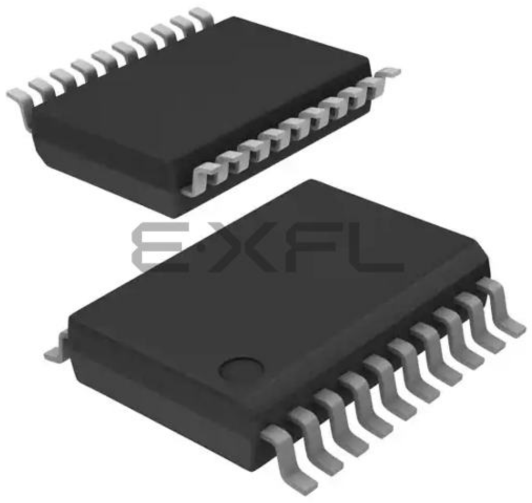
Microchip Technology - PIC16F18345T-I/SS Datasheet



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	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	18
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 17x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f18345t-i-ss

PIC16(L)F183XX Family Types

Device	Data Sheet Index	Program Flash Memory (Words)	Program Flash Memory (Kbytes)	Data Memory (bytes)	Data SRAM (bytes)	I/Os[2]	10-bit ADC (ch)	5-bit DAC	High-Speed/ Comparators	CWG	Clock Ref	Timers (8/16-bit)	CCP	10-bit PWM	NCO	EUSART	I^2C/SPI	CLC	DSM	PPS	XLP	PMD	Idle and Doze	Debug[1]
PIC16(L)F18313	**(1)**	2048	3.5	256	256	6	5	1	1	1	1	2/1	2	2	1	1	1/1	2	1	Y	Y	Y	Y	I
PIC16(L)F18323	**(1)**	2048	3.5	256	256	12	11	1	2	1	1	2/1	2	2	1	1	1/1	2	1	Y	Y	Y	Y	I
PIC16(L)F18324	**(2)**	4096	7	256	512	12	11	1	2	2	1	4/3	4	2	1	1	1/1	4	1	Y	Y	Y	Y	I
PIC16(L)F18325	**(3)**	8192	14	256	1024	12	11	1	2	2	1	4/3	4	2	1	1	2/2	4	1	Y	Y	Y	Y	I
PIC16(L)F18326	**(4)**	16384	28	256	2048	12	15	1	2	2	1	4/3	4	2	1	1	2/2	4	1	Y	Y	Y	Y	I
PIC16(L)F18344	**(2)**	4096	7	256	512	18	17	1	2	2	1	4/3	4	2	1	1	1/1	4	1	Y	Y	Y	Y	I
PIC16(L)F18345	**(3)**	8192	14	256	1024	18	17	1	2	2	1	4/3	4	2	1	1	2/2	4	1	Y	Y	Y	Y	I
PIC16(L)F18346	**(4)**	16384	28	256	2048	18	21	1	2	2	1	4/3	4	2	1	1	2/2	4	1	Y	Y	Y	Y	I

Note 1: Debugging Methods: (I) – Integrated on Chip;
2: One pin is input-only.

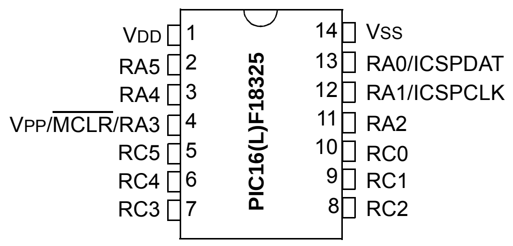
Data Sheet Index: (Unshaded devices are described in this document.)

1:	DS40001799	PIC16(L)F18313/18323 Data Sheet,	Full-Featured, Low Pin Count Microcontrollers with XLP
2:	DS40001800	PIC16(L)F18324/18344 Data Sheet,	Full Featured, Low Pin Count Microcontrollers with XLP
3:	DS40001795	PIC16(L)F18325/18345 Data Sheet,	Full Featured, Low Pin Count Microcontrollers with XLP
4:	DS40001839	PIC16(L)F18326/18346 Data Sheet,	Full Featured, Low Pin Count Microcontrollers with XLP

Note: For other small form-factor package availability and marking information, please visit **http://www.microchip.com/packaging** or contact your local sales office.

Pin Diagrams

FIGURE 1: 14-PIN PDIP, SOIC, TSSOP



Note: See Table 1 for location of all peripheral functions.

TABLE 1-3: PIC16(L)F18345 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RC3/ANC3/C1IN3-/C2IN3-/ MDMIN[(1)]/ CCP2[(1)]/CLCIN1[(1)]/	RC3	TTL/ST	CMOS	General purpose I/O.
	ANC3	AN	—	ADC Channel C3 input.
	C1IN3-	AN	—	Comparator C1 negative input.
	C2IN3-	AN	—	Comparator C2 negative input.
	MDMIN	TTL/ST	—	Modular Source input.
	CCP2	TTL/ST	CMOS	Capture/Compare/PWM 2 input.
	CLCIN1	TTL/ST	—	Configurable Logic Cell 1 input.
RC4/ANC4	RC4	TTL/ST	CMOS	General purpose I/O.
	ANC4	AN	—	ADC Channel C4 input.
RC5/ANC5/MDCIN2[(1)]/CCP1[(1)]	RC5	TTL/ST	CMOS	General purpose I/O.
	ANC5	AN	—	ADC Channel C5 input.
	MDCIN2	TTL/ST	—	Modular Carrier input 2.
	CCP1	TTL/ST	CMOS	Capture/Compare/PWM 1 input.
RC6/ANC6/$\overline{\text{SS1}}$[(1)]	RC6	TTL/ST	CMOS	General purpose I/O.
	ANC6	AN	—	ADC Channel C6 input.
	$\overline{\text{SS1}}$	TTL/ST	—	Slave Select 1 input.
RC7/ANC7	RC7	TTL/ST	CMOS	General purpose I/O.
	ANC7	AN	—	ADC Channel C7 input.
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

Legend: AN = Analog input or output CMOS= CMOS compatible input or output OD = Open-Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I^{2}C = Schmitt Trigger input with I^{2}C
HV = High Voltage XTAL = Crystal levels

Note 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers. See Register 13-2.

