AN858

Interfacing The MCP215X to a Host Controller

Author: *Mark Palmer*
Microchip Technology, Inc.

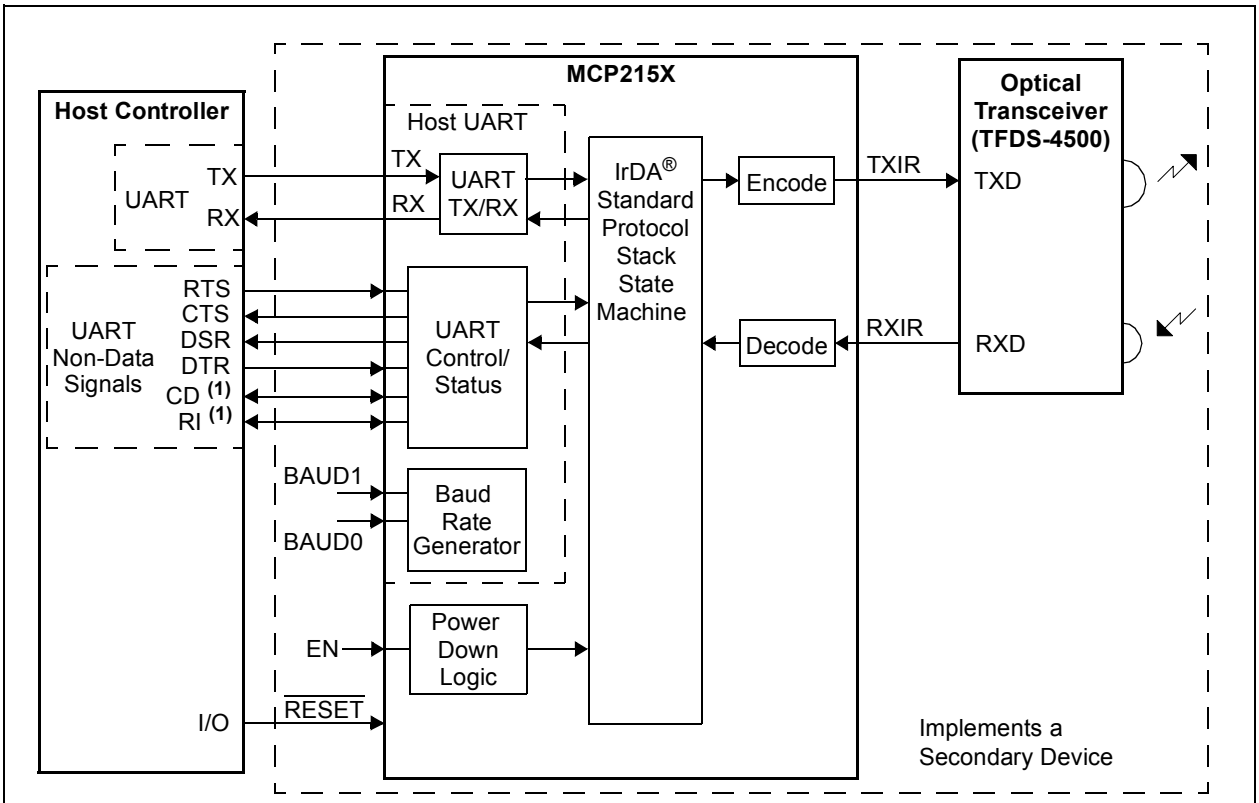
INTRODUCTION

This application note discusses the operation of the MCP215X Host UART interface, implements an embedded system (as an IrDA® Standard Secondary device), and describes the setup of Personal Digital Assistants (PDA) devices to operate as the IrDA Standard Primary device.

The Host UART interface includes non-data Flow Control signals. These are the signals between a Host Controller and a MCP215X device (see Figure 1). References in this document to the MCP215X device mean either the MCP2150 device or the MCP2155 device.

The embedded system is comprised of an Optical Transceiver circuit, a MCP215X device and a Host Controller (PIC16F87X). This typical embedded system implementation is shown in Figure 1.

FIGURE 1: TYPICAL MCP215X SYSTEM BLOCK DIAGRAM



Note 1: The CD and RI signals have different directions (and functions) between the MCP2150 and the MCP2155.

2: Please refer to MCP2150 Data Sheet (DS21655) or MCP2155 Data Sheet (DS21690) for the function of the Host UART signals (TX, RX, RTS, CTS, DSR, DTR, CD, and RI).

Figure 2 shows the two interfaces that the MCP215X has to offer. These are:

1. an IR Interface.
2. a Host UART Interface.

When the MCP215X is functioning on the IR Interface, the Host UART Interface is ignored.

After the reception of an IR packet, the MCP215X has a turnaround time of up to 100 ms. This time is negotiated during the Discovery process between the Primary Device and the MCP215X. During this turnaround time, the MCP215X will parse the received IR packet and respond according to the IrDA Standard Protocol, giving the Host UART a Receive Data Window and other tasks.

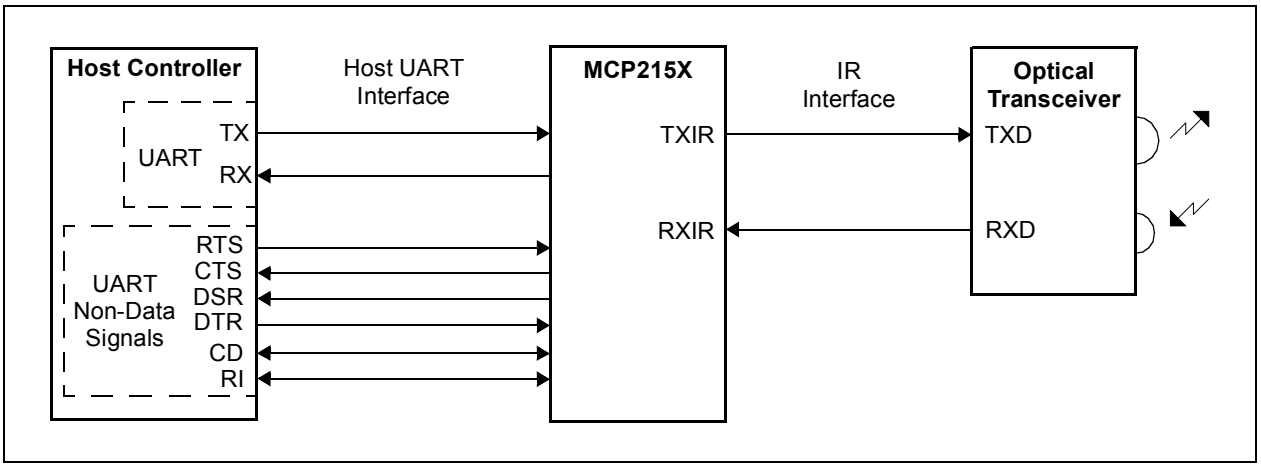
Data exchange on the Host UART Interface can only occur during the Receive Data Window, or after the MCP215X has received an IR packet containing “data” (IR data packet). For this reason, the Host UART Interface flow control must be observed by the Host Controller.

In order to ease the development of your application, an assembly code program that interfaces a PIC16F87X to a MCP215X is included. This program is discussed as well as being illustrated in the flowcharts labeled Figures 7 thru 14.

Using this program, captured waveforms of communication between a Host Controller (PIC16F87X) and a MCP2150 are presented.

The embedded system is a Secondary device and requires a Primary device to “talk” with. Step-by-step setup of a Palm™ Personal Digital Assistant (PDA) and an iPAQ PDA (running PocketPC) are shown along with the steps to operate the application.

FIGURE 2: MCP215X SYSTEM INTERFACE DIAGRAM



HOST UART FLOW CONTROL

The MCP215X uses up to eight signals for the Host UART interface, described in Table 1.

In addition to the UART Transmit and Receive functions (the TX and RX signals), there are three important functions associated with flow control. These functions do the following:

1. Indicates when the IR link is “established” (the CD signal on the MCP2150 and the DSR signal on the MCP2155).
2. Indicates when the Host Controller can transmit data to the MCP215X (the CTS signal).
3. Indicates when the Host Controller can receive data from the MCP215X (the RTS signal).

The DTR, DSR and RI signals are not associated with the Host UART Interface flow control. Depending on the MCP215X device, these signals may indicate device status information over the IR link or the signal may not have a function.

Establishing a Link

Until the MCP215X device has established a link with a Primary device, the Host UART Interface is essentially “non-operational”. That is, the Host Controller should not send data (the CTS signal will not be active) and the Host Controller will not receive data (even with the RTS signal driven active by the Host Controller).

The IR link is “established” once the MCP215X device has completed Discovery mode (with a Primary device). If the “IR Link is Established” signal does not become active, the Primary device has not completed the discovery phase with the MCP215X. A connection sequence overview is shown in Figure 3.

Note: A personal computer (PC) running the Windows® Operating System (O.S.) with an IR driver may show the IR Icons. There are three cases:

1. A single IR icon:
   This means the PC is searching for a Secondary device.
2. Two icons facing each other:
   This means the PC (Primary device) has recognized a Secondary device. The two devices are still in Normal Disconnect Mode (NDM) (a link has not been “established”).
3. Two icons “communicating”:
   This means that a link has been “established” (Discovery is complete). Once the link is established, the IR monitor window will display the negotiated data rate and the frequency of communication errors. For the Primary device (PC) to complete Discovery, an application (such as Hyperterminal) will need to be “connected” to the IR Driver.

MCP2150 (THE CD SIGNAL)

The CD signal is an output from the MCP2150 and indicates that the Primary device and the MCP2150 have “Established an IR Link”. That is, they have completed Discovery phase of the IrDA Standard and the MCP2150 is in Normal Response Mode (NRM). Therefore, the IR link is open and data may be transmitted between the Primary and Secondary devices (MCP2150 embedded system).

The CD signal will be active (driven low) as long as the IR link is open. Once the IR link has been closed, the CD signal will be driven inactive.

MCP2155 (THE DSR SIGNAL)

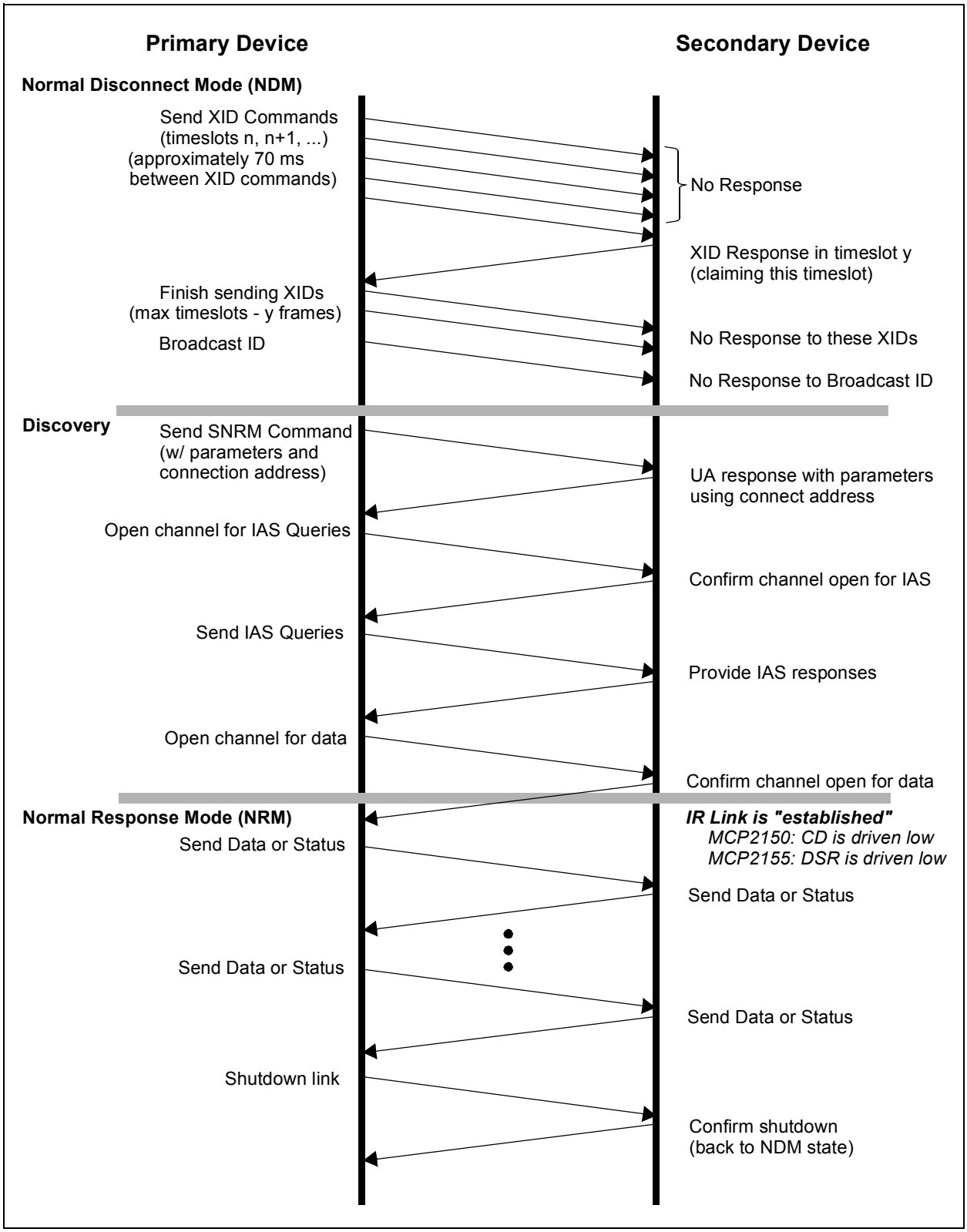
The DSR signal is an output from the MCP2155 and indicates that the Primary device and the MCP2155 have “Established an IR Link”. That is, they have completed Discovery phase of the IrDA Standard and the MCP2155 is in Normal Response Mode (NRM). Therefore, the IR link is open and data may be communicated between the Primary and Secondary devices (MCP2155 embedded system).

