LYNX-NET PROTOCOL SPECIFICATION

Version 2.01

PO BOX 950940
LAKE MARY, FL 32795
407-323-4467
www.marrickltd.com

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

OVERVIEW

The LynX-NET™ Protocol is utilized by all next generation Marrick Limited products. This new protocol was designed to enhance the communications integrity between the Marrick devices and their hosts. It also provides a wide range of enhanced functions, which have recently become available. The protocol is expandable and allows for customization as required. It is layered and object oriented which provides an easy model for implementation.

Every LynX-NET™ packet contains a network identifier, a Node ID, a sequence byte, a length byte, a data field and a frame check sequence byte. The packets of a LynX-NET™ interface are variable length based on the payload in the data field. The payload varies according to the target network identified in the first byte. See figure 1 below.

FIGURE 1 – LynX-NET™ Packet Format

NETWORK ID BYTE	NODE ID BYTE	SEQUENCE BYTE	LENGTH BYTE	DATA FIELD	FCS BYTE
0x00 Packet NAK 0x01 Packet ACK 0x02 Unsupported NET 0x03-0x0F Reserved 0x10 X-10 0x11 LynX-NODE 0x12 Custom 0x13 CEBus 0x14 LonWORKS 0x15 BACNet 0x16-0xDF Reserved 0xE0 LynX-NET Cmd 0xE1-0xFF Reserved	The Node ID is between 0x00 and 0xFF. This value indicates which interface on the network is the receiver or originator. 0x00 Default Node 0x01-0xFE Assigned by hardware. 0xFF Broadcast	The SEQUENCE byte is an arbitrary number that is assigned to the packet for identification and ordering. All upstream traffic (host to network) use numbers 0-127 (0x00-0x7F). All down stream traffic (network to host) use 128-255 (0x80-0xFF).	The LENGTH byte represents the packet data field length. The network ID, Node ID, and sequence bytes including the FCS are not counted. The minimum length is 0 for an empty data field.	The DATA field can be anything and is sent to the network as is (right or wrong…). No checking is done this field at this level. Each network type will have a firmware stack to handle the message in this field. If the network type is unrecognized or unsupported by the hardware, the packet is rejected at this level.	The FCS or Frame Check Sequence is the binary sum of all the packet bytes. Add the network, length, and sequence bytes together with all of the data bytes and then truncate to 8 bits. The result goes in this field.

NETWORK ID BYTE

The NETWORK ID field is 1 byte long and identifies the target network for the payload. It also identifies to the sender whether the packet was received properly or not. This is based on an ACKNOWLEDGED (ACK) or NOT ACKNOWLEDGED (NAK) response from the receiver. The sender is considered the device originating the transmission and the receiver is considered the target of the packet. Packets can originate from a gateway, bridge, remote device, or a host. All packets must be acknowledged by the receiver with either an ACK or NAK. Packets that do not get acknowledged should be retransmitted after the maximum network transport time has elapsed. The maximum network transport time is generally defined by the host-to-network interface. The acknowledgement guarantees only the delivery of the message and not the integrity or validity of the payload (data field). The upper layers will determine whether the payload was valid and respond accordingly. Table 1 below shows the definitions of the network field and their use.

TABLE 1 – LynX-NET™ Network definitions

VALUE	DESCRIPTION	USE
0x00	Packet NAK	Packet has failed the Frame Check Sequence (FCS)
0x01	Packet ACK	Packet successfully received by target
0x02	Unsupported NET	Reported by receiver if target network not supported
0x03	Unsupported Node ID	Reported by receiver if target interface is not supported
0x04-0x0F	(Reserved)	For future use
0x10	Protocol, X-10	Payload carries encoded X-10 commands
0x11	Protocol, LynX-NODE™	Payload carries LynX-NODE device commands
0x12	Protocol, Custom	Payload carries custom data or unknown protocol.
0x13	Protocol, CEBus	Payload carries CAL message
0x14	Protocol, LonWORKS™	Payload carries LonWORKS message
0x15	Protocol, BACNet	Payload carries BACNet encoded commands
0x16-0xDF	(Reserved)	For future use
0xE0	LynX-NET™ Command	Network Interface command protocol
0xE1-0xFF	(Reserved)	For future use

NODE ID BYTE
The Node ID is a number between 0x00 and 0xFF, which identifies which interface within a multiple interface device should receive the packet. The number is assigned by the hardware and can be found by interrogating any interface controller on a port with a LynX-NET Command sequence (See Network ID 0xE0).

SEQUENCE BYTE
The sequence byte provides a method of tracking packets and is automatically generated by the transmitter using the following rules. All upstream traffic (host to network) uses sequence numbers from 0-127 (0x00-0x7F) and all downstream traffic (network / gateway to host) uses sequence numbers from 128 to 255 (0x80-0xFF). This allows a method for identifying which way the packet is going and to eliminate host / target confusion in cases of identical sequence numbers. The starting number is arbitrary but must be within the range for the direction the packet is traveling.

LENGTH BYTE
The length byte is the length of the data field only. All packets must contain and Network ID Byte, a Node ID, a Sequence Byte, and a Frame Check Sequence Byte. The data field is optional and variable in length. The length byte is 0 if no data is present.

DATA
This field depends on the target network. See individual sections for more detail.

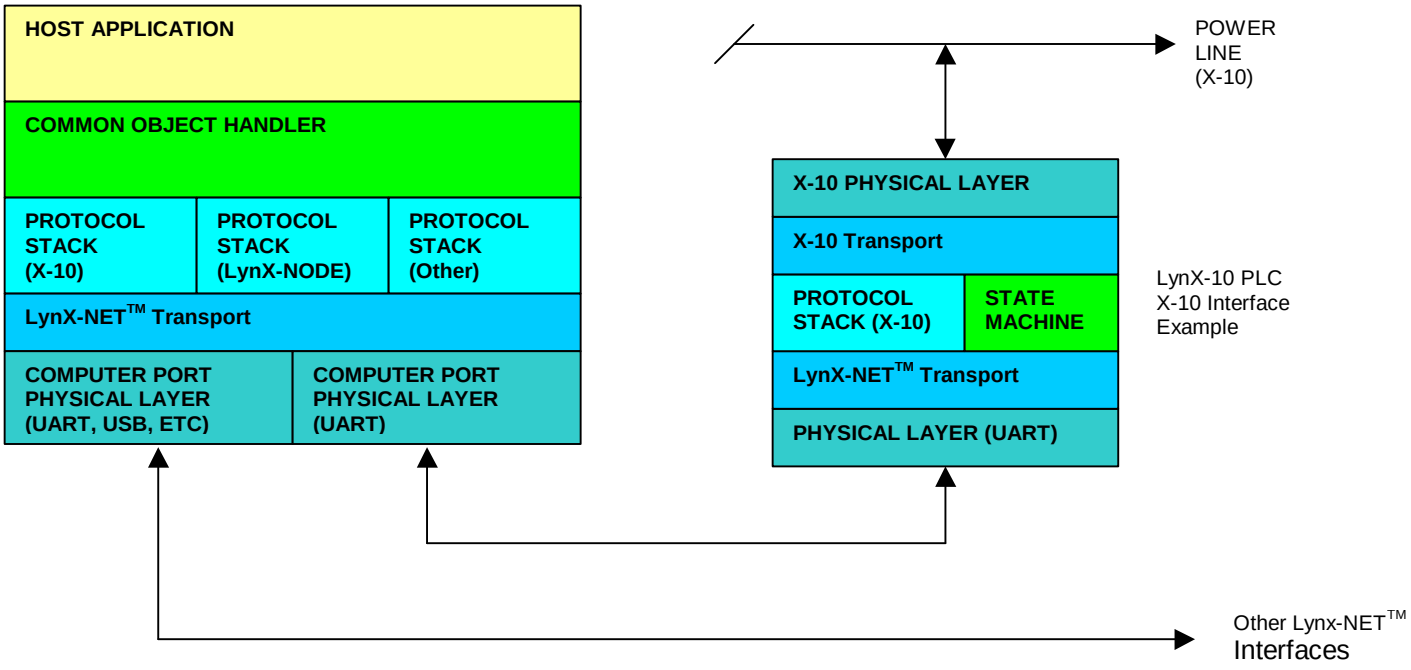
FRAME CHECK SEQUENCE (FCS)
The frame check sequence is used to insure that the all the bits in the packet were received correctly. The FCS is calculated by taking the binary sum of all the packet bytes (NETWORK, NODE ID, SEQUENCE, LENGTH, and DATA) and truncating to 8 bits. For example, take the string:

0x10, 0x00, 0x34, 0x03, 0x10, 0x00, 0xFF, FCS

This is an X-10 network command to turn all lights off on house code A (see Network ID, X-10 for more information). The FCS value will be 0x10+0x00+0x34+0x03+0x10+0x00+0xFF=0x156. To fit into the single byte FCS the value is truncated to 0x56. So the entire transmitted packet from the host to the X-10 interface controller in Node 0 will be:

0x10, 0x00, 0x34, 0x03, 0x10, 0x00, 0xFF, 0x56

Figure X – LynX-NET™ Network Structure



LYNX-NET™ PROTOCOL PAYLOADS

The payload or data field contains the actual commands or data to or from the target network device. Several are predefined, however, there are many reserved codes for future expansion. This document will describe the most common and will be expanded to cover more in time.

