

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

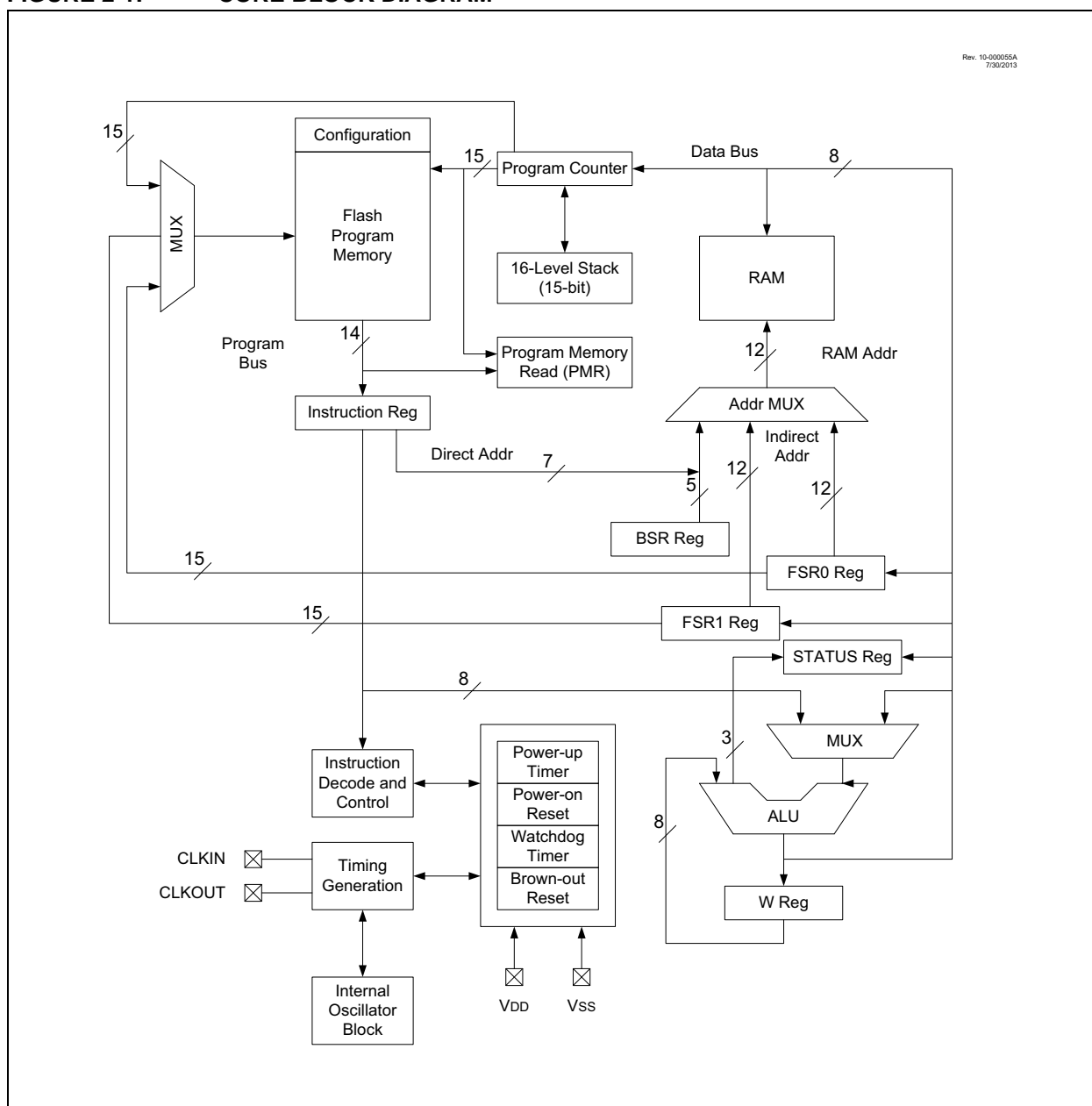
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	20MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	5
Program Memory Size	1.75KB (1K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	64 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 4x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	8-VDFN Exposed Pad
Supplier Device Package	8-DFN (2x3)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic12lf1501t-i-mc

2.0 ENHANCED MID-RANGE CPU

This family of devices contain an enhanced mid-range 8-bit CPU core. The CPU has 49 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16 levels deep and has Overflow and Underflow Reset capability. Direct, Indirect, and Relative addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

- Automatic Interrupt Context Saving
- 16-level Stack with Overflow and Underflow
- File Select Registers
- Instruction Set

FIGURE 2-1: CORE BLOCK DIAGRAM



PIC12(L)F1501

5.2 Clock Source Types

Clock sources can be classified as external, internal or peripheral.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (ECH, ECM, ECL modes).

Internal clock sources are contained within the oscillator module. The internal oscillator block has two internal oscillators that are used to generate the internal system clock sources: the 16 MHz High-Frequency Internal Oscillator (HFINTOSC) and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The peripheral clock source is a nominal 600 kHz internal RC oscillator, FRC. The FRC is traditionally used with the ADC module, but is sometimes available to other peripherals. See **Section 5.2.2.4 “Peripheral Clock Sources”**.

The system clock can be selected between external or internal clock sources via the System Clock Select (SCS) bits in the OSCCON register. See **Section 5.3 “Clock Switching”** for additional information.

5.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the FOSC<1:0> bits in the Configuration Words to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to:
 - Secondary oscillator during run-time, or
 - An external clock source determined by the value of the FOSC bits.

See **Section 5.3 “Clock Switching”** for more information.

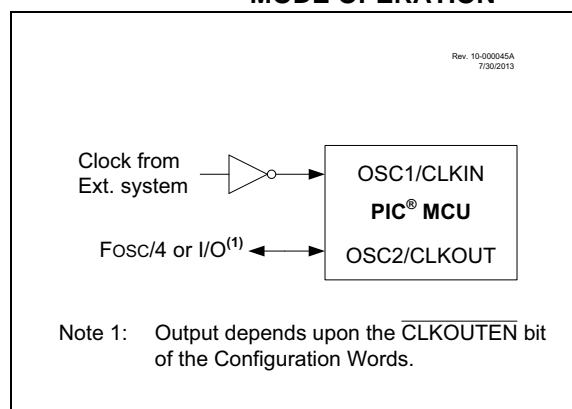
5.2.1.1 EC Mode

The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 5-2 shows the pin connections for EC mode.

EC mode has three power modes to select from through the Fosc bits in the Configuration Words:

- ECH – High power, 4-20 MHz
- ECM – Medium power, 0.5-4 MHz
- ECL – Low power, 0-0.5 MHz

FIGURE 5-2: EXTERNAL CLOCK (EC) MODE OPERATION



6.13 Power Control (PCON) Register

The Power Control (PCON) register contains flag bits to differentiate between a:

- Power-on Reset ($\overline{\text{POR}}$)
- Brown-out Reset ($\overline{\text{BOR}}$)
- Reset Instruction Reset ($\overline{\text{RI}}$)
- $\overline{\text{MCLR}}$ Reset ($\overline{\text{RMCLR}}$)
- Watchdog Timer Reset ($\overline{\text{RWDT}}$)
- Stack Underflow Reset (STKUNF)
- Stack Overflow Reset (STKOVF)

The PCON register bits are shown in Register 6-2.