The DSR signal will be active (driven low) as long as the IR link is open. Once the IR link has been closed, the DSR signal will be driven inactive.

TABLE 1: HOST UART SIGNALS

Signal	Device	Direction	Description
TX	MCP2150 and MCP2155	I	Asynchronous receive; from Host Controller UART.
RX	MCP2150 and MCP2155	O	Asynchronous transmit; to Host Controller UART.
CTS	MCP2150 and MCP2155	O	Clear To Send. Indicates the MCP215X is ready to receive data from the Host Controller. 1 = Host Controller should not send data. 0 = Host Controller may send data.
RTS	MCP2150 and MCP2155	I	Request To Send. Indicates a Host Controller is ready to receive data from the MCP215X. The MCP215X prepares to send data, if available. 1 = Host Controller not ready to receive data. 0 = Host Controller ready to receive data. At device power-up, this signal is used with the DTR signal to enter Device ID programming. 1 = Do not enter Device ID programming mode. 0 = Enter Device ID programming mode (if DTR is set).
DTR	MCP2150	I	Data Terminal Ready. The value on this pin is ignored after the MCP2150 is initialized. It is recommended that this pin be connected so that the voltage level is either VSS or VCC. At device power-up, this signal is used with the RTS signal to enter Device ID programming. 1 = Enter Device ID programming mode (if RTS is cleared). 0 = Do not enter Device ID programming mode.
	MCP2155		Data Terminal Ready. Indicates that the Embedded device connected to the MCP2155 is ready for IR data. The state of this bit is communicated to the IrDA Primary device, via the IrDA bit carried by IrCOMM. 1 = Embedded device not ready. 0 = Embedded device ready. At device power-up, this signal is used with RTS to enter Device ID programming. 1 = Enter Device ID programming mode (if RTS is cleared). 0 = Do not enter Device ID programming mode.
DSR	MCP2150	O	Data Set Ready. Indicates that the MCP2150 has completed reset. 1 = MCP2150 is initialized. 0 = MCP2150 is not initialized.
	MCP2155		Data Set Ready. Indicates that the MCP2155 has established a valid link with a Primary Device. This signal is locally emulated and not related to the DTR bit of the IrDA Primary Device. 1 = An IR link has not been established (No IR Link). 0 = An IR link has been established (IR Link).
CD	MCP2150	O	Carrier Detect. Indicates that the MCP2150 has established a valid link with a Primary device. 1 = An IR link has not been established (No IR Link). 0 = An IR link has been established (IR Link).
	MCP2155	I	Carrier Detect. The state of this bit is communicated to the IrDA Primary device. 1 = No Carrier Present. 0 = Carrier Present.
RI	MCP2150	O	Ring Indicator. The value on this pin is driven high.
	MCP2155	I	Ring Indicator. The state of this bit is communicated to the IrDA Primary device. 1 = No Ring Indicate Present. 0 = Ring Indicate Present.

FIGURE 3: CONNECTION SEQUENCE OVERVIEW



Data From Host Controller to MCP215X (CTS Operation)

The CTS signal is an output from the MCP215X device and is used to indicate when the Host Controller may transmit data on the Host UART.

The MCP215X device operation requires that communication only occur on the MCP215X's IR Interface or Host UART Interface at a given time. The MCP215X will indicate when the Host Controller can communicate on the Host UART via the CTS signal. When an IR packet begins (IrCOMM), the MCP215X handles IR data exclusively and the MCP215X Host UART Interface is not available. The CTS signal indicates when the Host Controller can send serial data and when the Host Controller should not send serial data, since asynchronous IR Data is being sent or received.

The MCP215X uses a 64-byte buffer for incoming data from the Host UART serial port. When the CTS signal is driven active (low), the 64-byte buffer is clear and can receive up to the maximum 64-byte buffer space.

When the CTS signal is driven low, this indicates the beginning of the UART Receive Buffer's "Receive Data Window" (the UART Receive Buffer is empty). This Receive Data Window is 11.9 ms and is "closed" early if the UART Receive Buffer receives 64 bytes before the 11.9 ms is complete.

Once the MCP215X has received 60 bytes of the 64 byte buffer, the CTS signal will be de-asserted (driven high). Though the MCP215X can continue to receive the additional 4 bytes, the CTS signal is de-asserted early in case the Host Controller UART contains a small FIFO buffer. This early indication of the CTS signal allows these devices time to respond so as not to overflow the MCP215X UART Receive Buffer.

Figure 4 through Figure 6 show the three possible cases for the CTS signal waveform.

FIGURE 4: CTS WAVEFORM FOR <60 BYTES INTO UART RECEIVE FIFO

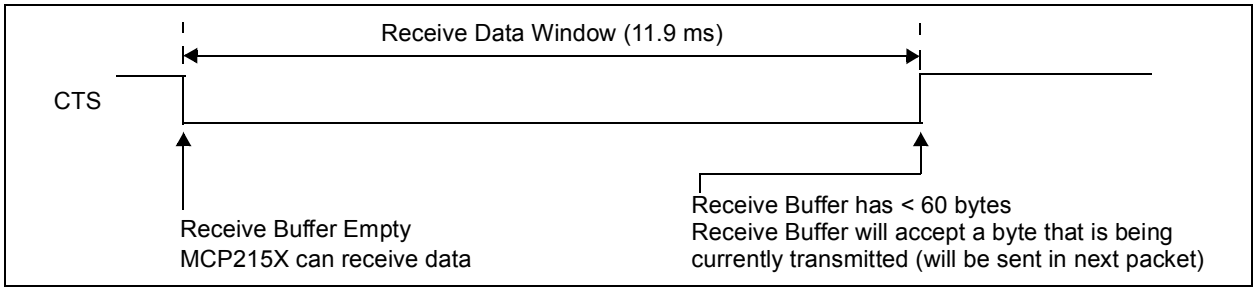


FIGURE 5: CTS WAVEFORM FOR BETWEEN 60 AND 64 BYTES INTO UART RECEIVE FIFO

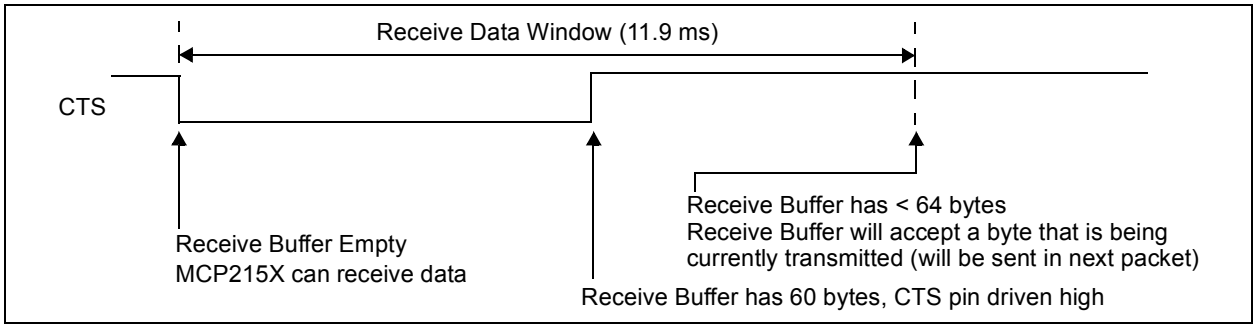


FIGURE 6: CTS WAVEFORM FOR 64 BYTES INTO UART RECEIVE FIFO

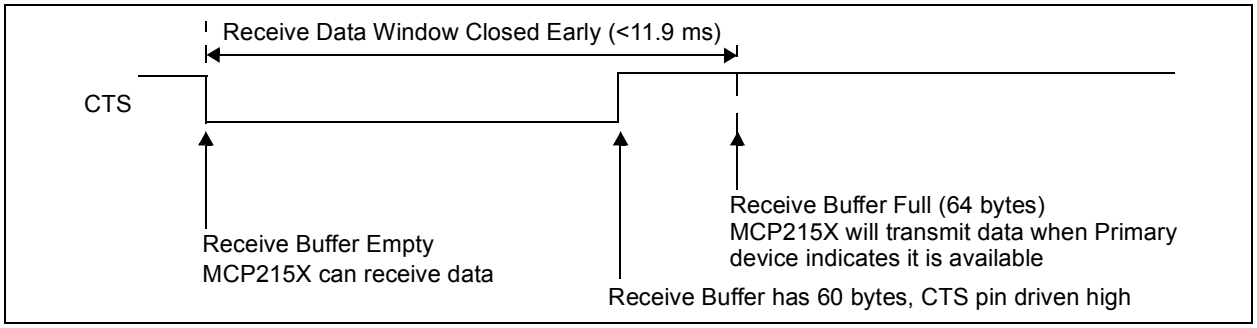


Figure 4 illustrates the case in which the Host Controller UART transmits less than 60 bytes during the Receive Data Window. In this case, the MCP215X CTS signal will de-assert (drive high) when the Receive Data Window time (11.9 ms) expires.

Figure 5 illustrates the case in which the Host Controller UART transmits more than 60 bytes, but less than 64 bytes, during the Receive Data Window. In this case, the MCP215X CTS signal will de-assert (drive high) once 60 bytes have been received, though the Receive Data Window will remain open the entire 11.9 ms.

Figure 6 illustrates the case in which the Host Controller UART transmits 64 bytes during the Receive Data Window. In this case, the MCP215X CTS signal will de-assert (drive high) once 60 bytes have been received. The Receive Data Window will close once the 64th byte is received.

Once the MCP215X Receive Data Window is closed, the MCP215X may transmit the data in the MCP215X UART Receive Buffer. While the MCP215X is ready to send the data, it will not do so until the Primary device indicates that it is available for the Secondary device (MCP215X). This time is completely dependent on the Primary device, and affects the system throughput.

Due to the Receive Data Window, the number of bytes that can be transmitted is dependent on the baud rate used for data transfer. Table 2 shows the maximum number of bytes that can be received by the MCP215X during the Received Data Window for the four different Host UART baud rates.

TABLE 2: RECEIVE DATA WINDOW TIME AND NUMBER OF BYTES TRANSFERRED

Baud Rate	Receive Data Window (ms)	Maximum Bytes Transferred	Comment
9600	11.9	12	**Note 1, 3**
19200	11.9	23	**Note 1, 3**
57600	11.9	68	**Note 2, 3**
115200	11.9	137	**Note 2, 3**

Note 1: CTS Trigger point of 60 bytes can not be reached. The CTS signal can be monitored for close of Receive Data Window.

2: Maximum bytes transferred exceeds MCP215X buffer size of 64 bytes. Once 64 bytes have been received, Receive Data Window will be closed.

3: Any byte that is in the process of being transmitted will be received by the MCP215X UART buffer (up to 64 bytes). This means that at 9600 baud, 13 bytes could be transferred/packet and at 19200 baud, 24 bytes could be transferred/packet (see Figure 15).

Host Controllers that are monitoring when the CTS signal will go active can stream 64 bytes if the Host UART baud rate is 57600 or greater (see Table 2). It is important to minimize the latency from the falling edge of the CTS signal to the 1st data byte transmitted. It is also important to ensure that there are no delays between bytes that would cause this transmission to require more than the 11.9 ms of the Receive Data Window.

If additional data bytes arrive at the MCP215X's TX pin after the Receive Data Window completes, unexpected operation may occur (such as the MCP215X UART Receive Buffer not being empty when CTS goes low for the next window time).

Note:	Data bytes received after the Receive Data Time Window may be lost, since the UART FIFO only stores up to 2 bytes.

The CTS high time after the completion of the Receive Data Time Window is dependent on the Primary device and not the MCP215X. This data packet is sent during the CTS high time, but there may be one or more exchanges of packets with the Primary device before more data may be sent. When the Primary device is ready for more data, and the MCP215X is ready to accept UART data, the CTS signal will be driven low.

Data From MCP215X to Host Controller (RTS Operation)

When the Host Controller drives the RTS signal active, the MCP215X is allowed to transmit data contained in the IR Receive buffer to the Host Controller. The Host Controller should use this signal to inhibit the MCP215X from sending data when the Host Controller does not have the processing bandwidth to “handle” the received data. In many applications, the Host Controller will not have bandwidth issues, and the RTS signal can be tied active (low).

