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	12
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)	1.8V ~ 3.6V
Data Converters	A/D 11x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	14-SOIC (0.154", 3.90mm Width)
Supplier Device Package	14-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf18325-i-sl

4.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory
  - Configuration Words
  - Device ID
  - Revision ID
  - User ID
  - Program Flash Memory
- Data Memory
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM
- Data EEPROM

4.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing 32K x 14 program memory space. Table 4-1 shows the memory sizes implemented. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figure 4-1).

TABLE 4-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Size (Words)	Last Program Memory Address
PIC16(L)F18325/18345	8192	1FFFh

TABLE 4-4: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-31 (CONTINUED)

Address	Name	PIC16(L)F18325	PIC16(L)F18345	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
Bank 30													
CPU CORE REGISTERS; see Table 4-2 for specifics													
F0Ch	—	—		Unimplemented								—	—
F0Dh	—	—		Unimplemented								—	—
F0Eh	—	—		Unimplemented								—	—
F0Fh	CLCDATA			—	—	—	—	MLC4OUT	MLC3OUT	MLC2OUT	MLC1OUT	---- 0000	---- 0000
F10h	CLC1CON			LC1EN	—	LC1OUT	LC1INTP	LC1INTN	LC1MODE<2:0>			0-00 0000	0-00 0000
F11h	CLC1POL			LC1POL	—	—	—	LC1G4POL	LC1G3POL	LC1G2POL	LC1G1POL	0--- xxxx	0--- uuuu
F12h	CLC1SEL0			—	—	LC1D1S<5:0>						--xx xxxx	--uu uuuu
F13h	CLC1SEL1			—	—	LC1D2S<5:0>						--xx xxxx	--uu uuuu
F14h	CLC1SEL2			—	—	LC1D3S<5:0>						--xx xxxx	--uu uuuu
F15h	CLC1SEL3			—	—	LC1D4S<5:0>						--xx xxxx	--uu uuuu
F16h	CLC1GLS0			LC1G1D4T	LC1G1D4N	LC1G1D3T	LC1G1D3N	LC1G1D2T	LC1G1D2N	LC1G1D1T	LC1G1D1N	xxxx xxxx	uuuu uuuu
F17h	CLC1GLS1			LC1G2D4T	LC1G2D4N	LC1G2D3T	LC1G2D3N	LC1G2D2T	LC1G2D2N	LC1G2D1T	LC1G2D1N	xxxx xxxx	uuuu uuuu
F18h	CLC1GLS2			LC1G3D4T	LC1G3D4N	LC1G3D3T	LC1G3D3N	LC1G3D2T	LC1G3D2N	LC1G3D1T	LC1G3D1N	xxxx xxxx	uuuu uuuu
F19h	CLC1GLS3			LC1G4D4T	LC1G4D4N	LC1G4D3T	LC1G4D3N	LC1G4D2T	LC1G4D2N	LC1G4D1T	LC1G4D1N	xxxx xxxx	uuuu uuuu
F1Ah	CLC2CON			LC2EN	—	LC2OUT	LC2INTP	LC2INTN	LC2MODE<2:0>			0-00 0000	0-00 0000
F1Bh	CLC2POL			LC2POL	—	—	—	LC2G4POL	LC2G3POL	LC2G2POL	LC2G1POL	0--- xxxx	0--- uuuu
F1Ch	CLC2SEL0			—	—	LC2D1S<5:0>						--xx xxxx	--uu uuuu
F1Dh	CLC2SEL1			—	—	LC2D2S<5:0>						--xx xxxx	--uu uuuu
F1Eh	CLC2SEL2			—	—	LC2D3S<5:0>						--xx xxxx	--uu uuuu
F1Fh	CLC2SEL3			—	—	LC2D4S<5:0>						--xx xxxx	--uu uuuu

Legend: x = unknown, u = unchanged, q =depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations unimplemented, read as '0'.

Note 1: Only on PIC16F18325/18345.

2: Register accessible from both User and ICD Debugger.

