Lenz
ELEKT NIK GMBH

XpressNet Specification

Includes using the
LI100/LI100F and LI101F
Computer Interfaces to
extend XpressNet to a PC

Also includes descriptions for older X-Bus V1 and X-Bus V2 Commands

6/2003

1 General

This document contains the description of the XpressNet protocols including those used by the LI100 series of, computer interfaces (LI100, LI100F and LI101F)[1]. XpressNet is a multipoint-to-multipoint data network used for model railroad control. The LI100 computer interface is the network-attached device that allows a PC to interact with other XpressNet devices for the purpose of controlling the model railroad. XpressNet is an RS-485 based network operating at a speed of 62.5 kilobits per second. The LI100 interface provides an interface between an RS-232 interface and the RS-485 network to enable PC programs to utilize the complete XpressNet V3 protocol set. The older X-Bus protocol set for use with older command stations is also included herein.

1.1 XpressNet Architecture

XpressNet is the network used to interconnect the input and control devices of a Digital plus by Lenz NMRA DCC system. XpressNet is also supported by a variety of other NMRA DCC manufacturers including Atlas, Roco and CVP products. (Prior versions of the X-Bus protocols were also supported by Arnold and ZTC systems.) The intention is to allow free interchange of all XpressNet equipment amongst any manufacturer who supports the protocol.

In many model railroad control networks the intelligence is contained in the central unit (command station). In XpressNet, the intelligence is distributed amongst each network-attached device. The command station is responsible for generating the DCC track packets, for maintaining the priority queue of DCC packets being sent to the track, and for maintaining the overall state of the entire system. Network-attached devices are responsible for presentation and maintenance of the user interface. Only request for a change or a request for information should be sent over XpressNet. XpressNet devices should not send refresh requests as this is not needed and only consumes bandwidth.

XpressNet is based on the EIA RS-485-Standard at the link layer using half duplex with differential signal transmission. The specific characteristics are:

1 start bit (0), 9 data bits, 1 stop bit (1), no parity bit

Baudrate: 62.5 kilobits per second

In order to guarantee each XpressNet device has an excellent and deterministic level of service, XpressNet devices are only allowed to transmit when they are provided a transmission window. This prevents traffic collisions and the need for retransmissions which lowers the need for the overall XPressNet transmission speed and increases its reliability. In most cases the transmission window is controlled by the Command Station who acts as the manager of the XpressNet communications.

The Command Station is responsible for providing a specific XpressNet device with a transmission window to allow them to send a specific request or instruction. Requests for information are normally answered in the same window. Other requests such as reading a CV in service mode may be answered in a subsequent window. Once the exchange of information with the device is completed the command station passes the transmission window for communication to the next active attached network device. This occurs before the requested sequence of instructions transmitted during the transmission window is processed. Later, when the instruction is processed, the need may arise for a broadcast message to be transmitted or for the need to not proved additional transmission windows to the attached network devices for a considerable period of time. For example a request to enter service mode will result in a broadcast message and no further timeslots provided except for the specific device that requested entry into service mode.

[1] Unless otherwise specified the term LI100 in this document refers to any of the three computer interfaces (LI100, LI100F, and LI101F)

The command station may also send to the network attached device information that the device did not explicitly request, but must by necessity react to (for example: indicating that the control of a locomotive is being taken over by another device).

Each XpressNet request and response is sent in a packet. An XpressNet packet consists of a series of 8 bit bytes. To initiate transmission, the Command Station transmits a ***Call Byte*** to the device. The call byte precedes the message header byte which describes the contents of the message. The call byte includes a parity bit, which refers to the parity of the call byte. A parity bit is inserted in bit 7 of the call byte. Bit 7 is supplemented in such a way that a straight number of ones are located at the call byte (straight parity). This call byte is not transferred by the LI100 to the PC as it is at a layer transparent to the higher layer commands and responses.

The next byte after the call byte is the ***Header Byte***. The header byte consists of two parts with each part consisting of a 4-bit Nibble. The upper nibble contains the identification of request to be performed; the lower nibble contains the number of data bytes in the packet. The number of data bytes in the packet does not include the header byte and the error byte. Thus the maximum number of bytes that can be in a single packet is 17, 15 data bytes plus one header byte and one error detection byte.

The last byte in a transmission is the ***Error Detection Byte***. It is formed using an X-Or-linkage of all preceding bytes inclusive of the header byte. It is the responsibility of each network device to examine whether the X-Or-linkage of all received bytes is identical to the error detection byte. If the X-OR received is different than the X-OR calculated than an error occurred during transmission and the packet should be discarded. Some transmissions only contain a byte and do not include either a header byte or an error detection byte. The call byte is not included in the calculation of the error detection byte.

