Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

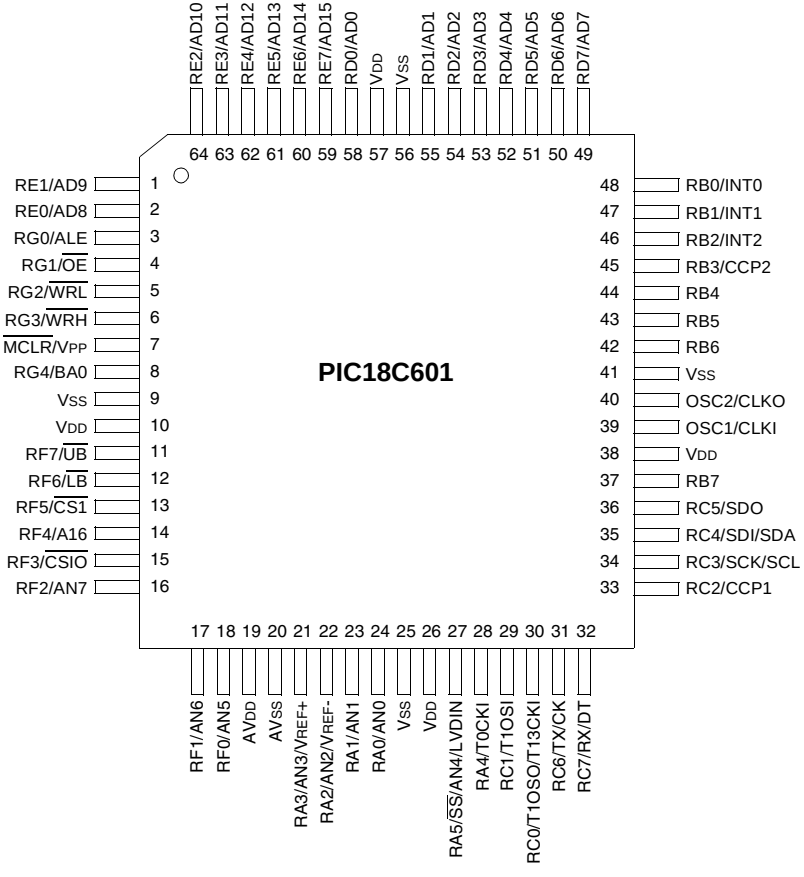
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	25MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	37
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lc801t-i-pt

Pin Diagrams

64-Pin TQFP



TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

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

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

This document contains device specific information for the following two devices:

1. PIC18C601
2. PIC18C801

The PIC18C601 is available in 64-pin TQFP and 68-pin PLCC packages. The PIC18C801 is available in 80-pin TQFP and 84-pin PLCC packages.

