LynX-PORT™

Remote or Local Universal Input / Output Controller

Model: 201

Marrick Limited LynX-PORT™ I/O Controller
Manual revision 1.20
Marrick Limited, Incorporated
P.O. Box 950940
Lake Mary, FL 32795
(407) 323-4467 Voice
(407) 322-1429 BBS
(407) 324-1291 FAX
EMAIL: 70571.2154@compuserve.com

Table of Contents

1.0 OVERVIEW OF X-10 TECHNOLOGY

X-10 control is one of the most prolific and low cost means to automate a home or business, especially if running new wiring is prohibitive. The only major drawback to the system is the computer interface. There have been many attempts to provide a reliable two-way connection from the computer or computer like equipment to the X-10 world. The first effort was by the Power House (X-10) people themselves. This was the TW523 module which provided the optically isolated connections to the power line. It did not provide the timing required to generate the correct control commands. Next came several simple interfaces that were merely level translators to the TW523 to hook into the computer's serial or parallel ports. These interfaces required the computer to perform all the monitoring and timing requirements greatly loading the computer system. Today with the advent of multitasking systems such as Microsoft Windows™ and UNIX™, the computer cannot service a single task for 1 or 2 seconds. This would cause disastrous results in the software. Next came the chips! This was the first attempt to free the computer from the burden of the X-10 timing. These chips have parallel interfaces on one side and the TW523 interface on the other. They also suffer from several problems in that the device still needs to be polled very rapidly so as not to miss any incoming X-10 signals. X10 signals can appear at any zero crossing of the 60 Hz power or every 16.67 mS.

The Marrick Limited solution was to go back to ground zero. We designed a complete TW523 (X-10) compatible controller and interface into a single chip. The LynX-PORT™ controller chip off-loads the computer of the complex timing and constant polling to ensure data integrity.

The Marrick LynX-PORT™ Remote I/O Controller provides 8 programmable relays, 8 optically coupled, active current limited inputs, and 4 - 8 bit analog channels. The board can be located remotely (via X-10 or RS-232) or locally using the serial port to connect to a local host. By using a TW-523, the host can also have access to the X-10 powerline signals. The following describes the various features of the Marrick LynX-PORT™ Remote I/O Controller.

2.0 TECHNICAL OVERVIEW

2.1. INTRODUCTION TO A LynX-PORT™ SESSION

The computer interface uses ASCII commands to carry out its functions. These commands are outlined below. The basic structure of the LynX-PORTTM protocol is a subset of the normal LynX-10TM protocol. The letter 'X' is used to tell the LynX-PORT board that an X-10 command is being issued. The ‘X’ is followed by either a ‘0’ (zero) or a ‘1’ to tell the board whether to send an address or command. Two more characters are sent to indicate the house code and key code. For example, if you wanted to turn on unit A4, you would send the ASCII string 'X003' (addresses unit A4) followed by ‘X102’ (commands addressed units on house code A to turn on). The controller would respond with an asterisk ('*') when each command is completed, or an 'E' if there was an error followed by a number to further explain the error. All return strings are terminated with an optional “Return String Terminator” (RST). The default RST is the carriage return character (ASCII 0x0D). Here is a typical X-10 session using a terminal emulator on the ...

COMPUTER:	X000	; Address unit A1
CONTROLLER:	*	; Done... (asterisk is followed by the programmed return string)
COMPUTER:	X102	; Turn on addressed units on house code A
CONTROLLER:	*	; Done...
CONTROLLER:	X014	; Controller reports X-10 Received (HC=B, Unit 5)
CONTROLLER:	X113	; Controller reports X-10 Received (HC=B, Off)

All X-10 commands that are sent or received take this form. Each X-10 command is 4 bytes long, however, some commands are either longer or shorter. If the controller needs more data, the green 'BUSY' LED will stay lit until enough bytes have been received to satisfy the command requested. The first byte is the COMMAND byte, the next byte (if required) is the TYPE byte and is used to further define the action. The next byte is the HOUSE CODE byte (ASCII letters for 0-F HEX) and describes house codes A-P. The last byte (if required) is the ADDRESS or X-10 COMMAND byte (0-F HEX) that describes the unit address. Following is an overview of all the commands and their extensions.

