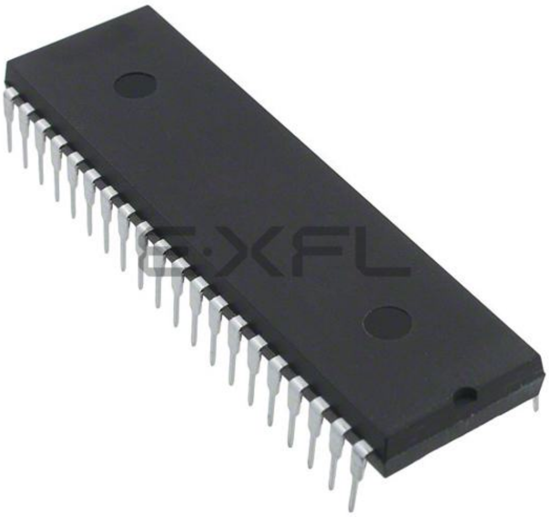


Welcome to E-XFL.COM

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	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	34
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f442-i-p

2.6 Oscillator Switching Feature

The PIC18FXX2 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low frequency clock source. For the PIC18FXX2 devices, this alternate clock source is the Timer1 oscillator. If a low frequency crystal (32 kHz, for example) has been attached to the Timer1 oscillator pins and the Timer1 oscillator has been enabled, the device can switch to a Low Power Execution mode. Figure 2-7 shows a block diagram of the system clock sources. The clock switching feature is enabled by programming the Oscillator Switching Enable ($\overline{\text{OSCSEN}}$) bit in Configuration Register1H to a '0'. Clock switching is disabled in an erased device. See Section 11.0 for further details of the Timer1 oscillator. See Section 19.0 for Configuration Register details.

FIGURE 2-7: DEVICE CLOCK SOURCES

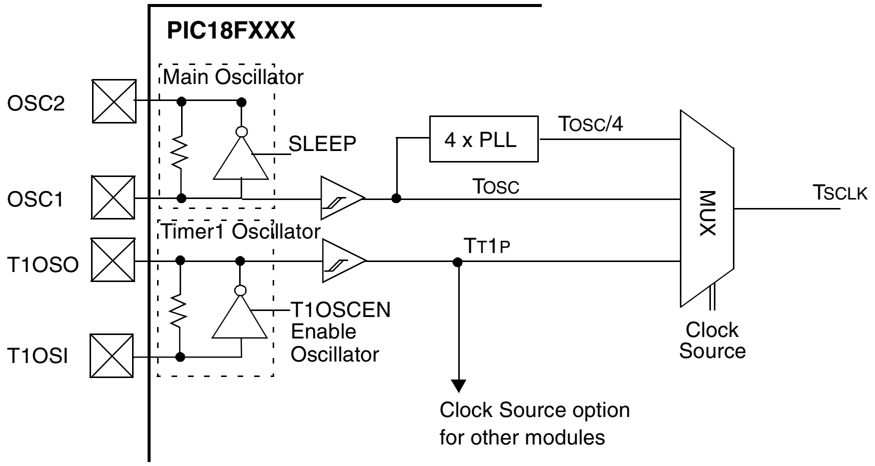


TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset `RESET` Instruction Stack Resets	Wake-up via WDT or Interrupt
IPR2	242	442	252	452	`---1 1111`	`---1 1111`	`---u uuuu`
PIR2	242	442	252	452	`---0 0000`	`---0 0000`	`---u uuuu`[1]
PIE2	242	442	252	452	`---0 0000`	`---0 0000`	`---u uuuu`
IPR1	242	442	252	452	`1111 1111`	`1111 1111`	`uuuu uuuu`
	242	442	252	452	`-111 1111`	`-111 1111`	`-uuu uuuu`
PIR1	242	442	252	452	`0000 0000`	`0000 0000`	`uuuu uuuu`[1]
	242	442	252	452	`-000 0000`	`-000 0000`	`-uuu uuuu`[1]
PIE1	242	442	252	452	`0000 0000`	`0000 0000`	`uuuu uuuu`
	242	442	252	452	`-000 0000`	`-000 0000`	`-uuu uuuu`
TRISE	242	442	252	452	`0000 -111`	`0000 -111`	`uuuu -uuu`
TRISD	242	442	252	452	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISC	242	442	252	452	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISB	242	442	252	452	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISA[5,6]	242	442	252	452	`-111 1111`[5]	`-111 1111`[5]	`-uuu uuuu`[5]
LATE	242	442	252	452	`---- -xxx`	`---- -uuu`	`---- -uuu`
LATD	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATC	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATB	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATA[5,6]	242	442	252	452	`-xxx xxxx`[5]	`-uuu uuuu`[5]	`-uuu uuuu`[5]
PORTE	242	442	252	452	`---- -000`	`---- -000`	`---- -uuu`
PORTD	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTC	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTB	242	442	252	452	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTA[5,6]	242	442	252	452	`-x0x 0000`[5]	`-u0u 0000`[5]	`-uuu uuuu`[5]

Legend: `u` = unchanged, `x` = unknown, `-` = unimplemented bit, read as '0', `q` = value depends on condition.
Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (`0008h` or `0018h`).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 3-2 for RESET value for specific condition.

5: Bit 6 of PORTA, LATA, and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other Oscillator modes, they are disabled and read ’0’.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read ’0’.

REGISTER 4-1: STKPTR REGISTER

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKOVF	STKUNF	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

bit 7[1] **STKOVF:** Stack Full Flag bit

1 = Stack became full or overflowed
0 = Stack has not become full or overflowed

bit 6[1] **STKUNF:** Stack Underflow Flag bit

1 = Stack underflow occurred
0 = Stack underflow did not occur

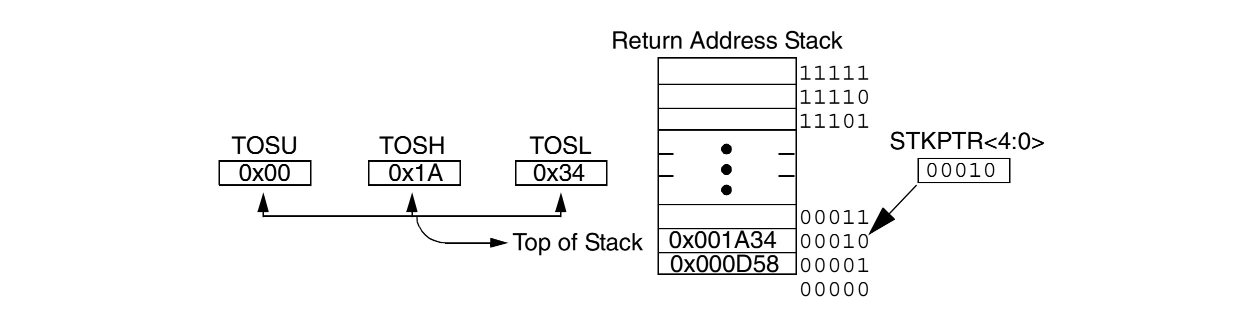
bit 5 **Unimplemented:** Read as '0'

bit 4-0 **SP4:SP0:** Stack Pointer Location bits

Note 1: Bit 7 and bit 6 can only be cleared in user software or by a POR.

