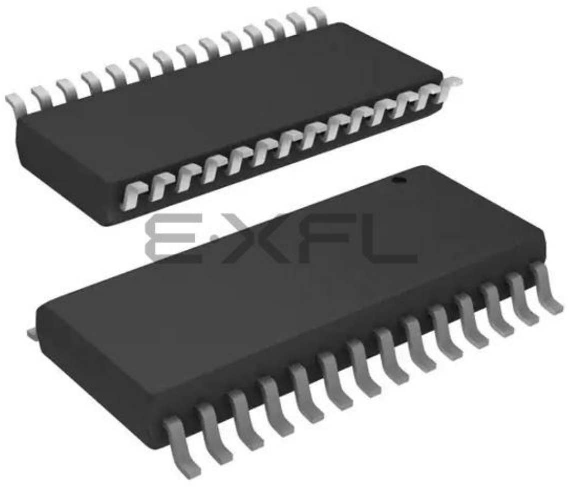
Microchip Technology - PIC18C252-E/SO Datasheet



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	Obsolete
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	22
Program Memory Size	32KB (16K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18c252-e-so

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:

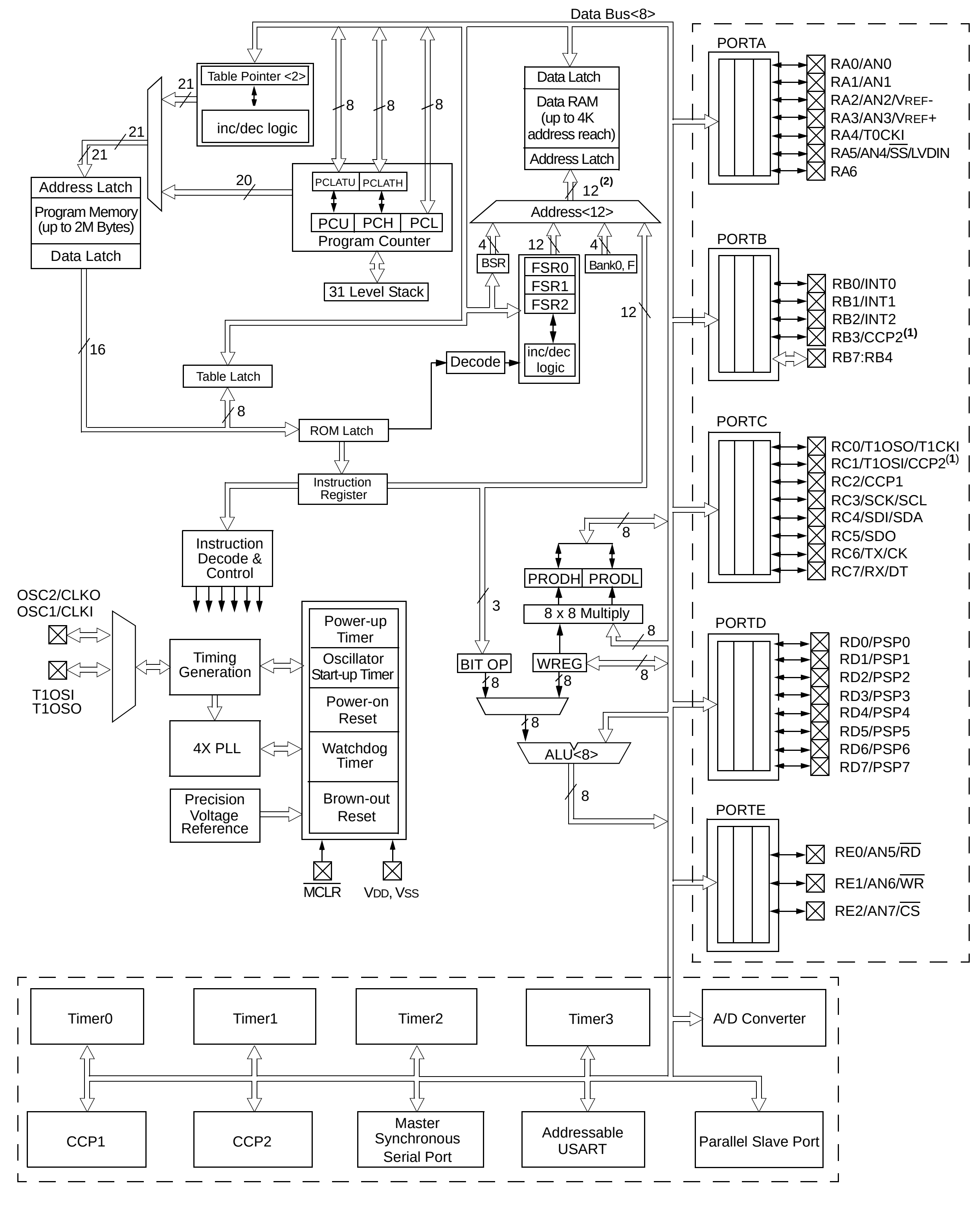
- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

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.

FIGURE 1-2: PIC18C4X2 BLOCK DIAGRAM



Note	**1:**	Optional multiplexing of CCP2 input/output with RB3 is enabled by selection of configuration bit.
	2:	The high order bits of the Direct Address for the RAM are from the BSR register (except for the `MOVFF` instruction).
	3:	Many of the general purpose I/O pins are multiplexed with one or more peripheral module functions. The multiplexing combinations are device dependent.

FIGURE 3-6: SLOW RISE TIME ($\overline{\text{MCLR}}$ TIED TO VDD)

