02/04/13

LRRS_3X Network Interface

The LRRS_3x application can broadcast status information on the network, if you so desire. **Commanding the LRRS_3x application via the network is only available if the option 'Allow Network Control' is checked in the system-options-operations menu**.

There are three types of information broadcast.

1. Maintenance Data – This is broadcast at a one hertz rate. This is information about the program itself, e.g. the number of Ls35 cards in the system, if anything is being simulated, the project file loaded, etc.
2. Status Data – This is broadcast at a variable rate from one hertz to twenty hertz. This data contains status information from each Ls35 card (and sub-cards) in the system.
3. FFT/IQ Data – This is broadcast at approximately 10 Hertz. This contains the FFT and IQ data for each receiver, on each Ls35 card.

Each type of data requires its own network port. The maintenance data is on the first port you assigned, and the status data is on the next 6 port. The FFT/IQ data is on the next 6 ports. Each group (or 35 card) gets its port bumped by 1, e.g. If your start port is 10000, then your maintenance port would be 10000, the Ls35 card 1 status port would be 10001, the Ls35 card 2 status port would be 10002, your FFT/IQ data port for card 1 would be 10007, your FFT/IQ data port for card 2 would be 10008, etc.

In addition to the broadcast data, if the LRRS_3x application allows it for its configuration, then you can also control some parts of the application. Control requires two ports, which are used for all Ls35 card groups. The first port is the command port, and the second port is the acknowledge port. The control command port is at the start port plus 16, and the acknowledge port is at start port plus 17. So for the example above, your command port would be 10016 and acknowledge port would be 10017. The protocol is similar to SNMP, but much simplified (RSNMP – really simple).

Setup of the network for Windows is **your** responsibility. If the Windows network is not setup, then see your network administrator.
Ensure you have privileges to broadcast and receive. This is broadcast protocol, but you can set the mask to be to a single client if you wish.
Ensure you have your firewalls and anti-virus software set up to allow data flow.
You can fight with your network administrator about those issues.

Lrrs_3x – Network Setup

There's not a lot to do here. On the startup banner, select System – Network Configuration.

From there you can Enable/Disable the network, set the beginning Port, and set the IP address you wish to communicate with. If you change either of those items, then the program MUST restart. It will do this automatically for you.

You can also set the broadcast rate (of the status data). The program does not have to restart if you change this.

The configuration information for the network is stored in a file called "Ls35NetCfg.cfg" in the System\HardwareConfiguration subdirectory where you have Lrrs_3x installed.

If that file does not exist (i.e. the first time you have the system), the default port is 0x8192 (33170 decimal), and the default broadcast mask is the first 3 octets of you PC's IP address, followed by 255, e.g. 192.168.1.255.

NOTE – You cannot command with an IP address set to 255 on any of its octets. You will only be able to broadcast the maintenance and status and FFT/IQ. Control is only allowed by a single PC.

Your Network Client, Status and FFT/IQ and Maintenance Only

Writing your own client application to receive the broadcast data can be trivial or as complex as you like. YOU are writing it. If you are not quite knowledgeable about writing code for threads and network interfaces, **STOP**. Do not read any further. Contact your local software engineer.

There is a separate document for the Control packets.

There's really not a lot to do here. You will need the following files:

- "NetLrx3xCmn_DefinesUnit.h".
- "Lrs3xSetupDefinesUnit.h".
- "Lrs3xStatusDefinesUnit.h".

They have all the necessary information for the content of the broadcast packets.

There are three basic structs for the three types of broadcast packets

1. netls3xmaintstatusstruct for the Maintenance packets
2. netls3xstatusdatastruct for the Status packets
3. netfftiqpktstruct for the FFT/IQ packets

You'll note at the top of the .h file is a

typedef void __fastcall (__closure *TLs3xUdpRxDataCallbackProc)(char* fromip, int fromport, char* bufdata, int numbytes);

This typedef is Borland's method to get a callback into a class. Since most of you use Visual, you can comment it out. It is only used by the example client application.

NOTE. You'll also see the #pragma that sets the data alignment to byte. THIS IS A MUST.

