Model LR-LAN-x

TECHNICAL REFERENCE MANUAL

February 1, 2008

TEL 1-401-270-4445
FAX: 1-401-270-5221

WEB: www.link-research.com

Table of Contents

1. General

This technical reference covers the following product part numbers, collectively referred to as the “Ethernet daughtercard“:

LR-LAN	(Ethernet Device Server daughtercard)
LR-LAN-S	(Ethernet Device Server daughtercard, stackable)

The Link Research Ethernet daughtercard utilizes the Lantronix, Inc. XPort™ Device Server module. This module contains the MAC and PHY portions of the Ethernet Interface, as well as the RJ45 jack itself. It is IEEE 802.3 compatible. The Lantronix XPort™ has the following features:

- Simple serial interface to the host processor (the DSP on the DSK), supporting data rates up to 921,600 bps.
- Flash memory to hold HTML and JAVA applet files.
- User configurable IP address and Port number.
- Email alert system.
- Full TCP/IP stack.

Complete documentation for the Lantronix XPort™ can be obtained from the Lantronix, Inc. website located at: www.lantronix.com.

A diagram of the Ethernet daughtercard is shown in Figure 1.

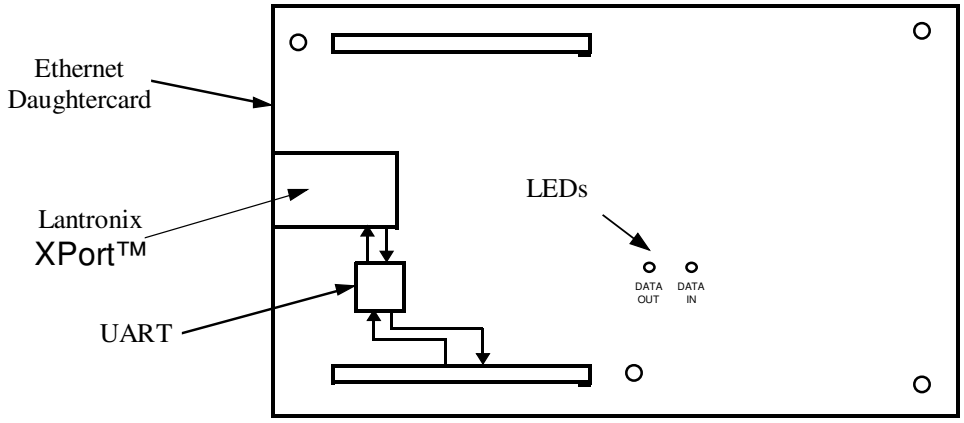

Figure 1

A UART IC is employed to interface the XPort™ to the DSP through the EMIF connectors located on the DSK. This UART IC is an NXP (formally Philips) SC16C2550B dual UART device. The UART contains a 16 byte transmit FIFO,

and a 16 byte receive FIFO. A datasheet for the SC16C2550B is available from the NXP website located at www.nxp.com

2. Uses of the Ethernet Daughtercard

The following sections illustrate several interesting ways in which the Link Research Ethernet daughtercard can be used.

2.1. Using a TCP/IP Client to communicate with a DSK

In this configuration, A TCP/IP capable client application, such as MATLAB, HyperTerminal, or a user developed TCP/IP capable client application can communicate with a DSK physically located anywhere on the LAN or internet. Figure 2 illustrates this setup. Software on the DSK need only be concerned with UART initialization, and receiving and transmitting data to the UART. On the client side, the TCP/IP host application must create (open) a socket to the IP address and port number previously programmed into the XPort™ . (See section 5.1). Once a valid socket has been established, the host application can use the socket APIs to send and receive data to and from the DSK.