6.10 Start-up Sequence

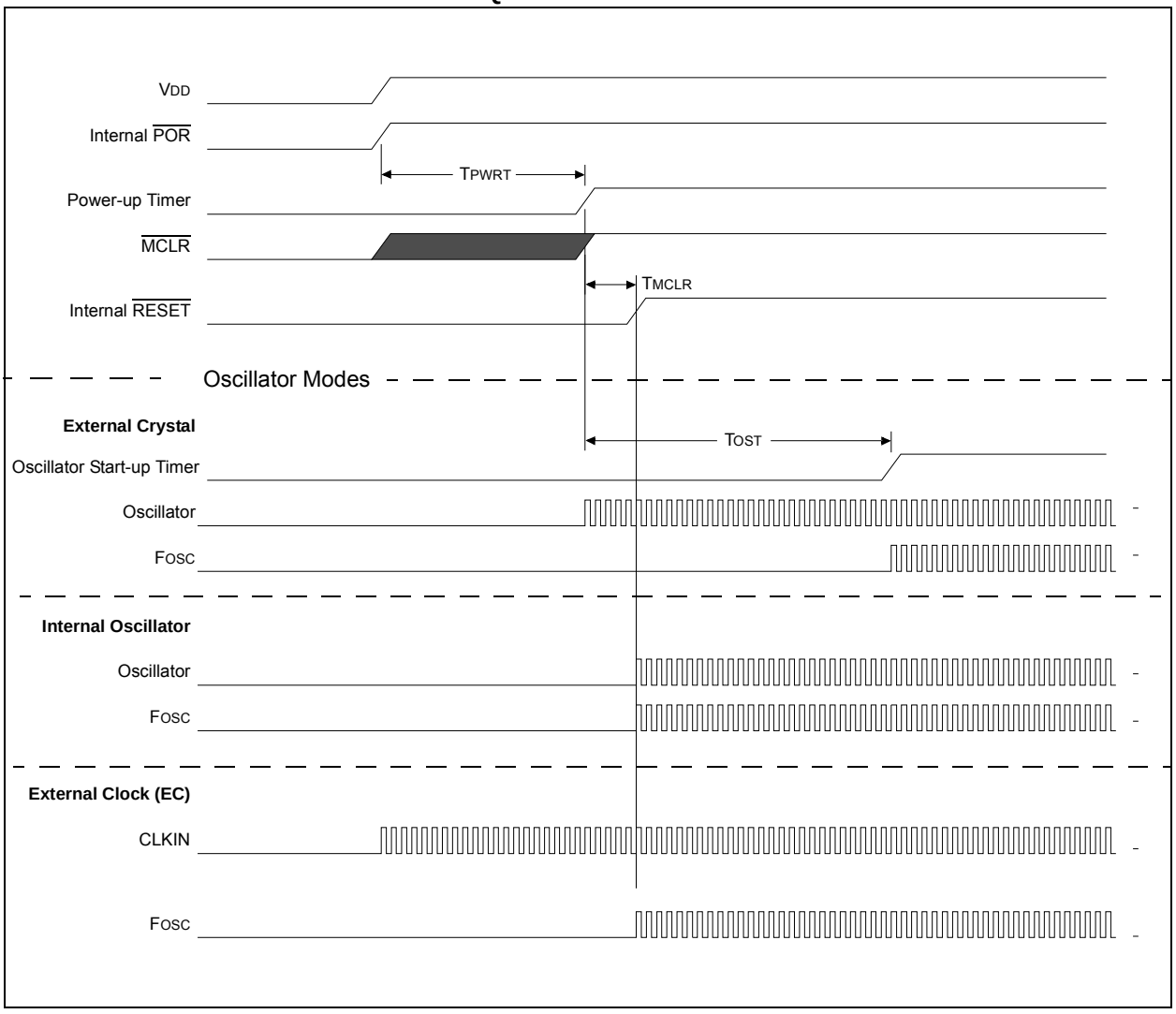
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. $\overline{\text{MCLR}}$ must be released (if enabled).
3. Oscillator start-up timer runs to completion (if required for oscillator source).

The total time-out will vary based on oscillator configuration and Power-up Timer Configuration. See **Section 7.0 "Oscillator Module"** for more information.

The Power-up Timer and oscillator start-up timer run independently of $\overline{\text{MCLR}}$ Reset. If $\overline{\text{MCLR}}$ is kept low long enough, the Power-up Timer will expire. Upon bringing $\overline{\text{MCLR}}$ high, the device will begin execution after 10 FOSC cycles (see Figure 6-3). This is useful for testing purposes or to synchronize more than one device operating in parallel.

FIGURE 6-3: RESET START-UP SEQUENCE



REGISTER 8-11: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4

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
CWG2IF	CWG1IF	TMR5GIF	TMR5IF	CCP4IF	CCP3IF	CCP2IF	CCP1IF
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 **CWG2IF:** CWG 2 Interrupt Flag bit
1 = CWG2 has gone into shutdown
0 = CWG2 is operating normally, or interrupt cleared

bit 6 **CWG1IF:** CWG1 Interrupt Flag bit
1 = CWG1 has gone into shutdown
0 = CWG1 is operating normally, or interrupt cleared

bit 5 **TMR5GIF:** Timer5 Gate Interrupt Flag bit
1 = The TMR5 gate has gone inactive (the gate is closed).
0 = The TMR5 gate has not gone inactive.

bit 4 **TMR5IF:** Timer5 Overflow Interrupt Flag bit
1 = TMR5 overflow occurred (must be cleared in software)
0 = No TMR5 overflow occurred

bit 3 **CCP4IF:** CCP4 Interrupt Flag bit

Value	CCPM Mode		
	Capture	Compare	PWM
1	Capture occurred (must be cleared in software)	Compare match occurred (must be cleared in software)	Output trailing edge occurred (must be cleared in software)
0	Capture did not occur	Compare match did not occur	Output trailing edge did not occur

bit 2 **CCP3IF:** CCP3 Interrupt Flag bit

Value	CCPM Mode		
	Capture	Compare	PWM
1	Capture occurred (must be cleared in software)	Compare match occurred (must be cleared in software)	Output trailing edge occurred (must be cleared in software)
0	Capture did not occur	Compare match did not occur	Output trailing edge did not occur

11.2 Data EEPROM

Data EEPROM consists of 256 bytes of user data memory. The EEPROM provides storage locations for 8-bit user defined data.

EEPROM can be read and/or written through:

- FSR/INDF indirect access (**Section 11.3 “FSR and INDF Access”**)
- NVMREG access (**Section 11.4 “NVMREG Access”**)
- External device programmer

Unlike program Flash memory, which must be written to by row, EEPROM can be written to byte by byte.

11.3 FSR and INDF Access

The FSR and INDF registers allow indirect access to the program Flash memory or EEPROM.

