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	25
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 24x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf18855-e-ss

TABLE 3: 40/44-PIN ALLOCATION TABLE (PIC16F18875) (CONTINUED)

I/O	40-Pin PDIP	40-Pin UQFN	44-Pin QFN	44-Pin TQFP	ADC	Voltage Reference	DAC	Comparators	Zero-Cross Detect	MSSP (SPI/I2C)	EUSART	DSM	Timers/SMT	CCP and PWM	CWG	CLC	NCO	Clock Reference (CLKR)	Interrupt-on-Change	Basic
RC0	15	30	34	32	ANC0	—	—	—	—	—	—	—	T1CKI(1) T3CKI(1) T3G(1) SMTWIN1(1)	—	—	—	—	—	IOCC0	SOSCO
RC1	16	31	35	35	ANC1	—	—	—	—	—	—	—	SMTSIG1(1)	CCP2(1)	—	—	—	—	IOCC1	SOSCI
RC2	17	32	36	36	ANC2	—	—	—	—	—	—	—	T5CKI(1)	CCP1(1)	—	—	—	—	IOCC2	—
RC3	18	33	37	37	ANC3	—	—	—	—	SCL1(3,4) SCK1(1)	—	—	T2IN(1)	—	—	—	—	—	IOCC3	—
RC4	23	38	42	42	ANC4	—	—	—	—	SDA1(3,4) SDI1(1)	—	—	—	—	—	—	—	—	IOCC4	—
RC5	24	39	43	43	ANC5	—	—	—	—	—	—	—	T4IN(1)	—	—	—	—	—	IOCC5	—
RC6	25	40	44	44	ANC6	—	—	—	—	—	CK(3)	—	—	—	—	—	—	—	IOCC6	—
RC7	26	1	1	1	ANC7	—	—	—	—	—	RX(1) DT(3)	—	—	—	—	—	—	—	IOCC7	—
RD0	19	34	38	38	AND0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD1	20	35	39	39	AND1	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD2	21	36	40	40	AND2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD3	22	37	41	41	AND3	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD4	27	2	2	2	AND4	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD5	28	3	3	3	AND5	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD6	29	4	4	4	AND6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RD7	30	5	5	5	AND7	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RE0	8	23	25	25	ANE0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RE1	9	24	26	26	ANE1	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RE2	10	25	27	27	ANE2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Note 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to Table 13-1 for details on which port pins may be used for this signal.

2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in Table 13-3.

3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.

4: These pins are configured for I^2C logic levels.; The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I^2C specific or SMbus input buffer thresholds.

3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

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

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing
- NVMREG access

3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing 32K x 14 program memory space. Table 3-1 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 3-1).

TABLE 3-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Size (Words)	Last Program Memory Address
PIC16(L)F18855/75	8192	1FFFh

TABLE 3-6: PIC16F18855/75 MEMORY MAP BANK 16-23

	BANK 16		BANK 17		BANK 18		BANK 19		BANK 20		BANK 21		BANK 22		BANK 23	
800h	Core Registers (Table 3-2)	880h	Core Registers (Table 3-2)	900h	Core Registers (Table 3-2)	980h	Core Registers (Table 3-2)	A00h	Core Registers (Table 3-2)	A80h	Core Registers (Table 3-2)	B00h	Core Registers (Table 3-2)	B80h	Core Registers (Table 3-2)
80Bh		88Bh		90Bh		98Bh		A0Bh		A8Bh		B0Bh		B8Bh	
80Ch	WDTCON0	88Ch	CPUDOZE	90Ch	FVRCON	98Ch	—	A0Ch	Unimplemented Read as '0'	A8Ch	Unimplemented Read as '0'	B0Ch	Unimplemented Read as '0'	B8Ch	Unimplemented Read as '0'
80Dh	WDTCON1	88Dh	OSCCON1	90Dh	—	98Dh	—								
80Eh	WDTPSL	88Eh	OSCCON2	90Eh	DAC1CON0	98Eh	—								
80Fh	WDTPSH	88Fh	OSCCON3	90Fh	DAC1CON1	98Fh	CMOUT								
810h	WDTTMR	890h	OSCSTAT	910h	—	990h	CM1CON0								
811h	BORCON	891h	OSCEN	911h	—	991h	CM1CON1								
812h	VREGCON(1)	892h	OSCTUNE	912h	—	992h	CM1NSEL								
813h	PCON0	893h	OSCFRQ	913h	—	993h	CM1PSEL								
814h	CCDCON	894h	—	914h	—	994h	CM2CON0								
815h	—	895h	CLKRCON	915h	—	995h	CM2CON1								
816h	—	896h	CLKRCLK	916h	—	996h	CM2NSEL								
817h	—	897h	MDCON0	917h	—	997h	CM2PSEL								
818h	—	898h	MDCON1	918h	—	998h	—								
819h	—	899h	MDSRC	919h	—	999h	—								
81Ah	NVMADRL	89Ah	MDCARL	91Ah	—	99Ah	—								
81Bh	NVMADRH	89Bh	MDCARH	91Bh	—	99Bh	—								
81Ch	NVMDATL	89Ch	—	91Ch	—	99Ch	—								
81Dh	NVMDATH	89Dh	—	91Dh	—	99Dh	—								
81Eh	NVMCON1	89Eh	—	91Eh	—	99Eh	—								
81Fh	NVMCON2	89Fh	—	91Fh	ZCDCON	99Fh	—								
820h	Unimplemented Read as '0'	8A0h	Unimplemented Read as '0'	920h	Unimplemented Read as '0'	9A0h	Unimplemented Read as '0'								
86Fh		8EFh		96Fh		9EFh		A6Fh		AEFh		B6Fh		BEFh	
870h	Common RAM Accesses 70h – 7Fh	8F0h	Common RAM Accesses 70h – 7Fh	970h	Common RAM Accesses 70h – 7Fh	9F0h	Common RAM Accesses 70h – 7Fh	A70h	Common RAM Accesses 70h – 7Fh	AF0h	Common RAM Accesses 70h – 7Fh	B70h	Common RAM Accesses 70h – 7Fh	BF0h	Common RAM Accesses 70h – 7Fh
87Fh		8FFh		97Fh		9FFh		A7Fh		AFFh		B7Fh		BFFh	

Legend: = Unimplemented data memory locations, read as '0'.