6.14 Register Definitions: Power Control

REGISTER 6-2: PCON: POWER CONTROL REGISTER

R/W/HS-0/q	R/W/HS-0/q	U-0	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-q/u	R/W/HC-q/u
STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

Legend:

HC = Bit is cleared by hardware

HS = Bit is set by hardware

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

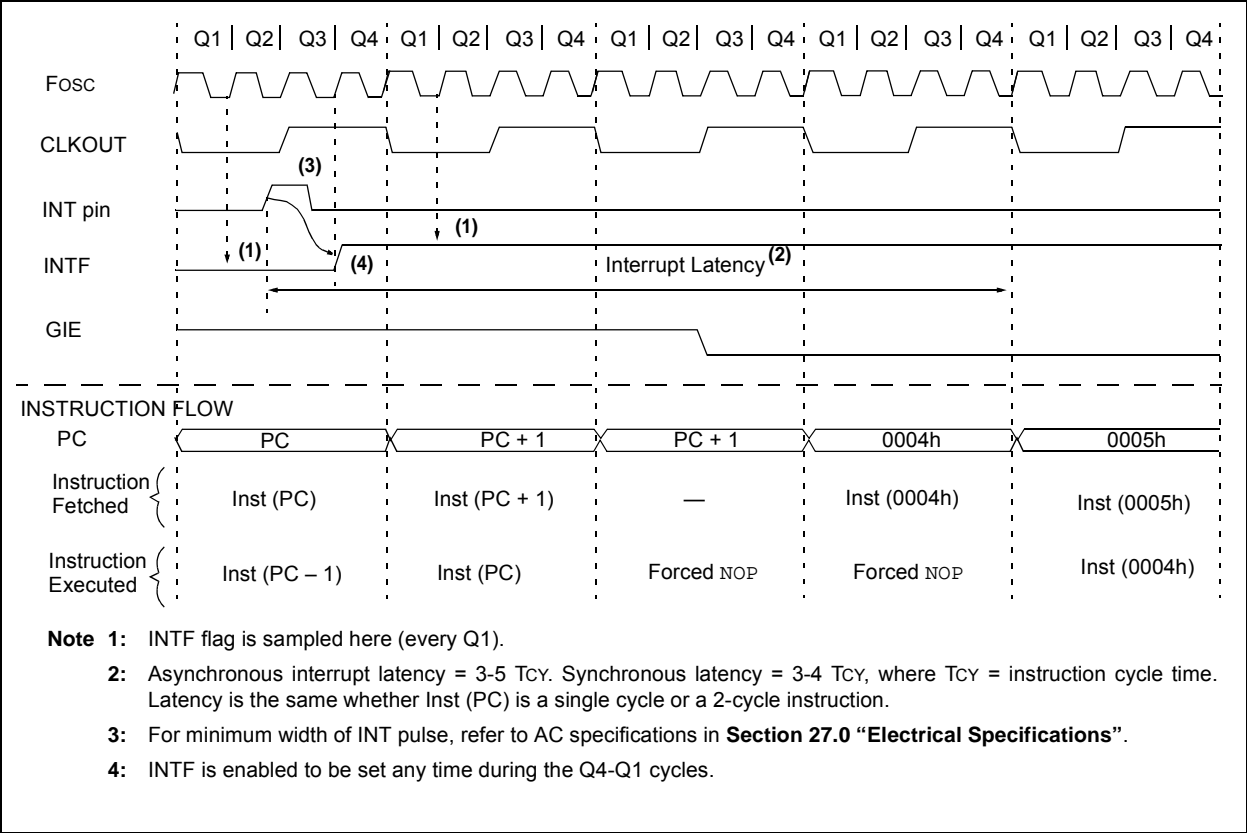
'1' = Bit is set

'0' = Bit is cleared

q = Value depends on condition

bit 7	STKOVF: Stack Overflow Flag bit 1 = A Stack Overflow occurred 0 = A Stack Overflow has not occurred or cleared by firmware
bit 6	STKUNF: Stack Underflow Flag bit 1 = A Stack Underflow occurred 0 = A Stack Underflow has not occurred or cleared by firmware
bit 5	Unimplemented: Read as '0'
bit 4	$\overline{\text{RWDT}}$: Watchdog Timer Reset Flag bit 1 = A Watchdog Timer Reset has not occurred or set by firmware 0 = A Watchdog Timer Reset has occurred (cleared by hardware)
bit 3	$\overline{\text{RMCLR}}$: $\overline{\text{MCLR}}$ Reset Flag bit 1 = A $\overline{\text{MCLR}}$ Reset has not occurred or set by firmware 0 = A $\overline{\text{MCLR}}$ Reset has occurred (cleared by hardware)
bit 2	$\overline{\text{RI}}$: RESET Instruction Flag bit 1 = A RESET instruction has not been executed or set by firmware 0 = A RESET instruction has been executed (cleared by hardware)
bit 1	$\overline{\text{POR}}$: 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	$\overline{\text{BOR}}$: Brown-Out Reset Status bit 1 = No Brown-out Reset occurred 0 = A Brown-out Reset occurred (must be set in software after a Power-on Reset or Brown-out Reset occurs)

FIGURE 7-3: INT PIN INTERRUPT TIMING



10.2.3 ERASING FLASH PROGRAM MEMORY

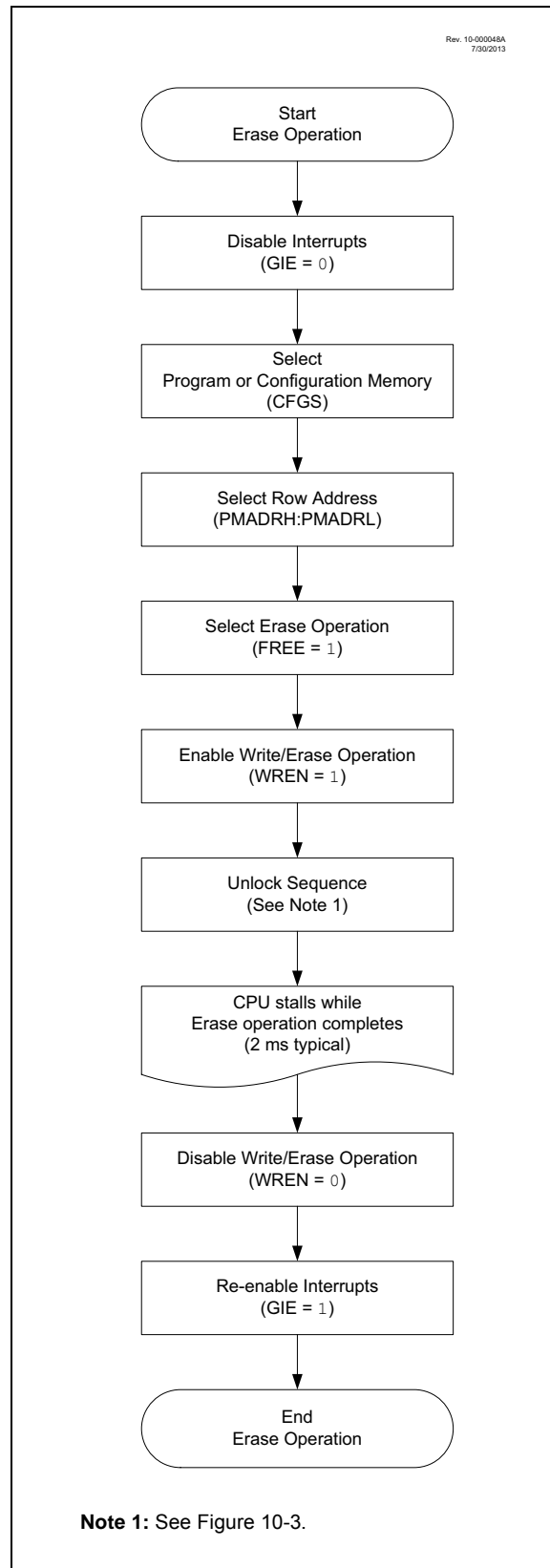
While executing code, program memory can only be erased by rows. To erase a row:

1. Load the PMADRH:PMADRL register pair with any address within the row to be erased.
2. Clear the CFGS bit of the PMCON1 register.
3. Set the FREE and WREN bits of the PMCON1 register.
4. Write 55h, then AAh, to PMCON2 (Flash programming unlock sequence).
5. Set control bit WR of the PMCON1 register to begin the erase operation.