11.3.1 FSR READ

With the intended address loaded into an FSR register, a `MOVIW` instruction or read of INDF will read data from the Program Flash Memory or EEPROM. The CPU operation is suspended during the read, and resumes immediately after. Read operations return a single word of memory. When the MSB of the FSR (ex: FSRxH) is set to 0x70, the lower 8-bit address value (in FSRxL) determines the EEPROM location that may be read from (through the INDF register). In other words, the EEPROM address range 0x00-0xFF is mapped into the FSR address space between 0x7000-0x70FF. Writing to the EEPROM cannot be accomplished via the FSR/INDF interface.

11.3.2 FSR WRITE

Writing/erasing the NVM through the FSR registers (ex. `MOVWI` instruction) is not supported in the PIC16(L)F18325/18345 devices.

REGISTER 12-11: LATB: PORTB DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0
LATB7	LATB6	LATB5	LATB4	—	—	—	—
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 7-4 **LATB<7:4>**: RB<5:4> Output Latch Value bits(1)

bit 3-0 **Unimplemented:** Read as '0'

Note 1: Writes to LATB are equivalent with writes to the corresponding PORTB register.Reads from LATB register return register values, not I/O pin values.

REGISTER 12-12: ANSELB: PORTB ANALOG SELECT REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
ANSB7	ANSB6	ANSB5	ANSB4	—	—	—	—
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 7-4 **ANSB<7:4>**: Analog Select between Analog or Digital Function
0 = Digital I/O. Pin is assigned to port or digital special function.
1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.

bit 3-0 **Unimplemented:** Read as '0'

Note 1: Setting ANSB[n] = 1 disables the digital input circuitry. Weak pull-ups, if available, are unaffected. The corresponding TRIS bit must be set to Input mode by the user to allow external control of the voltage on the pin.

12.6 PORTC Registers

12.6.1 DATA REGISTER

PORTC is a bidirectional port that is either 6-bit wide (PIC16(L)F18325) or 8-bit wide (PIC16(L)F18345). The corresponding data direction register is TRISC (Register 12-18). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 12-17) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

The PORT data latch LATC (Register 12-19) holds the output port data, and contains the latest value of a LATC or PORTC write.

12.6.2 DIRECTION CONTROL

The TRISC register (Register 12-18) controls the PORTC pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISC register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

12.6.3 INPUT THRESHOLD CONTROL

The INLVLC register (Register 12-24) controls the input voltage threshold for each of the available PORTC input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTC register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 35-4 for more information on threshold levels.

Note:	Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

12.6.4 OPEN-DRAIN CONTROL

The ODCONC register (Register 12-22) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONC bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONC bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

Note:	It is not necessary to set open-drain control when using the pin for I^2C; the I^2C module controls the pin and makes the pin open-drain.

12.6.5 SLEW RATE CONTROL

The SLRCONC register (Register 12-23) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONC bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONC bit is cleared, The corresponding port pin drive slews at the maximum rate available.

12.6.6 ANALOG CONTROL

The ANSELC register (Register 12-20) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELC bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

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

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

12.6.7 WEAK PULL-UP CONTROL

The WPUC register (Register 12-21) controls the individual weak pull-ups for each PORT pin.

12.6.8 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other functions are selected with the peripheral pin select logic. See **Section 13.0 "Peripheral Pin Select (PPS) Module"** for more information.

Analog output functions, such as ADC and comparator inputs, are not shown in the peripheral pin select lists. Digital output functions may continue to control the pin when it is in Analog mode.

TABLE 20-2: SUMMARY OF REGISTERS ASSOCIATED WITH CWGx

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
TRISA	—	—	TRISA5	TRISA4	—(2)	TRISA2	TRISA1	TRISA0	143
ANSELA	—	—	ANSA5	ANSA4	—	ANSA2	ANSA1	ANSA0	144
TRISB(1)	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	149
ANSELB(1)	ANSB7	ANSB6	ANSB5	ANSB4	—	—	—	—	150
TRISC	TRISC7(1)	TRISC6(1)	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	156
ANSELC	ANSC7(1)	ANSC6(1)	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0	157
PIR4	CWG2IF	CWG1IF	TMR5GIF	TMR5IF	CCP4IF	CCP3IF	CCP2IF	CCP1IF	111
PIE4	CWG2IE	CWG1IE	TMR5GIE	TMR5IE	CCP4IE	CCP3IE	CCP2IE	CCP1IE	106
CWG1CON0	EN	LD	—	—	—	MODE<2:0>			213
CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	214
CWG1CLKCON	—	—	—	—	—	—	—	CS	214
CWG1DAT	—	—	—	—	DAT<3:0>				215
CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	216
CWG1AS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—	217
CWG1AS1	—	—	—	AS4E	AS3E	AS2E	AS1E	AS0E	218
CWG1DBR	—	—	DBR<5:0>						218
CWG1DBF	—	—	DBF<5:0>						219
CWG1PPS	—	—	—	CWG1PPS<4:0>					162
CWG2CON0	EN	LD	—	—	—	MODE<2:0>			213
CWG2CON1	—	—	IN	—	POLD	POLC	POLB	POLA	214
CWG2CLKCON	—	—	—	—	—	—	—	CS	214
CWG2DAT	—	—	—	—	DAT<3:0>				215
CWG2STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	216
CWG2AS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—	217
CWG2AS1	—	—	—	AS4E	AS3E	AS2E	AS1E	AS0E	218
CWG2DBR	—	—	DBR<5:0>						218
CWG2DBF	—	—	DBF<5:0>						219
CWG2PPS	—	—	—	CWG2PPS<4:0>					162

Legend: – = unimplemented location, read as '0'. Shaded cells are not used by interrupts.