When data is in the MCP215X IR Receive Buffer and RTS is high, the MCP215X will ignore the RXIR pin (IR Receive). This means that the IrDA Standard Handshaking packets from the Primary device are not responded to.

If the Host Controller does not drive the RTS signal active (low) by a given time, the Primary device will see the non-response from the MCP215X as an “obstruction” and may shut down the link (dependant on the Primary device used).

If the IrDA link is lost due to the MCP215X not being able to transfer the “data” to the Host Controller (MCP215X is waiting for the RTS signal to be driven low), the MCP215X will continue to drive CD active (low), which is helpful determining the cause of the lost link.

Minimum Flow Control Interface Requirement

In some applications, the Host Controller of the MCP215X will be I/O limited. The minimum number of Flow Control signals required to operate the MCP215X is 1 (CTS) . All other signals are either ignored (if there is an output signal from the MCP215X) or driven to a known level (if there is an input signal to the MCP215X).

The MCP215X can only drive the CTS signal low once an IR link is established. At all other times, the CTS signal will either not be driven (in reset/initialization) or driven high (no IR link/not the Receive Data Window).

HOST CONTROLLER REQUIREMENTS/ LIMITATIONS WITH THE MCP2150

Implementing this minimum Flow Control Interface puts requirements and limitations on the Host Controller. These include:

1. The RTS signal:
   If receiving data from a Primary device, then RTS will need to be tied low. The Host Controller needs to ensure that every received byte can be serviced, so no bytes will be lost.
2. The DTR signal:
   The DTR signal will need to be tied low. This means that the Host Controller cannot modify the MCP215X Programmable Device ID.
3. The DSR signal:
   The DSR signal can be ignored. A useful Host Controller firmware check of MCP2150 initialization is lost.
4. The CD signal:
   The CD signal can be ignored. A useful Host Controller firmware check of MCP2150 IR Link is lost.
5. The RI signal:
   The RI signal can be ignored.

HOST CONTROLLER REQUIREMENTS/ LIMITATIONS WITH THE MCP2155

Implementing this minimum Flow Control Interface puts requirements and limitations on the Host Controller. These include:

1. The RTS signal:
   If receiving data from a Primary device, then RTS will need to be tied low. The Host Controller needs to ensure that every received byte can be serviced, so no bytes will be lost.
2. The DTR signal:
   The DTR signal will need to be tied low. This means that the Host Controller cannot modify the MCP215X Programmable Device ID.
3. The DSR signal:
   The DSR signal can be ignored. A useful Host Controller firmware check of MCP2155 IR Link is lost.
4. The CD signal:
   The CD signal should be tied low.
5. The RI signal:
   The RI signal may be tied high or low.

FLOW CONTROL FLOWCHARTS

Figure 7 through Figure 14 are flowcharts indicating a Host Controller application with the Flow Control operation. Figure 8 through Figure 10 are the flow control steps for an MCP2150 device, while Figure 12 through Figure 14 are the flow control steps for an MCP2155 device.