See Example 10-2.

After the “BSF PMCON1, WR” instruction, the processor requires two cycles to set up the erase operation. The user must place two NOP instructions immediately following the WR bit set instruction. The processor will halt internal operations for the typical 2 ms erase time. This is not Sleep mode as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the PMCON1 write instruction.

FIGURE 10-4: FLASH PROGRAM MEMORY ERASE FLOWCHART



PIC12(L)F1501

EXAMPLE 10-2: ERASING ONE ROW OF PROGRAM MEMORY

```
; This row erase routine assumes the following:
; 1. A valid address within the erase row is loaded in ADDRH:ADDRL
; 2. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)

      BCF      INTCON,GIE      ; Disable ints so required sequences will execute properly
      BANKSEL  PMADRL
      MOVF     ADDRL,W         ; Load lower 8 bits of erase address boundary
      MOVWF    PMADRL
      MOVF     ADDRH,W         ; Load upper 6 bits of erase address boundary
      MOVWF    PMADRH
      BCF      PMCON1,CFGSR    ; Not configuration space
      BSF      PMCON1,FREE     ; Specify an erase operation
      BSF      PMCON1,WREN     ; Enable writes

      MOVLW    55h             ; Start of required sequence to initiate erase
      MOVWF    PMCON2          ; Write 55h
      MOVLW    0AAh           ;
      MOVWF    PMCON2          ; Write AAh
      BSF      PMCON1,WR       ; Set WR bit to begin erase
      NOP      ; NOP instructions are forced as processor starts
      NOP      ; row erase of program memory.
      NOP      ;
      NOP      ; The processor stalls until the erase process is complete
      NOP      ; after erase processor continues with 3rd instruction

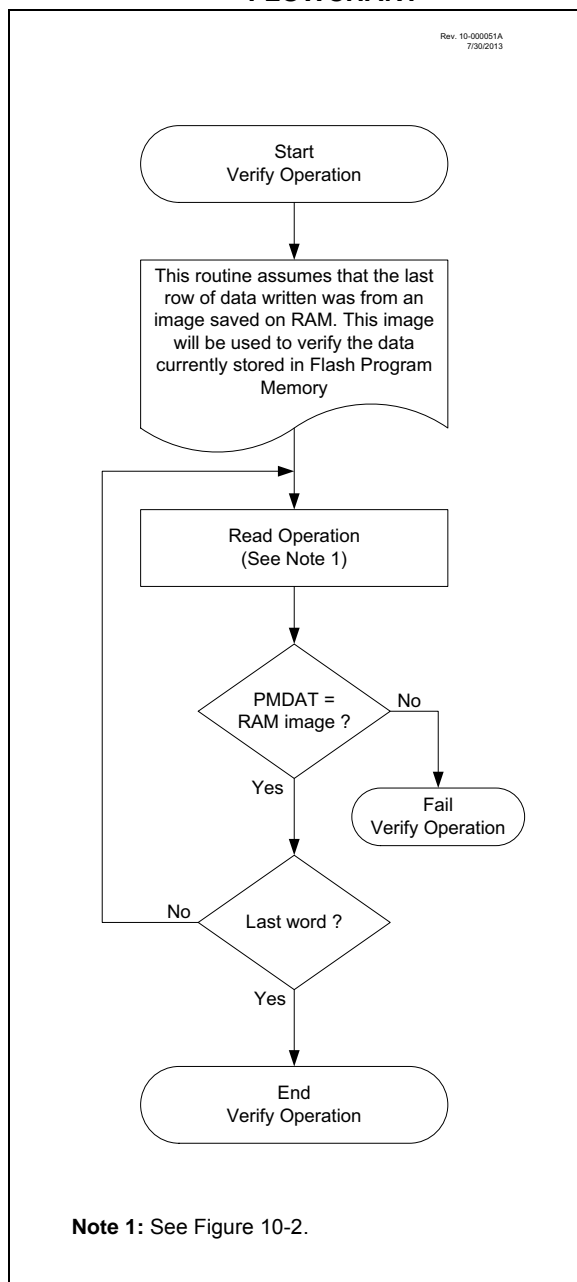
      BCF      PMCON1,WREN     ; Disable writes
      BSF      INTCON,GIE     ; Enable interrupts
```

Required
Sequence

10.5 Write Verify

It is considered good programming practice to verify that program memory writes agree with the intended value. Since program memory is stored as a full page then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

FIGURE 10-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART



11.3 PORTA Registers

11.3.1 DATA REGISTER

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 11-3). Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., disable the output driver). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). The exception is RA3, which is input-only and its TRIS bit will always read as '1'. Example 11-1 shows how to initialize an I/O port.

Reading the PORTA register (Register 11-2) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATA).

11.3.2 DIRECTION CONTROL

The TRISA register (Register 11-3) controls the PORTA pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISA register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

11.3.3 ANALOG CONTROL

The ANSELA register (Register 11-5) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELA bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELA bits has no effect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELA bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

EXAMPLE 11-1: INITIALIZING PORTA

```
BANKSEL PORTA      ;
CLRF PORTA         ;Init PORTA
BANKSEL LATA        ;Data Latch
CLRF LATA          ;
BANKSEL ANSELA      ;
CLRF ANSELA        ;digital I/O
BANKSEL TRISA       ;
MOVLW B'00111000'  ;Set RA<5:3> as inputs
MOVWF TRISA        ;and set RA<2:0> as
                   ;outputs
```

11.3.4 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 11-2.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

