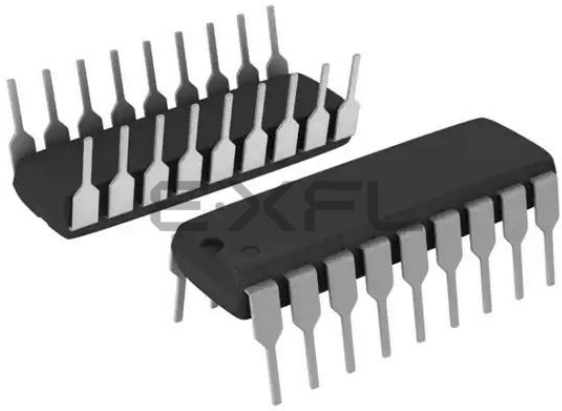


Welcome to E-XFL.COM

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	4MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	1.75KB (1K x 14)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	224 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°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/pic16lf627-04-p

1.0 PIC16F62X DEVICE VARIETIES

A variety of frequency ranges and packaging options are available. Depending on application and production requirements, the proper device option can be selected using the information in the PIC16F62X Product Identification System section (Page 167) at the end of this data sheet. When placing orders, please use this page of the data sheet to specify the correct part number.

1.1 FLASH Devices

FLASH devices can be erased and reprogrammed electrically. This allows the same device to be used for prototype development, pilot programs and production.

A further advantage of the electrically-erasable FLASH is that it can be erased and reprogrammed in-circuit, or by device programmers, such as Microchip's PICSTART® Plus, or PRO MATE® II programmers.

1.2 Quick-Turnaround Production (QTP) Devices

Microchip offers a QTP Programming Service for factory production orders. This service is made available for users who chose not to program a medium-to-high quantity of units and whose code patterns have stabilized. The devices are standard FLASH devices but with all program locations and configuration options already programmed by the factory. Certain code and prototype verification procedures apply before production shipments are available. Please contact your Microchip Technology sales office for more details.

1.3 Serialized Quick-Turnaround Production (SQTP^{sm}) Devices

Microchip offers a unique programming service where a few user-defined locations in each device are programmed with different serial numbers. The serial numbers may be random, pseudo-random or sequential.

Serial programming allows each device to have a unique number which can serve as an entry-code, password or ID number.

3.2.2.6 PCON Register

The PCON register contains flag bits to differentiate between a Power-on Reset, an external MCLR Reset, WDT Reset or a Brown-out Detect.

Note:	BOD is unknown on Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if BOD is cleared, indicating a brown-out has occurred. The BOD STATUS bit is a “don't care” and is not necessarily predictable if the brown-out circuit is disabled (by clearing the BODEN bit in the Configuration word).

REGISTER 3-6: PCON REGISTER (ADDRESS: 0Ch)

U-0	U-0	U-0	U-0	R/W-1	U-0	R/W-q	R/W-q
—	—	—	—	OSCF	—	POR	BOD
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **OSCF**: INTRC/ER oscillator frequency
1 = 4 MHz typical(1)
0 = 37 KHz typical

bit 2 **Unimplemented:** Read as '0'

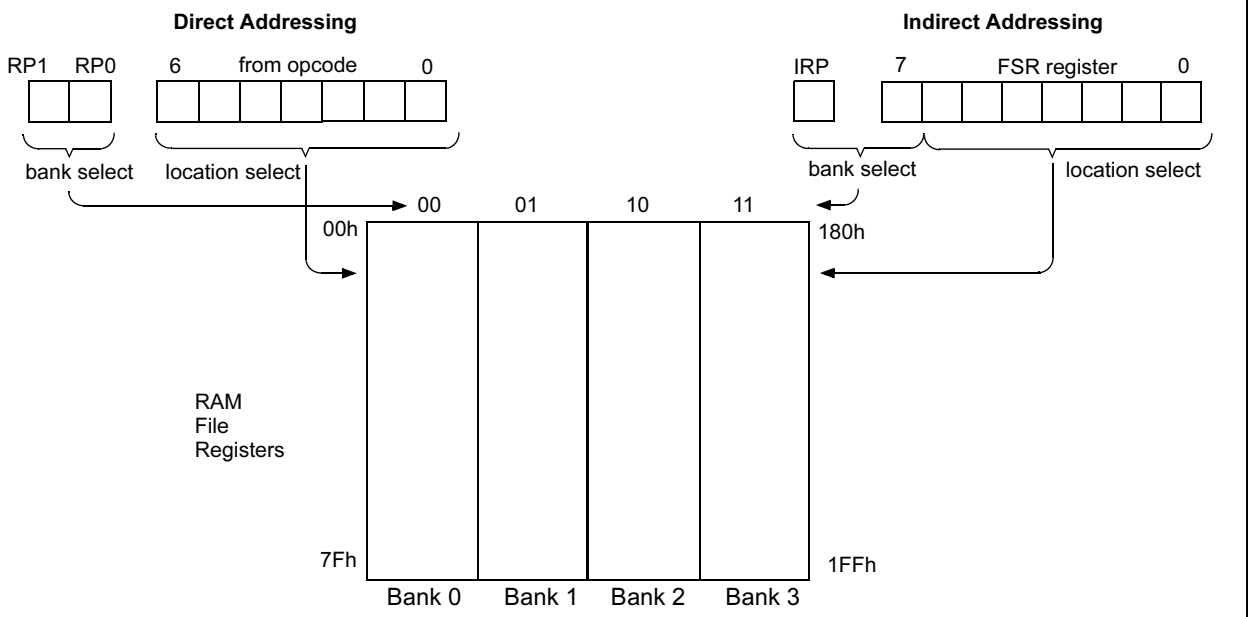
bit 1 **POR**: Power-on Reset STATUS bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **BOD**: Brown-out Detect STATUS bit
1 = No Brown-out Reset occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Note 1: When in ER Oscillator mode, setting OSCF = 1 will cause the oscillator frequency to change to the frequency specified by the external resistor.

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 3-4: DIRECT/INDIRECT ADDRESSING PIC16F62X



Note: For memory map detail see Figure 3-2.