At the hardware layer, XpressNet communications utilize standard USART 9-bit mode multi-processor communications. Bit 9 is added on the hardware side for transmissions from the command station to other XpressNet devices to define whether the call byte is an address (bit 9 = “1”), or data (bit 9 =”0”). Data communication from XpressNet devices to the command station takes place in such a way that that the ninth bit is always set to "0". This scheme allows the device to have an interrupt line to detect the 9th bit and quickly determine if it´s address or the broadcast-address is being sent.

1.1.1 XpressNet timing constraints

XpressNet is designed so that each device with an address of between 1 and 31 is provided a window for transmission during regular intervals. During this window a communication exchange takes place. Upon completion of the exchange the instructions are processed and then the next transmission window is opened.

An XpressNet device designed to work with XpressNet V3 and later systems must be designed so that it begins its transmission within 110 microseconds of receiving its transmission window. (older X-Bus based systems required this transmission to begin within 40 microseconds.) Command stations must be designed to accept transmissions received up to 120 microseconds after transmitting the window. The difference is to provide a design tolerance between the different types of devices.

Under normal conditions an XpressNet device must be designed to be able to handle the receipt of its next transmission window between 400 microseconds and 500 milliseconds after the receipt of the last window. The difference assumes the case of one XpressNet device with nothing to transmit while the later assumes a full complement of 31 XpressNet devices each needing to transmit the longest instruction which in turn requires the command station to send a broadcast. The exception to this is when the command station terminates normal communication which happens for example when it enters service mode programming. During service mode programming on the devise that requested entry to this mode is provided a window. Normal operations is returned once service mode is exited.

Between transmission windows the XPressNet device must be designed to be able to process broadcast messages or unsolicited information which can be received at any time.. For example, when the command station processes an Emergency Stop, it will then transmit a Broadcast to all devices or when another device takes control of a locomotive the device that last controlled the locomotive is informed of this action.

1.1.2 XpressNet Topology, wiring, and power considerations

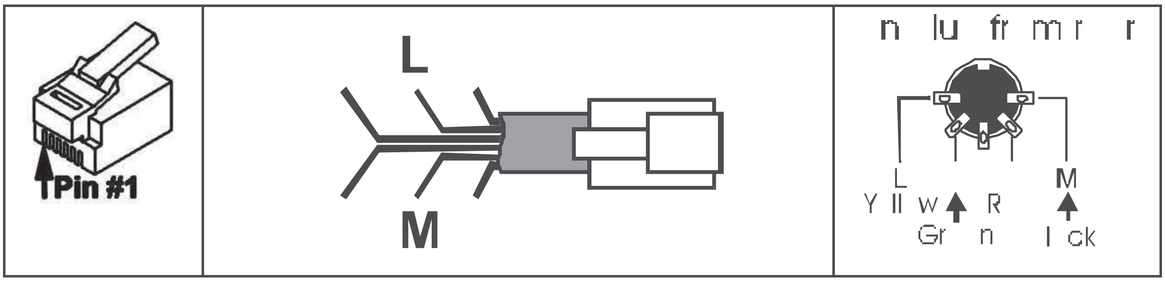
XpressNet uses 4 wires to interconnect the devices connected to the network. The wires used for XpressNet are labeled:

L - positive supply voltage for devices (12VDC)

M - supply voltage ground

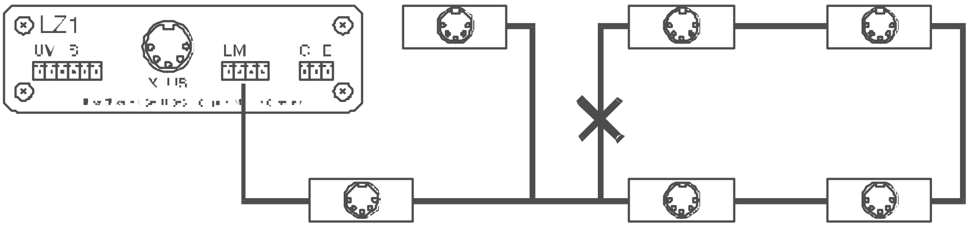
A - Receive/Transmit not inverting

B - Receive/Transmit inverting



Pin #	Description
Pin 1	not used
Pin 2	Ground "**M**"
Pin 3	- RS-485 "**B**"
Pin 4	+ RS-485 "**A**"
Pin 5	+12 volts "**L**"
Pin 6	not used

One must pay particular attention to the orientation of the **A** and **B**-lines. Exchanging these leads will result in no data communication. RS-485 normally has a distance limitation of 100 meters (300ft). However due to the XpressNet data transmission rate of 62.5k Baud (which is much less than the RS-485 limits), the connection between the devices on the network can be up to 1000m (3000ft) long. In addition this lower data rate removes the requirement that the network have a linear topology. XpressNet can have branch lines configured as a star or tree. For best network operation, a loop in the XpressNet wiring should be avoided. In the case of transmission problems with very extensive wiring or under unfavorable site conditions it may be necessary to use twisted wiring for the **A** and **B** connections and/or terminate the network with a 120W resistance. The command station LZ100 is delivered with a built in terminating resistor.