2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See Register 13-2.

3: These I^{2}C functions are bidirectional. The output pin selections must be the same as the input pin selections.

5.0 DEVICE CONFIGURATION

Device configuration consists of Configuration Words, Code Protection and Device ID.

5.1 Configuration Words

There are several Configuration Word bits that allow different oscillator and memory protection options. These are implemented as Configuration Word 1 at 8007h, Configuration Word 2 at 8008h, Configuration Word 3 at 8009h, and Configuration Word 4 at 800Ah.

Note: The $\overline{\text{DEBUG}}$ bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a ‘1’.

6.3 Low-Power Brown-out Reset (LPBOR)

The Low-Power Brown-Out Reset (LPBOR) circuit provides alternative protection against Brown-out conditions. When V_{DD} falls below the LPBOR threshold, the device is held in Reset. When this occurs, the BOR bit of the PCON0 register is cleared to indicate that a Brown-out Reset occurred. The BOR bit will be cleared when either the BOR or the LPBOR circuitry detects a BOR condition. The LPBOR feature can be used with or without BOR enabled.

When used while BOR is enabled, the LPBOR can be used as a secondary protection circuit in case the BOR circuit fails to detect the BOR condition. Additionally, when BOR is enabled except while in Sleep (BOREN<1:0> = `10`), the LPBOR circuit will hold the device in Reset while VDD is lower than the LPBOR threshold, and will also re-arm the POR. (See Table 35-11 for LPBOR Reset voltage levels).

When used without BOR enabled, the LPBOR circuit provides a single Reset trip point with the benefit of reduced current consumption.

6.3.1 ENABLING LPBOR

The LPBOR is controlled by the $\overline{\text{LPBOR}}$ bit of Configuration Words. When the device is erased, the LPBOR module defaults to disabled.

6.3.1.1 LPBOR Module Output

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. This signal is OR'd together with the Reset signal of the BOR module to provide the generic $\overline{\text{BOR}}$ signal, which goes to the PCON0 register and to the power control block.

6.4 $\overline{\text{MCLR}}$

The $\overline{\text{MCLR}}$ is an optional external input that can reset the device. The $\overline{\text{MCLR}}$ function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words (Table 6-2).

TABLE 6-2: $\overline{\text{MCLR}}$ CONFIGURATION

MCLRE	LVP	$\overline{\text{MCLR}}$
0	0	Disabled
1	0	Enabled
x	1	Enabled

6.4.1 $\overline{\text{MCLR}}$ ENABLED

When $\overline{\text{MCLR}}$ is enabled and the pin is held low, the device is held in Reset. The $\overline{\text{MCLR}}$ pin is connected to V_{DD} through an internal weak pull-up.

The device has a noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

Note:	A Reset does not drive the $\overline{\text{MCLR}}$ pin low.

6.4.2 $\overline{\text{MCLR}}$ DISABLED

When $\overline{\text{MCLR}}$ is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See **Section 12.2 "PORTA Registers"** for more information.

6.5 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a `CLRWDT` instruction within the time-out period. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register as well as the $\overline{\text{RWDT}}$ bit in the PCON0 register, are changed to indicate the WDT Reset. See **Section 10.0 "Watchdog Timer (WDT)"** for more information.

6.6 RESET Instruction

A `RESET` instruction will cause a device Reset. The $\overline{\text{RI}}$ bit in the PCON0 register will be set to '`0`'. See Table 6-4 for default conditions after a `RESET` instruction has occurred.

6.7 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON0 register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See **Section 4.4 "Stack"** for more information.

6.8 Programming Mode Exit

Upon exit of Programming mode, the device will behave as if a device Reset had just occurred.

6.9 Power-up Timer

The Power-up Timer provides a nominal 64 ms time-out on POR or Brown-out Reset.

The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the V_{DD} to rise to an acceptable level. The Power-up Timer is enabled by clearing the $\overline{\text{PWRTE}}$ bit in Configuration Words.

The Power-up Timer starts after the release of the POR and BOR.

For additional information, refer to Application Note AN607, *"Power-up Trouble Shooting"* (DS00607).

8.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt Enable bit(s) (PIEx bits) for the specific interrupt event(s)
- PEIE bit of the INTCON register

The PIR1, PIR2, PIR3 and PIR4 registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual interrupt enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- Critical registers are automatically saved to the shadow registers (See **"Section 8.5 "Automatic Context Saving"**)
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) should determine the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack, restoring the saved context from the shadow registers and setting the GIE bit.

For additional information on a specific interrupt's operation, refer to its peripheral chapter.

Note 1: Individual interrupt flag bits are set, regardless of the state of any other enable bits.

2: All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

8.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The interrupt is sampled during Q1 of the instruction cycle. The actual interrupt latency then depends on the instruction that is executing at the time the interrupt is detected. See Figure 8-2 and Figure 8-3 for more details.