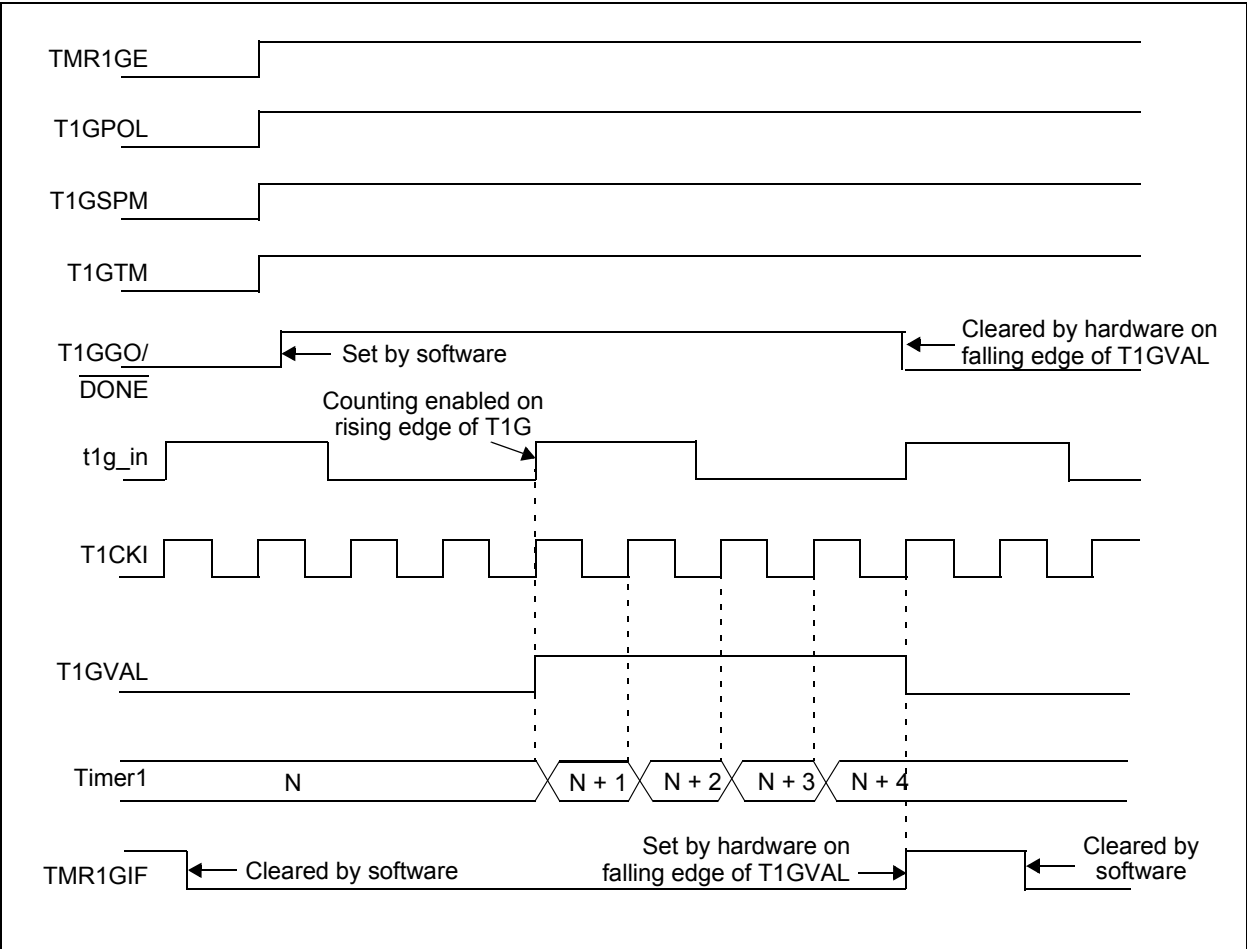
Analog input functions, such as ADC and comparator inputs, are not shown in the priority lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx registers. Digital output functions may control the pin when it is in Analog mode with the priority shown below in Table 11-2.

TABLE 11-2: PORTA OUTPUT PRIORITY

Pin Name	Function Priority ⁽¹⁾
RA0	ICSPDAT DACOUT1 CWG1B ⁽²⁾ PWM2 RA0
RA1	NCO1 ⁽²⁾ RA1
RA2	DACOUT2 CWG1A ⁽²⁾ CLC1 ⁽²⁾ C1OUT PWM1 RA2
RA3	None
RA4	CLKOUT CWG1B ⁽³⁾ CLC1 ⁽³⁾ PWM3 RA4
RA5	CWG1A ⁽³⁾ CLC2 NCO1 ⁽³⁾ PWM4 RA5

Note 1: Priority listed from highest to lowest.
 2: Default pin (see APFCON register).
 3: Alternate pin (see APFCON register).

FIGURE 19-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE



PIC12(L)F1501

21.1.9 SETUP FOR PWM OPERATION USING PWMx PINS

The following steps should be taken when configuring the module for PWM operation using the PWMx pins:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the PR2 register with the PWM period value.
4. Clear the PWMxDCH register and bits <7:6> of the PWMxDCL register.
5. Configure and start Timer2:
 - Clear the TMR2IF interrupt flag bit of the PIR1 register. See note below.
 - Configure the T2CKPS bits of the T2CON register with the Timer2 prescale value.
 - Enable Timer2 by setting the TMR2ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the PIR1 register is set. See note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the PWMxOE bit of the PWMxCON register.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note 1: In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.

2: For operation with other peripherals only, disable PWMx pin outputs.

21.2 Register Definitions: PWM Control

REGISTER 21-1: PWMxCON: PWM CONTROL REGISTER

R/W-0/0	R/W-0/0	R-0/0	R/W-0/0	U-0	U-0	U-0	U-0
PWMxEN	PWMxOE	PWMxOUT	PWMxPOL	—	—	—	—
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7 **PWMxEN:** PWM Module Enable bit
 1 = PWM module is enabled
 0 = PWM module is disabled
- bit 6 **PWMxOE:** PWM Module Output Enable bit
 1 = Output to PWMx pin is enabled
 0 = Output to PWMx pin is disabled
- bit 5 **PWMxOUT:** PWM Module Output Value bit
- bit 4 **PWMxPOL:** PWMx Output Polarity Select bit
 1 = PWM output is active-low
 0 = PWM output is active-high
- bit 3-0 **Unimplemented:** Read as '0'

FIGURE 24-3: DEAD-BAND OPERATION, CWGxDBR = 01H, CWGxDBF = 02H

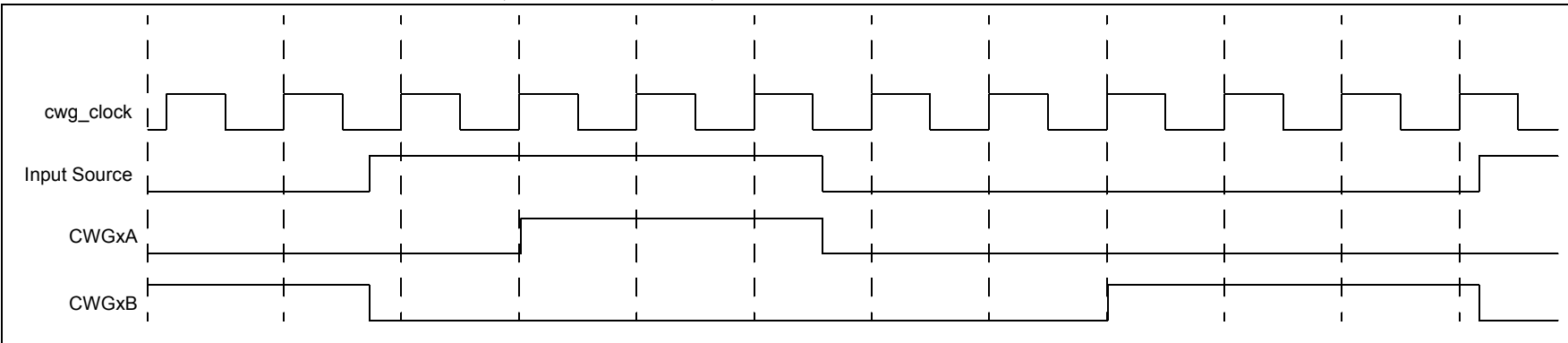
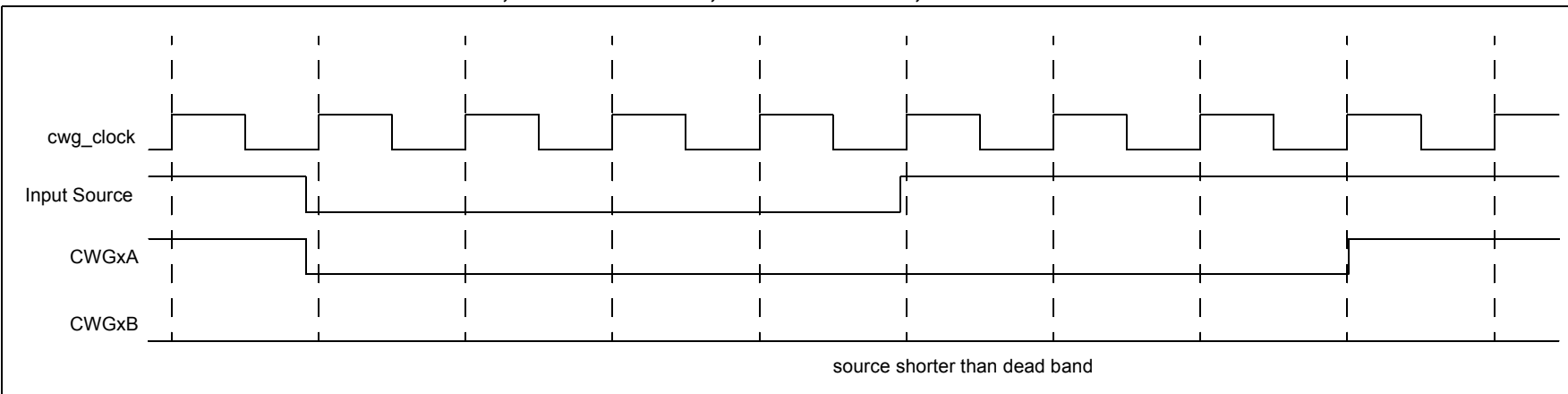