XpressNet devices can be powered directly from the XpressNet or powered through an external supply. The XpressNet device can be powered directly from the **L** and **M** 12 volts connections if the device draws less than 20mA when powered. This limit is to allow the bus to power multiple devices. If the devise requires more power than an external supply should be used.

1.1.3 Determining the version of the XpressNet

XpressNet is an evolution of the older X-Bus protocol but contains many more features and also allows for additional network operations. X-Bus V1 was supported in Command Stations up to Version 1.5. Command Stations up to Version 2.3 supported X-Bus V2. Other manufacturers including ZTC and Arnold/Rivarossi also used X-Bus V2. XpressNet is supported by systems starting with Version 3.0. Atlas, CVP, and Roco also currently support XpressNet. The version of the Command Station can be determined using the Command station status request. By convention versions greater than 20 implement X-Bus 2 commands while versions greater then 30 implement XpressNet commands. The new XpressNet 3.0 commands as well as the older X-Bus V1 and V2 commands are described within this document.

1.2 Receiving unrequested information

All XpressNet devices including the PC must be able to handle unrequested information. The Command Station may send unrequested information to any devices whenever conditions require, so those devices can react as fast as possible to specific events. Unrequested Information is either sent as a Broadcast transmission (if all XpressNet devices are to receive them) or as a directed message to a selected address (if the response concerns only a specific XpressNet device). XpressNet devices may receive these messages at any arbitrary time and must react accordingly. Unrequested information transmissions are:

• Broadcast of "Normal operation resumed"	(for all participants)
• Broadcast of "Track power Off"	(for all participants)
• Broadcast of "Emergency Stop"	(for all participants)
• Broadcast of "Service Mode entry”	(for all participants)
• Broadcast of "Feedback Broadcast"	(for all participants)

- Directed message of "Locomotive being operated by another device” (sent to the device who last controlling that locomotive)
- Directed Message of "Double Header occupied" (sent to the device, who last controlling that Double Header.)

The Command Station transmission "Transfer Errors", “Command station busy response”, “Instruction not supported by command station”, and “Double Header error response (X-Bus V1 and V2)” are not unrequested information, since these transmission can come in response to requests of a XpressNet device to the Command Station. They are thus transferred in response to an instruction to the command station.

1.3 Locomotive addresses in the digital plus system

Four-digit addressing was added in XpressNet V3.0. This has had consequences in the way that data communication on XpressNet occurs. The locomotive addresses 0 to 99, used prior to V3, can be transmitted in a single byte, which can take 256 different values (8 bits). This is not sufficient however for locomotive addresses from 0 to 9999, which require two bytes. The following views and computations are indicated in hexadecimals, since this permits a better representation.

In the NMRA specifications, short locomotive addresses and long locomotive addresses are differentiated. The short addresses (a single byte) have a range from 1 to 127 decimal. the long locomotive addresses (2 bytes) from 0 to 16383 decimal. The short addresses thus requires 7 bits to transmit, the long addresses requires 14 bits to be transmitted. The overlap between short and long addresses can become complicated and confusing for the user. To avoid this confusion XpressNet was designed so that the locomotive addresses from 1 to 99 are always short addresses and the locomotive addresses from 100 to 9999 are always long addresses. For operation and information instructions, address 0 is used to define the operation of a non-decoder equipped locomotive (controlled using analog DC).

The representation of the short addresses is simple: 0 to 99 decimal results in 0x00 to 0x63 hex.

With the long addresses the two highest bits of the High bytes are set to 1. I.e. the locomotive address with 14 bits length has an additional offset of 0xC000 hex. Thus: Locomotive address 100 to 9999 is 0xC064 to 0xE70F. Two byte are reserved in XpressNet commands for locomotive addresses.

