

Get Latched I/O

->Send 8 binary bytes: <200> <50> <0> <0> <0> <0> <0> <250>
 <-Receive 3 byte header: <6> <0> <6>
 <-Receive 256 bytes data: <unit 1 Data1> <unit 1 Data2> <unit 2 Data1> <unit 2 Data2> etc...

Data Explanation:

SECU 16, Type 11	Data1 = Latched Input Bits 0-7 Data2 = Latched Relay Bits 0-7
SECU 16I, Type 12	Data1 = Latched Input Bits 0-7 Data2 = Latched Input Bits 8-15
RLY08-XA, Type 13	Data1 = 0 Data2 = Latched Relay Bits 0-7

Get Real Time I/O

->Send 8 binary bytes: <42> <0> <0> <141> <181> <0> <37> <233>
 <-Receive 6 byte header: <n> <n> <n> <n> <n> <n>
 <-Receive 256 bytes data: <unit 1 Data1> <unit 1 Data2> <unit 2 Data1> <unit 2 Data2> etc...
 <-Receive 2 byte CRC <n> <n>

Note: You may have to program a short delay between sending the command and receiving the 264 byte response.

Data Explanation:

SECU 16, Type 11	Data1 = Real Time Input Bits 0-7 Data2 = Latched Relay Bits 0-7
SECU 16I, Type 12	Data1 = Real Time Input Bits 0-7 Data2 = Real Time Input Bits 8-15
RLY08-XA, Type 13	Data1 = 0 Data2 = Latched Relay Bits 0-7

Get Unit Type

->Send 8 binary bytes: <42> <0> <0> <1> <176> <0> <148> <39>
 <-Receive 6 byte header: <n> <n> <n> <n> <n> <n>
 <-Receive 256 bytes data: <unit 1 Type> <unit 2 Type> <unit 3 Type> <unit 4 Type> etc...
Note: only the first 128 bytes of data are valid
 <-Receive 2 byte CRC <n> <n>

Data Explanation:

SECU 16, Type = 11
SECU 16I, Type = 12
RLY08-XA, Type = 13

Get Unit Firmware Version

->Send 8 binary bytes: <42> <0> <0> <130> <176> <0> <246> <45>
 <-Receive 6 byte header: <n> <n> <n> <n> <n> <n>

<-Receive 256 bytes data: <unit 1 Version> <unit 2 Version> <unit 3 Version> <unit Version> etc...

Note: only the first 128 bytes of data are valid

<-Receive 2 byte CRC <n> <n>

Data Explanation:

Version is the firmware revision loaded into the unit

Initiate Get Unit Parameters

->Send 8 binary bytes: <200> <31> <pnum> <0> <0> <0> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA to retrieve parameter <pnum> from every unit

Data Explanation:

pnum = the parameter number to get

csum = the 8-bit sum of 200 + 31 + pnum + 0 + 0 + 0 + 0

Write Relay Output

->Send 8 binary bytes: <200> <51> <unit> <relaynum> <stat> <0> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note:

Data Explanation:

unit = the unit address to write to

relaynum = the relay number to set/clear (0 to 7)

stat = the desired status (1 = on, 0 = off)

csum = the 8-bit sum of 200 + 51 + unit + relaynum + stat + 0 + 0

Get Unit Parameters

->Send 8 binary bytes: <42> <0> <0> <139> <180> <0> <164> <120>

<-Receive 6 byte header: <n> <n> <n> <n> <n> <n>

<-Receive 256 bytes data: <unit 1 Param LSB> <unit 1 Param MSB> <unit 2 Param LSB> <unit 2 Param MSB>
etc...

Note: this data is what is stored at parameter <pnum> in each unit

<-Receive 2 byte CRC <n> <n>

Initiate CPU-XA RESCAN

->Send 8 binary bytes: <200> <32> <0> <0> <0> <0> <0> <232>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA rescan for active units

Initiate CPU-XA RESTART

->Send 8 binary bytes: <200> <52> <0> <0> <0> <0> <0> <252>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA do a hard reset (same as power-up)

Write Unit Parameter Data

->Send 8 binary bytes: <200> <33> <unit> <LSB data> <pnum> <MSB data> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA to write parameter <pnum> to unit