FIGURE 7: MCP2150 APPLICATION FLOW CHART

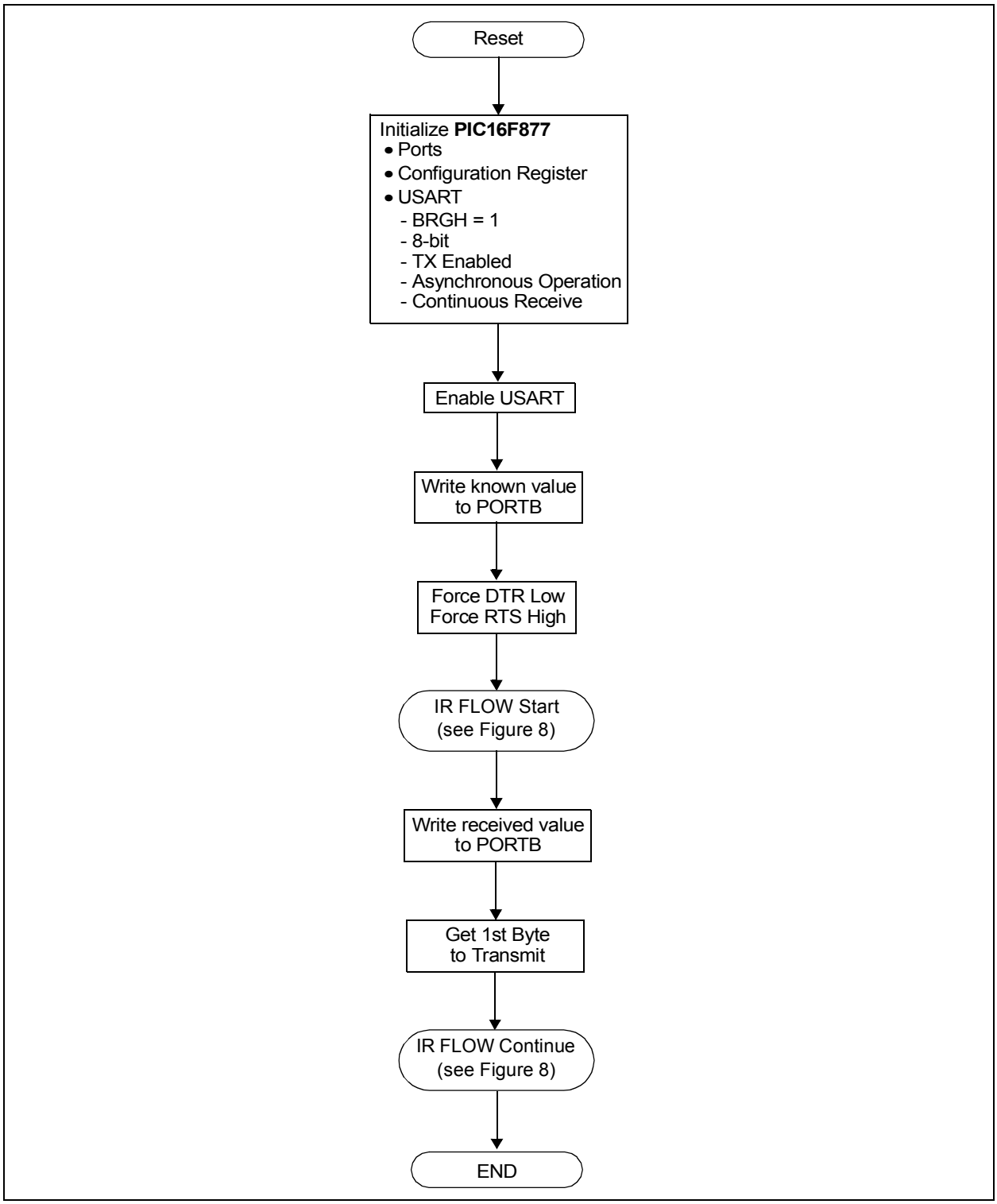


FIGURE 8: MCP2150 FLOW CONTROL FLOW CHART

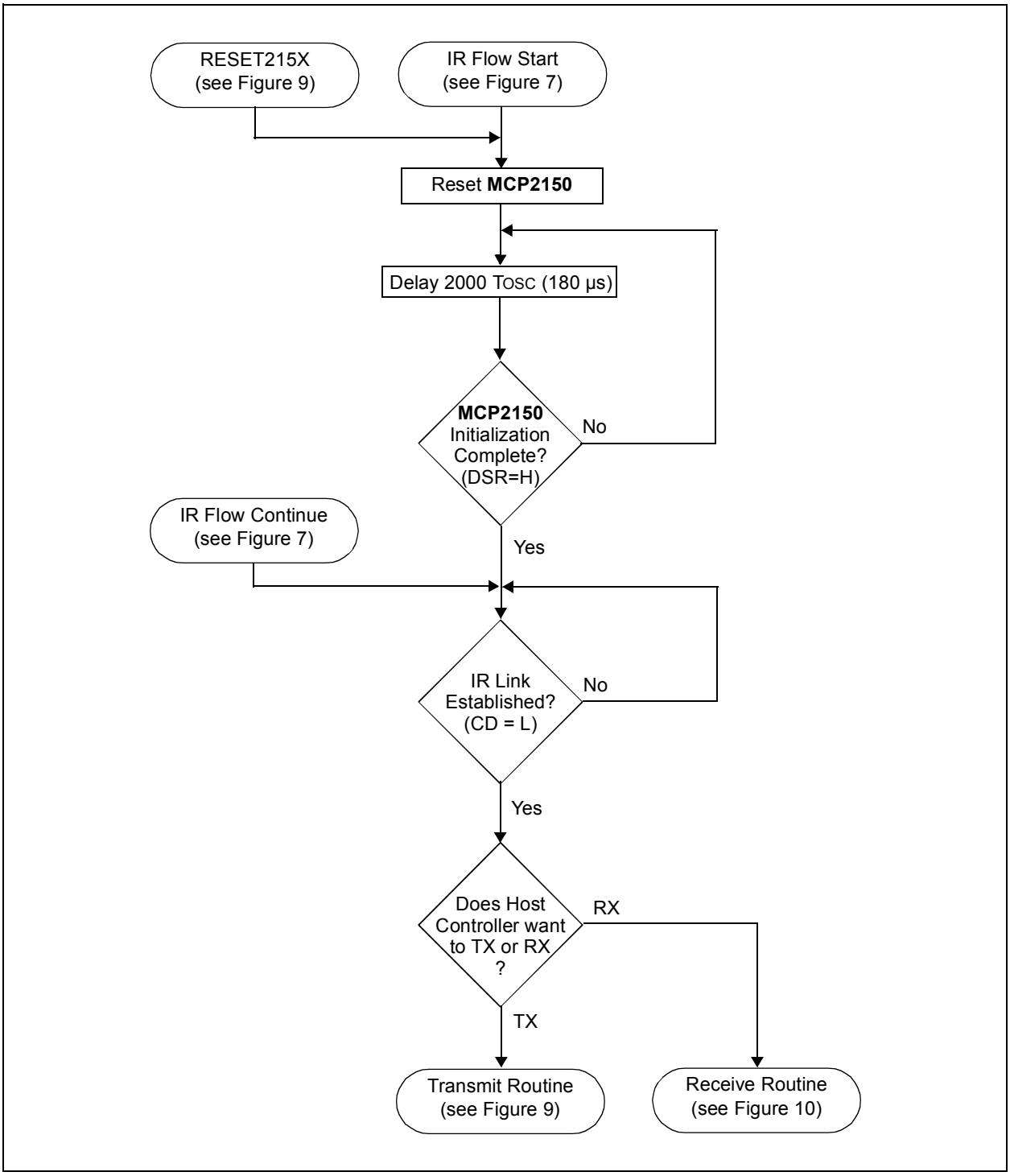


FIGURE 9: MCP2150 FLOW CONTROL FLOW CHART - TRANSMIT

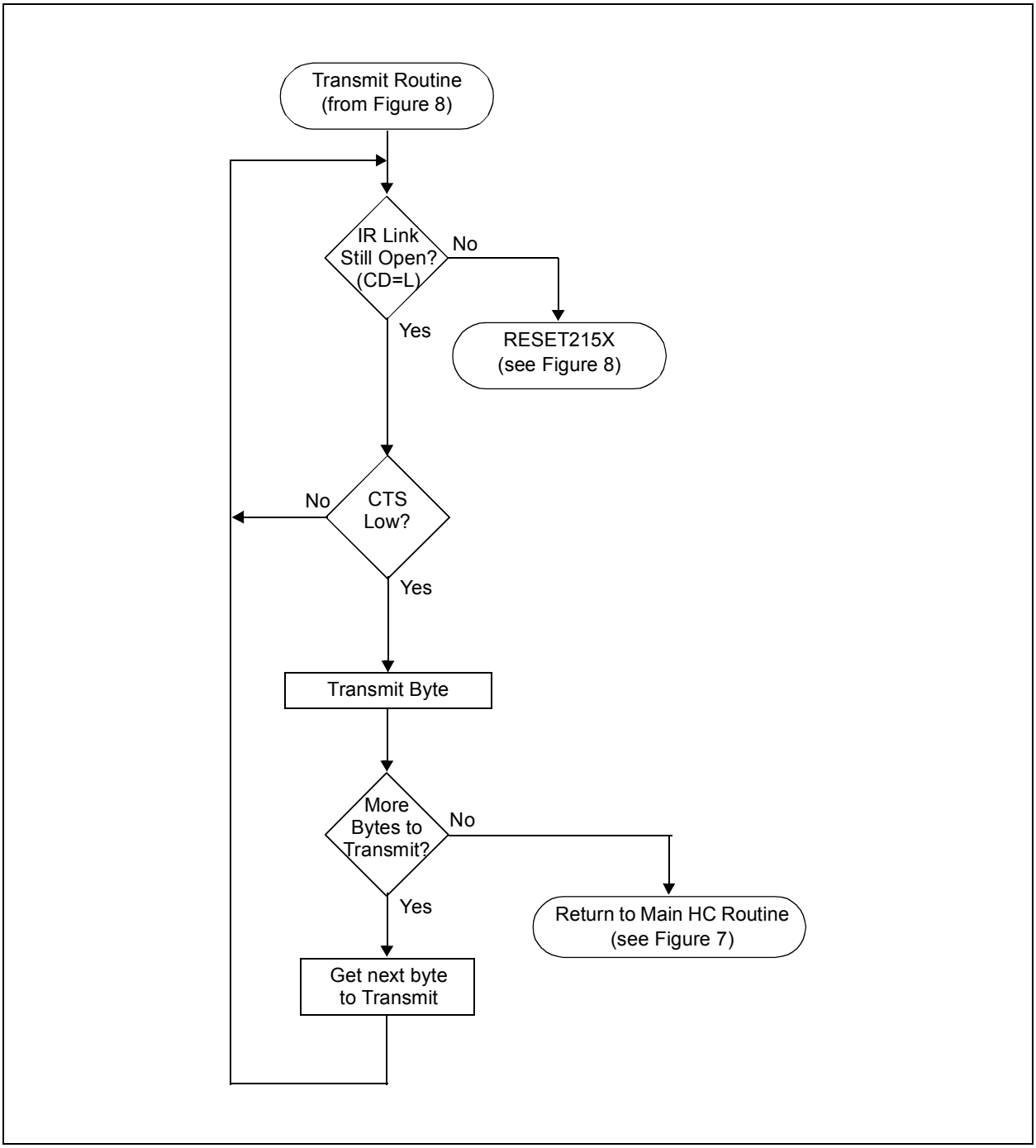


FIGURE 10: MCP2150 FLOW CONTROL FLOW CHART - RECEIVE

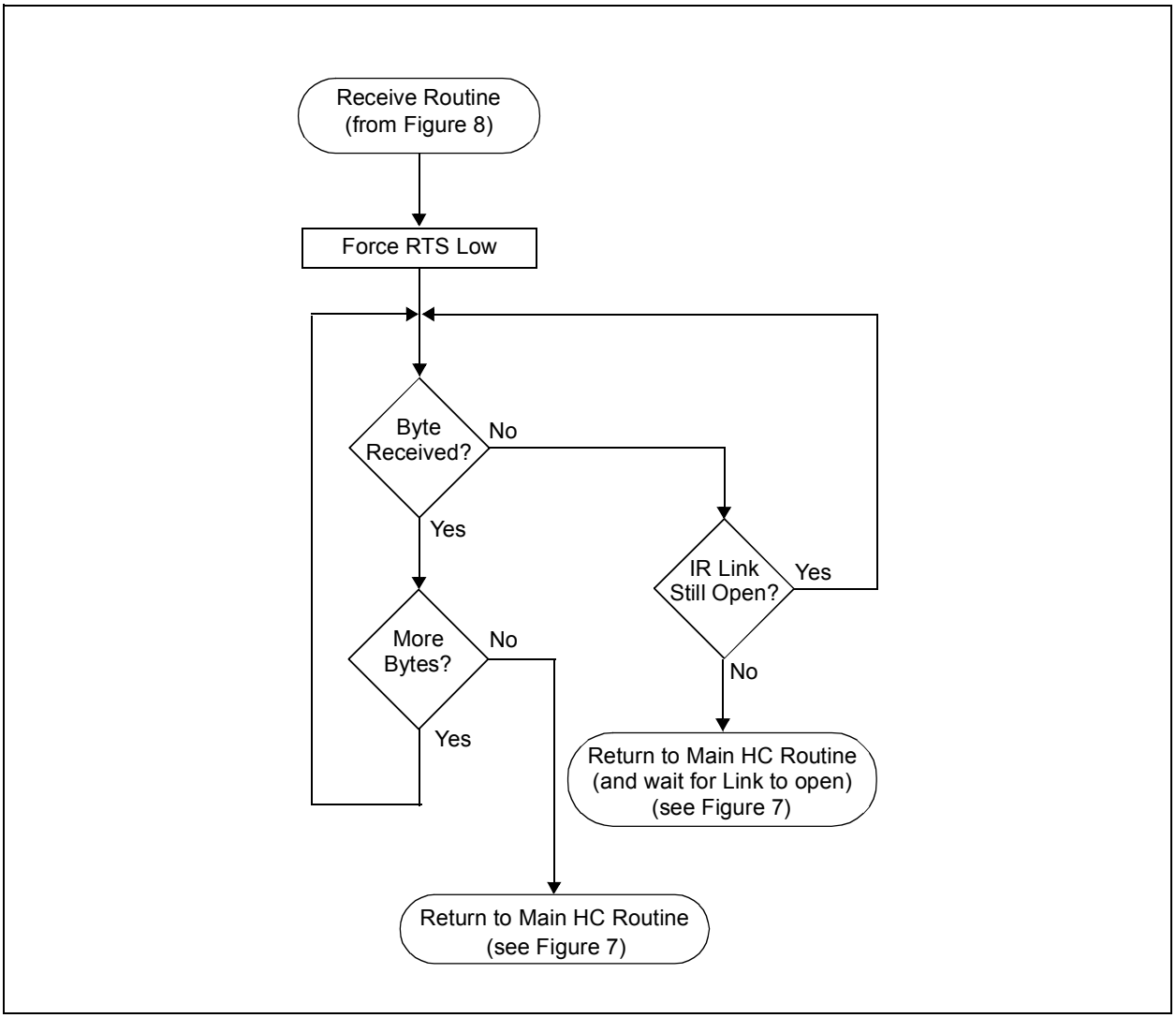


FIGURE 11: MCP2155 APPLICATION FLOW CHART

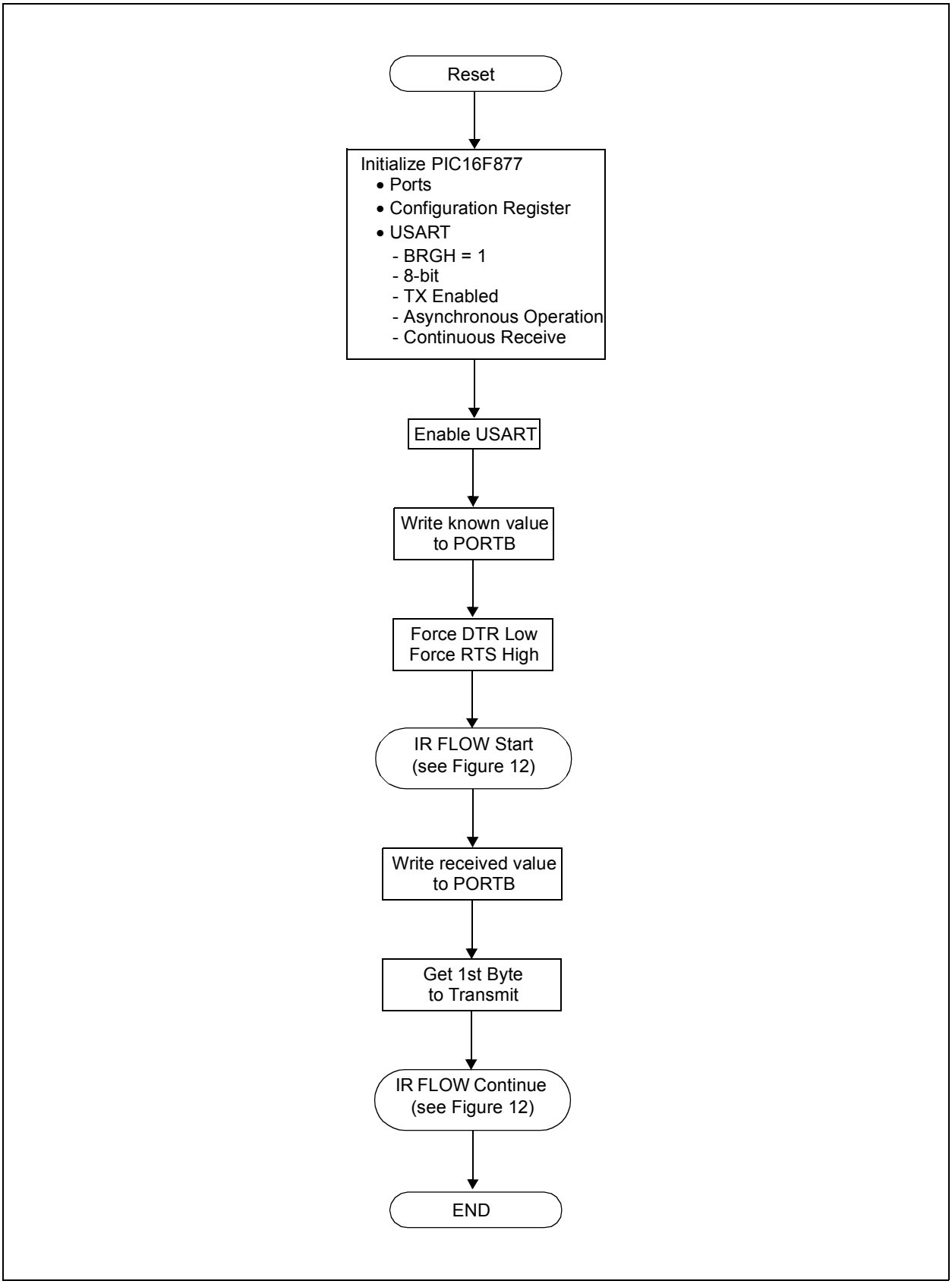


FIGURE 12: MCP2155 FLOW CONTROL FLOW CHART

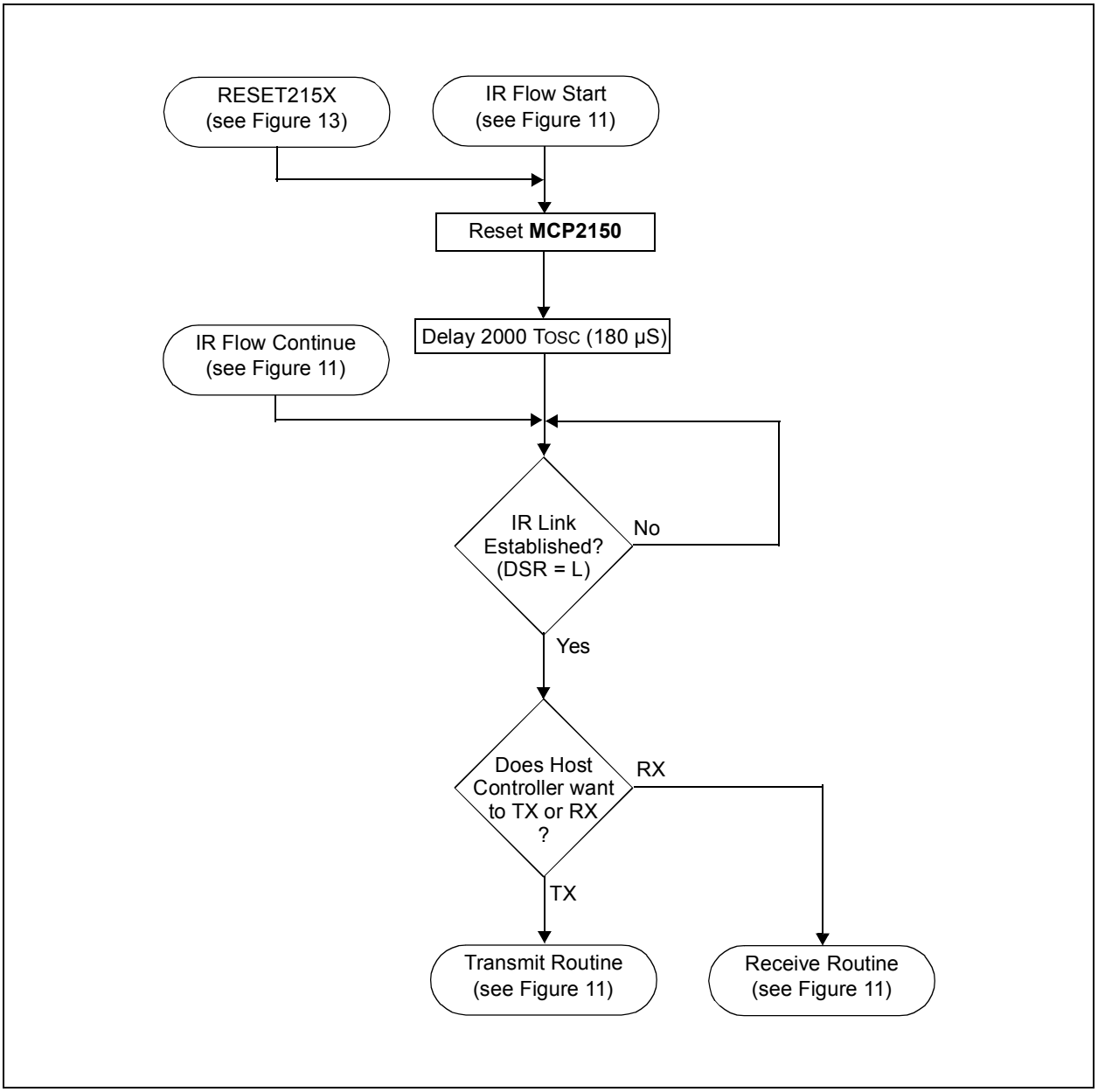


FIGURE 13: MCP2155 FLOW CONTROL FLOW CHART - TRANSMIT

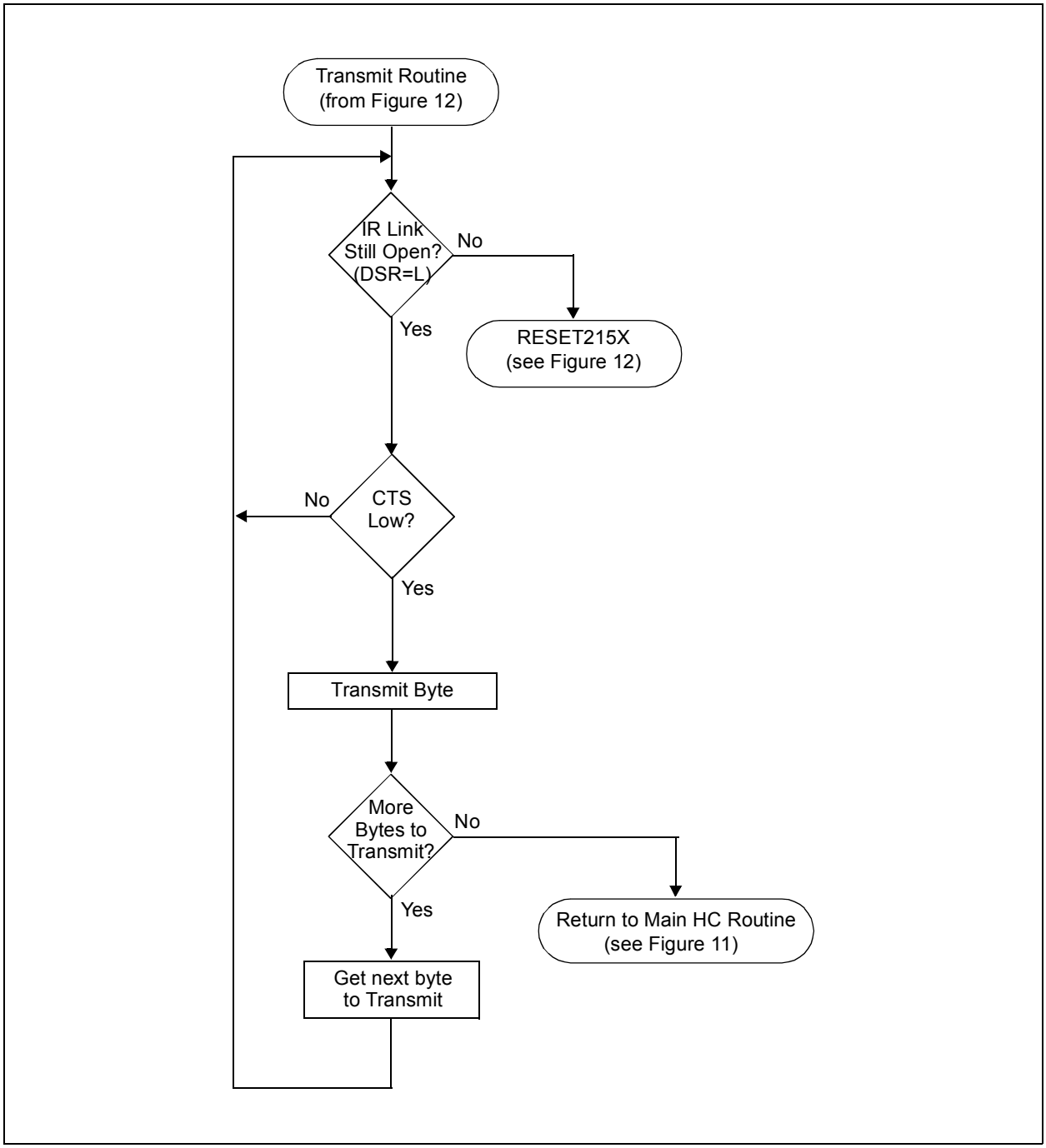
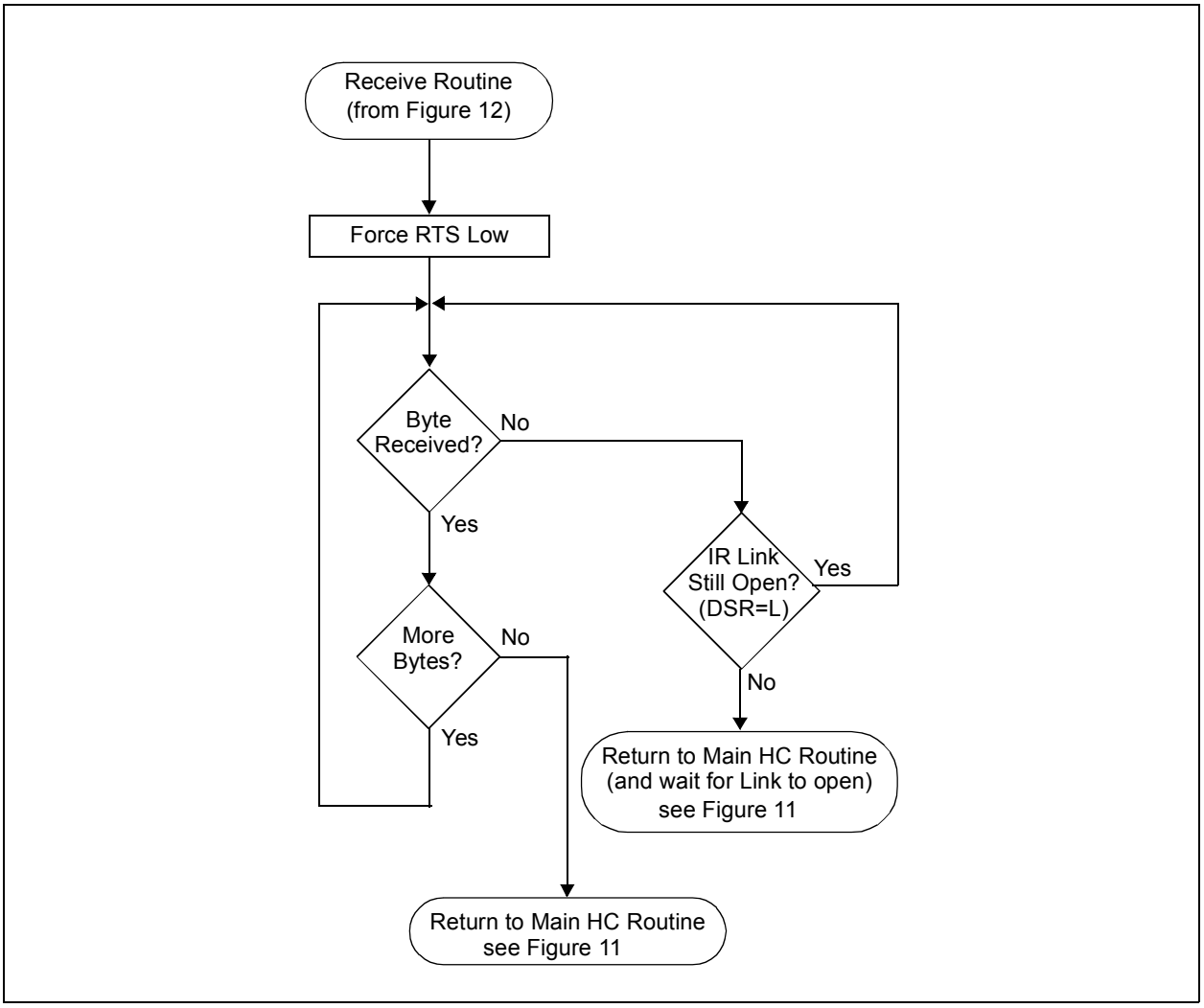


FIGURE 14: MCP2155 FLOW CONTROL FLOW CHART - RECEIVE



HOST UART WAVEFORMS

The following Host UART waveforms (Figure 15 through Figure 18) were generated using a PICmicro® connected to the MCP2150. The PICmicro USART was configured with a baud rate of 19200. The TX signal is driven by the Host Controller. The CTS signal is driven by the MCP215X device and is monitored by the Host Controller while data is being transmitted. The PICmicro program toggles an I/O pin called "Byte Strobe" for each byte that is transmitted on the PICmicro USART.

