

Using The MCP2150 To Add IrDA[®] Standard Wireless Connectivity

*Author: Steve Schlanger
Aegis Technologies LLC*

INTRODUCTION

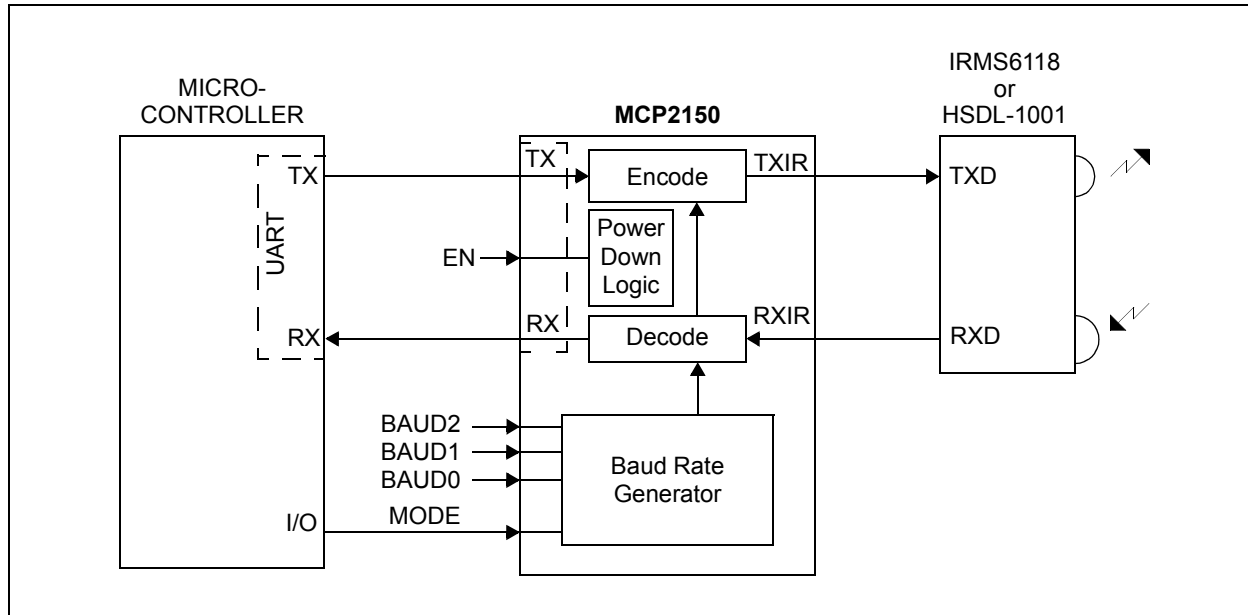
The MCP2150 is a cost effective, low pin count (18-pin) easy-to-use device for implementing IrDA[®] standard wireless connectivity. The MCP2150 provides support for the IrDA standard protocol stack plus bit encoding/decoding.

The MCP2150 encodes an asynchronous serial data stream, converting each data bit to the corresponding infrared (IR) formatted pulse. IR pulses that are received are decoded, and then handled by the protocol handler logic. The protocol handler will then send the appropriate data bytes to the host controller in UART formatted serial data.

The encoding/decoding functionality of the MCP2150 is designed to be compatible with the physical layer component of the IrDA standard. This part of the standard is referred to as “IrPHY”. A detailed discussion of this standard is beyond the scope of this application note, but a discussion regarding the encoding and decoding is in order. More detailed information is available from the IrDA standard website (www.IrDA.org).

The MCP2150 allows the easy addition of IrDA standard wireless connectivity to any embedded application that uses serial data. Figure 1 shows typical implementation of the MCP2150 in an embedded system.

FIGURE 1: SYSTEM BLOCK DIAGRAM



IrDA is a registered trademark of the Infrared Data Association.

INFRARED (IR) COMMUNICATIONS OVERVIEW

IR communications have advantages over wired serial connections. These advantages include:

- No connectors to wear out
- IR transceivers are smaller than common serial connectors
- Total immunity from Electromagnetic Interference (EMI) and power supply noise
- Very reliable, IR data is protected from errors using a 16-bit CRC algorithm
- Easy availability, many mobile devices have IR ports but no serial ports

Despite these advantages, IR communications have not been widely adopted for embedded use. This is primarily due to the cost and complexity of the IrDA standard that is commonly used to carry IR data. The MCP2150 addresses the cost and complexity issue for designers. The MCP2150 implements the IrCOMM common 9-wire “cooked” service class. This provides compatibility with Windows® operating systems, Microsoft® PocketPC, Palm PDAs, and Psion PDAs.

Note 1: IrDA standard infrared communication is supported for Windows® 95, Windows® 98, Windows® me and Windows® 2000. Microsoft does not support IrDA standard communications for Windows, versions 3.1 or lower, or for Windows NT®, versions 4.0 or lower.

2: Microsoft PocketPC is also known as Microsoft Windows CE 3.x. All versions of Windows® CE, including 1.x and 2.x, have support for IrDA standard infrared communication.

3: Palm introduced native support for IrDA standard communications starting with OS version 3.5. Commonly available terminal clients require Palm OS®, version 3.5 or higher. Palm introduced IR capability was introduced starting with OS version 3.0 but these older versions required developers to do many tasks, such as setting up the basics of the infrared link, manually. As a result, IR applications for Palm OS, versions prior to 3.5 are not common.

The MCP2150 allows the addition of IR connectivity to an embedded system with no more difficulty than adding a serial port connector. The designer need only supply data, clock, and handshake signals. The MCP2150 will then provide connectivity support to the tens of millions of IrDA standard infrared ports that are now deployed in the hands of the public.

INFRARED COMMUNICATION CONCEPTS

Sending data using IR light requires some hardware and the use of specialized communications protocols. These protocols and hardware requirements are described in detail by the IrDA standard specifications. A general description is given here to provide the MCP2150 user with enough information to make decisions about what kind of devices the MCP2150 can connect to and how these connections are implemented. The complete IrDA standard specifications are available for download from the IrDA standard website (www.IrDA.org). The hardware needed to encode and decode IR light to/from serial data is discussed later in this document.

Note: The MCP2150 is a stand-alone device encompassing all the required layers of the IrDA standard protocols. The MCP2150 does not require an IrDA encoder such as the MCP2120.

IrLAP LAYERS

The key parts and hierarchy of the IrDA standard protocols are identified as shown in Figure 2. The bottom layer is the physical layer, IrPHY. This is the part that converts the serial data to and from pulses of IR light. The basic problem is that the IR transceiver can't transmit and receive at the same time. The receiver has to wait for the transmitter to finish sending. This is sometimes referred to as a “Half-Duplex” connection. The IR Link Access Protocol (IrLAP) provides the structure for packets or “frames” of data to emulate data that would normally be free to stream back and forth.