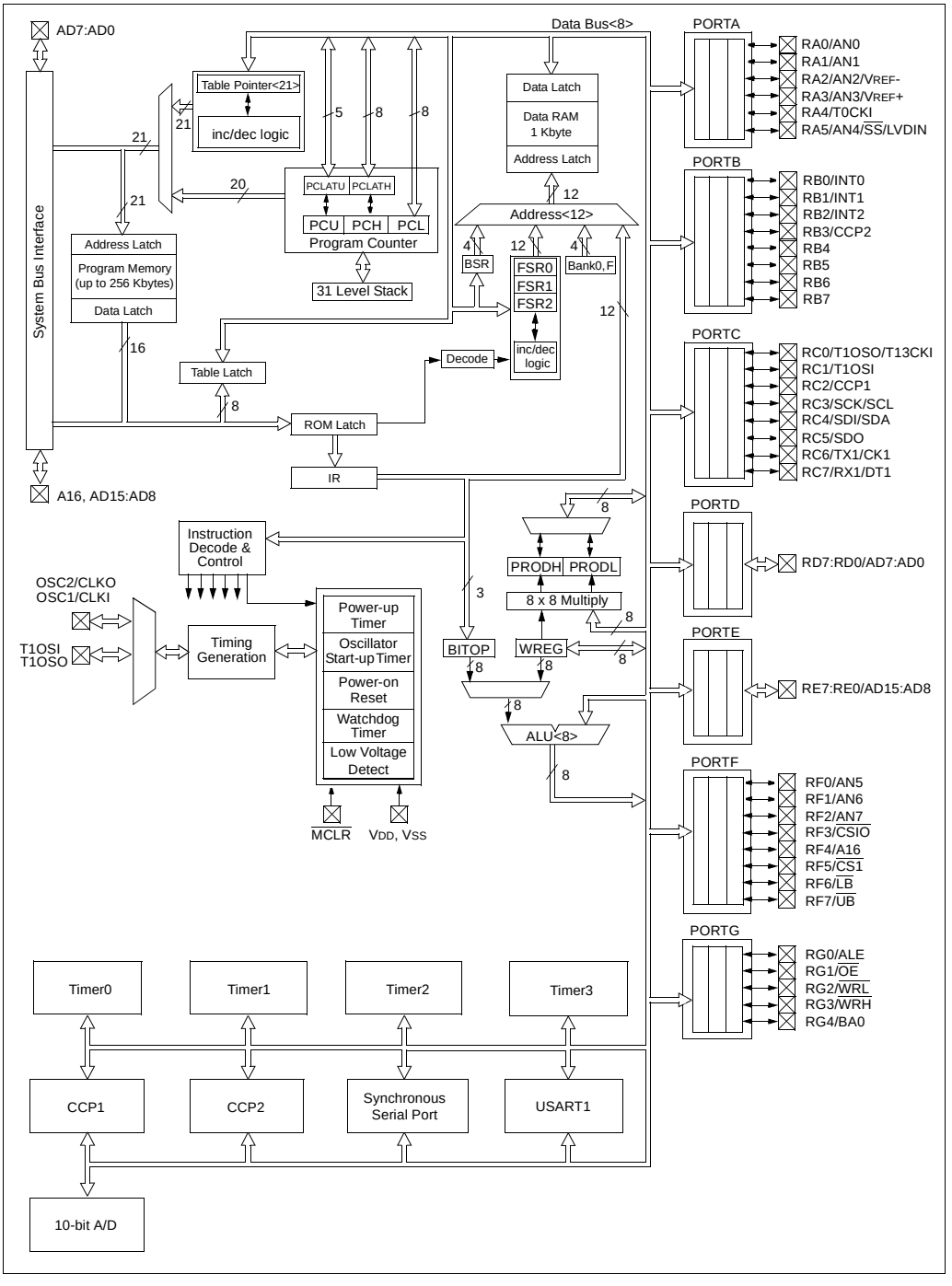
An overview of features is shown in Table 1-1.

Device block diagrams are provided in Figure 1-1 for the 64/68-pin configuration, and Figure 1-2 for the 80/84-pin configuration. The pinouts for both packages are listed in Table 1-2.

TABLE 1-1: DEVICE FEATURES

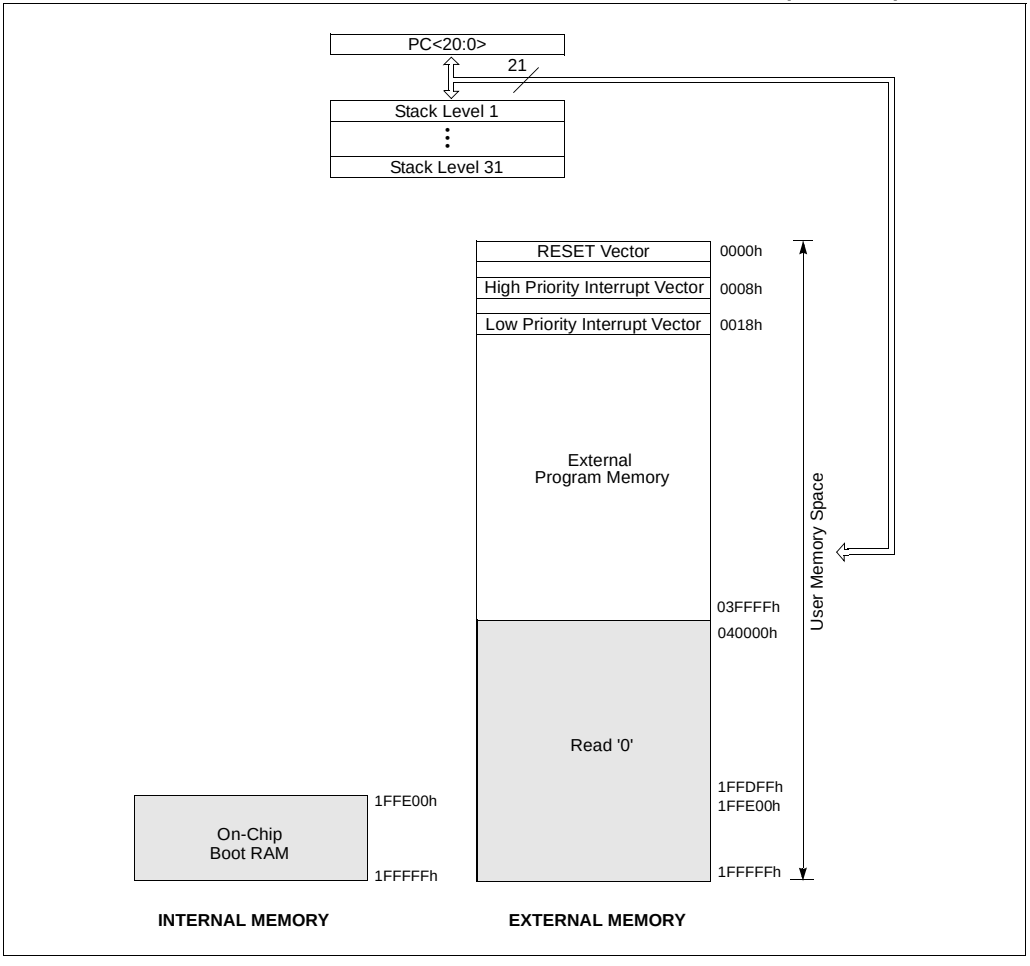
Features		PIC18C601	PIC18C801
Operating Frequency		DC - 25 MHz	DC - 25 MHz
External Program Memory	Bytes	256K	2M
	Max. # of Single Word Instructions	128K	1M
Data Memory (Bytes)		1536	1536
Interrupt Sources		15	15
I/O Ports		Ports A - G	Ports A - H, J
Timers		4	4
Capture/Compare/PWM modules		2	2
Serial Communications		MSSP, Addressable USART	MSSP, Addressable USART
10-bit Analog-to-Digital Module		8 input channels	12 input channels
RESETS (and Delays)		POR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)
Programmable Low Voltage Detect		Yes	Yes
8-bit External Memory Interface		Yes	Yes
8-bit De-multiplexed External Memory Interface		No	Yes
16-bit External Memory Interfaces		Yes	Yes
On-chip Chip Select Signals		$\overline{CS1}$	$\overline{CS1}$, $\overline{CS2}$
On-chip I/O Chip Select Signal		Yes	Yes
Instruction Set		75 Instructions	75 Instructions
Packages		64-pin TQFP 68-pin PLCC	80-pin TQFP 84-pin PLCC