FIGURE 24-4: DEAD-BAND OPERATION, CWGxDBR = 03H, CWGxDBF = 04H, SOURCE SHORTER THAN DEAD BAND



PIC12(L)F1501

CALL Call Subroutine

Syntax: [*label*] CALL k

Operands: $0 \leq k \leq 2047$

Operation: (PC) + 1 → TOS,
k → PC<10:0>,
(PCLATH<6:3>) → PC<14:11>

Status Affected: None

Description: Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

CLRWDTClear Watchdog Timer

Syntax: [*label*] CLRWDTClear Watchdog Timer

Operands: None

Operation: 00h → WDT
0 → WDT prescaler,
1 → \overline{TO}
1 → \overline{PD}

Status Affected: \overline{TO} , \overline{PD}

Description: CLRWDTClear Watchdog Timer instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits \overline{TO} and \overline{PD} are set.

CALLW Subroutine Call With W

Syntax: [*label*] CALLW

Operands: None

Operation: (PC) + 1 → TOS,
(W) → PC<7:0>,
(PCLATH<6:0>) → PC<14:8>

Status Affected: None

Description: Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

COMF Complement f

Syntax: [*label*] COMF f,d

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

Operation: (\bar{f}) → (destination)

Status Affected: Z

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

CLRF Clear f

Syntax: [*label*] CLRF f

Operands: $0 \leq f \leq 127$

Operation: 00h → (f)
1 → Z

Status Affected: Z

Description: The contents of register 'f' are cleared and the Z bit is set.

DECF Decrement f

Syntax: [*label*] DECF f,d

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

Operation: (f) - 1 → (destination)

Status Affected: Z

Description: Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

CLRWClear W

Syntax: [*label*] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Description: W register is cleared. Zero bit (Z) is set.

SWAPF Swap Nibbles in f

Syntax: [*label*] SWAPF f,d

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

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

Status Affected: None

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

XORLW Exclusive OR literal with W

Syntax: [*label*] XORLW k

Operands: $0 \leq k \leq 255$

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

Status Affected: Z

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

TRIS Load TRIS Register with W

Syntax: [*label*] TRIS f

Operands: $5 \leq f \leq 7$

Operation: $(W) \rightarrow \text{TRIS register 'f'}$

Status Affected: None

Description: Move data from W register to TRIS register.
 When 'f' = 5, TRISA is loaded.
 When 'f' = 6, TRISB is loaded.
 When 'f' = 7, TRISC is loaded.

XORWF Exclusive OR W with f

Syntax: [*label*] XORWF f,d

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

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

Status Affected: Z

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

PIC12(L)F1501

FIGURE 28-7: I_{DD} TYPICAL, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC12F1501 ONLY

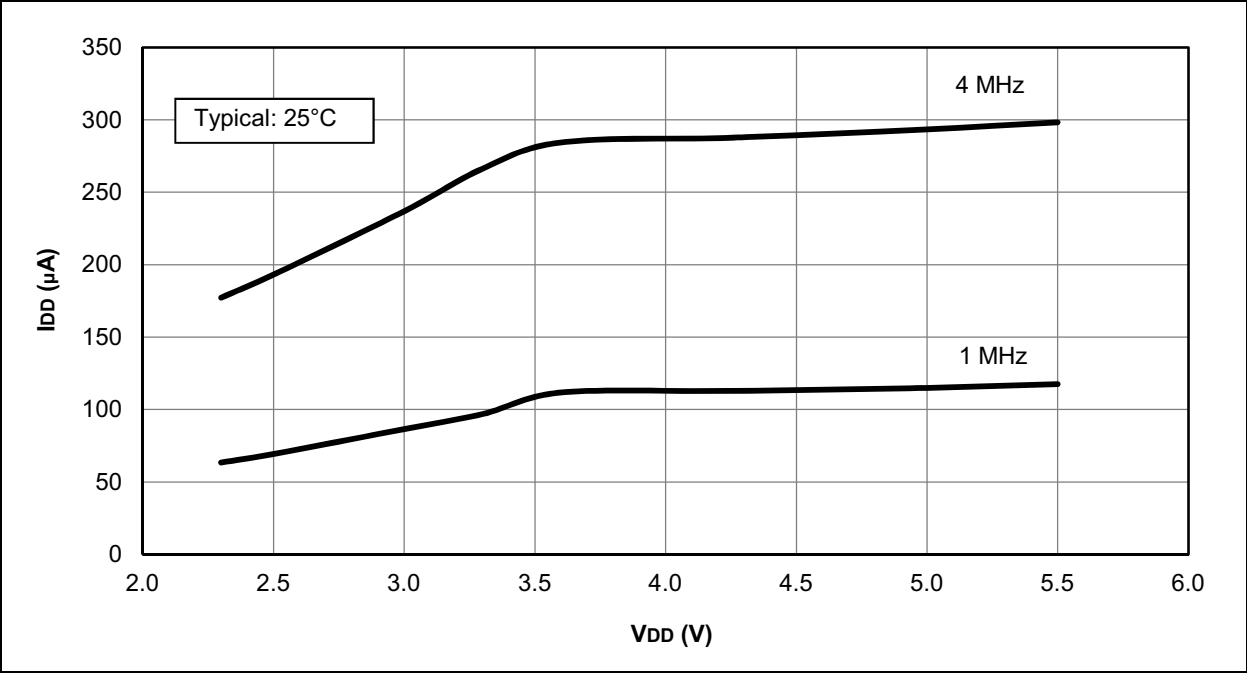
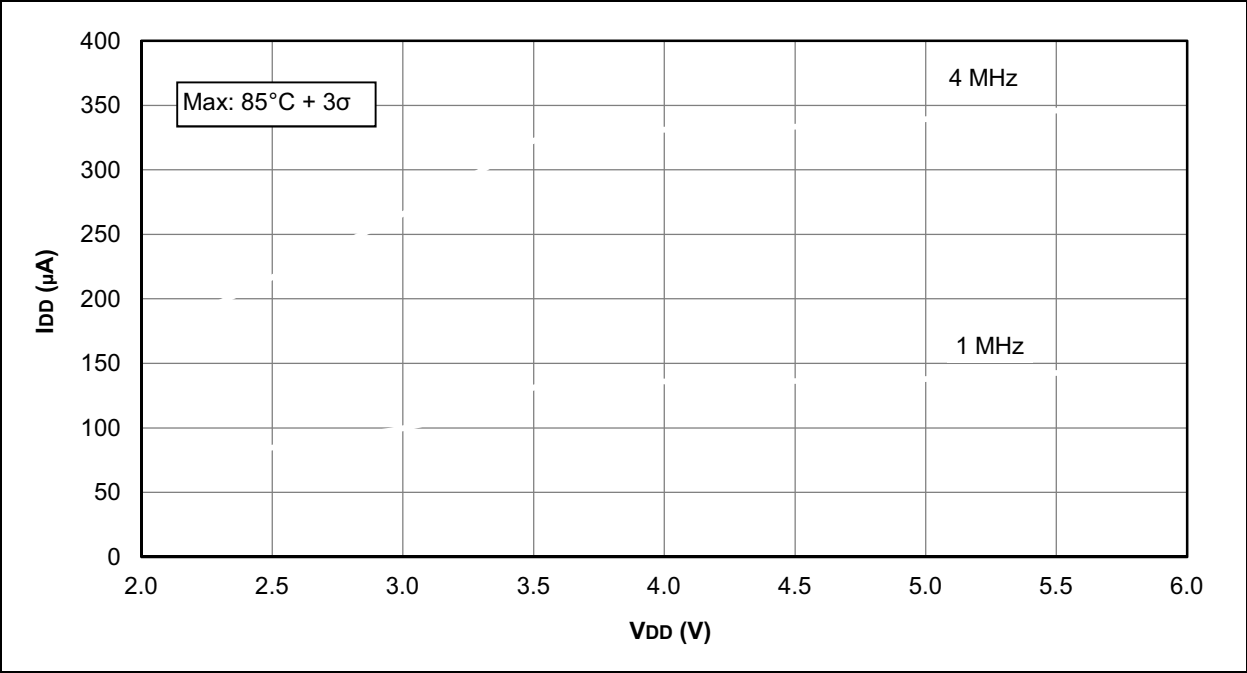


