Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	8MHz
Connectivity	-
Peripherals	POR, WDT
Number of I/O	5
Program Memory Size	1.5KB (1K x 12)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	41 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	8-TSSOP, 8-MSOP (0.118", 3.00mm Width)
Supplier Device Package	8-MSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic12f519t-i-ms

Table of Contents

1.0 General Description 5
2.0 PIC12F519 Device Varieties 7
3.0 Architectural Overview 9
4.0 Memory Organization 13
5.0 Flash Data Memory 21
6.0 I/O Port 23
7.0 Timer0 Module and TMR0 Register 31
8.0 Special Features Of The CPU 37
9.0 Instruction Set Summary 49
10.0 Development Support 57
11.0 Electrical Characteristics 61
12.0 DC and AC Characteristics Graphs and Charts 73
13.0 Packaging Information 83
Index 89
The Microchip Web Site 91
Customer Change Notification Service 91
Customer Support 91
Reader Response 92
Product Identification System 93

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN pin) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the PC is incremented every Q1 and the instruction is fetched from program memory and latched into the instruction register in Q4. It is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2 and Example 3-1.

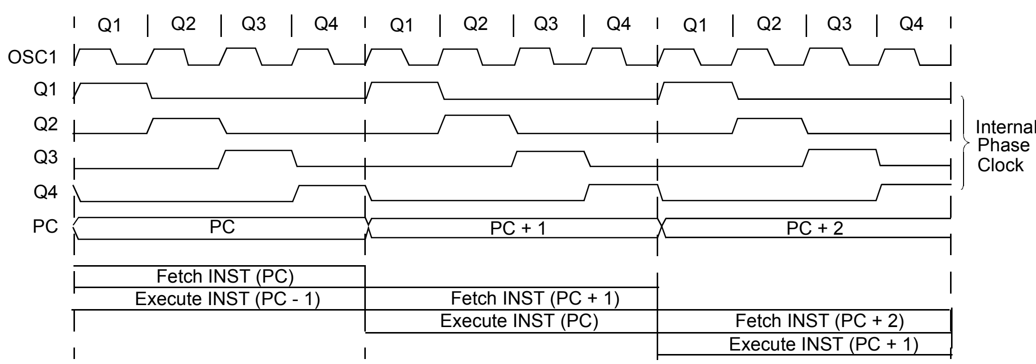
3.2 Instruction Flow/Pipelining

An instruction cycle consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle, while decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the PC to change (e.g., `GOTO`), then two cycles are required to complete the instruction (Example 3-1).

A fetch cycle begins with the PC incrementing in Q1.

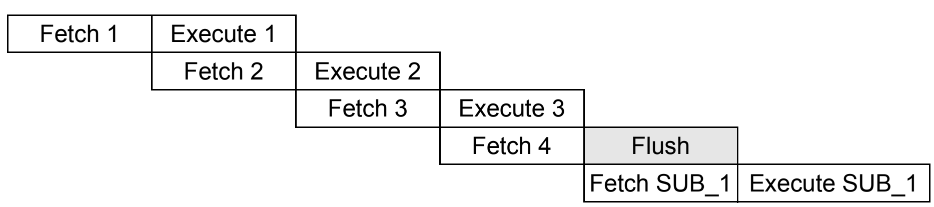
In the execution cycle, the fetched instruction is latched into the Instruction Register (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3 and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW

```
1. MOVLW 03H
2. MOVWF GPIO
3. CALL  SUB_1
4. BSF   GPIO, 1
```



All instructions are single cycle, except for any program branches. These take two cycles, since the fetch instruction is “flushed” from the pipeline, while the new instruction is being fetched and then executed.

4.3 STATUS register

This register contains the arithmetic status of the ALU, the Reset status and the page preselect bit.

The STATUS register can be the destination for any instruction, as with 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 $\overline{TO}$ and $\overline{PD}$ bits are not 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 '`000u u1uu`' (where u = unchanged).

Therefore, it is recommended that only `BCF`, `BSF` and `MOVWF` instructions be used to alter the STATUS register. These instructions do not affect the Z, DC or C bits from the STATUS register. For other instructions which do affect Status bits, see **Section 9.0 "Instruction Set Summary"**.

REGISTER 4-1: STATUS: STATUS REGISTER

R/W-0	U-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
GPWUF	—	PA0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
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 **GPWUF**: Wake-up From Sleep on Pin Change bit
`1` = Reset due to wake-up from Sleep on pin change
`0` = After power-up or other Reset

bit 6 **Unimplemented:** Read as '`0`'

bit 5 **PA0**: Program Page Preselect bit
`1` = Page 1 (000h-1FFh)
`0` = Page 0 (200h-3FFh)

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

bit 3 $\overline{\textbf{PD}}$: 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 (for `ADDWF` and `SUBWF` instructions)
`ADDWF:`
`1` = A carry from the 4th low-order bit of the result occurred
`0` = A carry from the 4th low-order bit of the result did not occur
`SUBWF:`
`1` = A borrow from the 4th low-order bit of the result did not occur
`0` = A borrow from the 4th low-order bit of the result occurred

bit 0 **C**: Carry/$\overline{\text{borrow}}$ bit (for `ADDWF`, `SUBWF` and `RRF`, `RLF` instructions)

`ADDWF:`	`SUBWF:`	`RRF or RLF:`
`1` = A carry occurred	`1` = A borrow did not occur	Load bit with LSb or MSb, respectively
`0` = A carry did not occur	`0` = A borrow occurred	

4.6 Program Counter

As a program instruction is executed, the Program Counter (PC) will contain the address of the next program instruction to be executed. The PC value is increased by one every instruction cycle, unless an instruction changes the PC.