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 4-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



4.2.3 PUSH AND POP INSTRUCTIONS

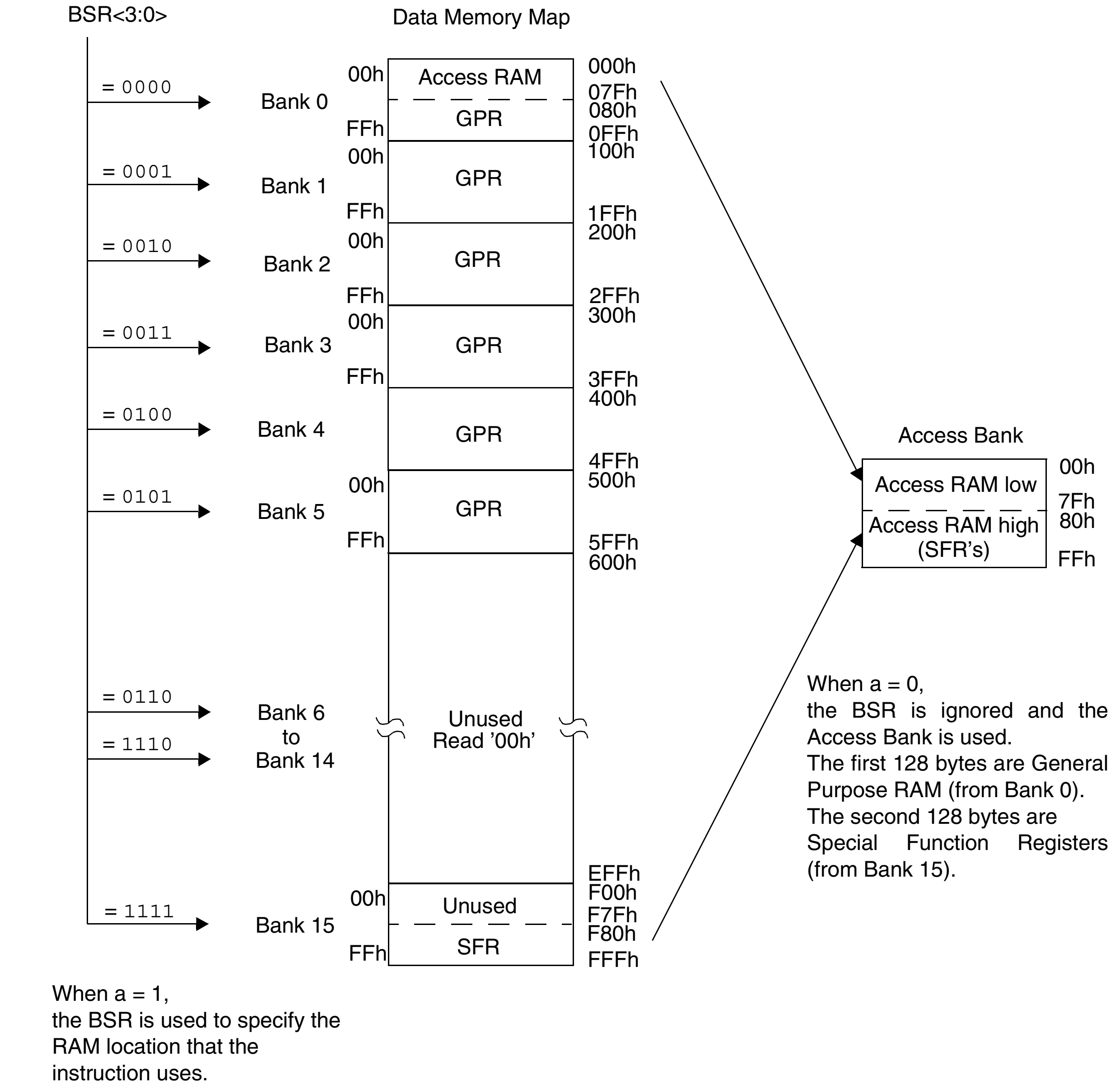
Since the Top-of-Stack (TOS) is readable and writable, the ability to push values onto the stack and pull values off the stack without disturbing normal program execution is a desirable option. To push the current PC value onto the stack, a `PUSH` instruction can be executed. This will increment the stack pointer and load the current PC value onto the stack. TOSU, TOSH and TOSL can then be modified to place a return address on the stack.

The ability to pull the TOS value off of the stack and replace it with the value that was previously pushed onto the stack, without disturbing normal execution, is achieved by using the `POP` instruction. The `POP` instruction discards the current TOS by decrementing the stack pointer. The previous value pushed onto the stack then becomes the TOS value.

4.2.4 STACK FULL/UNDERFLOW RESETS

These resets are enabled by programming the STVREN configuration bit. When the STVREN bit is disabled, a full or underflow condition will set the appropriate STKFUL or STKUNF bit, but not cause a device RESET. When the STVREN bit is enabled, a full or underflow will set the appropriate STKFUL or STKUNF bit and then cause a device RESET. The STKFUL or STKUNF bits are only cleared by the user software or a POR Reset.

FIGURE 4-7: DATA MEMORY MAP FOR PIC18F252/452



When a = 0,
the BSR is ignored and the Access Bank is used.
The first 128 bytes are General Purpose RAM (from Bank 0).
The second 128 bytes are Special Function Registers (from Bank 15).

When a = 1,
the BSR is used to specify the RAM location that the instruction uses.

FIGURE 4-9: INDIRECT ADDRESSING OPERATION

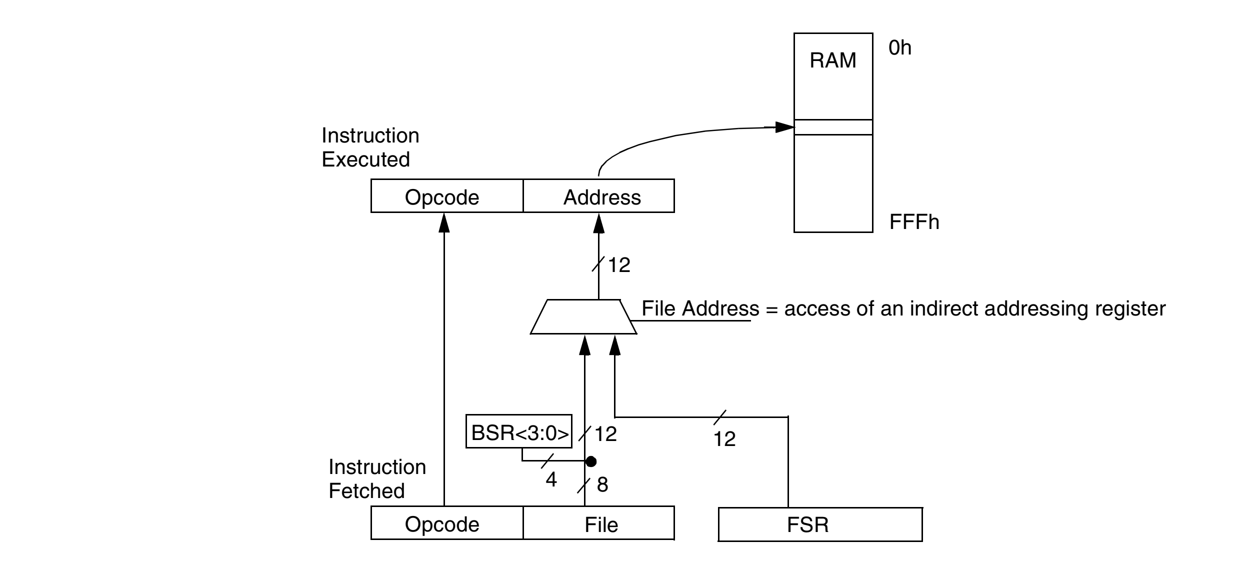
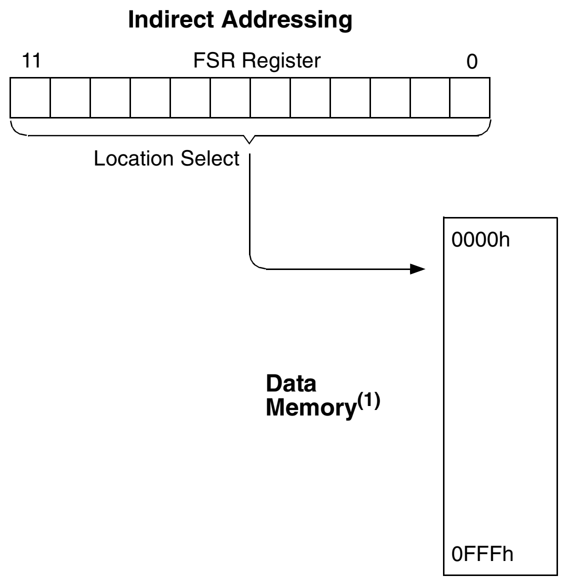