2.2. LynX-PORT™ PROTOCOL

LynX-PORT™ uses a condensed method for addressing and reporting unit codes. Since there are exactly 16 house codes (A-P) and 16 unit codes (1-16), it is convenient to use a hexadecimal equivalent mapping. For those not completely familiar with the hexadecimal number system, refer to Appendix B. Each house code is mapped to a hex number to allow only 1 digit per house code and 1 digit for unit code address.

Commands can take several forms, but most follow these simple rules.

1. All characters sent to the LynX-PORT are printable ASCII codes. That is, you can send them to a terminal or printer and see the letter or number. The actual value of the code is quite different. For example, the letter “A” is ASCII code 0x41 in hexadecimal or 65 decimal. The letter “A” in our context stands for 10 decimal. One exception is the carriage return character, used in some commands, and the return string characters returned after an X-10 reception. See MODE register for details on return string terminators (RST).
2. The code structure always begins with a command code. These codes are listed on the following pages along with examples of their structure. For example, to address an X-10 unit, the command is “X0xy” where “xy” are found in table 1 from the house and unit code of the module you desire to turn on. If you wanted to address unit C3, the string of characters would be “X022”

2.3. MAPPING X-10 UNITS TO LynX-PORT™ CODES

To make interfacing to computers easier, a special code system is used for addressing specific units and house codes. In table 1, every possible code that can be set on an X-10 module is mapped to its hexadecimal value used with the LynX-PORT™ protocol. For example, if you want to address unit E10, you would look up the letter “E” along the left axis, and the number “10” on the top axis of table 1. Where the two columns intersect are the two characters used with LynX-PORT™ to address that unit.

2.4. LynX-PORT™ PROTOCOL LOOKUP TABLES

Table 1 - Unit code to LynX-PORT™ protocol cross reference

UC / HC	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
B	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
C	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
D	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
E	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
F	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
G	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
H	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
I	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
J	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
K	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
L	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
M	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
N	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
O	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
P	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

(HC = House code, UC = Unit code -> mapping to 2 ASCII printable characters)

Table 2 - House code to hexidecimal conversion

House code	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Hex Value	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

2.4.1 LynX-PORT™ Protocol Lookup Table Notes

The programmer should use the table above to determine the actual characters sent to address a unit within an X-10 command. Use table 1 to look up the two characters that follow the “X0” to address the unit. Let’s say we want to address unit M15. We would go to table 1 and find the house code M on the left and the unit number or unit code 15 on the top. Tracing the intersection of these two we would find the two characters “CE” which is a character string representing the hexadecimal value of 0xCE. Therefore the entire command string sent to the LynX-PORT™ unit would be represented by “X0CE”. That’s the letter “X” followed by the number “0” (zero), followed by the letter “C” followed by the letter “E”.

3.0 COMMAND SUMMARIES

3.1 X-10 STATUS Command Summary

Command	Description	Example / Comments
S0 thru S2	(Reserved)	(Not used on this version)
S3	Returns the current status byte of the LynX-PORT IO unit. This takes the form "S=xx" where xx is the status byte.	"S3" Returns "S=43"
S4 thru SF	(Reserved)	Returns error E1

3.1.1 X-10 STATUS Command Notes

This command is used to check the internal status of the LynX-PORT unit while it is running. The bit definitions are as follows:

LynX-PORT Status byte bit definitions (Bit 0 is LSB)

BIT	Description
0	TW-523 Power (0=Power is OK at TW-523, 1=Power failed at TW-523)
1	Power frequency (0=60 Hz, 1=50Hz)
2	EEPROM status (0=Failed to initialize, 1=Working)
3	XOFF sent to host (0=Not sent, 1=Sent)
4	XON sent to host (0=Not sent, 1= Sent)
5	(Reserved for internal operation)
6	X-10 Command is active for this board (0=No, 1=Yes)
7	X-10 Extended Protocol in progress (0=No, 1=Yes)

NOTE - An "S" command followed by a carriage return will also return the same information.