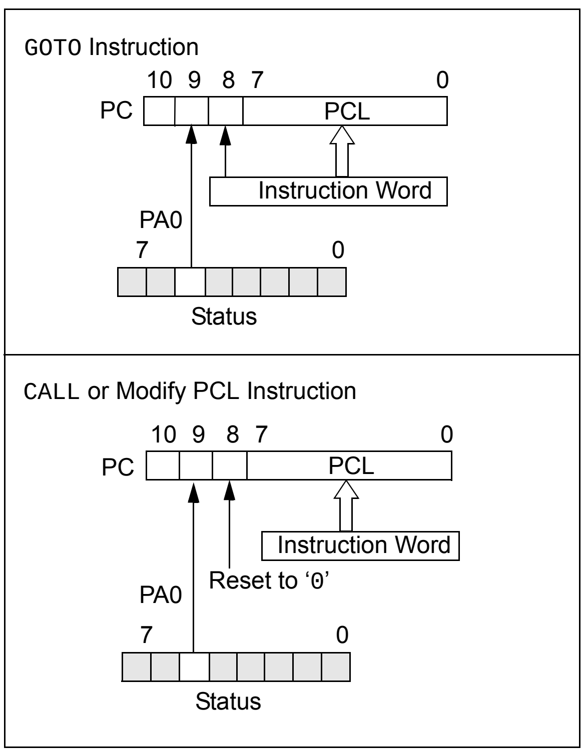
For a `GOTO` instruction, bits <8:0> of the PC are provided by the `GOTO` instruction word. The Program Counter (PCL) is mapped to PC<7:0>. Bit 5 of the STATUS register provides page information to bit 9 of the PC (Figure 4-3).

For a `CALL` instruction, or any instruction where the PCL is the destination, bits <7:0> of the PC again are provided by the instruction word. However, PC<8> does not come from the instruction word, but is always cleared (Figure 4-3).

Instructions where the PCL is the destination, or modify PCL instructions, include `MOVWF PCL`, `ADDWF PCL` and `BSF PCL,5`.

Note:	Because PC<8> is cleared in the `CALL` instruction or any modify PCL instruction, all subroutine calls or computed jumps are limited to the first 256 locations of any program memory page (512 words long).

FIGURE 4-3: LOADING OF PC BRANCH INSTRUCTIONS



4.6.1 EFFECTS OF RESET

The PC is set upon a Reset, which means that the PC addresses the last location in the last page (i.e., the oscillator calibration instruction). After executing `MOVLW XX`, the PC will roll over to location 00h and begin executing user code.

The STATUS register page preselect bits are cleared upon a Reset, which means that page 0 is pre-selected.

Therefore, upon a Reset, a `GOTO` instruction will automatically cause the program to jump to page 0 until the value of the page bits is altered.

4.7 Stack

The PIC12F519 device has a two-deep, 12-bit wide hardware PUSH/POP stack.

A `CALL` instruction will PUSH the current value of Stack 1 into Stack 2 and then PUSH the current PC value, incremented by one, into Stack Level 1. If more than two sequential `CALL`s are executed, only the most recent two return addresses are stored.

A `RETLW` instruction will POP the contents of Stack Level 1 into the PC and then copy Stack Level 2 contents into Stack Level 1. If more than two sequential `RETLW`s are executed, the stack will be filled with the address previously stored in Stack Level 2. Note that the W register will be loaded with the literal value specified in the instruction. This is particularly useful for the implementation of data look-up tables within the program memory.

Note 1:	There are no Status bits to indicate stack overflows or stack underflow conditions.
2:	There are no instruction mnemonics called PUSH or POP. These are actions that occur from the execution of the `CALL` and `RETLW` instructions.

REGISTER 6-1: GPIO: GPIO REGISTER

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	GP5	GP4	GP3	GP2	GP1	GP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **GP<5:0>**: GPIO I/O Pin bits
1 = GPIO pin is >V_{IH} min.
0 = GPIO pin is <V_{IL} max.

REGISTER 6-2: TRISGPIO: TRI-STATE GPIO REGISTER

U-0	U-0	W-1	W-1	W-1	W-1	W-1	W-1
—	—	TRISGPIO5	TRISGPIO4	TRISGPIO3	TRISGPIO2	TRISGPIO1	TRISGPIO0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **TRISGPIO<5:0>:** GPIO Tri-State Control bits
1 = GPIO pin configured as an input (tri-stated)
0 = GPIO pin configured as an output

7.0 TIMER0 MODULE AND TMR0 REGISTER

The Timer0 module has the following features:

- 8-bit timer/counter register, TMR0
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select:
  - Edge select for external clock

Figure 7-1 is a simplified block diagram of the Timer0 module.

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If TMR0 register is written, the increment is inhibited for the following two cycles (Figure 7-2 and Figure 7-3). The user can work around this by writing an adjusted value to the TMR0 register.

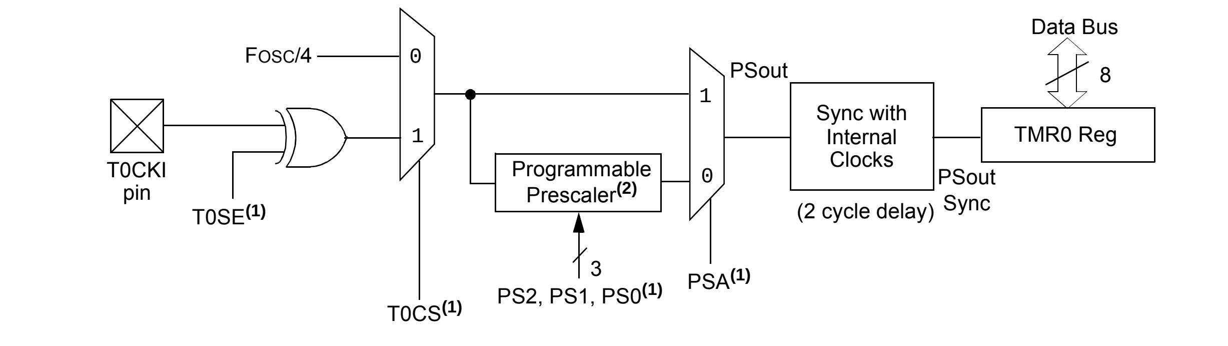
Counter mode is selected by setting the T0CS bit (OPTION<5>). In this mode, Timer0 will increment either on every rising or falling edge of pin T0CKI. The T0SE bit (OPTION<4>) determines the source edge. Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed in detail in **Section 7.1 "Using Timer0 with an External Clock"**.

The prescaler may be used by either the Timer0 module or the Watchdog Timer, but not both. The prescaler assignment is controlled in software by the control bit, PSA (OPTION<3>). Clearing the PSA bit will assign the prescaler to Timer0. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4,..., 1:256 are selectable. **Section 7.2 "Prescaler"** details the operation of the prescaler.

A summary of registers associated with the Timer0 module is found in Table 7-1.