FIGURE 4-10: INDIRECT ADDRESSING



Note 1: For register file map detail, see Table 4-1.

NOTES:

6.0 DATA EEPROM MEMORY

The Data EEPROM is readable and writable during normal operation over the entire VDD range. The data memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers (SFR).

There are four SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADR

The EEPROM data memory allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 256 bytes of data EEPROM with an address range from 0h to FFh.

The EEPROM data memory is rated for high erase/write cycles. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature, as well as from chip to chip. Please refer to parameter D122 (Electrical Characteristics, Section 22.0) for exact limits.

6.1 EEADR

The address register can address up to a maximum of 256 bytes of data EEPROM.

6.2 EECON1 and EECON2 Registers

EECON1 is the control register for EEPROM memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the EEPROM write sequence.

Control bits RD and WR initiate read and write operations, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at the completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a $\overline{\text{MCLR}}$ Reset, or a WDT Time-out Reset during normal operation. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR), due to the RESET condition forcing the contents of the registers to zero.

Note: Interrupt flag bit, EEIF in the PIR2 register, is set when write is complete. It must be cleared in software.

6.5 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

6.6 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

6.7 Operation During Code Protect

Data EEPROM memory has its own code protect mechanism. External Read and Write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal Data EEPROM, regardless of the state of the code protect configuration bit. Refer to “Special Features of the CPU” (Section 19.0) for additional information.

6.8 Using the Data EEPROM

The data EEPROM is a high endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in FLASH program memory.

A simple data EEPROM refresh routine is shown in Example 6-3.

> **Note:** If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124.

EXAMPLE 6-3: DATA EEPROM REFRESH ROUTINE

```
        clrf    EEADR                   ; Start at address 0
        bcf     EECON1,CFGS             ; Set for memory
        bcf     EECON1,EEPGD            ; Set for Data EEPROM
        bcf     INTCON,GIE              ; Disable interrupts
        bsf     EECON1,WREN             ; Enable writes
Loop                                    ; Loop to refresh array
        bsf     EECON1,RD               ; Read current address
        movlw   55h                     ;
        movwf   EECON2                  ; Write 55h
        movlw   AAh                     ;
        movwf   EECON2                  ; Write AAh
        bsf     EECON1,WR               ; Set WR bit to begin write
        btfsc   EECON1,WR               ; Wait for write to complete
        bra     $-2
        incfsz  EEADR,F                 ; Increment address
        bra     Loop                    ; Not zero, do it again

        bcf     EECON1,WREN             ; Disable writes
        bsf     INTCON,GIE              ; Enable interrupts
```

11.1 Timer1 Operation

Timer1 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

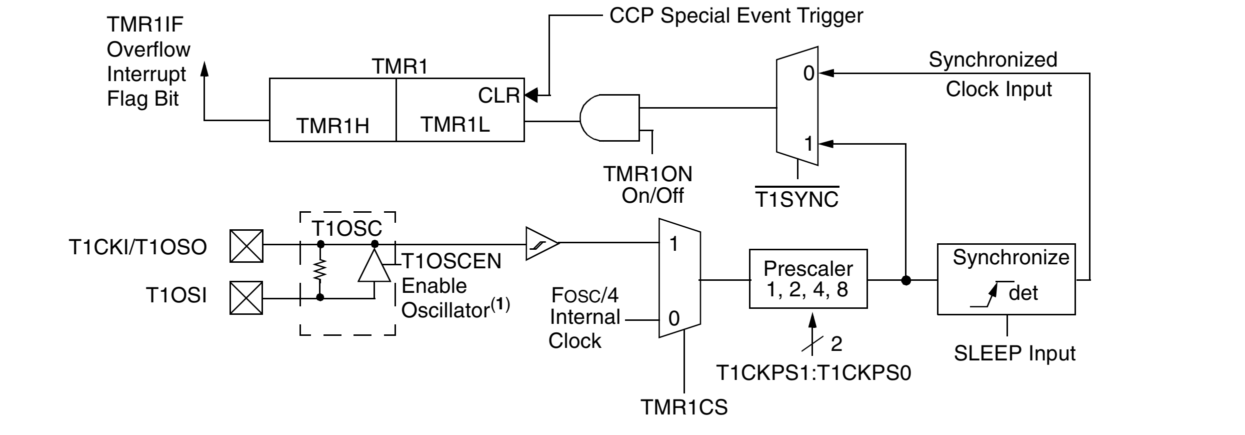
The Operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

When TMR1CS = 0, Timer1 increments every instruction cycle. When TMR1CS = 1, Timer1 increments on every rising edge of the external clock input or the Timer1 oscillator, if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI and RC0/T1OSO/T1CKI pins become inputs. That is, the TRISC<1:0> value is ignored, and the pins are read as '0'.