FIGURE 1-1: PIC18C601 BLOCK DIAGRAM



NOTES:

FIGURE 4-3: PROGRAM MEMORY MAP AND STACK FOR PIC18C601 (PGRM = 1)



4.1.2 BOOT LOADER

When configured as Program Memory, Boot RAM can be used as a temporary "Boot Loader" for programming purposes. If an external memory device is used as program memory, any updates performed by the user program will have to be performed in the "Boot RAM", because the user program cannot program and fetch from external memory, simultaneously.

A typical boot loader execution and external memory programming sequence would be as follows:

- The boot loader program is transferred from the external program memory to the last 2 banks of data RAM by TBLRD and MOVWF instructions.
- Once the "boot loader" program is loaded into internal memory and verified, open combination lock and set PGRM bit to configure the data RAM into program RAM.
- Jump to beginning of Boot code in Boot RAM. Program execution begins in Boot RAM to begin programming the external memory. System bus changes to an inactive state.
- Boot loader program performs the necessary external TBLWT and TBLWRD instructions to perform programming functions.
- When the boot loader program is finished programming external memory, jump to known valid external program memory location and clear PGRM bit in MEMCON register to set Boot RAM as data memory, or reset the part.

4.2 Return Address Stack

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC (Program Counter) is pushed onto the stack when a PUSH, CALL or RCALL instruction is executed, or an interrupt is acknowledged. The PC value is pulled off the stack on a RETURN, RETLW or a RETFIE instruction. PCLATU and PCLATH are not affected by any of the return instructions.

The stack operates as a 31-word by 21-bit stack memory and a five-bit stack pointer, with the stack pointer initialized to 00000b after all RESETS. There is no RAM associated with stack pointer 00000b. This is only a RESET value. During a CALL type instruction, causing a push onto the stack, the stack pointer is first incremented and the RAM location pointed to by the stack pointer is written with the contents of the PC. During a RETURN type instruction, causing a pop from the stack, the contents of the RAM location indicated by the STKPTR is transferred to the PC and then the stack pointer is decremented.

The stack space is not part of either program or data space. The stack pointer is readable and writable, and the data on the top of the stack is readable and writable through SFR registers. Status bits STKOVF and STKUNF in STKPTR register, indicate whether stack over/underflow has occurred or not.