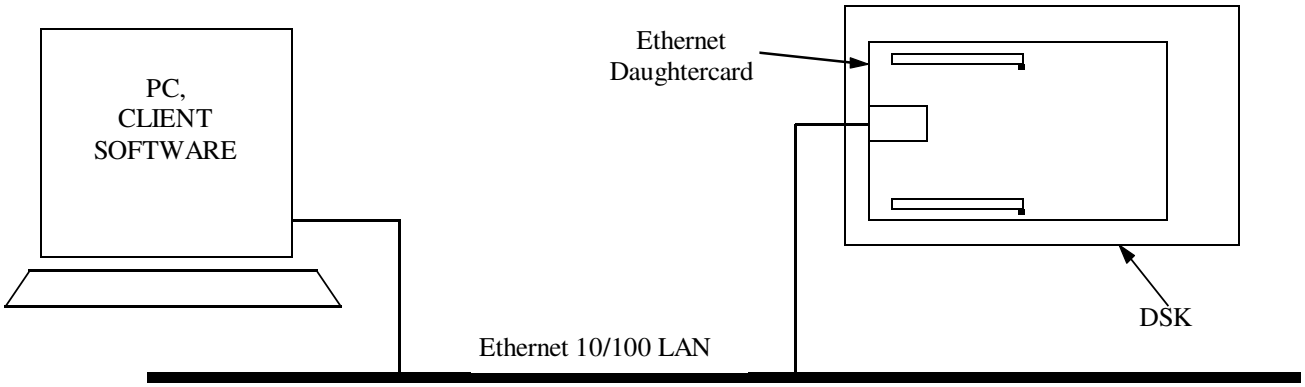

Figure 2

2.2. Web browser communicating with the DSK using JAVA applets

For this configuration, no special client-side application is required. Instead, any standard web browser may be used to communicate with a DSK attached to the LAN. In this case, JAVA applet code is used to open the socket to the XPort™ and subsequently, send and receive data. The demonstration program on the included CD named **dip_switch_demo** illustrates how this is done. The XPort™ contains FLASH memory cable of holding HTML as well as JAVA applet files. Section 5.2 shows the method used to download files to the XPort™ .

2.3. DSK-to-DSK communications over an Ethernet LAN

This configuration requires absolutely no TCP/IP programming whatsoever. However, there is slightly more effort involved in programming the 2 (or more) XPort™ devices. For this setup, One XPort™ must be configured to be a master, while the other must be configured as a slave.

3. Functional Description

The DSP/DSK communicates with the Ethernet daughtercard exactly as if it were communicating with a UART. No TCP/IP software is required on the DSK.

A typical configuration utilizing a DSK and the Ethernet daughtercard is shown in Figure 2. The TCP/IP client software would basically open a TCP/IP channel at the IP address and port number which has been pre-programmed into the X-Port device (see section 5.1.) The default TCP/IP port number is 10001. Once a channel is successfully opened, any data sent by the client will appear as *received data* to the UART, and any data sent by the DSK will appear in the client program as *received data*.

In the configuration shown in Figure 2, the Ethernet daughtercard assumes the role of a slave, and as such, cannot initially open the TCP/IP channel. The client application must open the channel. However, once the channel is open, both the DSK and the client application can send data at will.

4. PC Software Installation

A simple graphical user interface software package is provided on the installation CD to facilitate XPort™ configuration. This program is called ***Device Installer,*** and is made available by Lantronix, Inc. The user is encouraged to visit the Lantronix web site for the most current version of this useful software program.

Device Installer requires the Microsoft™ .NET foundation software. If this software is not already installed on the system, it must be installed before Device Installer is installed. The Microsoft™ .NET software can be installed from the supplied CD by following the following steps:

1. Insert the supplied CD into the CD ROM drive.
2. With Windows Explorer, change to the ***XPORT*** directory.
3. Unzip the file dotnetfx.zip to a temporary directory.
4. Run the ***dotnetfx.exe*** program.

5 Follow the instructions presented during installation.

Once .NET has been successfully installed, Device Installer can be installed using the following steps:

1 Insert the supplied CD into the CD ROM drive.
2 With Windows Explorer, change to the ***XPORT*** directory.
3 Unzip the file XPort_DeviceInstaller_3606.zip to a temporary directory
4 Run the ***Setup.exe*** program.
5 Follow the instructions presented during installation.

5. Configuring the Xport

5.1. Giving the Xport an IP address and Port number

There are several ways to assign an IP address and Port number to the Xport device. However, the easiest way is to use the *Device Installer program* and follow these simple steps:

1 With a powered-up Xport daughtercard attached to the LAN, run Device Installer.
2 Click the SEARCH button to locate the XPORT device. Don't worry about any error messages that appear at this point.
3 Select (i.e., click on) the XPORT device in the resulting list.
4 Click on **ASSIGN IP** in the button bar at the top.
5 Enter the desired IP address in the standard dot notation (for example: 192.168.0.25)
6 Click the **UPDATE** button.

Note: The default port number is 10001. Normally it is not necessary to change this value.

5.2. Downloading HTML and JAVA files to the Xport's FLASH memory

The Xport contains FLASH memory that can be used to store user defined web pages (.html files) and JAVA applets (.class files). Before these files can be stored in the Xport, they must be converted to a single COB file using the web2cob.exe program contained on the included CD. COB files are Lantronix proprietary formatted files similar to ZIP files. Web2cob.exe is an MSDOS program that must be run from an MSDOS command window. The command line format of web2cob.exe is:

Web2Cob [/o <output file>] [/d <directory>]

Where **output *file*** is the name of the output COB file, and ***directory*** is the name of the subdirectory where the *html* and *class* files are located.