3.2. DIRECT X-10 ACCESS Command Summary

Command	Description	Example / Comments
X0*xy*	Addresses unit *xy* for next command (*xy* from table 1)	"X067" address X-10 unit G8
X1*x***0**	Sends an ALL UNITS OFF command to house code *x* (*x* from table 2)	"X1F0" sends ALL UNITS OFF to house code P
X1*x***1**	Sends an ALL LIGHTS ON command to house code *x* (*x* from table 2)	"X141" sends ALL LIGHTS ON to house code E
X1*x***2**	Sends an ON command to house code *x* (*x* from table 2)	"X102" sends ON to house code A
X1*x***3**	Sends an OFF command to house code *x* (*x* from table 2)	"X113" sends OFF to house code B
X1*x***4**	Sends a single DIM to house code *x* (*x* from table 2)	"X154" sends DIM to house code F
X1*x***5**	Sends a single BRIGHT to house code *x* (*x* from table 2)	"X155" sends BRIGHT to house code F
X1*x***6**	Sends ALL LIGHTS OFF command to house code x (x from table 2)	"X106" sends ALL LIGHTS OFF to house code A
X1*x***7**	Sends EXTENDED CODE to house code *x* (*x* from table 2) ***note: see text***	"X127" sends EXTENDED CODE to house code C
X1*x***8**	Sends HAIL REQUEST to house code *x* (*x* from table 2) ***note: see text***	"X1B8" sends HAIL REQ to house code L
X1*x***9**	Sends HAIL ACKNOWLEDGE to house code *x* (*x* from table 2) ***note: see text***	"X1B9" sends HAIL ACK from house code L
X1*n***A**	Sends PRESET DIM with MSB of 0 and *n* level (*n* from table 3) ***note: see text***	"X15A" commands last addressed light to level 5
X1*n***B**	Sends PRESET DIM with MSB of 1 and *n* level (*n* from table 3) ***note: see text***	"X12B" commands last addressed light to level 18
X1*x***C**	Sends EXTENDED DATA to house code *x* (*x* from table 2) ***note: see text***	"X10C" sends EXTENDED DATA to house code A
X1*x***D**	Sends STATUS ON for house code *x* (*x* from table 2) ***note: see text***	"X15D" sends STATUS ON to house code F
X1*x***E**	Sends STATUS OFF for house code *x* (*x* from table 2) ***note: see text***	"X15E" sends STATUS OFF to house code F
X1*x***F**	Sends STATUS REQUEST to house code *x* (*x* from table 2) ***note: see text***	"X12F" requests status from last addressed unit on house code C
X2 thru XF	(Reserved)	Returns error E1

3.2.1 Direct X-10 Access Command Notes

The direct access commands allow the programmer to have direct control over X-10 modules and subsystems. This provides the most flexibility, but also requires more attention to detail. Code X0 is used to address units, and can be used several times in a row on the same house code to address a group of units. The X1 command along with its subcommand is used to tell the unit or units what to do, or to request information from advanced X-10 units. If a group is addressed, only a single command is needed to cause all the units to respond. This is useful when turning on or off several units together. It is also quicker to do so in that the individual commands are not sent. So if seven units are addressed and then a single on command is sent, there will be a net savings of 6 commands or over 2 seconds.

Certain X-10 codes such as HAIL REQUEST or STATUS REQUEST require advanced X-10 modules to operate. Not all modules will respond to these commands. Ask the manufacturer for details on what commands are supported and how they are implemented. There are two commands called EXTENDED CODE and EXTENDED DATA. These commands can be transmitted with the LynX-PORT board, but their documented implementation cannot do to the limits of the TW-523 interface module. These codes require a continuous data stream without the normal 3 power line cycle idle periods between them. The TW-523 will not receive these codes. Therefore, only the command itself without the data can be sent. This still can be utilized to activate remote equipment, or to tell a unit the next address it receives is data, not address. This would allow a “nibble” mode of data transfer between LynX-PORT subsystems. The Marrick Limited LynX-PORT™ remote I/O board uses this method for remote programming of settings and parameters.

3.3 RESET Command Summary

Command	Description	Example / Comments
R	Resets LynX-PORT™	“R” Resets LynX-PORT, opens all relays, and clears all timers.

3.4 LynX-PORT™ MICROCODE VERSION REQUEST Command Summary

Command	Description	Example / Comments
V0	Returns current version number of LynX-PORT™ ’s microcode to computer in the form Vxxx-xxx where xxx-xxx is the version.	“V0” returns current version to computer.
V1	Returns copyright of LynX-PORT™ microcode to computer. This is always “Copyright (c) xxxx Marrick Limited, Inc.” where xxxx is the year of release.	“V1” returns current date of copyright to computer.
V2 thru VF	(Reserved)	Returns error E1

Note: Sending carriage return after the “V” returns both the copyright and the version number.