Note 1: PIC16F18855/75 only.

TABLE 3-11: PIC16(L)F18875 MEMORY MAP, BANK 30

	Bank 30
F0Ch	—
F0Dh	—
F0Eh	—
F0Fh	—
F10h	RA0PPS
F11h	RA1PPS
F12h	RA2PPS
F13h	RA3PPS
F14h	RA4PPS
F15h	RA5PPS
F16h	RA6PPS
F17h	RA7PPS
F18h	RB0PPS
F19h	RB1PPS
F1Ah	RB2PPS
F1Bh	RB3PPS
F1Ch	RB4PPS
F1Dh	RB5PPS
F1Eh	RB6PPS
F1Fh	RB7PPS
F20h	RC0PPS
F21h	RC1PPS
F22h	RC2PPS
F23h	RC3PPS
F24h	RC4PPS
F25h	RC5PPS
F26h	RC6PPS
F27h	RC7PPS
F28h — F37h	—
F38h	ANSELA
F39h	WPUA
F3Ah	ODCONA
F3Bh	SLRCONA
F3Ch	INLVLA
F3Dh	IOCAP
F3Eh	IOCAN
F3Fh	IOCAF

	Bank 30
F40h	CCDNA
F41h	CCDPA
F42h	—
F43h	ANSELB
F44h	WPUB
F45h	ODCONB
F46h	SLRCONB
F47h	INLVLB
F48h	IOCBP
F49h	IOCBN
F4Ah	IOCBF
F4Bh	CCDNB
F4Ch	CCDPB
F4Dh	—
F4Eh	ANSELC
F4Fh	WPUC
F50h	ODCONC
F51h	SLRCONC
F52h	INLVLC
F53h	IOCCP
F54h	IOCCN
F55h	IOCCF
F56h	CCDNC
F57h	CCDPC
F58h	—
F59h	ANSELD
F5Ah	WPUD
F5Bh	ODCOND
F5Ch	SLRCOND
F5Dh	INLVLD
F5Eh	—
F5Fh	—
F60h	—
F61h	CCDND
F62h	CCDPD
F63h	—

	Bank 30
F64h	ANSELE
F65h	WPUE
F66h	ODCONE
F67h	SLRCONE
F68h	INLVLE
F69h	IOCEP
F6Ah	IOCEN
F6Bh	IOCEF
F6Ch	CCDNE
F6Dh	CCDPE
F6Eh	—
F6Fh	—

Legend: = Unimplemented data memory locations, read as '0'.

TABLE 3-13: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-31 (CONTINUED)

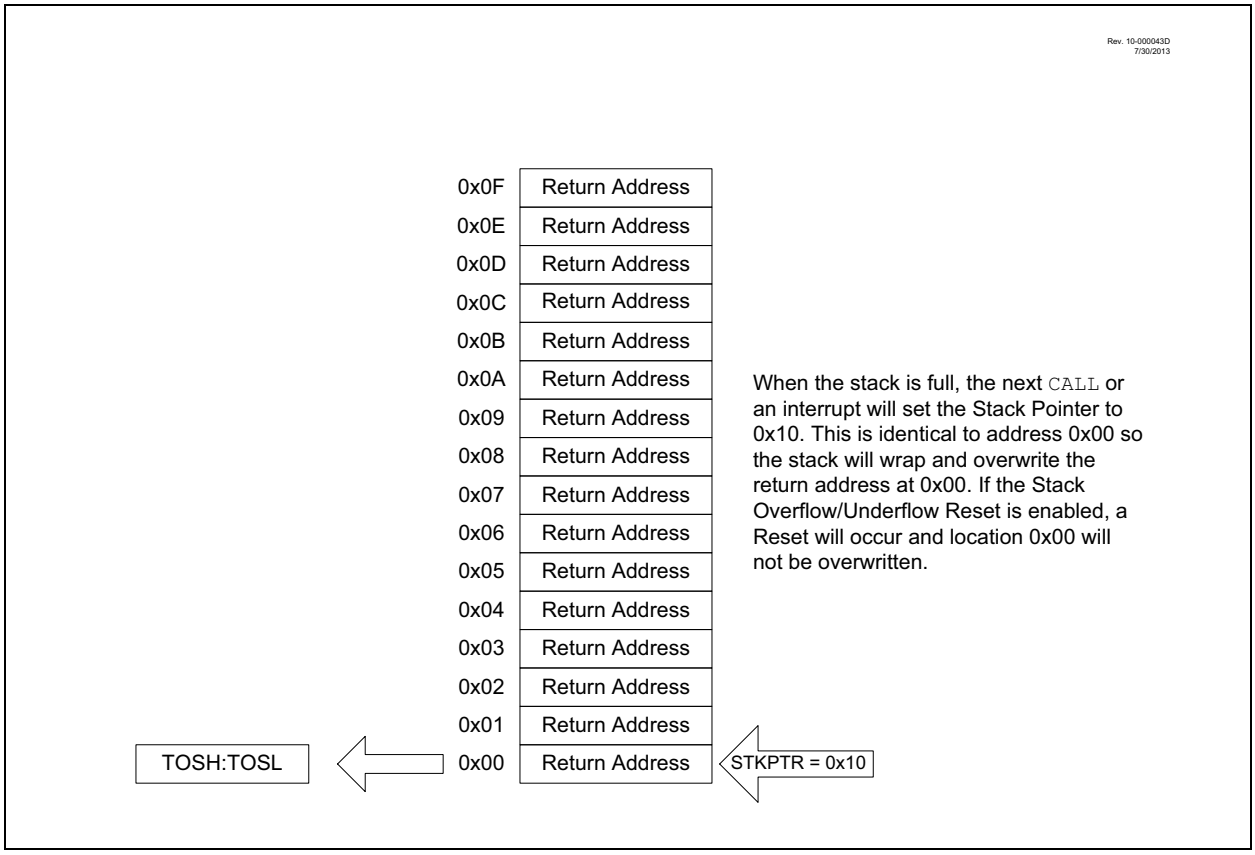
Address	Name	PIC16(L)F18855	PIC16(L)F18875	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
Banks 16													
CPU CORE REGISTERS; see Table 3-2 for specifics													
80Ch	WDTCON0			—	—	PS<4:0>					SEN	--qq qqq0	--qq qqq0
80Dh	WDTCON1			—	WDTCS<2:0>			—	WINDOW<2:0>			-qqq -qqq	-qqq -qqq
80Eh	WDTPSL			PSCNT<7:0>								0000 0000	0000 0000
80Fh	WDTPSH			PSCNT<7:0>								0000 0000	0000 0000
810h	WDTTMR			—	WDTTMR<3:0>				STATE	PSCNT<17:16>		-000 0000	-000 0000
811h	BORCON			SBOREN	—	—	—	—	—	—	BORRDY	1--- ---q	u--- ---u
812h	VREGCON(1)			—	—	—	—	—	—	VREGPM	Reserved	---- --01	---- --01
813h	PCON0			STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	0011 11qq	qqqq qquu
814h	CCDCON			CCDEN	—	—	—	—	—	CCDS<1:0>		0--- --xx	0--- --uu
815h	—	—		Unimplemented								—	—
816h	—	—		Unimplemented								—	—
817h	—	—		Unimplemented								—	—
818h	—	—		Unimplemented								—	—
819h	—	—		Unimplemented								—	—
81Ah	NVMADRL			NVMADR<7:0>								0000 0000	0000 0000
81Bh	NVMADRH			—(2)	NVMADR<14:8>							1000 0000	1000 0000
81Ch	NVMDATL			NVMDAT<7:0>								0000 0000	0000 0000
81Dh	NVMDATH			—	—	NVMDAT<13:8>						--00 0000	--00 0000
81Eh	NVMCON1			—	NVMREGS	LWLO	FREE	WRERR	WREN	WR	RD	-000 x000	-000 q000
81Fh	NVMCON2			NVMCON2<7:0>								0000 0000	0000 0000