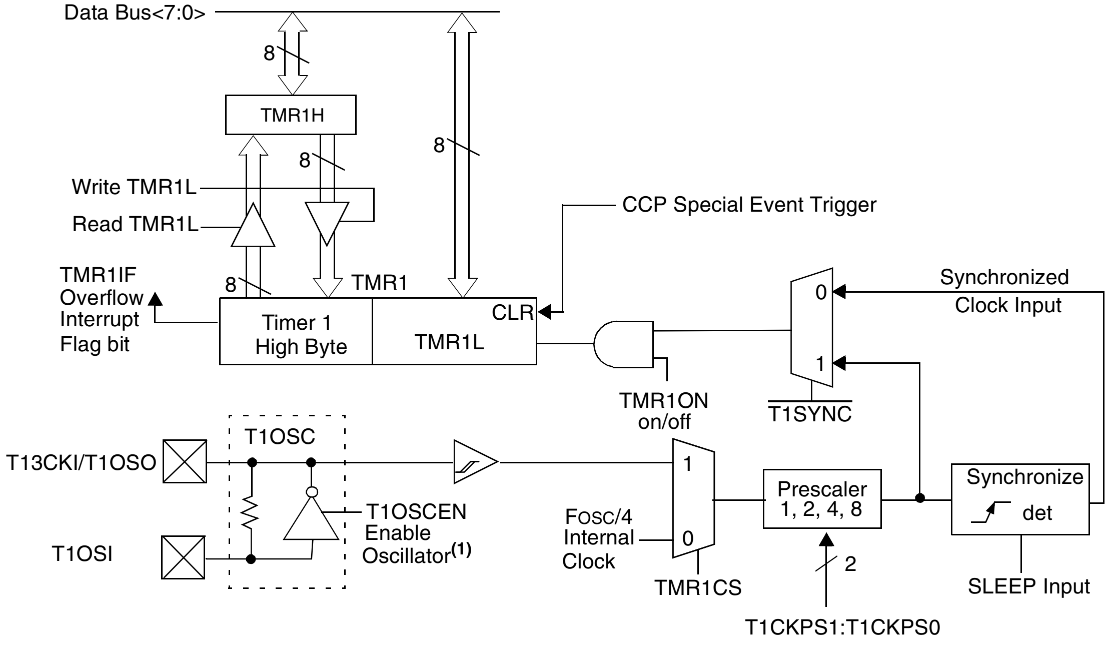
Timer1 also has an internal "RESET input". This RESET can be generated by the CCP module (Section 14.0).

FIGURE 11-1: TIMER1 BLOCK DIAGRAM



Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

FIGURE 11-2: TIMER1 BLOCK DIAGRAM: 16-BIT READ/WRITE MODE



Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

14.3 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on pin RC2/CCP1. An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

The event is selected by control bits CCP1M3:CCP1M0 (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 is overwritten by the new captured value.

14.3.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1 pin should be configured as an input by setting the TRISC<2> bit.

Note:	If the RC2/CCP1 is configured as an output, a write to the port can cause a capture condition.

14.3.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (either Timer1 and/or Timer3) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation may not work. The timer to be used with each CCP module is selected in the T3CON register.

14.3.3 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.

14.3.4 CCP PRESCALER

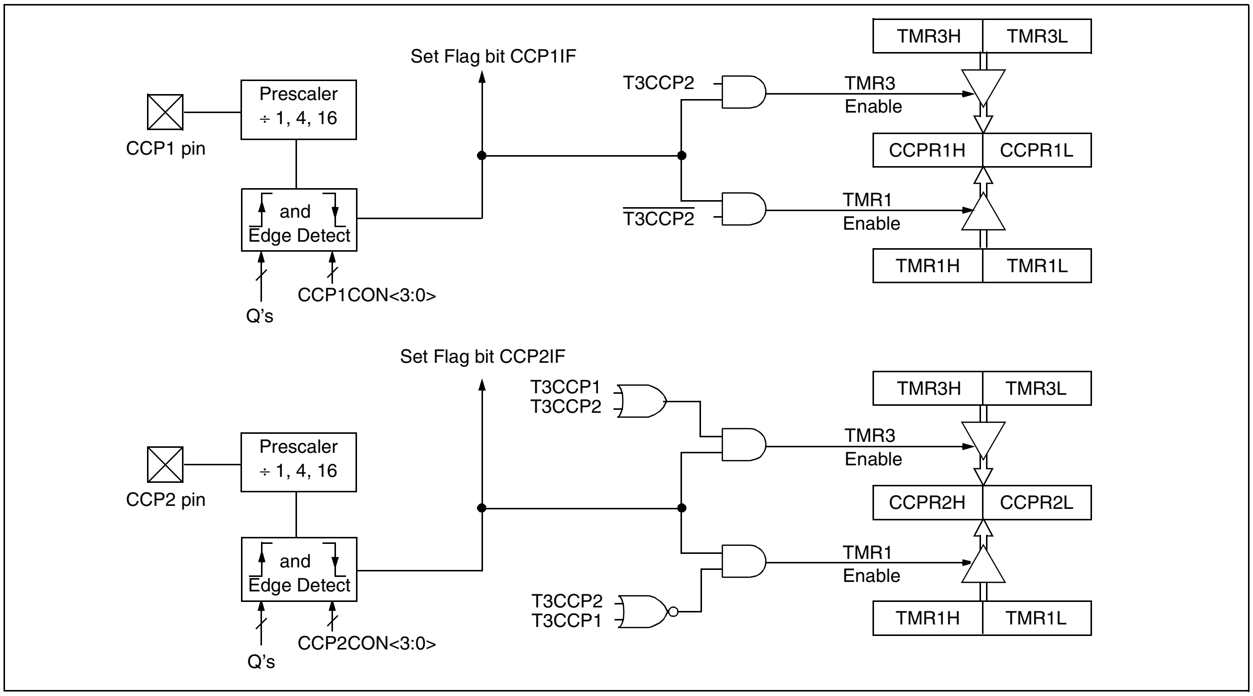
There are four prescaler settings, specified by bits CCP1M3:CCP1M0. Whenever the CCP module is turned off or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any RESET will clear the prescaler counter.

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 14-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 14-1: CHANGING BETWEEN CAPTURE PRESCALERS

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

FIGURE 14-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



15.3.3 ENABLING SPI I/O

To enable the serial port, SSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPCON registers, and then set the SSPEN bit. This configures the SDI, SDO, SCK, and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed. That is:

- SDI is automatically controlled by the SPI module
- SDO must have TRISC<5> bit cleared
- SCK (Master mode) must have TRISC<3> bit cleared
- SCK (Slave mode) must have TRISC<3> bit set
- $\overline{SS}$ must have TRISC<4> bit set

Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

15.3.4 TYPICAL CONNECTION

Figure 15-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge, and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data — Slave sends dummy data
- Master sends data — Slave sends data
- Master sends dummy data — Slave sends data