NETWORK ID 0x10 – X-10 PAYLOADS

X-10 is a command oriented packet structure, which provides a simple method of controlling lights and appliances. Figures 2 through 4 show all available X-10 commands as of the release of this document. They are coded into a packet structure to provide an easy method of transport. The X-10 network interface (i.e. a LynX-10 PLC) is responsible for translating this payload into actual X-10 power line carrier data and visa versa.

NETWORK ID 0x11 – LynX-NODE PAYLOADS

LynX-NODE is a command oriented packet structure, which provides a simple method of controlling a wide range of wired devices. The packet includes a source and destination address within like devices which allows intra-device communication. Addressing is set automatically via an isolation algorithm. The physical layer is based on RS-485; however, the packets can be encapsulated into other transport protocols and moved over any physical layer, including the Internet. Figure X and Y shows the available commands and their implementation.

NETWORK ID 0x12 – CUSTOM PAYLOADS

This payload is used for passing custom protocols or raw data to another interface contained in a device. For instance, if data is to be passed to an RS-485 port on a LynX-PORT II without any Marrick protocol encoding, this ID would be used. This is useful when attaching non-standard or custom devices to additional ports on any LynX-NET based system to allow custom coding schemes.

NETWORK ID 0xE0 – LynX-NET™ COMMAND PAYLOADS

The LynX-NET™ protocol provides a method for maintenance of attached network interface devices. This can include the enumeration of attached interfaces, resetting of attached devices, hardware diagnostics, and extended interface options. For example, the RS-232 interface speed is factory set to 1200 baud on the LynX-10 PLC. It can be changed using the network command protocol by directly accessing the internal interface registers.

LYNX-NET™ PROTOCOL – X-10 PAYLOAD DESCRIPTION

The following figures show the standard, extended, and Marrick enhanced X-10 commands and structures. The standard legacy X-10 commands are shown in figure 2. These include ON, OFF, DIM, BRIGHT, etc. Extended X-10 commands are shown in Figure 3. These include many new features now supported by X-10 devices. Figure 4 shows additional X-10 commands added by Marrick Limited.

FIGURE 2 – STANDARD X-10 COMMANDS

COMMAND	HOUSE CODE	UNIT CODES	DATA
0x00 CMD FAILURE 0x01 CMD SUCCESS 0x02 STATUS 0x03 (Reserved) 0x04 MONITOR DATA 0x05 ANALYZER DATA 0x06-0x07 (Reserved) 0x08 X-10 ADDRESS 0x09-0x0F (Reserved) **STANDARD X-10** 0x10 ALL UNITS OFF 0x11 ALL LIGHTS ON 0x12 ON 0x13 OFF 0x14 DIM 0x15 BRIGHT 0x16 ALL LIGHTS OFF 0x17 EXT. CODE 0x18 HAIL REQ 0x19 HAIL ACK 0x1A PRESET DIM 0 0x1B PRESET DIM 1 0x1C EXT DATA XFER 0x1D STATUS ON 0x1E STATUS OFF 0x1F STATUS REQ **ENHANCED X-10** 0x20 UNITS OFF ALL H/C 0x21 LIGHTS OFF ALL H/C 0x22 LIGHTS ON ALL H/C 0x23-0x27 (Reserved) 0x28 DIM REL n 0x29 DIM PRESET n 0x2A-0x2F (Reserved) **OTHER X-10** 0x30 (Reserved) 0x31 Extended Code 1 0x32 Extended Code 2 0x33 Extended Code 3 0x34 Extended Code 4 0x35-0x7F (Reserved)	0x00-0x0F Valid HC 0x10-0xFE (Reserved) 0xFF No HC This byte is the House code of X-10 units to be commanded or sent data. In traditional X-10 these are the letters A-P. They are mapped here as 0-F hex. If house code is not used, use 0xFF for NO HOUSE CODE and skip UC.	0x00-0x0F Valid UC 0x10-0xFE (Reserved) 0xFF EOL A list of unit codes to be addressed before the command is sent. The End Of List (EOL) token is used to denote the start of the next field. This field is optional depending on the command, however, the EOL token must always be present.	The Data field contains any associated information required by the current command. This can include things such as dim levels or other data.

FIGURE 3 – EXTENDED X-10 COMMANDS

COMMAND	HOUSE CODE	UNIT CODES	DATA

SHUTTERS AND SUNSHADES (TYPE 0)

0x80	(Reserved)
0x81	OPEN TO LEVEL n, ENABLE SUN PROTECTION
0x82	LIMIT OPENING TO LEVEL n, ENABLE SUN PROTECTION
0x83	OPEN TO LEVEL n, DISABLE SUN PROTECTION
0x84	OPEN ALL SHUTTERS FULL ON HOUSE CODE n, DISABLE SUN PROTECTION
0x85	OPEN ALL SHUTTERS FULL ALL HOUSE CODES, DISABLE SUN PROTECTION
0x86	(Reserved)
0x87	INCLUDE THIS UNIT IN LIFESTYLE MODE n
0x88	BEGIN LIFESTYE MODE n
0x89	EXCLUDE THIS UNIT FROM LIFESTYLE MODE n
0x8A	EXCLUDE THIS UNIT FROM ALL LIFESTYLES
0x8B	CLOSE ALL SHUTTERS THIS HOUSE CODE, ENABLE SUN PROTECTION
0x8C	CLOSE ALL SHUTTERS ALL HOUSE CODES, ENABLE SUN PROTECTION
0x8D	(Reserved)
0x8E	SELF TEST SHUTTER CONTROLLER - MECHANICAL
0x8F	SELF TEST SHUTTER CONTROLLER - ELECTRICAL, MECHANICAL

SENSORS (TYPE 1)

0x90	(Reserved)
0x91	REQUEST AVERAGE LIGHT DATA FROM UNIT
0x92	REQUEST CURRENT TEMPERATURE FROM UNIT
0x93	REQUEST STATUS FROM UNIT
0x94	REQUEST CURRENT LIGHT DATA FROM UNIT
0x95	REQUEST AVERAGE TEMPERATURE FROM UNIT (16 MIN AVERAGE)
0x96-0x9A	(Reserved)
0x9B	AMBIENT LIGHT DATA FROM UNIT (RESPONSE TO REQUEST)
0x9C	TEMPERATURE DATA FROM UNIT (RESPONSE TO REQUEST)
0x9D	STATUS DATA (RESPONSE TO REQUEST)
0x9E-0x9F	(Reserved)

SECURITY (TYPE 2)

0xA0-0xAF (Reserved)

DIMMERS AND APPLANCES (TYPE 3)

0xB0	INCLUDE UNIT IN GROUP m AT CURRENT LEVEL/SETTING
0xB1	PRESET LEVEL n
0xB2	INCLUDE UNIT IN GROUP m AT LEVEL n
0xB3	ALL UNITS ON THIS HOUSE CODE
0xB4	ALL UNITS OFF THIS HOUSE CODE
0xB5	REMOVE UNIT FROM GROUP m
0xB6	EXECUTE GROUP FUNCTION
0xB7	REQUEST OUTPUT STATUS (UNIT OR GROUP)
0xB8	OUTPUT STATUS ACK (REPLY TO UNIT REQUEST)
0xB9	OUTPUT STATUS ACK (REPLY TO GROUP REQUEST)
0xBA	OUTPUT STATUS NAK (NO UNIT IN GROUP)
0xBB	CONFIGURE MODULES THIS HOUSE CODE
0xBC-0xBF	(Reserved)

RESERVED TYPES (TYPE 4-6)

FIGURE 4 – MARRICK EXTENDED X-10 COMMANDS

COMMAND	DATA

MARRICK EXTENSIONS (TYPE 7)

0xF0	READ / WRITE X-10 OPTIONS
0xF1	READ TIME DURATION SINCE CARRIER PRESENT
0xF2	READ STATISTICS COUNTER n
0xF3	CLEAR STATISTICS COUNTERS
0xF4-0xFB	(Reserved)
0xFC	READ / WRITE RECEIVER SENSITIVITY (0xFF = HIGHEST)
0xFD	READ / WRITE TRANSMITTER POWER (0xFF = HIGHEST)
0xFE	SELECT X-10 CHANNEL (0x00=NORMAL, 0xFF=ALTERNATE, 0x01 through 0xFE=RESERVED)
0xFF	RAW DATA TO POWER LINE (BYTE 1 = LENGTH OF RAW DATA THAT FOLLOWS – UP TO 16 BYTES)