Once the .cob file is created, Device Installer can be used to download the file into the Xport's FLASH memory. To do this, Run Device Installer and locate the Xport device by clicking on the SEARCH button. Next, select the device. Now click on the UPGRADE button. In ***WebPage2***, select the cob file and click OPEN, and then OK. After a few seconds, the download procedure is complete.

6. Hardware Description

6.1. Accessing the Xport from the DSK

From the standpoint of the DSK, communicating over the LAN is done essentially by communicating with the UART located on the Ethernet daughtercard. The UART IC is memory mapped into the DSP's address/data space. The actual physical addresses are dependent on the particular DSK in use. Table 1 shows the base memory address of the Ethernet daughtercard for each of the supported DSKs. Table 2 lists the individual UART register address offsets from the base address.

For example, to access the Line Control Register (LCR) of the UART on a 6416 DSK, take the base address from Table 1 and add the register offset value from table 2 as follows:

0xA0200000 + 0x0C = 0xA020080C

Tip: The absolute memory addresses of the UART registers are provided in the demonstration program. Simply locate the "include file" associated with the demonstration program for the particular DSK you are using.

6.2. Interrupts

Because the Ethernet daughtercard has been designed to operate with many TI and Spectrum Digital DSKs, and for future obsolescence-proof, the routing of the interrupt signal from the UART to the DSK's peripheral expansion connector has been designed with a jumper block . Some DSKs respond to an active-high interrupt signal, while others respond to an active low signal. Also, each DSK has a different number of external interrupt lines made available at different pins on

the peripheral connector. Additionally, some users may be using some of the external interrupt lines for other peripheral devices. Therefore, the Ethernet daughtercard has been designed to allow the following:

1. The interrupt signal generated by the UART can be jumpered to any available DSK interrupt line.
2. DSPs expecting either active high or active low interrupt inputs are supported.

TABLE 1			
DSK	**BASE MEMORY ADDRESS**	**Page**	**CE2 (Chip Enable) control register value**
5416	0x0800	0x10	N/A
5510	0x500000	N/A	N/A
6416	0xA0200000	N/A	0x5A35E823
6711	0xA0200000	N/A	0x5A35E823
6713	0xA0200000	N/A	0x03D00F21

TABLE 2				
		Offset from Base memory address for each DSK		
REGISTER	**REGISTER NAME**	**6711, 6713, 6416**	**5510**	**5416**
RBR	Read Buffer Register	0x00	0x00	0x00
THR	Transmit Holding Register	0x00	0x00	0x00
DLL	Divisor (low byte)	0x00	0x00	0x00
DLM	Divisor (high byte)	0x04	0x02	0x01
IER	Interrupt Enable Register	0x04	0x02	0x01
ISR	Interrupt Status Register	0x08	0x04	0x02
FCR	FIFO Control Register	0x08	0x04	0x02
LCR	Line Control Register	0x0C	0x06	0x03
MCR	Modem Control Register	0x10	0x08	0x04

LSR	Line Status Register	0x14	0x0A	0x05
MSR	Modem Status Register	0x18	0x0C	0x06
SPR	Scratch Pad Register	0x1C	0x0E	0x07

Configuring hardware interrupts on the UART daughtercard requires that the user install a jumper wire using the following 2 steps:

1. Refer to Table 3 to relate the board E-point(s) to the desired external interrupt(s) for the particular DSK in use.
2. Refer to Table 4 to relate the board E-point(s) to the desired UART interrupt source configuration.

Here are some common configurations:

1) With the Ethernet daughtercard attached to a 6416 DSK, attach the interrupt signal from the UART to external interrupt EXT_INT4

Solution: Jumper E1 to E17

2) With the Ethernet daughtercard, attached to a 5510 DSK, route the UART interrupt signal to external interrupt INT0.

Solution: Jumper E2 to E15

6.3. LEDs

A pair of LEDs, one green and one red, are provided to monitor the serial data between the UART and the XPort™ . The Green LED is labeled “DATA OUT”, and indicates when data is being transmitted by the DSK, to the XPort™, to be sent over the Ethernet connection. Likewise, the Red LED is labeled “DATA IN”, and indicates when data is coming into the XPort™.over the Ethernet connection, and is being received by the DSK, from the XPort™.