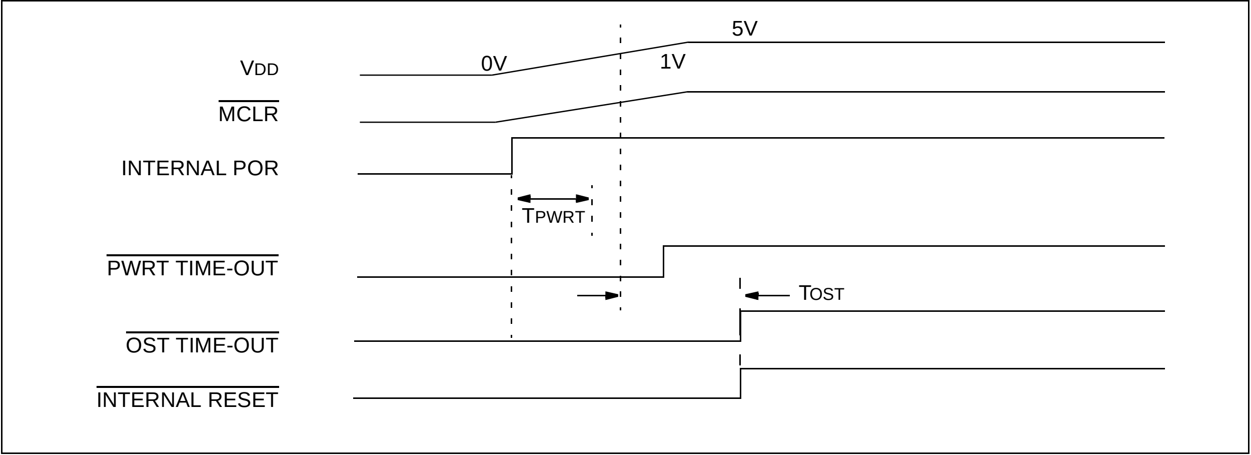
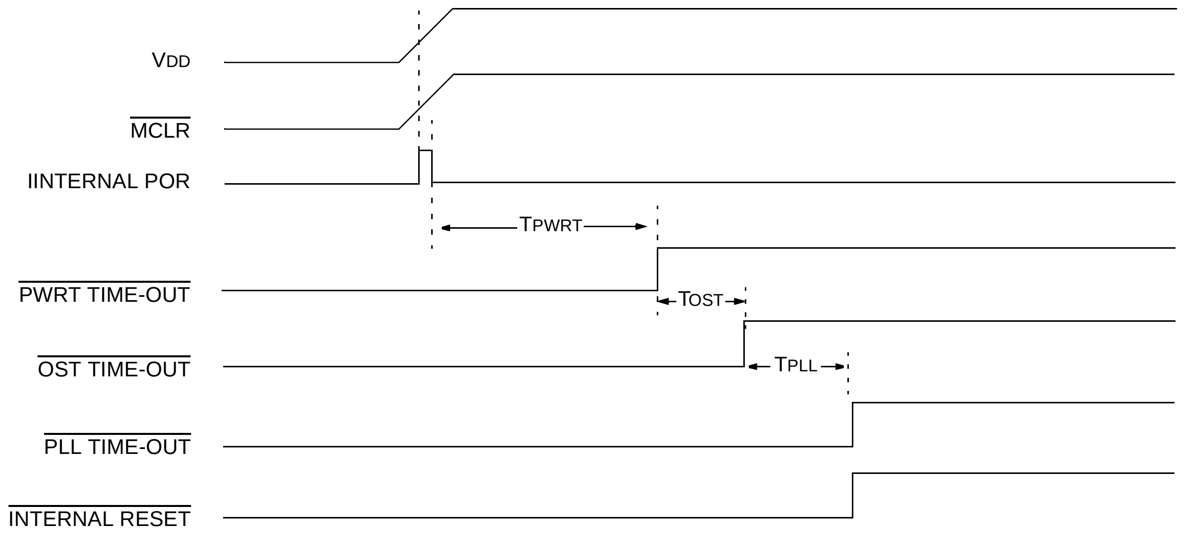

VDD
$\overline{\text{MCLR}}$
INTERNAL POR
$\overline{\text{PWRT TIME-OUT}}$
$\overline{\text{OST TIME-OUT}}$
$\overline{\text{INTERNAL RESET}}$
0V
1V
5V
TPWRT
TOST

FIGURE 3-7: TIME-OUT SEQUENCE ON POR W/ PLL ENABLED ($\overline{\text{MCLR}}$ TIED TO VDD)


VDD
$\overline{\text{MCLR}}$
IINTERNAL POR
$\overline{\text{PWRT TIME-OUT}}$
$\overline{\text{OST TIME-OUT}}$
$\overline{\text{PLL TIME-OUT}}$
$\overline{\text{INTERNAL RESET}}$
TPWRT
TOST
TPLL

Note: TOST = 1024 clock cycles.
TPLL ≈ 2 ms max. First three stages of the PWRT timer.

4.10 Access Bank

The Access Bank is an architectural enhancement, which is very useful for C compiler code optimization. The techniques used by the C compiler may also be useful for programs written in assembly.

This data memory region can be used for:

- Intermediate computational values
- Local variables of subroutines
- Faster context saving/switching of variables
- Common variables
- Faster evaluation/control of SFRs (no banking)

The Access Bank is comprised of the upper 128 bytes in Bank 15 (SFRs) and the lower 128 bytes in Bank 0. These two sections will be referred to as Access RAM High and Access RAM Low, respectively. Figure 4-6 and Figure 4-7 indicate the Access RAM areas.

A bit in the instruction word specifies if the operation is to occur in the bank specified by the BSR register or in the Access Bank. This bit is denoted by the ’a’ bit (for access bit).

When forced in the Access Bank (a = ’0’), the last address in Access RAM Low is followed by the first address in Access RAM High. Access RAM High maps the Special Function registers, so that these registers can be accessed without any software overhead. This is useful for testing status flags and modifying control bits.

4.11 Bank Select Register (BSR)

The need for a large general purpose memory space dictates a RAM banking scheme. The data memory is partitioned into sixteen banks. When using direct addressing, the BSR should be configured for the desired bank.

BSR<3:0> holds the upper 4 bits of the 12-bit RAM address. The BSR<7:4> bits will always read ’0’s, and writes will have no effect.

A `MOVLB` instruction has been provided in the instruction set to assist in selecting banks.

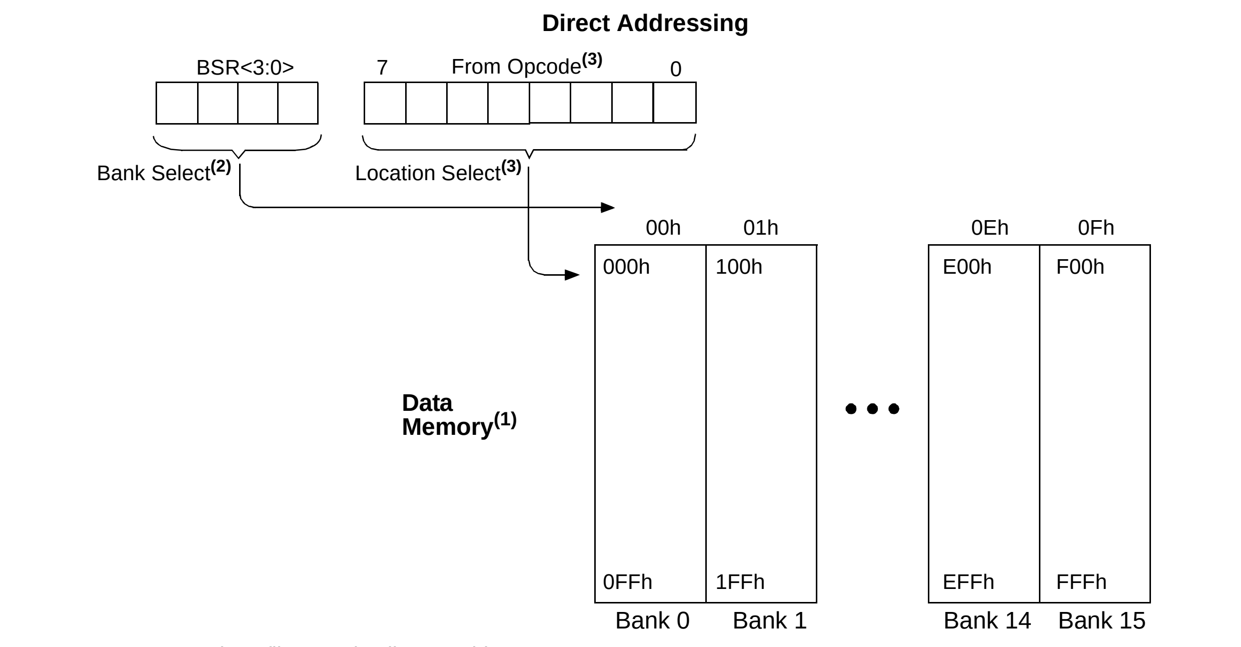
If the currently selected bank is not implemented, any read will return all '0's and all writes are ignored. The STATUS register bits will be set/cleared as appropriate for the instruction performed.