FIGURE 5-1: BLOCK DIAGRAM OF RA0/AN0:RA1/AN1 PINS

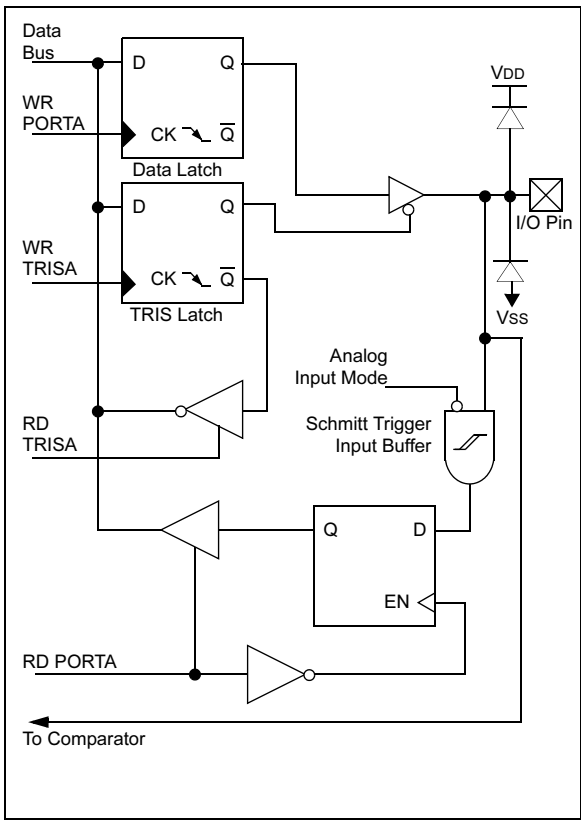


FIGURE 5-2: BLOCK DIAGRAM OF RA2/VREF PIN

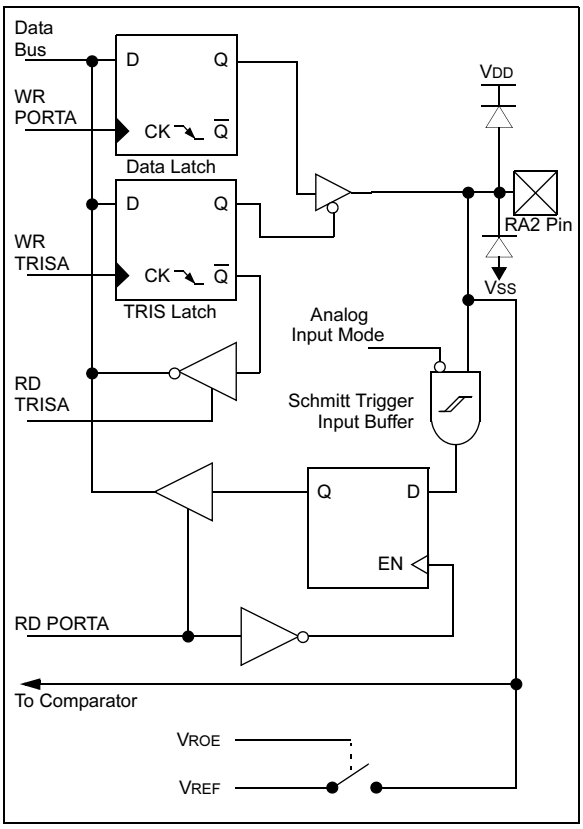


FIGURE 5-3: BLOCK DIAGRAM OF THE RA3/AN3 PIN

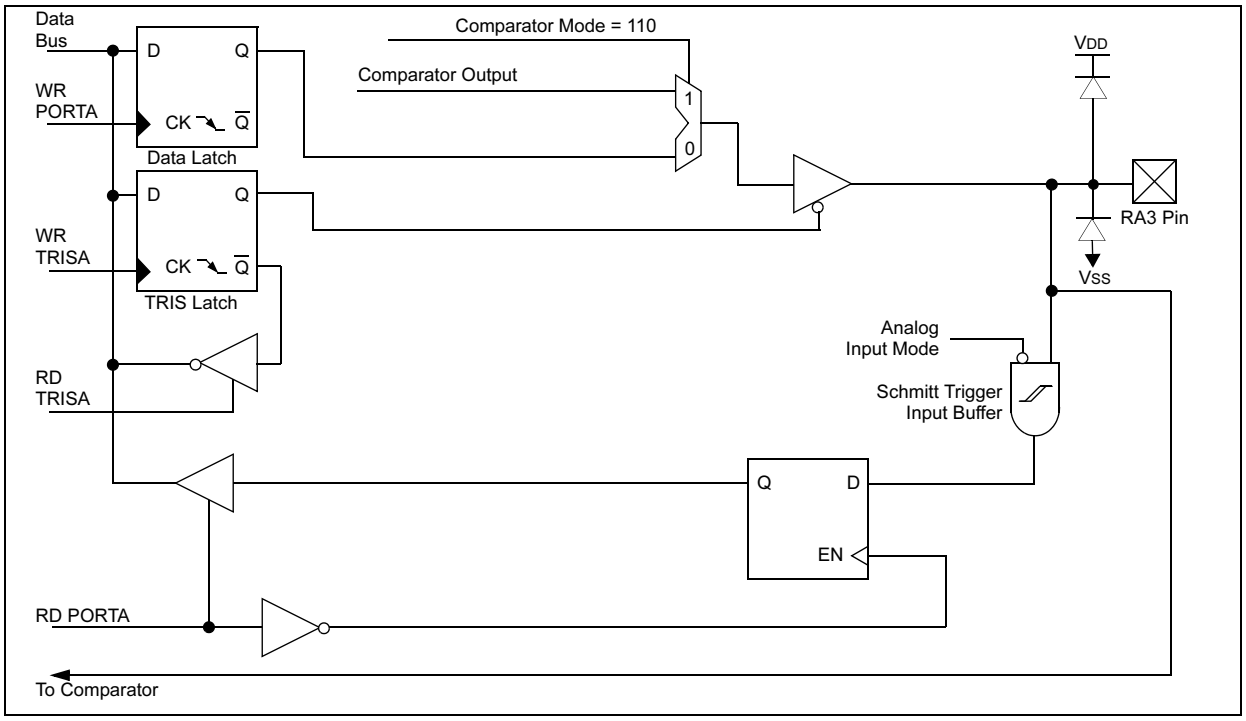


FIGURE 5-8: BLOCK DIAGRAM OF RB0/INT PIN

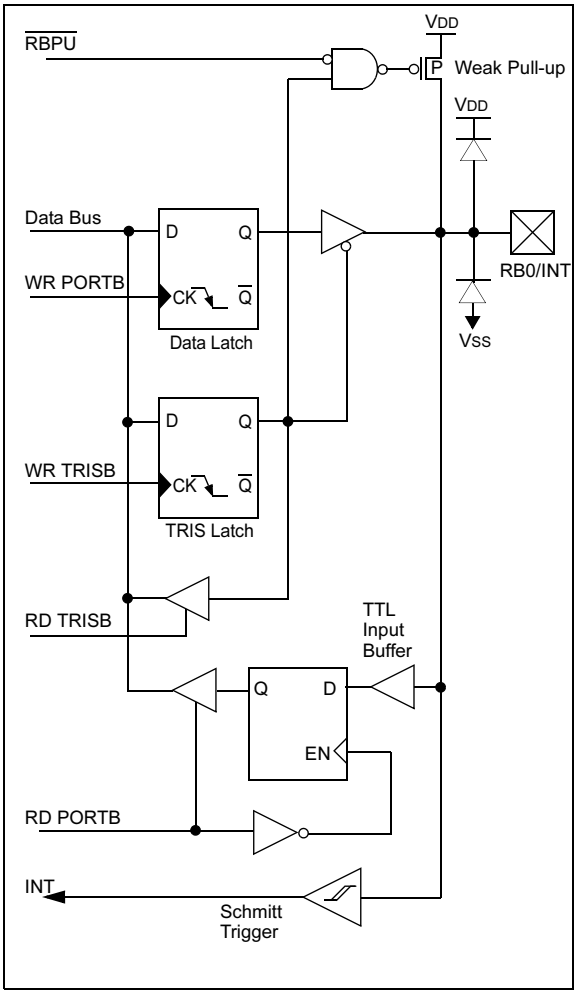
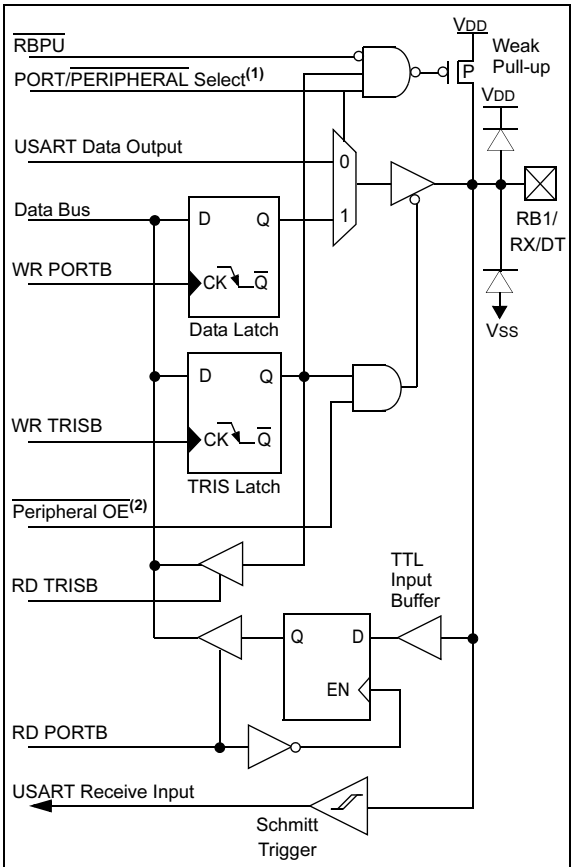


