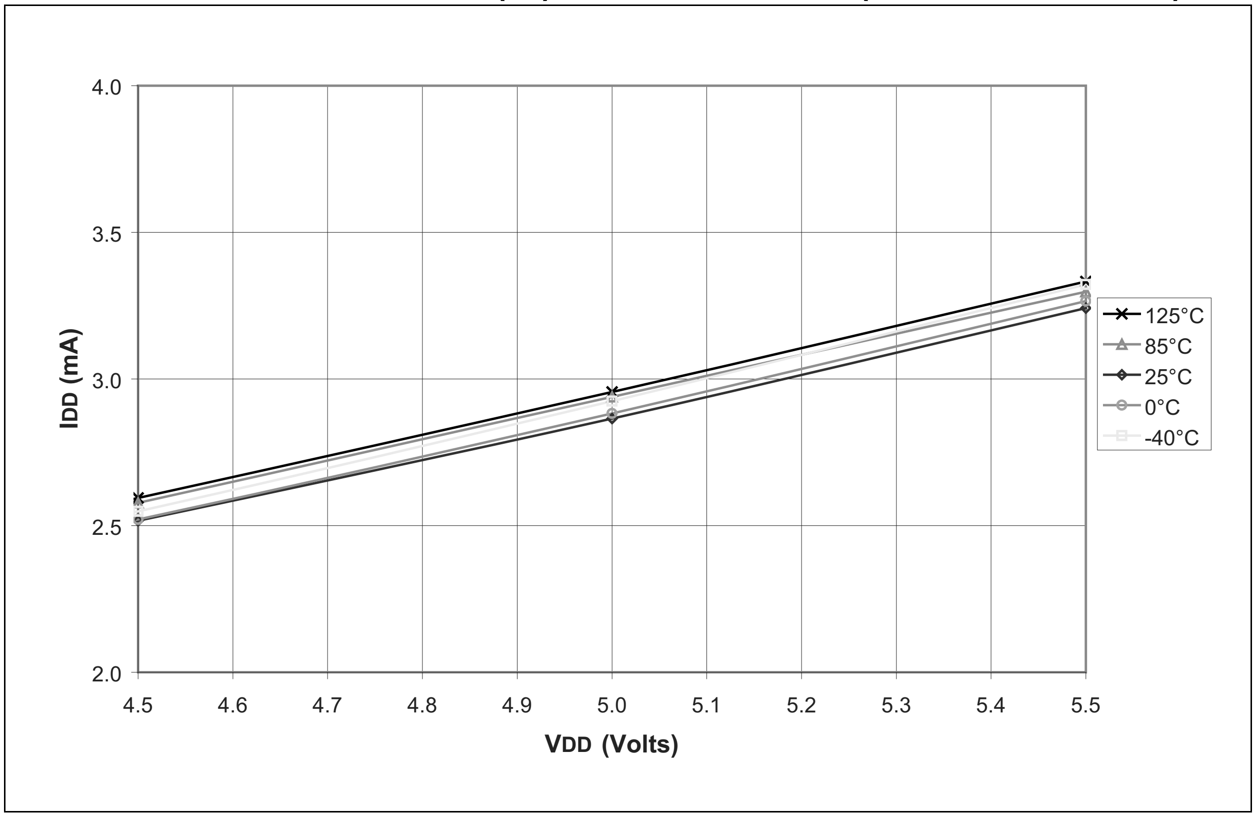


FIGURE 18-19: SUPPLY CURRENT (IDD) vs. VDD, FOSC = 20 MHz (HS OSCILLATOR MODE)


IDD (mA)
VDD (Volts)
125°C
85°C
25°C
0°C
-40°C