Each Bank extends up to FFh (256 bytes). All data memory is implemented as static RAM.

A `MOVFF` instruction ignores the BSR, since the 12-bit addresses are embedded into the instruction word.

Section 4.12 provides a description of indirect addressing, which allows linear addressing of the entire RAM space.

FIGURE 4-8: DIRECT ADDRESSING



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

2: The access bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

3: The `MOVFF` instruction embeds the entire 12-bit address in the instruction.

If an indirect addressing operation is done where the target address is an FSRnH or FSRnL register, the write operation will dominate over the pre- or post-increment/decrement functions.

FIGURE 4-9: INDIRECT ADDRESSING OPERATION

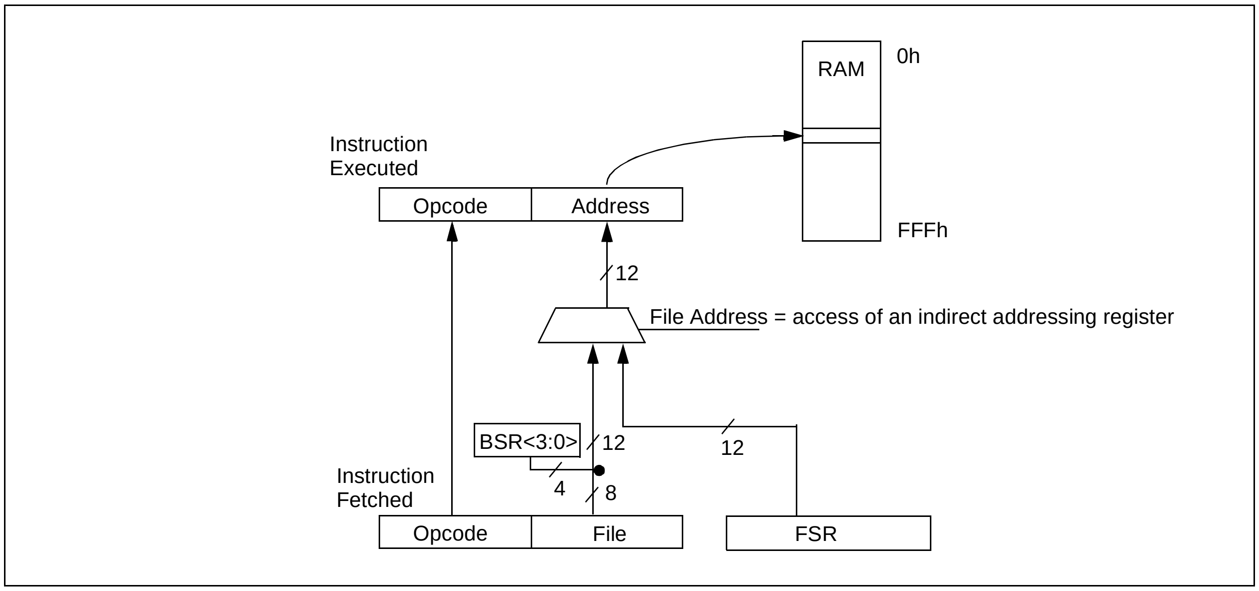
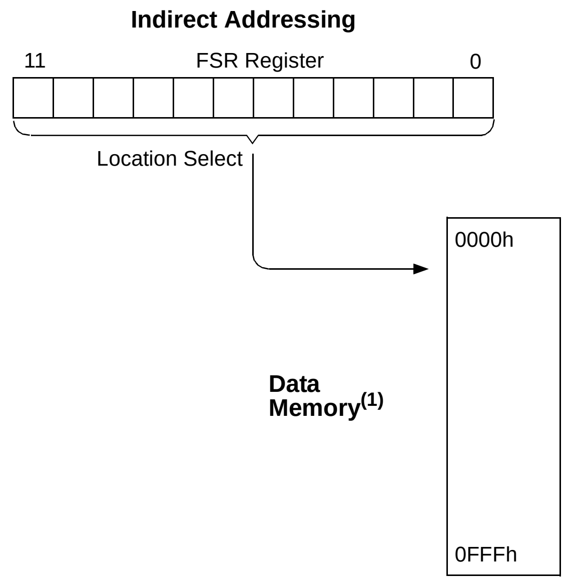


FIGURE 4-10: INDIRECT ADDRESSING



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

TABLE 8-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/INT0	bit0	TTL/ST(1)	Input/output pin or external interrupt input1. Internal software programmable weak pull-up.
RB1/INT1	bit1	TTL/ST(1)	Input/output pin or external interrupt input2. Internal software programmable weak pull-up.
RB2/INT2	bit2	TTL/ST(1)	Input/output pin or external interrupt input3. Internal software programmable weak pull-up.
RB3/CCP2(3)	bit3	TTL/ST(4)	Input/output pin. Capture2 input/Compare2 output/PWM output when CCP2MX configuration bit is enabled. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6	bit6	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock.
RB7	bit7	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.

2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

3: A device configuration bit selects which I/O pin the CCP2 pin is multiplexed on.

4: This buffer is a Schmitt Trigger input when configured as the CCP2 input.

TABLE 8-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

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
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	`xxxx xxxx`	`uuuu uuuu`
LATB	LATB Data Output Register									
TRISB	PORTB Data Direction Register								`1111 1111`	`1111 1111`
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	`0000 000x`	`0000 000u`
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	`1111 -1-1`	`1111 -1-1`
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	`11-0 0-00`	`11-0 0-00`

Legend: `x` = unknown, `u` = unchanged. Shaded cells are not used by PORTB.

TABLE 8-7: PORTD FUNCTIONS

Name	Bit#	Buffer Type	Function
RD0/PSP0	bit0	ST/TTL[(1)]	Input/output port pin or parallel slave port bit0.
RD1/PSP1	bit1	ST/TTL[(1)]	Input/output port pin or parallel slave port bit1.
RD2/PSP2	bit2	ST/TTL[(1)]	Input/output port pin or parallel slave port bit2.
RD3/PSP3	bit3	ST/TTL[(1)]	Input/output port pin or parallel slave port bit3.
RD4/PSP4	bit4	ST/TTL[(1)]	Input/output port pin or parallel slave port bit4.
RD5/PSP5	bit5	ST/TTL[(1)]	Input/output port pin or parallel slave port bit5.
RD6/PSP6	bit6	ST/TTL[(1)]	Input/output port pin or parallel slave port bit6.
RD7/PSP7	bit7	ST/TTL[(1)]	Input/output port pin or parallel slave port bit7.

Legend: ST = Schmitt Trigger input, TTL = TTL input

Note 1: Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in Parallel Slave Port mode.