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

Note 1: Register present on PIC16F18855/75 devices only.

2: Unimplemented, read as '1'.

FIGURE 3-7: ACCESSING THE STACK EXAMPLE 4



3.4.2 OVERFLOW/UNDERFLOW RESET

If the STVREN bit in Configuration Words is programmed to '1', the device will be Reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the first level, setting the appropriate bits (STKOVF or STKUNF, respectively) in the PCON register.

3.5 Indirect Addressing

The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the File Select Registers (FSR). If the FSRn address specifies one of the two INDFn registers, the read will return '0' and the write will not occur (though Status bits may be affected). The FSRn register value is created by the pair FSRnH and FSRnL.

The FSR registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional Data Memory
- Linear Data Memory
- Data EEPROM Memory
- Program Flash Memory

REGISTER 7-15: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4

U-0	U-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	TMR5IF	TMR4IF	TMR3IF	TMR2IF	TMR1IF
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 7-6 **Unimplemented:** Read as '0'

bit 5 **TRM6IF:** Timer6 Interrupt Flag bit
1 = The TMR6 postscaler overflowed, or in 1:1 mode, a TMR6 to PR6 match occurred (must be cleared in software)
0 = No TMR6 event has occurred

bit 4 **TRM5IF:** Timer5 Overflow Interrupt Flag bit
1 = TMR5 overflow occurred (must be cleared in software)
0 = No TMR5 overflow occurred

bit 3 **TRM4IF:** Timer4 Interrupt Flag bit
1 = The TMR4 postscaler overflowed, or in 1:1 mode, a TMR4 to PR4 match occurred (must be cleared in software)
0 = No TMR4 event has occurred

bit 2 **TRM3IF:** Timer3 Overflow Interrupt Flag bit
1 = TMR3 overflow occurred (must be cleared in software)
0 = No TMR3 overflow occurred

bit 1 **TRM2IF:** Timer2 Interrupt Flag bit
1 = The TMR2 postscaler overflowed, or in 1:1 mode, a TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 event has occurred

bit 0 **TRM1IF:** Timer1 Overflow Interrupt Flag bit
1 = TMR1 overflow occurred (must be cleared in software)
0 = No TMR1 overflow occurred

8.2.3.2 Peripheral Usage in Sleep

Some peripherals that can operate in Sleep mode will not operate properly with the Low-Power Sleep mode selected. The Low-Power Sleep mode is intended for use with these peripherals:

- Brown-out Reset (BOR)
- Watchdog Timer (WDT)
- External interrupt pin/interrupt-on-change pins
- Timer1 (with external clock source)

It is the responsibility of the end user to determine what is acceptable for their application when setting the VREGPM settings in order to ensure operation in Sleep.

Note:	The PIC16LF18855/75 does not have a configurable Low-Power Sleep mode. PIC16LF18855/75 is an unregulated device and is always in the lowest power state when in Sleep, with no wake-up time penalty. This device has a lower maximum V_{DD} and I/O voltage than the PIC16F18855/75. See **Section 37.0 "Electrical Specifications"** for more information.

8.2.4 IDLE MODE

When the Idle Enable (IDLEN) bit is clear (IDLEN = 0), the SLEEP instruction will put the device into full Sleep mode (see **Section 8.2 "Sleep Mode"**). When IDLEN is set (IDLEN = 1), the SLEEP instruction will put the device into IDLE mode. In IDLE mode, the CPU and memory operations are halted, but the peripheral clocks continue to run. This mode is similar to DOZE mode, except that in IDLE both the CPU and PFM are shut off.

Note:	Peripherals using FOSC will continue running while in Idle (but not in Sleep). Peripherals using HFINTOSC, LFINTOSC, or SOSC will continue running in both Idle and Sleep.

Note:	If CLKOUT is enabled (CLKOUT = 0, Configuration Word 1), the output will continue operating while in Idle.

8.2.4.1 Idle and Interrupts

IDLE mode ends when an interrupt occurs (even if GIE = 0), but IDLEN is not changed. The device can re-enter IDLE by executing the SLEEP instruction.

If Recover-on-Interrupt is enabled (ROI = 1), the interrupt that brings the device out of Idle also restores full-speed CPU execution when doze is also enabled.