FIGURE 28-8: I_{DD} MAXIMUM, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC12F1501 ONLY



PIC12(L)F1501

FIGURE 28-19: I_{DD} TYPICAL, HFINTOSC, PIC12F1501 ONLY

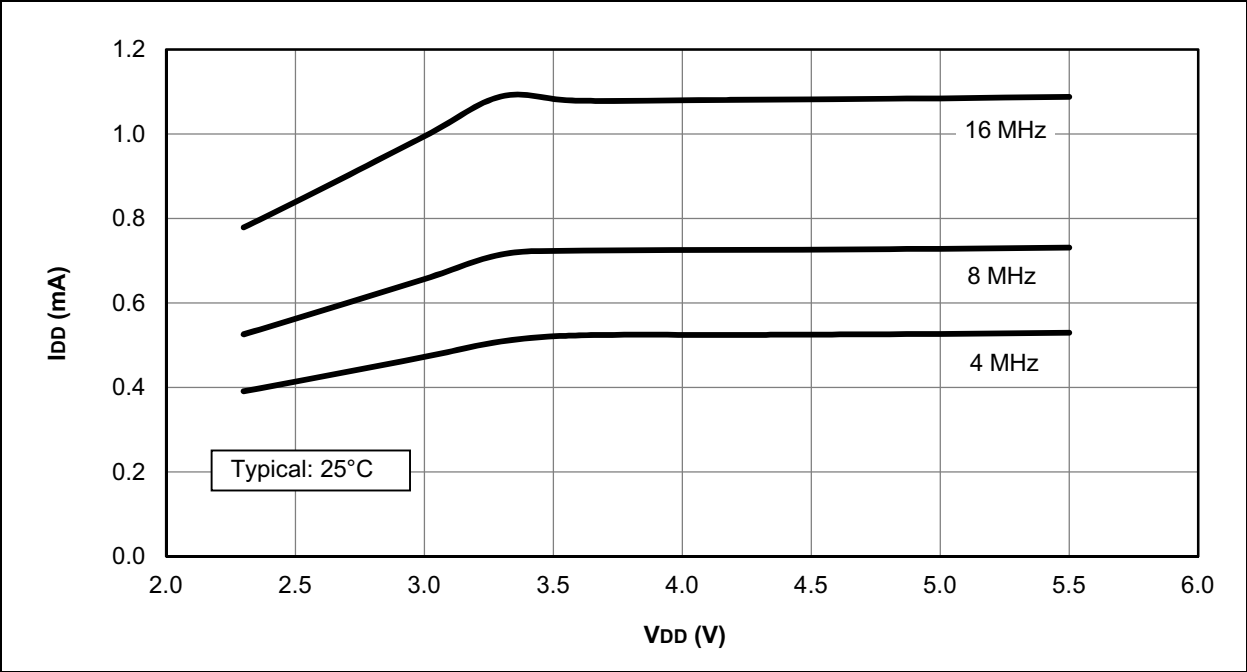
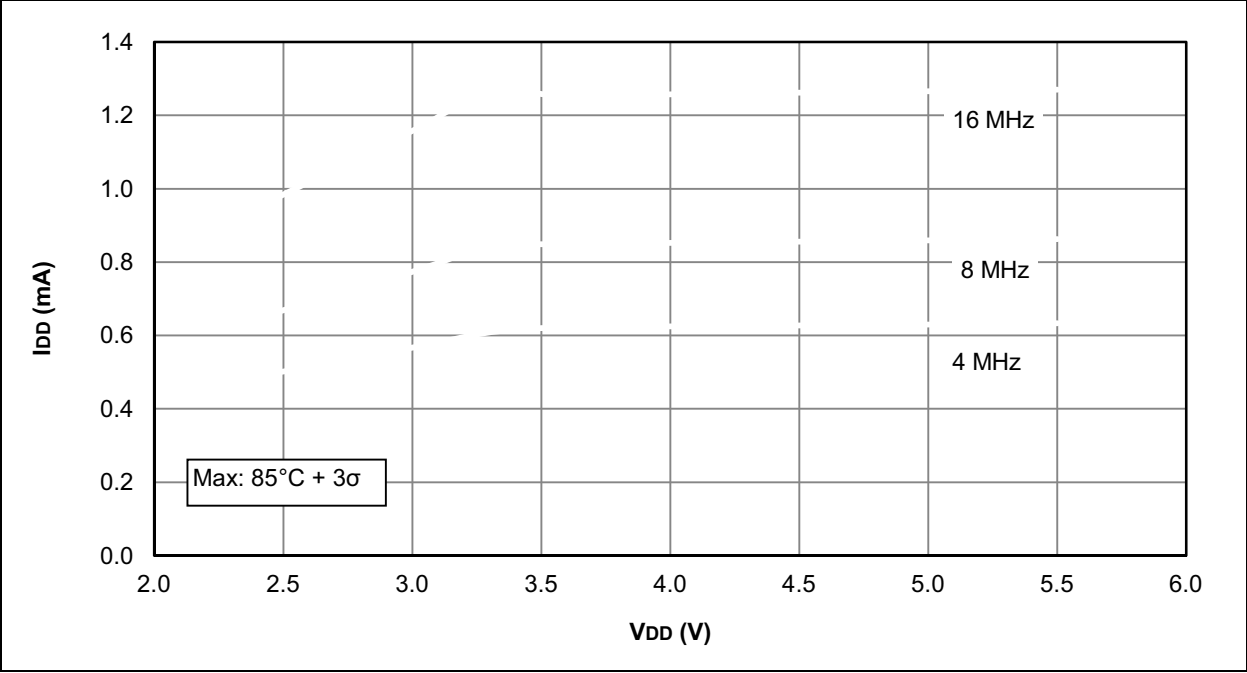