TABLE 8-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

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
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	`xxxx xxxx`	`uuuu uuuu`
LATD	LATD Data Output Register								`xxxx xxxx`	`uuuu uuuu`
TRISD	PORTD Data Direction Register								`1111 1111`	`1111 1111`
TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction bits			`0000 -111`	`0000 -111`

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

REGISTER 8-1: TRISE REGISTER

R-0	R-0	R/W-0	R/W-0	U-0	R/W-1	R/W-1	R/W-1
IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0
bit 7							bit 0

bit 7	**IBF:** Input Buffer Full Status bit 1 = A word has been received and waiting to be read by the CPU 0 = No word has been received
bit 6	**OBF**: Output Buffer Full Status bit 1 = The output buffer still holds a previously written word 0 = The output buffer has been read
bit 5	**IBOV**: Input Buffer Overflow Detect bit (in Microprocessor mode) 1 = A write occurred when a previously input word has not been read (must be cleared in software) 0 = No overflow occurred
bit 4	**PSPMODE**: Parallel Slave Port Mode Select bit 1 = Parallel Slave Port mode 0 = General purpose I/O mode
bit 3	**Unimplemented:** Read as '0'
bit 2	**TRISE2**: RE2 Direction Control bit 1 = Input 0 = Output
bit 1	**TRISE1**: RE1 Direction Control bit 1 = Input 0 = Output
bit 0	**TRISE0**: RE0 Direction Control bit 1 = Input 0 = Output

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

9.1 Timer0 Operation

Timer0 can operate as a timer or as a counter.

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

Counter mode is selected by setting the T0CS bit. In Counter mode, Timer0 will increment either on every rising, or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit (T0SE). Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed below.

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

9.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not readable or writable.

The PSA and T0PS2:T0PS0 bits determine the prescaler assignment and prescale ratio.

Clearing bit PSA will assign the prescaler to the Timer0 module. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4,..., 1:256 are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g. `CLRF TMR0, MOVWF TMR0, BSF TMR0, x`....etc.) will clear the prescaler count.

Note: Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count, but will not change the prescaler assignment.

9.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed "on-the-fly" during program execution).

9.3 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF bit. The interrupt can be masked by clearing the TMR0IE bit. The TMR0IE bit must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from SLEEP, since the timer is shut-off during SLEEP.

9.4 16-Bit Mode Timer Reads and Writes

TMR0H is not the high byte of the timer/counter in 16-bit mode, but is actually a buffered version of the high byte of Timer0 (refer to Figure 9-2). The high byte of the Timer0 counter/timer is not directly readable nor writable. TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16-bits of Timer0 without having to verify that the read of the high and low byte were valid due to a rollover between successive reads of the high and low byte.

A write to the high byte of Timer0 must also take place through the TMR0H buffer register. Timer0 high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16-bits of Timer0 to be updated at once.

TABLE 9-1: REGISTERS ASSOCIATED WITH TIMER0

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
TMR0L	Timer0 Module's Low Byte Register								`xxxx xxxx`	`uuuu uuuu`
TMR0H	Timer0 Module's High Byte Register								`0000 0000`	`0000 0000`
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	`0000 000x`	`0000 000u`
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	`1111 1111`	`1111 1111`
TRISA	—	—	PORTA Data Direction Register						`--11 1111`	`--11 1111`

Legend: `x` = unknown, `u` = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by Timer0.

12.2 Timer1 Oscillator

The Timer1 oscillator may be used as the clock source for Timer3. The Timer1 oscillator is enabled by setting the T1OSCEN (T1CON<3>) bit. The oscillator is a low power oscillator rated up to 200 KHz. See Section 10.0 for further details.

12.3 Timer3 Interrupt

The TMR3 Register pair (TMR3H:TMR3L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR3 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit TMR3IF (PIR2<1>). This interrupt can be enabled/disabled by setting/clearing TMR3 interrupt enable bit, TMR3IE (PIE2<1>).

12.4 Resetting Timer3 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a “special event trigger” (CCP1M3:CCP1M0 = `1011`), this signal will reset Timer3.

Note: The special event triggers from the CCP module will not set interrupt flag bit TMR3IF (PIR1<0>).

Timer3 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer3 is running in Asynchronous Counter mode, this RESET operation may not work. In the event that a write to Timer3 coincides with a special event trigger from CCP1, the write will take precedence. In this mode of operation, the CCPR1H:CCPR1L registers pair effectively becomes the period register for Timer3.

TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER3 AS A TIMER/COUNTER

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
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	`0000 000x`	`0000 000u`
PIR2	—	—	—	—	BCLIF	LVDIF	TMR3IF	CCP2IF	`0000 0000`	`0000 0000`
PIE2	—	—	—	—	BCLIE	LVDIE	TMR3IE	CCP2IE	`0000 0000`	`0000 0000`
IPR2	—	—	—	—	BCLIP	LVDIP	TMR3IP	CCP2IP	`0000 0000`	`0000 0000`
TMR3L	Holding Register for the Least Significant Byte of the 16-bit TMR3 Register								`xxxx xxxx`	`uuuu uuuu`
TMR3H	Holding Register for the Most Significant Byte of the 16-bit TMR3 Register								`xxxx xxxx`	`uuuu uuuu`
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	`--00 0000`	`--uu uuuu`
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	`-000 0000`	`-uuu uuuu`

Legend: `x` = unknown, `u` = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

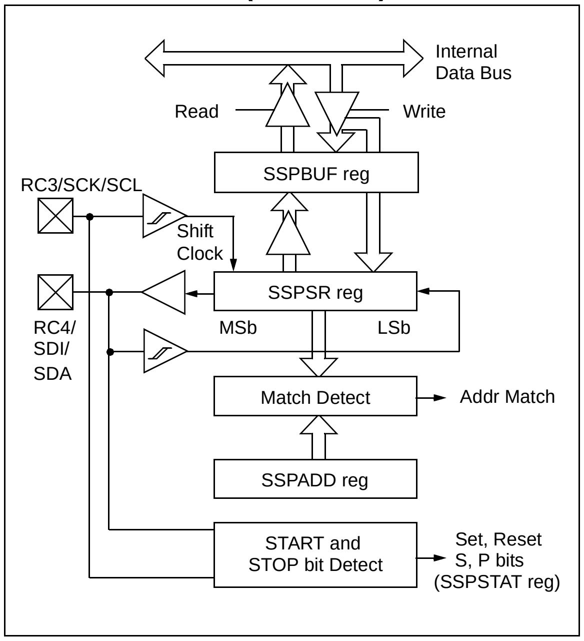
14.4 MSSP I²C Operation

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. These are the RC3/SCK/SCL pin, which is the clock (SCL), and the RC4/SDI/SDA pin, which is the data (SDA). The user must configure these pins as inputs or outputs through the TRISC<4:3> bits.

The MSSP module functions are enabled by setting MSSP enable bit SSPEN (SSPCON<5>).

FIGURE 14-7: MSSP BLOCK DIAGRAM (I²C MODE)



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

- 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)

The SSPCON1 register allows control of the I²C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I²C modes to be selected:

- I²C Master mode, clock = OSC/4 (SSPADD +1)
- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address), with START and STOP bit interrupts enabled
- I²C Slave mode (10-bit address), with START and STOP bit interrupts enabled
- I²C Firmware controlled master operation, slave is idle

Selection of any I²C mode with the SSPEN bit set, forces the SCL and SDA pins to be open drain, provided these pins are programmed to be inputs by setting the appropriate TRISC bits.