8.2.4.2 Idle and WDT

When in Idle, the WDT Reset is blocked and will instead wake the device. The WDT wake-up is not an interrupt, therefore ROI does not apply.

Note:	The WDT can bring the device out of Idle, in the same way it brings the device out of Sleep. The DOZEN bit is not affected.

11.0 CYCLIC REDUNDANCY CHECK (CRC) MODULE

The Cyclic Redundancy Check (CRC) module provides a software-configurable hardware-implemented CRC checksum generator. This module includes the following features:

- Any standard CRC up to 16 bits can be used
- Configurable Polynomial
- Any seed value up to 16 bits can be used
- Standard and reversed bit order available
- Augmented zeros can be added automatically or by the user
- Memory scanner for fast CRC calculations on program memory user data
- Software loadable data registers for calculating CRC values not from the memory scanner

11.1 CRC Module Overview

The CRC module provides a means for calculating a check value of program memory. The CRC module is coupled with a memory scanner for faster CRC calculations. The memory scanner can automatically provide data to the CRC module. The CRC module can also be operated by directly writing data to SFRs, without using the scanner.

11.2 CRC Functional Overview

The CRC module can be used to detect bit errors in the Flash memory using the built-in memory scanner or through user input RAM memory. The CRC module can accept up to a 16-bit polynomial with up to a 16-bit seed value. A CRC calculated check value (or checksum) will then be generated into the CRCACC<15:0> registers for user storage. The CRC module uses an XOR shift register implementation to perform the polynomial division required for the CRC calculation.

EXAMPLE 11-1: BASIC CRC OPERATION EXAMPLE

CRC-16-ANSI

$x^{16} + x^{15} + x^2 + 1$ (17 bits)

Standard 16-bit representation = 0x8005

```
CRCXORH = 0b10000000
CRCXORL = 0b0000010- (1)
```

Data Sequence:
0x55, 0x66, 0x77, 0x88

```
DLEN = 0b0111
PLEN = 0b1111
```

Data entered into the CRC:

SHIFTM = 0:

```
01010101 01100110 01110111 10001000
```

SHIFTM = 1:

```
10101010 01100110 11101110 00010001
```

Check Value (ACCM = 1):

SHIFTM = 0: 0x32D6

```
CRCACCH = 0b00110010
CRCACCL = 0b11010110
```

SHIFTM = 1: 0x6BA2

```
CRCACCH = 0b01101011
CRCACCL = 0b10100010
```

Note 1: Bit 0 is unimplemented. The LSb of any CRC polynomial is always '1' and will always be treated as a '1' by the CRC for calculating the CRC check value. This bit will be read in software as a '0'.

11.3 CRC Polynomial Implementation

Any standard polynomial up to 17 bits can be used. The PLEN<3:0> bits are used to specify how long the polynomial used will be. For an x^n polynomial, PLEN = n-2. In an n-bit polynomial the x^n bit and the LSb will be used as a '1' in the CRC calculation because the MSb and LSb must always be a '1' for a CRC polynomial. For example, if using CRC-16-ANSI, the polynomial will look like 0x8005. This will be implemented into the CRCXOR<15:1> registers, as shown in Example 11-1.

11.10.1 BURST MODE

When MODE = 01, the scanner is in Burst mode. In Burst mode, CPU operation is stalled beginning with the operation after the one that sets the SCANGO bit, and the scan begins, using the instruction clock to execute. The CPU is held until the scan stops. Note that because the CPU is not executing instructions, the SCANGO bit cannot be cleared in software, so the CPU will remain stalled until one of the hardware end-conditions occurs. Burst mode has the highest throughput for the scanner, but has the cost of stalling other execution while it occurs.

11.10.2 CONCURRENT MODE

When MODE = 00, the scanner is in Concurrent mode. Concurrent mode, like Burst mode, stalls the CPU while performing accesses of memory. However, while Burst mode stalls until all accesses are complete, Concurrent mode allows the CPU to execute in between access cycles.

11.10.3 TRIGGERED MODE

When MODE = 11, the scanner is in Triggered mode. Triggered mode behaves identically to Concurrent mode, except instead of beginning the scan immediately upon the SCANGO bit being set, it waits for a rising edge from a separate trigger clock, the source of which is determined by the SCANTRIG register.

11.10.4 PEEK MODE

When MODE = 10, the scanner is in Peek mode. Peek mode waits for an instruction cycle in which the CPU does not need to access the NVM (such as a branch instruction) and uses that cycle to do its own NVM access. This results in the lowest throughput for the NVM access (and can take a much longer time to complete a scan than the other modes), but does so without any impact on execution times, unlike the other modes.

TABLE 11-1: SUMMARY OF SCANNER MODES

MODE<1:0>		Description		
		First Scan Access	CPU Operation	
11	Triggered	As soon as possible following a trigger	Stalled during NVM access	CPU resumes execution following each access
10	Peek	At the first dead cycle	Timing is unaffected	CPU continues execution following each access
01	Burst	As soon as possible	Stalled during NVM access	CPU suspended until scan completes
00	Concurrent			CPU resumes execution following each access

11.10.5 INTERRUPT INTERACTION

The INTM bit of the SCANCON0 register controls the scanner's response to interrupts depending on which mode the NVM scanner is in, as described in Table 11-2.

TABLE 11-2: SCAN INTERRUPT MODES

INTM	MODE<1:0>	
	MODE == Burst	MODE != Burst
1	Interrupt overrides SCANGO to pause the burst and the interrupt handler executes at full speed; Scanner Burst resumes when interrupt completes.	Scanner suspended during interrupt response; interrupt executes at full speed and scan resumes when the interrupt is complete.
0	Interrupts do not override SCANGO, and the scan (burst) operation will continue; interrupt response will be delayed until scan completes (latency will be increased).	Scanner accesses NVM during interrupt response. If MODE != Peak the interrupt handler execution speed will be affected.