In XpressNet V3, both long and short addresses use the same two bytes to transfer the address. In the following descriptions, a two-byte locomotive address (characterized through AH and AL for High and Low byte of the locomotive address), use the following convention:

- for locomotives with addresses from 0 to 99: AH = 0x0000 and AL = 0x0000 to 0x0063
  For example, a locomotive with address = 75 decimal means AH=0 and AL=0x4b

- for locomotives with addresses 100 to 9999: AH/AL = 0xC064 to 0xE707
  (= 0x0064 + 0xC000 to 0x270F + 0xC000) For example: a loco with address = 500 decimal means AH=0xc1 and AL=0xf4.

1.4 Programming considerations for XpressNet devices

Within the XpressNet architecture, each input devices such as hand held controllers, signal towers and computer interfaces that is connected to the network is responsible for its own interface and for adhering to the XpressNet protocol specification. This allows for the support of a wide range of devices and user interfaces without the need to change other devices on the network including the command station software.

Using XpressNet, one communicates to the command station information such as which locomotive is to receive a new speed, which switch is to be switched etc. The command station converts this information to the DCC track format and sends it to a power station. The power station then adds track power and places the signal on the track. The command station does this without knowing whether the decoder is present or not. Because of this the commands must be frequently repeated on the track to ensure that the decoder gets the command and is able to process it. The command station is responsible for maintaining the information that must be re-transmitted and for determining the frequency for retransmission. This essentially means that an XpressNet device need only send an instruction one time to the command station (e.g. locomotive #3 move forward at speed 10) and the command station sends this information to the track, so that the decoder can then process this message. In addition, the command station creates a data record for this locomotive instruction so that it can be sent again to the track as frequently as necessary. From the XpressNet equipment standpoint the control is thus simplified as only changes need to be sent to the command station. XpressNet equipment can also seek information on this data record (or parts of it) by sending an information request to the command station (e.g. what is the current speed and direction of locomotive #3).

Communications across the XpressNet network require that strict XpressNet timing requirements be adhered to. Each device connected to the network is responsible for maintaining these timing requirements.

XpressNet transfers information between devices connected to the network using packets of information. Each packet consists of a header byte, several data bytes and an error detection byte. Each byte contains 8 data bits.

The header byte consists of two parts with each part consisting of a 4-bit Nibble. The upper Nibble contains the function to be performed; the lower Nibble contains the number of data bytes in the packet. The number of data bytes in the packet does not include the header byte and the error byte. Thus the maximum number of bytes that can be in a single packet is 17, 15 data bytes plus one header byte and one error detection byte.

The data bytes complete the instruction begun with the header. The meaning of each data byte for each request and response is described in the following sections of this document.

The error detection byte forms the conclusion of the data communication and is transmitted as the last byte in the packet. It is formed using an X-Or-linkage of all preceding bytes inclusive of the header byte. It is the responsibility of each network device to examine whether the X-Or-linkage of all received bytes is identical to the error detection byte. If the X-OR received is different than the X-OR calculated than an error occurred during transmission and the packet should be discarded.

When developing a XpressNet device you should also be aware of the following conditions:

1. There are instructions that receive replies from the Command Station (e.g. requesting the Command Station status).
2. There are instructions that do not receive replies from the Command Station (e.g. locomotive control instructions).
3. The Command Station transmits instructions to the XpressNet device that were not requested by the device as a result of actions initiated by another device. Example: the “Emergency Stop” broadcast or information such as “Locomotive is being operated by another device”. This means that it is not possible to send an instruction and only receive and associated answer.

When selecting a new locomotive to control it is generally good practice to first request the current speed and direction of that locomotive. This allows the device to continue operating the locomotive with the same settings as it was previously operating in (speed, direction, functions).

On many systems, including the LZ100 and the LZV100, it is possible to operate the speed and direction of the MU consist from either the locomotive address or the consist address. This is because the command station maintains the relationship in its database. On entry level systems such as the compact there is no internal database and the speed and direction will only be effective at the consist address as per the NMRA RPs. When operating an MU function control is only available at the locomotive address.

When an event causes the command station to broadcast an Emergency Stop or Track power Off message, the comand station will broadcast this event 2 or 3 times to ensure that all conected XpressNet devices see the event. After sending this broadcast the comand station wil continue to provide transmission windows to connected XpressNet devicses to allow them to alter the events that led up to the event from happening again. For example sending revised speed and or direction commands to prevent a collusion. These new comands are not sent to the track until a normal operation has resumed and a “Normal operation resumed“ broadcast is transmitted. Because of this all XpressNet devices must take care to check the status of the system each time they power up or are reconnected to XpressNet.