14.4.1 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISC<4:3> set). The MSSP module will override the input state with the output data when required (slave-transmitter).

When an address is matched or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{\text{ACK}}$) pulse and load the SSPBUF register with the received value currently in the SSPSR register.

There are certain conditions that will cause the MSSP module not to give this $\overline{\text{ACK}}$ pulse. These are if either (or both):

a) The buffer full bit BF (SSPSTAT<0>) was set before the transfer was received.

b) The overflow bit SSPOV (SSPCON<6>) was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit SSPIF (PIR1<3>) is set. The BF bit is cleared by reading the SSPBUF register, while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I²C specification, as well as the requirement of the MSSP module, are shown in timing parameter #100 and parameter #101.

14.4.1.1 Addressing

Once the MSSP module has been enabled, it waits for a START condition to occur. Following the START condition, the 8-bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match, and the BF and SSPOV bits are clear, the following events occur:

a) The SSPSR register value is loaded into the SSPBUF register.
b) The buffer full bit BF is set.
c) An $\overline{\text{ACK}}$ pulse is generated.
d) MSSP interrupt flag bit SSPIF (PIR1<3>) is set (interrupt is generated if enabled) on the falling edge of the ninth SCL pulse.

In 10-bit address mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{\text{W}}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '`1111 0 A9 A8 0`', where `A9` and `A8` are the two MSbs of the address. The sequence of events for 10-bit address is as follows, with steps 7-9 for slave-transmitter:

1. Receive first (high) byte of Address (bits SSPIF, BF and bit UA (SSPSTAT<1>) are set).
2. Update the SSPADD register with second (low) byte of Address (clears bit UA and releases the SCL line).
3. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
4. Receive second (low) byte of Address (bits SSPIF, BF, and UA are set).
5. Update the SSPADD register with the first (high) byte of Address. If match releases SCL line, this will clear bit UA.
6. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
7. Receive Repeated START condition.
8. Receive first (high) byte of Address (bits SSPIF and BF are set).
9. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.

14.4.1.2 Reception

When the R/$\overline{\text{W}}$ bit of the address byte is clear and an address match occurs, the R/$\overline{\text{W}}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register.

When the address byte overflow condition exists, then no Acknowledge ($\overline{\text{ACK}}$) pulse is given. An overflow condition is defined as either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON<6>) is set.

An MSSP interrupt is generated for each data transfer byte. Flag bit SSPIF (PIR1<3>) must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

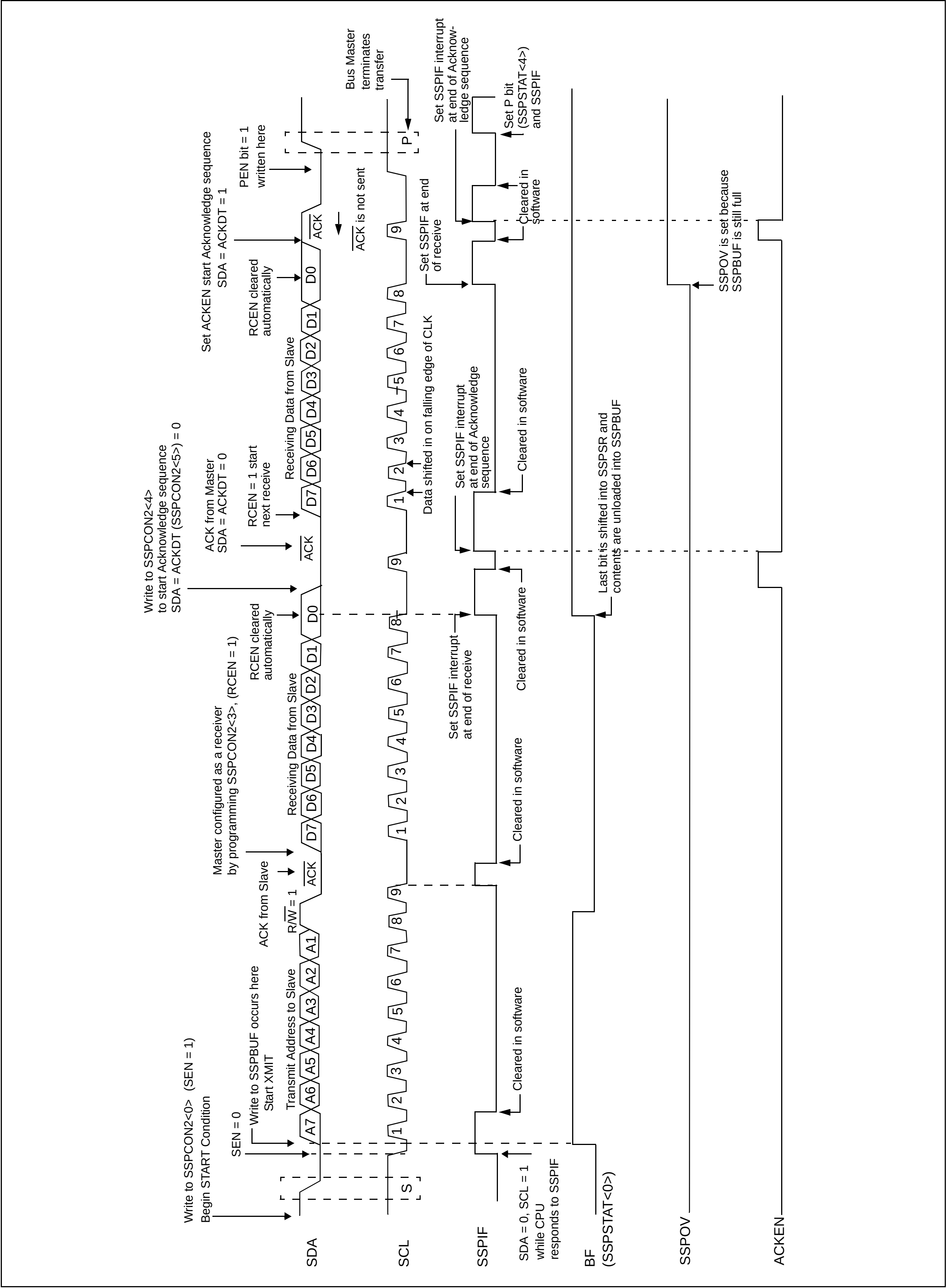
14.4.1.3 Transmission

When the R/$\overline{\text{W}}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{\text{W}}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{\text{ACK}}$ pulse will be sent on the ninth bit and pin RC3/SCK/SCL is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then pin RC3/SCK/SCL should be enabled by setting bit CKP (SSPCON<4>). The master must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 14-9).

An MSSP interrupt is generated for each data transfer byte. The SSPIF bit must be cleared in software and the SSPSTAT register is used to determine the status of the byte. The SSPIF bit is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{\text{ACK}}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not $\overline{\text{ACK}}$), then the data transfer is complete. When the $\overline{\text{ACK}}$ is latched by the slave, the slave logic is reset (resets SSPSTAT register) and the slave monitors for another occurrence of the START bit. If the SDA line was low ($\overline{\text{ACK}}$), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Pin RC3/SCK/SCL should be enabled by setting bit CKP.