X-10 COMMAND PROTOCOL OVERVIEW

GENERAL PROTOCOL

CODE	DESCRIPTION AND USE
0x00	**COMMAND FAIL** This is returned by the X-10 network interface if the requested command times out or for any reason cannot be completed. The UC, HC, and Data fields are not used. The interface must use the original sequence number when rejecting a command so it can be retransmitted. Immediately following the COMMAND FAIL code is the FAILURE TYPE CODE which is shown below. Payload Example: 0x00 0x01 Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x52, 0x02, 0x00, 0x01, 0x64 **FAILURE CODE** 0x01 Unsupported X-10 command 0x02 Insufficient data for command 0x03 Bad command format (order of data, too much data) 0x04 Missing data delimiter in packet (0xFF missing between command and data) 0x05 Data value out of range 0x06 Transmission failure (too many attempts, X-10 command discarded) 0x07-0x0F (Reserved) 0x10 X-10 buffer overflow (commands still running without room for more). 0x11-0xFE (Reserved) 0xFF Unknown internal failure (Should issue reset command when this occurs)
0x01	**COMMAND SUCCESS** This is returned after successful completion of the command. This means that the interface has successfully transmitted the X-10 command to the target device without detectable collisions or errors. Payload example: 0x01 Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x52, 0x01, 0x01, 0x64
0x02	**STATUS** This code is returned if the X-10 protocol engine needs to report activity. This can include events such as asynchronous errors, reception errors, FIFO levels, etc. The code is followed by a status identifier, which indicates the issue to the host. A unique sequence number will be used for each status report. This code only originates from the interface's X-10 sub-system and should not be issued from the host. **STATUS CODES** 0x00 X-10 buffer level has fallen below threshold point (OK to resume X-10 transmissions) 0x01 X-10 buffer level has exceeded threshold point (hold any X-10 commands) 0x02-0x0F (Reserved) 0x10 Receiver decoder error (incoming signal not properly encoded or interference) 0x11 Collision detected (signal detected during transmission) 0x12-0x1D (Reserved) 0x1E X-10 Communications On-line 0x1F Power Failure (May or may not be received due to baud rate) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0xAA, 0x02, 0x02, 0x01, 0xBF (The X-10 command FIFO threshold has been exceeded, hold any more commands)
0x03	**(Reserved for future use)**

DATA HEADERS

CODE	DESCRIPTION AND USE
0x04	**MONITOR DATA** This is returned by the X-10 network interface if the receiver is in MONITOR mode. Monitor mode is used by software to watch ALL bits that occur within the 1 millisecond period immediately following the zero crossing of the power line voltage. This is helpful when debugging an X-10 installation. A bit is captured every 1/120th of a second and stored. When 8 bits (8 cycles of data) have been recorded, the byte is transmitted to the host**. Note: X-10 reception is disabled when in this mode**. See code **0xF0 CONFIGURE X-10 OPTIONS** for more details on this mode. Payload within LynX-NET^{TM} Packet example: 0x10, 0x00, 0x52, 0x12, 0x04, 0xFF, 0x03, 7A
0x05	**ANALYZER DATA** This is returned by the X-10 network interface if the receiver is in ANALYZER mode. Analyzer mode is used by software to sample the power line every 130 microseconds synchronized with the zero crossing of the power line voltage. The data is arranged into bytes and transmitted 60 times a second to the host (for 60Hz systems). The data is synchronized with the zero crossing. *Note: the communications interface must be running faster than the data is being generated. This requires a minimum serial interface speed of 9600 bps for proper data transmission*. *A slower speed will result in corrupted data*. **Note: All X-10 function is disabled in this mode**. See code **0xF0 CONFIGURE X-10 OPTIONS** for more details on this mode. Payload within LynX-NET^{TM} Packet example: 0x10, 0x00, 0x52, 0x12, 0x05, 0xFF, (16 bytes of data), (1 byte of check sum)
0x06-0x07	**(Reserved for future use)**

STANDARD X-10 COMMANDS

CODE	DESCRIPTION AND USE
0x08	**X-10 UNIT ADDRESS** This is used to identify individual X-10 devices on the network without issuing a command. This will always be used when receiving commands from an X-10 network as devices are addressed. Commands without unit codes will normally follow. This command can handle multiple unit codes within the same house code. Payload example: 0x08, 0x01, 0x01, 0xFF (Address for unit B2 – 0xFF is EOL) Payload within LynX-NET^{TM} Packet example: 0x10, 0x00, 0x81, 0x04, 0x08, 0x01, 0x01, 0xFF, 0x9E (Packet from LynX-10 PLC)
0x09-0x0F	**(Reserved for future use)**

STANDARD X-10 COMMANDS (Continued)

CODE	DESCRIPTION AND USE
0x10	**ALL UNITS OFF** This command will instruct all units (lights and appliances) on the provided house code to turn off. This command does not use a unit code and the unit code will always be 0xFF (EOL). Payload example: 0x10, 0x04, 0xFF (All units off on house code E – note EOL in UC) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x05, 0x03, 0x10, 0x04, 0xFF, 0x2B (Packet being sent to LynX-10 PLC)
0x11	**ALL LIGHTS ON** This command will instruct all lights on the provided house code to turn on. This command does not use a unit code and the unit code will always be 0xFF (EOL). Payload example: 0x11, 0x04, 0xFF (All units off on house code E – note EOL in UC) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x01, 0x03, 0x11, 0x04, 0xFF, 0x28 (Packet being sent to LynX-10 PLC)
0x12	**ON** This command will instruct one or more X-10 units on a single house code to turn on. This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. Payload example: 0x12, 0x04, 0x00, 0x01, 0xFF (Units E1 and E2 ON) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x21, 0x05, 0x12, 0x04, 0x00, 0x01, 0xFF, 0x4C
0x13	**OFF** This command will instruct one or more X-10 units on a single house code to turn off. This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. Payload example: 0x13, 0x04, 0x00, 0x01, 0xFF (Units E1 and E2 OFF) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x22, 0x05, 0x13, 0x04, 0x00, 0x01, 0xFF, 0x4E
0x14	**DIM** This command will instruct one or more X-10 units on a single house code to dim n times where n is 1-255 (0x01-0xFF). This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. A dim count of zero (0) is automatically incremented to 1. Payload example: 0x14, 0x00, 0x00, 0x05, 0x06, 0xFF, 0x10 (Units A1, A6, and A7 DIM 16 times) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x25, 0x07, 0x14, 0x00, 0x00, 0x05, 0x06, 0xFF, 0x10, 0x6A
0x15	**BRIGHT** This command will instruct one or more X-10 units on a single house code to brighten n times where n is 1-255 (0x01-0xFF). This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. A bright count of zero (0) is automatically incremented to 1. Payload example: 0x15, 0x01, 0x00, 0x01, 0x02, 0xFF, 0x10 (Units B1, B2, and B3 BRIGHT 16 times) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x26, 0x07, 0x15, 0x01, 0x00, 0x01, 0x02, 0xFF, 0x10, 0x65

STANDARD X-10 COMMANDS (Continued)

CODE	DESCRIPTION AND USE
0x16	**ALL LIGHTS OFF** This command will instruct all lights (no appliance modules) on a single house code to turn off. Payload example: 0x16, 0x02, 0xFF (All lights off on house code C) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x1A, 0x03, 0x16, 0x02, 0xFF, 0x44
0x17	**EXTENDED CODE TRANSFER** This command will send an extended code frame onto the power line without data. This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. This is used primarily for legacy systems that had to work around the limitations of the TW523 such as LynX-10 systems. See Extended Code 0x31-0x34 for universal extended data transmission. Payload example: 0x17, 0x02, 0x00, 0x0F, 0xFF (Send extended code command to units C1 and C16) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x3E, 0x06, 0x17, 0x02, 0x00, 0x0F, 0xFF, 0x7B
0x18	**HAIL REQUEST** This command will send a hail request to a house code. If any 'entire house code' devices such as a LynX-PORT are using that house code, they should respond with a HAIL ACKNOWLEDGE. Payload example: 0x18, 0x0F, 0xFF (Send hail request to house code P) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x44, 0x03, 0x18, 0x0F, 0xFF, 0x7D
0x19	**HAIL ACKNOWLEDGE** This command will send a hail acknowledge to a house code. Devices should only use this command when responding to a HAIL REQUEST and wishing ownership of that entire house code. Payload example: 0x19, 0x0F, 0xFF (Send hail request to house code P) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x44, 0x03, 0x19, 0x0F, 0xFF, 0x7E
0x1A	**PRESET DIM 0** This command will send a preset dim command for levels between 0 and 15. The dim level (0x00-0x0F) is placed into house code. For a unit to respond to this command it must first be addressed (see code 0x08). Only one device can receive a preset dim command at a time. Payload example: 0x1A, 0x08, 0xFF (Send preset dim with level 0x08) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x23, 0x03, 0x1A, 0x08, 0xFF, 0x57

STANDARD X-10 COMMANDS (Continued)

CODE	DESCRIPTION AND USE
0x1B	**PRESET DIM 1** This command will send a preset dim command for levels between 16 and 31. The dim level (0x00-0x0F) is placed into house code. For a unit to respond to this command it must first be addressed (see code 0x08). Only one device can receive a preset dim command at a time. Payload example: 0x1B, 0x04, 0xFF (Send preset dim with level 0x04) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x23, 0x03, 0x1B, 0x04, 0xFF, 0x54
0x1C	**EXTENDED DATA TRANSFER** This command will send an extended data frame onto the power line without data. This command can use multiple unit codes within the same house code. If the unit code is missing, no unit address will be sent – only the house code. This is used primarily for legacy systems that had to work around the limitations of the TW523 such as LynX-10 systems. See Extended Code 0x31-0x34 for universal extended data transmission. Payload example: 0x1C, 0x02, 0x00, 0x0E, 0xFF (Send extended code 0x43 to units C1 and C15) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x3E, 0x06, 0x1C, 0x02, 0x00, 0x0E, 0xFF, 0x7F
0x1D	**STATUS ON** This command is used for replying to a STATUS REQUEST to indicate a unit is ON. Payload example: 0x1D, 0x00, 0xFF (Send status ON reply for house code A) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x9E, 0x03, 0x1D, 0x00, 0xFF, 0xCD
0x1E	**STATUS OFF** This command is used for replying to a STATUS REQUEST to indicate a unit is OFF. Payload example: 0x1E, 0x00, 0xFF (Send status OFF reply for house code A) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0xA4, 0x03, 0x1E, 0x00, 0xFF, 0xD4
0x1F	**STATUS REQUEST** This command is used to request the status of a given unit. If the unit is capable of two-way X-10 communication, it shall respond within 250 milliseconds with one of the two above status reporting commands. This command allows only one unit code at a time. If the unit code is missing, no unit address will be sent – only the house code. Payload example: 0x1F, 0x00, 0xFF (Request status for house code A – no unit code) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x3E, 0x03, 0x1F, 0x00, 0xFF, 0x6F