Data Explanation:

unit = the unit address to write to

LSB data = the low byte of the 16 bit data

pnum = the parameter number to write

MSB data = the high byte of the 16 bit data

csum = the 8-bit sum of 200 + 33+ unit + LSB + pnum + MSB

Write CPU-XA Parameter Data

->Send 8 binary bytes: <200> <40> <pnum> <MSB data> <LSB data> <0> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command writes a parameter <pnum> into the CPU-XA

Data Explanation:

pnum = the parameter number to write

LSB data = the low byte of the 16 bit data

MSB data = the high byte of the 16 bit data

csum = the 8-bit sum of 200 + 40 + pnum + MSB + LSB + 0

Parameter Number Explanation:

1	-	not used
2	power mode	1 = low power mode, 0 = normal mode
3	max unit	highest unit number to scan to
4	DST status	1= DST in force, 0 = DST not in force
5	DST enable	1 = check for DST, 0 = do not check for DST
6	auto touch response	0 = do not respond 1 = respond with virtual button match 0xf2, modnum, LSBbutton, MSBbutton 2 = respond with grid location (0-59) 0xf3, modnum, LSBgrid, 0
7	rescan time	auto rescan in minutes (0 = off)
8	net timeout	time in 1/10ths sec to wait for a module to respond
9	retry counter	give up talking to a module after this many retries.
10	security code 1	(system interface only)
11	security code 2	(system interface only)
12	security code 3	(system interface only)
13	pager config	0 = 1 pager, 15 messages, 1 = 5 pagers, 2 messages each
14	cpu-xa num	CPU-XA address (set to 0)
15	auto X10	send X10 (0 = off, 1 = on) 0xfe (RX),hc,kc when rx'd 0xfb (TX),hc,kc when tx'd
16	auto I/O	send 0xff if remote i/o status has changed (0 = off, 1 = on)
17	auto ir	send IRnumber when compare exists, (0 = off, 1 = on)

18	send ASCII Ir	0xfd (RX) or 0xfc (TX) , IRnumber 0 = off 1 = Send ASCII string on IR recognize +T000xxx xxx = IR number recognized
19	Leopard	0 = CPUXA, 1 = Leopard
20	Compare IR	1-1023, Max IR to compare
21	RCS Status Response	1=On, 0= Off, Show RCS Temperatures in Vars 64-
22	Clear Variables	First Var # to leave after power-up

Send X-10

->Send 8 binary bytes: <200> <55> <house> <key> <repeat> <0> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA to add the X-10 house and key codes to it's send buffer

Note: You may have to program a short delay between sending the command and receiving the 3 byte response.

Data Explanation:

house = the X-10 house code to send (see Below)

key = the X-10 key code to send (see Below)

repeat=the number of times to repeat this command (use 1mostly for Dim and Bright)

csum = the 8-bit sum of 200 + 55+ house + key + repeat+ 0

X10 House Code	CPU-XA House Code	X10 Key Code	CPU-XA Key Code
A	0	1	0
B	1	2	1
C	2	3	2
D	3	4	3
E	4	5	4
F	5	6	5
G	6	7	6
H	7	8	7
I	8	9	8
J	9	10	9
K	10	11	10
L	11	12	11
M	12	13	12
N	13	14	13
O	14	15	14
P	15	16	15
		All Units Off	16
		All Lights On	17
		On	18
		Off	19
		Dim	20
		Bright	21
		All Lights Off	22
		Extend Code	23
		Hail Req.	24
		Hail Ack.	25
		Preset Dim 0	26
		Extend Data	27
		Status ON	28
		Status OFF	29
		Status Req	30
		Preset Dim 1	31

Load Interpreter

->Send 8 binary bytes: <200> <53> <lo_linenum> <hi_linenum> <lo_data> <hi_data> <index> <csum>
<-Receive 3 byte header: <6> <0> <6>

Note: This command loads a 16 bit value into a specified interpreter position

Data Explanation:

lo_linenum = LSB of the line number to load

hi_linenum = MSB of the line number to load

lo_data = LSB of the data to load

hi_data = MSB of the data to load

index= specific position in "line number" to load

csum = the 8-bit sum of 200+ 53+ lo_linenum+hi_linenum+lo_data+hi_data+index

Start Interpreter

->Send 8 binary bytes: <200> <54> <0> <0> <0> <0> <0> <254>

<-Receive 3 byte header: <6> <0> <6>

Note: This command causes the internal command interpreter to begin execution

Set CPUXA Real Time Clock

->Send 50 binary bytes: <241> <61> <n2> <n3>..... <n47> <n48> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command loads and starts the CPU-XA Real Time Clock.