FIGURE 5-9: BLOCK DIAGRAM OF RB1/RX/DT PIN



Note 1: Port/Peripheral select signal selects between port data and peripheral output.
2: Peripheral OE (output enable) is only active if peripheral select is active.

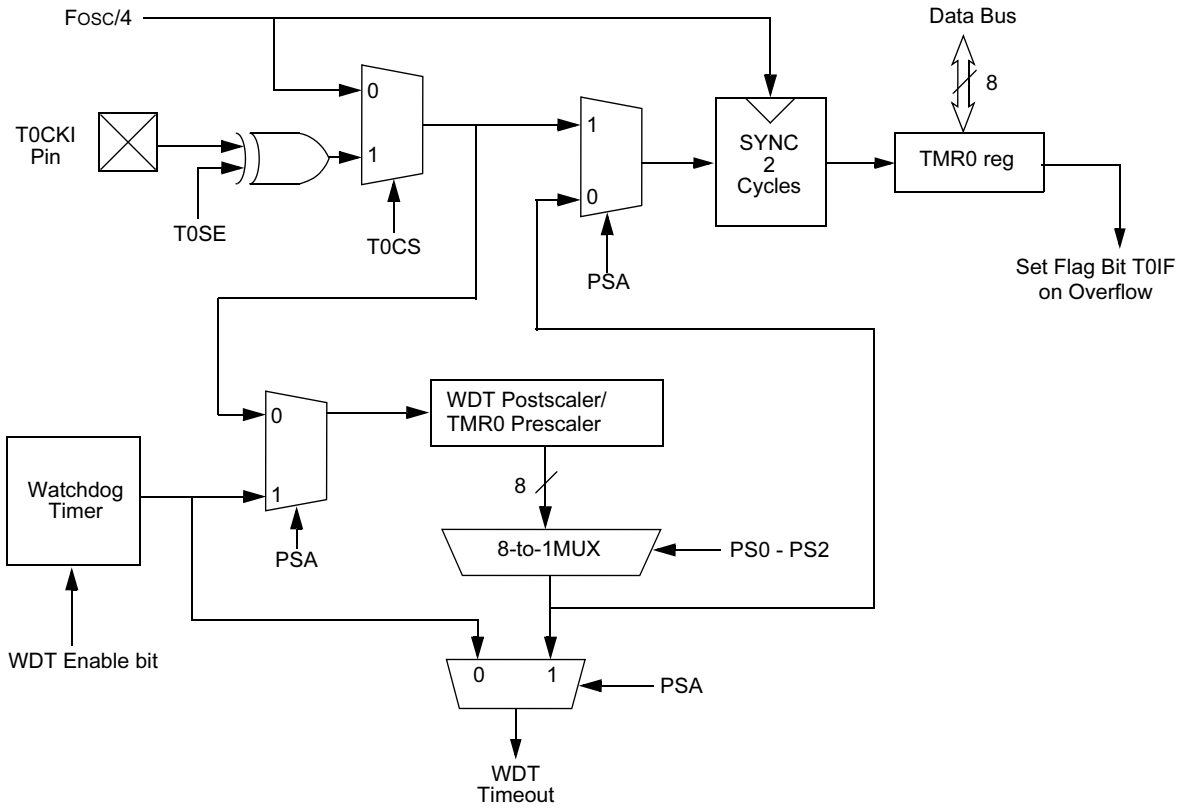
6.3 Timer0 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer. A prescaler assignment for the Timer0 module means that there is no postscaler for the Watchdog Timer, and vice-versa.

The PSA and PS2:PS0 bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF 1, MOVWF 1, BSF 1, x....etc.`) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

FIGURE 6-1: BLOCK DIAGRAM OF THE TIMER0/WDT



Note 1: T0SE, T0CS, PSA, PS0-PS2 are bits in the Option Register.

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<2:17> 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

Steps to follow when setting up an Asynchronous Reception:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH. (Section 12.1).
2. Enable the asynchronous serial port by clearing bit SYNC, and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.

TABLE 12-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

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
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for Asynchronous Reception.

12.5.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical except in the case of the SLEEP mode. Also, bit SREN is a don't care in Slave mode.

If receive is enabled, by setting bit CREN, prior to the SLEEP instruction, then a word may be received during SLEEP. On completely receiving the word, the RSR register will transfer the data to the RCREG register and if enable bit RCIE bit is set, the interrupt generated will wake the chip from SLEEP. If the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

Steps to follow when setting up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, then set enable bit RCIE.
3. If 9-bit reception is desired, then set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete and an interrupt will be generated, if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.

TABLE 12-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

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
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used for Synchronous Slave Transmission.

TABLE 12-12: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

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
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used for Synchronous Slave Reception.

14.2.6 INTERNAL 4 MHz OSCILLATOR

The internal RC oscillator provides a fixed 4 MHz (nominal) system clock at VDD = 5V and 25°C, see "Electrical Specifications" section for information on variation over voltage and temperature.

14.2.7 CLKOUT

The PIC16F62X can be configured to provide a clock out signal by programming the configuration word. The oscillator frequency, divided by 4 can be used for test purposes or to synchronize other logic.

14.3 Special Feature: Dual Speed Oscillator Modes

A software programmable Dual Speed Oscillator mode is provided when the PIC16F62X is configured in either ER or INTRC Oscillator modes. This feature allows users to dynamically toggle the oscillator speed between 4 MHz and 37 kHz. In ER mode, the 4 MHz setting will vary depending on the value of the external resistor. Also in ER mode, the 37 kHz operation is fixed and does not vary with resistor value. Applications that require low current power savings, but cannot tolerate putting the part into SLEEP, may use this mode.