ENHANCED X-10 COMMANDS

CODE	DESCRIPTION AND USE
0x20	**ALL UNITS OFF ALL HOUSE CODES** This command will send an ALL UNITS OFF command to all house codes (A-P). This will effectively turn every unit off. Payload example: 0x20, 0xFF (Send preset dim with level 0x04) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x12, 0x02, 0x20, 0xFF, 0x43
0x21	**ALL LIGHTS OFF ALL HOUSE CODES** This command will send an ALL LIGHTS OFF command to all house codes (A-P). This will effectively turn every light off. Some wall switches and other devices do not respond to ALL LIGHTS OFF. They appear as appliance type devices and only respond to ALL UNITS OFF, so some lights may not go out when using this command. Payload example: 0x21, 0xFF (Send extended code 0x43 to units C1 and C15) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x3F, 0x02, 0x21, 0xFF, 0x71
0x22	**ALL LIGHTS ON ALL HOUSE CODES** This command will send an ALL LIGHTS ON command to all house codes (A-P). This will effectively turn every light on. Some wall switches and other devices do not respond to ALL LIGHTS ON. They appear as appliance type devices and only turn on when directly addressed and commanded on using the standard ON command. Payload example: 0x22, 0xFF (Send extended code 0x43 to units C1 and C15) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x41, 0x04, 0x22, 0xFF, 0x76
0x23-0x27	**(Reserved for future use)**
0x28	**DIM REL n** This command is reserved for future generation Marrick Limited X-10 interfaces
0x29	**DIM PRESET n** This command will send a preset dim command using the legacy PRESET DIM functions (see STANDARD X-10 COMMANDS for more information). This command can only handle a single unit and house code at a time. The n is any value from 1-31 (0x01-0x1F). A value of 0 will be automatically incremented to 1. Payload example: 0x29, 0x02, 0x01, 0xFF, 0x10 (Set unit C2 to level 16 using legacy preset dim) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x33, 0x05, 0x29, 0x02, 0x01, 0xFF, 0x10, 0x83
0x2A-0x2F	**(Reserved for future use)**

OTHER X-10 COMMANDS

0x30	**(Reserved for future use)**
0x31	**EXTENDED CODE 1** This is a general command to send any extended data and command onto the power line using Extended Code 1 for Data and Control. This command can use only one unit code at a time. Payload example: 0x31, 0x03, 0x05, 0xFF, 0xF2, 0x5A (Sends Extended Code 1 to House Code D, Unit 6 with data of 0xF2 and command of 0x5A) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2A, 0x06, 0x31, 0x03, 0x05, 0xFF, 0xF2, 0x5A, 0xC4
0x32	**EXTENDED CODE 2** This is a general command to send any extended data and command onto the power line using Extended Code 2 for Meter Reading and DSM. This command can use only one unit code at a time. Payload example: 0x32, 0x01, 0x0A, 0xFF, 0xA2, 0x22 (Sends Extended Code 2 to House Code B, Unit 11 with data of 0xA2 and command of 0x22) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2B, 0x06, 0x32, 0x01, 0x0A, 0xFF, 0xA2, 0x22, 0x41
0x33	**EXTENDED CODE 3** This is a general command to send any extended data and command onto the power line using Extended Code 3 for Security Messaging. This command can use only one unit code at a time. Payload example: 0x33, 0x02, 0x02, 0xFF, 0x11, 0x45 (Sends Extended Code 3 to House Code C, Unit 3 with data of 0x11 and command of 0x45) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2C, 0x06, 0x32, 0x01, 0x0A, 0xFF, 0xA2, 0x22, 0x42
0x34	**EXTENDED CODE 4** Same format as the above, but not defined at this time. This may be assigned in the future, or it can be used as a special user defined code.
0x35 – 0x7F	**(Reserved for future use)**

EXTENDED X-10 COMMANDS (Shutters and sunshades – Type 0)

0x80	**(Reserved for future use)**
0x81	**OPEN TO LEVEL n, ENABLE SUN PROTECTION** This command will open a shutter / sunshade to position n (normally 0=fully closed, 25=fully open) and enable sun protection as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes. Payload example: 0x81, 0x0F, 0x01, 0xFF, 0x19 (Fully open (0x19) sunshade unit P2 and enable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x15, 0x05, 0x81, 0x0F, 0x01, 0xFF, 0x19, 0xD3
0x82	**LIMIT OPENING TO LEVEL n, ENABLE SUN PROTECTION** This command will set the opening limit of a shutter / sunshade to a maximum position of n (normally 0=fully closed, 25=fully open) and enable sun protection as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes. Payload example: 0x82, 0x00, 0x04, 0xFF, 0x10 (Set opening limit to 16 on sunshade unit A5 and enable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x22, 0x05, 0x82, 0x00, 0x04, 0xFF, 0x10, 0xCC
0x83	**OPEN TO LEVEL n, DISABLE SUN PROTECTION** This command will open a shutter / sunshade to position n (normally 0=fully closed, 25=fully open) and disable sun protection as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes. Payload example: 0x83, 0x05, 0x0F, 0xFF, 0x10 (Open sunshade unit F16 to position16 and disable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x54, 0x05, 0x83, 0x05, 0x0F, 0xFF, 0x10, 0x0F
0x84	**OPEN ALL SHUTTERS / SUNSHADES FULL THIS HOUSE CODE, DISABLE SUN PROTECTION** This command will open all shutter / sunshade to full open position on house code n and disable sun protection as defined by X-10 USA. This command has no unit code or data. Payload example: 0x84, 0x02, 0xFF (Fully open all sunshades on house code C and disable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2C, 0x03, 0x84, 0x02, 0xFF, 0xC4
0x85	**OPEN ALL SHUTTERS / SUNSHADES FULL ALL HOUSE CODES, DISABLE SUN PROTECTION** This command will open all shutter / sunshade to full open position and disable sun protection as defined by X-10 USA. This command has no unit code or data. Payload example: 0x85, 0xFF (Fully open all sunshades on all house codes and disable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2C, 0x02, 0x85, 0xFF, 0xC2
0x86	**(Reserved for future use)**

EXTENDED X-10 CODE (Shutters and sunshades – Type 0 – continued)

0x87 **INCLUDE SHUTTER / SUNSHADE AT LEVEL n IN LIFE STYLE MODE m**

This command will include shutter / sunshade unit n in the life style mode m as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes. The bits for the data field are shown below:

D7	D6	D5	D4	D3	D2	D1	D0	
M4	M2	M1	L16	L8	L4	L2	L1	M=LIFE STYLE MODE (0-7) WITH L=OPENING LEVEL (0-31)

Payload example: 0x87, 0x01, 0x00, 0xFF, 0x31
(Include shutter / sunshade B1 at opening level 17 in life style mode 1)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x4A, 0x05, 0x87, 0x01, 0x00, 0xFF, 0x31, 0x17

0x88 **BEGIN LIFE STYLE MODE m**

This command will begin the life style mode m and all previously included shutter / sunshade units will move to their positions as defined by X-10 USA. This command has no unit or house code.

D7	D6	D5	D4	D3	D2	D1	D0	
M4	M2	M1	0	0	0	0	0	M=LIFE STYLE MODE (0-7)

Payload example: 0x88, 0xFF, 0x80
(Begin life style mode 4)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x55, 0x03, 0x88, 0xFF, 0x80, 0x6F

0x89 **EXCLUDE SHUTTER / SUNSHADE FROM LIFE STYLE MODE m**

This command will exclude a shutter / sunshade unit from life style mode m as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes.

D7	D6	D5	D4	D3	D2	D1	D0	
M4	M2	M1	0	0	0	0	0	M=LIFE STYLE MODE (0-7)

Payload example: 0x89, 0x03, 0x01, 0xFF, 0xE0
(Exclude shutter / sunshade D2 from life style mode 7 (0xE0))

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x1D, 0x05, 0x89, 0x03, 0x01, 0xFF, 0xE0, 0x9E

0x8A **EXCLUDE SHUTTER / SUNSHADE FROM ALL LIFE STYLES**

This command will exclude a shutter / sunshade unit from all life styles as defined by X-10 USA. This command can only handle one unit at a time and therefore does not allow multiple unit codes.