REGISTER 12-38: SLRCOND: PORTD SLEW RATE CONTROL REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
SLRD7	SLRD6	SLRD5	SLRD4	SLRD3	SLRD2	SLRD1	SLRD0
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 **SLRD<7:0>:** PORTD Slew Rate Enable bits
For RD<7:0> pins, respectively
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

REGISTER 12-39: INLVLD: PORTD INPUT LEVEL CONTROL REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
INLVLD7	INLVLD6	INLVLD5	INLVLD4	INLVLD3	INLVLD2	INLVLD1	INLVLD0
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 **INLVLD<7:0>:** PORTD Input Level Select bits
For RD<7:0> pins, respectively
1 = ST input used for PORT reads and interrupt-on-change
0 = TTL input used for PORT reads and interrupt-on-change

18.8 Comparator Response Time

The comparator output is indeterminate for a period of time after the change of an input source or the selection of a new reference voltage. This period is referred to as the response time. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response time to a comparator input change. See the Comparator and Voltage Reference Specifications in Table 37-14 for more details.

18.9 Analog Input Connection Considerations

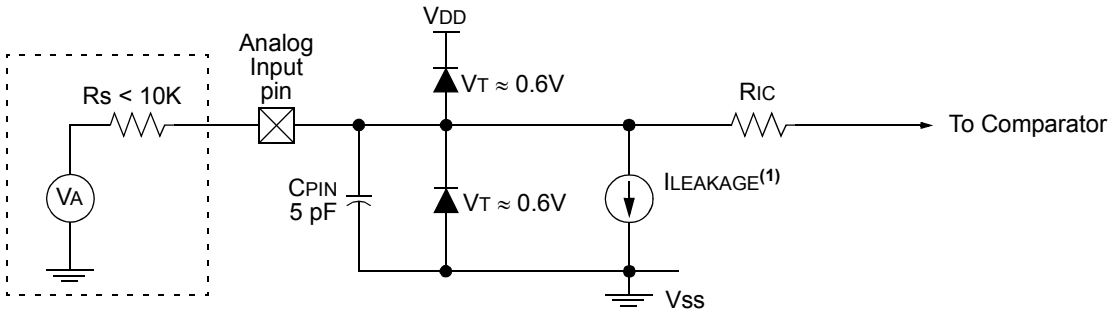
A simplified circuit for an analog input is shown in Figure 18-3. Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to V_{DD} and V_{SS}. The analog input, therefore, must be between V_{SS} and V_{DD}. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up may occur.

A maximum source impedance of 10 kΩ is recommended for the analog sources. Also, any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current to minimize inaccuracies introduced.

Note 1: When reading a PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert as an analog input, according to the input specification.

2: Analog levels on any pin defined as a digital input, may cause the input buffer to consume more current than is specified.

FIGURE 18-3: ANALOG INPUT MODEL



Legend: C_{PIN} = Input Capacitance
$I_{LEAKAGE}$ = Leakage Current at the pin due to various junctions
R_{IC} = Interconnect Resistance
R_S = Source Impedance
V_A = Analog Voltage
V_T = Threshold Voltage

Note 1: See I/O Ports in Table 37-4.

22.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) module provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 32 input signals and, through the use of configurable gates, reduces the 32 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

The output can be directed internally to peripherals and to an output pin.

The CLC modules available are shown in Table 22-1.

TABLE 22-1: AVAILABLE CLC MODULES

Device	CLC1	CLC2	CLC3	CLC4
PIC16(L)F18855/75	●	●	●	●

Note: The CLC1, CLC2, CLC3 and CLC4 are four separate module instances of the same CLC module design. Throughout this section, the lower case 'x' in register and bit names is a generic reference to the CLC number (which should be substituted with 1, 2, 3, or 4 during code development). For example, the control register is generically described in this chapter as CLCxCON, but the actual device registers are CLC1CON, CLC2CON, CLC3CON and CLC4CON. Similarly, the LCxEN bit represents the LC1EN, LC2EN, LC3EN and LC4EN bits.

Refer to Figure 22-1 for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset
  - Clocked J-K with Reset

22.2 CLCx Interrupts

An interrupt will be generated upon a change in the output value of the CLCx when the appropriate interrupt enables are set. A rising edge detector and a falling edge detector are present in each CLC for this purpose.

The CLCxIF bit of the associated PIR5 register will be set when either edge detector is triggered and its associated enable bit is set. The LCxINTP enables rising edge interrupts and the LCxINTN bit enables falling edge interrupts. Both are located in the CLCxCON register.

To fully enable the interrupt, set the following bits:

- CLCxIE bit of the PIE5 register
- LCxINTP bit of the CLCxCON register (for a rising edge detection)
- LCxINTN bit of the CLCxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The CLCxIF bit of the PIR5 register, must be cleared in software as part of the interrupt service. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

22.3 Output Mirror Copies

Mirror copies of all LCxCON output bits are contained in the CLCxDATA register. Reading this register reads the outputs of all CLCs simultaneously. This prevents any reading skew introduced by testing or reading the LCxOUT bits in the individual CLCxCON registers.

22.4 Effects of a Reset

The CLCxCON register is cleared to zero as the result of a Reset. All other selection and gating values remain unchanged.

22.5 Operation During Sleep

The CLC module operates independently from the system clock and will continue to run during Sleep, provided that the input sources selected remain active.

The HFINTOSC remains active during Sleep when the CLC module is enabled and the HFINTOSC is selected as an input source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and as a CLC input source, when the CLC is enabled, the CPU will go idle during Sleep, but the CLC will continue to operate and the HFINTOSC will remain active.

This will have a direct effect on the Sleep mode current.

22.6 CLCx Setup Steps

The following steps should be followed when setting up the CLCx:

- Disable CLCx by clearing the LCxEN bit.
- Select desired inputs using CLCxSEL0 through CLCxSEL3 registers (See Table 22-2).
- Clear any associated ANSEL bits.
- Set all TRIS bits associated with inputs.
- Clear all TRIS bits associated with outputs.
- Enable the chosen inputs through the four gates using CLCxGLS0, CLCxGLS1, CLCxGLS2, and CLCxGLS3 registers.
- Select the gate output polarities with the LCxGyPOL bits of the CLCxPOL register.
- Select the desired logic function with the LCxMODE<2:0> bits of the CLCxCON register.
- Select the desired polarity of the logic output with the LCxPOL bit of the CLCxPOL register. (This step may be combined with the previous gate output polarity step).
- If driving a device pin, set the desired pin PPS control register and also clear the TRIS bit corresponding to that output.
- If interrupts are desired, configure the following bits:
  - Set the LCxINTP bit in the CLCxCON register for rising event.
  - Set the LCxINTN bit in the CLCxCON register for falling event.
  - Set the CLCxIE bit of the PIE5 register.
  - Set the GIE and PEIE bits of the INTCON register.
- Enable the CLCx by setting the LCxEN bit of the CLCxCON register.