3.5 LynX-PORT™ PROGRAMMING Command Summary

Command	Description	Example / Comments
Pxx ↵	Returns the word (2 bytes) stored in the 16 bit register in the form "Pxx=yyyy" where xx is the requested location (00-3F) and yyyy is the hex data in that register. All internal registers are accessed with this command	"P00" followed by a carriage return will return "P00=AA55" which is the EEPROM signature.
Pxx=yyyy↵	This will program the register at location xx (00-3F) with data yyyy.	"P01=0100" will set the house code to 'B' and clear the mode register.

3.5.1 LynX-PORT™ Programming Command Notes

The symbol ↵ represents a carriage return or ASCII 0x0D. Also, all EEPROM locations can be accessed as shown above. The table below shows the mapping of all of the internal registers of the LynX-PORT™ Remote I/O Controller model 201.

Register Map Locations and Definitions

Location	High Byte description	Low Byte description
00	Signature high (0xAA)	Signature low (0x55)
01	Main house code (00-0F)	Mode register
02	FIFO Threshold (00-0C)	Maximum X-10 collisions during TX
03...07	(Reserved)	(Reserved)
08	Counter - Data comm. errors	
09	Counter - Bad commands	
0A	Counter - Bad data	
0B	Counter - X-10 collisions	
0C	Counter - X-10 TX failures	
0D	Counter - X-10 lost receptions	
0E	Counter - Async FIFO overruns	
0F	Counter - Lost power at TW523	
10	Relay 1 setup	Input 1 setup
11	Relay 2 setup	Input 2 setup
12	Relay 3 setup	Input 3 setup
13	Relay 4 setup	Input 4 setup
14	Relay 5 setup	Input 5 setup
15	Relay 6 setup	Input 6 setup
16	Relay 7 setup	Input 7 setup
17	Relay 8 setup	Input 8 setup
18...1F	(Reserved)	(Reserved)

Register Map Locations (continued)

20	Relay 1 timer (16 bit)	
21	Relay 2 timer (16 bit)	
22	Relay 3 timer (16 bit)	
23	Relay 4 timer (16 bit)	
24	Relay 5 timer (16 bit)	
25	Relay 6 timer (16 bit)	
26	Relay 7 timer (16 bit)	
27	Relay 8 timer (16 bit)	
28...2F	(Reserved)	(Reserved)
30	Analog ch 1 grp 1 low data threshold	House and unit code for command
31	Analog ch 2 grp 1 low data threshold	House and unit code for command
32	Analog ch 3 grp 1 low data threshold	House and unit code for command
33	Analog ch 4 grp 1 low data threshold	House and unit code for command
34	Analog ch 1 grp 2 low data threshold	House and unit code for command
35	Analog ch 2 grp 2 low data threshold	House and unit code for command
36	Analog ch 3 grp 2 low data threshold	House and unit code for command
37	Analog ch 4 grp 2 low data threshold	House and unit code for command
38	Analog ch 1 grp 1 Hi data threshold	House and unit code for command
39	Analog ch 2 grp 1 Hi data threshold	House and unit code for command
3A	Analog ch 3 grp 1 Hi data threshold	House and unit code for command
3B	Analog ch 4 grp 1 Hi data threshold	House and unit code for command
3C	Analog ch 1 grp 2 Hi data threshold	House and unit code for command
3D	Analog ch 2 grp 2 Hi data threshold	House and unit code for command
3E	Analog ch 3 grp 2 Hi data threshold	House and unit code for command
3F	Analog ch 4 grp 2 Hi data threshold	House and unit code for command

3.5.2 Relay and Input bit definitions

Below is a table of each bit in the 16 bit word programmed into registers 0x10 through 0x17. These effect the relay and input behavior.