FIGURE 15-2: SPI MASTER/SLAVE CONNECTION

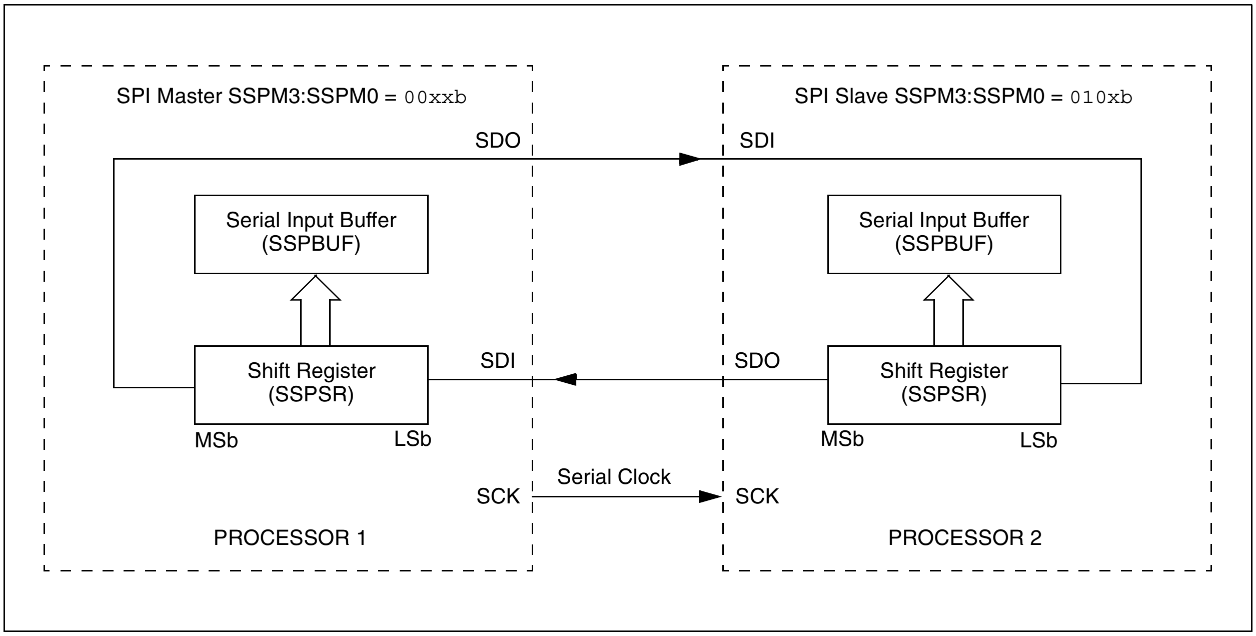

SPI Master SSPM3:SSPM0 = 00xxb
SDO
Serial Input Buffer (SSPBUF)
Shift Register (SSPSR)
SDI
MSb
LSb
SCK
PROCESSOR 1
Serial Clock
SPI Slave SSPM3:SSPM0 = 010xb
SDI
Serial Input Buffer (SSPBUF)
SDO
Shift Register (SSPSR)
MSb
LSb
SCK
PROCESSOR 2

FIGURE 15-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

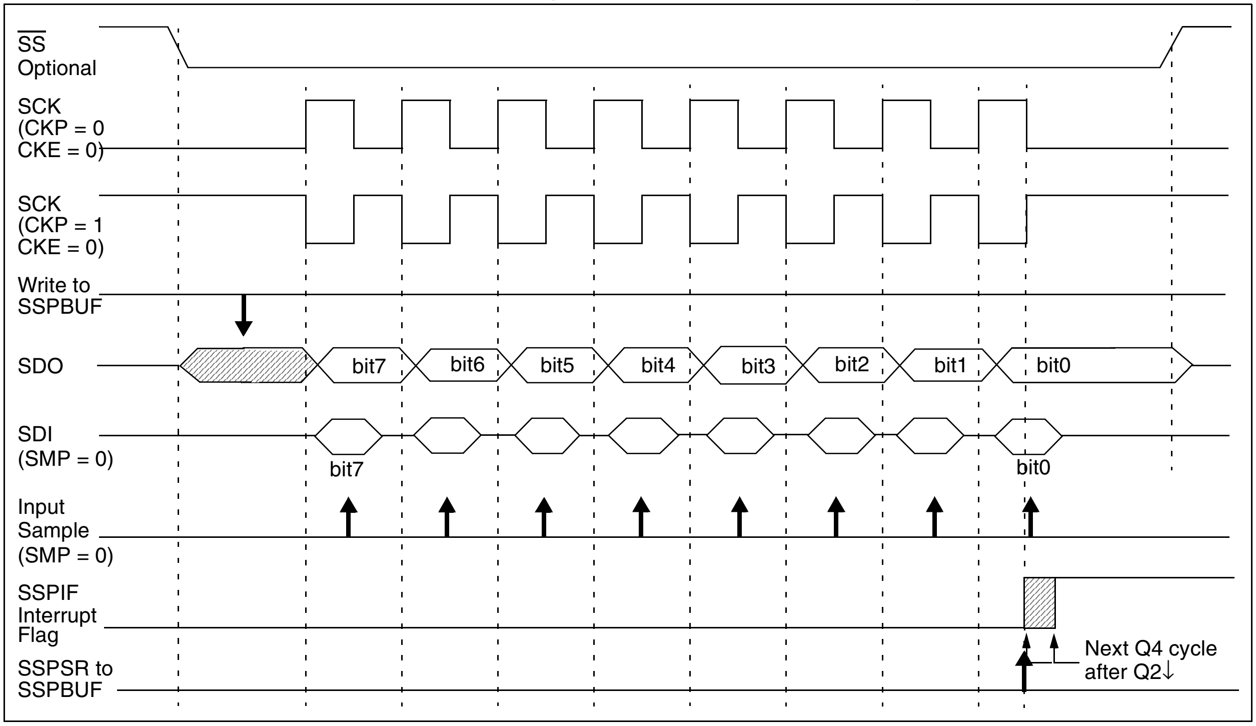
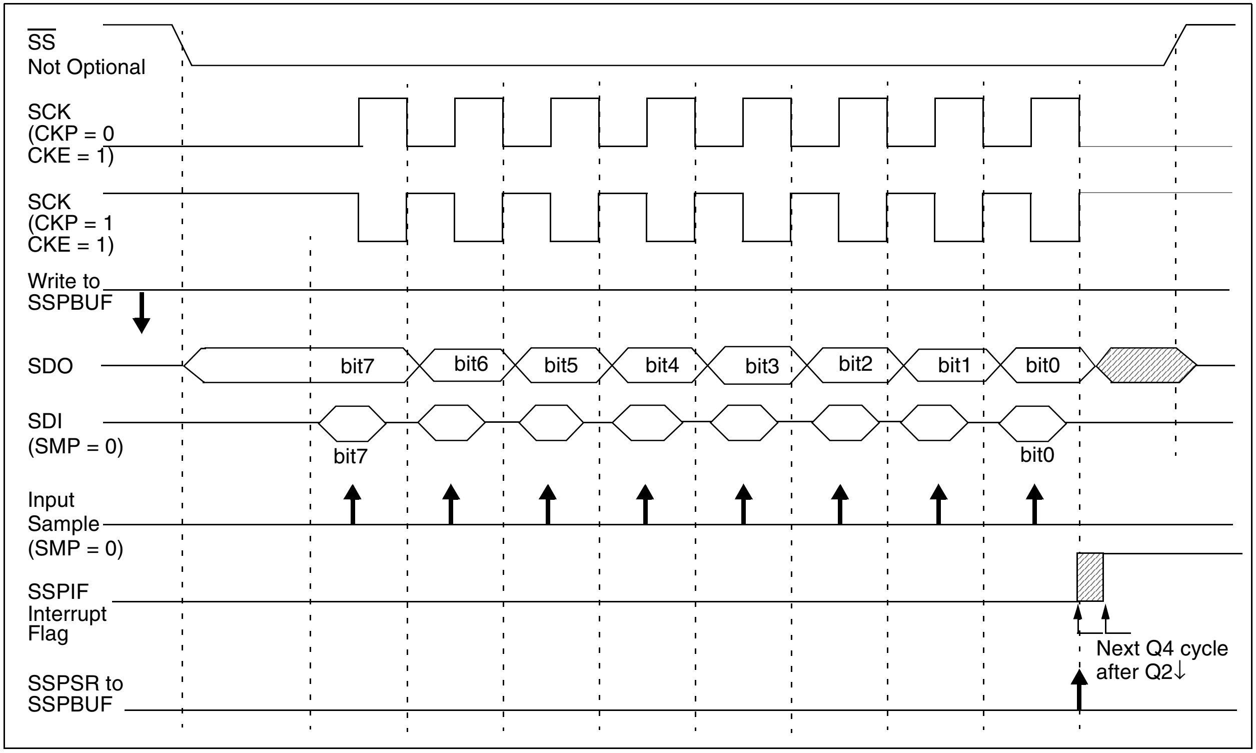