The OSCF bit in the PCON register is used to control Dual Speed mode. See Section 3.2.2.6, Register 3-4.

14.4 RESET

The PIC16F62X differentiates between various kinds of RESET:

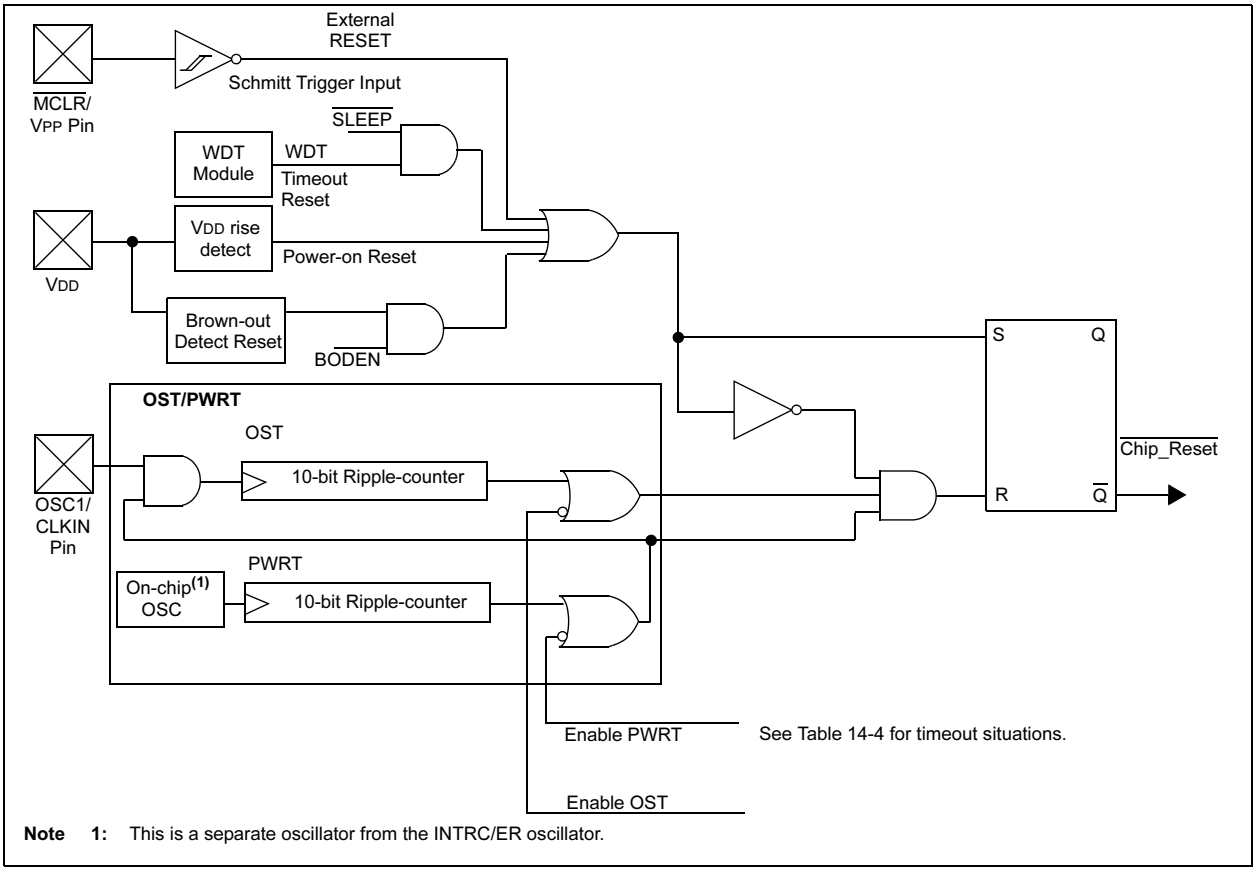
a) Power-on Reset (POR)
b) $\overline{MCLR}$ Reset during normal operation
c) $\overline{MCLR}$ Reset during SLEEP
d) WDT Reset (normal operation)
e) WDT Wake-up (SLEEP)
f) Brown-out Detect (BOD)

Some registers are not affected in any RESET condition; their status is unknown on POR and unchanged in any other RESET. Most other registers are reset to a "RESET state" on Power-on Reset, $\overline{MCLR}$ Reset, WDT Reset and $\overline{MCLR}$ Reset during SLEEP. They are not affected by a WDT Wake-up, since this is viewed as the resumption of normal operation. $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different RESET situations as indicated in Table 14-5. These bits are used in software to determine the nature of the RESET. See Table 14-8 for a full description of RESET states of all registers.

A simplified block diagram of the on-chip RESET circuit is shown in Figure 14-6.

The $\overline{MCLR}$ Reset path has a noise filter to detect and ignore small pulses. See Table 17-6 for pulse width specification.

FIGURE 14-6: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is a separate oscillator from the INTRC/ER oscillator.

15.1 Instruction Descriptions

ADDLW **Add Literal and W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → (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) → (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) → (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) → (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

IORLW **Inclusive OR Literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → (W)

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

Description: The contents of the W register is OR'ed with the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Example
```
IORLW   0x35
```
Before Instruction
W = 0x9A
After Instruction
W = 0xBF
Z = 0

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f,d

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

Operation: (W) .OR. (f) → (dest)

Status Affected: Z

Encoding:

00	0100	dfff	ffff

Description: Inclusive OR the W register with register 'f'. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Example
```
IORWF   REG1, 0
```
Before Instruction
REG1 = 0x13
W = 0x91
After Instruction
REG1 = 0x13
W = 0x93
Z = 1

MOVLW **Move Literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: k → (W)

Status Affected: None

Encoding:

11	00xx	kkkk	kkkk

Description: The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

Words: 1

Cycles: 1

Example
```
MOVLW   0x5A
```
After Instruction
W = 0x5A

MOVF **Move f**

Syntax: [*label*] MOVF f,d

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

Operation: (f) → (dest)

Status Affected: Z

Encoding:

00	1000	dfff	ffff

Description: The contents of register f is moved to a destination dependent upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.

Words: 1

Cycles: 1

Example
```
MOVF    REG1, 0
```
After Instruction
W = value in REG1 register
Z = 1

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f,d

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

Operation: (f) - (W) → (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

Example 1:
```
SUBWF   REG1, 1
```
Before Instruction
REG1 = 3
W = 2
C = ?
After Instruction
REG1 = 1
W = 2
C = 1; result is positive
Z = DC = 1

Example 2: Before Instruction
REG1 = 2
W = 2
C = ?
After Instruction
REG1 = 0
W = 2
C = 1; result is zero
Z = DC = 1

Example 3: 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>) → (dest<7:4>),
(f<7:4>) → (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