FIGURE 28-20: I_{DD} MAXIMUM, HFINTOSC, PIC12F1501 ONLY



PIC12(L)F1501

FIGURE 28-50: WDT TIME-OUT PERIOD

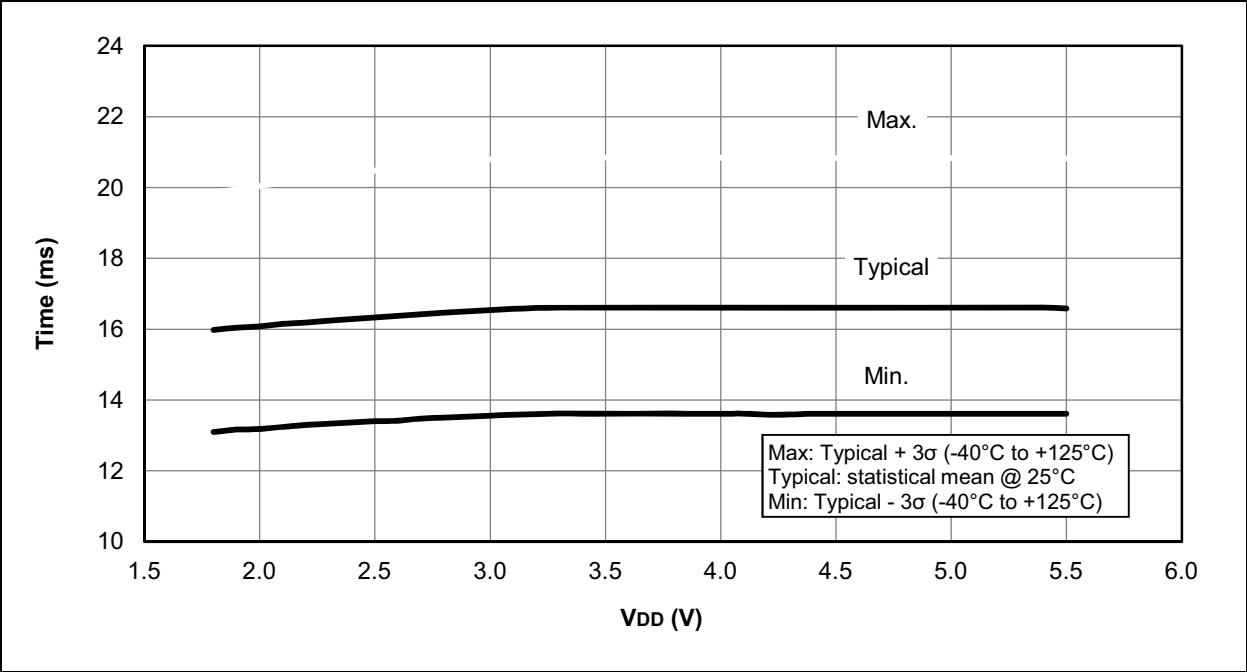


FIGURE 28-51: PWRT PERIOD

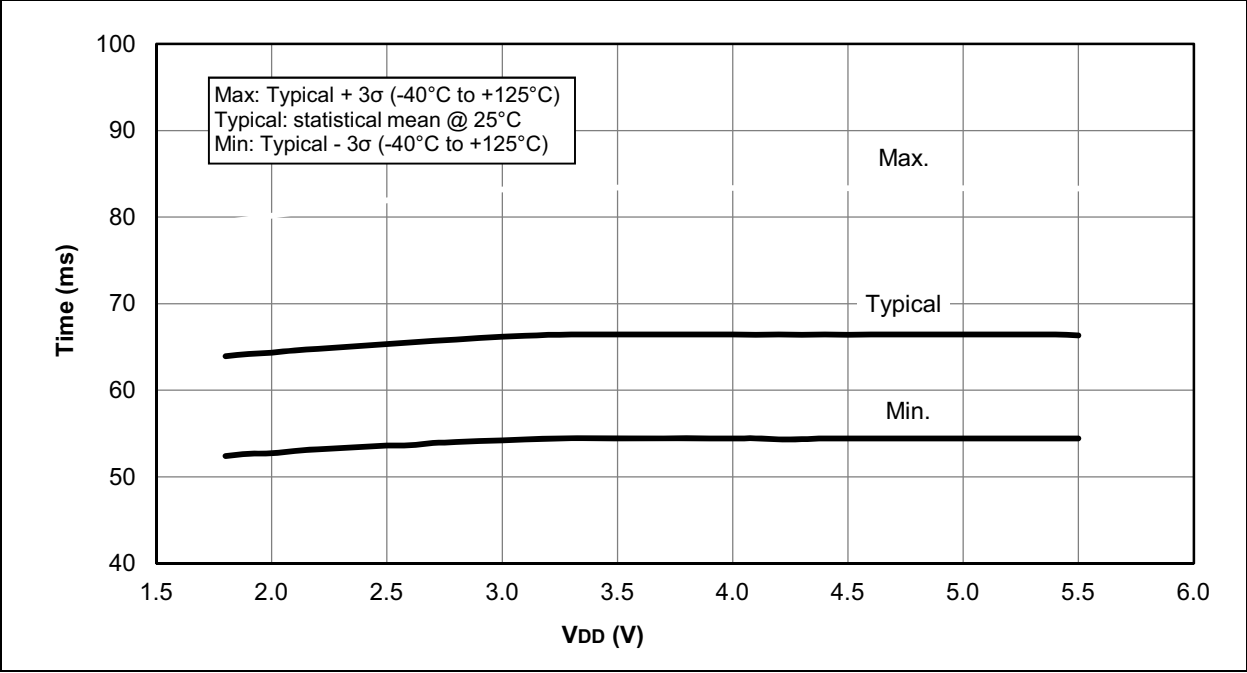


FIGURE 28-58: LFINTOSC FREQUENCY OVER V_{DD} AND TEMPERATURE, PIC12LF1501 ONLY

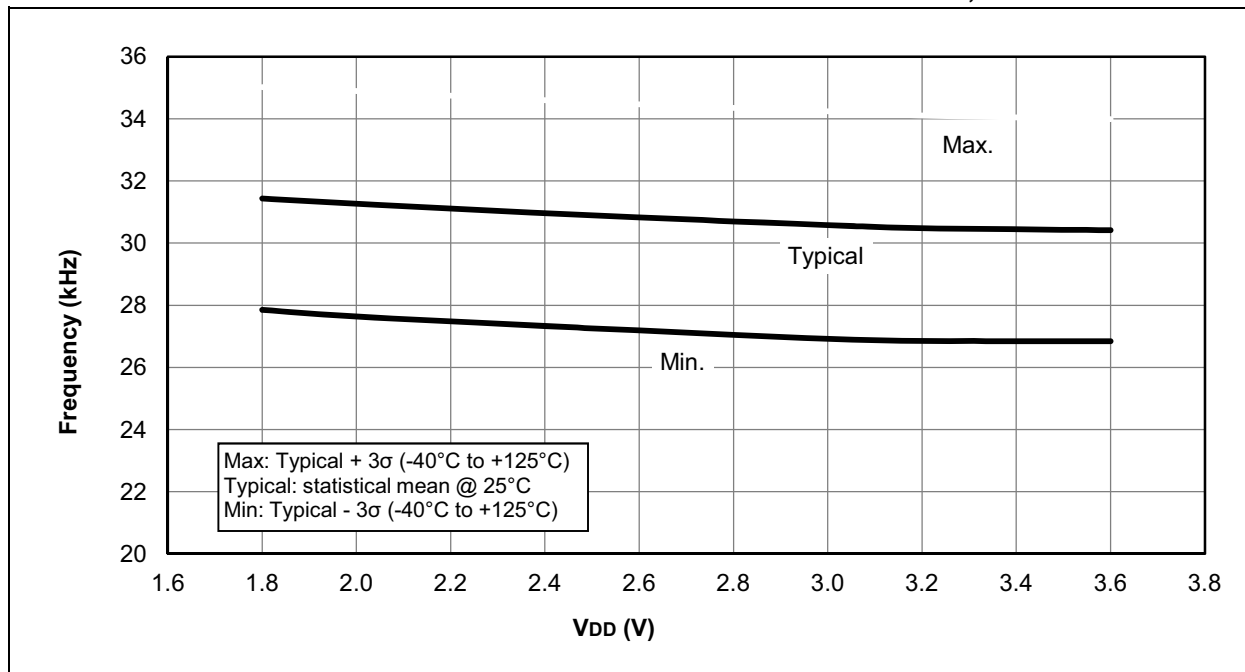
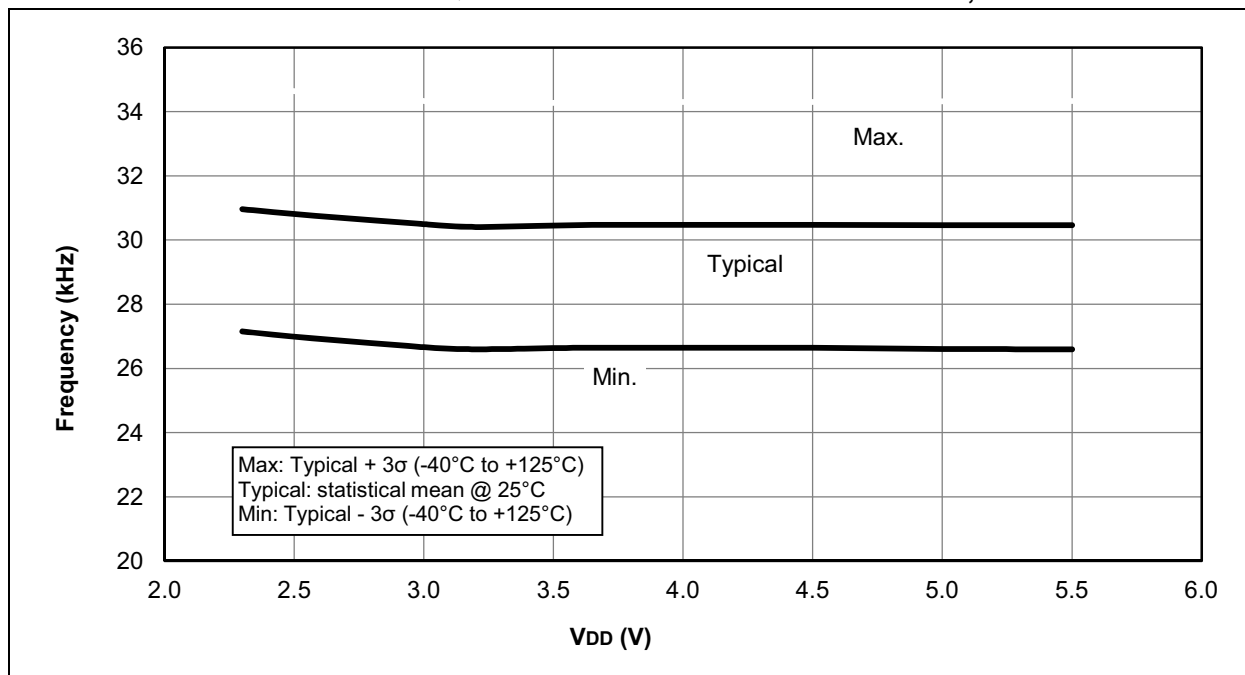


FIGURE 28-59: LFINTOSC FREQUENCY OVER V_{DD} AND TEMPERATURE, PIC12F1501 ONLY



29.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
 - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
 - MPLAB XC Compiler
 - MPASM™ Assembler
 - MPLINK™ Object Linker/
MPLIB™ Object Librarian
 - MPLAB Assembler/Linker/Librarian for
Various Device Families
- Simulators
 - MPLAB X SIM Software Simulator
- Emulators
 - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