REGISTER 8-9: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
TMR6IF	C2IF	C1IF	NVMIF	SSP2IF	BCL2IF	TMR4IF	NCO1IF
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	HS = Hardware set

bit 7 **TMR6IF:** TMR6 to PR6 Match Interrupt Flag bit
1 = TMR6 to PR6 match occurred (must be cleared in software)
0 = No TMR6 to PR6 match occurred

bit 6 **C2IF:** Comparator C2 Interrupt Flag bit
1 = Comparator 2 interrupt asserted
0 = Comparator 2 interrupt not asserted

bit 5 **C1IF:** Comparator C1 Interrupt Flag bit
1 = Comparator 1 interrupt asserted
0 = Comparator 1 interrupt not asserted

bit 4 **NVMIF**: NVM Interrupt Flag bit
1 = The NVM has completed a programming task
0 = NVM interrupt not asserted

bit 3 **SSP2IF:** Master Synchronous Serial Port (MSSP2) Interrupt Flag bit
1 = The Transmission/Reception/Bus Condition is complete (must be cleared in software)
0 = Waiting for the Transmission/Reception/Bus Condition in progress

bit 2 **BCL2IF:** MSSP2 Bus Collision Interrupt Flag bit
1 = A bus collision was detected (must be cleared in software)
0 = No bus collision was detected

bit 1 **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = TMR4 to PR4 match occurred (must be cleared in software)
0 = No TMR4 to PR4 match occurred

bit 0 **NCO1IF:** NCO Interrupt Flag bit
1 = The NCO has rolled over.
0 = No NCO interrupt is asserted.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

11.4.3 NVMREG WRITE TO EEPROM

Writing to the EEPROM is accomplished by the following steps:

1. Set the NVMREGS and WREN bits of the NVMCON1 register.
2. Write the desired address (address +7000h) into the NVMADRH:NVMADRL register pair (Table 11-2).
3. Perform the unlock sequence as described in **Section 11.4.2 "NVM Unlock Sequence"**.

A single EEPROM byte is written with NVMDATA. The operation includes an implicit erase cycle for that byte (it is not necessary to set the FREE bit), and requires many instruction cycles to finish. CPU execution continues in parallel and, when complete, WR is cleared by hardware, NVMIF is set, and an interrupt will occur if NVMIE is also set. Software must poll the WR bit to determine when writing is complete, or wait for the interrupt to occur. WREN will remain unchanged.

Once the EEPROM write operation begins, clearing the WR bit will have no effect; the operation will run to completion.

11.4.4 NVMREG ERASE OF PROGRAM FLASH MEMORY

Program Flash memory can only be erased one row at a time. No automatic erase occurs upon the initiation of the write to program Flash memory.

To erase a program Flash memory row:

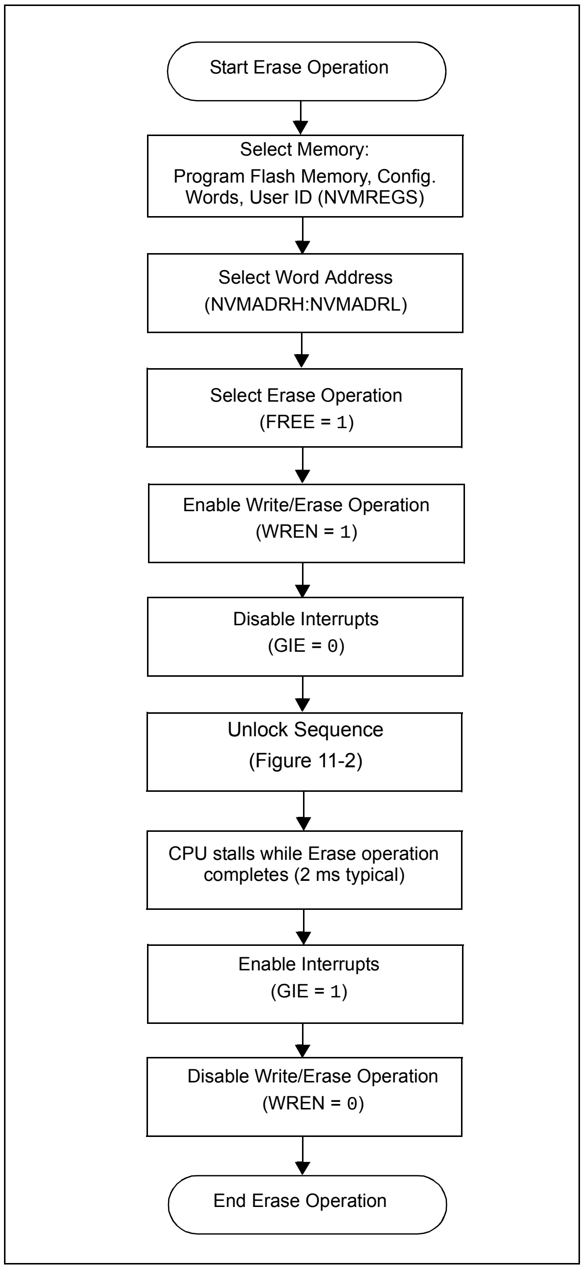
1. Clear the NVMREGS bit of the NVMCON1 register to erase program Flash memory locations, or set the NVMREGS bit to erase user ID locations.
2. Write the desired address into the NVMADRH:NVMADRL register pair (Table 11-2).
3. Set the FREE and WREN bits of the NVMCON1 register.
4. Perform the unlock sequence as described in **Section 11.4.2 "NVM Unlock Sequence"**.

If the program Flash memory address is write-protected, the WR bit will be cleared and the erase operation will not take place.

While erasing program Flash memory, CPU operation is suspended, and resumes when the operation is complete. Upon completion, the NVMIF is set, and an interrupt will occur if the NVMIE bit is also set.

Write latch data is not affected by erase operations, and WREN will remain unchanged.