4.2.1 TOP-OF-STACK ACCESS

The top of the stack is readable and writable. Three register locations, TOSU, TOSH and TOSL, allow access to the contents of the stack location indicated by the STKPTR register. This allows users to implement a software stack, if necessary. After a CALL, RCALL or interrupt, the software can read the pushed value by reading the TOSU, TOSH and TOSL registers. These values can be placed on a user defined software stack. At return time, the software can replace the TOSU, TOSH and TOSL and do a return.

The user should disable the global interrupt enable bits during this time to prevent inadvertent stack operations.

4.2.2 RETURN STACK POINTER (STKPTR)

The STKPTR register contains the stack pointer value, the STKFUL (stack full) status bit, and the STKUNF (stack underflow) status bits. Register 4-1 shows the STKPTR register. The value of the stack pointer can be 0 through 31. The stack pointer increments when values are pushed onto the stack and decrements when values are popped off the stack. At RESET, the stack pointer value will be 0. The user may read and write the stack pointer value. This feature can be used by a Real Time Operating System for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit can only be cleared in software or by a POR. Any subsequent push operation that causes stack overflow will be ignored.

The action that takes place when the stack becomes full, depends on the state of STVREN (stack overflow RESET enable) configuration bit in CONFIG4L register. Refer to Section 4.2.4 for more information. If STVREN is set (default), stack over/underflow will set the STKFUL bit, and reset the device. The STKFUL bit will remain set and the stack pointer will be set to 0.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the stack pointer will increment to 31. All subsequent push attempts will be ignored and STKPTR remains at 31.

When the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC and sets the STKUNF bit, while the stack pointer remains at 0. The STKUNF bit will remain set until cleared in software, or a POR occurs.

Note:	Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the RESET vector, where the stack conditions can be verified and appropriate actions can be taken.

5.4 Chip Selects

Chip select signals are used to select regions of external memory and I/O devices for access. The PIC18C801 has three chip selects and all are programmable. The chip select signals are $\overline{CS1}$, $\overline{CS2}$ and $\overline{CSIO}$. $\overline{CS1}$ and $\overline{CS2}$ are general purpose chip selects that are used to enable large portions of program memory. $\overline{CSIO}$ is used to enable external I/O expansion. The PIC18C601uses two of these programmable chip selects: $\overline{CS1}$ and $\overline{CSIO}$.

Two SFRs are used to control the chip select signals. These are CSEL2 and CSELIO (see Register 5-2 and Register 5-3). A chip select signal is asserted low when the CPU makes an access to a dedicated range of addresses specified in the chip select registers, CSEL2 and CSELIO. The 8-bit value found in either of these registers is decoded as one of 256, 8K banks of program memory. If both chip select registers are 00h, all of the chip select signals are disabled and their corresponding pins are configured as I/O. Since the last 512 bytes of program memory are dedicated to internal program RAM, the chip select signals will not activate if the program memory address falls in this range.

REGISTER 5-2: CSEL2 REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CSL7	CSL6	CSL5	CSL4	CSL3	CSL2	CSL1	CSL0
bit 7							bit 0

bit 7-0 **CSL<7:0>**: Chip Select 2 Address Decode bits

XXh = All eight bits are compared to the Most Significant bits PC<20:13> of the program counter. If PC<20:13> ≥ CSL<7:0> register, then the $\overline{CS2}$ signal is low. If PC<20:13> < CSL<7:0>, $\overline{CS2}$ is high.

00h = $\overline{CS2}$ is inactive

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 5-3: CSELIO REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CSIO7	CSIO6	CSIO5	CSIO4	CSIO3	CSIO2	CSIO1	CSIO0
bit7							bit0

bit 7-0 **CSIO<7:0>**: Chip Select IO Address Decode bits

XXh = All eight bits are compared to the Most Significant bits PC<20:13> of the program counter. If PC<20:13> = CSIO<7:0>, then the $\overline{CSIO}$ signal is low. If not, $\overline{CSIO}$ is high.

00h = $\overline{CSIO}$ is inactive

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

8.1.6 INT INTERRUPTS

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge triggered: either rising, if the corresponding INTEDGx bit is set in the INTCON2 register, or falling, if the INTEDGx bit is clear. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit INTxIF is set. This interrupt can be disabled by clearing the corresponding enable bit INTxIE. Flag bit INTxIF must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt. All external interrupts (INT0, INT1 and INT2) can wake-up the processor from SLEEP, if bit INTxIE was set prior to going into SLEEP. If the global interrupt enable bit GIE is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits INT1IP (INTCON3 register) and INT2IP (INTCON3 register). There is no priority bit associated with INT0; it is always a high priority interrupt source.