Note	**1:**	The Byte Strobe signal is used so that the number of bytes transmitted can easily be counted.
	2:	The "Byte Strobe" is not implemented in the application code shown in Appendix A.

Figure 15 illustrates that during the CTS low time, 24 bytes are transmitted, as is indicated by the "Byte Strobe". The time between marker Ax and marker Bx is shown at the bottom of the screen capture, shown as a 'Δ'.

FIGURE 15: ONE PACKET OF 24 BYTES TRANSMITTED

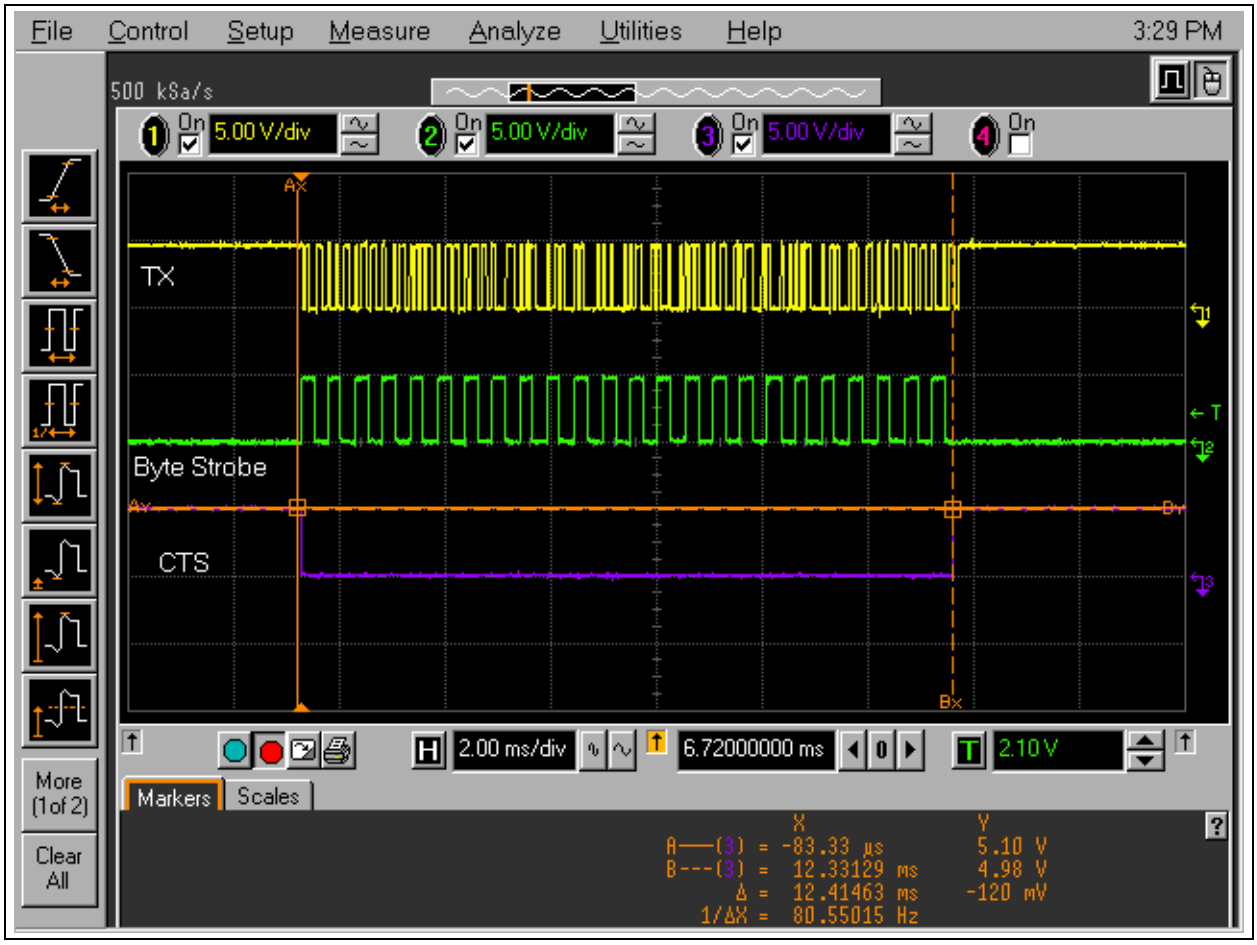


Figure 16 shows two 24-byte packets being sent to a PDA. The CTS signal rises near the end of the 24th byte. The delay between the two 24 byte data packets is dependent on the Primary device and the MCP215X.

The time between marker Ax and marker Bx is shown at the bottom of the screen capture as a 'Δ'.

FIGURE 16: TWO PACKETS OF 24 BYTES TRANSMITTED

Figure 17 shows the transmission of the first data byte (a hex value of 31h ('1')). Marker A and Marker B verify that the baud rate is 19200, shown at bottom of the screen capture.