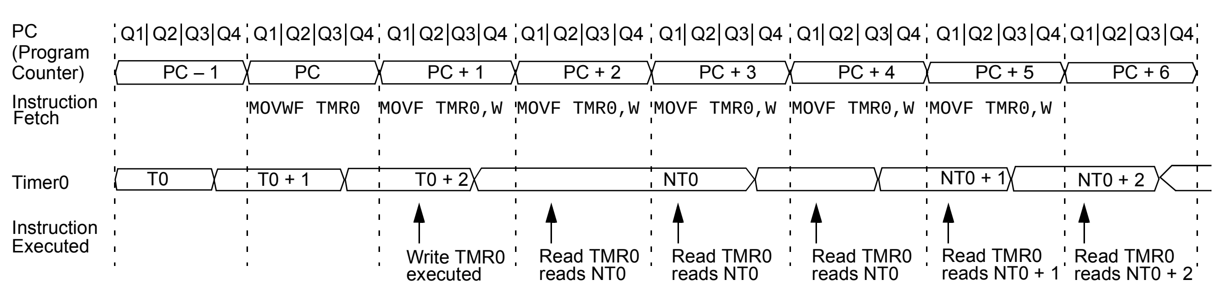
The Timer0 contained in the CPU core follows the standard baseline definition.

FIGURE 7-1: TIMER0 BLOCK DIAGRAM



Note 1: Bits T0CS, T0SE, PSA, PS2, PS1 and PS0 are located in the OPTION register.
2: The prescaler is shared with the Watchdog Timer.

FIGURE 7-2: TIMER0 TIMING: INTERNAL CLOCK/NO PRESCALE



7.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module or as a postscaler for the Watchdog Timer (WDT), respectively (see **Section 8.6 "Watchdog Timer (WDT)"**). For simplicity, this counter is being referred to as "prescaler" throughout this data sheet.

Note:	The prescaler may be used by either the Timer0 module or the WDT, but not both. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the WDT and vice versa.

The PSA and PS<2:0> bits (OPTION<3:0>) determine prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0`, `MOVWF TMR0`, etc.) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the WDT. The prescaler is neither readable nor writable. On a Reset, the prescaler contains all '0's.

7.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed "on-the-fly" during program execution). To avoid an unintended device Reset, the following instruction sequence (Example 7-1) must be executed when changing the prescaler assignment from Timer0 to the WDT.

EXAMPLE 7-1: CHANGING PRESCALER (TIMER0 → WDT)

```
CLRWDT                ;Clear WDT
CLRF   TMR0           ;Clear TMR0 and Prescaler
MOVLW  b'00xx1111'
OPTION

CLRWDT                ;PS<2:0> are 000 or 001
MOVLW  b'00xx1xxx'    ;Set Postscaler to
OPTION                ;desired WDT rate
```

To change the prescaler from the WDT to the Timer0 module, use the sequence shown in Example 7-2. This sequence must be used even if the WDT is disabled. A `CLRWDT` instruction should be executed before switching the prescaler.

EXAMPLE 7-2: CHANGING PRESCALER (WDT → TIMER0)

```
CLRWDT                ;Clear WDT and
                      ;prescaler
MOVLW  b'xxxx0xxx'    ;Select TMR0, new
                      ;prescale value and
                      ;clock source
OPTION
```

PC will then roll over to the users program at address 0x000. The user then has the option of writing the value to the OSCCAL Register (05h) or ignoring it.

OSCCAL, when written to with the calibration value, will "trim" the internal oscillator to remove process variation from the oscillator frequency.

Note:	Erasing the device will also erase the pre-programmed internal calibration value for the internal oscillator. The calibration value must be read prior to erasing the part so it can be reprogrammed correctly later.

For the PIC12F519 device, only bits <7:1> of OSCCAL are used for calibration. See Register 4-3 for more information.

Note:	The bit 0 of the OSCCAL register is unimplemented and should be written as '0' when modifying OSCCAL for compatibility with future devices.

8.3 Reset

The device differentiates between various kinds of Reset:

- Power-on Reset (POR)
- $\overline{\text{MCLR}}$ Reset during normal operation
- $\overline{\text{MCLR}}$ Reset during Sleep
- WDT Time-out Reset during normal operation
- WDT Time-out Reset during Sleep
- Wake-up from Sleep on pin change

Some registers are not reset in any way, and they are unknown on Power-on Reset (POR) and unchanged in any other Reset. Most other registers are reset to "Reset state" on Power-on Reset (POR), $\overline{\text{MCLR}}$, WDT or Wake-up on pin change Reset during normal operation. They are not affected by a WDT Reset during Sleep or $\overline{\text{MCLR}}$ Reset during Sleep, since these Resets are viewed as resumption of normal operation. The exceptions to this are $\overline{\text{TO}}$, $\overline{\text{PD}}$ and GPWUF bits. They are set or cleared differently in different Reset situations. These bits are used in software to determine the nature of Reset. See Table 8-3 for a full description of Reset states of all registers.

TABLE 8-3: RESET CONDITIONS FOR REGISTERS

Register	Address	Power-on Reset	$\overline{\text{MCLR}}$ Reset, WDT Time-out, Wake-up On Pin Change
W	—	qqqq qqq0[1]	qqqq qqq0[1]
INDF	00h	xxxx xxxx	uuuu uuuu
TMR0	01h	xxxx xxxx	uuuu uuuu
PCL	02h	1111 1111	1111 1111
STATUS	03h	0-01 1xxx	q-0q quuu[2], [3]
FSR	04h	110x xxxx	11uu uuuu
OSCCAL	05h	1111 111-	uuuu uuu-
PORTB	06h	--xx xxxx	--uu uuuu
OPTION	—	1111 1111	1111 1111
TRIS	—	--11 1111	--11 1111
EECON	21h	---0 x000	---0 q000
EEDATA	25h	xxxx xxxx	uuuu uuuu
EEADR	26h	--xx xxxx	--uu uuuu

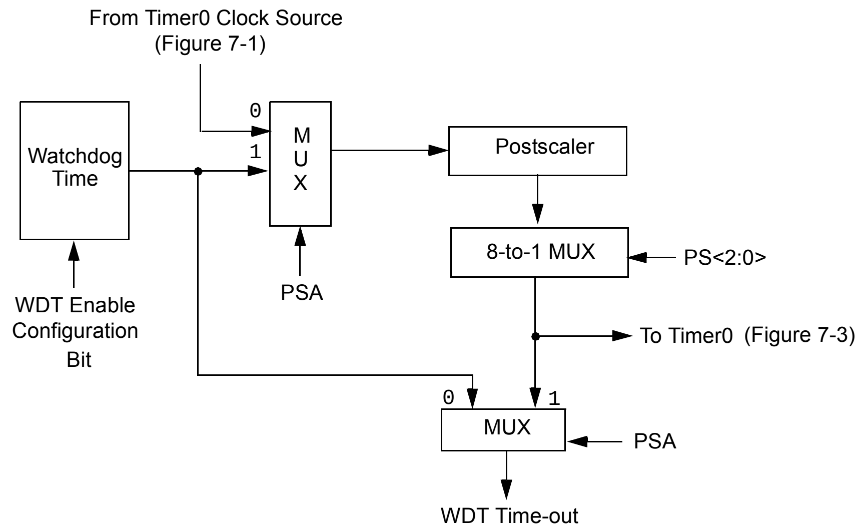
Legend: u = unchanged, x = unknown, – = unimplemented bit, read as '0', q = value depends on condition.