8.1.7 TMR0 INTERRUPT

In 8-bit mode (which is the default), an overflow (0FFh $\rightarrow$ 00h) in the TMR0 register will set flag bit TMR0IF. In 16-bit mode, an overflow (0FFFFh $\rightarrow$ 0000h) in the TMR0H:TMR0L registers will set flag bit TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit TMR0IE (INTCON register). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit TMR0IP (INTCON2 register). See Section 10.0 for further details on the Timer0 module.

8.1.8 PORTB INTERRUPT-ON-CHANGE

An input change on PORTB<7:4> sets flag bit RBIF (INTCON register). The interrupt can be enabled/disabled by setting/clearing enable bit RBIE (INTCON register). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit RBIP (INTCON2 register).

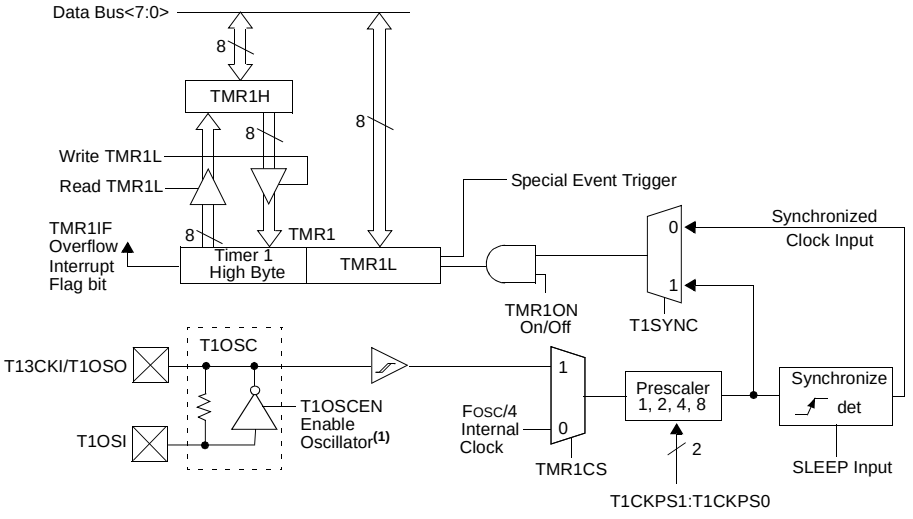
8.2 Context Saving During Interrupts

During an interrupt, the return PC value is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (See Section 4.3), the user may need to save the WREG, STATUS and BSR registers in software. Depending on the user's application, other registers may also need to be saved. Example 8-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 8-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF    W_TEMP                    ; W_TEMP is in Low Access bank
MOVFF    STATUS, STATUS_TEMP       ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP             ; BSR located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR             ; Restore BSR
MOVF     W_TEMP, W                 ; Restore WREG
MOVFF    STATUS_TEMP, STATUS       ; Restore STATUS
```

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



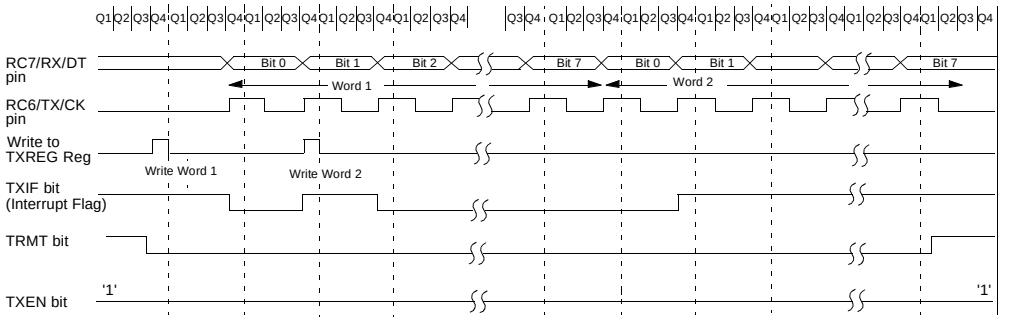
Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This reduces power drain.

TABLE 14-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other RESETS
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	-000 0000	-000 0000
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu
CCPR1L	Capture/Compare/PWM Register1 (LSB)								xxxx xxxx	uuuu uuuu
CCPR1H	Capture/Compare/PWM Register1 (MSB)								xxxx xxxx	uuuu uuuu
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000
CCPR2L	Capture/Compare/PWM Register2 (LSB)								xxxx xxxx	uuuu uuuu
CCPR2H	Capture/Compare/PWM Register2 (MSB)								xxxx xxxx	uuuu uuuu
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	--00 0000
PIR2	—	—	—	—	BCLIF	LVDIF	TMR3IF	CCP2IF	---- 0000	---- 0000
PIE2	—	—	—	—	BCLIE	LVDIE	TMR3IE	CCP2IE	---- 0000	---- 0000
IPR2	—	—	—	—	BCLIP	LVDIP	TMR3IP	CCP2IP	---- 0000	---- 0000
TMR3L	Holding register for the Least Significant Byte of the 16-bit TMR3 register								xxxx xxxx	uuuu uuuu
TMR3H	Holding register for the Most Significant Byte of the 16-bit TMR3 register								xxxx xxxx	uuuu uuuu
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	uuuu uuuu