FIGURE 17: FIRST BYTE TRANSMITTED 31H (LSB FIRST)

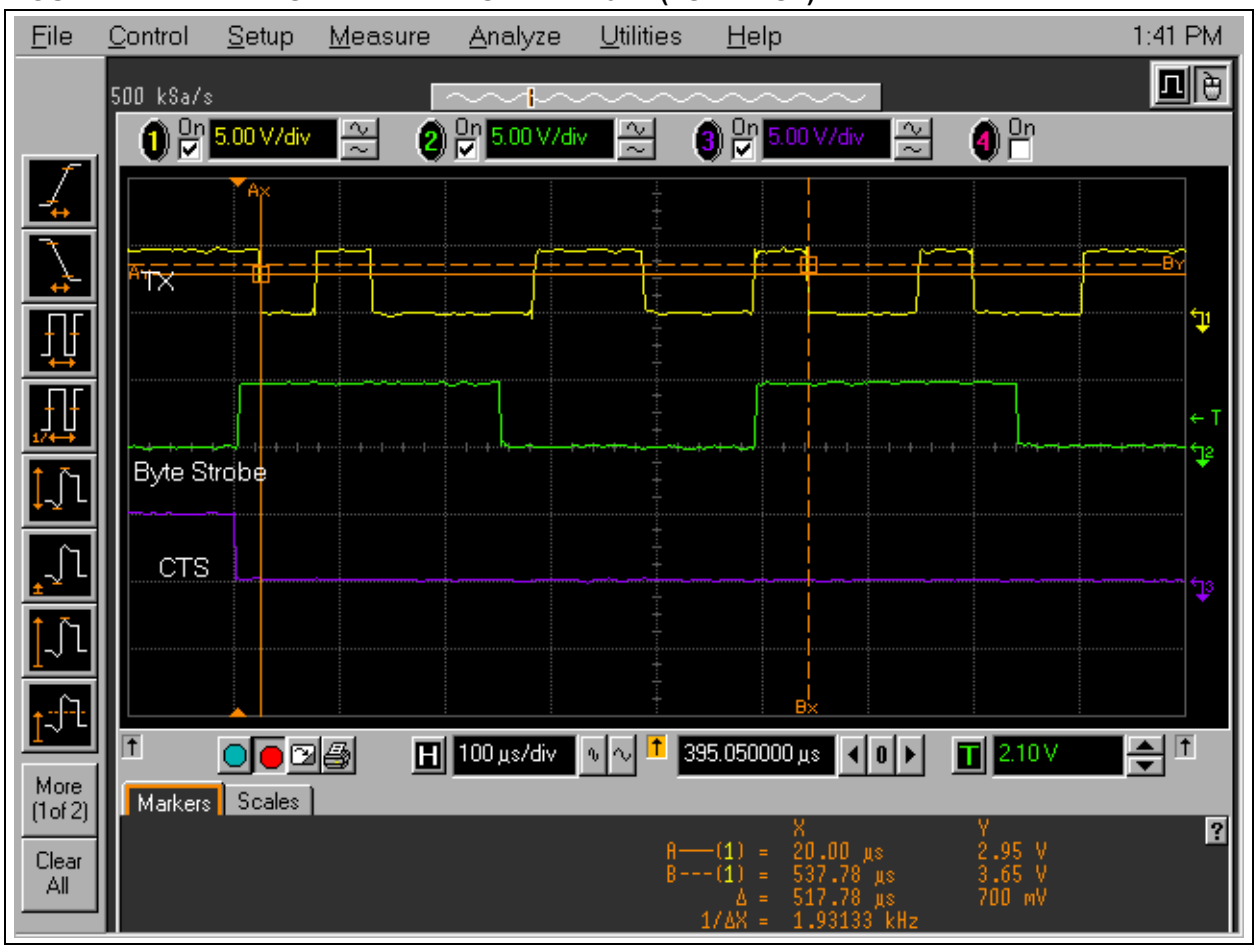
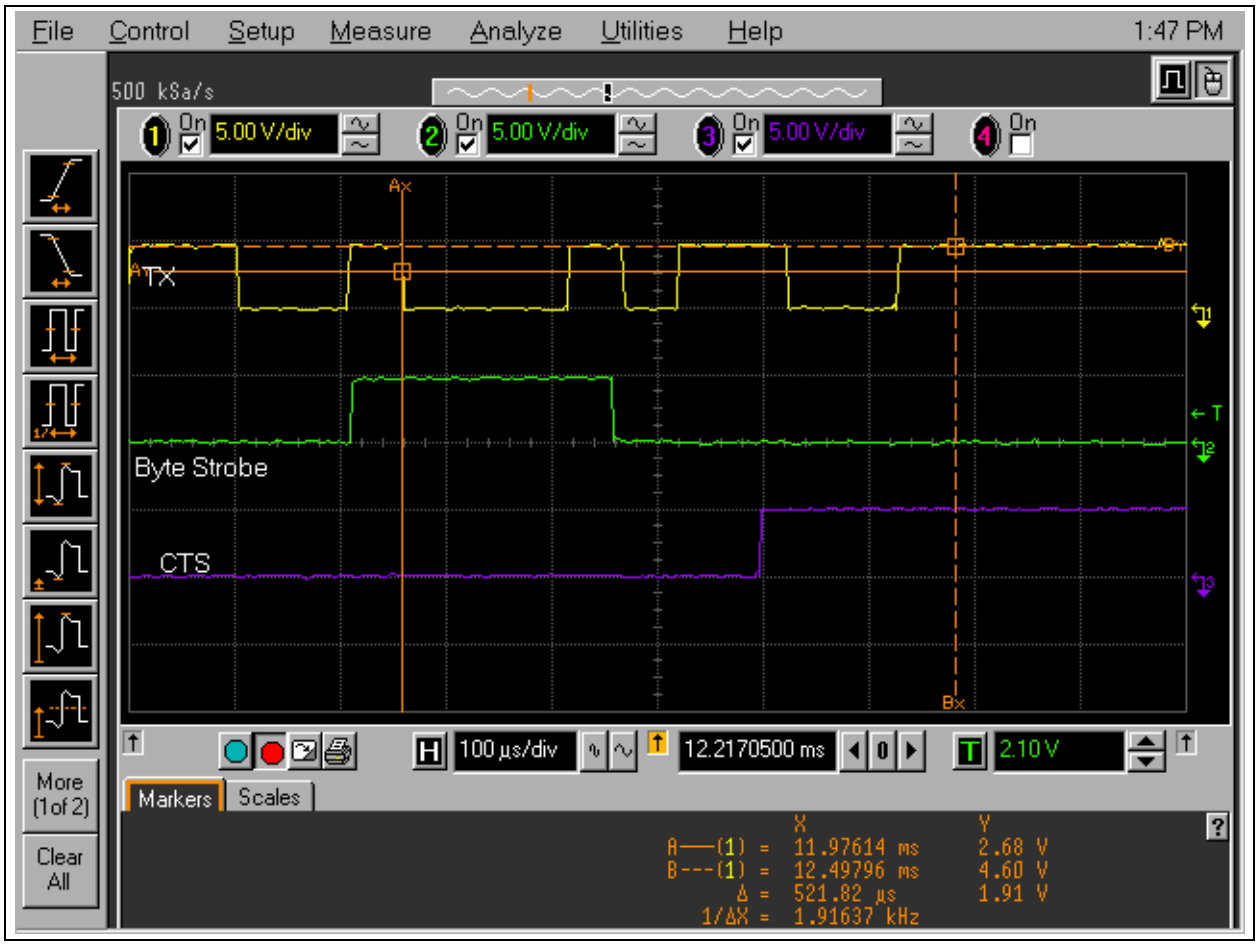


Figure 18 shows that during the transfer of the 24th byte, the CTS signal is driven high during the last byte transmitted and that no additional bytes are transferred after CTS is high. The MCP215X completes reception of the last byte transmitted by the Host Controller. The time between marker Ax and marker Bx is shown at the bottom of the screen capture as a 'Δ'.

FIGURE 18: LAST BYTE TRANSFERRED WHEN CTS WAS DETECTED LOW



OVERVIEW OF THE DEMO

This application example is intended to demonstrate the steps required to interface a MCP215X device with a PICmicro microcontroller, and to communicate with an IrDA Standard Primary device.

This application example was implemented with a PICDEM™ 2 Plus Demo Board, using a PIC16F877 as the Host Controller, and an MCP2150 Developer's Board. The program in the PIC16F877 monitors the Host UART signals and waits for an IR link to be established. Once the link is established, the PIC16F877 waits for a character to be received and then displays that character on PORTB (the LEDs of the PICDEM 2 Plus Demo Board). After a single character has been received, the PIC16F877 sends a character string to the MCP215X, which will be sent over the IR link to a Primary device.

The Primary device can be many devices, but the setup for two different devices will be described. These are:

- Palm PDA running a Terminal Emulation program called Online
- iPAQ PDA running PocketPC
  (PocketPC 2002 does not have the same setup procedure as version PocketPC)

Hardware Configuration

This system can be easily implemented using existing hardware boards available from Microchip Technology. These boards are:

- PICDEM 2 Plus Demo Board (DM163022)
- MCP2150 Developer's Board (in the MCP2120/ MCP2150 Developer's Kit, DM163008)

Figure 19 shows the 10 wire interconnection between a PICDEM 2 Plus Demo Board and the MCP2150 Developer's Board. The RESET pin of the MCP2150 is controlled by the PICDEM 2 Plus Demo Board, and should be disconnected from the MCP2150 Developer's Board circuitry.

Host Controller Operation

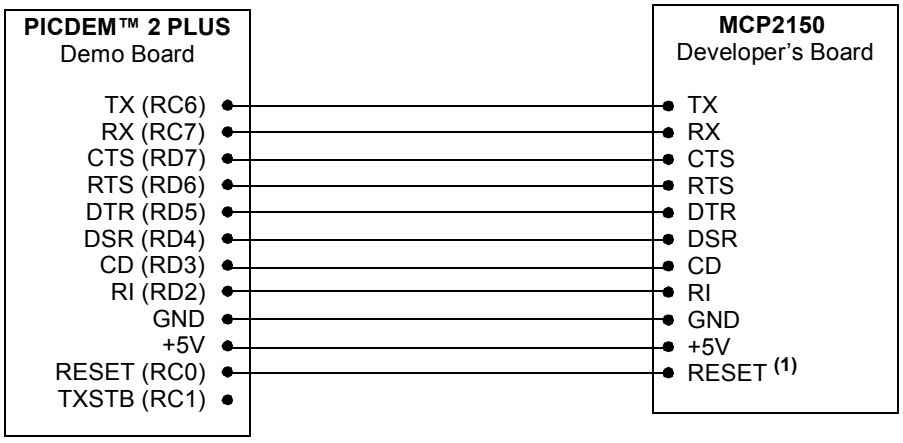
The PIC16F877 firmware in Appendix A performs the following operations.

1. The firmware initializes the PIC16F877 and MCP215X.
2. The PIC16F877 waits for a link to be established.

Note:	When the PDA sends a character, a link is established.

3. The PIC16F877 drives the RTS signal low.
4. The MCP215X may transfer the received character to the Host Controller.
5. This value is moved to PORTB to be viewed on the LEDs.
6. The PIC16F877 then transmits a string of data to the Primary device (following the Flow Control).
7. Once the string has been completely transmitted, the PIC16F877 enters an infinite loop to terminate operation.

FIGURE 19: PICDEM 2 PLUS TO MCP2150 DEVELOPER'S BOARD CONNECTIONS



Note 1: The MCP2150 RESET pin must NOT be connected to the MCP2150 Developer's Board circuitry (either lift the pin or use sockets with pin 4 removed so the RESET pin is open. However, the RESET pin **can** be connected to/from the PICDEM 2 Demo Board without drive conflicts).

THE PRIMARY DEVICE

The Primary device must be configured to communicate over it's IR port using IrCOMM. Many applications that use the IR port use a different application protocol (such as IrOBEX). These are different languages, and a device "talking" IrOBEX will not communicate with a device "talking" IrCOMM. The MCP215X devices require the Primary device to communicate using IrCOMM (9-wire "cooked" protocol).

The configuration for a Palm OS® Operating System - based system, and a PocketPC O.S.-based system, will be shown.

The Palm OS™ does not embed an application which can be configured to utilize the IR port in IrCOMM, so a third party application called Online (a simple Terminal emulation program) is used for this.

The PocketPC O.S. comes with a communication application that can be configured to support IrCOMM.

Note:	Each version of an O.S. or software application may have changes that will cause the setup steps to vary from what is documented. If you have setup problems, you may want to load the versions used in this application note onto your PDA.

Table 3 shows the system setup that was used to create the configuration steps for the Palm/Online operation.

TABLE 3: PALM SYSTEM SETUP

Item	Product	Comment
Manufacturer	Palm	
Model	m105	
O.S. Version	3.5	
3rd Party Terminal Emulation Program	Online	www.markspace.com Version 1.4.1

Table 4 shows the system setup that was used to create the configuration steps for the iPAQ operation.

TABLE 4: POCKETPC SYSTEM SETUP

Item	Product	Comment
Manufacturer	Compaq	
Model	3650	
O.S. Version	WinCE 3.0.9348 Build 9616	PocketPC devices
3rd Party Terminal Emulation Program	N.A.	Not Required, comes standard with communications program

USING A PALM OS® PDA AS A PRIMARY DEVICE

The Palm OS PDA does not come with an embedded application program that allows connection to the IR port with the IrCOMM application layer. A 3rd party program called Online, available from www.markspace.com, is used in this example.

Note 1:	Each version of an O.S. or software application may have changes that will cause the setup steps to vary from what is documented. If you have setup problems, you may want to load the documented versions of the programs/O.S. onto your PDA.
2:	The version of Online used allowed a 30-day free demonstration period.

Configuring the Online program Settings

1. Install the program Online (Version 1.4.1) on the Palm PDA.
   - Online.prc loaded via Hotsync or “Beaming”.
2. The program will probably be installed in the “Unfilled” folder. This is found in the pull-down menu at the top right.
3. After selecting the “Unfilled” folder, there will be an icon called Online.
4. Select Online (tap on Icon).

Note:	The 1st time the program is run, a welcome window will be displayed. Select “OK”.

   a) Select Demo at the bottom of the Window.

Note:	The 1st time the program is run, a please register window will be displayed. Select “OK”.

5. At the bottom left of the screen, below the Home icon (and left of the Graffiti area), is the pull-down menus icon. Tap on the icon and this opens the Menu pull downs.
   a) Under the Options menu, Select “Communications”.
   b) For Method, select “Serial”.
   c) For Port, select “IrCOMM”.

Note:	Depending on the Palm device and the version of the Palm OS, the IR selection may be different, such as needing to select “Infrared”.

   d) For Baud, select “115200”.

Note:	Standard IR baud rates are available. You may select any baud rate supported by the MCP215X device. The selection of 9600 baud is useful with the MCP2120 Developer’s Board to act as an IR data sniffer.

   e) For Data Bits, Select “8”.
   f) For Parity, select “N”.
   g) For Stop Bits, select “1”.
   h) Uncheck RTS/CTS.
   i) Uncheck XON/XOFF.

Note:	Some Palm devices/OS versions may replace steps h and i with a handshake pull down menu. For Handshake, select “None”.

6. The Online program is now configured for use.

USING THE PROGRAM ONLINE AND THE MCP215X DEMO OPERATION

1. Select (tap on Icon) Online to open the program.
   - The program will probably be installed in the "Unfilled" folder. This is found in the pull-down on the top right.
2. Select the "Demo" button (at bottom middle).
3. Select the "On" button (at bottom left), which will cause the "On" button to appear in reverse video (black block, white text).
4. Select either the "abc" button for a keyboard, or the "123" button for a number pad (below assumes that the "123" button is selected).
5. On the keyboard, type in a single character (such as the number "3").
6. Point the Palm device towards the MCP2150 board's IR transceivers. The CD LED of the MCP2150 board is NOT "On".
7. Select the "Done" button at the bottom left of the keyboard window.
   - The CD LED is turned on (indicating that the link has been established).
   - The Data (ASCII 3 $\rightarrow$ 33h) is received by the MCP215X and written to PORTB (the LEDs).
8. The PIC16F877 now sends the stored Character String (called MENU) to the MCP215X, which sends it in packets to the PDA.
9. To disconnect the link, select the "On" button (at bottom left). The "On" button will now appear in regular video (white block, black text).
   - When the session is closed, the CD light on the MCP2150 Demo Board will go off.
10. Select the "Home" Icon (bottom left) to allow the Online program to be restarted. The "Online" icon should appear on the screen.

USING A PocketPC O.S. PDA (iPAQ) AS A PRIMARY DEVICE

The PocketPC O.S. comes standard with a communication application program that allows connection to the IR port with the IrCOMM application layer. This program is located in the Program->Connections folder.

Configuring PocketPC Modem Settings

Note:	Each version of an O.S. may have changes that will cause the setup steps to vary from what is documented. If you have setup problems, you may want to load the O.S. used in this application note (WinCE 3.0.9348 Build 9616) onto your PDA.

Steps:

1. Find the connection folder and select it.
2. If no connection is setup, create a connection.
   a) Select Modem at bottom of screen.
   b) Select “New Connection...” in window.
   c) Type in Name for connection (such as “Ir Test”).
   d) For Modem, select “Generic IrDA Modem”.
   e) Select Baud Rate (115200).
   f) Select Advanced: (8 data bits, No Parity, 1 Stop bit, Hardware Flow Control).

      Under Terminal:

      - Check the box
        - Use Terminal before connecting.
      - Uncheck the box
        - Use Terminal after connecting.
      - Check the box
        - Enter dialing commands manually.

   g) Select “ok” in top right corner.
   h) Select Next.
   i) Do the following:
      - Uncheck the box
        - Cancel calls if not connected in “xxx” seconds.
      - Uncheck the box
        - Wait for the dial tone before dialing.
   j) Select Finish.

USING THE PocketPC MODEM PROGRAM AND THE MCP215X DEMO OPERATION

Now the unit should be ready to make a connection.

1. In the Connections folder, find the icon with the connection name from step 2c in the Configuring PocketPC Modem Settings.
2. Select that icon.
3. Type in:
   a) User Name.
   b) Password.
   c) Leave Domain blank.
   d) Check the box - “Save Password”.
   e) Dial From: Select “Work”.
4. Select Connect.

Note:	This does not cause an IR link to be established.

5. In the bottom right, select the keyboard (the keyboard will come up).
6. Point the iPAQ device towards the MCP2150 boards IR transceivers. Notice on the MCP2150 Board, that the CD LED is NOT “On”.
7. Select any character (send the number “3”). After a delay:
   - The CD LED is turned on, indicating that the IR link has been established.
   - The Data (3) is received by the MCP215X and written to PORTB (the LEDs).
8. The PIC16F877 now sends the stored Character String (called MENU) to the MCP215X, which sends it in packets to the PDA.
9. To disconnect the link, select “OK” in top right corner to close the iPAQ window.
   - When the session is closed, the CD light on the MCP2150 Demo Board will go off.

REFERENCES

The IrDA Standards download page can be found at:

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

Manufacturers of 3rd Party Products are shown in Table 5.

SUMMARY

The MCP215X Host UART interface is easy to implement, with a small overhead compared to non-flow controlled UARTs. This makes the MCP215X well suited for implementing IrDA solutions in consumer, industrial, automotive, and telecommunications applications.

TABLE 5: 3RD PARTY PRODUCTS

Product		Company	Company Web Site Address
Name	**Description**		
Online	Terminal Emulation Program for Palm O.S.	Mark/Space	www.markspace.com
Palm OS	Palm PDA Operating System	Palm	www.palm.com
O.S. (PocketPC)	Microsoft PDA Operating System (PocketPC)	Microsoft	www.microsoft.com
PDA	Palm OS PDA	Palm	www.palm.com
PDA	Palm OS PDA	Handspring	www.handspring.com
PDA	PocketPC O.S. PDA (iPAQ and Jornada)	HP	www.hp.com

APPENDIX A: PIC16F877 SOURCE CODE

EXAMPLE A-1: PIC16F877 CODE TO INTERFACE TO THE MCP215X

```
    LIST    C=132
    include P16F877.inc
        ERRORLEVEL -302
;
;*****************************************************************************
;
; SELECT THE MCP215x Device to interface (Host UART Signals) to.
;    This code supports the MCP2155 and MCP2150.  The conditional assembly is
;    defined here.  The allowable choices in this version are 50h or 55h
;
;MCP215X       equ     H'55'   ; assemble for MCP2155
MCP215X       equ     H'50'   ; assemble for MCP2150
;
; The use of these Assembler Directives is to verify that a valid target
; product was selected for the Firmware generation. If not, an ERROR MESSAGE
; will be generated.
;
        if ( MCP215X != H'55' && MCP215X != H'50' )
           error "MCP215x Device Selected NOT VALID"
        endif
;
    if MCP215X==H'50'           ;
           messg "MCP2150 has been Selected"
    endif
;
    if MCP215X==H'55'           ;
           messg "MCP2155 has been Selected"
    endif
;
;*****************************************************************************
;
; Revision History
;  1.0    06/24/02  Initial Release
;
;*****************************************************************************
;
; MCP2150 Developer's Board with PICDEM-2 Demo Board Demo
;
;    PIC16F877 code to interface to MCP215x Controller
;    Program resets MCP215x and waits for "IR connection"
;    Once a connection is established, the Host controller
;    monitors the CTS signal (for a Low) to sends a stream
;    of bytes.
```

Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 2

```
;
;    After the Table has completed being transmitted, the
;    program "stops" (that is the program loops forever)
;
; NOTE: The MCP2150 Developer's Board requires that the RESET pin of the
;       MCP215x device be disconnected from the MCP2150 Developer Board
;       circuitry, and connected to the specified I/O pin of the PIC16F877
;       device on the PICDEM-2 Demo Board
;
; PICDEM-2 Requirements
;    Device:           PIC16F877
;    Clock Frequency:  20.00 MHz
;    UART:             User Defined Baud
;
; MCP215x Requirements
;    Clock Frequency:  11.0952 MHz
;
; PIC16F877 PORT Functions
;    PORTA
;      Function         ---   ---   NA    NA    NA    NA    NA    HB
;      TRIS Direction   ---   ---    O     O     O     O     O     O
;      Initial value    ---   ---    H     H     H     H     H     H
;
;    PORTB
;      Function         LED7  LED6  LED5  LED4  LED3  LED2  LED1  LED0
;      TRIS Direction    O     O     O     O     O     O     O     O
;      Initial value     H     H     H     H     H     H     H     H
;
;    PORTC
;      Function         RX    TX    NA    NA    NA    NA    NA   RST215X
;      TRIS Direction    I     I     O     O     O     O     O     O
;      Initial value    ---   ---    H     H     H     H     H     H
;
;*** PORTD (For MCP2150)
;***   Function         CTS   RTS   DTR   DSR   CD    RI    NA    NA
;***   TRIS Direction    I     O     O     I     I     I     I     I
;***   Initial value    ---    H     H    ---   ---   ---   ---   ---
;
;*** PORTD (For MCP2155)
;***   Function         CTS   RTS   DTR   DSR   CD    RI    NA    NA
;***   TRIS Direction    I     O     O     I     O     O     I     I
;***   Initial value    ---    H     H    ---    H     H    ---   ---
;
;    PORTE
;      Function         ---   ---   ---   ---   ---   NA    NA    NA
;      TRIS Direction   ---   ---   ---   ---   ---    O     O     O
;      Initial value    ---   ---   ---   ---   ---    H     H     H
;
;*****************************************************************************


#define  reset  H'00'           ; Reset vector
;*****************************************************************************
;       Configuration Bits
        __CONFIG _CP_OFF & _PWRTE_ON & _HS_OSC & _WDT_OFF
        __IDLOCS H'0010'
```


Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 3

```
;*************************************************************************
;       PORT Bits
;
#define rxd     PORTC, 7        ; input,  serial data from MCP215x
#define txd     PORTC, 6        ; output  UART overrides TRIS bit,
                                ;         serial data to MCP215x
#define cts     PORTD, 7        ; input,  MCP215x is ready to receive data
#define rts     PORTD, 6        ; output, PIC16F877 (Host Controller) is ready
                                ;         to receive data. At RESET,
                                ;         Low for pgm mode, High for normal
#define dtr     PORTD, 5        ; output, force high or low (LOW). At RESET,
                                ;         High for pgm mode, Low for normal
#define dsr     PORTD, 4        ; input,  Indicates MCP2150 has completed
                                ;         Reset, or
                                ;         Indicates MCP2155 has established
                                ;         a valid link,
                                ;         high for no link, low for link
#define cd      PORTD, 3        ; input,  Indicates MCP2150 has established
                                ;         a valid link,
                                ;         high for link, low for no link; or
                                ; output, The MCP2155 communicates this value
                                ;         is to the Primary Device.
                                ;         For this application, this signal
                                ;         (CD) can be static
#define ri      PORTD, 2        ; input,  MCP2150 - Driven high
                                ; output, MCP2155 - This value is communicated
                                ;         to the Primary Device.
                                ;         For this application, this signal
                                ;         (RI) can be static
#define rst215x PORTC,0         ; output, used to reset the MCP2155
                                ;         high for normal operation, low to
                                ;         RESET device
;
ddra    equ     B'00000000'     ; Data Direction for porta
;
ddrb    equ     B'00000000'     ; portb is an output port
;
ddrc    equ     B'11000000'     ; Data Direction for portc
;***
;*** Conditional assembly on PORTD Data Direction values
;***
    if MCP215X==H'50'           ;
ddrd    equ     B'10011111'     ; Data Direction for portd
    endif
;
    if MCP215X==H'55'           ;
ddrd    equ     B'10010011'     ; Data Direction for portd
    endif
;
;
;
ddre    equ     B'00000000'     ; porte is an output port
;
cfgopt  equ     B'11001000'     ; option reg setup
;
```

Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 4

```
;*******************************************************************************
;
; Additional Conditional Assembly Flags
;
;*******************************************************************************
;
ICD     EQU     0        ; When ICD is TRUE, Address 0x00
                         ;   must be a NOP and RB7:RB6 are use
                         ;   by the ICD module (override TRIS
                         ;   settings).
;
;*******************************************************************************
;       Constants
;
;
; Host UART Data Rate/BRG Value (BRGH = 1)
;                  SPBRG Value
;    Baud Rate    @ 20MHz
;    9600
;    19200
;    57600
;   115200
;
;                     SPBRG Value
B9600at20MHz    EQU    D'129'
B19200at20MHz   EQU    D'64'
B57600at20MHz   EQU    D'21'
B115200at20MHz  EQU    D'10'
;
;*******************************************************************************
;       Registers
;
    cblock    H'20'
        delreg          ; register for timing delays & scratchpad
        MENUCNTR        ; Pointer to the Menu character to send
        MENUBYTES       ; This is the # of bytes in the MENU
        hostdata        ; Host Data to Transmit
        BYTERX          ; Received Byte on UART
    endc
;
;*******************************************************************************
        org     H'00'         ; use 00h as reset vector
    if ICD
        NOP                   ; Use of the ICD requires the first
    endif                     ;   instruction to be a NOP

        goto    START
;
;
```


Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 5

```
;*************************************************************************
;       Start Routine
;       Initialization is done here
;       (Ports, Option Register, and UART).
;       Option Register has:
;            RBPU disabled,
;            RB0 on Rising Edge,
;            TMR0 Clock Source internal
;            T0CKI inc on L-to-H
;            Prescaler assigned to WDT
;            WDT = 1:1
;*************************************************************************
START   clrf    STATUS          ; Bank 0
        movlw   0xFF            ; Force PORTs to display High when configured
        movwf   PORTA           ;    as Output
        movwf   PORTB           ;
        movwf   PORTC           ;
        movwf   PORTD           ;
        movwf   PORTE           ;

        bsf     STATUS, RP0     ; Bank 1
        movlw   ddra
        movwf   TRISA           ; Configure PORTA
        movlw   ddrb
        movwf   TRISB           ; Configure PORTB
        movlw   ddrc
        movwf   TRISC           ; Configure PORTC
        movlw   ddrd
        movwf   TRISD           ; Configure PORTD
        movlw   ddre
        movwf   TRISE           ; Configure PORTE
;
        movlw   cfgopt          ; setup option reg
        movwf   OPTION_REG
;
;  Initialize UART
;      BRGH = 1
;      8-bit
;      TX Enabled
;      Async. Operation
;      Continuous receive
;  Enable UART
;  Write value (0xFF) to PORTB
;     (ICD uses RB7:RB6, so with ICD 0x3F will be displayed)
;
        movlw   0x24            ; BRGH = 1, 8-bit, TX Enabled, Async.
        movwf   TXSTA           ;
        movlw   B19200at20MHz   ;
        movwf   SPBRG           ;
        clrf    STATUS          ; Bank 0
        movlw   0x90            ; Enable serial port, continuous receive
        movwf   RCSTA           ;
;
        clrf    PORTB           ; clear outputs (Display on LEDs)
;
```

Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 6

```
; Initialize MCP215x Flow Control signals,
; Reset MCP215x,
; Delay for 1us,
; then release Reset
;
RESET215X
        bcf     dtr             ; dtr low is the normal mode for the MCP215x

        bcf     rst215x         ; Reset the MCP215x
        nop                     ; Delay to ensure MCP215x RESET pin is
        nop                     ;    deteceted (driven) low
        nop
        nop
        nop
        bsf     rst215x         ; Release the MCP215x
;
; MCP215x requires 2000 Tosc (at 11.0592MHz = 180 us)
; delay before the device initialization should be
; complete
;
        movlw   H'FF'           ;
        call    DELAY
;
; The following delay is done only on the MCP2150, since the MCP2150 has a
; signal (DSR) which is used to indicate if the MCP2150 has completed RESET.
; There is no corresponding signal on the MCP2155.
;
    if MCP215X==H'50'           ; Conditional Assemble for MCP2150
;
; Has MCP2150 completed initialization?,
; if not continue to wait
;
WAIT2150                        ; Now test the state of the DSR pin
        btfss   dsr
        goto    WAIT2150        ; NO, wait more time
        goto    MAIN            ; YES, continue
;
    endif                       ; End of Conditional Assemble for MCP2150

;*****************************************************************************
```


Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 7

```
;  Main Routine - MCP215x Has completed initialization
;
; Wait for MCP215x to establish a link.
; Indicate to MCP215x to Send byte that established link
; Wait for byte to be received by PIC16F877
; (while waiting, test to ensure link is still present)
; Read Byte and display on PORTB
; Call Subroutine which Transmits entire Table of Data
; Then Loop forever.
;
;*****************************************************************************
MAIN

    if MCP215X==H'50'           ; Conditional Assemble for MCP2150
WAITCD  btfsc   cd              ; Has the MCP2150 made a link?
        goto    WAITCD          ; NO, wait for a link to be established
    endif                       ; End of Conditional Assemble for MCP2150
;
    if MCP215X==H'55'           ; Conditional Assemble for MCP2155
WAITDSR btfsc   dsr             ; Has the MCP2155 made a link?
        goto    WAITDSR         ; NO, wait for a link to be established
        bcf     cd              ;*** Light the CD LED to show that DSR was low
    endif                       ; End of Conditional Assemble for MCP2155
;
        bcf     rts             ; YES, Host can receive the "Dummy" byte
RXWAIT1
        btfsc   PIR1, RCIF      ; Has a byte been received yet?
        goto    GOTBYTE1        ; YES
;
    if MCP215X==H'50'           ; Conditional Assemble for MCP2150
        btfsc   cd              ; NO, so test if MCP2150 link still active?
    endif                       ; End of Conditional Assemble for MCP2150
;
    if MCP215X==H'55'           ; Conditional Assemble for MCP2155
        btfsc   dsr             ; NO, so test if MCP2155 link still active?
    endif                       ; End of Conditional Assemble for MCP2155
;
        goto    MAIN            ;     NO, Link was lost, so start over
        goto    RXWAIT1         ;     YES, Have not received a byte yet
GOTBYTE1
        movf    RCREG, W        ; Get byte into W register and this clears
                                ;   the RCIF flag. Link is established,
        movwf   PORTB           ;   display on PORTB then send bytes
;
        call    SENDDATA        ; Send the MENU character string
;
LP4EVER
    if MCP215X==H'55'           ; Conditional Assemble for MCP2155
        btfsc   dsr             ; Is MCP2155 link still active?
        bsf     cd              ; NO, Turn of CD LED
                                ; YES, Do not change state of CD LED
    endif                       ; End of Conditional Assemble for MCP2155
        goto    LP4EVER         ; Program Completed transmission of
                                ;   characters, wait here for system
                                ;   reset
;
```

Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 8

```
; Send String (MENU) routine
;
; This routine Transmits the String (MENU) Data to the MCP215x
; The First byte of the String (Menu) is the length of the Data
;        MENUCNTR is pointer into Table MENU to get the Table lookup data
;        MENUBYTES contains the number of bytes of the String (MENU) still to
;        be transmitted. hostdata contains the value returned from MENU, to
;        be transmitted
;
; Determine if PIC16F877 can transmit UART data (monitor CTS signal)
; After Calling Serial Send Routine, decrement the number of bytes to send
; Test to see if still more bytes to send.
;
;   CTS Window 12ms.
;      Baud Rate     Max Bytes Transferred
;        9600            12
;       19200            23
;       57600            67    Exceeds MCP215x Buffer Size of 64 Bytes -
;                              Ensure only 64 bytes are sent during MCP215x
;                              Transmit Window
;      115200           138    Exceeds MCP215x Buffer Size of 64 Bytes -
;                              Ensure only 64 bytes are sent during MCP215x
;                              Transmit Window
;
;*****************************************************************************
;
SENDDATA
        clrf    MENUCNTR        ; MENU Counter = 0
        call    MENU            ; Get next byte of data from the MENU Data Table
        movwf   MENUBYTES       ; This is the # of bytes in the MENU
                                ;   (Menu size must be > 1)
MENULOOP1
        incf    MENUCNTR, F     ; Point to next location in the MENU
        call    MENU            ; Get next byte of data from the MENU Data Table
        movwf   hostdata        ; Store this byte in register "hostdata"
;
    if MCP215X==H'50'           ; Conditional Assemble for MCP2150
MENULP1 btfsc   cd              ; Is the link still active?
    endif                       ; End of Conditional Assemble for MCP2150
;
    if MCP215X==H'55'           ; Conditional Assemble for MCP2155
MENULP1 btfsc   dsr             ; Is the link still active?
    endif                       ; End of Conditional Assemble for MCP2155
;
        goto    RESET215X       ; NO, link closed for unknown reason,
                                ;    RESET MCP215x
        btfsc   cts             ; YES, Can the Host can send Data?
        goto    MENULP1         ; NO, wait for MCP215x to be ready for data
        call    SERSND          ; YES, Send the Data Byte
        decf    MENUBYTES, F    ; Decrement the number of available bytes
        btfss   STATUS, Z       ; If MENUBYTES = 0, The complete MENU has
                                ;    been sent
        goto    MENULOOP1       ; More of the MENU needs to be sent
;
        return                  ; Back to main loop
;
```


Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 9

```
;*******************************************************************************
;
; Serial Send Routine
; This routine uses the uart to send a single data byte to
; the MCP215x with hardware handshake.
; Data is passed in register called "hostdata"
;
; Wait for UART to be ready for next byte to be loaded
; Ensure the MCP215x can still receive data (test CTS signal)
; Load data to send Data, then return
;
;*******************************************************************************
SERSND  bsf     STATUS, RP0     ; Bank 1
SERSLP  btfss   TXSTA, TRMT     ; check if UART ready
        goto    SERSLP          ; not ready, wait
        bcf     STATUS, RP0     ; Bank 0
;
    if MCP215X==H'50'           ; Conditional Assemble for MCP2150
SERS1   btfsc   cd              ; Is the link still active?
    endif                       ; End of Conditional Assemble for MCP2150
;
    if MCP215X==H'55'           ; Conditional Assemble for MCP2155
SERS1   btfsc   dsr             ; Is the link still active?
    endif                       ; End of Conditional Assemble for MCP2155
;
        goto    RESET215X       ; NO, link closed for unknown reason,
                                ;    RESET MCP215x
        btfsc   cts             ; YES, check the printer handshake
        goto    SERS1           ; if CTS=1 then do not print
        movf    hostdata,w      ; get the byte to send
        movwf   TXREG           ; send the byte
        return
;
;*******************************************************************************
;
;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
;
;
        org     H'0400'         ; use 0400h as Start of String Table Routine
;
;
```

Example A-1: PIC16F877 Code to Interface to the MCP215X - Page 10

```
;*******************************************************************************
;
;   String Table
;   This table stores the MENU string, MENUCNTR is the offset.
;   The string is terminated by a null.
;
;   Caution: Do not let MENU String cross 256 word boundary
;            (that is the reason for the ORG directive)
;
;*******************************************************************************
;
MENU    movlw   HIGH (MENU)     ; Get the upper address bits where this table
        movwf   PCLATH          ;    is located and load into the PCLATH
                                ;    register
        movf    MENUCNTR, W     ; get the offset
        addwf   PCL,f           ; add the offset to PC
        DT      D'239'          ; the first byte is the byte count
                                                        ; 1 Characters
        DT      "12345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "2BCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "32345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "4bcdefgh", 0x0D, 0x0A                  ; 10 Characters
        DT      "52345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "6BCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "72345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "8bcdefgh", 0x0D, 0x0A                  ; 10 Characters
        DT      "92345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "ABCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "B2345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "Cbcdefgh", 0x0D, 0x0A                  ; 10 Characters
        DT      "D2345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "EBCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "F2345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "1bcdefgh", 0x0D, 0x0A                  ; 10 Characters
        DT      "22345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "3BCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "42345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "5bcdefgh", 0x0D, 0x0A                  ; 10 Characters
        DT      "62345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "7BCDEFGH", 0x0D, 0x0A                  ; 10 Characters
        DT      "82345678", 0x0D, 0x0A                  ; 10 Characters
        DT      "9bcdefgh", 0x0D, 0x0A                  ; 10 Characters
;
; NOTE:  0x0D = Carriage Return, 0x0A = Line Feed
;


    end
```


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, KEELOQ, MPLAB, PIC, PICmicro, PICSTART and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

dsPIC, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICDEM, PICDEM.net, rfPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn 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.

Printed on recycled paper.

DNV Certification, Inc. USA
ANSI-RAB QMS ACCREDITED
DNV
DNV MSC The Netherlands Accredited by the RvA
ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

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: 480-792-4338

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

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

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

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

Kokomo
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

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

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
Bei Hai Wan Tai 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, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200 Fax: 86-28-86766599

China - Fuzhou
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

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: 86-755-82350361 Fax: 86-755-82366086

China - Hong Kong SAR
Microchip Technology Hongkong Ltd.
Unit 901-6, 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'Shaugnessey 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: 65-6334-8870 Fax: 65-6334-8850

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

EUROPE

Austria
Microchip Technology Austria GmbH
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

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

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

Germany
Microchip Technology GmbH
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

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

10/18/02