Payload example: 0x8A, 0x03, 0x02, 0xFF
(Exclude shutter / sunshade D3 from all life styles)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x1D, 0x04, 0x8A, 0x03, 0x02, 0xFF, 0xBF

EXTENDED X-10 CODES (Shutters and sunshades – Type 0 – continued)

0x8B	**CLOSE ALL SHUTTERS / SUNSHADES THIS HOUSE CODE, ENABLE SUN PROTECTION** This command will close all shutters / sunshades and enable sun protection as defined by X-10 USA. This command has no unit code or data. Payload example: 0x8B, 0x0E, 0xFF (Fully close all shutters / sunshades on house code O and enable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x4B, 0x03, 0x8B, 0x0E, 0xFF, 0xF6
0x8C	**CLOSE ALL SHUTTERS / SUNSHADES ALL HOUSE CODES, ENABLE SUN PROTECTION** This command will close all shutters / sunshades on all house codes and enable sun protection as defined by X-10 USA. This command has no house code, unit code or data. Payload example: 0x8C, 0xFF (Fully close all shutters / sunshades on all house codes and enable sun protection) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x6F, 0x02, 0x8C, 0xFF, 0x0C
0x8D	**(Reserved for future use)**
0x8E	**SELF TEST SHUTTER / SUNSHADE CONTROLLER - MECHANICAL** This command will cause the shutter / sunshade controller to enter mechanical test mode. In most cases this will cycle the shutter / sunshade from fully closed to fully open and back again returning to it's original position. This command has no data. Payload example: 0x8E, 0x01, 0x01, 0xFF (Test shutter / sunshade controller unit B2) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x11, 0x04, 0x8E, 0x01, 0x01, 0xFF, 0xB4
0x8F	**SELF TEST SHUTTER / SUNSHADE CONTROLLER – MECHANICAL & ELECTRICAL** This command will cause the shutter / sunshade controller to enter mechanical test mode. In most cases this will cycle the shutter / sunshade from fully closed to fully open and back again returning to it's original position. Also it will test its internal electronics returning the non-volatile storage to its factory default settings. This command has no data. Payload example: 0x8F, 0x01, 0x01, 0xFF (Test shutter / sunshade controller unit B2) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x11, 0x04, 0x8F, 0x01, 0x01, 0xFF, 0xB5

EXTENDED X-10 CODES (Sensors - Type 1)

0x90	**(Reserved for future use)**
0x91	**REQUEST AVERAGE LIGHT LEVEL FROM SENSOR** This command will request the average ambient light level data from a light sensor unit. If the unit is capable of responding, it will respond using code 0x9B (see below for more details). This command has no data. Payload example: 0x91, 0x04, 0x03, 0xFF (Request average ambient light level from sensor unit E4) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x1F, 0x04, 0x91, 0x04, 0x03, 0xFF, 0xCA
0x92	**REQUEST CURRENT TEMPERATURE FROM SENSOR** This command will request the current temperature level data from a temperature sensor unit. If the unit is capable of responding, it will respond using code 0x9C (see below for more details). This command has no data. Payload example: 0x92, 0x04, 0x00, 0xFF (Request current temperature from sensor unit E1) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x29, 0x04, 0x92, 0x04, 0x00, 0xFF, 0xD2
0x93	**REQUEST STATUS FROM SENSOR** This command will request the current status from a unit. If the unit is capable of responding, it will respond using code 0x9D (see below for more details). This command has no data. Payload example: 0x93, 0x04, 0x01, 0xFF (Request current temperature from sensor unit E2) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x33, 0x04, 0x93, 0x04, 0x01, 0xFF, 0xDE
0x94	**REQUEST CURRENT LIGHT LEVEL FROM SENSOR** This command will request the current ambient light level data from a light sensor unit. If the unit is capable of responding, it will respond using code 0x9B (see below for more details). This command has no data. Payload example: 0x94, 0x04, 0x03, 0xFF (Request current ambient light level from sensor unit E4) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x2C, 0x04, 0x94, 0x04, 0x03, 0xFF, 0xDA
0x95	**REQUEST AVERAGE TEMPERATURE FROM SENSOR (16 MINUTE AVERAGE)** This command will request the average temperature level data from a temperature sensor unit. If the unit is capable of responding, it will respond using code 0x9C (see below for more details). This command has no data. Payload example: 0x95, 0x04, 0x00, 0xFF (Request average temperature from sensor unit E1) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x42, 0x04, 0x95, 0x04, 0x00, 0xFF, 0xEE
0x96-0x9A	**(Reserved for future use)**

EXTENDED X-10 CODES (Sensors – Type 1 – continued)

0x9B	**AMBIENT LIGHT DATA FROM SENSOR (RESPONSE TO REQUEST ABOVE)** This command is in response to a sensor request and usually will not be issued directly. The response is shown below. If issued (as from a computer sensor), only one house code unit code is allowed at a time. Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x82, 0x05, 0x9B, 0x04, 0x03, 0xFF, 0x77, 0xAF (Sensor unit E4 responded with ambient light data of 0x77)
0x9C	**TEMPERATURE DATA FROM SENSOR (RESPONSE TO REQUEST ABOVE)** This command is in response to a sensor request and usually will not be issued directly. The response is shown below. If issued (as from a computer sensor), only one house code unit code is allowed at a time. Payload within LynX-NET™ Packet example: 0x10, 0x00, 0xA4, 0x05, 0x9C, 0x04, 0x00, 0xFF, 0xBC, 0x14 (Sensor unit E1 responded with temperature data of 0xBC)
0x9D	**STATUS DATA FROM UNIT (RESPONSE TO REQUEST ABOVE)** This command is in response to a sensor request and usually will not be issued directly. The response is shown below. If issued (as from a computer sensor unit), only one house code unit code is allowed at a time. Payload within LynX-NET™ Packet example: 0x10, 0x00, 0xB4, 0x05, 0x9D, 0x04, 0x01, 0xFF, 0x05, 0x6F (Sensor unit E2 responded with status data of 0x05)
0x9E-0x9F	**(Reserved for future use)**

EXTENDED X-10 CODES (Security – Type 2)

0xA0-0xAF	**(Reserved for future use)**

EXTENDED X-10 CODES (Dimmers and appliances – Type 3)

0xB0 — **INCLUDE UNIT IN GROUP m AT CURRENT LEVEL / SETTING**

This command will allow a unit to participate in a group. Groups can be either relative or absolute – that is, a group number can be 0-3, or 0-3 relative to a setting number (0-15). Absolute members will ignore any setting number associated with group execution commands where relative groups will need an exact settings match to execute. This command can only handle one unit code at a time. The bit assignments for the data byte are shown below:

D7	D6	D5	D4	D3	D2	D1	D0	
G2	G1	0	0	0	0	0	0	GROUP (0-3) FOR ALL SETTINGS
G2	G1	1	1	S8	S4	S2	S1	GROUP (0-3) FOR SETTING (0-15)

Payload example: 0xB0, 0x03, 0x01, 0xFF, 0x32
(Include unit D2 in Group 0 for settings 2)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x22, 0x05, 0xB0, 0x03, 0x01, 0xFF, 0x32, 0x1C

0xB1 — **PRESET UNIT TO LEVEL n**

This command will set a unit to level n (normally 0-63). A unit that has no DIM capability will be 'ON' for any non-zero value and 'OFF' for level zero. Units that have DIM capability will set their level to the level requested by n. This command can only handle one unit code at a time.

Payload example: 0xB1, 0x00, 0x00, 0xFF, 0x20
(Set unit A1 to level 32 (0x20))

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x31, 0x05, 0xB1, 0x00, 0x00, 0xFF, 0x20, 0x16

0xB2 — **INCLUDE UNIT IN GROUP m AT LEVEL n**

This command will allow a unit to participate in a group with a predefined level setting. The group number (0-3) for this command is absolute and will respond to any setting for that group. This command can only handle one unit code at a time. The bit assignments for the data byte are shown below:

D7	D6	D5	D4	D3	D2	D1	D0	
G2	G1	L32	L16	L8	L4	L2	L1	GROUP (0-3) TO LEVEL (0-63)

Payload example: 0xB2, 0x01, 0x01, 0xFF, 0x5F
(Include unit B2 in Group 1 (0x01) with level 31 (0x1F))

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x45, 0x05, 0xB2, 0x01, 0x01, 0xFF, 0x5F, 0x6C

0xB3 — **ALL UNITS ON – THIS HOUSE CODE**

This command will turn on all units on the specified house code. This command has no unit code or data.

Payload example: 0xB3, 0x01, 0xFF
(Turn on all units on house code B)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x2F, 0x03, 0xB3, 0x01, 0xFF, 0xF5

0xB4 — **ALL UNITS OFF – THIS HOUSE CODE**

This command will turn off all units on the specified house code. This command has no unit code or data.

Payload example: 0xB4, 0x00, 0xFF
(Turn off all units on house code A)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x2F, 0x03, 0xB4, 0x00, 0xFF, 0xF5

EXTENDED X-10 CODES (Dimmers and appliances – Type 3 – continued)

0xB5 **REMOVE UNIT FROM GROUP m**

This command will remove a unit from a group. The data byte contains bits that correlate to each group so multiple assignments can be removed with one command. The mapping of these bits is shown below. This command can handle only one unit code at a time.

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	G3	G2	G1	G0	REMOVE THIS UNIT FROM GROUP G0-G3.
1	1	1	1	G3	G2	G1	G0	REMOVE ALL UNITS THIS HOUSE CODE FROM GROUP G0-G3.