You will need at least three sockets. One for the maintenance packet, and two each for the number of LS-35 cards in the system.

For the status data and the FFT/IQ data, you'll note the structure contains the CardNumber. This is the index into an array of those types of data, one for each Ls35 card used by the Lrrs_3x application. Make sure you do enough idiot checking to keep from getting an index out of bounds error.

Just copy the data from the network buffer into your data structures, and do with them as you please.

Now, just what the data means is a mystery to me for some of it. I'm not an RF type person. The variable names in the struct are self explanatory, if you're an RF type.

Note about FFT/IQ structure

The structure for the FFT/IQ data is shared. Each Ls35 has a single port for this type data. Each Ls35 has 2 receivers. Each receiver can have FFT data and each can have IQ data. You'll note part of the netfftiqpktstruct is the variable IsIqPkt. The Buf[] in the netfftiqpktstruct can be typecast to a netiqelementstruct* for an array of I and Q data points, or as is as an array of floats for FFT data.

This Buf[] is a float array of MAXNET_IQDATABUFPOINTS * 2 data points. The value of MAXNET_IQDATABUFPOINTS currently is 4000, double the max size I have seen. The card, however can contain up to 16384 elements. If you have a setting that exceeds the MAXNET_IQDATABUFPOINTS, an error sound will emit and an error message will be entered into the application log. This would be your queue to contact Lumistar to have the value changed. You'll have to supply the setup information that caused the error message as well. The reasoning to keep the number of MAXNET_IQDATABUFPOINTS to 4000 is that it keeps the total size of the netfftiqpktstruct to under 32 kBytes, so a single send/receive socket command can be used on all PCs with the WINAPI (some PCs will handle 64 kBytes, but all will accept 32 kBytes).

Part of the netfftiqpktstruct is the NumElements in the Buf[] array. If this is an IqDataPkt, then NumElements * 2 will be the total number of floats in the array. Again, much easier to typecast the Buf[] to an array of netiqelementstruct.
I'm not real sure on why the number of elements change, but it has to do with the FFT span. For an in depth discussion of this, contact RT Logic. The FFT span is part of the netfftiqpktstruct so you can see what it is. It may later be added to the command set. Likewise, the type of FFT performed on the data is sent over. This too may eventually make it to the control set.

Example Application

Included is an example client application. This example application is written in Borland C++. Again, since most of you write in Visual, you'll have to adjust the code from Borland specific to Visual specific, which is mainly the GUI work and strings.
It doesn't show much, but it does show data from the three types of broadcast sockets, and some stats on packets received.

For control, the right side of the GUI is enabled if the LRRS_3X application allows it and you have opened the sockets.

This application uses my specific tools libraries (which you cannot have). You have your own libraries if you do this for a living.

It also uses a general network utility unit called "NetLrx3xCmn_UtilsUnit.h". This sets up the client socket threads. You can see what's going on there, and set your threads up that way, or your own way.

There is little to no idiot checking in the example application. It is up to you to provide as much, or as little, as you wish in your application.

The code has most of the documentation about the client example application. I tried to keep it simple, but with Control capabilities, it just gets messy. If you don't need Control capabilities, you'll see a line with the text
"// EVERYTHING BELOW HERE IS IF YOU ARE DOING CONTROL COMMANDS"
You can just ignore those. They are the button pushes to control the LRRS_3X application.

How to Plot the FFT and I/Q Data

The data contained in the netfftiqpktstruct Buf[] is as it comes from the Ls35 card.

For I/Q data, each element is an I/Q pair, and can have a value between -1.0 and +1.0. To plot it, the center of your plot is (0.0,0.0). There are 4 quadrants that can result from these points, as follows:

```
-,+          +,+
      0,0
-,-          +,-
```

Hint: To keep the plot from reaching the edge on your plot, you could set your scaling so the min and max points are -1.25 and +1.25.

For FFT data, the x is just the element number, and the y is the dB value. The min dB value is -90.0. The max dB value is +10.0.

It is up to you to make it fancy and measure things if you wish.