FIGURE 15-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



15.4 I²C Mode

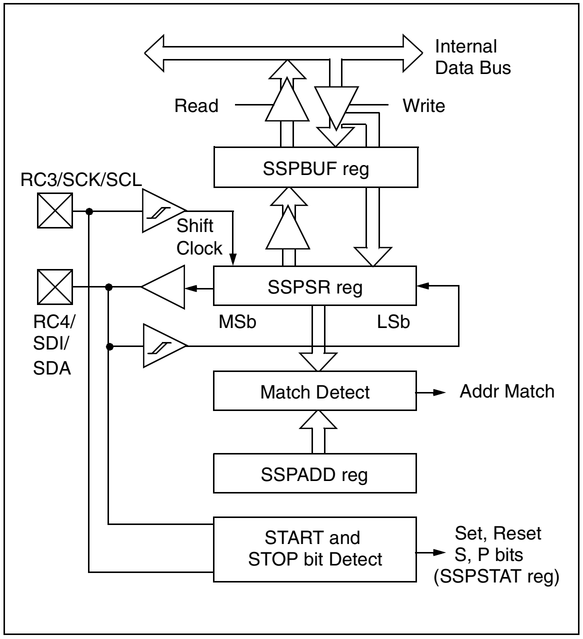
The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on START and STOP bits in hardware to determine a free bus (multi-master function). The MSSP module implements the Standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial clock (SCL) - RC3/SCK/SCL
- Serial data (SDA) - RC4/SDI/SDA

The user must configure these pins as inputs or outputs through the TRISC<4:3> bits.

FIGURE 15-7: MSSP BLOCK DIAGRAM (I²C MODE)



15.4.1 REGISTERS

The MSSP module has six registers for I²C operation. These are:

- MSSP Control Register1 (SSPCON1)
- MSSP Control Register2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- MSSP Shift Register (SSPSR) - Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON, SSPCON2 and SSPSTAT are the control and status registers in I²C mode operation. The SSPCON and SSPCON2 registers are readable and writable. The lower 6 bits of the SSPSTAT are read only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

SSPADD register holds the slave device address when the SSP is configured in I²C Slave mode. When the SSP is configured in Master mode, the lower seven bits of SSPADD act as the baud rate generator reload value.

In receive operations, SSPSR and SSPBUF together, create a double buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

REGISTER 15-3: SSPSTAT: MSSP STATUS REGISTER (I^2C MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF
bit 7							bit 0

bit 7 **SMP:** Slew Rate Control bit
In Master or Slave mode:
1 = Slew rate control disabled for Standard Speed mode (100 kHz and 1 MHz)
0 = Slew rate control enabled for High Speed mode (400 kHz)

bit 6 **CKE:** SMBus Select bit
In Master or Slave mode:
1 = Enable SMBus specific inputs
0 = Disable SMBus specific inputs

bit 5 **D/$\overline{\text{A}}$:** Data/$\overline{\text{Address}}$ bit
In Master mode:
Reserved

In Slave mode:
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** STOP bit
1 = Indicates that a STOP bit has been detected last
0 = STOP bit was not detected last

Note: This bit is cleared on RESET and when SSPEN is cleared.

bit 3 **S:** START bit
1 = Indicates that a start bit has been detected last
0 = START bit was not detected last

Note: This bit is cleared on RESET and when SSPEN is cleared.

bit 2 **R/$\overline{\text{W}}$:** Read/Write bit Information (I^2C mode only)
In Slave mode:
1 = Read
0 = Write

Note: This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next START bit, STOP bit, or not $\overline{\text{ACK}}$ bit.

In Master mode:
1 = Transmit is in progress
0 = Transmit is not in progress

Note: ORing this bit with SEN, RSEN, PEN, RCEN, or ACKEN will indicate if the MSSP is in IDLE mode.

bit 1 **UA:** Update Address (10-bit Slave mode only)
1 = Indicates that the user needs to update the address in the SSPADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit
In Transmit mode:
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty

In Receive mode:
1 = Data transmit in progress (does not include the $\overline{\text{ACK}}$ and STOP bits), SSPBUF is full
0 = Data transmit complete (does not include the $\overline{\text{ACK}}$ and STOP bits), SSPBUF is empty

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

REGISTER 19-2: CONFIGURATION REGISTER 2 LOW (CONFIG2L: BYTE ADDRESS 300002h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	BORV1	BORV0	BOREN	$\overline{\text{PWRTEN}}$
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3-2 **BORV1:BORV0:** Brown-out Reset Voltage bits

`11` = VBOR set to 2.5V
`10` = VBOR set to 2.7V
`01` = VBOR set to 4.2V
`00` = VBOR set to 4.5V

bit 1 **BOREN:** Brown-out Reset Enable bit

`1` = Brown-out Reset enabled
`0` = Brown-out Reset disabled

bit 0 **$\overline{\text{PWRTEN}}$:** Power-up Timer Enable bit

`1` = PWRT disabled
`0` = PWRT enabled

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 19-3: CONFIGURATION REGISTER 2 HIGH (CONFIG2H: BYTE ADDRESS 300003h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3-1 **WDTPS2:WDTPS0:** Watchdog Timer Postscale Select bits

`111` = 1:128
`110` = 1:64
`101` = 1:32
`100` = 1:16
`011` = 1:8
`010` = 1:4
`001` = 1:2
`000` = 1:1

bit 0 **WDTEN:** Watchdog Timer Enable bit

`1` = WDT enabled
`0` = WDT disabled (control is placed on the SWDTEN bit)

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 19-10: CONFIGURATION REGISTER 7 LOW (CONFIG7L: BYTE ADDRESS 30000Ch)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3(1)	EBTR2(1)	EBTR1	EBTR0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3 **EBTR3:** Table Read Protection bit(1)
1 = Block 3 (006000-007FFFh) not protected from Table Reads executed in other blocks
0 = Block 3 (006000-007FFFh) protected from Table Reads executed in other blocks

bit 2 **EBTR2:** Table Read Protection bit(1)
1 = Block 2 (004000-005FFFh) not protected from Table Reads executed in other blocks
0 = Block 2 (004000-005FFFh) protected from Table Reads executed in other blocks

bit 1 **EBTR1:** Table Read Protection bit
1 = Block 1 (002000-003FFFh) not protected from Table Reads executed in other blocks
0 = Block 1 (002000-003FFFh) protected from Table Reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit
1 = Block 0 (000200h-001FFFh) not protected from Table Reads executed in other blocks
0 = Block 0 (000200h-001FFFh) protected from Table Reads executed in other blocks

Note 1: Unimplemented in PIC18FX42 devices; maintain this bit set.