Note 1: Bits <7:1> of W register contain oscillator calibration values due to MOVLW XX instruction at top of memory.

2: See Table 8-4 for Reset value for specific conditions.

3: If Reset was due to wake-up on pin change, then bit 7 = 1. All other Resets will cause bit 7 = 0.

FIGURE 8-11: WATCHDOG TIMER BLOCK DIAGRAM



Note 1: PSA, PS<2:0> are bits in the OPTION register.

TABLE 8-6: SUMMARY OF REGISTER ASSOCIATED WITH THE WATCHDOG TIMER

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
OPTION	$\overline{GPWU}$	$\overline{GPPU}$	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: Shaded boxes = Not used by Watchdog Timer.

8.7 Time-out Sequence, Power-down and Wake-up from Sleep Status Bits (TO, PD, GPWUF)

The $\overline{\text{TO}}$, $\overline{\text{PD}}$ and (GPWUF) bits in the STATUS register can be tested to determine if a Reset condition has been caused by a power-up condition, a $\overline{\text{MCLR}}$ or Watchdog Timer (WDT) Reset.

TABLE 8-7: $\overline{\text{TO}}$/$\overline{\text{PD}}$/(GPWUF) STATUS AFTER RESET

GPWUF	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Reset Caused By
0	0	0	WDT wake-up from Sleep
0	0	u	WDT time-out (not from Sleep)
0	1	0	$\overline{\text{MCLR}}$ wake-up from Sleep
0	1	1	Power-up
0	u	u	$\overline{\text{MCLR}}$ not during Sleep
1	1	0	Wake-up from Sleep on pin change

Legend: u = unchanged

Note 1: The $\overline{\text{TO}}$, $\overline{\text{PD}}$ and GPWUF bits maintain their status (u) until a Reset occurs. A low-pulse on the $\overline{\text{MCLR}}$ input does not change the $\overline{\text{TO}}$, $\overline{\text{PD}}$ and GPWUF Status bits.

8.8 Power-down Mode (Sleep)

A device may be powered down (Sleep) and later powered up (wake-up from Sleep).

8.8.1 SLEEP

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

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{\text{TO}}$ bit (STATUS<4>) is set, the $\overline{\text{PD}}$ bit (STATUS<3>) is cleared and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, driving low or high-impedance).

Note:	A Reset generated by a WDT time-out does not drive the $\overline{\text{MCLR}}$ pin low.

For lowest current consumption while powered down, the T0CKI input should be at V_{DD} or V_{SS} and the GP3/$\overline{\text{MCLR}}$/V_{PP} pin must be at a logic high level if $\overline{\text{MCLR}}$ is enabled.

8.8.2 WAKE-UP FROM SLEEP

The device can wake-up from Sleep through one of the following events:

5. An external Reset input on GP3/$\overline{\text{MCLR}}$/V_{PP} pin, when configured as $\overline{\text{MCLR}}$.
6. A Watchdog Timer Time-out Reset (if WDT was enabled).
7. A change on input pin GP0, GP1 and GP3 when wake-up on change is enabled.

These events cause a device Reset. The $\overline{\text{TO}}$, $\overline{\text{PD}}$ and GPWUF bits can be used to determine the cause of device Reset. The $\overline{\text{TO}}$ bit is cleared if a WDT time-out occurred (and caused wake-up). The $\overline{\text{PD}}$ bit, which is set on power-up, is cleared when SLEEP is invoked. The GPWUF bit indicates a change in state while in Sleep at pins GP0, GP1 and GP3 (since the last file or bit operation on GPIO port).

Note:	**Caution:** Right before entering Sleep, read the input pins. When in Sleep, wake up occurs when the values at the pins change from the state they were in at the last reading. If a wake-up on change occurs and the pins are not read before re-entering Sleep, a wake-up will occur immediately even if no pins change while in Sleep mode.

The WDT is cleared when the device wakes from Sleep, regardless of the wake-up source.

IORWF	Inclusive OR W with f
Syntax:	[*label*] IORWF f,d
Operands:	0 ≤ f ≤ 31 d ∈ [0,1]
Operation:	(W).OR. (f) → (dest)
Status Affected:	Z
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’.

MOVWF	Move W to f
Syntax:	[*label*] MOVWF f
Operands:	0 ≤ f ≤ 31
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from the W register to register ‘f’.

MOVF	Move f
Syntax:	[*label*] MOVF f,d
Operands:	0 ≤ f ≤ 31 d ∈ [0,1]
Operation:	(f) → (dest)
Status Affected:	Z
Description:	The contents of register ‘f’ are moved to destination ‘d’. If ‘d’ is ‘0’, destination is the W register. If ‘d’ is ‘1’, the destination is file register ‘f’. ‘d’ = 1 is useful as a test of a file register, since status flag Z is affected.

NOP	No Operation
Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.

MOVLW	Move Literal to W
Syntax:	[*label*] MOVLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W)
Status Affected:	None
Description:	The eight-bit literal ‘k’ is loaded into the W register. The “don’t cares” will assembled as ‘0’s.

OPTION	Load OPTION Register
Syntax:	[*label*] Option
Operands:	None
Operation:	(W) → Option
Status Affected:	None
Description:	The content of the W register is loaded into the OPTION register.

RETLW	Return with Literal in W
Syntax:	[*label*] RETLW k
Operands:	$0 \le k \le 255$
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the eight-bit literal ‘k’. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.