Input setup register bit definitions (0x10 thru 0x17 low byte)

Bit	Description
0	INPUT - Polarity (0=normal, 1=Inverted)
1	INPUT - ON Active (0=ON state disabled, 1=ON state enabled)
2	INPUT - OFF Active (0=OFF state disabled, 1=OFF state enabled)
3	INPUT - Send X-10 command (0=Off, 1=On)
4	INPUT - Connect input to slave relay (0=No slave, 1=Slave to selected relay)
5	INPUT - Slave relay address bit 0
6	INPUT - Slave relay address bit 1
7	INPUT - Slave relay address bit 2

Relay setup register bit definitions (0x10 thru 0x17 high byte)

8	RELAY - Enable X-10(0=Ignores incoming X-10 ON/OFF commands, 1=accepts X-10 ON/OFF commands)
9	RELAY - Exclusive relay operation (0=independant, 1=Exclusive within group)
10	RELAY - Switch to next group in relay on timeout when exclusive (0=No, 1=Yes)
11	RELAY - Toggle relay when actived from slave input (0=No, 1=Yes)
12	RELAY - Echo state change to X-10 (0=No, 1=Yes)
13	(Reserved)
14	RELAY - Group number bit 0
15	RELAY - Group number bit 1

3.5.3 Relay timer description and other options

Each relay has associated with it a 16 bit timer with a resolution of 0.5 seconds. This means that any relay can be programmed to automatically open some time after it has been commanded to close. This can be as short as a half of a second to over 9 hours in length. The relays can also be programmed to automatically trip the next relay in an exclusive group. This is useful for sprinklers and other automatic cascading systems. Optionally the relays can be programmed to send an X-10 command when their state changes. This makes it easy to watch the activity of the remote automatic system.

To setup a relay to time out after a period simply program the timer with the value in half second increments. For instance, if you want to program relay 1 to time-out after 60 seconds you would issue a **P20=0078**. This command will load location 0x20 with 0x0078 which equals 120 in decimal (120 * 0.5 seconds = 60 seconds). The LynX-PORT configuration utility will do this directly for you.

To optionally have that relay turn on the next relay in the group you must first define which relays will be grouped together. The group can consist of any of the relays, however, they have priority from 1 (highest) to 8 (lowest). You can program up to 4 groups. A relay can exist as its own group, but the function is useless without another relay in the group to act on. Next select the *Switch to next relay in group* option bit. This will turn on the next relay when it times out. For example, Relay 1 is now programmed to time out after 60 seconds (see above). To program it to turn on the next relay in group 0, you would issue the command string **P10=17nn** where **nn** is the current setup for Input 1. This would program relay 1 to accept X-10 commands, set its operation as exclusive in group 0, and to automatically turn on the next relay in the group when it times out.

3.5.4 Analog register definitions

The following describes the use of the analog window function designed as a “watch dog” window comparitor for the analog inputs. The general idea is that if an analog input exceeds a given “window” voltage, an X-10 command is transmitted. To use this feature simply program the channel registers with the high and low thresholds along with the units you wish to activate when the limits are exceeded. For example, lets examine how to setup channel 1 to trip when it drops below 2.4 volts and send unit A1 an OFF command, and trip when it goes above 2.6 volts and send unit A2 an ON command. We would issue the following commands: **P30=7B00** and **P34=8501**. You can also use the LynX-PORT configuration utility which is much quicker and easier. Location 0x30 is the group 1 (you have 2 groups to use, but only one at a time) low threshold setting for channel 1. The 7B is the hexidecimal value for the 2.4V trip point. It is relative to +5V. The 00 is the house code (0=A) and unit code (0=1) for the off command. Optionally in the mode register, you can program the LynX-PORT to send X-10 STATUS OFF / STATUS ON commands in place of the basic ON / OFF commands. This can be useful when you do not wish to activate a device when the thresholds are exceeded. Location 0x34 is the group 1 high threshold setting for channel 1. The 85 is the hexidecimal value of the 2.6 volt trip point relative to +5V and the 01 is the house and unit code. To calculate the value to place into the high order byte of the register, use this equation:

Register value = (Vt * 256) / 5

Note that this will give you a decimal value. You must then convert it into a hexidecimal value and use that in the high order byte of the register as shown above.

All channels work exactly the same. You can switch between group 1 and group 2 by changing bit 5 of the mode register. You can also change whether the LynX-PORT unit will send ON / OFF or STATUS ON / STATUS OFF commands via X-10 and the serial port when the thresholds are exceeded by changing bit 3 of the mode register.