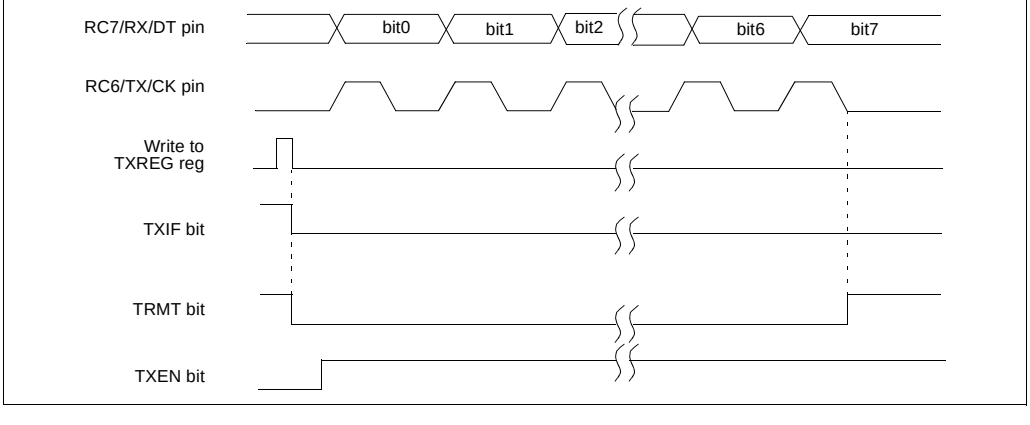
Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by Capture and Timer1.

FIGURE 16-6: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode; SPBRG = '0'; continuous transmission of two 8-bit words.

FIGURE 16-7: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



18.2 Operation

Depending on the power source for the device voltage, the voltage normally decreases relatively slowly. This means that the LVD module does not need to be constantly operating. To decrease current consumption, the LVD circuitry only needs to be enabled for short periods, where the voltage is checked. After doing the check, the LVD module may be disabled.

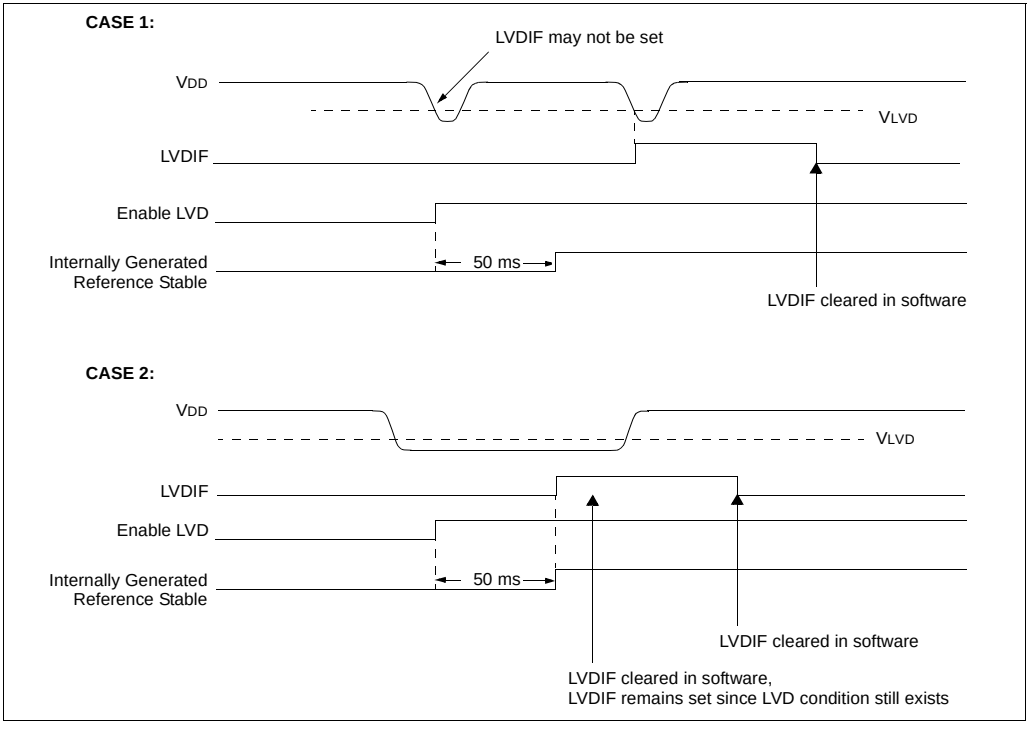
Each time that the LVD module is enabled, the circuitry requires some time to stabilize. After the circuitry has stabilized, all status flags may be cleared. The module will then indicate the proper state of the system.

The following steps are needed to setup the LVD module:

1. Write the value to the LVDL3:LVDL0 bits (LVDCON register), which selects the desired LVD trip point.
2. Ensure that LVD interrupts are disabled (the LVDIE bit is cleared or the GIE bit is cleared).
3. Enable the LVD module (set the LVDEN bit in the LVDCON register).
4. Wait for the LVD module to stabilize (the IRVST bit to become set).
5. Clear the LVD interrupt flag, which may have falsely become set, until the LVD module has stabilized (clear the LVDIF bit).
6. Enable the LVD interrupt (set the LVDIE and the GIE bits).

Figure 18-3 shows typical waveforms that the LVD module may be used to detect.

FIGURE 18-3: LOW VOLTAGE DETECT WAVEFORMS



19.2 Watchdog Timer (WDT)

The Watchdog Timer is a free running on-chip RC oscillator, which does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKI pin. That means that the WDT will run, even if the clock on the OSC1/CLKI and OSC2/CLKO pins of the device has been stopped; for example, by execution of a SLEEP instruction.

During normal operation, a WDT time-out generates a device RESET (Watchdog Timer Reset). If the device is in SLEEP mode, a WDT time-out causes the device to wake-up and continue with normal operation (Watchdog Timer Wake-up). The $\overline{TO}$ bit in the RCON register will be cleared upon a WDT time-out.

By default, the Watchdog Timer is disabled by configuration to allow software control over Watchdog Timer operation. If the WDT is enabled by configuration, software execution may not disable this function. When the Watchdog Timer is disabled by configuration, the SWDTEN bit in the WDTCON register enables/disables the operation of the WDT.