Note 1: PIC16(L)F18345 only.
2: Unimplemented, read as '0'.

REGISTER 22-6: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	—	—	—	—	ADRES<9:8>	
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 7-2 **Reserved**: Do not use.

bit 1-0 **ADRES<9:8>**: ADC Result Register bits
Upper two bits of 10-bit conversion result

REGISTER 22-7: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<7:0>							
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 7-0 **ADRES<7:0>**: ADC Result Register bits
Lower eight bits of 10-bit conversion result

REGISTER 23-4: NCO1ACCH: NCO1 ACCUMULATOR REGISTER – HIGH BYTE

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
NCO1ACC<15:8>							
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 7-0 **NCO1ACC<15:8>:** NCO1 Accumulator, high byte

REGISTER 23-5: NCO1ACCU: NCO1 ACCUMULATOR REGISTER – UPPER BYTE(1)

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	NCO1ACC<19:16>			
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 7-4 **Unimplemented:** Read as ‘0’

bit 3-0 **NCO1ACC<19:16>:** NCO1 Accumulator, upper byte

Note 1: The accumulator spans registers NCO1ACCU:NCO1ACCH:NCO1ACCL. The 24 bits are reserved but not all are used.This register updates in real-time, asynchronously to the CPU; there is no provision to guarantee atomic access to this 24-bit space using an 8-bit bus. Writing to this register while the module is operating will produce undefined results.

REGISTER 23-6: NCO1INCL(1,2): NCO1 INCREMENT REGISTER – LOW BYTE

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-1/1
NCO1INC<7:0>							
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 7-0 **NCO1INC<7:0>:** NCO1 Increment, low byte

Note 1: The logical increment spans NCO1INCU:NCO1INCH:NCO1INCL.

2: NCO1INC is double-buffered as INCBUF; INCBUF is updated on the next falling edge of NCOCLK after writing to NCO1INCL;NCO1INCU and NCO1INCH should be written prior to writing NCO1INCL.

TABLE 26-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER0

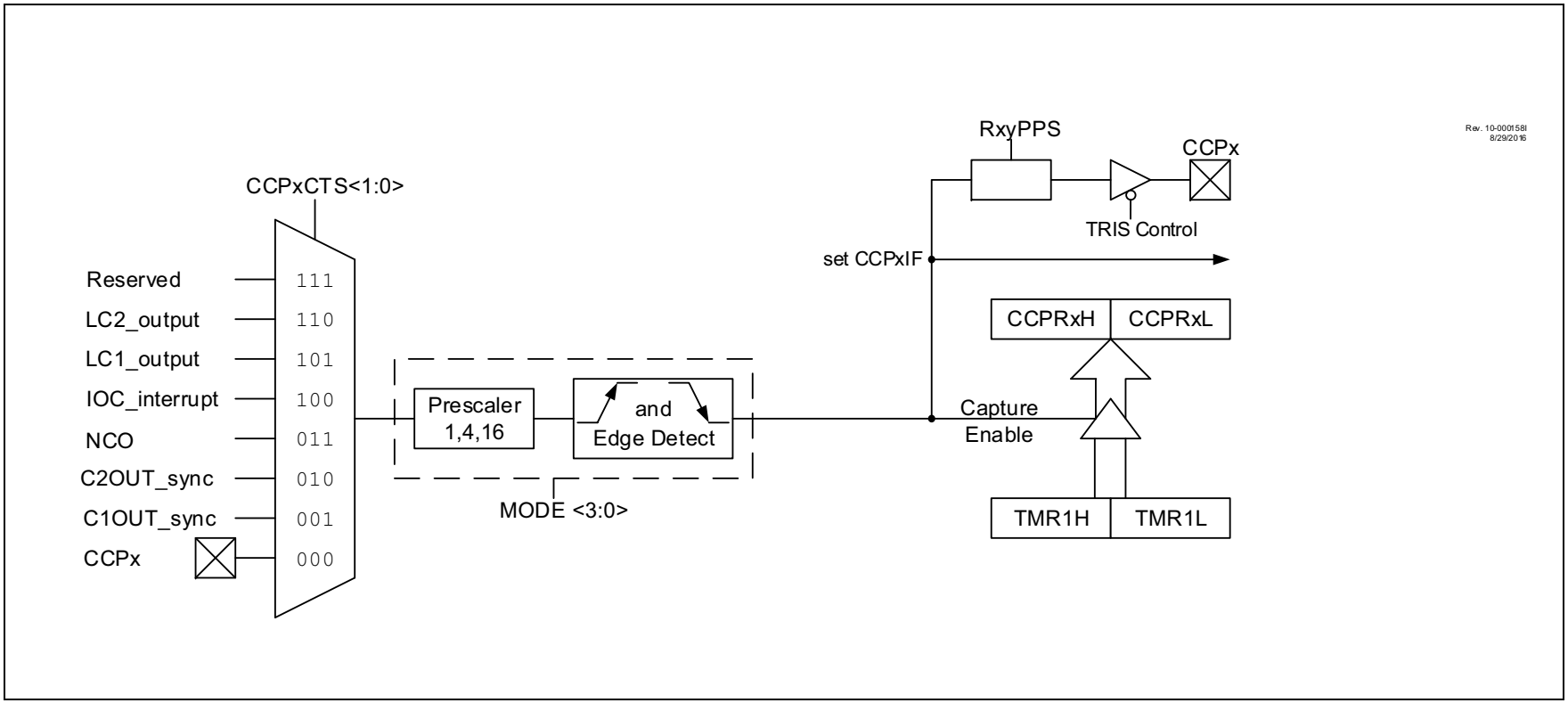
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
TRISA	—	—	TRISA5	TRISA4	—(2)	TRISA2	TRISA1	TRISA0	143
ANSELA	—	—	ANSA5	ANSA4	—	ANSA2	ANSA1	ANSA0	144
TRISB(1)	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	149
ANSELB(1)	ANSB7	ANSB6	ANSB5	ANSB4	—	—	—	—	150
TRISC	TRISC7(1)	TRISC6(1)	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	156
ANSELC	ANSC7(1)	ANSC6(1)	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0	157
TMR0L	TMR0L<7:0>								279
TMR0H	TMR0H<7:0> or TMR0<15:8>								279
T0CON0	T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>				280
T0CON1	T0CS<2:0>			T0ASYNC	T0CKPS<3:0>				281
T0CKIPPS	—	—	—	T0CKIPPS<4:0>					162
TMR0PPS	—	—	—	TMR0PPS<4:0>					162
ADACT	—	—	—	ADACT<4:0>					246
CLCxSELy	—	—	LCxDyS<5:0>						229
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{DONE}}$	T1GVAL	T1GSS<1:0>		293
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	101
PIR0	—	—	TMR0IF	IOCIF	—	—	—	INTF	107
PIE0	—	—	TMR0IE	IOCIE	—	—	—	INTE	102