FIGURE 14-19: I^{2}C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)


SDA
SCL
SSPIF
SDA = 0, SCL = 1 while CPU responds to SSPIF
BF (SSPSTAT<0>)
SSPOV
ACKEN
Write to SSPCON2<0> (SEN = 1) Begin START Condition
SEN = 0
Write to SSPBUF occurs here Start XMIT
Transmit Address to Slave
ACK from Slave
R/W = 1
Master configured as a receiver by programming SSPCON2<3>, (RCEN = 1)
Receiving Data from Slave
RCEN cleared automatically
Write to SSPCON2<4> to start Acknowledge sequence SDA = ACKDT (SSPCON2<5>) = 0
ACK from Master SDA = ACKDT = 0
RCEN = 1 start next receive
Set ACKEN start Acknowledge sequence SDA = ACKDT = 1
PEN bit = 1 written here
Bus Master terminates transfer
ACK is not sent
S
P
A7 A6 A5 A4 A3 A2 A1
ACK
D7 D6 D5 D4 D3 D2 D1 D0
1 2 3 4 5 6 7 8 9
Data shifted in on falling edge of CLK
Set SSPIF interrupt at end of receive
Set SSPIF interrupt at end of Acknowledge sequence
Set SSPIF at end of receive
Set SSPIF interrupt at end of Acknow-ledge sequence
Set P bit (SSPSTAT<4>) and SSPIF
Cleared in software
Last bit is shifted into SSPSR and contents are unloaded into SSPBUF
SSPOV is set because SSPBUF is still full

15.3 USART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner, (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the RC6/TX/CK and RC7/RX/DT I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTA<7>).

15.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 15-1. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer register TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and interrupt bit TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set, regardless of the state of enable bit TXIE, and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. TRMT is a read only bit, which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (Section 15.1).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.

TABLE 15-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

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
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	PSPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE[(1)]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000	0000 0000
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREG	USART Transmit Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

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

Note 1: The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18C2X2 devices. Always maintain these bits clear.

18.3 Power-down Mode (SLEEP)

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

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

For lowest current consumption in this mode, place all I/O pins at either V_{DD} or V_{SS}, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D and disable external clocks. Pull all I/O pins that are hi-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on-chip pull-ups on PORTB should be considered.

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

18.3.1 WAKE-UP FROM SLEEP

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

1. External RESET input on $\overline{\text{MCLR}}$ pin.
2. Watchdog Timer Wake-up (if WDT was enabled).
3. Interrupt from INT pin, RB port change, or a Peripheral Interrupt.

The following peripheral interrupts can wake the device from SLEEP:

1. PSP read or write.
2. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
3. TMR3 interrupt. Timer3 must be operating as an asynchronous counter.
4. CCP capture mode interrupt.
5. Special event trigger (Timer1 in Asynchronous mode using an external clock).
6. MSSP (START/STOP) bit detect interrupt.
7. MSSP transmit or receive in Slave mode (SPI/I^{2}C).
8. USART RX or TX (Synchronous Slave mode).
9. A/D conversion (when A/D clock source is RC).

Other peripherals cannot generate interrupts, since during SLEEP, no on-chip clocks are present.

External $\overline{\text{MCLR}}$ Reset will cause a device RESET. All other events are considered a continuation of program execution and will cause a "wake-up". The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the RCON register can be used to determine the cause of the device RESET. The $\overline{\text{PD}}$ bit, which is set on power-up, is cleared when `SLEEP` is invoked. The $\overline{\text{TO}}$ bit is cleared, if a WDT time-out occurred (and caused wake-up).

When the `SLEEP` instruction is being executed, the next instruction (PC + 2) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is set (enabled), the device executes the instruction after the `SLEEP` instruction and then branches to the interrupt address. In cases where the execution of the instruction following SLEEP is not desirable, the user should have a `NOP` after the `SLEEP` instruction.

18.3.2 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If an interrupt condition (interrupt flag bit and interrupt enable bits are set) occurs **before** the execution of a `SLEEP` instruction, the `SLEEP` instruction will complete as a `NOP`. Therefore, the WDT and WDT postscaler will not be cleared, the $\overline{\text{TO}}$ bit will not be set and $\overline{\text{PD}}$ bits will not be cleared.
- If the interrupt condition occurs **during or after** the execution of a `SLEEP` instruction, the device will immediately wake up from SLEEP. The `SLEEP` instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the $\overline{\text{TO}}$ bit will be set and the $\overline{\text{PD}}$ bit will be cleared.

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test the $\overline{\text{PD}}$ bit. If the $\overline{\text{PD}}$ bit is set, the `SLEEP` instruction was executed as a `NOP`.

To ensure that the WDT is cleared, a `CLRWDT` instruction should be executed before a `SLEEP` instruction.

SLEEP — Enter SLEEP mode

Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT postscaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description:	The power-down status bit ($\overline{PD}$) is cleared. The time-out status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared. The processor is put into SLEEP mode with the oscillator stopped.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to sleep

Example: `SLEEP`

Before Instruction
```
TO  =  ?
PD  =  ?
```
After Instruction
```
TO  =  1 †
PD  =  0
```

† If WDT causes wake-up, this bit is cleared.

SUBFWB — Subtract f from WREG with borrow

Syntax:	[*label*] SUBFWB f [,d [,a]
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	(WREG) – (f) – ($\overline{C}$) → dest
Status Affected:	N,OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description:	Subtract register 'f' and carry flag (borrow) from WREG (2's complement method). If 'd' is 0, the result is stored in WREG. If 'd' is 1, the result is stored in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBFWB   REG, 1, 0`

Before Instruction
```
REG    =   3
WREG   =   2
C      =   1
```
After Instruction
```
REG    =   FF
WREG   =   2
C      =   0
Z      =   0
N      =   1     ; result is negative
```