The XpressNet architecture allows any device to gain control of any locomotive at any time. If an XpressNet device sends an operation command to a locomotive (for example: speed, or function) that is currently being controlled by another XpressNet device, the comand dtation will transmit this new comand to the track. The XpressNet device that last controlled the locomotive will receive a "Locomotive is being operated by another device" unrequested message. Following is an example on how a device should function in such an environment:

1. request loco information and display it (speed, function).
2. If information "locomotive is being controled by another device" is received, blink with display.
3. Wait for a key to be pressed (do nothing until key pressed). and check receive buffer for unrequested "locomotive is being controled by another device" message.
4. If unrequested message is received, go to step 2 above
5. If key is pressed to control locomotive and the locomotive is not being controlled by another device, send the new information (e.g. new speed ot function).
6. If key pressed and loco is being operated by another device either ask the user if they wish to take control and , goto 1. or take control by sending an operations command and go to step 1.

One additional important note: many of the new XpressNet V3 responses from the command station no longer contain the address of the locomotive that the response pertains to. The reason is that these responses can only follow a specific request that does include the locomotive address. This request/response sequence must be considered in an XpressNet device's design. For example, one should avoid having an output-area of a program for a locomotive request that is different from an input area, so that the sequencing of messages is not confused and so the program can understand the unrequested information and how it relates to ongoing operations (such as the message that locomotive control for a specific address has been taken over by another XpressNet device which has changed the speed or direction of the locomotive).

1.5 LI100 Computer Interface Specifications

The LI100, LI100F, and LI101F computer interfaces are special XpressNet devices. These computer interfaces have the responsibility for handling low level XpressNet communications while exporting the higher level XpressNet requests and responses to the PC. The LI100 communicates with the PC over a serial interface.

Since the structure of XpressNet requires precise timing of the network participants and the transmission of data does not correspond to the usual PC style of communications, an interface is needed between the PC and XpressNet. In terms of a network, the LI100 is the actual XpressNet participant, and is responsible to ensure that the necessary timing and communications take place over the network. The LI100 is also responsible for the serial communication with the PC that allows the PC to function independently from the actual network. The LI100 interface is responsible for passing instructions received from the PC on to other network devices (which for the most part will be the command station) and for passing on to the PC any answers or instructions it receives from the network. Because the LI100 deals with XpressNet communications, the Call byte is not exchanged between the PC and the LI100.

The interface between the LI100 and the PC is through a standard serial interface with the following characteristics.

- Baud rate: 9600 baud (LI100),
  9600 baud or 19200 baud (LI100F),
  19200 baud, 38400 baud, 57600 baud or 115200 baud (LI101F)

- 8 data bits, 1 start bit, 1 stop bit, no parity bit

- CTS hardware handshake (the LI100F and LI101F do this in hardware, the LI100 in software)

The hardware handshake used allows the LI100F/LI101F to control the flow of information from the PC. The LI100F/LI101F uses the CTS signal to permit or stop the data communication from the PC. The RTS signal of the PC is not observed. The PC is responsible for handling the flow of information from the LI100F/LI101F at its full data transmission rate so that the LI100F/LI101F need not store the information it receives from the network.

1.5.1 Differences between the LI100, LI100F, and LI101F

The LI100, LI100F, and LI101F all use the same XpressNet instruction set as contained in Section 2 of this document. All instructions except for those contained within this section are handled transparently, i.e. without each further examination, are passed on between PC and the command station or other XPressNet device. The basic differences between the interfaces are in the RS232 capabilities and the method used to set internal device parameters.

The LI100 and LI100F use internal dip switches to manually set the allowed configurations. The LI101F uses a series of special requests contained in the following sections for configuration. The LI100 uses software flow control while the LI100F and LI101F use hardware flow control,

The LI101F also only supports genuine RS232- interfaces which exhibit the standardized voltage levels and handshake lines. USB serial adapters do not meet these requirements and should not be used with the LI100F.

1.5.2 Messages from the LI100

XpressNet instructions are divided into two categories: requests for information (a specific answer is expected), and instructions (which do not entail a command station answer). In order for the PC program to be able to determine which messages were successfully received by the command station, the LI100 acknowledges all messages that do not otherwise entail a command station response.

The following messages are generated by the LI100, and sent to the PC, when no command station answer is available. The messages in the following table are in decimal representation.

Header byte	Message	X-Or	Meaning
01	01	00	Error occurred between the interfaces and the PC (Timeout during data communication with PC)
01	02	03	Error occurred between the interfaces and the command station (Timeout during data communication with command station)
01	03	02	Unknown communications error (command station addressed LI100 with request for acknowledgement)
01	04	05	Instruction was successfully sent to command or normal operations have resumed after a timeout
01	05	04	The Command Station is no longer providing the LI100 a timeslot for communication
01	06	07	Buffer overflow in the LI100

The specific meaning of each of these LI100 message follows:

01 / 01 / 00: The first byte of the packet sent by the PC to the LI100 specifies the number of bytes that are in packet to be transmitted. If this number of bytes is not received by the LI100 before its timeout occurs, this error message is sent back to the PC.