FIGURE 2: IrDA STANDARD LAYERS

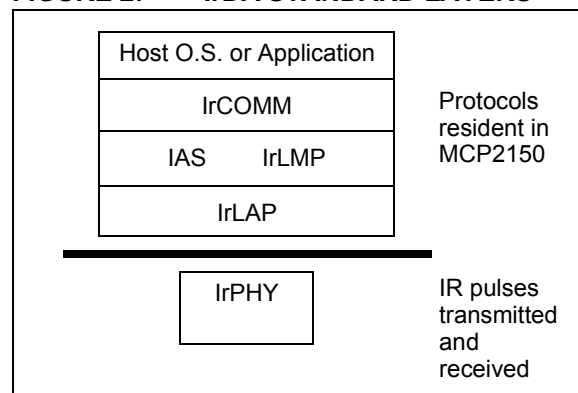


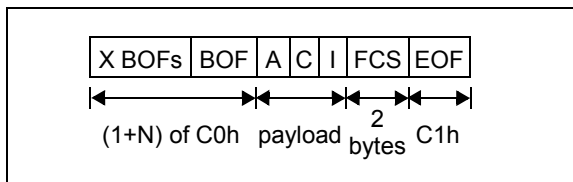
Figure 3 shows how the IrLAP frame is organized. The frame is preceded by some number of Beginning of Frame (BOF) characters. The value of the BOF is generally 0xC0, but 0xFF may be used if the last BOF character is a 0xC0. The purpose of multiple BOFs is to give the other station some warning that a frame is coming.

The IrLAP frame begins with an address byte (“A” field), then a control byte (“C” field). The control byte is used to differentiate between different types of frames and is also used to count frames. Frames can carry status, data, or commands. The IrLAP protocol has a command syntax of its own and these commands are part of the control byte. Lastly, IrLAP frames carry data. This data is the information or “I” field. The integrity of the frame is ensured with a 16-bit CRC, referred to as the Frame Check Sequence (FCS). The 16-bit CRC value is transmitted LSB first. The end of the frame is marked with an EOF character which is always a 0xC1. The frame structure described here is used for all versions of the IrDA standard protocols for serial wire replacement at speeds up to 115.2 kbaud.

Note 1: Another IrDA standard which is entering general usage is IR Object Exchange (IrOBEX). This standard is not used for serial connection emulation.

2: IrDA communication standards faster than 115.2 kbaud use a different CRC method and physical layer.

FIGURE 3: IrLAP FRAME



In addition to defining the frame structure, IrLAP provides the “housekeeping” function of opening and closing connections, and maintaining connections once they’re open. Part of this housekeeping are the critical parameters that determine the performance of the link. These parameters control how many BOFs are used, what is the speed of the link, how fast can either party change from receiving to transmitting, etc. IrLAP has the responsibility of negotiating these parameters to the highest common set so that both sides can communicate as fast and as reliably as possible.

IrLMP

When two devices that contain the IrDA standard feature connect, there is generally one device that has something to do, and the other device has a resource to do it. For example, a laptop may have a job to print and an IrDA standard compatible printer has the resources to print it. In IrDA standard terminology, the laptop is the Primary device and the printer is the Secondary device. When these two devices connect, the Primary device must ascertain the capabilities of the Secondary device to determine if the Secondary device is capable of doing the job. This determination is made by the Primary device asking the Secondary device a series of questions. Depending on the answers to these questions, the Primary device may or may not elect to connect to the Secondary device.

The queries from the Primary device are carried to the Secondary device using IrLMP. The responses to these queries can be found in the Information Access Service (IAS) of the Secondary device. The Primary device compares the IAS responses with its requirements and then makes the decision if a connection should be made.

The MCP2150 identifies itself to the Primary device as a modem.

Note: The MCP2150 identifies itself as a modem to ensure that it is identified as a serial device with a limited amount of memory.

The MCP2150 is not a modem, and the non-data circuits are not handled in a modem fashion.

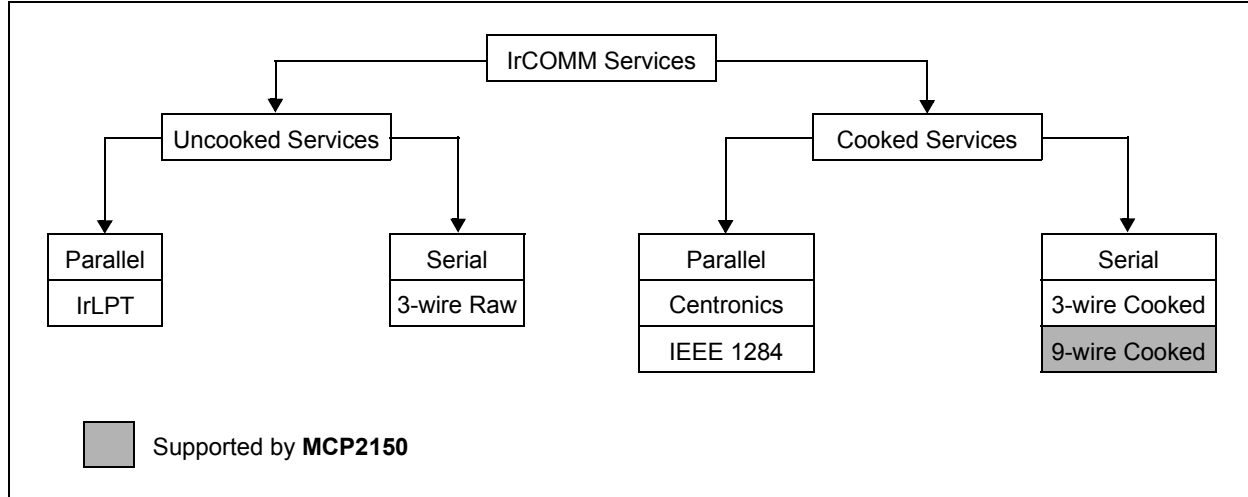
AN758

IrCOMM

The IrCOMM standard is simply a syntax that allows the Primary device to consider the Secondary device as a serial device. IrCOMM allows for emulation of

serial or parallel (printer) connections of various capabilities. The MCP2150 supports the 9-wire “cooked” service class of IrCOMM. Other service classes supported by IrCOMM are shown in Figure 4.