RLF	Rotate Left f through Carry
Syntax:	[*label*] RLF f,d
Operands:	$0 \le f \le 31$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Description:	The contents of register ‘f’ are rotated one bit to the left through the Carry flag. If ‘d’ is ‘0’, the result is placed in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

RRF	Rotate Right f through Carry
Syntax:	[*label*] RRF f,d
Operands:	$0 \le f \le 31$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Description:	The contents of register ‘f’ are rotated one bit to the right through the Carry flag. If ‘d’ is ‘0’, the result is placed in the W register. If ‘d’ is ‘1’, the result is placed back in register ‘f’.

SLEEP	Enter SLEEP Mode
Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT; 0 → WDT prescaler; 1 → $\overline{TO}$; 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$, GPWUF
Description:	Time-out Status bit ($\overline{TO}$) is set. The Power-down Status bit ($\overline{PD}$) is cleared. GPWUF is unaffected. The WDT and its prescaler are cleared. The processor is put into Sleep mode with the oscillator stopped. See **Section 8.8 “Power-down Mode (Sleep)”** on Sleep for more details.

SUBWF	Subtract W from f
Syntax:	[*label*] SUBWF f,d
Operands:	$0 \le f \le 31$ $d \in [0,1]$
Operation:	(f) – (W) → (dest)
Status Affected:	C, DC, Z
Description:	Subtract (two’s complement method) the 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’.

SWAPF	Swap Nibbles in f
Syntax:	[*label*] SWAPF f,d
Operands:	$0 \le f \le 31$ $d \in [0,1]$
Operation:	(f<3:0>) → (dest<7:4>); (f<7:4>) → (dest<3:0>)
Status Affected:	None
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’.

10.2 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for all PIC MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

10.3 MPLAB C18 and MPLAB C30 C Compilers

The MPLAB C18 and MPLAB C30 Code Development Systems are complete ANSI C compilers for Microchip's PIC18 and PIC24 families of microcontrollers and the dsPIC30 and dsPIC33 family of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

10.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

10.5 MPLAB ASM30 Assembler, Linker and Librarian

MPLAB ASM30 Assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

10.6 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC® DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB C18 and MPLAB C30 C Compilers, and the MPASM and MPLAB ASM30 Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

11.3 AC Characteristics

TABLE 11-5: EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions (unless otherwise specified) Operating Temperature -40°C ≤ TA ≤ +85°C (industrial), -40°C ≤ TA ≤ +125°C (extended) Operating Voltage VDD range is described in **Section 11.0 "Electrical Characteristics"**				
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ(1)**	**Max.**	**Units**	**Conditions**
1A	FOSC	External CLKIN Frequency(2)	DC	—	4	MHz	XT Oscillator mode
			DC	—	200	kHz	LP Oscillator mode
		Oscillator Frequency(2)	DC	—	4	MHz	EXTRC Oscillator mode
			0.1	—	4	MHz	XT Oscillator mode
			DC	—	200	kHz	LP Oscillator mode
1	TOSC	External CLKIN Period(2)	250	—	—	ns	XT Oscillator mode
			5	—	—	μs	LP Oscillator mode
		Oscillator Period(2)	250	—	—	ns	EXTRC Oscillator mode
			250	—	10,000	ns	XT Oscillator mode
			5	—	—	μs	LP Oscillator mode
2	TCY	Instruction Cycle Time	200	4/FOSC	DC	ns	
3	TosL, TosH	Clock in (OSC1) Low or High Time	50*	—	—	ns	XT Oscillator
			2*	—	—	μs	LP Oscillator
4	TosR, TosF	Clock in (OSC1) Rise or Fall Time	—	—	25*	ns	XT Oscillator
			—	—	50*	ns	LP Oscillator

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

2: 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. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

FIGURE 12-4: TYPICAL IPD vs. VDD (SLEEP MODE, ALL PERIPHERALS DISABLED)

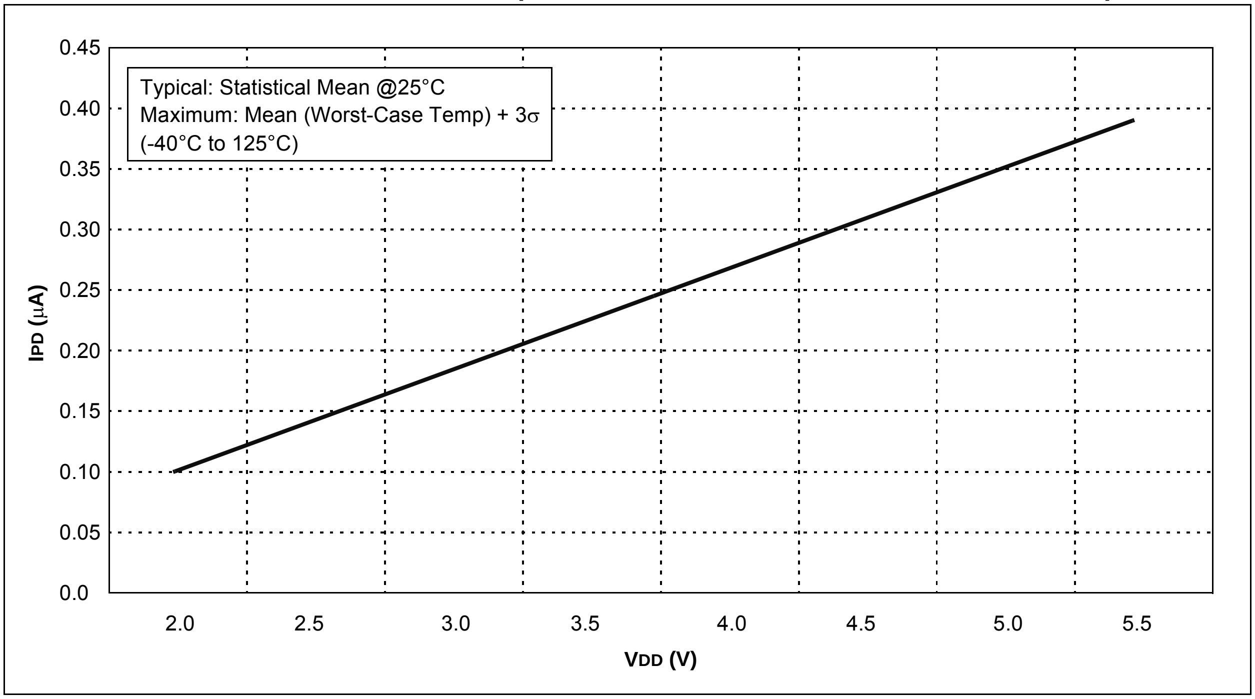


FIGURE 12-5: MAXIMUM IPD vs. VDD (SLEEP MODE, ALL PERIPHERALS DISABLED)