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the Timer0 module.

Note 1: PIC16(L)F18345 only.

2: Unimplemented, read as '1'.

FIGURE 29-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



30.2.3 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK line. The master determines when the slave (Processor 2, Figure 30-5) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and Status bits appropriately set).

The clock polarity is selected by appropriately programming the CKP bit of the SSPxCON1 register and the CKE bit of the SSPxSTAT register. This then, would give waveforms for SPI communication as shown in Figure 30-6, Figure 30-8, Figure 30-9 and Figure 30-10, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 * TCY)
- FOSC/64 (or 16 * TCY)
- Timer2 output/2
- FOSC/(4 * (SSPADD + 1))

Figure 30-6 shows the waveforms for Master mode.

When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 30-6: SPI MODE WAVEFORM (MASTER MODE)

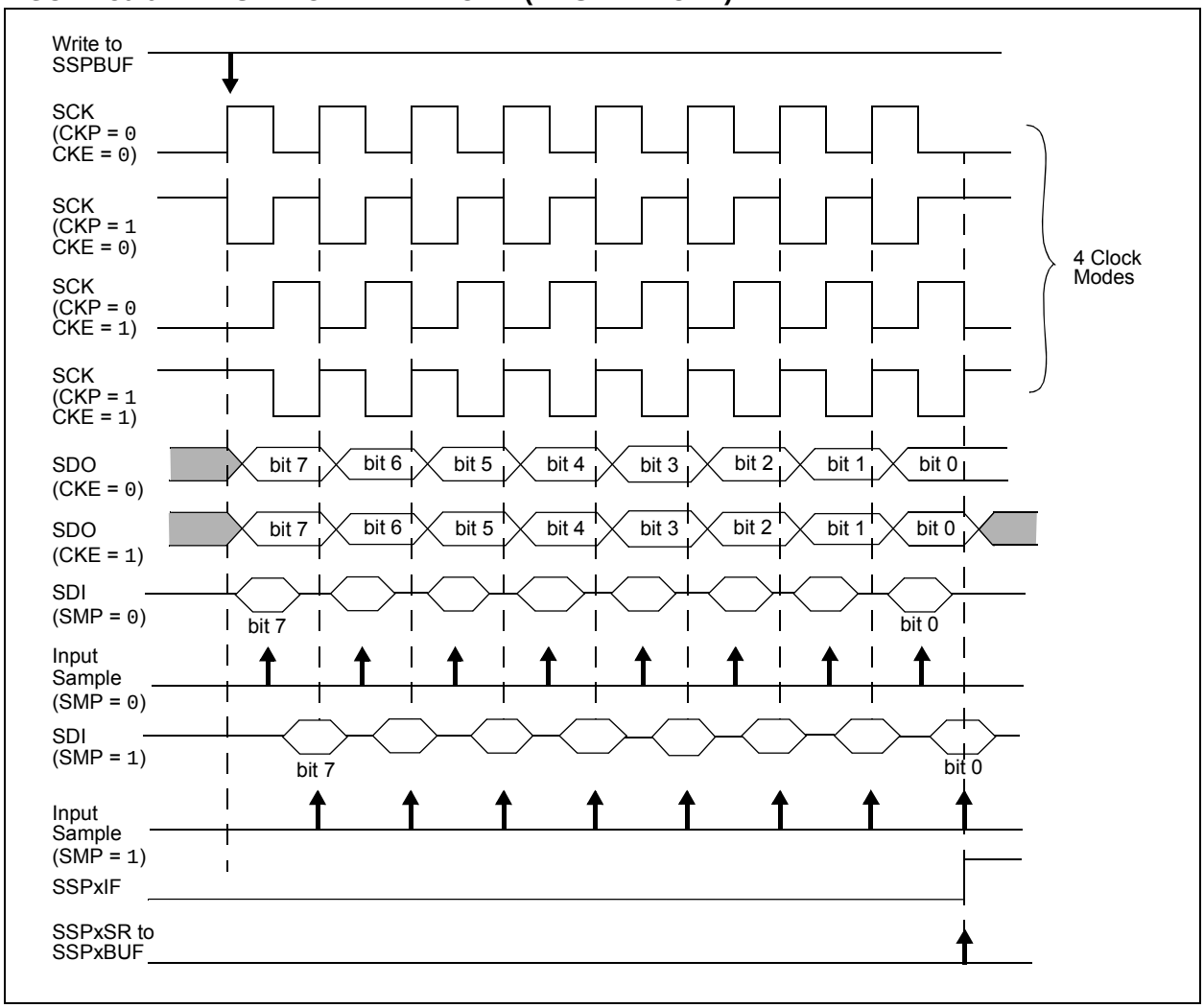
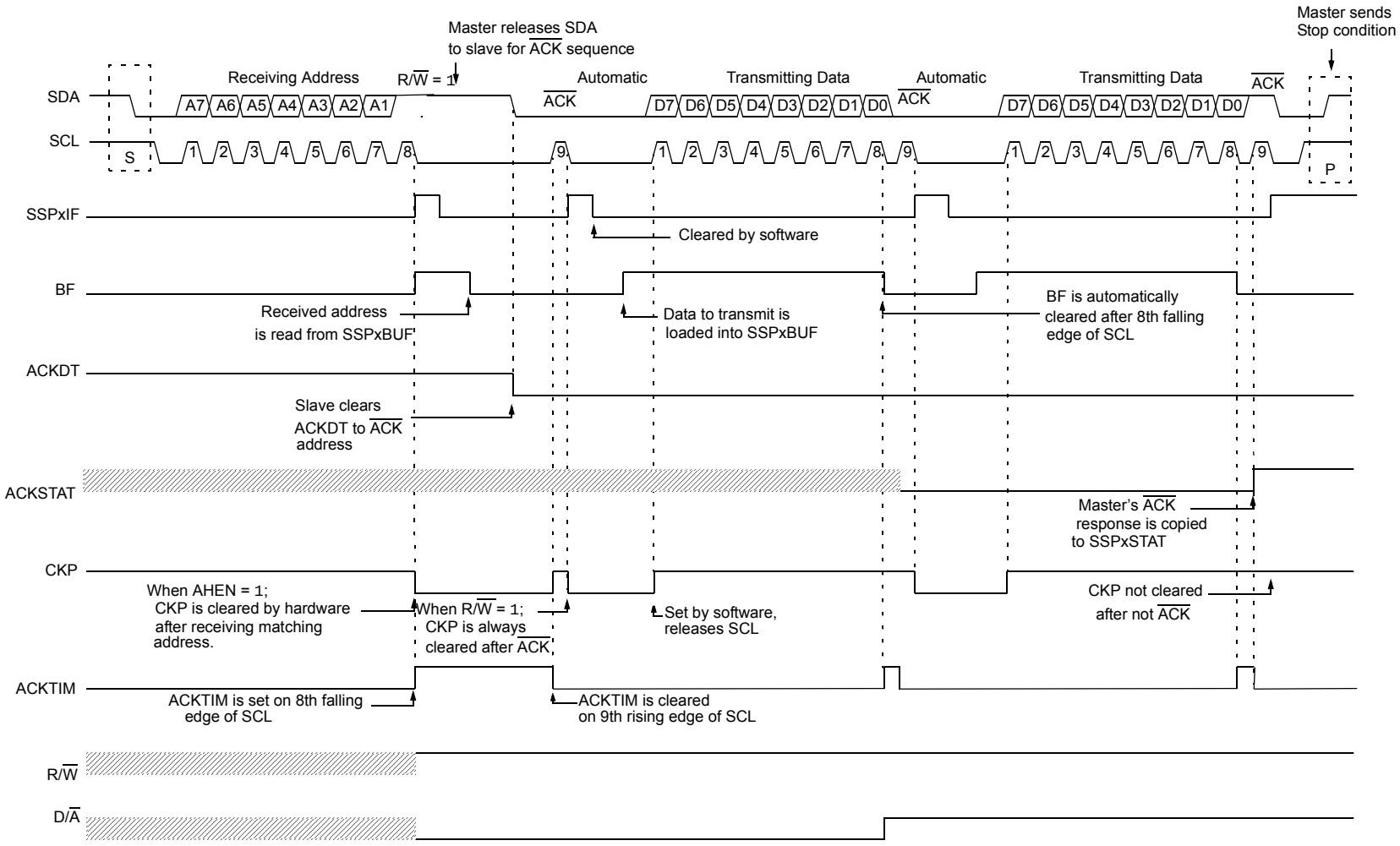


FIGURE 30-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)



30.6.8 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN bit of the SSPxCON2 register. When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSPx module then goes into Idle mode (Figure 30-30).