FIGURE 11-3: NVM ERASE FLOWCHART



11.4.5 NVMREG WRITE TO PROGRAM FLASH MEMORY

Program memory is programmed using the following steps:

1. Load the address of the row to be programmed into NVMADRH:NVMADRL.
2. Load each write latch with data.
3. Initiate a programming operation.
4. Repeat steps 1 through 3 until all data is written.

Before writing to program memory, the word(s) to be written must be erased or previously unwritten. Program memory can only be erased one row at a time. No automatic erase occurs upon the initiation of the write.

Program memory can be written one or more words at a time. The maximum number of words written at one time is equal to the number of write latches. See Figure 11-4 (row writes to program memory with 32 write latches) for more details.

The write latches are aligned to the Flash row address boundary defined by the upper ten bits of NVMADRH:NVMADRL, (NVMADRH<6:0>:NVMADRL<7:5>) with the lower five bits of NVMADRL, (NVMADRL<4:0>) determining the write latch being loaded. Write operations do not cross these boundaries. At the completion of a program memory write operation, the data in the write latches is reset to contain 0x3FFF.

The following steps should be completed to load the write latches and program a row of program memory. These steps are divided into two parts. First, each write latch is loaded with data from the NVMDATH:NVMDATL using the unlock sequence with LWLO = `1`. When the last word to be loaded into the write latch is ready, the LWLO bit is cleared and the unlock sequence executed. This initiates the programming operation, writing all the latches into Flash program memory.

Note:	The special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. If the unlock sequence is interrupted, writing to the latches or program memory will not be initiated.

1. Set the WREN bit of the NVMCON1 register.
2. Clear the NVMREGS bit of the NVMCON1 register.
3. Set the LWLO bit of the NVMCON1 register. When the LWLO bit of the NVMCON1 register is ‘`1`’, the write sequence will only load the write latches and will not initiate the write to Flash program memory.
4. Load the NVMADRH:NVMADRL register pair with the address of the location to be written.
5. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.
6. Execute the unlock sequence (**Section 11.4.2 “NVM Unlock Sequence”**). The write latch is now loaded.
7. Increment the NVMADRH:NVMADRL register pair to point to the next location.
8. Repeat steps 5 through 7 until all but the last write latch has been loaded.
9. Clear the LWLO bit of the NVMCON1 register. When the LWLO bit of the NVMCON1 register is ‘`0`’, the write sequence will initiate the write to Flash program memory.
10. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.
11. Execute the unlock sequence (**Section 11.4.2 “NVM Unlock Sequence”**). The entire program memory latch content is now written to Flash program memory.

Note:	The program memory write latches are reset to the blank state (0x3FFF) at the completion of every write or erase operation. As a result, it is not necessary to load all the program memory write latches. Unloaded latches will remain in the blank state.

An example of the complete write sequence is shown in Example 11-4. The initial address is loaded into the NVMADRH:NVMADRL register pair; the data is loaded using indirect addressing.

11.4.9 WRERR BIT

The WRERR bit can be used to determine if a write error occurred.

WRERR will be set if one of the following conditions occurs:

- If WR is set while the NVMADRH:NMVADRL points to a write-protected address
- A Reset occurs while a self-write operation was in progress
- An unlock sequence was interrupted

The WRERR bit is normally set by hardware, but can be set by the user for test purposes. Once set, WRERR must be cleared in software.

TABLE 11-4: ACTIONS FOR PROGRAM FLASH MEMORY WHEN WR = 1

Free	LWLO	Actions for Program Flash Memory when WR = 1	Comments
0	0	Write the write latch data to program Flash memory row. See **Section 11.4.4 “NVMREG Erase of Program Flash Memory”**	• If WP is enabled, WR is cleared and WRERR is set • Write latches are reset to 3FFh • NVMDATH:NVMDATL is ignored
0	1	Copy NVMDATH:NVMDATL to the write latch corresponding to NVMADR LSBs. See **Section 11.4.4 “NVMREG Erase of Program Flash Memory”**	• Write protection is ignored • No memory access occurs
1	x	Erase the 32-word row of NVMADRH:NVMADRL location. See **Section 11.4.3 “NVMREG Write to EEPROM”**	• If WP is enabled, WR is cleared and WRERR is set • All 32 words are erased • NVMDATH:NVMDATL is ignored

REGISTER 14-2: PMD1: PMD CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
NCOMD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD
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	q = Value depends on condition

bit 7	**NCOMD:** Disable Numerically Control Oscillator bit `1` = NCO1 module disabled `0` = NCO1 module enabled
bit 6	**TMR6MD:** Disable Timer TMR6 bit `1` = TMR6 module disabled `0` = TMR6 module enabled
bit 5	**TMR5MD:** Disable Timer TMR5 bit `1` = TMR5 module disabled `0` = TMR5 module enabled
bit 4	**TMR4MD:** Disable Timer TMR4 bit `1` = TMR4 module disabled `0` = TMR4 module enabled
bit 3	**TMR3MD:** Disable Timer TMR3 bit `1` = TMR3 module disabled `0` = TMR3 module enabled
bit 2	**TMR2MD:** Disable Timer TMR2 bit `1` = TMR2 module disabled `0` = TMR2 module enabled
bit 1	**TMR1MD:** Disable Timer TMR1 bit `1` = TMR1 module disabled `0` = TMR1 module enabled
bit 0	**TMR0MD:** Disable Timer TMR0 bit `1` = TMR0 module disabled `0` = TMR0 module enabled

REGISTER 22-3: ADACT: A/D AUTO-CONVERSION TRIGGER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ADACT<4:0>				
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 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **ADACT<4:0>:** Auto-Conversion Trigger Selection bits(1)

10001	=	Timer5 overflow(2)
10000	=	Timer3 overflow(2)
1111	=	CCP4
1110	=	CCP3
1101	=	CCP2
1100	=	CCP1
1011	=	CLC4
1010	=	CLC3
1001	=	CLC2
1000	=	CLC1
0111	=	Comparator C2
0110	=	Comparator C1
0101	=	Timer2-PR2 match
0100	=	Timer1 overflow(2)
0011	=	Timer0 overflow(2)
0010	=	Timer6-PR6 match
0001	=	Timer4-PR4 match
0000	=	No auto-conversion trigger selected

Note 1: This is a rising edge sensitive input for all sources.