Payload example: 0xB5, 0x03, 0x01, 0xFF, 0x02
(Remove unit D2 from group 1)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x71, 0x05, 0xB5, 0x03, 0x01, 0xFF, 0x02, 0x40

0xB6 **EXECUTE GROUP m FUNCTION**

This command will begin execution of a group. All members of that group (absolute and relative to a setting) will change their outputs to match their individual settings assigned by the 'INCLUDE IN GROUP' functions. This command's data byte is bit mapped with specific settings as shown below:

D7	D6	D5	D4	D3	D2	D1	D0	
G2	G1	0	0	0	0	0	0	G=GROUP (0-3)
G2	G1	1	1	S8	S4	S2	S1	G=GROUP (0-3) S=SETTING (0-15)

Payload example: 0xB6, 0x00, 0xFF, 0xF1
(Execute group 3 on House Code A with setting 1)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x10, 0x04, 0xB6, 0x00, 0xFF, 0xF1, 0xCA

0xB7 **REQUEST OUTPUT STATUS (UNIT OR GROUP)**

This command will request the output status of a unit or a unit group. The data byte defines the type of request. The data byte is bit mapped as shown below:

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	0	0	0	0	STATUS OF SINGLE UNIT
0	0	0	1	0	0	0	0	STATUS AFTER POWER UP
G2	G1	1	0	0	0	0	0	GROUP (0-3) STATUS
G2	G1	1	1	S8	S4	S2	S1	GROUP (0-3) SETTING (0-15) STATUS

Payload example: 0xB7, 0x03, 0x01, 0xFF, 0x00
(Request output status of unit D2)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x4F, 0x05, 0xB7, 0x03, 0x01, 0xFF, 0x00, 0x1E

EXTENDED X-10 CODES (Dimmers and appliances – Type 3 – continued)

0xB8 **OUTPUT STATUS ACK (REPLY TO UNIT REQUEST)**

This command is in response to a single-unit output status request (code 0xB7 with data either 0x00 or 0x10). The response is normally returned from a remote unit to the host (as shown below). If issued (as from a software unit), only one unit code is allowed at a time. The data returned is bit mapped as shown below:

D7	D6	D5	D4	D3	D2	D1	D0
A2	A1	B32	B16	B8	B4	B2	B1

```
LEVEL (0-63) IN BITS D0-D5
BIT A1: 0=Dimmer, 1=Appliance
BIT A2: 1=Lamp connected
```

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0xA7, 0x05, 0xB8, 0x07, 0x05, 0xFF, 0x82, 0x01
(Unit H6 reports: Lamp connected, Dimmer Module, brightness level 2)

0xB9 **OUTPUT STATUS ACK (REPLY TO GROUP REQUEST)**

This command is in response to a group output status request (code 0xB7 with data above or equal to 0x20). The response is normally returned from a remote unit to the host (as shown below). If issued (as from a software unit), only one unit code is allowed at a time. The data returned is bit mapped as shown below:

D7	D6	D5	D4	D3	D2	D1	D0
G2	G1	B32	B16	B8	B4	B2	B1

```
LEVEL (0-63) IN BITS D0-D5
GROUP (0-4) IN BITS D6-D7
```

Payload within LynX-NET™ Packet example:
0x10, 0x00,0xA8, 0x05, 0xB9, 0x07, 0x06, 0xFF, 0xC5, 0x47
(Unit H7 reports member of group 3 – output currently set to level 5)

0xBA **OUTPUT STATUS NAK (NOT IN GROUP)**

This command is in response to a group output status request (code 0xB7 with data above or equal to 0x20). The response is issued if the unit addressed is not in the group requested. The response is normally returned from a remote unit to the host (as shown below). If issued (as from a software unit), only one unit code is allowed at a time.

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0xAC, 0x04, 0xBA, 0x07, 0x07, 0xFF, 0x20, 0xA7
(Unit H8 reports that it is not a member of group 0)

0xBB **CONFIGURE MODULES THIS HOUSE CODE**

This command sets various attributes of modules on the house code specified. This command addresses all units, therefore it does not require a unit code. The functions are bit mapped into the data byte as shown below:

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	C2	C1

```
BIT C1: 1=AUTOACK 'EXTENDED' ON
BIT C2: 1=AUTOACK 'STANDARD' ON
```

Payload example: 0xBB, 0x03, 0xFF, 0x03
(Enable both auto acknowledge methods for all units on house code D)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x55, 0x04, 0xBB, 0x03, 0xFF, 0x03, 0x29

0xBC-0xBF **(Reserved for future use)**

EXTENDED X-10 CODES (Marrick extensions – Type 7)

0xF0 **READ / WRITE X-10 OPTIONS**

This command is intercepted by the internal X-10 sub-system to set or get X-10 option settings. Options are bit mapped and are shown below. By leaving the data byte off the payload, the options can be read. The return packet has a new sequence number and will have the command (0xF0), delimiter token (0xFF), and data byte in the payload.

D7	D6	D5	D4	D3	D2	D1	D0_
R	R	LB	PD	TP	PM2	PM1	PM0

```
PMx: POWER LINE RECEIVE MODE
     000 = X-10 DECODE (DEFAULT)
     001 = RAW DATA
     010 = MONITOR
     011 = ANALYZER
     1xx = (RESERVED)
TP : TX PHASES     0=3 PHASE
                   1=1 PHASE
PD : PRESET DIM    0=NORMAL
                   1=TRANSLATE HC
LB : LOOP BACK     0=OFF (DEFAULT)
                   1=ON
R  : (Reserved bits)
```

POWER LINE RECEIVE MODE: Determines how the power line receiver is used. When in X-10 DECODE mode (PM0=0, PM1=0, PM2=0), normal X-10 protocol is observed and decoded. This requires the detection of a proper preamble and correct number of bits as well as proper Manchester encoding. When in RAW DATA mode (PM0=1, PM1=0, PM2=0), the bits are not decoded, but passed along to the host as raw data including the preamble. Data is only sent to the host following a proper preamble. When in MONITOR mode (PM0=0, PM1=1, PM2=0), all bits are sent continuously to the host in packets of 16 bytes each (See RAW MODE in this section for more details). When in ANALYZER mode (PM0=1, PM1=1, PM2=0), the power line is sampled every 130 uS (for 60 Hz), which will return 128 samples for every cycle. This allows software to analyze noise that can be seen by the receiver circuitry to trouble shoot X-10 installations. (Note: In ANALYZER mode the UART must be set for at least 9600 bps to gather all data. Otherwise, cycles will be skipped in order to maintain a steady flow of data to the host)

TX PHASES: 3 phase is used when bridging a 3-phase power system. 1 phase reduces transmitted noise on the power line and is used on single or dual phase systems.

PRESET DIM TRANSLATE: The house code of the legacy PRESET DIM command is the actual level. However, the level does not normally directly correlate to the house code. That is, A does not equal level 0 and P does not equal level 15. Older software used a translation table to convert house code to level data. If this bit is set, the translation is done in firmware so A=0, B=1, C=2, etc…

LOOP BACK: This bit enables the X-10 receiver state machine during transmission, which allows the transmission data to be received as it would by another device. This can be used for self-test or to verify the transmission was correctly sent. If this bit is set, data will be received and decoded during transmissions and sent to the host as they complete.

Payload example: 0xF0, 0xFF, 0x08
(X-10 Options set: No loop back, no house code translation on Preset Dim, 1 phase per cycle transmission, and use normal X-10 decoders)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x22, 0x02, 0xF0, 0xFF, 0x08, 0x2B

To read current settings, simply issue the command without data following the 0xFF.

EXTENDED X-10 CODES (Marrick extensions – Type 7 - continued)

0xF1	**READ TIME DURATION SINCE CARRIER (i.e. 60Hz) PRESENT** This command is intercepted by the internal X-10 sub-system. The command then returns the duration since the power has been present on the power line used for X-10 communications in HEX. The returned packet uses the same sequence number as the command. The packet includes the original command plus 5 bytes holding the HEX values of the duration. The first 2 bytes are days, the next byte is hours, the next minutes, and the last byte seconds. Payload example: 0xF1, 0xFF (Host requests duration of time carrier present) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x25, 0x02, 0xF1, 0xFF, 0x27 Payload within LynX-NET™ Packet example (return data from interface): 0x10, 0x00, 0xBF, 0x08, 0xF1, 0xFF, 0x01, 0xA1, 0x14, 0x25, 0x04, 0xA6 (Interface (i.e. LynX-10 PLC) reports the carrier has been present for 417 (0x01A1) days, 20 (0x14) hours, 37 (0x25) minutes, and 4 (0x04) seconds – great power company!)

EXTENDED X-10 CODES (Marrick extensions – Type 7 - continued)

0xF2 **READ STATISTICS COUNTER n**

This command is intercepted by the internal X-10 sub-system. The command then returns the value of the statistics counter requested by n. Each counter is 2 bytes long and is returned in hex. The returned packet uses the same sequence number as the request command. The packet will contain the original command (3 bytes) plus the 2 new counter bytes. The table below shows the various counters that can be retrieved.

TABLE x – STATISTICS COUNTERS

n	Description
0x00	X-10 valid transmitted packets
0x01	X-10 valid received packets
0x02	X-10 reception errors – coding faults
0x03	X-10 transmission failures
0x04	X-10 collisions
0x05	Power Failures
0x06-0xFF	(Reserved)

Payload example: 0xF2, 0xFF, 0x04
(Host requests statistics counter 4 – X-10 collisions)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x2A, 0x03, 0xF2, 0xFF, 0x04, 0x32

Payload within LynX-NET™ Packet example (return data from interface):
0x10, 0x00, 0x2A, 0x05, 0xF2, 0xFF, 0x04, 0x22, 0x07, 0x5D
(The interface reports there have been 8711 (0x2207) collisions during X-10 transmissions since this statistic counter has been cleared)

0xF3 **CLEAR STATISTICS COUNTER n**

This command is intercepted by the internal X-10 sub-system. The X-10 module will then set statistics counter n (as shown above) to zero. If the counter number is missing (i.e. 0xF3 0xFF), all counters will be cleared.