FIGURE 4: IrCOMM SERVICE CLASSES



EMBEDDED SYSTEM HARDWARE

The MCP2150 provides an alternative to a wired serial connection. Devices with serial ports can be divided into two categories, DTE devices and DCE devices. These terms correspond to the IrDA standard terms, Primary device (DTE) and Secondary device (DCE). Examples of DTE devices are PCs, PDAs, or terminals. An example of a DTE serial port is shown in Figure 5. The characteristic feature of a DTE device is that the Carrier Detect (CD) and Ring Indicate are inputs. Most embedded applications are considered to be DTE devices. An example would be a digital weighing scale implemented using a PICmicro[®] microcontroller. The scale may have a serial port for data logging purposes. The scale does not generate a carrier. The scale may, however, be used with a modem which does generate a carrier. The CD signal would therefore be an input to the scale and the scale would be considered a DTE device.

Note: These definitions are useful for determining how cables are wired if the scale has to be connected to a PC. It is not relevant for this discussion if the scale can be used with a modem or not.

FIGURE 5: DTE SERIAL PORT SIGNALS

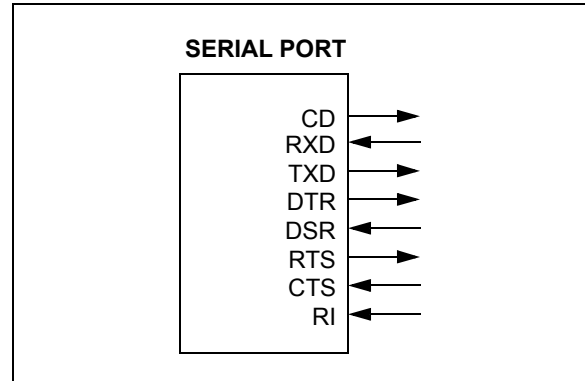
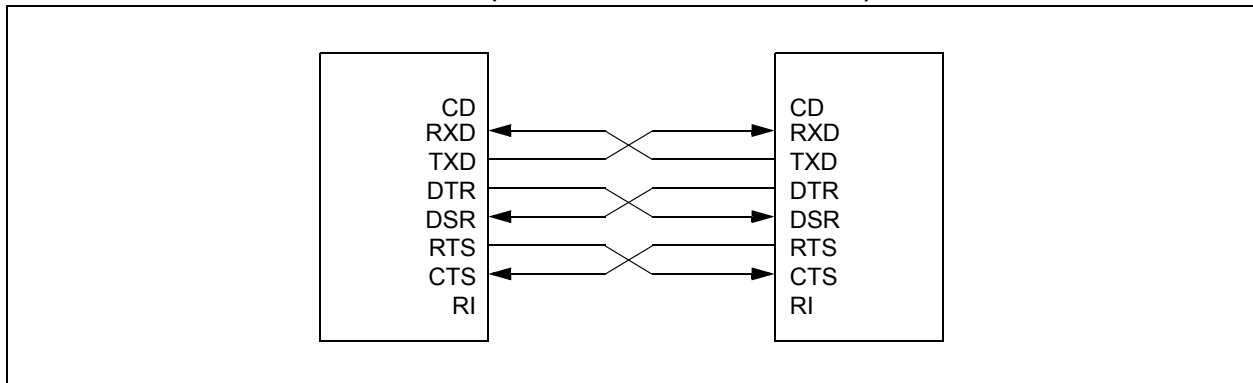


Figure 6 shows how two DTE devices with serial ports are connected with a serial cable. This connection is analogous to how the MCP2150 connects to the embedded application. The MCP2150 is designed to connect to DTE devices.

Note 1: A serial cable that has its signals crossed-over is sometimes referred to as a "Null Modem" cable. The name came into use because DTE devices may also connect to each other over a great distance using modems. The Null Modem cable emulates how two modems would be used to connect DTE devices together.

FIGURE 6: DTE TO DTE DEVICES (NULL MODEM CONNECTION)



AN758

The MCP2150 emulates a null modem connection as shown in Figure 7. The application on the DTE device sees a virtual serial port. This serial port emulation is provided by the IrDA standard protocols. The link between the DTE device and the embedded application is made using the MCP2150. The connection between the MCP2150 and the embedded application is wired as if there were a null modem connection.

The CD signal of the MCP2150 is used to indicate if a valid IrDA standard infrared link has been established between the MCP2150 and the IrDA standard Primary device (DTE host). Users are encouraged to monitor the CD signal closely to make sure that any communication tasks can be completed. The DTR signal simply indicates if the MCP2150 has been powered on. The MCP2150 has to generate the Clear To Send (CTS) signal locally because of buffer limitations described later.

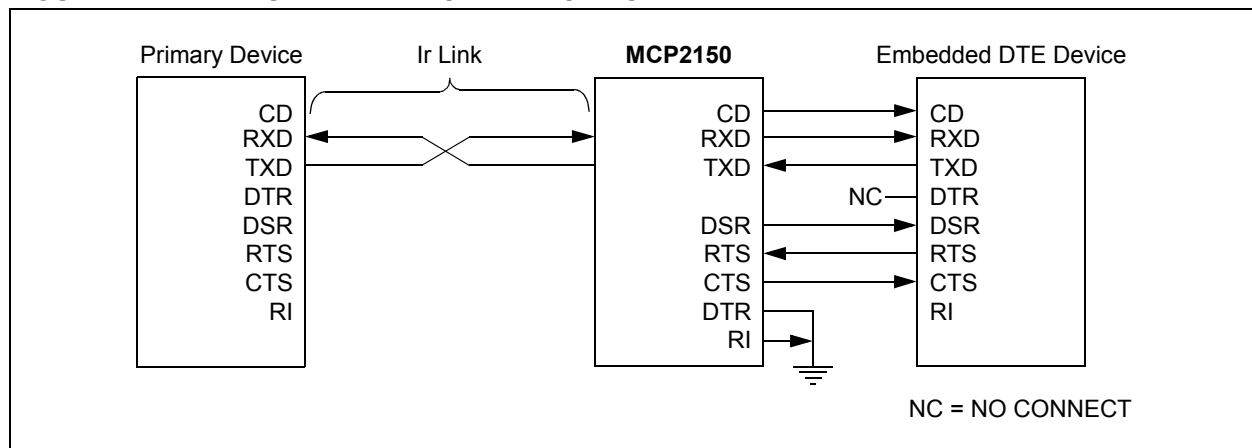
Note: The current DTE version of the MCP2150 generates non-data signals locally. Only TXD and RXD are carried back and forth to the primary device. Thus the MCP2150 emulates a 3-wire serial connection.

Hardware Handshaking

The MCP2150 uses a 64-byte buffer for incoming data from the IR Host. Another 64-byte buffer is provided to buffer data from the UART serial port. When an IR packet begins the IrComm, the MCP2150 handles IR data exclusively. So the UART serial port buffer is not available. A hardware handshaking pin (CTS) is provided to inhibit the host controller from sending serial data while IR data is being sent or received.