2: Trigger corresponds to when the peripheral’s interrupt flag is set.

23.2 Fixed Duty Cycle (FDC) Mode

In Fixed Duty Cycle (FDC) mode, every time the accumulator overflows (NCO_overflow), the output is toggled. This provides a 50% duty cycle with a constant frequency, provided that the increment value remains constant. The FDC frequency can be calculated using Equation 23-2. The FDC frequency is half of the overflow frequency since it takes two overflow events to generate one FDC clock period. For more information, see Figure 23-2.

EQUATION 23-2: FDC FREQUENCY

$$F_{fdc} = F_{overflow} / 2$$

The FDC mode is selected by clearing the N1PFM bit in the NCO1CON register.

23.3 Pulse Frequency (PF) Mode

In Pulse Frequency (PF) mode, every time the accumulator overflows (NCO_overflow), the output becomes active for one or more clock periods. Once the clock period expires, the output returns to an inactive state. This provides a pulsed output. The output becomes active on the rising clock edge immediately following the overflow event. For more information, see Figure 23-2.

The value of the active and inactive states depends on the polarity bit, N1POL, in the NCO1CON register.

The PF mode is selected by setting the N1PFM bit in the NCO1CON register.

23.3.1 OUTPUT PULSE WIDTH CONTROL

When operating in PF mode, the active state of the output can vary in width by multiple clock periods. Various pulse widths are selected with the N1PWS<2:0> bits in the NCO1CLK register.

When the selected pulse width is greater than the accumulator overflow time frame, the output of the NCO1 does not toggle.

23.4 Output Polarity Control

The last stage in the NCO1 module is the output polarity. The N1POL bit in the NCO1CON register selects the output polarity. Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

The NCO1 output can be used internally by source code or other peripherals. Accomplish this by reading the N1OUT (read-only) bit of the NCO1CON register.

The NCO1 output signal is available to the following peripherals:

- CWG

24.4 Operation During Sleep

The DAC continues to function during Sleep. When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DAC1CON0 register are not affected.

24.5 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DAC1OUT pin.
- The DAC1R<4:0> range select bits are cleared.

24.6 Register Definitions: DAC Control

REGISTER 24-1: DACCON0: VOLTAGE REFERENCE CONTROL REGISTER 0

R/W-0/0	U-0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0	R/W-0/0
DAC1EN	—	DAC1OE	—	DAC1PSS<1:0>		—	DAC1NSS
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	**DAC1EN:** DAC1 Enable bit 1 = DAC is enabled 0 = DAC is disabled
bit 6	**Unimplemented:** Read as ‘0’
bit 5	**DAC1OE:** DAC1 Voltage Output 1 Enable bit 1 = DAC voltage level is output on the DAC1OUT pin 0 = DAC voltage level is disconnected from the DAC1OUT pin
bit 4	**Unimplemented:** Read as ‘0’
bit 3-2	**DAC1PSS<1:0>:** DAC1 Positive Source Select bits 11 = Reserved, do not use 10 = FVR output 01 = V_{REF+} pin 00 = V_{DD}
bit 1	**Unimplemented:** Read as ‘0’
bit 0	**DAC1NSS:** DAC1 Negative Source Select bits 1 = V_{REF-} pin 0 = V_{SS}

30.5.2 SLAVE RECEPTION

When the R/$\overline{W}$ bit of a matching received address byte is clear, the R/$\overline{W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and acknowledged.

When the Overflow condition exists for a received address, then not Acknowledge is given. An Overflow condition is defined as either bit BF of the SSPxSTAT register is set, or bit SSPOV of the SSPxCON1 register is set. The BOEN bit of the SSPxCON3 register modifies this operation. For more information see Register 30-4.

An MSSP interrupt is generated for each transferred data byte. Flag bit, SSPxIF, must be cleared by software.

When the SEN bit of the SSPxCON2 register is set, SCL will be held low (clock stretch) following each received byte. The clock must be released by setting the CKP bit of the SSPxCON1 register.

30.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSPx module configured as an I^{2}C slave in 7-bit Addressing mode. Figure 30-14 and Figure 30-15 is used as a visual reference for this description.

This is a step-by-step process of what typically must be done to accomplish I^{2}C communication.

1. Start bit detected.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{W}$ bit clear is received.
4. The slave pulls SDA low sending an $\overline{ACK}$ to the master, and sets SSPxIF bit.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF clearing the BF flag.
7. If SEN = `1`; Slave software sets CKP bit to release the SCL line.
8. The master clocks out a data byte.
9. Slave drives SDA low sending an $\overline{ACK}$ to the master, and sets SSPxIF bit.
10. Software clears SSPxIF.
11. Software reads the received byte from SSPxBUF clearing BF.
12. Steps 8-12 are repeated for all received bytes from the master.
13. Master sends Stop condition, setting P bit of SSPxSTAT, and the bus goes idle.