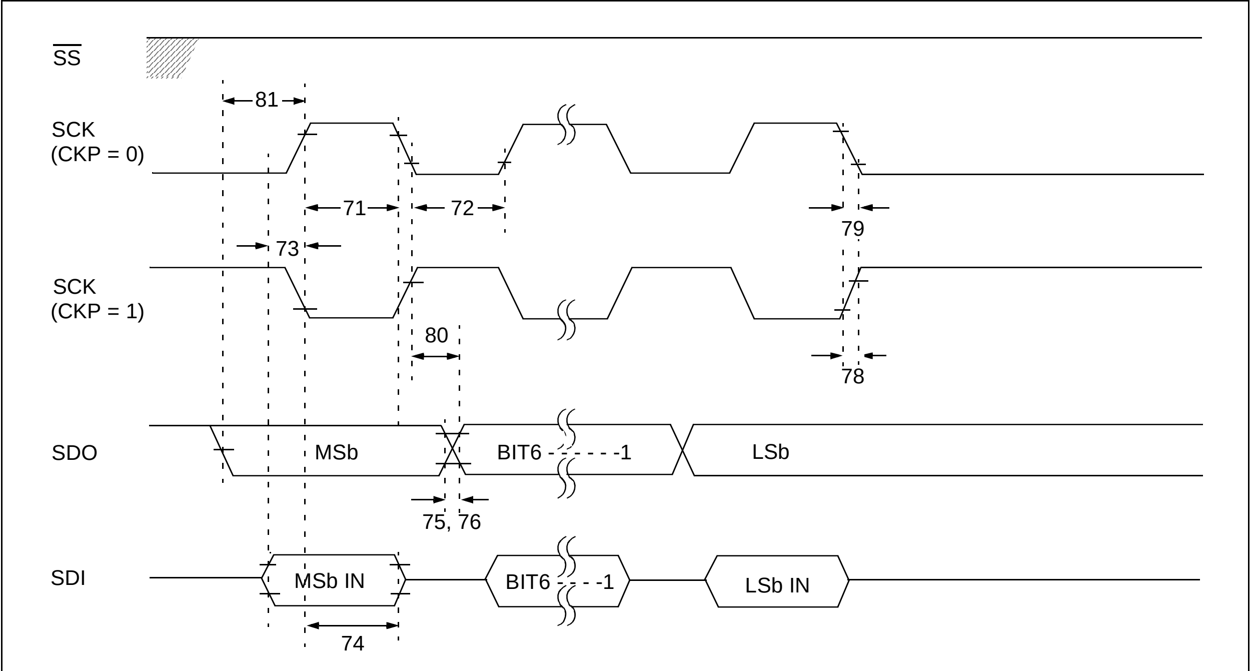
Example 2: `SUBFWB   REG, 0, 0`

Before Instruction
```
REG    =   2
WREG   =   5
C      =   1
```
After Instruction
```
REG    =   2
WREG   =   3
C      =   1
Z      =   0
N      =   0     ; result is positive
```

Example 3: `SUBFWB   REG, 1, 0`

Before Instruction
```
REG    =   1
WREG   =   2
C      =   0
```
After Instruction
```
REG    =   0
WREG   =   2
C      =   1
Z      =   1     ; result is zero
N      =   0
```

FIGURE 21-13: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)



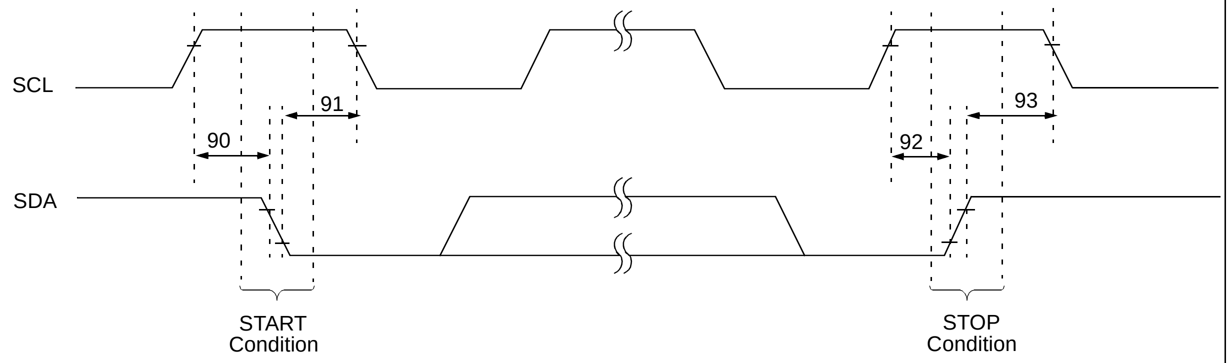
Note: Refer to Figure 21-4 for load conditions.

TABLE 21-12: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
71	TscH	SCK input high time (Slave mode)	Continuous	1.25TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		100	—	ns	
73A	TB2B	Last clock edge of Byte1 to the 1st clock edge of Byte2		1.5TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18**C**XXX	—	25	ns	
			PIC18**LC**XXX		45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
78	TscR	SCK output rise time (Master mode)	PIC18**C**XXX	—	25	ns	
			PIC18**LC**XXX		45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18**C**XXX	—	50	ns	
			PIC18**LC**XXX		100	ns	
81	TdoV2scH, TdoV2scL	SDO data output setup to SCK edge		TCY	—	ns	

Note 1: Requires the use of Parameter # 73A.
2: Only if Parameter # 71A and # 72A are used.

FIGURE 21-18: MASTER SSP I^2C BUS START/STOP BITS TIMING WAVEFORMS



Note: Refer to Figure 21-4 for load conditions.

TABLE 21-17: MASTER SSP I^2C BUS START/STOP BITS REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	START condition Setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	Only relevant for Repeated START condition
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	START condition Hold time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	After this period the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	STOP condition Setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	STOP condition Hold time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		

Note 1: Maximum pin capacitance = 10 pF for all I^2C pins.

FIGURE 22-25: MINIMUM AND MAXIMUM VIN vs. VDD, (TTL INPUT, -40°C TO +125°C)