01 / 02 / 03: The response by the command station to an instruction sent to it must also take place within a certain time, otherwise this error message is sent to the PC.

01 / 03 / 02: If a data transmission error occurs during a data communication to the command station, then the XpressNet equipment that had the transmission error is again addressed and must acknowledge

receipt of this error. The LI100 also transmits this error message to the PC. If this condition arises frequently, then the wiring should be examined. This message can also arise, if errors were made with the computation of the x-Or-byte.

01 / 04 / 05: This message is sent to the PC whenever an instruction, which the LI100 sent to the command station, did not require an answer (e.g. a locomotive speed and direction operations request). This message also serves as a confirmation that communications has resumed after a timeout with the command station had occurred.

One important consideration should be noted. It is impossible for the PC (or LI100) to determine that a specific DCC packet has actually been transmitted to the track by the command station and acted upon by the decoder. The message "instruction successfully sent" only means only that the associated instruction was successfully sent to the command station.

01 / 05 / 04: The command station provides a time slot for communication to each XpressNet attached device in a certain time interval. If this does not occur, this message is sent to the PC. The CTS signal is then also taken away by the LI100. Once the command station returns to providing transmission windows to the LI100, then this message is confirmed to the PC with the before described instruction and the normal operation can resume.

01 / 06 / 07: A data buffer is maintained by the LI100 so that no information transmitted by the command station is lost. If this buffer cannot be emptied, then this message is sent to the PC. If the PC receives this message, one should assume that one or more data records were lost. If necessary this information must again be requested by the PC.

The XpressNet V3 LI100 responses described above differ partially from the messages used prior to V3. Existing PC programs must be adapted to process these new LI100 messages.

1.5.3 Special PC programming considerations

For the most part a PC application must be designed in the same way as any other XPressNet device with one exception. Communications across the XpressNet network require that strict XpressNet timing requirements be adhered to. A PC program cannot easily satisfy these strict interface requirements because the program rarely has direct access to the interface hardware. To greatly simplify PC programming and to maintain the reliability of the XpressNet network, the LI100 interface provides the gateway to XpressNet. The LI100 takes over the time-critical portion of XpressNet communications and utilizes the standard asynchronous communication with hardware handshake over a serial interface to communicate with the PC. This standard serial interface approach is supported by almost all PC programming software. But this does not mean that the model railroad layout can be controlled as fast as the high computing power of today's PC´s will allow. In most cases the limiting factor is the DCC data transmission to the track. Every network-attached device (including the PC) must be aware of this. The programmer of PC software must always have in mind that it the software is about to control a real time system in which a given command must be checked to ensure that it was accepted. The LI100 controls the data flow from the PC by the use of the CTS hardware flow control.

The information transmitted across the RS-485 network differs in timing and some header information than the information transmitted across the RS-232 interface to the PC. The LI100's role is to provide this translation and act as a bridge between these two different communication media. The LI100 provides network transparency for the PC program.

Prior to the LI100F, the LI100 interface only operated at 9600 bits/second. The LI101F increased this to 19.2 thousand bits/second. Starting with the LI101F this baud rate of has been increased to up to 115.2 thousand bits/second. While this data rate may appear relatively slow, it is sufficient for the purposes of NMRA DCC control. For comparison: XpressNet works at 62.5 thousand bits/second, to ensure that the network can service all devices with excellent performance. However the NMRA DCC effective data rate to the track is approximately 4500 bits per second for 0 bits and 9600 bits/second for 1 bits. The faster baud rate of the LI101F is to allow the computer to both interact with the command station and other devices on the XpressNet simultaneously,

In order to ensure that an instruction from the PC will be received by the LI100 (i.e. that it will be read from the PC send buffer), it is important to check that the CT signal is active and if there is enough space in the buffer. In order to avoid receiving a buffer overflow, the program must ensure that bytes are always received from the interface module and, for example, transferred into a software-FIFO. The actual receive buffer of a PC UART component is only a few bytes long. An overflow will develop quickly if unanticipated information arrives but is not retrieved due to this omission in the code.

When developing a PC program you should also be aware of the following conditions:

1. There are instructions that receive replies from the command station (e.g. requesting the command station status).
2. There are instructions that do not receive replies from the command station (e.g. locomotive control instructions).
3. The command station transmits instructions to the PC that were not requested by the PC as a result of actions initiated by another device. Example: the “Emergency Stop” broadcast or information such as “Locomotive is being operated by another device”.