30.5.2.2 7-bit Reception with AHEN and DHEN

Slave device reception with AHEN and DHEN set operate the same as without these options with extra interrupts and clock stretching added after the eighth falling edge of SCL. These additional interrupts allow time for the slave software to decide whether it wants to $\overline{ACK}$ the receive address or data byte.

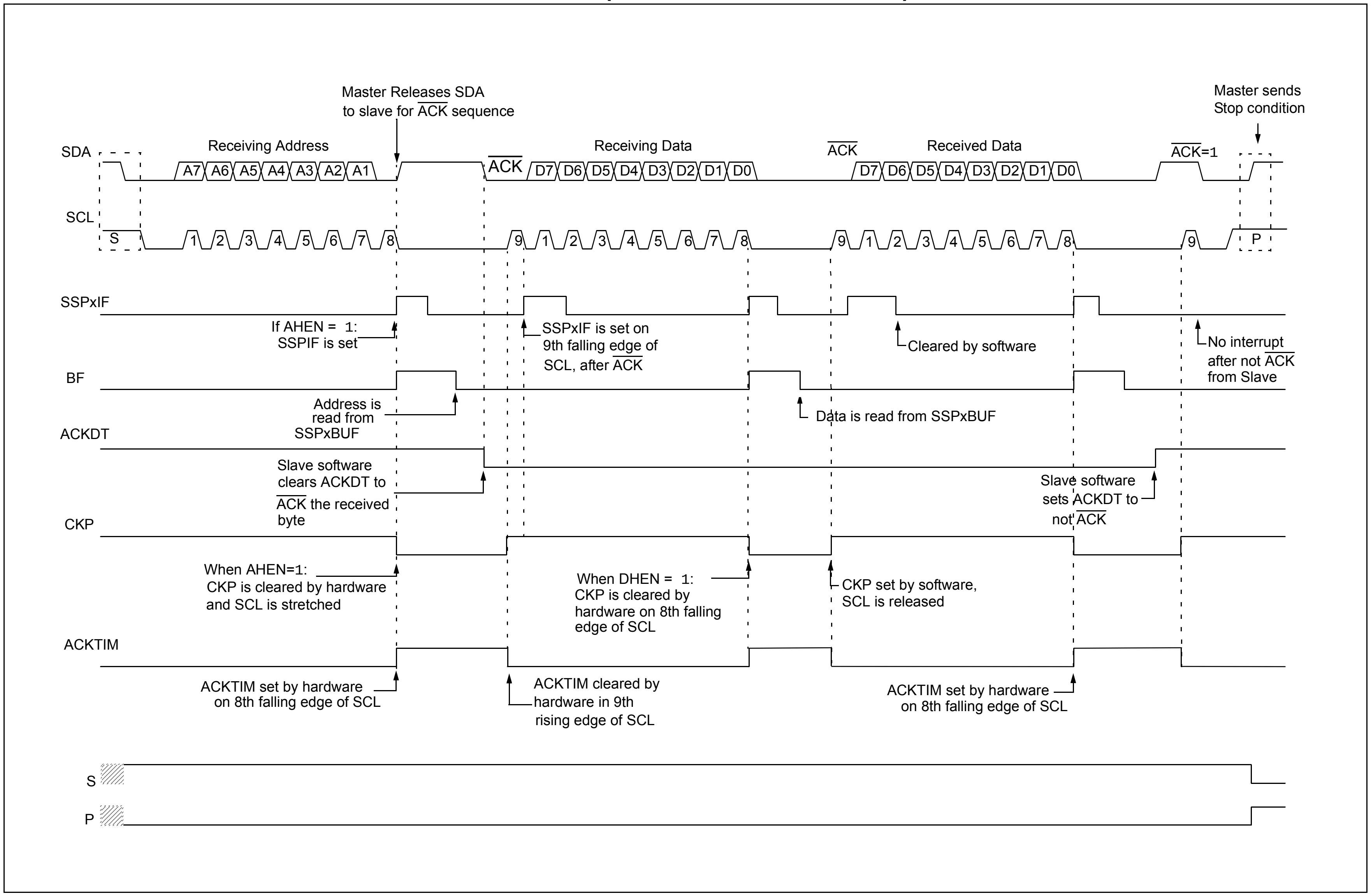
This list describes the steps that need to be taken by slave software to use these options for I^{2}C communication. Figure 30-16 displays a module using both address and data holding. Figure 30-17 includes the operation with the SEN bit of the SSPxCON2 register set.

1. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with R/$\overline{W}$ bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCL.
3. Slave clears the SSPxIF.
4. Slave can look at the ACKTIM bit of the SSPxCON3 register to determine if the SSPxIF was after or before the $\overline{ACK}$.
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets $\overline{ACK}$ value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP.
8. SSPxIF is set after an $\overline{ACK}$, not after a NACK.
9. If SEN = `1` the slave hardware will stretch the clock after the $\overline{ACK}$.
10. Slave clears SSPxIF.

Note: SSPxIF is still set after the ninth falling edge of SCL even if there is no clock stretching and BF has been cleared. Only if NACK is sent to master is SSPIF not set

11. SSPxIF set and CKP cleared after eighth falling edge of SCL for a received data byte.
12. Slave looks at ACKTIM bit of SSPxCON3 to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an $\overline{ACK}$ = `1`, or the master sending a Stop condition. If a Stop is sent and Interrupt on Stop Detect is disabled, the slave will only know by polling the P bit of the SSPxSTAT register.