Example
```
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) → 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

Example

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

16.19 PICDEM 18R PIC18C601/801 Demonstration Board

The PICDEM 18R demonstration board serves to assist development of the PIC18C601/801 family of Microchip microcontrollers. It provides hardware implementation of both 8-bit Multiplexed/De-multiplexed and 16-bit Memory modes. The board includes 2 Mb external FLASH memory and 128 Kb SRAM memory, as well as serial EEPROM, allowing access to the wide range of memory types supported by the PIC18C601/801.

16.20 PICDEM LIN PIC16C43X Demonstration Board

The powerful LIN hardware and software kit includes a series of boards and three PICmicro microcontrollers. The small footprint PIC16C432 and PIC16C433 are used as slaves in the LIN communication and feature on-board LIN transceivers. A PIC16F874 FLASH microcontroller serves as the master. All three microcontrollers are programmed with firmware to provide LIN bus communication.

16.21 PICDEM USB PIC16C7X5 Demonstration Board

The PICDEM USB Demonstration Board shows off the capabilities of the PIC16C745 and PIC16C765 USB microcontrollers. This board provides the basis for future USB products.

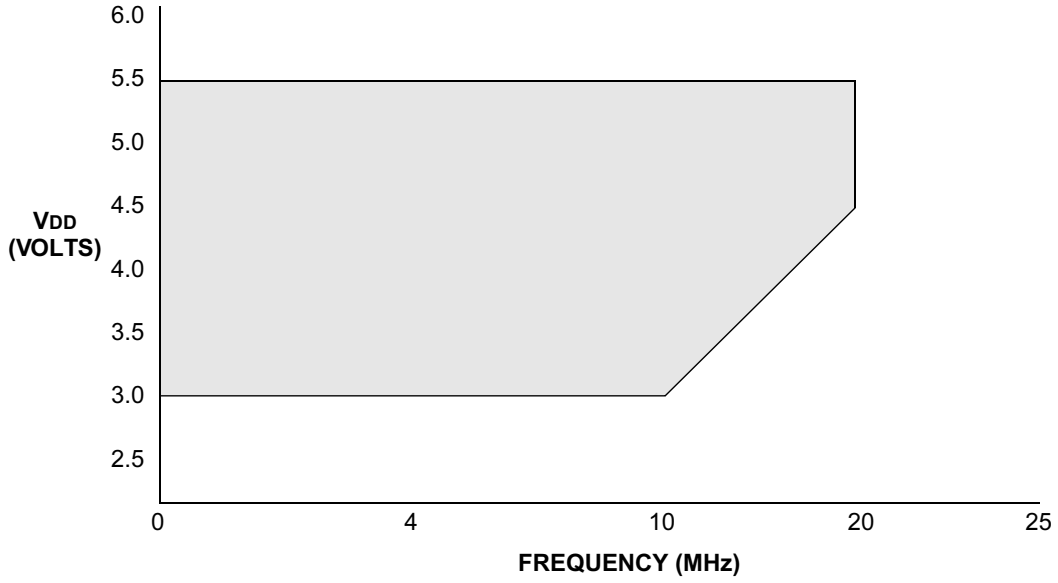
16.22 Evaluation and Programming Tools

In addition to the PICDEM series of circuits, Microchip has a line of evaluation kits and demonstration software for these products.