Payload example: 0xF3, 0xFF, 0x04
(Host clears statistics counter 4)

Payload within LynX-NET™ Packet example:
0x10, 0x00, 0x2B, 0x03, 0xF2, 0xFF, 0x04, 0x33

0xF4-0xFB **(Reserved for future use)**

EXTENDED X-10 CODES (Marrick extensions – Type 7 - continued)

0xFC	**READ / WRITE X-10 RECEIVER SENSITIVITY** This command is intercepted by the internal X-10 sub-system. The command will then read or write the receiver's sensitivity register. This value should be between 0x00 and 0xFF. 0xFF is the highest sensitivity, and is the factory default. To get the current sensitivity value, leave off the data byte from the payload and a new packet will be sent to the host containing the current sensitivity data. Payload example: 0xFC, 0xFF, 0xFF (Sets receiver sensitivity to maximum) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x11, 0x03, 0xFC, 0xFF, 0xFF, 0x1E Example returned LynX-NET™ Packet after request (no data byte in payload): 0x10, 0x00, 0xB7, 0x03, 0xFC, 0xFF, 0x7F, 0x44 (X-10 sub-system reports receiver sensitivity set to 0x7F or 50%).
0xFD	**READ / WRITE X-10 TRANSMITTER POWER** This command is intercepted by the X-10 sub-system. The command will then read or write a value to the X-10 transmitter output power level register. This value should between 0x00 and 0xFF. 0xFF is the highest output power and is the factory default. To read the current output power setting, leave the data byte out of the payload (see command 0xFC above for more details) Payload example: 0xFD, 0xFF, 0x7F (Set transmitter power level to 50% (half 0xFF)) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x12, 0x03, 0xFD, 0xFF, 0x7F, 0xA0
0xFE	**SELECT X-10 CHANNEL** This command is intercepted by the X-10 sub-system. The command will read or write the X-10 channel number register. The value should be between 0x00 and 0x07. Channel 0x00 is the factory default and standard X-10 transmitter position. To read the current channel, leave off the channel number following the 0xFF delimiter (see command 0xFC above for more detail). *Channels 0x01 through 0x07 are reserved for future use*. Payload example: 0xFE, 0xFF, 0x00 (Selects channel 0, the default X-10 channel, for operation) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x13, 0x03, 0xFE, 0xFF, 0x00, 0x23
0xFF	**RAW DATA POWER LINE ACCESS** This command is intercepted by the X-10 sub-system. This command allows direct access to the power line using the Custom Preamble and Post values. Using this command allows any data format of bits to be transmitted using the 120KHz X-10 transmitter. If the RAW decoder is enabled (see X-10 Options command 0xF0), the raw data will be received without X-10 decoding. The payload can be up to 16 bytes (128 bits) long. After the second 0xFF (Delimiter token) the data bytes can be any 8 bit value. *No encoding is done to the data, but it is advised to use Manchester encoding to simplify decoding and provide reliable transport*. This enables any of the follow scenarios: **1. Data only transmission between devices** **2. Custom device implementation** **3. Adaptability to any new X-10 standard** **4. Custom formatting of commands within the host** Payload example: 0xFF, 0xFF, 0x55, 0xAA, 0x69, 0x56,…,0x99 (16th byte) (Transmits a preamble, 16 bytes of data, and a postamble) Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x14, 0x12, 0xFF, 0xFF, 0x55, 0xAA, 0x69, 0x56,…,0x99, 0x(TBD)

LYNX-NET™ PROTOCOL – LynX-NODE PAYLOAD DESCRIPTION

The following figures show the communications structure of the LynX-NODE advanced automation protocol. It can be carried within a TCP or UDP packet as well as within a LynX-NET packet.

FIGURE x – LynX-NODE COMMANDS

COMMAND	SOURCE ADDRESS	DEST ADDRESS	DATA
0x00 CMD FAILURE 0x01 CMD SUCCESS 0x02 STATUS 0x03- (Reserved) 0x07 **STUFF** 0x08- (Reserved) 0xFF	0x0000- Normal 0xFFFE 0xFFFF (Reserved) This word is the source address of this command. If it originates from the master controller, it is always 0x0000. The address 0xFFFF is reserved.	0x0000- Normal 0xFFFE 0xFFFF Broadcast This word is the destination address of the command. Addresses 0x0000 through 0xFFFE are valid. Address 0xFFFF is used as a broadcast address to all units listening on the network	The Data field contains any associated information required by the current command.

LynX-NODE COMMAND PROTOCOL OVERVIEW

GENERAL PROTOCOL

CODE	DESCRIPTION AND USE
0x00	**COMMAND FAIL** This is returned by the LynX-NODE network interface if the requested command times out or for any reason cannot be completed. The Source Address and Destination Address are reversed from the originating command. The data field carries a code that indicates the failure. The failure codes are shown below. Payload Example: 0x00 0x01 0x0E 0x00 0x00 0x01 Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x52, 0x06, 0x00 0x01 0x0E 0x00 0x00 0x01, 0x?? **FAILURE CODE** 0x01 Unsupported LynX-NODE command 0x02 Insufficient data for command 0x03 Bad command format (order of data, too much data) 0x04 (Reserved) 0x05 Data value out of range 0x06 Transmission failure (unable to deliver to network) 0x07-0xFE (Reserved) 0xFF Unknown internal failure (Should issue reset command when this occurs)
0x01	**COMMAND SUCCESS** This is returned after successful completion of the command. This means that the interface has successfully transmitted the LynX-NODE command to the target device without detectable errors. Payload example: 0x01 0x02 0x1A 0x00 0x00 Payload within LynX-NET™ Packet example: 0x10, 0x00, 0x52, 0x05, 0x01 0x02 0x1A 0x00 0x00, 0x??

LYNX-NET™ PROTOCOL – LynX-NET™ COMMAND PAYLOAD

The following figures show the commands for a LynX-NET™ network. These commands are issued with a Network ID of 0xE0 that tells the various interface devices that the message is for them. The payload contains a command along with associated optional data. The figure below outlines the various commands.

FIGURE X – LynX-NET™ COMMANDS

COMMAND		DATA (OPTIONAL)
0x00	COMMAND FAILURE	FAILURE CODE FOLLOWS RESPONSE WITH 1 BYTE
0x01	COMMAND SUCCESS	NO DATA
0x02-0x05	RESERVED	
0x06	ENUMERATE DEVICES	
0x07	ENUMERATE INTERFACES	
0x08	ENUMERATE PROTOCOLS	
0x09	REQUEST FIRMWARE VERSION	THE VERSION IS RETURNED IN THE DATA FIELD
0x0A	REQUEST SERIAL NUMBER	
0x0B	REQUEST FIRWARE REVISION	
0x0C-0x0F	RESERVED	
0x10	READ DEVICE REGISTER	DATA HOLDS REGISTER TO READ, RETURNED DATA INCLUDES REGISTER AND DATA BYTE
0x11	WRITE DEVICE REGISTER	DATA HOLDS REGISTER TO WRITE AND DATA BYTE
0x12-0xEF	RESERVED	
0xF0	RESET FACTORY DEFAULTS	

LynX-NET™ COMMAND PROTOCOL OVERVIEW

0x00 — **COMMAND FAILURE**

This response is sent to the host when a requested LynX-NET™ network command has failed or is unsupported. A byte follows the data that indicates the error code.

Payload within LynX-NET™ Packet example:
0xE0, 0x00, 0x33, 0x02, 0x00, 0x00, 0x15
(Command with sequence number 0x33 failed – unsupported command)

Error Codes:

0x00	UNSUPPORTED COMMAND
0x01	UNSUPPORTED REGISTER
0x02	INVALID LENGTH (Too many / too few bytes in payload)
0x03-0xFE	RESERVED
0xFF	UNKNOWN ERROR (Should issue RESET command if this occurs)

0x01 — **COMMAND SUCCESS**

This response is sent to the host when a requested interface device or network command has been successfully completed or is about to successfully complete. Note: Either a COMMAND FAILURE or COMMAND SUCCESS byte will be returned for all commands. If a command has data to return, that will be transported in a new packet with a new down stream (0x80-0xFF) sequence number. The returned data and command will be identified in the payload. The interface device will be identified by the Node ID.