FIGURE 30-16: I^{2}C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)


Master Releases SDA to slave for ACK sequence
Master sends Stop condition
SDA
Receiving Address
A7 A6 A5 A4 A3 A2 A1
ACK
Receiving Data
D7 D6 D5 D4 D3 D2 D1 D0
ACK
Received Data
D7 D6 D5 D4 D3 D2 D1 D0
ACK=1
SCL
S
1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9
P
SSPxIF
If AHEN = 1: SSPIF is set
SSPxIF is set on 9th falling edge of SCL, after ACK
Cleared by software
No interrupt after not ACK from Slave
BF
Address is read from SSPxBUF
Data is read from SSPxBUF
ACKDT
Slave software clears ACKDT to ACK the received byte
Slave software sets ACKDT to not ACK
CKP
When AHEN=1: CKP is cleared by hardware and SCL is stretched
When DHEN = 1: CKP is cleared by hardware on 8th falling edge of SCL
CKP set by software, SCL is released
ACKTIM
ACKTIM set by hardware on 8th falling edge of SCL
ACKTIM cleared by hardware in 9th rising edge of SCL
ACKTIM set by hardware on 8th falling edge of SCL
S
P

31.5 EUSART1 Operation During Sleep

The EUSART1 will remain active during Sleep only in the Synchronous Slave mode. All other modes require the system clock and therefore cannot generate the necessary signals to run the Transmit or Receive Shift registers during Sleep.

Synchronous Slave mode uses an externally generated clock to run the Transmit and Receive Shift registers.

31.5.1 SYNCHRONOUS RECEIVE DURING SLEEP

To receive during Sleep, all the following conditions must be met before entering Sleep mode:

- RC1STA and TX1STA Control registers must be configured for Synchronous Slave Reception (see **Section 31.4.2.4 “Synchronous Slave Reception Set-up”**).
- If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
- The RCIF interrupt flag must be cleared by reading RC1REG to unload any pending characters in the receive buffer.

Upon entering Sleep mode, the device will be ready to accept data and clocks on the RX/DT and TX/CK pins, respectively. When the data word has been completely clocked in by the external device, the RCIF interrupt flag bit of the PIR1 register will be set. Thereby, waking the processor from Sleep.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit of the INTCON register is also set, then the Interrupt Service Routine at address 004h will be called.

31.5.2 SYNCHRONOUS TRANSMIT DURING SLEEP

To transmit during Sleep, all the following conditions must be met before entering Sleep mode:

- The RC1STA and TX1STA Control registers must be configured for synchronous slave transmission (see **Section 31.4.2.2 “Synchronous Slave Transmission Set-up”**).
- The TXIF interrupt flag must be cleared by writing the output data to the TX1REG, thereby filling the TSR and transmit buffer.
- If interrupts are desired, set the TXIE bit of the PIE1 register and the PEIE bit of the INTCON register.
- Interrupt enable bits TXIE of the PIE1 register and PEIE of the INTCON register must set.

Upon entering Sleep mode, the device will be ready to accept clocks on TX/CK pin and transmit data on the RX/DT pin. When the data word in the TSR has been completely clocked out by the external device, the pending byte in the TX1REG will transfer to the TSR and the TXIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TX1REG is available to accept another character for transmission, which will clear the TXIF flag.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit is also set then the Interrupt Service Routine at address 0004h will be called.

TABLE 31-3: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART1 Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	FOSC/[64 (n+1)]
0	0	1	8-bit/Asynchronous	FOSC/[16 (n+1)]
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	FOSC/[4 (n+1)]
1	0	x	8-bit/Synchronous	
1	1	x	16-bit/Synchronous	

Legend: x = Don't care, n = value of SP1BRGH, SP1BRGL register pair.

TABLE 31-4: BAUD RATE FOR ASYNCHRONOUS MODES

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 32.000 MHz			FOSC = 20.000 MHz			FOSC = 18.432 MHz			FOSC = 11.0592 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	—	—	—	—	—	—	—	—	—
1200	—	—	—	1221	1.73	255	1200	0.00	239	1200	0.00	143
2400	2404	0.16	207	2404	0.16	129	2400	0.00	119	2400	0.00	71
9600	9615	0.16	51	9470	-1.36	32	9600	0.00	29	9600	0.00	17
10417	10417	0.00	47	10417	0.00	29	10286	-1.26	27	10165	-2.42	16
19.2k	19.23k	0.16	25	19.53k	1.73	15	19.20k	0.00	14	19.20k	0.00	8
57.6k	55.55k	-3.55	3	—	—	—	57.60k	0.00	7	57.60k	0.00	2
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 8.000 MHz			FOSC = 4.000 MHz			FOSC = 3.6864 MHz			FOSC = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	300	0.16	207	300	0.00	191	300	0.16	51
1200	1202	0.16	103	1202	0.16	51	1200	0.00	47	1202	0.16	12
2400	2404	0.16	51	2404	0.16	25	2400	0.00	23	—	—	—
9600	9615	0.16	12	—	—	—	9600	0.00	5	—	—	—
10417	10417	0.00	11	10417	0.00	5	—	—	—	—	—	—
19.2k	—	—	—	—	—	—	19.20k	0.00	2	—	—	—
57.6k	—	—	—	—	—	—	57.60k	0.00	0	—	—	—
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

34.2 Instruction Descriptions

ADDFSR — Add Literal to FSRn

Syntax:	[*label*] ADDFSR FSRn, k
Operands:	-32 ≤ k ≤ 31 n ∈ [0, 1]
Operation:	FSR(n) + k → FSR(n)
Status Affected:	None
Description:	The signed 6-bit literal 'k' is added to the contents of the FSRnH:FSRnL register pair. FSRn is limited to the range 0000h-FFFFh. Moving beyond these bounds will cause the FSR to wrap-around.

ADDLW — Add literal and W

Syntax:	[*label*] ADDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) + k → (W)
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register.