The LI100 transmits back to the PC the command confirmation 1/4/5 when an instruction is properly received and confirmed by the Command Station. An instruction is considered to be successfully sent if the acknowledgement arrives with the expected answer, an error message or unsolicited information. If instruction sent by the PC is not accepted or properly received by the LI100, the LI100 transmits back to the PC the associated error (e.g. “Command Station Busy” or “Transfer Error”) instead of the LI100F confirmation, 1/4/5, and/or the expected response (e.g. locomotive information). An additional exception exists for the creation of the LI100F confirmation, 1/4/5. This happens when the LI100F does not receive; in it’s receive buffer, an anticipated answer from the Command Station to the instruction sent by the PC. There may be a delay for a few milliseconds between sending the instruction to the command station and the LI100F being able to determine if the instruction was successfully transmitted. During this time another device may cause a broadcast or other information to be transmitted to the LI100F which the LI100F places in its buffer for transmission to the PC which must not be interpreted as an answer to the proceeding instruction. The 1/4/5 command confirmation is omitted if this occurs. This must always be taken into consideration.

In the current system, it is not possible to send an instruction and only receive and associated answer. This inevitably results in a buffer overflow, which causes unpredictable results in the PC application software.

1.5.4 Determining the Version number of the LI100F and LI101F

The determination of the version number of the LI100F and LI101F is an action which takes place only between the PC and LI100. The instruction structure and the associated LI100 response correspond to the XpressNet format described in chapters 2.

Instruction for the determination of the version and code number:

	Header byte	**X-Or-byte**
Binary:	1111 0000	1111 0000
Hex:	0xF0	0xF0
Decimal:	240	240

The response from the LI100:

	Header byte	**Data Byte 1**	**Data Byte 2**	**X-Or-byte**

Binary:	0000 0010	VVVV VVVV	CCCC CCCC	X-Or-byte
Hex:	0x02	VV	CC	X-Or-byte
Decimal:	2	VV	CC	X-Or-byte

Description:

VV indicates the hardware version number of the LI100 indicates in hexadecimals at BCD representation. CC indicates the software release number of the LI100 in hexadecimals at BCD representation.

Example:

The PC sends the command: 0xF0 0xF0
The LI100 answers: 0x02 0x30 0x01 0x33
This means that the LI100 hardware version is 3.0, and that the LI100 software version is 01

The possible responses are

LI100F: hardware version 3.0, software version 01 or 02

LI101F: hardware version 1.0, software version 01 (or others if changes are made in future)

Comments:

This instruction is not supported by the LI100. It is one of the series of instructions that the LI101F and LI101F evaluates internally. All other traffic, which does not correspond to one of these instructions, is passed on by the LI100 to the command station and its contents are not examined by the LI100.

1.5.5 Determing and changing the XpressNet address for the LI101F

The LI100 and LI100F have their XpressNet address set via an internal dip switch. Starting with the LI101F the XPressNet address is set in software. This is an action, which takes place only between PC and LI101F. The instruction structure and the associated LI101F response correspond to the XpressNet format described in chapters 2.

Instruction for setting the LI101F's XpressNet address:

	Header byte	Data Byte 1	Data Byte 2	X-Or-byte
Binary:	1111 0010	0000 0001	ADR	X-Or-byte
Hex:	0xF2	0x01	ADR	X-Or-byte
Decimal:	242	1	ADR	X-Or-byte

The response from the LI101F:

	Header byte	Data Byte 1	Data Byte 2	X-Or-byte
Binary:	1111 0010	0000 0001	ADR	X-Or-byte
Hex:	0xF2	0x01	ADR	X-Or-byte
Decimal:	242	1	ADR	X-Or-byte

Description:

Addr in Data Byte 1 of the request indicates the XpressNet device address, which is the LI101F should use. The permitted range of XpressNet addresses lies between 1 and 31 Decimal. The actual XpressNet address currently being used by the LI101F is contained in Data Byte 2 of the LI101F response. Usually the XpressNet address transmitted in the request and received in the response are identical.

Comments:

If the device address transmitted in Data Byte 2 of the request is not within the range of 1 to 31, then the LI101F answers with its XpressNet address currently in use. This allows the address to be determined, without changing it.

This instruction is not supported by the LI100 and LI100F. It is one of the series of instructions that the LI101F evaluates internally. All other traffic, which does not correspond to one of these instructions, is passed on by the LI101F to the command station and its contents are not examined by the LI101F.

1.5.6 Determing and changing the Baud Rate for the LI101F

The LI100F has its baud rate set via an internal dip switch. Starting with the LI101F the XPressNet address is set in software. This is an action, which takes place only between PC and LI101F. The instruction structure and the associated LI101F response correspond to the XpressNet format described in chapters 2.