Payload within LynX-NET™ Packet example:
0xE0, 0x00, 0x33, 0x01, 0x01, 0x15
(Command with sequence number 0x33 succeeded)

0x02-0x05 — **(Reserved for future use)**

LynX-NET™ COMMAND PROTOCOL OVERVIEW - CONTINUED

0x06 **ENUMERATE DEVICES**

This command will cause all addressed Node IDs to respond with their installed devices. The command from the host must have 0xFF as the Node ID if all devices are to respond. If a specific Node ID is used, only devices installed on that Node will be reported. If the Node ID is valid, the Node will respond with all installed devices by adding the device ID after the 0x06-command byte. Its Node ID will also appear in the Node ID field. Note: Node 0 is always local. Below are several examples:

Payload within LynX-NET™ Packet example 1:
0xE0, 0x00, 0x22, 0x01, 0x06, 0x??
(Tests to see if there are devices installed on the main board)

Reply may appear like this:
0xE0, 0x00, 0xAA, 0x02, 0x06, 0x??
(The main board indicates it has no devices installed – LynX-10 PLC example)

Payload within LynX-NET™ Packet example 2:
0xE0, 0xFF, 0x22, 0x01, 0x06, 0x0A
(Enumerate all devices)

Reply may appear like this:

0xE0, 0x00, 0xAB, 0x03, 0x06, 0x??, 0x??, 0x??	Node 0 has
0xE0, 0x01, 0xAC, 0x02, 0x06, 0x??, 0x??	Node 1 has
0xE0, 0x02, 0xAD, 0x02, 0x06, 0x??, 0x??	Node 2 has
0xE0, 0x03, 0xAE, 0x01, 0x06, 0x??	Node 3 has no installed devices

(Responses from various Nodes on a LynX-NET™ Network. This may be a LynX-PORT II, a LynX-NODE Gateway with add-in cards, USB devices on a hub, or several daisy-chained devices)

0x07 **ENUMERATE INTERFACES**

This command will cause all addressed Node IDs to respond with their available interface types. The command from the host must have 0xFF as the Node ID if all Nodes are to respond. If a specific Node ID is used, only the interface type installed in that Node will be reported. If the Node ID is valid, the Node will respond Interface Type ID after the 0x07-command byte. Its Node ID will also appear in the Node ID field. Note: Node 0 is always local and the default. If no other Nodes are available, it can have an Interface Type ID. Other devices, like the LynX-PORT II, may have several Interface Type IDs, however, Node 0 (the local Node) may not have one. Below are several examples:

Payload within LynX-NET™ Packet example 1:
0xE0, 0x01, 0x22, 0x01, 0x07, 0x0B
(Requests the interface type ID of Node 1)

Reply may appear like this:
0xE0, 0x01, 0xAA, 0x02, 0x07, 0x02, 0x96
(Node 1 indicates it is a RS-485 type interface)

Payload within LynX-NET™ Packet example 2:
0xE0, 0xFF, 0x22, 0x01, 0x07, 0x0A
(Enumerate all interfaces)

Reply may appear like this:

0xE0, 0x00, 0xAB, 0x03, 0x07, 0x95	Node 0 has no interface installed
0xE0, 0x01, 0xAC, 0x02, 0x07, 0x11, 0xA7	Node 1 is a PSC05 interface (X-10)
0xE0, 0x02, 0xAD, 0x02, 0x07, 0x02, 0x9A	Node 2 is an RS-485 interface

(Responses from various Nodes on a LynX-NET™ Network. This may be a LynX-PORT II, a LynX-NODE Gateway with add-in cards, USB Devices, several daisy-chained devices, etc)

TYPE ID
0x00 – Unknown
0x01 – RS-232
0x02 – RS-485

0x0F – InterLynX Bus

0x10 – Marrick PLC (X-10 Power line)
0x11 – PSC05 / TW523 (X-10 Power line)

0x20 – IR, Consumer (TV, VCR, Etc.)
0x21 – IR, Data

0x30 – RF, General

0x40 – Ethernet, Wired (1,10,100,1000 Mb/s)
0x41 – Ethernet, RF

0x50 – Universal Serial Bus

LynX-NET™ COMMAND PROTOCOL OVERVIEW - CONTINUED

0x08 — **ENUMERATE PROTOCOLS**

This command will cause all addressed interfaces to respond with their installed protocols. The command from the host must have 0xFF as the Node ID if all protocol stacks are to respond. If a specific Node ID is used, only protocols installed on that interface will be reported. If the Node ID is valid, the interface will respond with all installed protocols by adding the valid Network IDs it can support after the 0x08-command byte. Its Node ID will also appear in the Node ID field.

Below are several examples

Payload within LynX-NET™ Packet example 1:
0xE0, 0x01, 0x22, 0x01, 0x08, 0x0C
(Tests to see if there is installed protocols on interface node 1)

Reply may appear like this:
0xE0, 0x01, 0xAA, 0x02, 0x08, 0x11, 0xA6
(Node 1 indicates it can support only LynX-NODE protocol)

Payload within LynX-NET™ Packet example 2:
0xE0, 0xFF, 0x22, 0x01, 0x08, 0x0A
(Enumerate all protocols)

Reply may appear like this:

0xE0, 0x00, 0xAB, 0x03, 0x08, 0x10, 0x13, 0xB9	Node 0 supports X-10 & CEBus
0xE0, 0x01, 0xAC, 0x02, 0x08, 0x13, 0xAA	Node 1 supports only CEBus
0xE0, 0x02, 0xAD, 0x02, 0x08, 0x11, 0xAA	Node 2 supports only LynX-NODE
0xE0, 0x03, 0xAE, 0x01, 0x08, 0x9A	Node 3 has no installed protocols

(Responses from various interfaces on a shared physical LynX-NET™ Network. This may be a LynX-PORT II, a LynX-NODE Gateway with add-in cards, or several daisy-chained devices)

0x09 — **REQUEST MANUFACTURER AND MODEL NUMBER**

This command will cause the device indicated by the Node ID to respond with its manufacturer and model number. The returned data will follow the original command as shown below. The manufacturer and model number data are coded in BCD and will contain 4 additional bytes.

Payload within LynX-NET™ Packet example:
0xE0, 0x00, 0x25, 0x01, 0x09, 0x0F
(Requests manufacture and model of interface device in Node 0)

Reply may appear like this:
0xE0, 0x00, 0x95, 0x05, 0x09, 0x00, 0x00, 0x01, 0x05, 0x89
(Device at Node 0 reports it's a Marrick limited (0000) LynX-10 PLC (0105))

0x0A — **REQUEST SERIAL NUMBER**

This command will cause the device indicated by the Node ID to respond with its manufacturer's serial number. The returned data will follow the original command as shown below. The serial number is coded in BCD and will contain 8 additional bytes.

Payload within LynX-NET™ Packet example:
0xE0, 0x00, 0x30, 0x01, 0x0A, 0x1B
(Requests serial number of interface device in Node 0)

Reply may appear like this:
0xE0, 0x00, 0xA3, 0x09 , 0x0A, 0x00, 0x00, 0x00, 0x00, 0x00, 0x53, 0x13, 0x44, 0x40
(Interface at Node 0 reports it's serial number as 0000000000531344)

LynX-NET™ COMMAND PROTOCOL OVERVIEW – CONTINUED

0x0B	**REQUEST FIRMWARE VERSION** This command will cause the device indicated by the Node ID to respond with its current firmware version. The returned data will follow the original command as shown below. The version data is coded in BCD and will contain 2 additional bytes. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x25, 0x01, 0x0B, 0x11 (Requests firmware revision of interface device in Node 0) Reply may appear like this: 0xE0, 0x00, 0x95, 0x03, 0x0B, 0x01, 0x15, 0x99 (Device at Node 0 reports its firmware version as 1.15)
0x0C-0x0F	**(Reserved for future use)**
0x10	**READ DEVICE NON-VOLITILE MEMORY REGISTER** This command will return the contents of a non-volatile register within an interface device. All registers are 8 bits long (1 byte). The command requires 1 byte for the register address. Valid register addresses can be from 0x00 to 0xFF. Registers are device specific, and therefore all registers may not be supported in an interface device. See individual interface device specifications for location and function of each register. The response will contain the command, the address, and the byte of data from that register if valid. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x2A, 0x02, 0x10, 0x0A, 0x26 (Requests the contents of register 0x0A of interface in Node 0) Reply may appear like this: 0xE0, 0x00, 0x9F, 0x03, 0x10, 0x0A, 0xF3, 0x8F (Register 0x0A of device on Node 0 has value of 0xF3)
0x11	**WRITE DEVICE NON-VOLITILE MEMORY REGISTER** This command will alter the contents of a non-volatile register within an interface device. All registers are 8 bits long (1 byte). The command requires 1 byte for the register address plus 1 byte of data. Valid register addresses can be from 0x00 to 0xFF. Registers are device specific, and therefore all registers may not be supported in an interface device. See individual interface device specifications for location and function of each register. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x1C, 0x03, 0x11, 0x0C, 0x4A, 0x66 (Request to write register 0x0C of device at Node 0 with byte 0x4A)
0x12-0xEF	**(Reserved for future use)**

LynX-NET™ COMMAND PROTOCOL OVERVIEW – CONTINUED

0xF0	**RESET FACTORY DEFAULTS** This command will return a device in a given node to the factory default settings. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x52, 0x01, 0xF0, 0x66 (Device at Node 0 will reset to its factory settings)
0xF1-0xF5	**(Reserved for future use)**
0xF6	**FACTORY TEST ENABLE** This command will begin the built in self-test and diagnostics. The tests will continue until the beginning of a new packet is detected. **Tests will vary from device to device**. Please refer to the devices' documentation for specific behavior of this command. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x33, 0x01, 0xF6, 0x00, 0x0A (Device at Node 0 will enter test mode 0x00)
0xF7-0xFD	**(Reserved for future use)**
0xFE	**RESET PROTOCOL STACK** This command will reset a protocol stack and clear all related buffers. It will then initialize the protocol stack's state engine. A network ID of 0x00 will reset all the stacks at that node. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x33, 0x01, 0xFE, 0x10, 0x22 (Device at Node 0 will reset it's X-10 protocol stack)
0xFF	**RESET DEVICE** This command will force a device to reset. Settings are reloaded from non-volatile storage. Payload within LynX-NET™ Packet example: 0xE0, 0x00, 0x52, 0x01, 0xFF, 0x66 (Device at Node 0 will reset after shut-down of internal firmware)