ADDWF — Add W and f

Syntax:	[*label*] ADDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) + (f) → (destination)
Status Affected:	C, DC, Z
Description:	Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ADDWFC — ADD W and CARRY bit to f

Syntax:	[*label*] ADDWFC f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) + (f) + (C) → dest
Status Affected:	C, DC, Z
Description:	Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.

ANDLW — AND literal with W

Syntax:	[*label*] ANDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .AND. (k) → (W)
Status Affected:	Z
Description:	The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register.

ANDWF — AND W with f

Syntax:	[*label*] ANDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
Description:	AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ASRF — Arithmetic Right Shift

Syntax:	[*label*] ASRF f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f<7>)→ dest<7> (f<7:1>) → dest<6:0>, (f<0>) → C,
Status Affected:	C, Z
Description:	The contents of register 'f' are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.

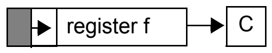


TABLE 35-2: SUPPLY CURRENT (I_{DD})(1,2)

PIC16LF18325/18345		Standard Operating Conditions (unless otherwise stated)						
PIC16F18325/18345		**Standard Operating Conditions (unless otherwise stated)**						
Param. No.	**Symbol**	**Device Characteristics**	**Min.**	**Typ.†**	**Max.**	**Units**	**Conditions**	
							VDD	**Note**
D100	IDDXT4	XT = 4 MHz	—	321	455	uA	3.0V	
D100	IDDXT4	XT = 4 MHz	—	332	479	uA	3.0V	
D101	IDDHFO16	HFINTOSC = 16 MHz	—	1.3	1.8	mA	3.0V	
D101	IDDHFO16	HFINTOSC = 16 MHz	—	1.4	1.9	mA	3.0V	
D102	IDDHFOPLL	HFINTOSC = 32 MHz	—	2.2	2.8	mA	3.0V	
D102	IDDHFOPLL	HFINTOSC = 32 MHz	—	2.3	2.9	mA	3.0V	
D103	IDDHSPLL32	HS+PLL = 32 MHz	—	2.2	2.8	mA	3.0V	
D103	IDDHSPLL32	HS+PLL = 32 MHz	—	2.3	2.9	mA	3.0V	
D104	IDDIDLE	IDLE Mode, HFINTOSC = 16 MHz	—	804	1283	uA	3.0V	
D104	IDDIDLE	IDLE Mode, HFINTOSC = 16 MHz	—	816	1284	uA	3.0V	
D105	IDDDOZE(3)	DOZE mode, HFINTOSC = 16 MHz, DOZE Ratio = 16	—	863	—	uA	3.0V	
D105	IDDDOZE(3)	DOZE mode, HFINTOSC = 16 MHz, DOZE Ratio = 16	—	875	—	uA	3.0V	

† Data in “Typ.” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

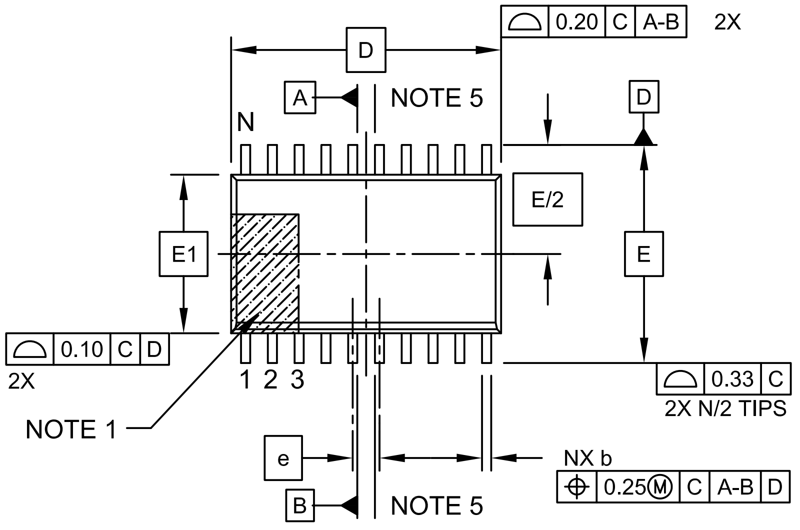
Note 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; $\overline{\text{MCLR}}$ = VDD; WDT disabled.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

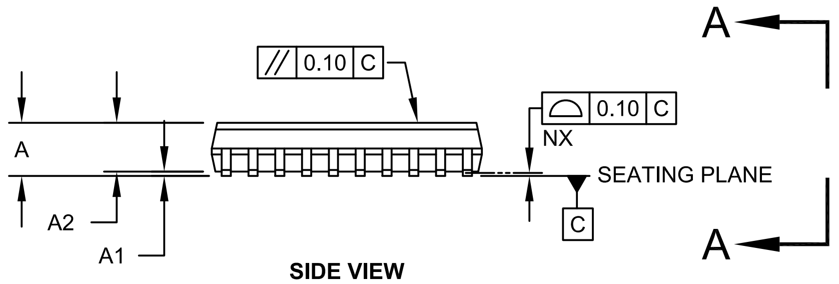
3: IDDDOZE = [IDDIDLE*(N-1)/N] + IDDHFO16/N where N = DOZE Ration (see Register 9-2).

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

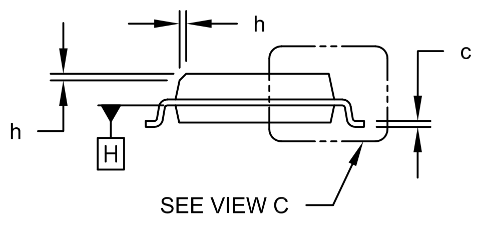
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW



SIDE VIEW



VIEW A-A

Microchip Technology Drawing C04-094C Sheet 1 of 2