3.6 LynX-PORT™ Analog Port Command Summary

Command	Description	Example / Comments
A0	Reads differencially between channels 1 (+) and 1 (-). Returns in the form A0=xx where xx is the hexidecimal representation of the ADC conversion.	"A0" returns "A0=5F" which is the value of channel 1 minus channel 2.
A1	Reads differencially between channels 3 (+) and 4 (-). Returns in the form A1=xx where xx is the hexidecimal representation of the ADC conversion.	"A1" returns "A1=4C" which is the value of channel 3 minus channel 4.
A2	Reads differencially between channels 1 (-) and 2 (+). Returns in the form A2=xx where xx is the hexidecimal representation of the ADC conversion.	"A2" returns "A2=F1" which is the value of channel 2 minus channel 1.
A3	Reads differencially between channels 3 (-) and 4 (+). Returns in the form A3=xx where xx is the hexidecimal representation of the ADC conversion.	"A3" returns "A3=16" which is the value of channel 4 minus channel 3.
A4	Reads channel 1 of ADC. Returns in the form A4=xx where xx is the hexidecimal representation of the ADC conversion.	"A4" returns "A4=65" which is the value of channel 1 referenced to +5V.
A5	Reads channel 3 of ADC. Returns in the form A5=xx where xx is the hexidecimal representation of the ADC conversion.	"A5" returns "A5=19" which is the value of channel 3 referenced to +5V.
A6	Reads channel 2 of ADC. Returns in the form A6=xx where xx is the hexidecimal representation of the ADC conversion.	"A6" returns "A6=22" which is the value of channel 2 referenced to +5V.
A7	Reads channel 4 of ADC. Returns in the form A7=xx where xx is the hexidecimal representation of the ADC conversion.	"A7" returns "A7=FA" which is the value of channel 4 referenced to +5V.
A8 thru AF	(Reserved)	Returns error E1

4.0 ERROR CODES

4.1 LynX-PORT™ ERROR CODES:

NOTE: SENT TO COMPUTER FROM LynX-PORT™

E0 : RECEPTION ERROR (X10)

E1 : BAD COMMAND RECEIVED FROM COMPUTER

E1 : BAD DATA RECEIVED FROM COMPUTER

E3 : X10 COLLISION DETECTED DURING TRANSMISSION

E4 : X10 TRANSMISSION FAILURE (TIME OUT)

E5 : X10 LOST RECEPTION (LOST TX ECHOES FROM TW-523)

E6 : SERIAL COMMUNICATION RX FIFO OVER-RUN

E7 : CARRIER LOST (50/60Hz POWER FAILURE AT TW-523)

NOTE: All error codes map to statistics counters one to one. Example: if an E4 error is received by the computer, the statistics counter 4 will be incremented by 1.

5.0 HARDWARE CONSIDERATIONS

5.1 COMMUNICATIONS

The LynX-PORT™ I/O Controller communicates with a local host at a fixed rate of 1200 bits per second (bps). This may seem slow, but the actual symbol rate on the power line is only 60 symbols per second. It takes 11 symbols to complete 1 address or command and each group of 11 symbols are transmitted twice to assure reception. Most commands such as “ON” and “OFF” require an address (House code and unit ID) and a command (House code and command). Each are 11 symbols long including the start field. These are sent twice which means the actual rate at which things turn on or off is about 44 cycles of the 60 cycle power or over 2/3 of a second per 4 bytes of data from the computer or about 22 bps. The full setup for the computer’s serial port is:

1200 BPS, NO PARITY, 8 DATA BITS, 1 STOP BIT

5.2 HAND SHAKING

The LynX-PORT™ has a 16 byte queue that holds transmitted data from the computer. Once the command begins sending X-10 data on the power line, the LynX-PORT™ can continue to receive data until the FIFO threshold is reached. At this point, the coprocessor chip de-asserts Clear To Send (CTS) and will optionally send an XOFF (0x13) code to the computer (see MODE register settings). When the level of the internal FIFO falls below the threshold level, the CTS signal is re-asserted and optionally the XON (0x11) code is sent to the computer. From the factory, the threshold is set at 8 (half the size of the FIFO). This will allow two commands to be in the queue as they are being processed. Optionally this threshold can be set anywhere between 1 and 15 bytes. We recommend leaving it set for 8 bytes for optimal performance and best communications integrity. ***NOTE: If the internal FIFO is over-run, data will be lost and an E6 error will be sent to the computer for each dropped character***.

Return data from the power line is decoded into raw X10 commands that all begin with an ‘X’ as described in the command section above and sent directly to the computer with no handshaking. A receive queue must be large enough or serviced frequently enough to prevent overrun of receive data. Most communications drivers have sufficient queues including those for Windows, Windows NT, Windows 95, UNIX, OS/2, and DOS.