The WDT time-out period values may be found in the Electrical Specifications section under parameter #31. Values for the WDT postscaler may be assigned by using configuration bits WDPS<3:1> in CONFIG2H register. If the Watchdog Timer is disabled by configuration, values for the WDT postscaler may be assigned using the SWDPS bits in the WDTCON register.

Note 1:	The CLRWDT and SLEEP instructions clear the WDT and the postscaler, if assigned to the WDT, and prevent it from timing out and generating a device RESET condition.
2:	When a CLRWDT instruction is executed and the prescaler is assigned to the WDT, the prescaler count will be cleared, but the prescaler assignment is not changed.

19.2.1 CONTROL REGISTER

Register 19-5 shows the WDTCON register. This is a readable and writable register. It contains control bits to control the Watchdog Timer from user software. If the Watchdog Timer is enabled by configuration, this register setting is ignored.

REGISTER 19-5: WDTCON REGISTER

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	SWDPS2	SWDPS1	SWDPS0	SWDTEN
bit 7							bit 0

bit 7-4 **Unimplemented**: Read as '0'

bit 3-1 **SWDPS2:SWDPS0:** Software Watchdog Timer Postscale Select bits
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit
1 = Watchdog Timer is on
0 = Watchdog Timer is turned off if it is not disabled

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

RCALL **Relative Call**

Syntax: [*label*] RCALL n

Operands: $-1024 \le n \le 1023$

Operation: $(PC) + 2 \rightarrow TOS$,
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1101	1nnn	nnnn	nnnn

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC+2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' Push PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE RCALL` Jump

Before Instruction
PC = Address `(HERE)`

After Instruction
PC = Address `(Jump)`
TOS = Address `(HERE+2)`

RESET **Reset**

Syntax: [*label*] RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{MCLR}$ Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a $\overline{MCLR}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: `RESET`

After Instruction
Registers = Reset Value
Flags* = Reset Value

21.13 PICDEM 3 Low Cost PIC16CXXX Demonstration Board

The PICDEM 3 demonstration board is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with an LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 3 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer with an adapter socket, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 3 demonstration board to test firmware. A prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM 3 demonstration board is a LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM 3 demonstration board provides an additional RS-232 interface and Windows software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals.