REGISTER 23-24: ADSTPTH: ADC THRESHOLD SETPOINT REGISTER HIGH

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
ADSTPT<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 **ADSTPT<15:8>**: ADC Threshold Setpoint MSB. Most Significant Byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ADERR, see Register 21-1 for more details.

\

REGISTER 23-25: ADSTPTL: ADC THRESHOLD SETPOINT REGISTER LOW

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ADSTPT<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 **ADSTPT<7:0>**: ADC Threshold Setpoint LSB. Least Significant Byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ADERR, see Register 21-1 for more details.

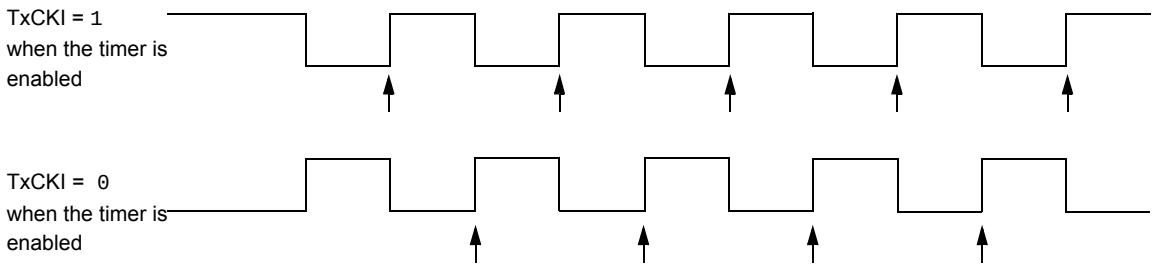
REGISTER 23-26: ADERRH: ADC CALCULATION ERROR REGISTER HIGH

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ADERR<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 **ADERR<15:8>**: ADC Calculation Error MSB. Most Significant Byte of ADC Calculation Error. Calculation is determined by ADCALC bits of ADCON3, see Register 21-1 for more details.

FIGURE 28-2: TIMER1 INCREMENTING EDGE



Note 1: Arrows indicate counter increments.