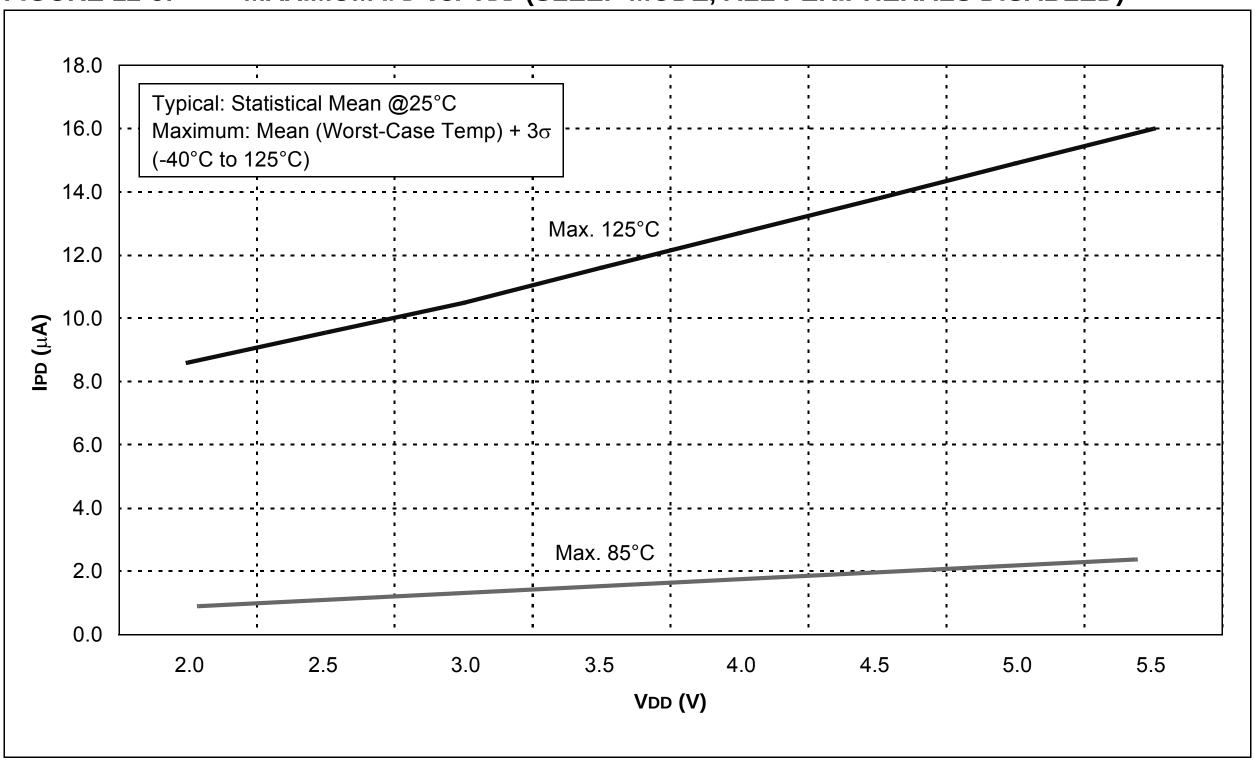


FIGURE 12-8: WDT TIME-OUT vs. VDD OVER TEMPERATURE (NO PRESCALER)

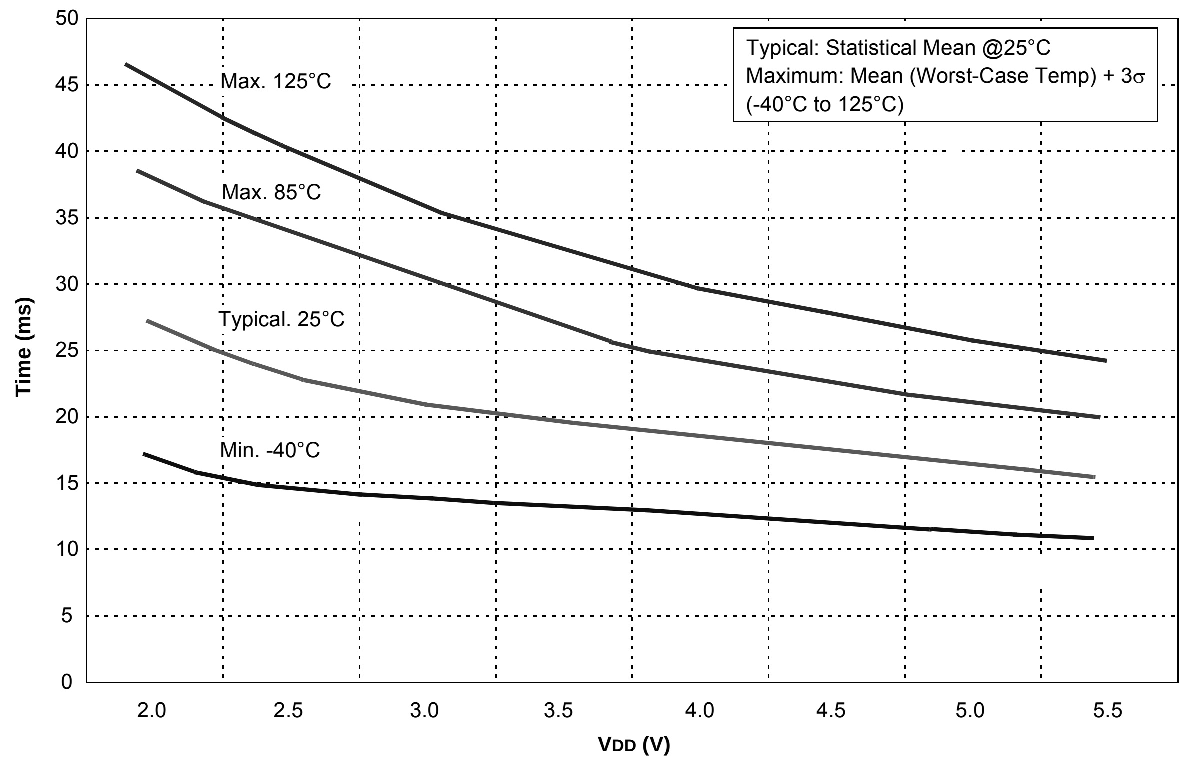


FIGURE 12-9: VOL vs. IOL OVER TEMPERATURE (VDD = 3.0V)