- KEELOQ evaluation and programming tools for Microchip's HCS Secure Data Products
- CAN developers kit for automotive network applications
- Analog design boards and filter design software
- PowerSmart battery charging evaluation/ calibration kits
- IrDA® development kit
- microID development and RFLab™ development software
- SEEVAL® designer kit for memory evaluation and endurance calculations
- PICDEM MSC demo boards for Switching mode power supply, high power IR driver, delta sigma ADC, and flow rate sensor

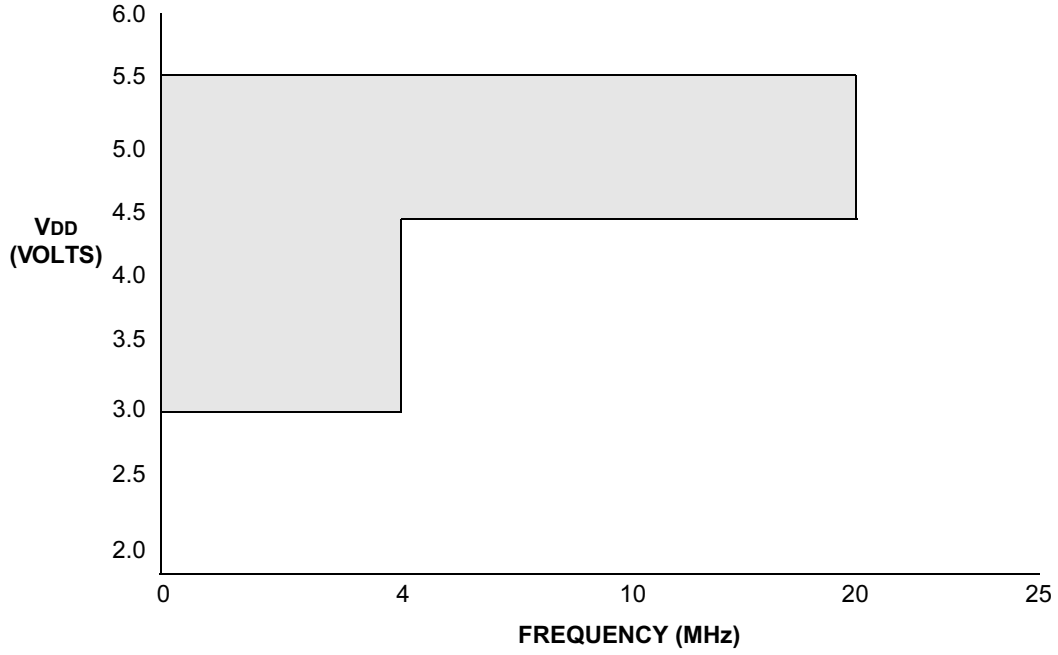
Check the Microchip web page and the latest Product Line Card for the complete list of demonstration and evaluation kits.

FIGURE 17-1: PIC16F62X VOLTAGE-FREQUENCY GRAPH, 0°C ≤ TA ≤ +70°C



Note 1: The shaded region indicates the permissible combinations of voltage and frequency.

FIGURE 17-2: PIC16F62X VOLTAGE-FREQUENCY GRAPH, -40°C ≤ TA < 0°C, +70°C < TA ≤ 85°C



Note 1: The shaded region indicates the permissible combinations of voltage and frequency.

TABLE 17-4: EXTERNAL CLOCK TIMING REQUIREMENTS

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	External CLKIN Frequency(1)	DC	—	4	MHz	XT and ER Osc mode, VDD = 5.0V
			DC	—	20	MHz	HS Osc mode
			DC	—	200	kHz	LP Osc mode
		Oscillator Frequency(1)		—	4	MHz	ER Osc mode, VDD = 5.0V
			0.1	—	4	MHz	XT Osc mode
			1	—	20	MHz	HS Osc mode
				—	200	kHz	LP Osc mode
			3.65	4	4.28	MHz	INTRC mode (fast), VDD = 5.0V
				37		kHz	INTRC mode (slow)
4	INTRC	Internal Calibrated RC	3.65	4.00	4.28	MHz	VDD = 5.0V
5	ER	External Biased ER Frequency	10 kHz		8 MHz		VDD = 5.0V