Note: When the CTS output from the IrComm is high, no data should be sent from the Host Controller. The UART FIFO will store up to two bytes. Any additional data bytes will be lost.

FIGURE 7: IrDA STANDARD CONNECTION TO EMBEDDED DTE DEVICE



Buffers and Throughput

The maximum IR data rate of the MCP2150 is 115.2 kbaud. The actual throughput will be less due to several factors, the most significant of which are under the control of the developer. One factor beyond the control of the designer is the overhead associated with the IrDA standards. The MCP2150 uses a fixed data block size of 64 bytes. To carry 64 bytes of data, the MCP2150 must send 72 bytes (64+8). The additional eight bytes are used by the IrDA standard protocol. When the Primary device receives the frame, it must wait for a minimum latency period before sending a packet of its own. This turnaround time is set by IrLAP when the parameters of the link are negotiated. A common turnaround time is 1 ms, although longer and shorter times are encountered. 1 ms represents approximately 12 byte times at a data rate of 115.2 kbaud. The minimum size frame that the Primary device can respond with is 6 bytes. The MCP2150 will add the 12 byte-time latency of its own, again assuming a 1 ms latency. This means that the maximum throughput will be 64 data bytes out of a total of 64 + 38 byte times. Thus, the maximum theoretical throughput will be limited to about $64/(64+38) = 63\%$ of the IR data rate. Actual maximum throughput will be between 38.4 kbaud and 57.6 kbaud. This difference is due to processing time of the receiving station and other factors.

The most significant factor in data throughput is how well the data frames are filled. If only 1 byte is sent at a time, then the maximum throughput is $1/(1+38) = 2.5\%$ of the IR data rate. The best way to maximize throughput is to align the amounts of data with the packet size of the MCP2150. Throughput examples are shown in Table 1.

Most operating systems do not give the user direct access to how data streams are divided into IrDA standard frames. Despite this limitation, the developer can affect how well the "packetizing" is done by passing fixed amounts of data at a time to the operating system.

Note 1: Some operating systems, i.e. Palm OS[®], will only send 62 data bytes in a 64 byte packet. Developers should try passing various sizes of data "chunks", if maximum throughput is needed.

2: Data transported using the IrDA standard is fully protected with a CRC-16 algorithm. For embedded applications, file transfer protocols should not be needed. If you need to pass a large amount of data to/from the MCP2150, then file transfer protocols, i.e. Xmodem, will slow down the transfer process considerably and reliability will not be improved. Also, terminal clients using a serial connection assume a turnaround time for the error-checking process that may not be possible with an IrDA standard link.

TABLE 1: IrDA STANDARD THROUGHPUT EXAMPLES @ 115.2 KBAUD

MCP2150 Data Packet Size (Bytes)	Overhead (Bytes)	Primary Device Minimum Response (Bytes)	Primary Device Turn-around Time ⁽¹⁾ (Bytes)	MCP2150 Turn-around Time ⁽¹⁾ (Bytes)	Total Bytes Transmitted	Throughput% (Data/Total)
64	8	6	12	12	102	62.7%
1	8	6	12	12	39	2.6%

Note 1: Number of bytes calculated based on a common turn-around time of 1 ms.

AN758

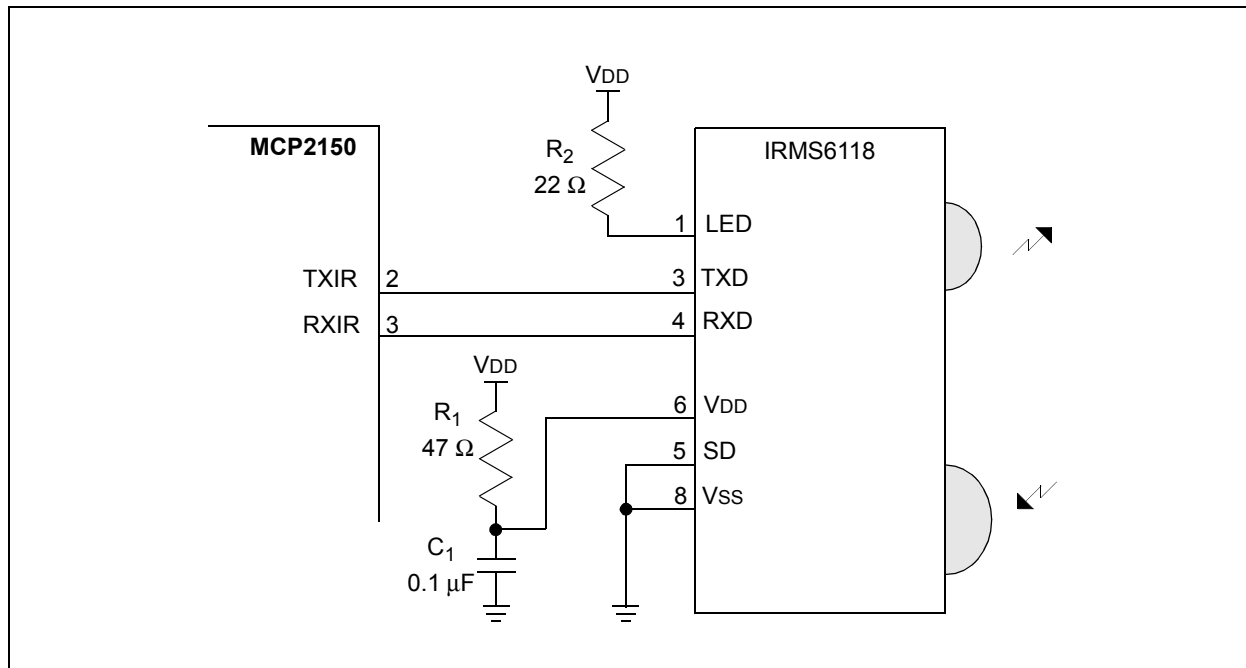
SYSTEM HARDWARE

Figure 8 shows that very few components are needed to implement IrDA standard wireless connectivity. The IR light pulses are converted to electrical pulses by the optical transceiver. The MCP2150 is connected directly to the optical transceiver. Resistor, R_1 and capacitor, C_1 are used to decouple the power supply of the optical transceiver from the rest of the system, since some transceivers have limited tolerance for power supply noise. This circuit will reduce 10 kHz power supply ripple by about 30 dB, if a good quality tantalum capacitor is used. Resistor, R_2 is used to limit the current of the emitter LED. Most transceivers use an external resistor for this purpose. Many infrared transceivers will emit an IR pulse when the transmit pin (TXD) is high, and will indicate a bit received by setting the receive pin (RXD) low.