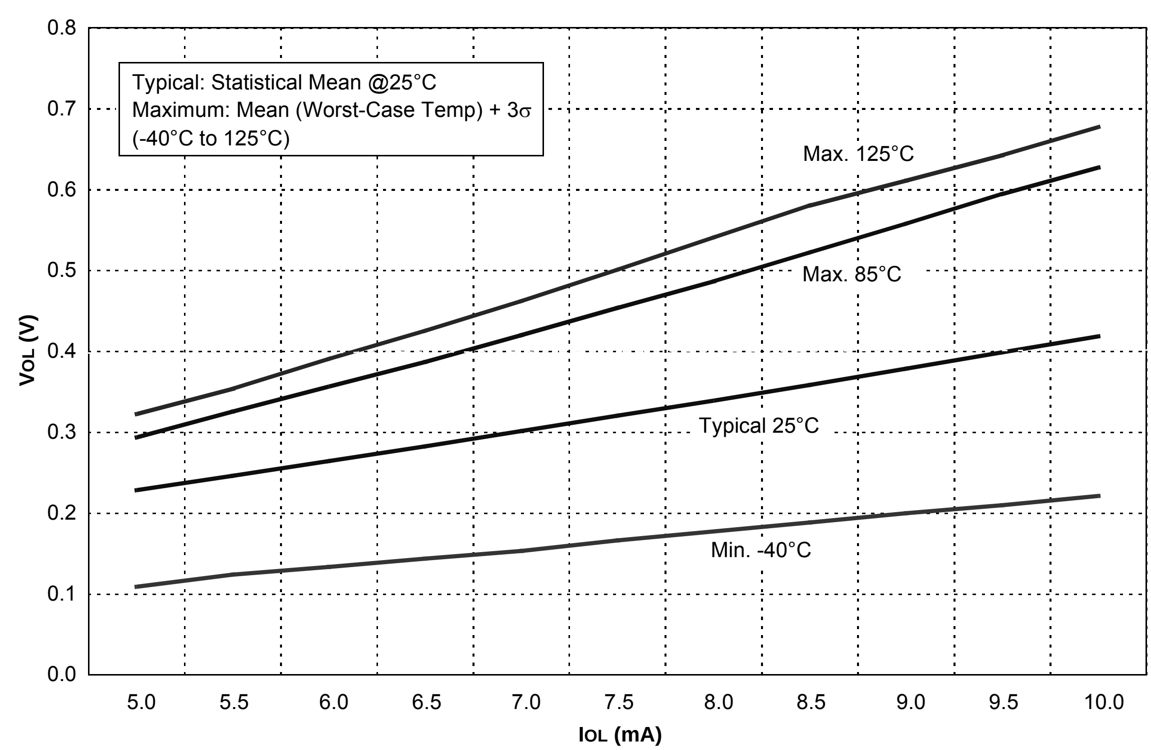


FIGURE 12-14: SCHMITT TRIGGER INPUT THRESHOLD VIN vs. VDD

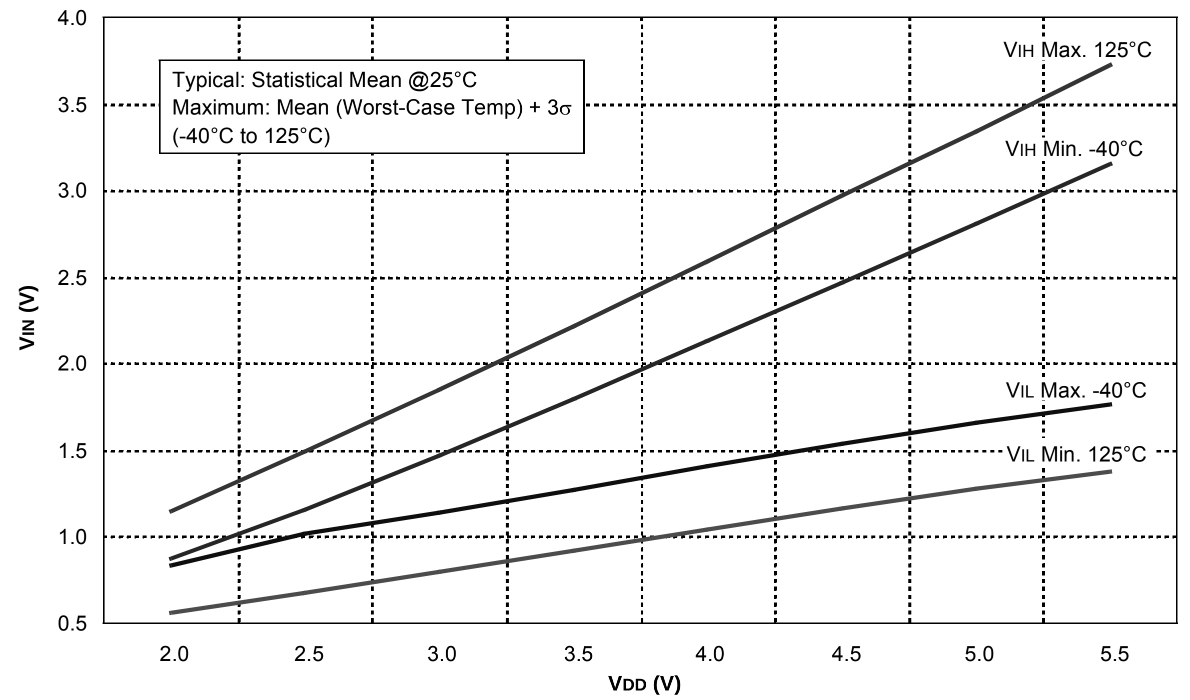
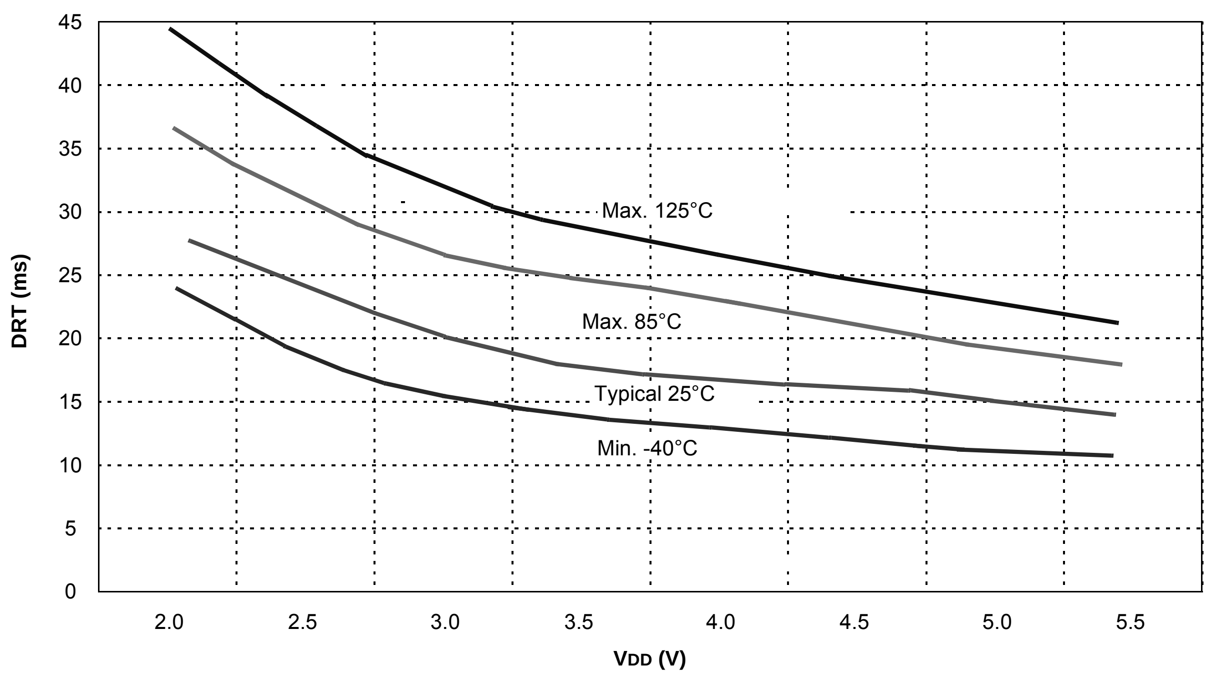
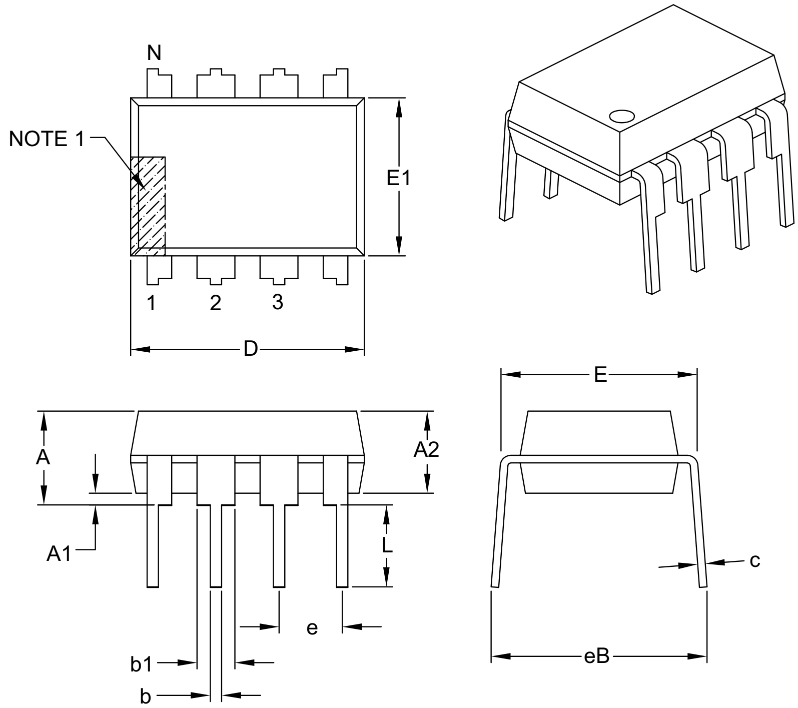


FIGURE 12-15: DEVICE RESET TIMER (XT AND LP) vs. VDD



8-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	8		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.348	.365	.400
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

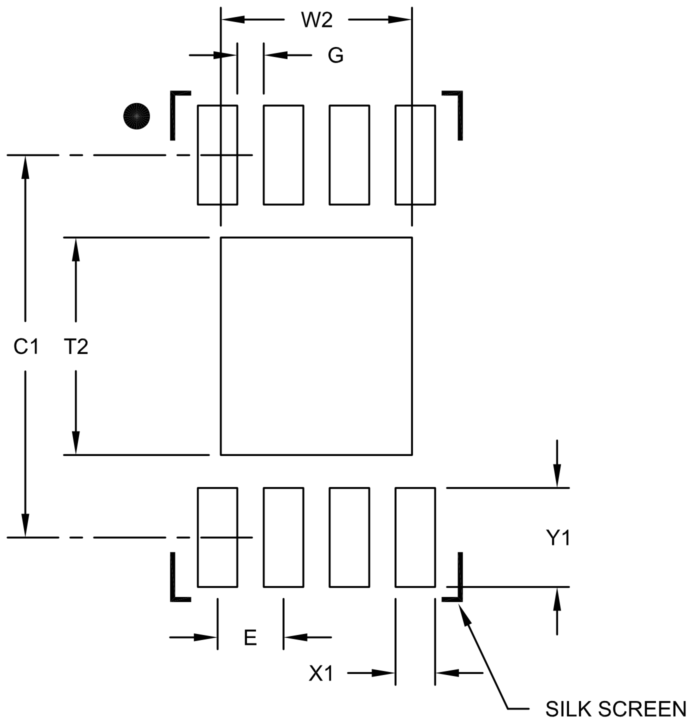
1. Pin 1 visual index feature may vary, but must be located with the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-018B

8-Lead Plastic Dual Flat, No Lead Package (MC) – 2x3x0.9 mm Body [DFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Contact Pitch	E	0.50 BSC		
Optional Center Pad Width	W2			1.45
Optional Center Pad Length	T2			1.75
Contact Pad Spacing	C1		2.90	
Contact Pad Width (X8)	X1			0.30
Contact Pad Length (X8)	Y1			0.75
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2123A

T

Timer0
    Timer0 (TMR0) Module 31
    TMR0 with External Clock 33
Timing Diagrams and Specifications 67
Timing Parameter Symbology and Load Conditions 67
TRIS Registers 23

W

Wake-up from Sleep 47
Watchdog Timer (WDT) 37, 45
    Period 45
    Programming Considerations 45
WWW Address 91
WWW, On-Line Support 3

Z

Zero bit 9