1	Tosc	External CLKIN Period(1)	250	—	—	ns	XT and ER Osc mode
			50	—	—	ns	HS Osc mode
			5	—	—	µs	LP Osc mode
		Oscillator Period(1)	250	—	—	ns	ER Osc mode
			250	—	10,000	ns	XT Osc mode
			50	—	1,000	ns	HS Osc mode
			5			µs	LP Osc mode
				250		ns	INTRC mode (fast)
				27		µs	INTRC mode (slow)
2	Tcy	Instruction Cycle Time	1.0	TCY	DC	ns	TCY = 4/FOSC
3	TosL, TosH	External CLKIN (OSC1) High External CLKIN Low	100 *	—	—	ns	XT oscillator, TOSC L/H duty cycle*

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (Tcy) equals four times the input oscillator time-based period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "Min." values with an external clock applied to the OSC1 pin. When an external clock input is used, the "Max" cycle time limit is "DC" (no clock) for all devices.

.

Note: The graphs and tables provided in this section are for design guidance and are not tested.

FIGURE 18-2: MAXIMUM IDD vs FOSC OVER VDD (HS MODE)

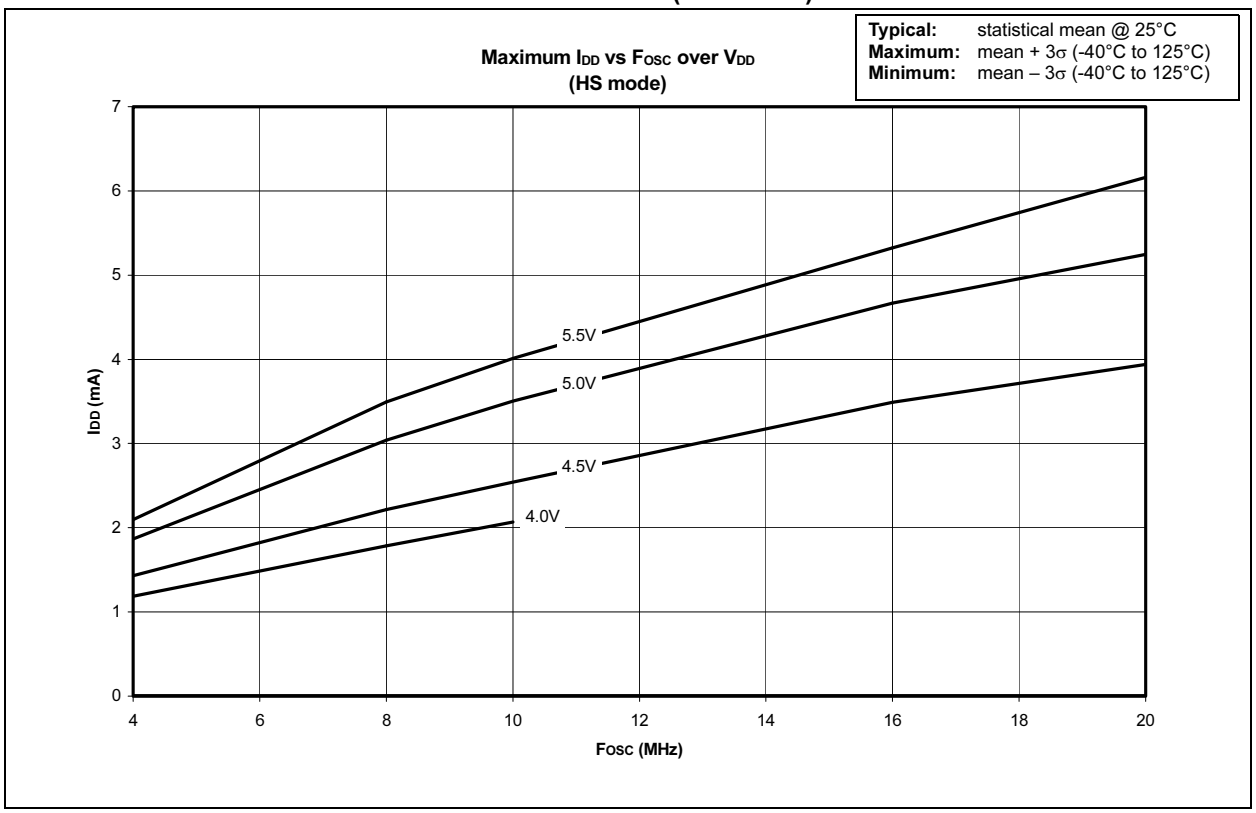
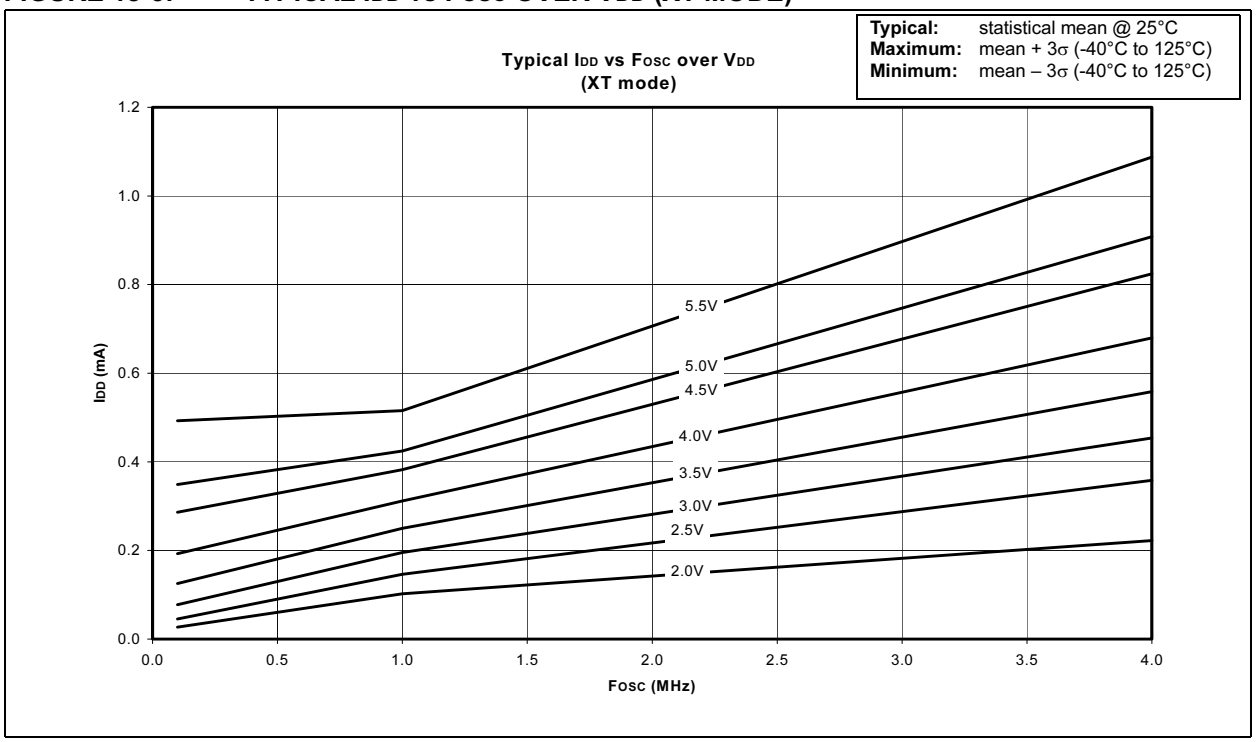


FIGURE 18-3: TYPICAL IDD vs FOSC OVER VDD (XT MODE)



.

Note: The graphs and tables provided in this section are for design guidance and are not tested.

FIGURE 18-4: TYPICAL IDD vs FOSC OVER VDD (XT MODE)

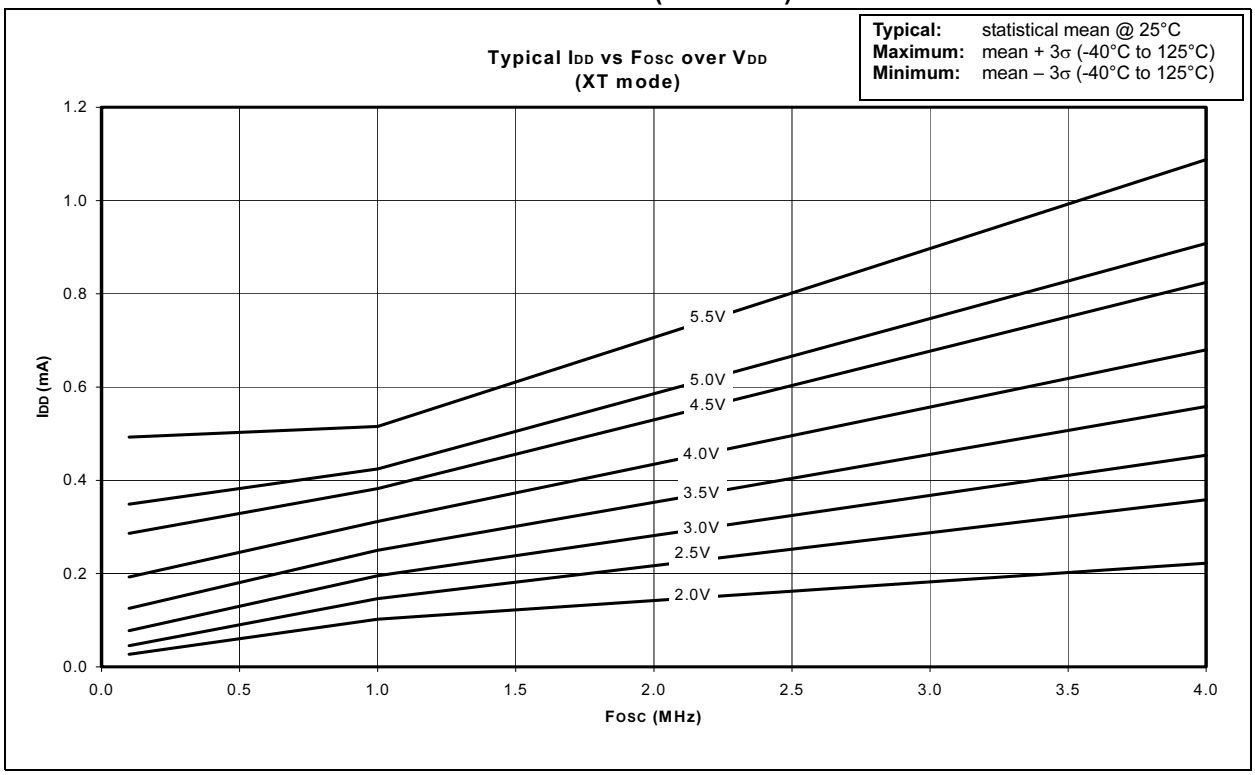
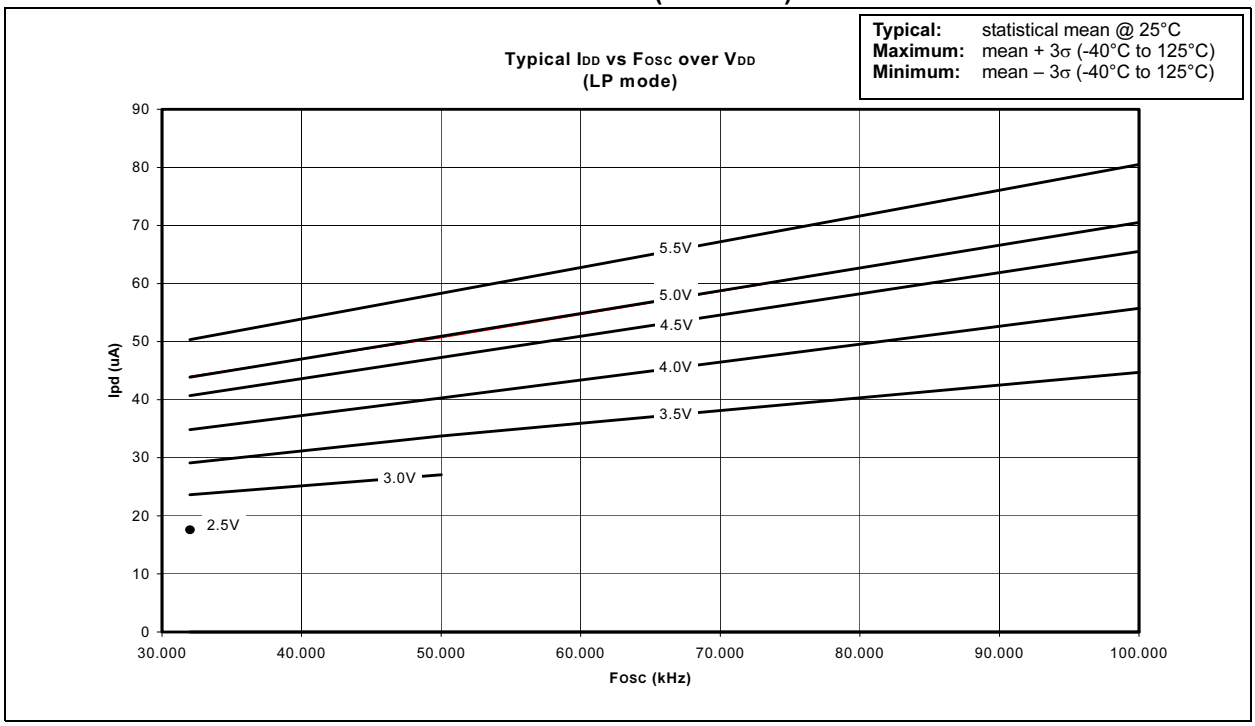


FIGURE 18-5: TYPICAL IDD vs FOSC OVER VDD (LP MODE)



Note: The graphs and tables provided in this section are for design guidance and are not tested.