The output impedance of the transceiver receive circuit may be $4\text{ k}\Omega$ or more, so the MCP2150 should be located as close to the transceiver as possible. A ground plane under the transceiver will improve electromagnetic interference (EMI) performance and reduce susceptibility to EMI.

For battery powered applications, it may be an advantage to turn off power to the MCP2150. If power is turned off completely, care should be taken so that none of the I/O pins are exposed to a signal greater than $V_{SS} \pm 0.6\text{V}$. In some systems, it may be preferable to shut down the MCP2150 and leave other parts of the system active, thus exposing the MCP2150 to active signals while shut down. If this is the case, then the EN input should be used. If the EN pin (pin 6) is low, the device is disabled. The current consumption in this mode will be typically less than $1\text{ }\mu\text{A}$ and active I/O signals from the rest of the system will not be a problem.

FIGURE 8: TYPICAL IrPHY CONFIGURATION



ENCODING

Figure 9 shows one-half (1st half) of an asynchronous serial byte sent by the MCP2150. Data to be transmitted is input to the MCP2150 on the TX pin (pin 12). The TX bit value in Figure 9 shows a data word to be sent.

Note 1: The signal on the TXIR pin does not actually line up in time with the bit value that was transmitted on the TX pin as shown in Figure 9. The TX bit value is shown to represent the value to be transmitted on the TXIR pin.

2: The sampling of the TX pin is level sensitive, not edge sensitive.

3: The MCP2150 does not indicate over-run errors. Care should be exercised to make sure the TX pin is low during the stop bit time.

4: An extended time period where TX is low (A BREAK), will result in the MCP2150 sending a string of 00h bytes as long as the TX pin is low.

The MCP2150 has a fixed IR transmit pulse width which is greater than 1.6 μ s.

Increasing Transmit Distance

The IrDA standard specifies a transmission distance of 1 m, with the emitter and received misaligned up to ± 15 degrees. Some applications require a greater distance. This can be achieved with an increase in emitter power, a lens for the receiver, or both. Figure 10 shows how adding LEDs can be used to increase the transmission distance.

Note 1: For every doubling of distance the emitter power must be increased by a factor of 4. Thus if a transmission distance of 2 m is needed, three emitter LEDs of similar efficiency to the LED built into the transceiver, would need to be added. For 4 m distance, 15 LEDs would be need to be added.

2: Few IR LEDs are fast enough for use in IrDA standard applications. The TON and TOFF for this device should be less than 100 ns.

Infrared emitters should have a wavelength centered at 875 nm. The author has used the Vishay/Temic TSSF4500 with excellent results. Typically, LEDs used in television-type remote controls have a wavelength of 950 nm and a TON and TOFF of 2 μ s or more. These type of LEDs are not recommended for IrDA standard applications.

FIGURE 9: Ir TRANSMISSION

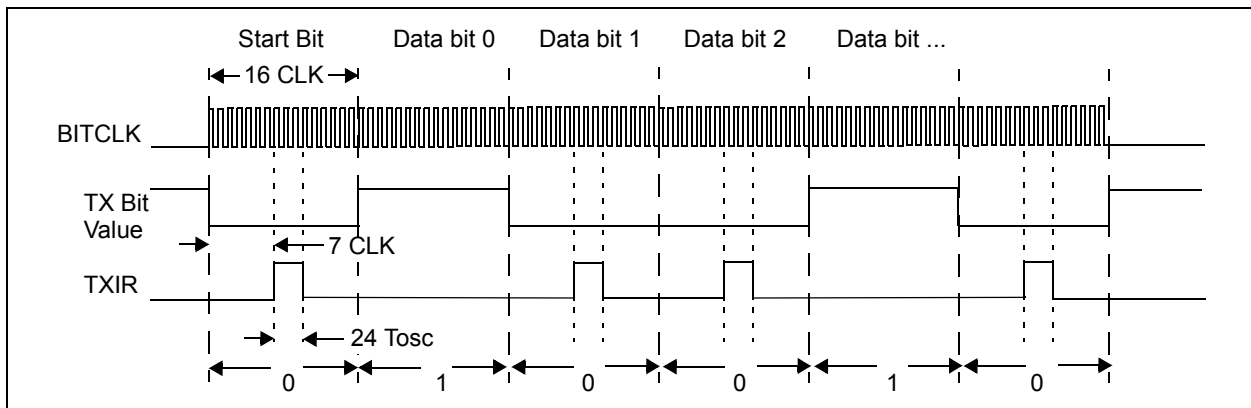
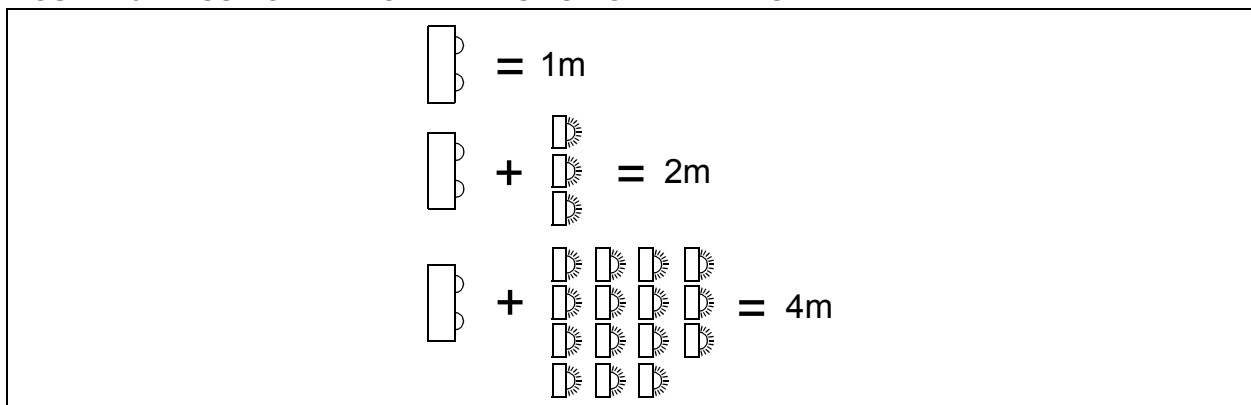