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 19-11: CONFIGURATION REGISTER 7 HIGH (CONFIG7H: BYTE ADDRESS 30000Dh)

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
bit 7							bit 0

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **EBTRB:** Boot Block Table Read Protection bit
1 = Boot Block (000000-0001FFh) not protected from Table Reads executed in other blocks
0 = Boot Block (000000-0001FFh) protected from Table Reads executed in other blocks

bit 5-0 **Unimplemented:** Read as ‘0’

Legend:		
R = Readable bit	C =Clearable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 19-12: DEVICE ID REGISTER 1 FOR PIC18FXX2 (DEVID1: BYTE ADDRESS 3FFFFEh)

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

bit 7-5 **DEV2:DEV0:** Device ID bits

`000` = PIC18F252
`001` = PIC18F452
`100` = PIC18F242
`101` = PIC18F442

bit 4-0 **REV4:REV0:** Revision ID bits

These bits are used to indicate the device revision.

Legend:		
R = Readable bit	P =Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 19-13: DEVICE ID REGISTER 2 FOR PIC18FXX2 (DEVID2: BYTE ADDRESS 3FFFFFh)

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3:** Device ID bits

These bits are used with the DEV2:DEV0 bits in the Device ID Register 1 to identify the part number.

Legend:		
R = Readable bit	P =Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

NOTES:

FIGURE 23-11: TYPICAL AND MAXIMUM IDD vs. VDD (TIMER1 AS MAIN OSCILLATOR, 32.768 kHz, C1 AND C2 = 47 pF)

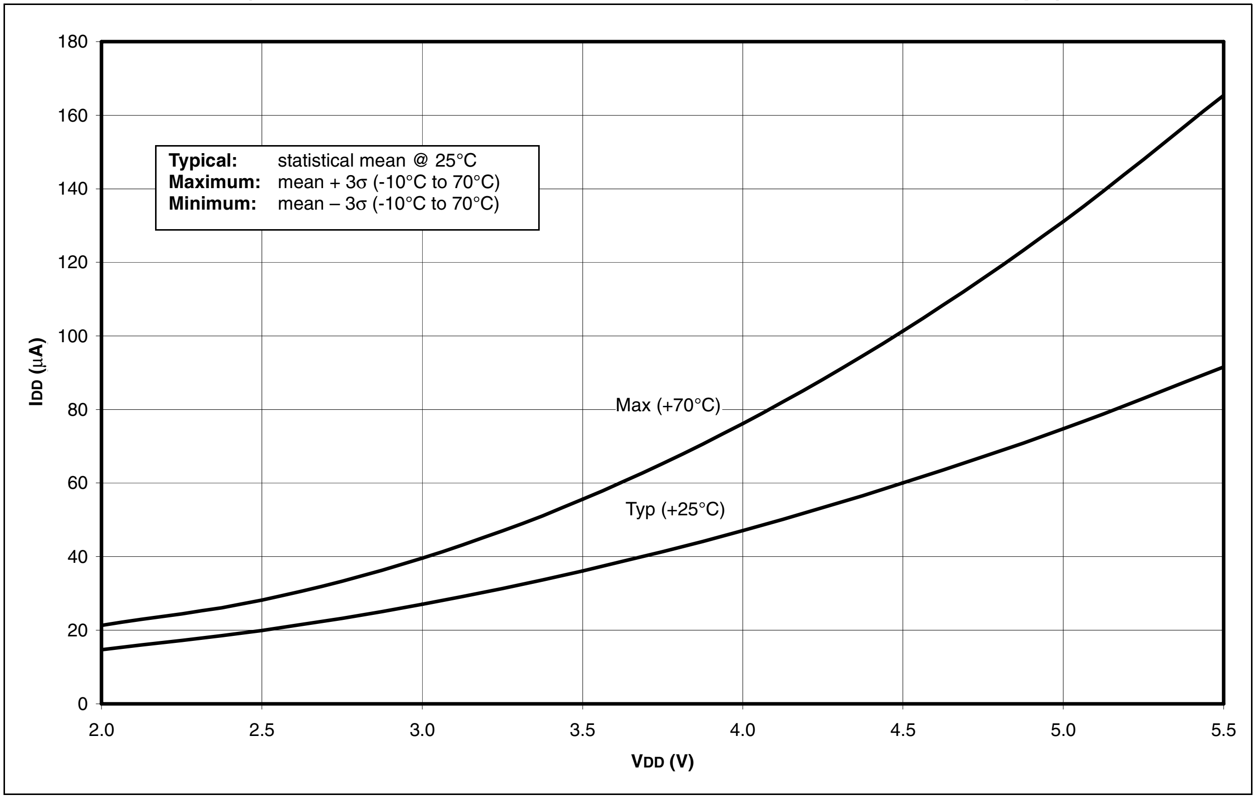


FIGURE 23-12: AVERAGE FOSC vs. VDD FOR VARIOUS VALUES OF R (RC MODE, C = 20 pF, +25°C)

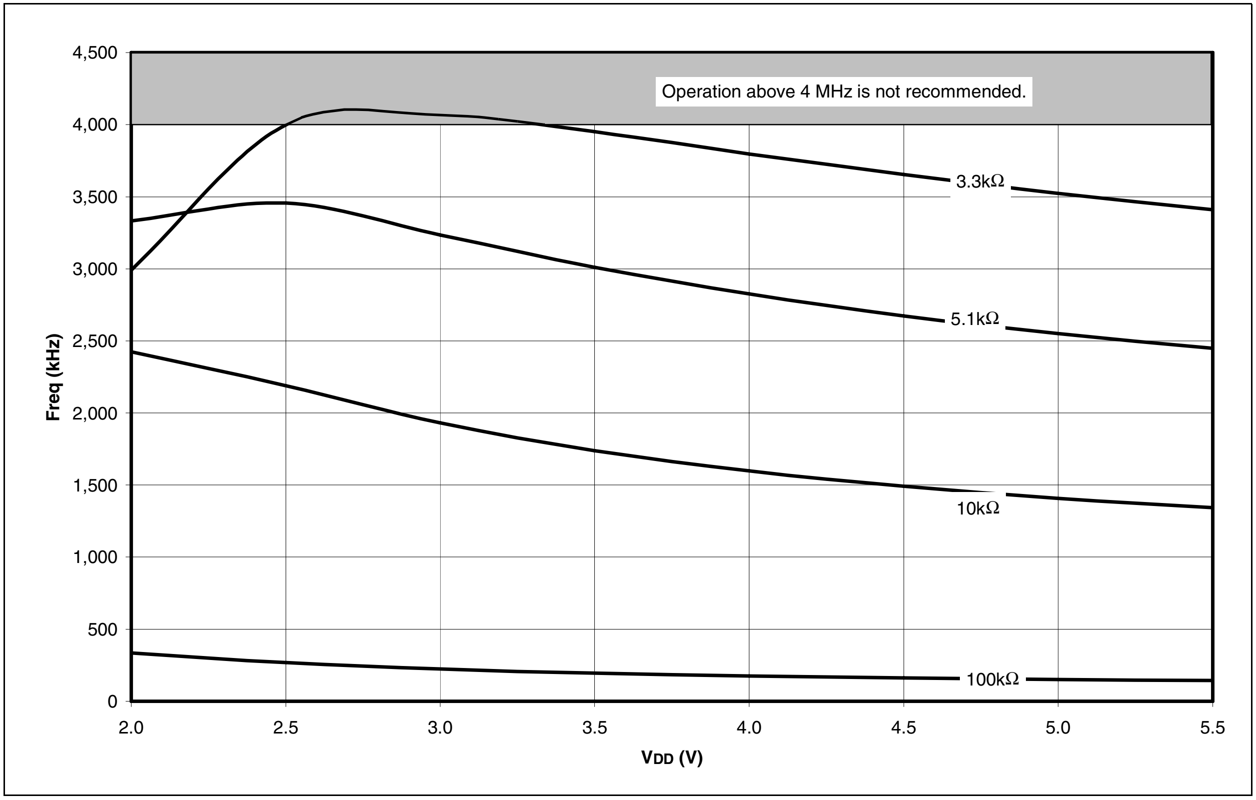


FIGURE 23-17: TYPICAL AND MAXIMUM ΔITMR1 vs. VDD OVER TEMPERATURE (-10°C TO +70°C, TIMER1 WITH OSCILLATOR, XTAL = 32 kHz, C1 AND C2 = 47 pF)