30.6.8.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

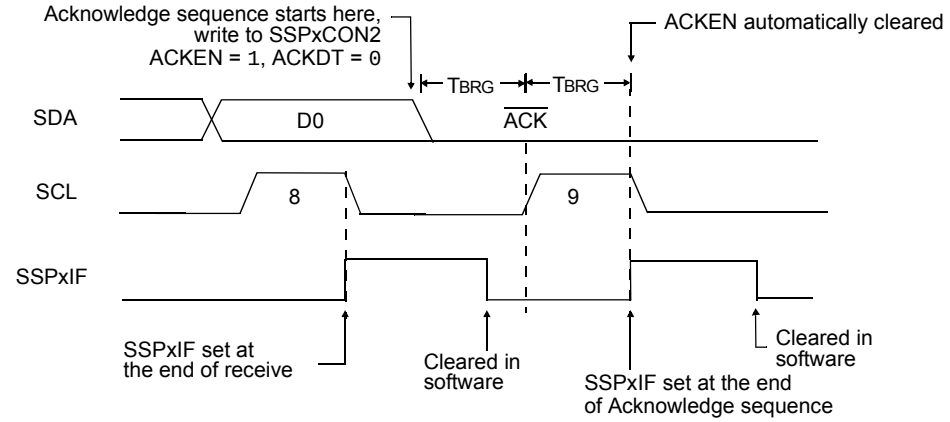
30.6.9 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN bit of the SSPxCON2 register. At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit of the SSPxSTAT register is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 30-31).

30.6.9.1 WCOL Status Flag

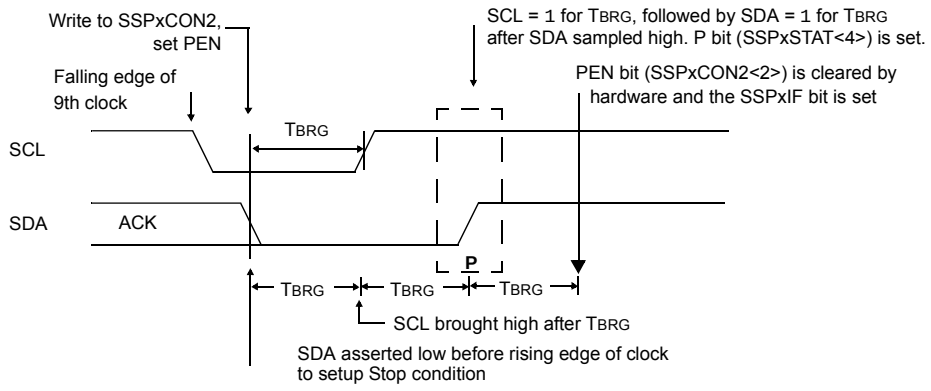
If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

FIGURE 30-30: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 30-31: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one Baud Rate Generator period.

FIGURE 31-7: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION

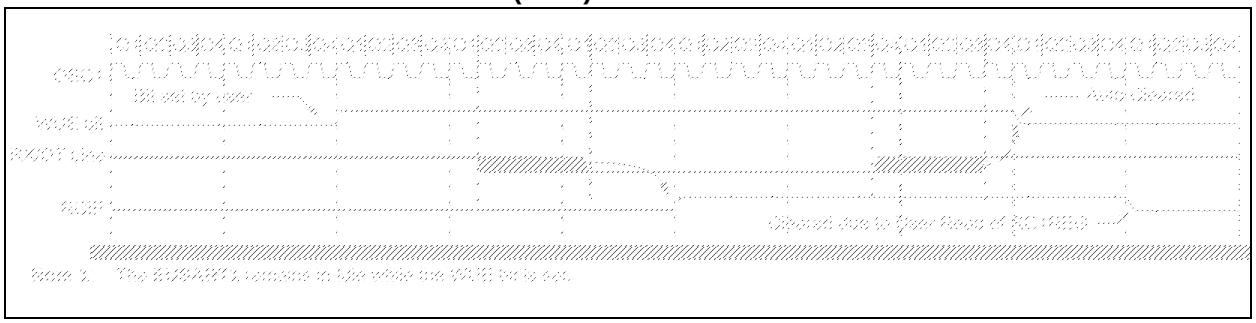
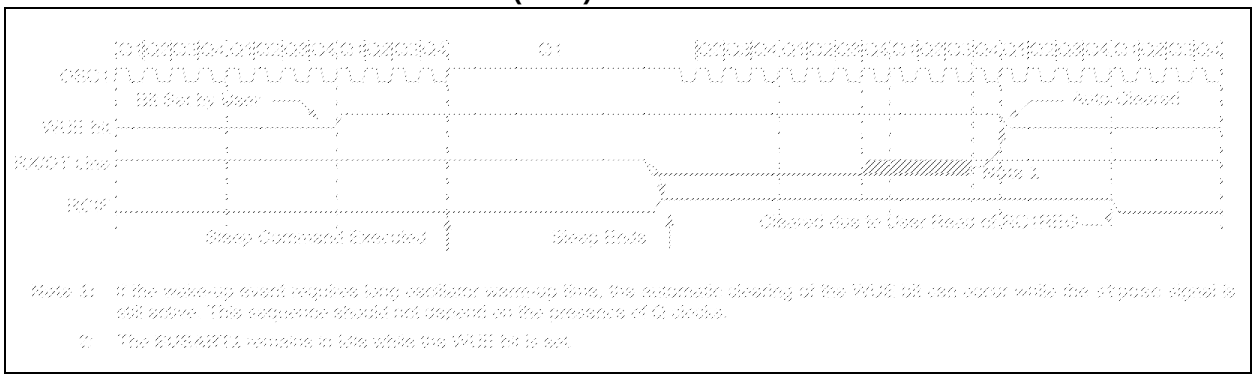


FIGURE 31-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



31.3.4 BREAK CHARACTER SEQUENCE

The EUSART1 module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TX1STA register. The Break character transmission is then initiated by a write to the TX1REG. The value of data written to TX1REG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TX1STA register indicates when the transmit operation is active or idle, just as it does during normal transmission. See Figure 31-9 for the timing of the Break character sequence.