FIGURE 10: USING ADDITIONAL LEDs FOR GREATER DISTANCE



AN758

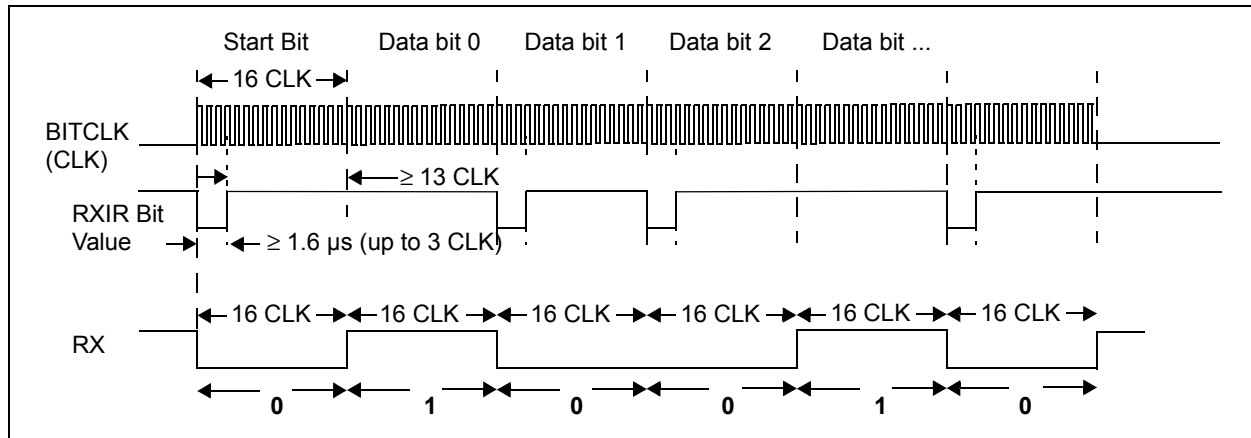
DECODING

The modulated signal (data) from the IR transceiver module (on RXIR pin) needs to be demodulated to form the received data (on RX pin). After demodulation occurs, the data that is received is transmitted by the MCP2150 UART (on the RX pin). Figure 11 shows the decoding of the modulated signal.

Note: The signal on the RX pin does not actually line up in time with the bit value that was received on the RXIR pin as shown in Figure 11. The RXIR bit value is shown to represent the value to be transmitted on the RX pin.

Many illumination sources, such as fluorescent lamps or sun light can introduce light noise that can interfere with proper data reception. For best results, the IR transceiver should not be pointed directly at a visible light source. Also, sunlight is rich in IR light. If the ambient IR light level is too high, then the IR data source may not be sufficient to trigger the receiver. For best results, IR transmission should not take place in direct sunlight.

FIGURE 11: IR DATA RECEPTION



HARDWARE DATA RATE SELECTION

The MCP2150 will encode and decode serial data at the currently selected data rate, or baud rate. The selection of this data rate is flexible and easy to use. Figure 12 shows how to use the BAUD1:BAUD0 input pins to implement Hardware select mode. Jumpers or I/O signals from another controller may be used, or these inputs may be tied directly to fixed voltage levels, if the data rate does not have to change.

After the MCP2150 is reset, the BAUD2:BAUD0 input pins are sampled. If all three of these inputs are high, then Software select mode is used. For any other inputs, Hardware select mode is active. This setting is latched when the device is reset, either from the **RESET** pin or a power-on reset. After the device reset, changing the value of the BAUD2:BAUD0 pins has no effect on the devices baud rate.

From Table 2, if a 9.6 kbaud data rate is desired at the device frequency of 11.0592 MHz, then the BAUD1:BAUD0 pins should be low.

TURNAROUND LATENCY

An IR link can be compared to a one-wire data connection. The IR transceiver can transmit or receive, but not both at the same time. A delay of one bit-time is suggested between the time a byte is received and another byte is transmitted.

FIGURE 12: USING HARDWARE DATA RATE SELECTION

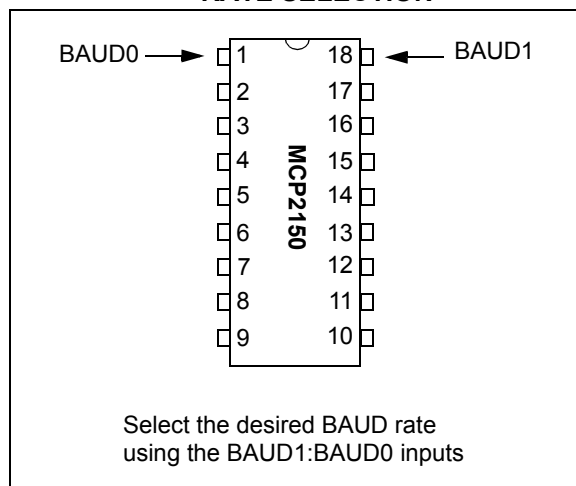


TABLE 2: HARDWARE MODE - BAUD RATE SELECTION

BAUD1:BAUD0	Baud Rate @ 11.0592 MHz	Bit Rate
00	9600	Fosc / 1152
01	19200	Fosc / 576
10	57600	Fosc / 192
11	115200	Fosc / 96

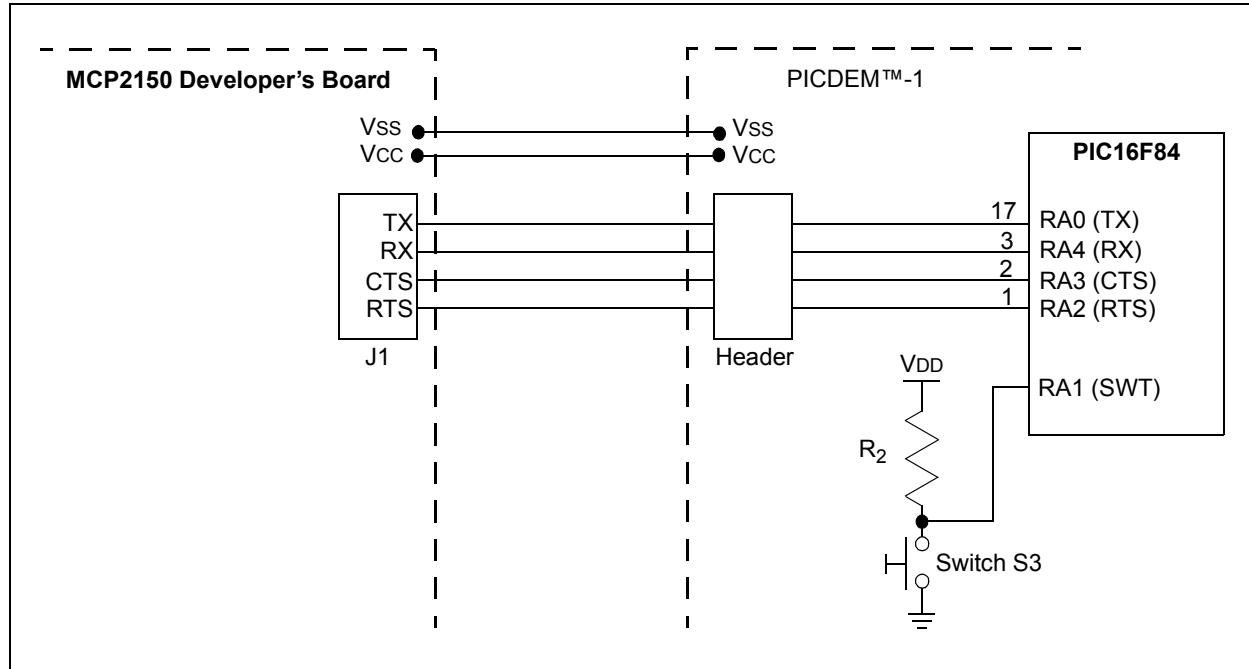
AN758

USING THE MCP2150 DEVELOPER'S BOARD

Figure 13 shows two examples of how to use the MCP2150 with PICmicro microcontrollers. The first example shows how wireless IR communication can be added to a minimum system using the PIC16F84. The