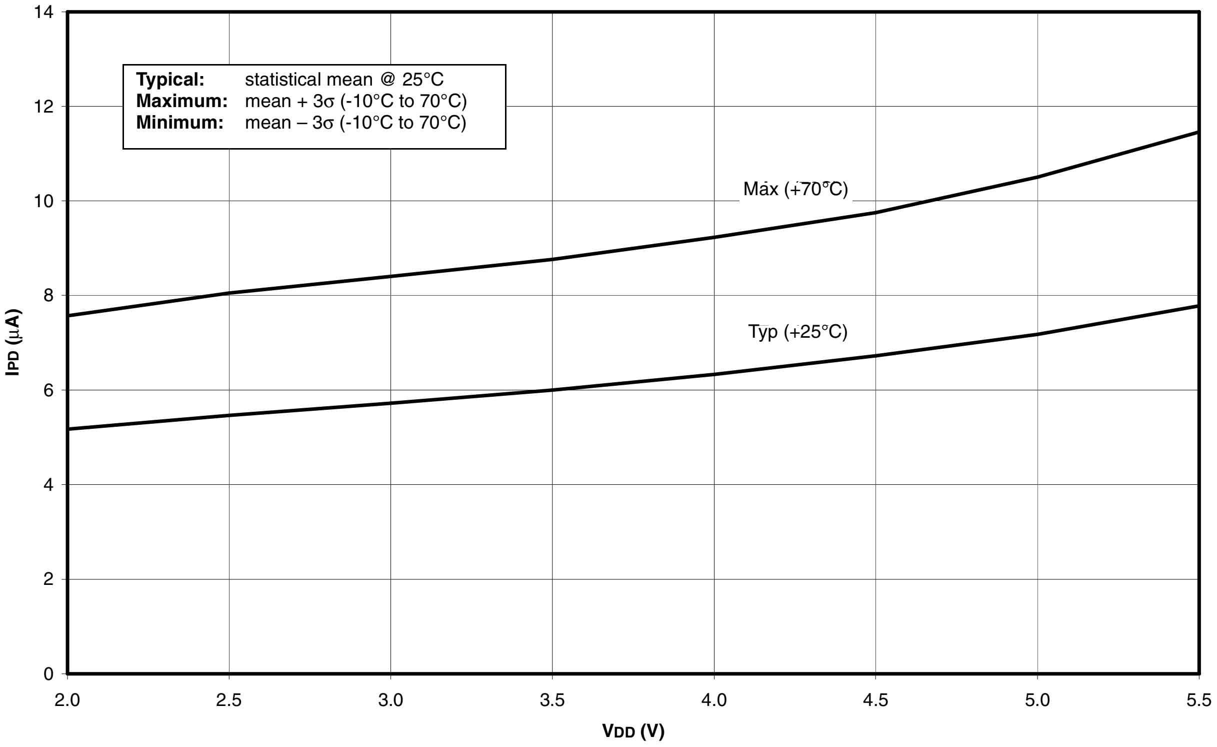
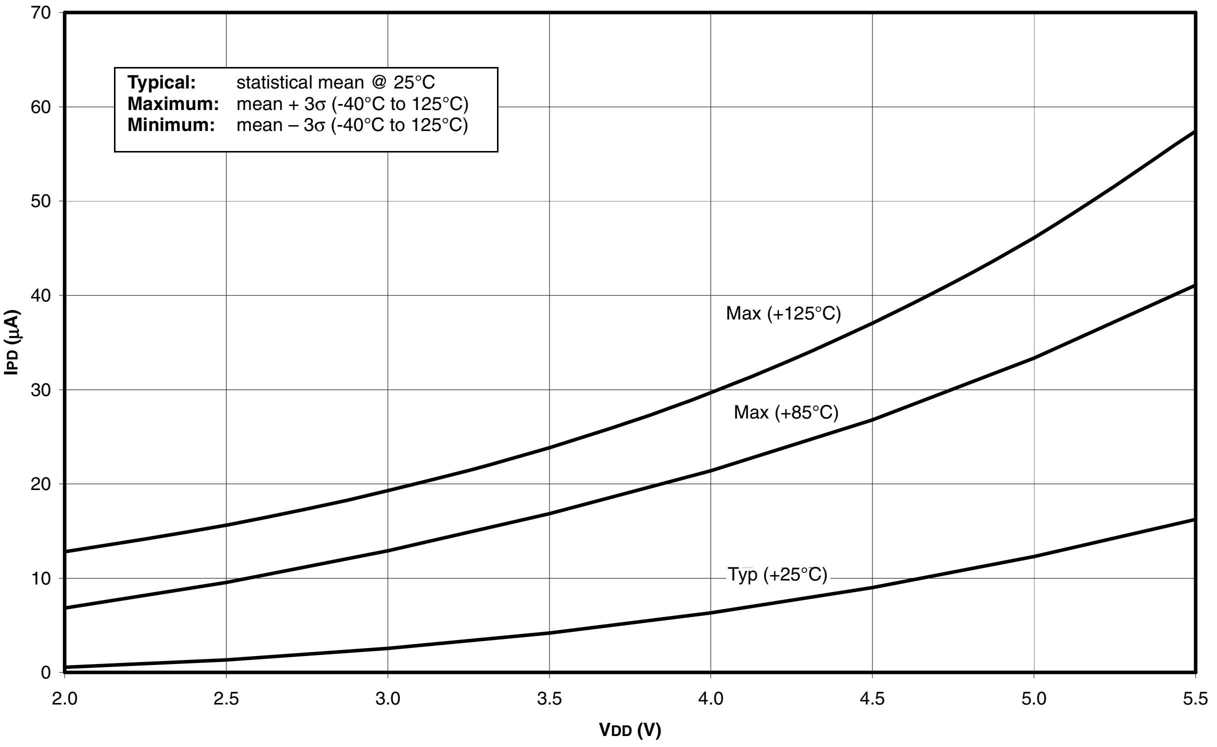


FIGURE 23-18: TYPICAL AND MAXIMUM ΔIWDT vs. VDD OVER TEMPERATURE (WDT ENABLED)


Typical: statistical mean @ 25°C
Maximum: mean + 3σ (-40°C to 125°C)
Minimum: mean – 3σ (-40°C to 125°C)
Max (+125°C)
Max (+85°C)
Typ (+25°C)
IPD (μA)
VDD (V)