31.3.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART1 for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TX1REG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TX1REG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TX1REG becomes empty, as indicated by the TXIF, the next data byte can be written to TX1REG.

REGISTER 31-4: RC1REG(1): RECEIVE DATA REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RC1REG<7:0>							
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 7-0 **RC1REG<7:0>:** Lower eight bits of the received data; read-only; see also RX9D (Register 31-2)

Note 1: RC1REG (including the ninth bit) is double buffered, and data is available while new data is being received.

REGISTER 31-5: TX1REG(1): TRANSMIT DATA REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX1REG<7:0>							
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 7-0 **TX1REG<7:0>:** Lower eight bits of the received data; read-only; see also RX9D (Register 31-1)

Note 1: TX1REG (including the ninth bit) is double buffered, and can be written when previous data has started shifting.

REGISTER 31-6: SP1BRGL(1): BAUD RATE GENERATOR REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SP1BRG<7:0>							
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 7-0 **SP1BRG<7:0>:** Lower eight bits of the Baud Rate Generator

Note 1: Writing to SP1BRG resets the BRG counter.

CALL **Call Subroutine**

Syntax:	[*label*] CALL k
Operands:	$0 \le k \le 2047$
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<6:3>) → PC<14:11>
Status Affected:	None
Description:	Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

CALLW **Subroutine Call With W**

Syntax:	[*label*] CALLW
Operands:	None
Operation:	(PC) +1 → TOS, (W) → PC<7:0>, (PCLATH<6:0>) → PC<14:8>
Status Affected:	None
Description:	Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

CLRF **Clear f**

Syntax:	[*label*] CLRF f
Operands:	$0 \le f \le 127$
Operation:	00h → (f) 1 → Z
Status Affected:	Z
Description:	The contents of register 'f' are cleared and the Z bit is set.

CLRW **Clear W**

Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → (W) 1 → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

CLRWDT **Clear Watchdog Timer**

Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT 0 → WDT prescaler, 1 → $\overline{TO}$ 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

COMF **Complement f**

Syntax:	[*label*] COMF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	$(\bar{f}) \rightarrow$ (destination)
Status Affected:	Z
Description:	The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

DECF **Decrement f**

Syntax:	[*label*] DECF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → (destination)
Status Affected:	Z
Description:	Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

MOVWI **Move W to INDFn**

Syntax: [*label*] MOVWI ++FSRn
[*label*] MOVWI --FSRn
[*label*] MOVWI FSRn++
[*label*] MOVWI FSRn--
[*label*] MOVWI k[FSRn]

Operands: n ∈ [0,1]
mm ∈ [00,01, 10, 11]
-32 ≤ k ≤ 31

Operation: W → INDFn
Effective address is determined by
- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:
- FSR + 1 (all increments)
- FSR - 1 (all decrements)

Unchanged

Status Affected: None

Mode	Syntax	mm
Preincrement	++FSRn	00
Predecrement	--FSRn	01
Postincrement	FSRn++	10
Postdecrement	FSRn--	11

Description: This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

Note: The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h-FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

The increment/decrement operation on FSRn WILL NOT affect any Status bits.

NOP **No Operation**

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Description: No operation.

Words: 1

Cycles: 1

Example:
```
NOP
```

RESET **Software Reset**

Syntax: [*label*] RESET

Operands: None

Operation: Execute a device Reset. Resets the $\overline{RI}$ flag of the PCON register.

Status Affected: None

Description: This instruction provides a way to execute a hardware Reset by software.

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE k

Operands: None

Operation: TOS → PC,
1 → GIE

Status Affected: None

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:
```
RETFIE
```
After Interrupt
PC = TOS
GIE = 1

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**	
							V_{DD}	**Note**
D100	$IDD_{XT}4$	XT = 4 MHz	—	321	455	uA	3.0V	
D100	$IDD_{XT}4$	XT = 4 MHz	—	332	479	uA	3.0V	
D101	$IDD_{HFO}16$	HFINTOSC = 16 MHz	—	1.3	1.8	mA	3.0V	
D101	$IDD_{HFO}16$	HFINTOSC = 16 MHz	—	1.4	1.9	mA	3.0V	
D102	IDD_{HFOPLL}	HFINTOSC = 32 MHz	—	2.2	2.8	mA	3.0V	
D102	IDD_{HFOPLL}	HFINTOSC = 32 MHz	—	2.3	2.9	mA	3.0V	
D103	$IDD_{HSPLL}32$	HS+PLL = 32 MHz	—	2.2	2.8	mA	3.0V	
D103	$IDD_{HSPLL}32$	HS+PLL = 32 MHz	—	2.3	2.9	mA	3.0V	
D104	IDD_{IDLE}	IDLE Mode, HFINTOSC = 16 MHz	—	804	1283	uA	3.0V	
D104	IDD_{IDLE}	IDLE Mode, HFINTOSC = 16 MHz	—	816	1284	uA	3.0V	
D105	IDD_{DOZE}(3)	DOZE mode, HFINTOSC = 16 MHz, DOZE Ratio = 16	—	863	—	uA	3.0V	
D105	IDD_{DOZE}(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 I_{DD} measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD}; $\overline{MCLR}$ = V_{DD}; 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: IDD_{DOZE} = [IDD_{IDLE}*(N-1)/N] + IDD_{HFO}16/N where N = DOZE Ration (see Register 9-2).