n2 = 61; //set RTC
n3 = seconds ones digit 0-9 (23:59:3x)
n4 = seconds tens digit 0-5 (23:59:x4)
n5 = minutes ones digit 0-9 (23:5x:34)
n6 = minutes tens digit 0-6 (23:x9:34)
n7 = hours ones digit 0-3 (2x:59:34)
n8 = hours tens digit 0-2 (x3:59:34)
n9 = day ones digit 0-9 (12/0x/99)
n10 = day tens digit 0-3 (12/x4/99)
n11 = month ones digit 0-9 (1x/04/99)
n12 = month tens digit 0-1 (x2/04/99)
n13 = year ones digit 0-9 (12/04/9x)
n14 = year tens digit 0-9 (12/04/x9)
n15 = day of the week 0-6 (Sunday-Saturday)

csum = the 8 bit sum of 241 + 61 + n2 + n3 + n4...n47 + n48

Get CPUXA Real Time Clock

->Send 8 binary bytes: <200> <62> <0> <0> <0> <0> <0> <6>

<-Receive 3 byte header: <6> <0> <6>

<- Receive year <n>

<- Receive month <n>

<- Receive month day <n>

<- Receive hour <n>

<- Receive minute <n>

<- Receive weekday <n>

Write CPUXA Variable Data

->Send 8 binary bytes: <200> <41> <vnum> <LSB data> <MSB data> <0> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command writes data to vnum.

Data Explanation:

vnum = the variable number to write to (0-63)
0 - 60 Regular program variables
61 Leopard Beep Counter
Leopard will beep x times then stop
62 time conversion variable
63 Leopard Screen number
(set to 0 for user loaded Bitmap)
(set to 1-24 for C-Max designed screens)
LSB data = the low byte of the 16 bit data
MSB data = the high byte of the 16 bit data
csum = the 8-bit sum of 200 + 41+ vnum + LSB + MSB

CPU-XA Notes:

- 1) **The CPU-XA interface cable to the TW-523 is a special cable** and should not be substituted with a normal telephone extension cord. The correct cable pinout is listed below:

CPU-XA		TW-523
1	----->	1
2	----->	2
3	----->	3
4	----->	4

- 2) The following cable pinouts are for interfacing the CPU-XA to a Macintosh computer:

Macintosh (Din 8 Male)			CPU-XA (DB-9 Male)	
Signal	Pin		Pin	Signal
HSK o	1	_____	4	DTR
		_____	7	RTS
HSK i	2	_____	8	CTS
TxD -	3	_____	3	Tx
Gnd	4	_____	5	Gnd
RxD +	8	_____		
RxD -	5	_____	2	Rx
TxD +	6	_____	9	
GPI	7	_____	1	DCD
Shield		_____		

- 3) To obtain a linear value from the SECU-16 and SECU-16I analog inputs, you must remove a resistor (WHICH ONE?) from the circuit board. Removing the resistor provides a linear output, but also disables the supervisory capability for digital input.

Send Remote IR Command

->Send 8 binary bytes: <200> <92> <mnum> <znum> <IRlo> <IRhi> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command sends previously learned IR command Irhi-IRlo to module mnum, zone znum

Data Explanation:

mnum = the SECU16IR address 1-128

znum = the zone on the specific SECU16IR 0-15 (znum = 255 for ALL zones)

IRlo = LSB of the ir number to send

IRhi = MSB of the ir number to send

csum = the 8-bit sum of 200 + 92+ mnum + znum + IRlo + IRhi + 0

Send Leviton X-10 Preset Dim

->Send 8 binary bytes: <200> <97> <house> <key> <dim> <49> <0> <csum>

<-Receive 3 byte header: <6> <0> <6>

Note: This command instructs the CPU-XA to add the X-10 house, key and dim codes to it's send buffer

Data Explanation:

house = the X-10 house code to send (see Below)

key = the X-10 key code to send (see Below)

dim =the light level 0 to 63 (0 = OFF, 63 = FULL ON)

csum = the 8-bit sum of 200 + 97+ house + key + dim + 49 + 0