7. UART Driver Software

A UART API library is included on the CD to allow easy access to the UART, and consequently, the Xport. This software handles all of the low level hardware access functions, including initializing the physical UART device, configuring the UART for a particular data rate, etc. The software includes the interrupt routine

that handles both transmit and receive service requests from the UART device. The software also manages transmit and receive circular buffers, whose length is user configurable.

To use the library, simply locate the DSK-specific **uartlib.lib** library file on the included CD and include it in a CCS project. Also, add an include file (see dip_switch_demo.h) to any C source files that will use the library API functions.

The application API consists of 7 high level functions. These are defined in table 7.

8. Uart API Function Description

TABLE 7	
Function	Description
void initialize_UART (int, unsigned char);	Initializes the UART, first argument specifies channel number (1,2,3,or 4), second argument specifies the data rate. See the “mainxxxx.h” file for the definitions for the data rate constants.
int UART_send_byte (int, unsigned char);	Sends a single 8 bit character to a UART channel. First argument specifies the channel, second argument specifies the character.
int UART_recv_byte (int, unsigned char*);	Receives a single 8 bit character from a UART channel. First argument specifies the channel, second argument specifies a pointer to the character.
int UART_send_block (int, unsigned char*,unsigned int);	Sends a block of 8 bit characters to a UART channel. First argument specifies the channel, second argument specifies a pointer to the character array, third channel specifies the number of characters to send.
int UART_recv_block (int, unsigned char*,unsigned int);	Reads a block of 8 bit characters from a UART channel. First argument specifies the channel, second argument specifies a pointer to the character array, third channel specifies the number of characters to read.
unsigned int UART_recv_count(int);	Returns the number of characters available to be read. Argument specifies the UART channel to query.
unsigned int UART_xmit_count(int);	Returns the number of characters in the transmit output buffer. Can be used to determine if there is enough room in the transmit buffer for more data. Argument specifies the UART channel to query.

9. The Demonstration Program

The CD which ships with the Link Research Ethernet Device Server daughtercard contains an application program which demonstrates how a web browser can be used to monitor and control DSK operation. This application consists of a complete Code Composer Studio project, as well as HTML and JAVA files that must be downloaded into the XPort™. The program demonstrates communication between the browser JAVA applet and the DSK. This technique can be extended to allow the DSK to communicate with any TCP/IP client application, for example, MATLAB™, or a TCP/IP client application of the user's creation.

The particular demonstration supplied with the Ethernet Device Server daughtercard allows a browser to monitor the 4 user DIP switches present on the DSK. The program running on the DSK monitors the 4 user dip switches, and for each switch that is turned on, a corresponding white square is shown in the browser window. Also, the DSK illuminates the corresponding LED.

9.1. Running the Demonstration Program

To run the demo, perform the following steps:

1. With the Ethernet daughtercard attached to the DSK, and the DSK connected to the PC, start Device Installer and download the file "Dip_Switch_Demo.cob" located on the included CD to the ***WebPage 2*** section of FLASH memory. Use the procedure described in section 5.2.

2. Start CCS and verify that it can communicate with the DSK.

3. On the supplied CD, locate the directory for the particular DSK being used. Under this directory, there will be a directory called ***dip_switch_demo***. Copy this entire directory to the "My Projects" directory within your CCS installation directory.

4. Open CCS and load the dip_switch_demo project. Open the .cdb file. If a message apears asking if you want to convert to the current DSP/BIOS version, click YES. Save the .cdb file. Perform a "Rebuild All" command.

5. Download the program (.out file) to the DSK. Run the program.

6. Open a browser window on the PC, for example Internet Explorer.

7 In the URL edit box, type the numerical IP address of the XPort™ and press GO.

8 If the appropriate version of the Java Runtime Environment (JRE) is ***not*** installed on your system, a message will appear asking you if you want to install the JRE. Click Yes. This will be a fairly lengthy download, but only needs to be done once.

9 At this point, a blue rectangle containing 4 white rectangles should appear on the screen. These rectangles will show the state of the 4 dip switches on the DSK. By activating one or more of the dip switches, the rectangles will change from not-filled to filled. Also notice that when a dip switch is activated, the corresponding LED on the DSK is illuminated.

9.2. The Possibilities

This demonstrates two way communication between the DSK and a Java applet running in a browser window on the PC. Using this as a starting point, many interesting applications can be created. Here are some examples:

1 Data from the DSK can be sent to the PC and stored on disk.

2 A GUI (graphical user interface) can be created as a Java applet, and used to control various parameters on the DSK.

3 A DSK application can be controlled and monitored from any location having LAN access, and for that matter, anywhere having internet access.

4 DSP/BIOS instrumentation variables can be sent to any PC anywhere on the internet, with little performance impact on the DSP application.