PIC16F84 sends an IR message of "Hello World" when switch S3 is pressed. IR bytes received by the PIC16F84 are displayed in binary form. This example uses hardware select mode and a firmware UART for the PIC16F84. Another example shows a PIC16F84 using its internal hardware UART and software select mode.

FIGURE 13: EMBEDDED IrDA STANDARD APPLICATION EXAMPLE



REFERENCES

The IrDA standards download page can be found at:

<http://www.irda.org/standards/specifications>

Manufacturers of Optical Transceivers are shown in Table 3.

TABLE 3: OPTICAL TRANSCEIVER MANUFACTURERS

Company	Company Web Site Address
Infineon	www.infineon.com
Agilent	www.agilent.com
Vishay/Temic	www.vishay.com
Rohm	www.rohm.com

MEMORY USAGE

The PIC16F84 program uses the following resources:

Program Memory: 137 words

Data Memory: 9 bytes

SUMMARY

The MCP2150 has a uniquely flexible combination of hardware, software, or FOSC selection of the data rate. The high integration, low power, and Windows compatibility make the MCP2150 well suited to implementing IrDA standard solutions in consumer, industrial, automotive, and telecommunications applications.

Software License Agreement

The software supplied herewith by Microchip Technology Incorporated (the "Company") for its PICmicro® Microcontroller is intended and supplied to you, the Company's customer, for use solely and exclusively on Microchip PICmicro Microcontroller products.

The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

APPENDIX A: PIC16F84 SOURCE CODE

EXAMPLE A-1: PIC16F84 Source Code

```

;*****
;
;           MCP2150 Demo with PICDEM 1 Board
;           Use with PIC16F84, 3.6864Hz clock
;           Checksum=9383 (cp on)
;*****
; Revision History
; 1.0      05/16/01  Initial Release
;
;*****
; Notes:
; This demo code sends/receives serial data at a fixed
; data rate. This rate can be from 9.6 to 38.4KB. The
; bitreg delay values for the various data rates are given
; below. The data sent is a string which is stored in a table. The
; string is sent when the PICDEM RA1 button is pressed.
; Any bytes received are displayed on the PortB LEDs.
; This version of the code assumes that the MCP2150
; jumpers have been set to match the data rate of this code.
;*****
LIST      C=132
include   P16F84.inc
#define   reset   H'00'           ;Reset vector
;*****
; Configuration Bits
    __CONFIG __CP_OFF & __PWRTE_ON & __XT_OSC & __WDT_OFF
    __IDLOCS H'0010'
;*****
; PortA Bits
;
#define   rts     PORTA,0         ;output, set low to allow data from MCP2150
#define   swt     PORTA,1         ;input, low when switch pressed
#define   rxd     PORTA,2         ;input, serial data from MCP2150
#define   txd     PORTA,3         ;output, serial data to MCP2150
#define   cts     PORTA,4         ;input, handshake from host
;
;
cfga     equ     B'00010110'     ;configuration for PORTA
;
#define   clear   PORTB,7         ;output, to MCP2150 MCLR
cfgb     equ     H'00'           ;PORTB is an output port
;
cfgopt   equ     B'11001000'     ;option reg setup
;

```

Example A-1: PIC16F84 Source Code - Page 2

```

;*****
; Constants
;
bytesz    equ    D'08'           ;there are 8 bits per byte
bitval    equ    D'08'           ;data bit delay for 19.2KB
;
;
;Data Rate Constants
; Rate   cyc   Bitval
;   9.6   96   20
;  19.2   48   08
;  38.4   24   02
;
;*****
; Registers
;
cblock    H'0C'
    areg                ;GP scratchpad
    breg                ;GP scratchpad
    bitreg              ;storage for data bit delay
    baudreg            ;storage for baud rate
    cmdreg              ;reg for commands
    delreg              ;reg for timing delays & scratchpad
    bitcnt              ;bit counter
    flags
    state                ;reg for state counter
endc
;
;*****
org        H'00'           ;use 00h as reset vector
goto      start
;
;
;*****
; String Table
; This table stores a string, breg is the offset. The string
; is terminated by a null.
;*****
string1    clrf    PCLATH           ;this routine is on page 0
           movf    breg,w           ;get the offset
           addwf   PCL,f           ;add the offset to PC
           DT      "Hello World" ;
           DT      H'0D',H'0A'     ;the string also contains a CR+LF
           DT      H'00'           ;terminate with 00h
;
;
;*****
; Delay Routine
; Each unit change of delay value changes the delay by 4 cycles.
; The delay value is passed in W.
;*****
delay      movwf   delreg
dellp     nop
           decfsz  delreg,f
           goto    dellp
           retlw   0
;

```

AN758

Example A-1: PIC16F84 Source Code - Page 3

```
*****
; Transmit serial Routine
; This routine sends the areg byte to the serial port (Txd).
; The CTS input pin should be low before the byte is sent.
;
*****
txser    btfsc    cts                ;check the cts input
         goto    txser              ;not ready, wait
;
         bcf     txd                ;begin the start bit
         nop
         nop
         nop
         nop
;
txdb     movf    bitreg,w
         call   delay
         nop
         nop
         btfsc  areg,0              ;if bit=0 then rxd=0
         goto  txdb1              ;if bit=1 then rxd=1
txdb0    nop
         nop
         bcf     txd                ;ir detected, bit=0
         rrf     areg,f            ;rotate the byte
         decfsz bitcnt,f          ;all bits rev'd?
         goto  txdb              ;ir recv'd, toggle routine
         goto  txsp
;
txdb1    nop
         bsf     txd                ;rotate the byte
         rrf     areg,f            ;all bits rev'd?
         goto  txdb              ;
         goto  txsp
;
txsp     nop
         nop
         nop
         movlw  bytesz            ;delay until the end of the 8th data bit
         movwf  bitcnt
         movf   bitreg,w
         call  delay
         bsf    txd                ;8th data bit ends here
         movf  bitreg,w            ;do the stop bit delay
         call  delay
         movf  bitreg,w            ;delay beyond the stop bit to allow for slow sys-
tems
         call  delay
         retlw 0
;
;
```