Instruction for setting the LI101F Baud Rate:

	Header byte	Data Byte 1	Data Byte 2	X-Or-byte
Binary:	1111 0010	0000 0010	BAUD	X-Or-byte
Hex:	0xF2	0x02	BAUD	X-Or-byte
Decimal:	242	2	BAUD	X-Or-byte

The response from the LI101F:

	Header byte	Data Byte 1	Data Byte 2	X-Or-byte
Binary:	1111 0010	0000 0010	BAUD	X-Or-byte
Hex:	0xF2	0x02	BAUD	X-Or-byte
Decimal:	242	2	BAUD	X-Or-byte

Description:

In the request BAUD gives codes the desired new transmission Baud rate, which the LI101F should begin using for communication over the serial interface. The coding reads as follows:

BAUD = 1:	19200 baud (default baud rate)
BAUD = 2:	38400 baud
BAUD = 3:	57600 baud
BAUD = 4:	115200 baud.

The technical manual of the LI101F should be checked to determine if the specific range of possible baud rates is fully supported.

Comments:

This instruction is not supported by the LI100 and LI100F. It is one of the series of instructions that the LI101F evaluates internally. All other traffic, which does not correspond to one of these instructions, is passed on by the LI101F to the command station and its contents are not examined by the LI101F.

If the Baud rate value specified in Data Byte 2 of the request is within the range 1 to 4, then the LI101F will sends the response using the old existing Baud rate. The LI101F will only switch to the new baud rate upon completion of this response. A value outside of this range is not defined and should not be used. See manual for the LI101F for more information.

While 115200baud is possible, its use may require a too high a time requirement by the PC during the operation of the serial interface. 57600baud is therefore recommended .

1.5.7 BREAK signal and RESET of the LI101F

If the PC and the LI101F have different baud rates, then they will not be able to communicate. If this occurs the PC can reset the LI101F to its default settings by transmitting a BREAK signal. The BREAK signal must be sent for a duration of at least 200 milliseconds. It is independent of the Baud rate. The LI101F receives the byte value 0 with a framing error and after a further timing check it performs an equipment RESET. This reset causes the transmission Baud rate of the LI101F to be set to 19200baud. Following the transmission of the BREAK signal, the PC program can begin communication with the LI101F at 19200 Baud. The recommended proceeding at the start of the PC program is:

- transmit BREAK signal (the LI101F will have its baud rate set to19200baud)
- determine the version number and address of the computer interface (using 19200baud)
- if no response try communicating at 9600 baud (for use with LI100 and LI100F)
- if connected to an LI101F then switch to the desired baud rate (if not equal 19200
- determine the command station status and version etc. (using new Baud rate)

1.6 Future Expansion

XpressNet 3.0 represents a major architectural advancement and has also opened up the ability for much additional improvements such as the accommodation of RailCom.

In the time since X-Bus was first designed, DCC has also been expanded to include 128 speed steps and extended 4 digit addresses. This has necessitated changing the network protocol to accommodate these extended features. For reasons of clarity and data consistency and to simplify future versions some Xbus V1. and V2 commands are considered legacy and may no longer be supported in all command stations. Starting with XpressNet 4.0 support for these older commands will likely be terminated completely. Because of this, PC applications and other XpressNet devices should transition to the newer XpressNet V3 command structure.

The following are considered legacy commands.

Requests transmitted by XpressNet devices:

1.) “Emergency stop a locomotive (X-Bus V1 and V2)” (see 2.2.5.1)
2.) “Emergency stop selected locomotives (X-Bus V1 and V2)” (see 2.2.6)
3.) “Locomotive information requests (X-Bus V1)” (see 2.2.19.1)
4.) “Locomotive information requests (X-Bus V1 and V2)” (see 2.2.19.2)
5.) “Locomotive operations (X-Bus V1)” (see 2.2.20.1)
6.) “Locomotive operations (X-Bus V2)” (see 2.2.20.2)
7.) “Double Header operations (X-Bus V1 and V2)” (see 2.2.21)

Responses from the Command Station

8.) “Software-version(X-Bus V1 and V2)” (see 2.1.6.1)
9.) “Locomotive information response (X-Bus V1)” (see 2.1.12)
10.) “Locomotive information response (X-Bus V2)” (see 2.1.13)
11.) “Double Header information response (X-Bus V1)” (see 2.1.18)
12.) “Double Header information response (X-Bus V2)” (see 2.1.19)
13.) “Double Header error response (X-Bus V1 and V2)” (see 2.1.20)