2: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

FIGURE 28-3: TIMER1 GATE ENABLE MODE

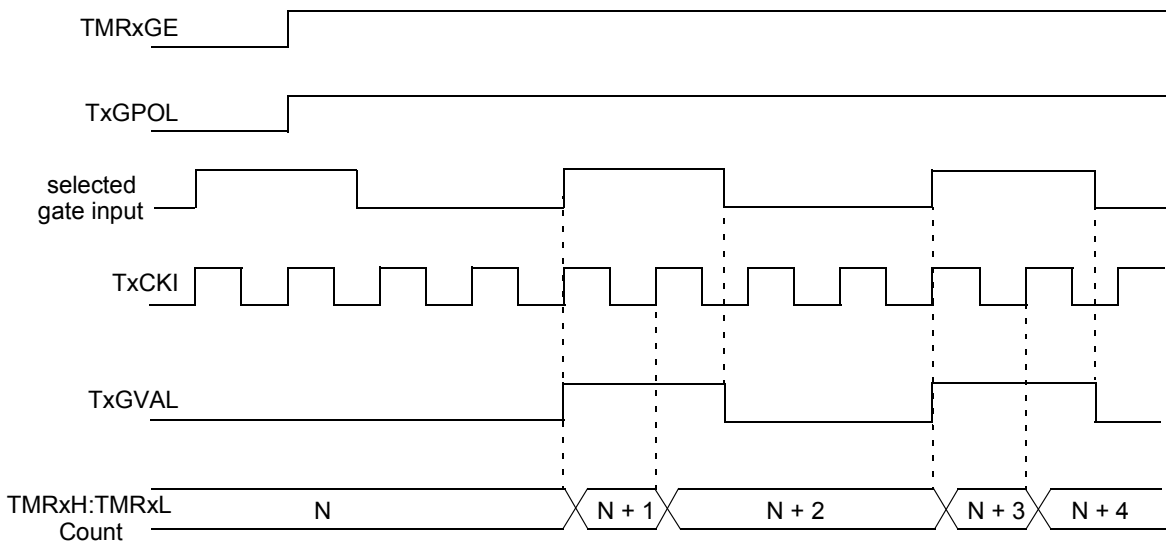


FIGURE 31-7: SPI DAISY-CHAIN CONNECTION

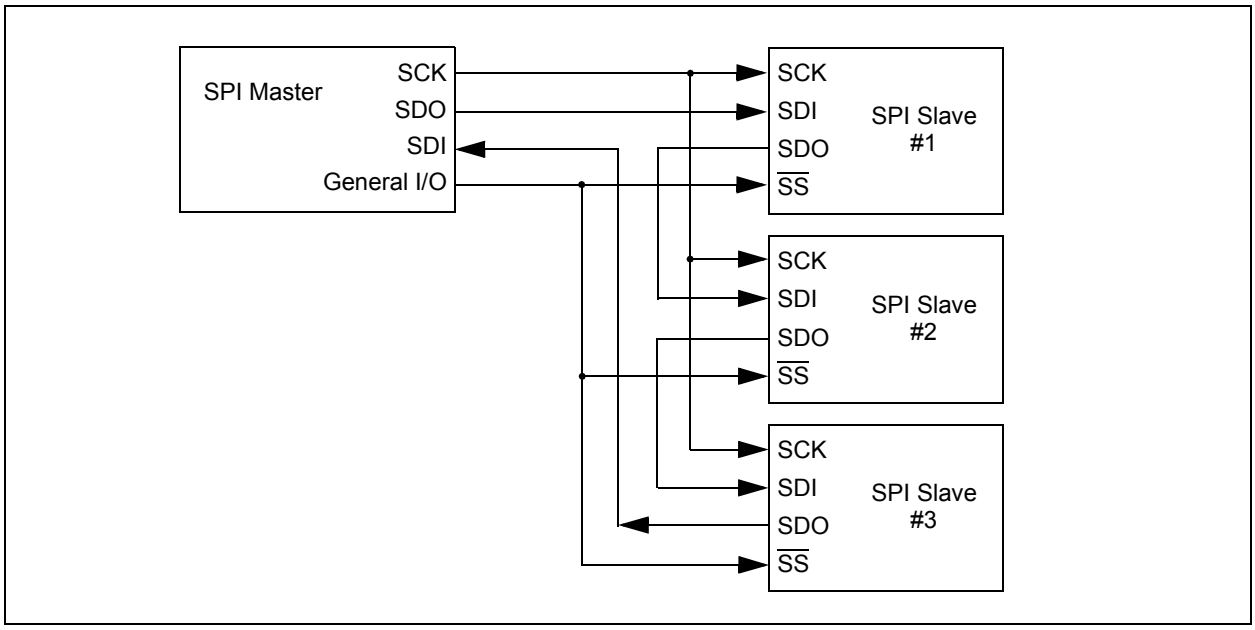
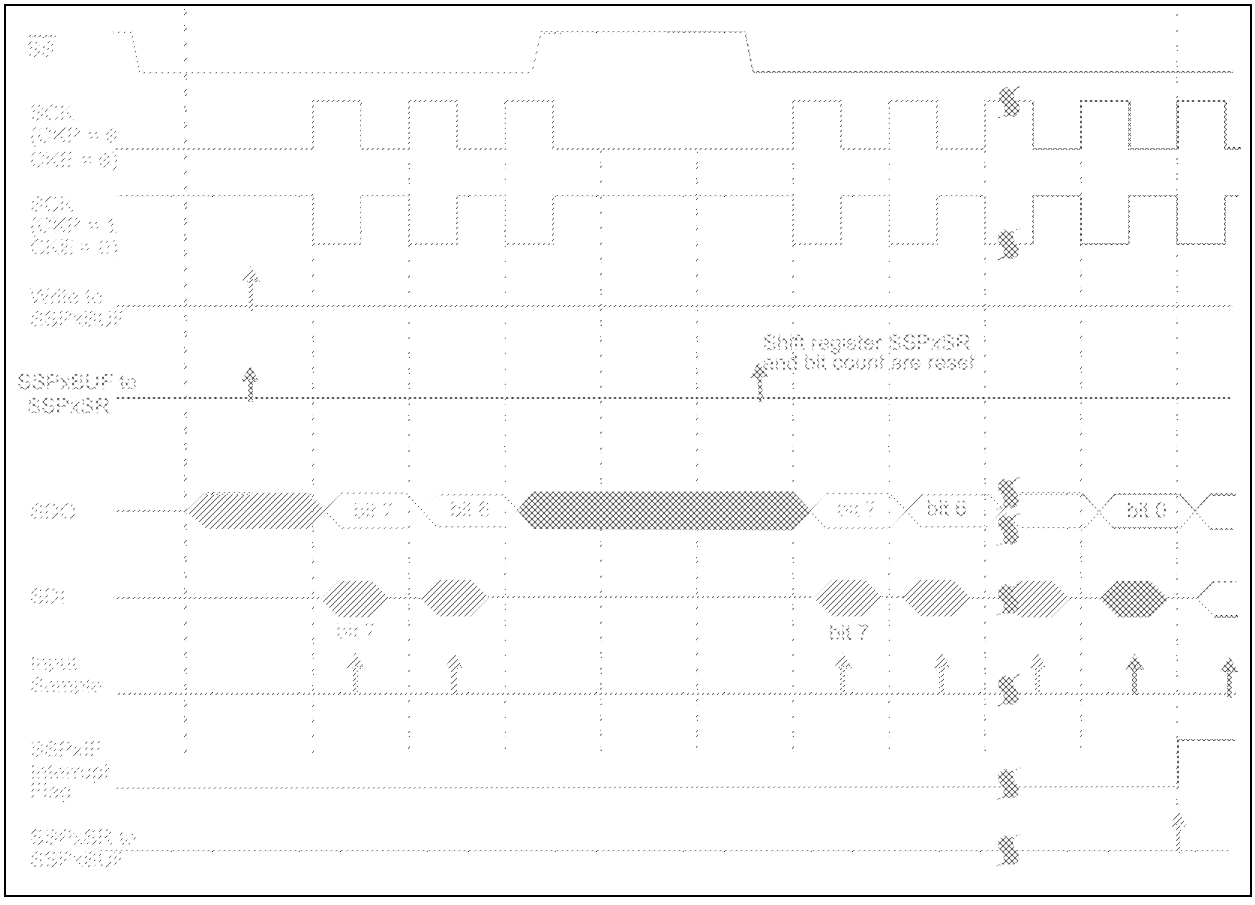


FIGURE 31-8: SLAVE SELECT SYNCHRONOUS WAVEFORM



35.0 IN-CIRCUIT SERIAL PROGRAMMING™ (ICSP™)

ICSP™ programming allows customers to manufacture circuit boards with unprogrammed devices. Programming can be done after the assembly process, allowing the device to be programmed with the most recent firmware or a custom firmware. Five pins are needed for ICSP™ programming:

- ICSPCLK
- ICSPDAT
- $\overline{MCLR}/V_{PP}$
- V_{DD}
- V_{SS}

In Program/Verify mode the program memory, User IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the **"***PIC16(L)F1783XX Memory Programming Specification***"** (DS400001738).

35.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the ICSPCLK and ICSPDAT pins low then raising the voltage on $\overline{MCLR}/V_{PP}$ to V_{IHH}.

35.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using V_{DD} only, without high voltage. When the LVP bit of Configuration Words is set to '1', the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to '0'.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. $\overline{MCLR}$ is brought to V_{IL}.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{MCLR}$ must be held at V_{IL} for as long as Program/Verify mode is to be maintained.