21.14 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. All necessary hardware is included to run basic demo programs, which are supplied on a 3.5-inch disk. A programmed sample is included and the user may erase it and program it with the other sample programs using the PRO MATE II device programmer, or the PICSTART Plus development programmer, and easily debug and test the sample code. In addition, the PICDEM 17 demonstration board supports downloading of programs to and executing out of external FLASH memory on board. The PICDEM 17 demonstration board is also usable with the MPLAB ICE in-circuit emulator, or the PICMASTER emulator and all of the sample programs can be run and modified using either emulator. Additionally, a generous prototype area is available for user hardware.

21.15 KEELOQ Evaluation and Programming Tools

KEELOQ evaluation and programming tools support Microchip's HCS Secure Data Products. The HCS evaluation kit includes a LCD display to show changing codes, a decoder to decode transmissions and a programming interface to program test transmitters.

FIGURE 22-16: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)

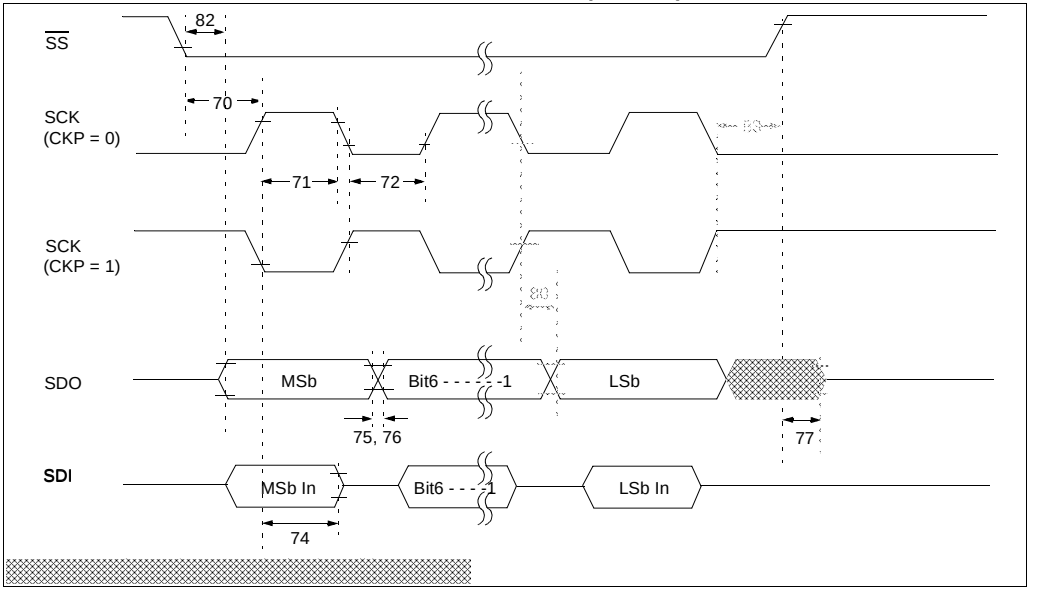


TABLE 22-15: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}\downarrow$ to SCK$\downarrow$ or SCK$\uparrow$ input		TCY	—	ns	
71	TscH	SCK input high time (Slave mode)	Continuous	1.25TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73A	TB2B	Last clock edge of Byte1 to the 1st clock edge of Byte2		1.5TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18**C**601/801	—	25	ns	
			PIC18**LC**601/801	—	45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
77	TssH2doZ	$\overline{SS}\uparrow$ to SDO output hi-impedance		10	50	ns	
78	TscR	SCK output rise time (Master mode)	PIC18**C**601/801	—	25	ns	
			PIC18**LC**601/801	—	45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18**C**601/801	—	50	ns	
			PIC18**LC**601/801	—	100	ns	
82	TssL2doV	SDO data output valid after $\overline{SS}\downarrow$ edge	PIC18**C**601/801	—	50	ns	
			PIC18**LC**601/901	—	100	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}\uparrow$ after SCK edge		1.5TCY + 40	—	ns	

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

APPENDIX E: DEVELOPMENT TOOL VERSION REQUIREMENTS

This lists the minimum requirements (software/ firmware) of the specified development tool to support the devices listed in this data sheet.

MPLAB® IDE: TBD

MPLAB® SIMULATOR: TBD

MPLAB® ICE 3000:

PIC18C601/801 Processor Module:
Part Number - TBD

PIC18C601/801 Device Adapter:

Socket	Part Number
64-pin TQFP	TBD
68-pin PLCC	TBD
80-pin TQFP	TBD
84-pin PLCC	TBD

MPLAB® ICD: TBD

PRO MATE® II: TBD

PICSTART® Plus: TBD

MPASM™ Assembler: TBD

MPLAB® C18 C Compiler: TBD

Note:	Please read all associated README.TXT files that are supplied with the development tools. These "read me" files will discuss product support and any known limitations.

NOTES:

