Model LR-F2812COM

Technical Reference Manual

May 1, 2006

Document Number: 9-403-36

Table of Contents

NOTE: All versions of the model LR-F2812COM are compatible with the F28335 eZdsp. Please refer to the Link Research Application Note AN-05, "Using the model LR-F2812COM-x with the F28335 eZdsp development system" available on the Link Research website, www.link-research.com.

1. Introduction

The Model LR-F2812COM-x daughtercard provides a quick, cost effective way to add 2 high speed RS-232 serial communication ports to the F2812 or F28335 eZdsp development system from Spectrum Digital, Inc. The daughtercard connects to the eZdsp through connectors P4 and P8, and is shipped with an extra set of connectors, which the user solders in to P4 and P8 on the eZdsp. The product includes everything needed to get up and running quickly, including a 6 foot, male/female DB-9 cable, and a CD containing several sample applications. The sample applications range in complexity from a polled, single port, serial channel, to a more complex, dual port, interrupt driven, buffered, DSP/BIOS application.

2. Getting Started with the First Sample Program

1. An extra pair of connectors is included with the Model LR-F2812COM. One connector is a double row of 20 pins/row. This connector should be soldered into position P8 on the eZdsp development board. Next, the remaining connector, having a single row of 20 pins, should be soldered into position P4 on the eZdsp. It is recommended that water-soluble solder along with appropriate flux be used for this operation. Examples would be Kester type 331 solder, and type 2331-ZX flux. This will allow the flux residue to be washed off the board with plain hot/warm water.

2. The eZdsp must be configured to supply +5 Volts to the LR-F2812COM daughtercard. On the reverse side of the eZdsp board, a provision has been made to solder a small jumper wire between two adjacent solder pads. The board allows either +3.3V or +5 V to be applied to pins 1and 2 of P8, and pin 1 of P4. Insure that the +5V jumper is installed.

3. Install the two long nylon standoffs on the daughtercard using the two nylon screws. Similarly, install the four short nylon standoffs on the eZdsp board. Plug the LR-F2812COM daughtercard into the eZdsp, insuring that all 30 pins on the daughtercard properly mate with the corresponding 30 pins on the eZdsp. Figure 1 shows the orientation of the daughtercard in relation to the eZdsp board.

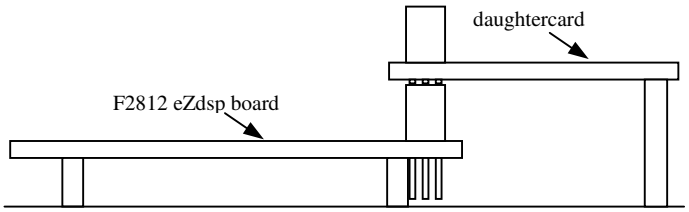


Figure 1

4. Connect one end of the supplied RS-232 cable to the Port #1 connector on the daughtercard. Connect the other end to the PC's serial port. (Note that some older PCs may have a 25 pin D connector for their serial port. If this is the case, it will be necessary to obtain a "25-pin to 9-pin serial port converter", which is readily available from any electronic/computer store.

5. Apply power to the eZdsp board in the usual manner.

6. Insert the CD supplied with the Model LR-F2812COM into the PC's CD ROM drive. The installation screen should appear in a few seconds. If this does not occur, simple navigate to the CD ROM drive in Windows File Manager and double click on the file: SETUP.EXE

7. During the installation, the user must enter a directory path where the software will be installed. It is recommended that the user install the software to the "MyProjects" subdirectory under the eZdsp software installation. For example: "C:\<eZdsp_root>\myprojects".

8. Open Code Composer Studio and click on **Project->Open**. Navigate to the **C:\<eZdsp_root>MyProjects\Link_Research\Serial_Port\F2812-> loopback1-basic** directory and open the project file **loopback1.pjt**.

9. On the PC, run HyperTerminal, and set its port parameters to 9600 bps, 8 bits, no parity, and 1 stop bits.

10. Compile and run **loopback1** in Code Composer Studio. If everything is operating correctly, a welcome message should appear in HyperTerminal, indicating that you are connected to Port #1. Now, whatever is typed at the HyperTerminal screen will be echoed back to HyperTerminal by the eZdsp.

If the results of the above steps are not successful, please refer to the Troubleshooting section of this manual.

3. The Remaining Sample Programs

The first sample program described above is quite basic. It uses no interrupts, no data buffers, and is a non-DSP/BIOS application. It was designed this way to get the user started quickly. However, in a real world DSP system, constant polling of a serial port is no way to handle a relatively slow speed serial communication port.
Each succeeding sample application adds 1 feature to the previous one. The chart below outlines this scheme:

Project	Feature added to previous
Loopback2-buffers	Adds transmit and receive buffers to the loopback1-basic project. Also, adds code to handle both serial ports.
Loopback3-interrupts	Adds transmit and receive interrupts to the loopback2-buffers project. This eliminates the need for polling the UART registers.
Loopback4-DSP/BIOS	Finally, this project converts the loopback3-interrupts project into a DSP/BIOS project. Interrupts are now configured using the DSP/BIOS graphical user interface.

The user is encouraged to experiment with all of the sample programs.

4. Baud Rate Calculation

The sample programs' include files list many of the common data rates used today. However, non-standard data rates can also be used by calculating the *baud rate divisor* directly using the following formula:

$$\text{Baud Rate Divisor} = \frac{\text{LSPCLK}}{8 \times \text{Desired Baud Rate}} - 1$$

where:

LSPCLK is one of the DSPs internal clock, whose speed depends on the DSP's external crystal and internal register settings. For the sample programs running on the eZdsp, LSPCLK = 37,500,000 MHz.

As an example, if a data rate of 7200 bps is desired, the calculation would be as follows:

$$\text{Baud Rate Divisor} = \text{ROUND}\left[\frac{37{,}500{,}000}{8 \times 7200} - 1\right] = 650$$

5. Hardware

The LR-F2812COM board physically interfaces to the eZdsp through connectors P4 and P8. Electrically, the board connects to the TMS320F2812 DSP according to the following chart:

DSP signal	Daughtercard Function	Connector/Pin	Direction
SCITXDA	Transmit data – Port 1	P8-3	From eZdsp to LR-F2812COM
SCIRXDA	Receive data – Port 1	P8-4	From LR-F2812COM to eZdsp
SCITXDB	Transmit data – Port 2	P4-18	From eZdsp to LR-F2812COM
SCIRXDB	Receive data – Port 2	P4-19	From LR-F2812COM to eZdsp

Two LEDs are provided for each of the two serial ports. The Green LED is the transmit data LED, and the red LED is the receive data LED. (This can be remembered by noting that “R” stands for Red and for Receive).

Each RS-232 interface utilizes a female DB-9 connector. This makes the daughtercard look like a modem (a DCE) to the outside world. As such, the daughtercard can connect Directly to a PC using a “straight-thru” cable, such as the one supplied with the board. In cases where a DTE interface is required, for example, where two model LR-F2812COM boards are to be connected back to back, then a null modem device is required to convert one of the LR-F2812COMs to a DTE. These null modem devices are readily available at any computer or electronics store.

The DB-9 pinout used on the LR-F2812COM utilizes a subset of the standard RS232 signals. The pin out of the DB-9 connector used on the model LR-F2812COM is as follows:

DB-9 Pin	Signal Name
1	See note 1
2	TD
3	RD
4	See note 1
5	GND
6	See note 1
7	See note 2
8	See note 2
9	Not Used

Note 1: Pins 1, 4, and 6 are tied together at the DB-9 connector.
Note 2: Pins 7 and 8 are tied together at the DB-9 connector.

6. Software API

A set of 6 high level functions are provided in source code format which greatly facilitate using the F2812-COM-x daughtercard. These functions manage 2 buffers, one for transmit data and one for receive data. The length of these buffers is 256 bytes each, but can be easily modified by changing #define statements in loopback<x>.h.

Before the serial port functions can be used, the function **Initialize_RS232_Port()** must be called. This function does several things – it enables the SCI function of the DSP, it sets the baud rate, parity setting, number of data bits, and number of stop bits to be used. It also initializes the transmit and receive buffer pointers. As written, the software sets up the UART for 8 data bits, no parity, and 2 stop bits. The port number (1 or 2) is passed as the first parameter, and the data rate is passed as the second argument. The data rate must be in the form listed in the define statements in the loopback<x>.h file.

The high level functions that can be used to access the serial port are as follows:

`int send_char(int port, unsigned char chr);`

This function places a single character in the transmit buffer. The first argument specifies the port number (1 or 2), and the second argument is the character to be transmitted. The function returns 0 if successful, -1 otherwise.

`int send_block(int port, unsigned char* chr, unsigned int length);`

This function sends a block of characters to the transmit buffer. . The first argument specifies the port number (1 or 2), the second argument is a pointer to the block, and the third argument is the length of the block to send. The function returns 0 if successful, -1 otherwise.

INT RECV_CHAR(INT PORT, UNSIGNED CHAR* CHR);

This function returns one character from the receive buffer. The first argument specifies the port number (1 or 2), the second argument is a pointer to the character to be received. The function returns 0 if successful, -1 otherwise.

`int recv_block(int port, unsigned char* chr, unsigned int length);`

This function returns a block of characters from the receive buffer. The first argument specifies the port number (1 or 2), the second argument is a pointer to the area in memory that will receive the data, and the third argument is the length of the block to receive. The function returns 0 if successful, -1 otherwise.

UNSIGNED INT GET_RECV_COUNT(INT PORT);

This functions returns the number of characters in the receive buffer. The one and only argument is the port number (1 or 2).

VOID ..... SERVICE_SERIAL_PORT(INT PORT);

This is the function that handles all the low-level operations of the serial port. It polls the UART for available receive data, and transfers this data to the receive buffer. It also polls the UART to see if the transmitter is ready to accept another character. If so, it sends the next character from the transmit buffer. This function should be called periodically, at a rate fast enough to prevent receiver overruns. This rate is dependent on the data rate the UART is programmed for. The one and only argument is the port number (1 or 2).

Note that in their current form, none of these functions should be called from inside an ISR. Should it prove necessary that these functions be called from inside an ISR, they can be easily modified for such operation. Please contact Link Research for assistance.

7. Specifications

Number and type of serial ports:

Model F2812-COM	Two RS232 compatible ports
Model F2812-COM-1	Two 1mm Fiber Optic ports[1]
Model F2812-COM-2	One RS-232, one Fiber optic port[1]

Connectors:

Model F2812-COM	Two Standard DB-9, female
Model F2812-COM-1	Two 1mm Fiber Optic connectors[1]
Model F2812-COM-2	One DB-9, female, one Fiber optic[1]

Baud rates:

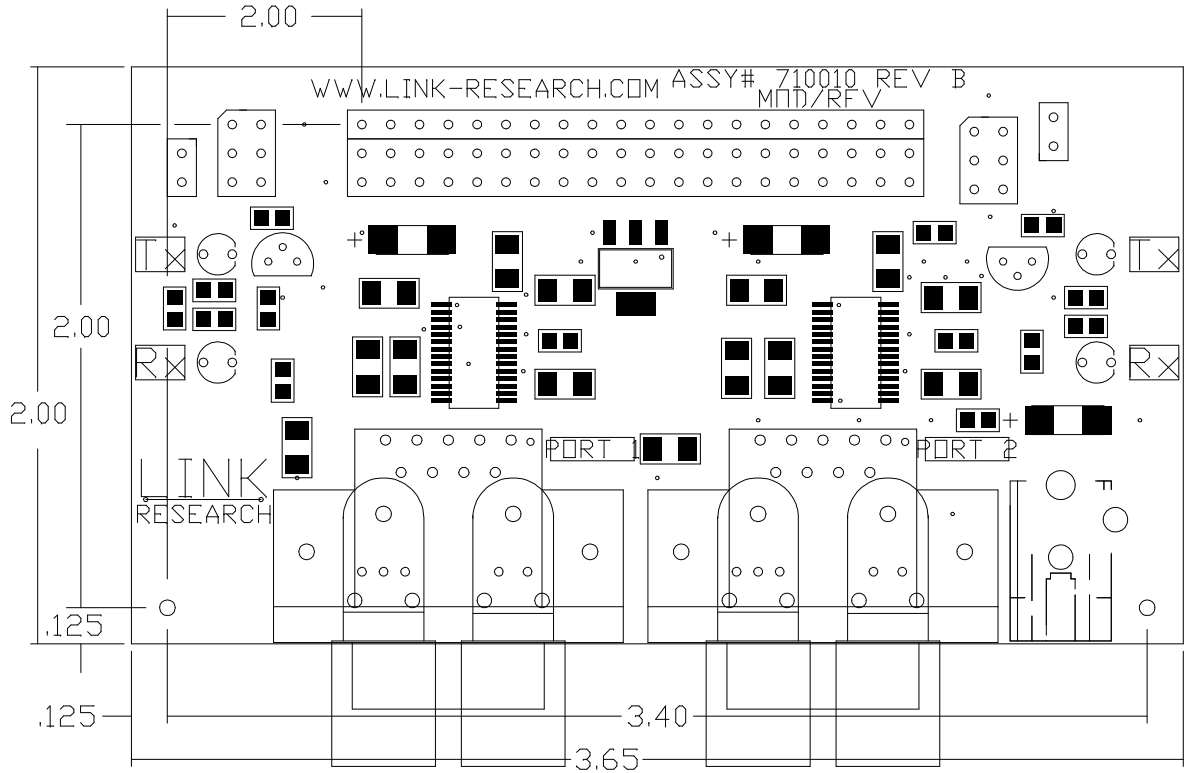
All versions	921.6 kbps max.

Current Draw (both ports at 921.6 kbps, looped)

LR-F2812COM	25 mA max
LR-F2812COM-1	90 mA max

Note 1: The fiber connectors are manufactured by Industrial Fiber Optics, Inc. The transmitter is P/N: IF-E91D, and the receiver is P/N: IF-D96F

8. PCB Mechanical Detail Drawing



Version History

Date	Version	Description
1-May-2006	1.0	Original
30-Mar-2007	1.1	Changed serial port data rates
7-Dec-2007	1.2	Changed max data rate and part numbers for fiber version, Added max current draw values.
13-Dec-2007	1.3	Added note pertaining to the F28335 eZdsp dev. System.
22-Jan-2009	1.4	Added Figure 1