 - MPLAB ICD 3
 - PICKit™ 3
- Device Programmers
 - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards,
Evaluation Kits and Starter Kits
- Third-party development tools

29.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

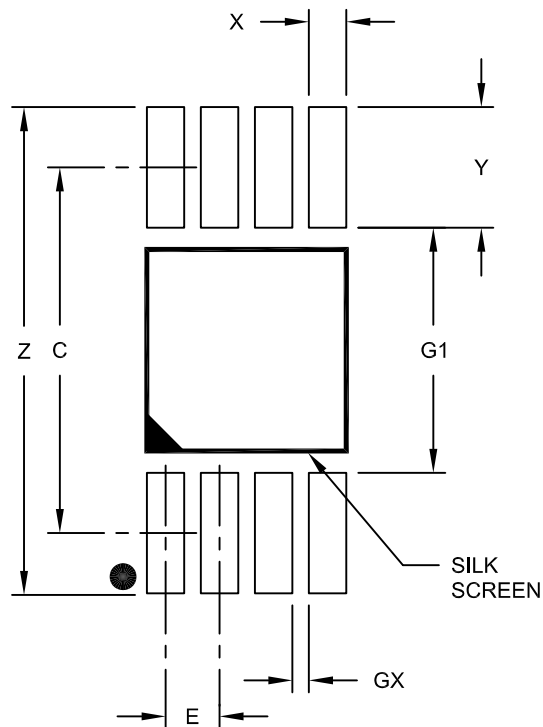
File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

PIC12(L)F1501

8-Lead Plastic Micro Small Outline Package (MS) [MSOP]

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	0.65 BSC		
Contact Pad Spacing	C		4.40	
Overall Width	Z			5.85
Contact Pad Width (X8)	X1			0.45
Contact Pad Length (X8)	Y1			1.45
Distance Between Pads	G1	2.95		
Distance Between Pads	GX	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

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

Microchip Technology Drawing No. C04-2111A