6.0 SOFTWARE CONSIDERATIONS

6.1 MODE REGISTER

The mode control register sets the programmable features of the LynX-PORT™. When the controller is powered up or reset, the settings saved in the EEPROM will be automatically reloaded. The usage of each bit is described below. To set these bits, the programmer must write an entire hexadecimal value to the mode register using the "P01=yyxx" command where xx is the hexadecimal byte and yy is the house code of the board. The mode register can be read back out by issuing a "P01" string followed by a carriage return (0x0D).

Figure 1 - Mode register bit definitions

Bit	Description
0	Debug mode (0=Off, 1=On)
1	(Reserved)
2	XON / XOFF handshaking (0=Off, 1=On)
3	X-10 reply code type (0=ON / OFF, 1=STATUS ON / STATUS OFF)
4	(Reserved)
5	Analog window data group (0=Group 1, 1=Group 2)
6	Return string type - bit 0 (00=CR, 01=LF, 10=NL, 11=none)
7	Return string type - bit 1

6.1.1 DEBUG Mode

The LynX-PORT™ can be put into a mode that will send both the 'X' command and the raw bits received off the power line to the computer. When DEBUG mode is enabled in the MODE register, any data received from the TW-523 will be sent to the computer as raw bits followed by the 'X' interpreted commands. The raw bits are prefaced by the letter "Y" to indicate the data type to allow software parsing. Example: Received data might look like this in DEBUG mode.

```
Y1110010101010101100101
X0C2
Y1110010101010101100110
X1C2
```

The characters after the "Y" represent the undecoded raw X-10 bits from the TW-523. A "1" character represents the presence of the 120khz signal on a zero crossing. A "0" character represents the absence of the 120khz signal. See the Power House TW-523 specification for details on how these bits are encoded. This mode is enabled by setting bit 0 of the MODE register (see MODE register above)

6.1.2 Return String Terminators

When a command is successfully completed, the LynX-PORT™ sends an asterisk ("*" - 0x2A) to the computer. The LynX-PORT™ has the ability to return several string termination characters following the asterisk. These are CR (carriage return - 0x0D), LF (line feed - 0x0A), NL (new line - CR & LF), or no string terminator. This allows the programmer to use the return string character that the operating system can handle. See the MODE register for details on how to set this up.

6.1.3 50/60Hz Automatic Detection and Switching

The LynX-PORT™ automatically detects the frequency of the line current. This is monitored every cycle of the power line and the internal timing is automatically adjusted to either 50 Hz (Europe) or 60 Hz (USA and Canada). No user intervention is required.
NOTE: The Power House TW-523 must be the correct version for the region of the world in which it will be used.

6.1.4 Power requirements

The LynX-PORT™ requires 500mA minimum supply current at 9-16VAC or DC. The onboard power supply regulates the voltage to +5VDC for all circuitry. The raw input voltage is provided at 2 terminals on each input to supply power to external devices or provide loop current for contact closures.

Appendix A - Extended Protocol

The LynX-PORT™ IO unit can be programmed and monitored over the power line using other X-10 equipment such as the Marrick LynX-10™ or LynX-10PC™ Coprocessor. The protocol uses the EXTENDED CODE and EXTENDED DATA commands originally defined by Power House. The setup software provided with the LynX-PORT™ unit uses this protocol to configure remote boards using either a local LynX-PORT™ or any of the LynX-10™ Coprocessor family.

The original format of the extended code and extended data commands required the actual data to follow the command with out any power line cycle breaks between them. This is the case for DIM and BRIGHT commands. However, the standard TW-523 interface unit designed by Power House, does not support this continuous code format. Therefore, you would never see the extended data parts of these commands since the TW-523 would reject it as a bad command and not pass it to the equipment.

The fix was to only send the command with out the data. The data is now encoded in the house code section of the command much like the PRESET DIM command operates. The format is as follows. All commands to a remote device begin with the EXTENDED CODE command sent to that device's house code. This signals the board that extended data is about to be sent and places the board in programming (extended protocol) mode. The data now occupies the house code component of a series of EXTENDED DATA commands. The exact number of these depends on the actual encoded command being sent. The table below shows the extended functions and the required number of "nibbles" inserted into the house code. The encoding is shown below as well. See the SDK for more details on this protocol.