FIGURE 18-26: MAXIMUM IDD vs VDD OVER TEMPERATURE (-40 TO +125°C) INTERNAL 4 MHz OSCILLATOR

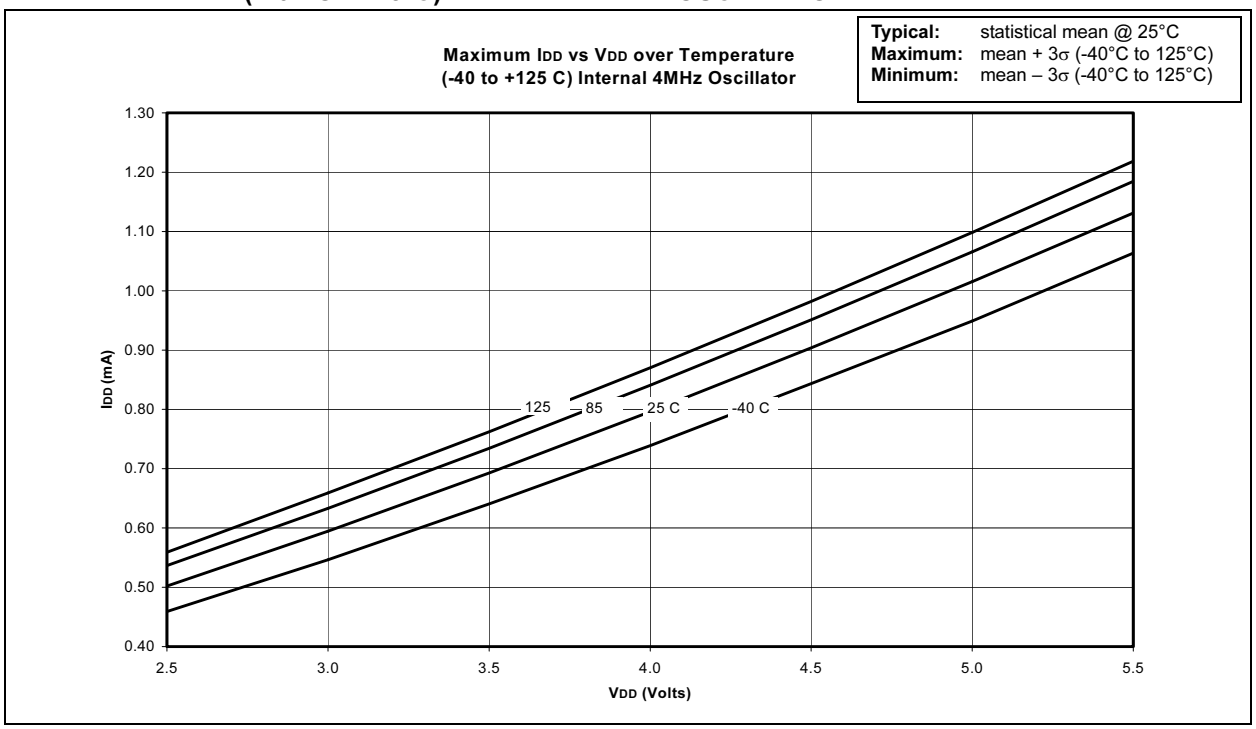
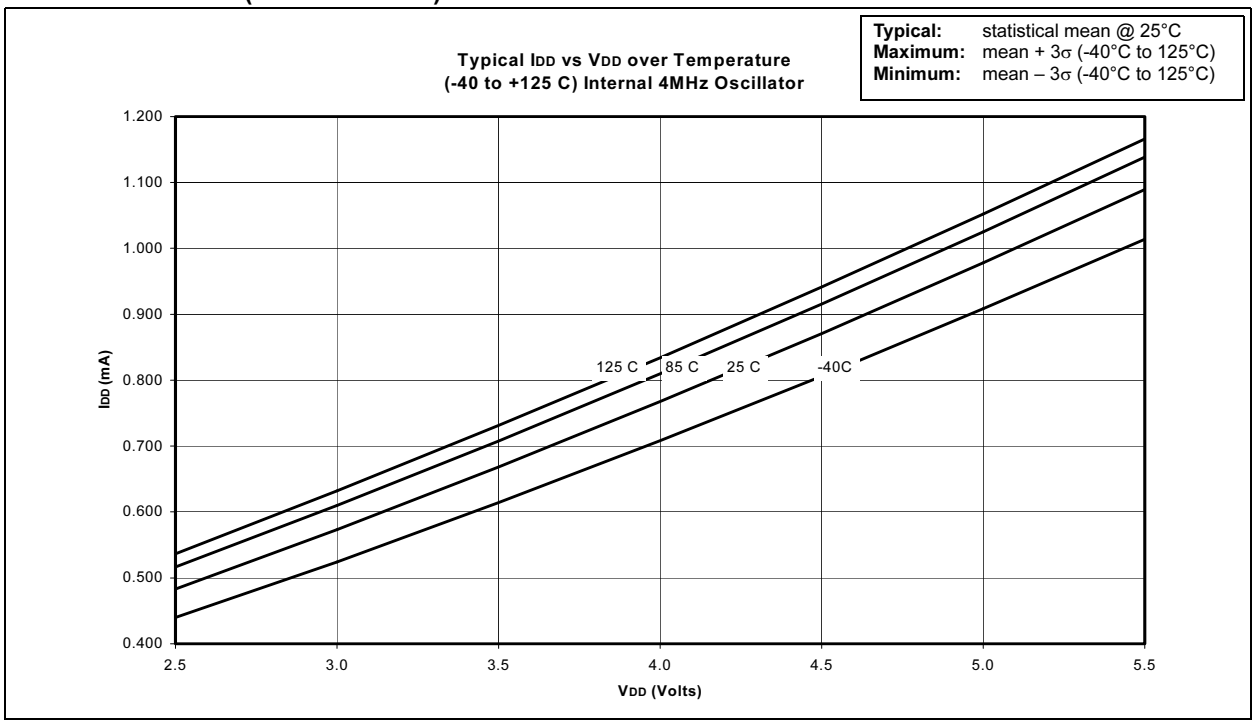


FIGURE 18-27: TYPICAL IDD vs VDD OVER TEMPERATURE (-40 TO +125°C) INTERNAL 4 MHz OSCILLATOR



PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	-XX	X	/XX	XXX
Device	Frequency Range	Temperature Range	Package	Pattern

Device	PIC16F62X:	Standard VDD range 3.0V to 5.5V
	PIC16F62XT	VDD range 3.0V to 5.5V (Tape and Reel)
	PIC16LF62X:	VDD range 2.0V to 5.5V
	PIC16LF62XT:	VDD range 2.0V to 5.5V (Tape and Reel)
Frequency Range	04 =	200 kHz (LP osc)
	04 =	4 MHz (XT and ER osc)
	20 =	20 MHz (HS osc)
Temperature Range	- =	0°C to +70°C
	I =	-40°C to +85°C
	E =	-40°C to +125°C
Package	P =	PDIP
	SO =	SOIC (Gull Wing, 300 mil body)
	SS =	SSOP (209 mil)
Pattern	3-Digit Pattern Code for QTP (blank otherwise).	

Examples:

a) PIC16F627 - 04/P 301 = Commercial Temp., PDIP package, 4 MHz, normal VDD limits, QTP pattern #301.

b) PIC16LF627 - 04I/SO = Industrial Temp., SOIC package, 200 kHz, extended VDD limits.

* JW Devices are UV erasable and can be programmed to any device configuration. JW Devices meet the electrical requirement of each oscillator type.

Sales and Support

Data Sheets
Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 792-7277
3. The Microchip Worldwide Site (www.microchip.com)

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

New Customer Notification System
Register on our web site (www.microchip.com/cn) to receive the most current information on our products.