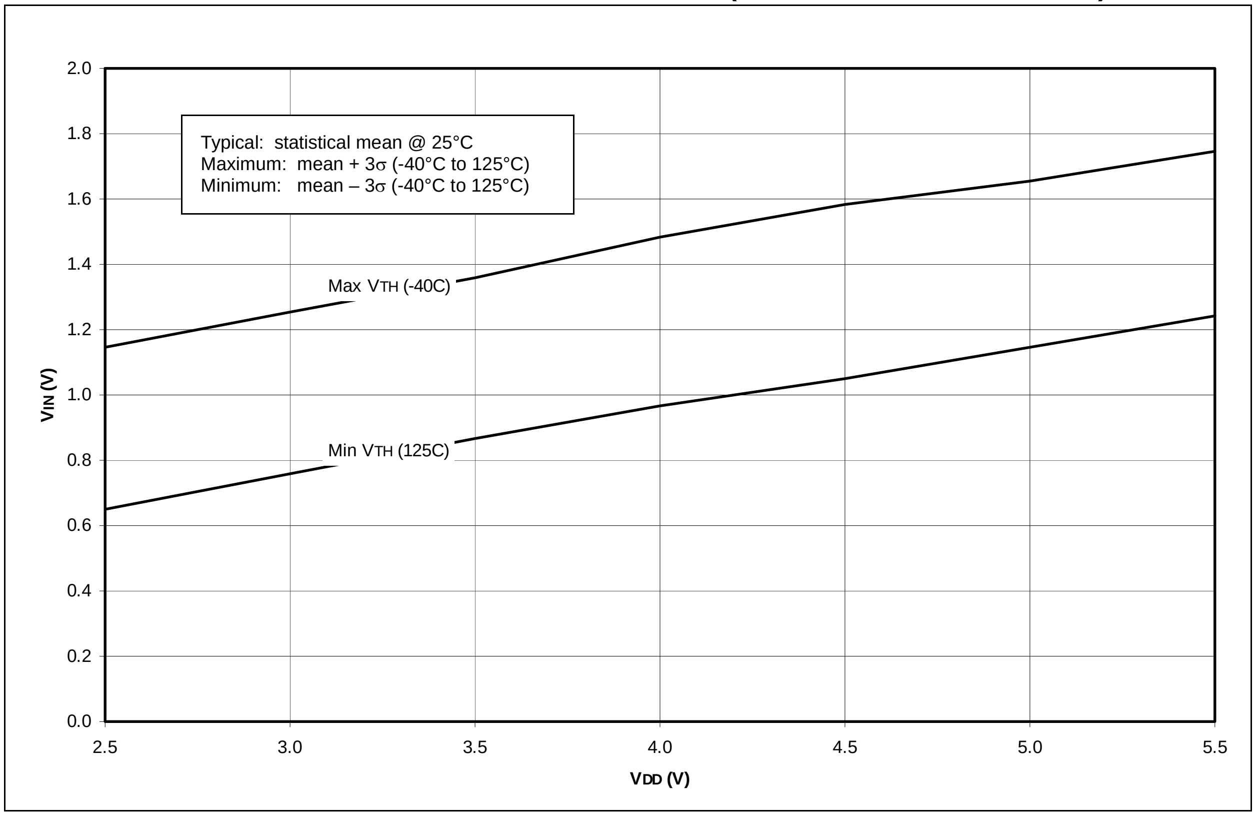
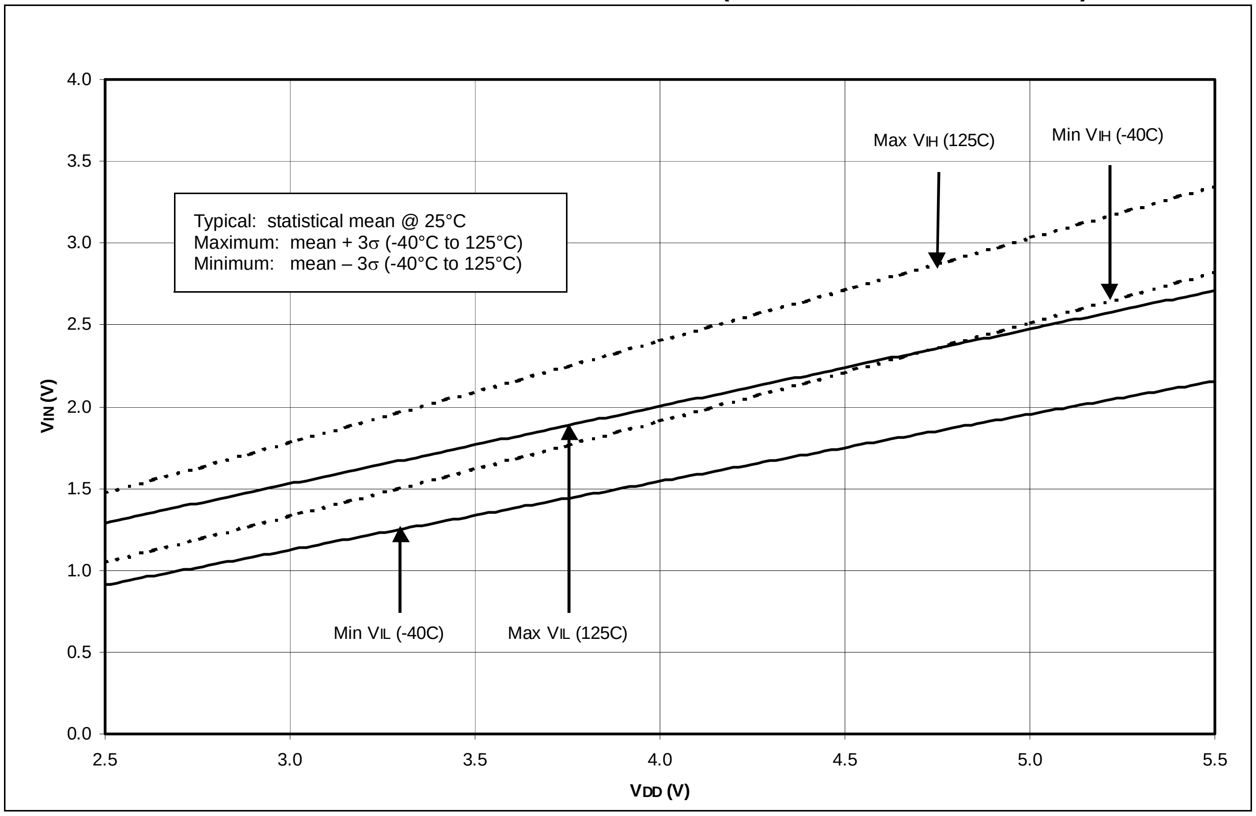


FIGURE 22-26: MINIMUM AND MAXIMUM VIN vs. VDD (I^2C INPUT, -40°C TO +125°C)



STOP Condition Receive or Transmit 143
Time-out Sequence on Power-up 32, 33
USART Asynchronous Master Transmission 156
USART Asynchronous Reception 158
USART Synchronous Reception 161
USART Synchronous Transmission 160
Wake-up from SLEEP via Interrupt 186
Timing Diagrams and Specifications 246
A/D Conversion 262
Brown-out Reset (BOR) 248
Capture/Compare/PWM (CCP) 250
CLKOUT and I/O 247
External Clock 246
I^{2}C Bus Data 257
I^{2}C Bus START/STOP Bits 256
Oscillator Start-up Timer (OST) 248
Parallel Slave Port (PSP) 251
Power-up Timer (PWRT) 248
RESET 248
Timer0 and Timer1 249
USART Synchronous Receive
(Master/Slave) 260
USART SynchronousTransmission
(Master/Slave) 260
Watchdog Timer (WDT) 248
TRISE Register 87
PSPMODE Bit 85, 90
TSTFSZ 227
Two-Word Instructions
Example Cases 41
TXSTA Register
BRGH Bit 151

U

Universal Synchronous Asynchronous Receiver
Transmitter. *See* USART.
USART 149
Asynchronous Mode 155
Associated Registers, Receive 158
Associated Registers, Transmit 156
Master Transmission 156
Receive Block Diagram 157
Receiver 157
Reception 158
Transmit Block Diagram 155
Transmitter 155
Baud Rate Generator (BRG) 151
Associated Registers 151
Baud Rate Error, Calculating 151
Baud Rate Formula 151
Baud Rates, Asynchronous Mode
(BRGH = 0) 153
Baud Rates, Asynchronous Mode
(BRGH = 1) 154
Baud Rates, Synchronous Mode 152
High Baud Rate Select (BRGH Bit) 151
Sampling 151
RCSTA Register 150
Serial Port Enable (SPEN Bit) 149
Synchronous Master Mode 159
Associated Registers, Reception 161
Associated Registers, Transmit 159
Reception 161
Timing Diagram, Synchronous Receive 260
Timing Diagram, Synchronous
Transmission 260
Transmission 160
Associated Registers 159
Synchronous Slave Mode 162
Associated Registers, Receive 163
Associated Registers, Transmit 162
Reception 163
Transmission 162
TXSTA Register 149

W

Wake-up from SLEEP 179, 185
Timing Diagram 186
Using Interrupts 185
Watchdog Timer (WDT) 179, 183
Associated Registers 184
Block Diagram 184
Postscaler 184
Programming Considerations 183
RC Oscillator 183
Time-out Period 183
Timing Diagram 248
Waveform for General Call Address Sequence 133
WCOL 137, 139, 142
WCOL Status Flag 137
WDT 183
WWW, On-Line Support 5

X

XORLW 227
XORWF 228