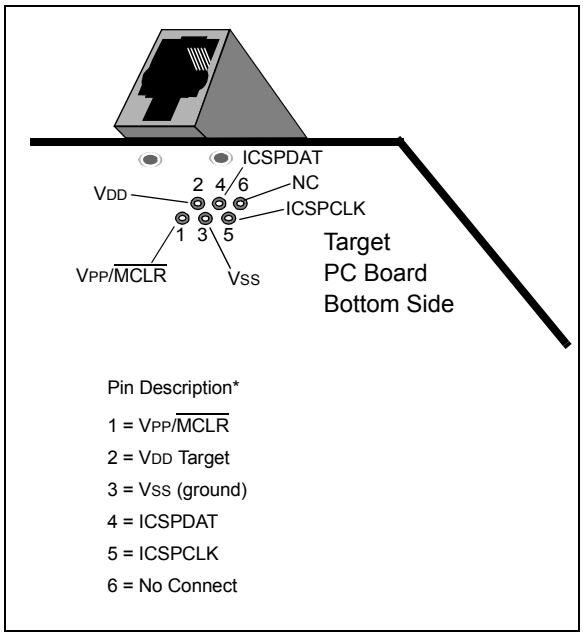
If low-voltage programming is enabled (LVP = 1), the $\overline{MCLR}$ Reset function is automatically enabled and cannot be disabled. See **Section 5.4 "MCLR"** for more information.

The LVP bit can only be reprogrammed to '0' by using the High-Voltage Programming mode.

35.3 Common Programming Interfaces

Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6-connector) configuration. See Figure 35-1.

FIGURE 35-1: ICD RJ-11 STYLE CONNECTOR INTERFACE



Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to Figure 35-2.

For additional interface recommendations, refer to your specific device programmer manual prior to PCB design.

It is recommended that isolation devices be used to separate the programming pins from other circuitry. The type of isolation is highly dependent on the specific application and may include devices such as resistors, diodes, or even jumpers. See Figure 35-3 for more information.

DECFSZ **Decrement f, Skip if 0**

Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2-cycle instruction.

GOTO **Unconditional Branch**

Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 2047$
Operation:	k → PC<10:0> PCLATH<6:3> → PC<14:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction.

INCF **Increment f**

Syntax:	[*label*] INCF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

INCFSZ **Increment f, Skip if 0**

Syntax:	[*label*] INCFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2-cycle instruction.

IORLW **Inclusive OR literal with W**

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

IORWF **Inclusive OR W with f**

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
Description:	Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

FIGURE 37-12: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

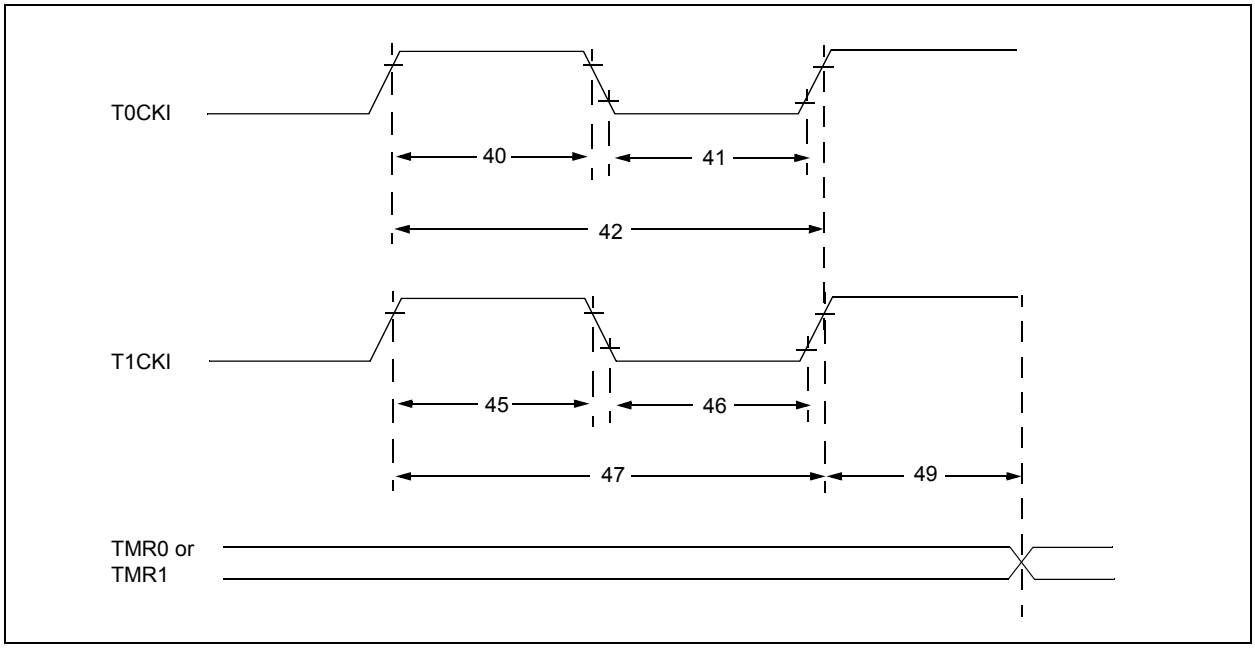


TABLE 37-18: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Standard Operating Conditions (unless otherwise stated) Operating Temperature -40°C ≤ TA ≤ +125°C									
Param. No.	**Sym.**	**Characteristic**		**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
40*	TT0H	T0CKI High Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	TT0L	T0CKI Low Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	TT0P	T0CKI Period		Greater of: 20 or TCY + 40 / N	—	—	ns	N = prescale value
45*	TT1H	T1CKI High Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	TT1L	T1CKI Low Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
47*	TT1P	T1CKI Input Period	Synchronous	Greater of: 30 or TCY + 40 / N	—	—	ns	N = prescale value
			Asynchronous	60	—	—	ns	
48	FT1	Secondary Oscillator Input Frequency Range (oscillator enabled by setting bit T1OSCEN)		32.4	32.768	33.1	kHz	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		2 TOSC	—	7 TOSC	—	Timers in Sync mode

* These parameters are characterized but not tested.

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