Data Mapping to House codes for extended protocol

House Code	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Data value (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Extended Protocol Commands

Command	Description	Data required
0	Extended Code Acknowledge	N/A (From LynX-PORT)
1	Extended Code Fail	N/A (From LynX-PORT)
2	Read EEPROM location	2 nibbles (address)
3	Write EEPROM location	6 nibbles (address + data)
4	Read ADC	1 nibble (Channel + mode)
5	Read aux device	2 nibbles (address)
6	Write aux device	4 nibbles (address + data)
7...C	(Reserved)	N/A
D	Version request	N/A
E	Status request	N/A
F	Reset	N/A

Appendix B - Hexadecimal Numbering Overview

Table 6 - Hexadecimal and decimal relationship

Bit 3	Bit 2	Bit 1	Bit 0	Decimal	Hexadecimal
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	8
1	0	0	1	9	9
1	0	1	0	10	A
1	0	1	1	11	B
1	1	0	0	12	C
1	1	0	1	13	D
1	1	1	0	14	E
1	1	1	1	15	F

The HEXADECIMAL system is an outgrowth of the binary system of 1's and 0's used in computers. In hexadecimal, one character is used to define four bits. This is handy when using a large number of bits as in today's 32 bit microprocessors. Writing 32 ones and zeros in a row can take up some space. In hexadecimal (usually called "HEX"), it would only take 4. In the table above, all the possible combinations of the 1's and 0's are defined with both the decimal (known as base 10 - the numbers we humans use) and hexadecimal (base 16) numbers. Since our normal base 10 system only has 10 characters in it, we need to extend the characters with letters to reach 16. This is why A,B,C and so on, up to the letter F are used. This provides 6 more characters to indicate the remaining combinations. Putting two hex characters next to each other now provides a place holder for not 10 but 16. That is, the most significant digit stands for the number of 16's there are. For example, if you had the hex value 32 (usually represented by 0x32 to indicate hex), there would be (3x16) + (2x1) counts or a decimal 50. The next digit would be the counts of 256 (16x larger), and the next would be the counts for 4096. If we had the hex value 1A5F, that would be equal to the decimal value of,

(1x4096)+(10x256)+(5x16)+(15x1) = 6751

or in binary: 0001101001011111. You can break the binary apart into 4 bit groups and see where the hex number is derived from. The 0001 is 1, 1010 is A, 0101 is 5, and 1111 is F. You can see how convenient this method is. There are calculators designed to do math in base 16. The Windows 3.1 Calculator program has a scientific mode that supports hex arithmetic. Try it out for your self. Hexadecimal numbers are sometime written with a subscript 16 like this: **19_{16}** which indicates base 16. The 19 in this case would be equivalent to a decimal 25. If they are mixed with base 10 numbers, this notation really helps.

Most computer books discuss hexadecimal arithmetic, so if this is still not clear, you may wish to make a trek to your favorite book store and check out the ever growing computer section for some help.

Shown below in table 6 is the American Standard Code for Information Interchange or ASCII. The left axis is the least significant digit in both hex and binary, and the top is the most significant digit. This table is provided for your reference.

MSD LSD	**0 0000**	**1 0001**	**2 0010**	**3 0011**	**4 0100**	**5 0101**	**6 0110**	**7 0111**
0 0000	NUL	DLE	SP	0	@	P	‘	p
1 0001	SOH	DC1	!	1	A	Q	a	q
2 0010	STX	DC2	“	2	B	R	b	r
3 0011	ETX	DC3	#	3	C	S	c	s
4 0100	EOT	DC4	$	4	D	T	d	t
5 0101	ENG	NAK	%	5	E	U	e	u
6 0110	ACK	SYN	&	6	F	V	f	v
7 0111	BEL	ETB	‘	7	G	W	g	w
8 1000	BS	CAN	(	8	H	X	h	x
9 1001	HT	EM	)	9	I	Y	i	y
A 1010	LF	SUB	*	:	J	Z	j	z
B 1011	VT	ESC	+	;	K	[	k	{
C 1100	FF	FS	,	<	L	\	l	|
D 1101	CR	GS	-	=	M	]	m	}
E 1110	SO	RS	.	>	N	↑	n	~
F 1111	SI	VS	/	?	O	←	o	DEL