Example A-1: PIC16F84 Source Code - Page 4

```

;*****
;   Receive Serial Routine
;   This routine gets an incoming serial byte and stuffs it
;   into areg
;*****
rxser    nop                ;delay from the beginning of the start bit
        nop
        nop
        nop
        nop
        nop
        nop
        nop

;
rxdb     movf    bitreg,w
        call   delay
        nop
        nop
        rrf    areg,f        ;rotate the byte
        btfsc  rxd          ;if rxd=0 then the bit=0
        goto   rxdb1        ;if rxd=1 then bit=1
rxdb0nop
        nop
        bcf    areg,7        ;clear the bit
        decfsz bitcnt,f     ;all bits rev'd?
        goto   rxdb         ;ir recv'd, toggle routine
        goto   rxsp

;
rxdb1nop
        bsf    areg,7        ;set the bit
        decfsz bitcnt,f     ;all bits rev'd?
        goto   rxdb         ;
        goto   rxsp

;
rxsp     movlw  bytesz       ;reset the bit counter
        movwf bitcnt
        movf  bitreg,w      ;do the stop bit delay
        call  delay
        retlw 0

;
;

```

AN758

Example A-1: PIC16F84 Source Code - Page 5

```
;*****  
;   Start Routine  
;   The post-reset setup is done here  
;*****  
start   movlw    trisa                ;setup I/O  
        movwf   fsr  
        movlw   cfga  
        movwf   indf  
  
;       movlw   trisb  
        movwf   fsr  
        movlw   cfgb  
        movwf   indf  
  
;       movlw   Option_reg           ;setup option reg  
        movwf   fsr  
        movlw   cfgopt  
        movwf   indf  
  
;       movlw   H'00'                ;clear outputs  
        movwf   portb  
        bsf    clear                 ;allow MCP2150 to come out of reset  
        bsf    txd                   ;setup quiescent state  
        bcf    rts                    ;rts is set low to allow data from MCP2150  
  
;       movlw   bitval                ;  
        movwf   bitreg  
        movlw   bytesz                ;setup bit count  
        movwf   bitcnt  
  
;       goto    main  
  
;*****  
;   Main Routine  
;*****  
main    btfss   swt                   ;check for keypress  
        goto    send                 ;key is pressed, send the bytes  
        btfss   rxd                   ;check for an incoming byte from MCP2120  
        goto    getser               ;there's an incoming byte, go get it  
        goto    main  
  
;  
;
```

Example A-1: PIC16F84 Source Code - Page 6

```

;*****
;   Send routine
;   This routine sends the data found in sndtab
;*****
send   clrfs   breg           ;clear the offset
sndlp  call    string1       ;get the byte
       movwf  areg           ;save the byte
       incf   breg,f         ;increment the table pointer
       movf   areg,f         ;move the byte to test it
       btfsc STATUS,Z       ;if z=1 then we're done
       goto  sendex         ;we're done, do the exit
       call  txser          ;send the byte in areg
       goto  sndlp

;
sendex btfss   swt           ;check for key release
       goto  sendex         ;key is pressed, wait for release
       movlw H'FF'         ;do a debounce delay
       call  delay
       goto  main           ;return to waiting

;
;
;*****
;   Get serial routine
;   This routine gets a serial byte and displays the value
;   on the PICDEM PORTB leds.
;*****
getser call    rxser         ;get the serial byte
       movf   areg,w         ;w = serial byte
       movwf PORTB         ;move the byte to the output
       goto  main

;
;
;
end

```

AN758

NOTES:

NOTES:

AN758

NOTES:

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks


The Microchip name and logo, the Microchip logo, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, KEELoQ, SEEVAL, MPLAB and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

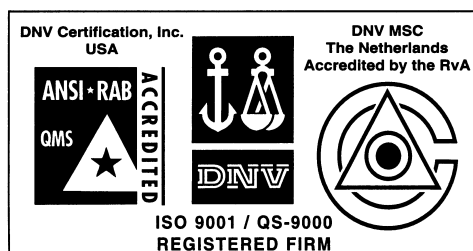
Total Endurance, ICSP, In-Circuit Serial Programming, Filter-Lab, MXDEV, microID, *FlexROM*, *fuzzyLAB*, MPASM, MPLINK, MPLIB, PICC, PICDEM, PICDEM.net, ICEPIC, Migratable Memory, FanSense, ECONOMONITOR, Select Mode and microPort are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Term Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2001, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELoQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.



MICROCHIP

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office

2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: <http://www.microchip.com>

Rocky Mountain

2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

Atlanta

500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Austin - Analog

8303 MoPac Expressway North
Suite A-201
Austin, TX 78759
Tel: 512-345-2030 Fax: 512-345-6085

Boston

2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Boston - Analog

Unit A-8-1 Millbrook Tarry Condominium
97 Lowell Road
Concord, MA 01742
Tel: 978-371-6400 Fax: 978-371-0050

Chicago

333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas

4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Dayton

Two Prestige Place, Suite 130
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit

Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Los Angeles

18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York

150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose

Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto

6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia

Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing

Microchip Technology Consulting (Shanghai)
Co., Ltd., Beijing Liaison Office
Unit 915
New China Hong Kong Manhattan Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu

Microchip Technology Consulting (Shanghai)
Co., Ltd., Chengdu Liaison Office
Rm. 2401, Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-6766200 Fax: 86-28-6766599

China - Fuzhou

Microchip Technology Consulting (Shanghai)
Co., Ltd., Fuzhou Liaison Office
Rm. 531, North Building
Fujian Foreign Trade Center Hotel
73 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7557563 Fax: 86-591-7557572

China - Shanghai

Microchip Technology Consulting (Shanghai)
Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

China - Shenzhen

Microchip Technology Consulting (Shanghai)
Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-2350361 Fax: 86-755-2366086

Hong Kong

Microchip Technology Hongkong Ltd.
Unit 901, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India

Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaughnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan

Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471-6166 Fax: 81-45-471-6122

Korea

Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore

Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan

Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark

Microchip Technology Denmark ApS
Regus Business Centre
Lautrup høj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France

Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - 1er Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany

Arizona Microchip Technology GmbH
Gustav-Heinemann Ring 125
D-81739 Munich, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Germany - Analog

Lochhamer Strasse 13
D-82152 Martinsried, Germany
Tel: 49-89-895650-0 Fax: 49-89-895650-22

Italy

Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom

Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

06/01/01