EZCronGuideFile

COLLABORATORS			
	TITLE : EZCronGuideFile		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 23, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

EZCronGuideFile

1.1 X10 CM11A Protocol

X10 CM11A Protocol
AmigaGuide Conversion by
Jim Hines

1.2 transmission_coding

1. X-10 Transmission Coding (overview).

1.1 Housecodes and Device Codes.

The housecodes and device codes range from A to P and 1 to 16 respectively although they do not follow a binary sequence. The encoding format for these codes is as follows

Housecode	Device Code	Binary Value
A	1	0110
B	2	1110
C	3	0010
D	4	1010
E	5	0001
F	6	1001
G	7	0101
H	8	1101
I	9	0111
J	10	1111
K	11	0011
L	12	1011
M	13	0000
N	14	1000
O	15	0100
P	16	1100

1.2 Function Codes.

Function	Binary Value
All Units Off	0000
All Lights On	0001
On	0010
Off	0011
Dim	0100
Bright	0101
All Lights Off	0110
Extended Code	0111
Hail Request	1000
Hail Acknowledge	1001
Pre-set Dim (1)	1010
Pre-set Dim (2)	1011
Extended Data Transfer	1100
Status On	1101
Status Off	1110
Status Request	1111

1.3 serial

2. Serial Parameters.

The serial parameters for communications between the interface and PC are as follows:

Baud Rate:	4,800bps
Parity:	None
Data Bits:	8
Stop Bits:	1

Cable connections:

```
    Signal  DB9 Connector   RJ11 Connector
    ======  ============    ==============

    SIN         Pin 2           Pin 1
    SOUT        Pin 3           Pin 3
    GND         Pin 5           Pin 4
    RI          Pin 9           Pin 2
```

where:

```
    SIN     Serial input to PC (output from the interface)
    SOUT    Serial output from PC (input to the interface)
    GND     Signal ground
    RI      Ring signal (input to PC)
```

1.4 transmission

3. X-10 Transmission.

3.1. Standard Transmission.

An X-10 transmission from the PC to the interface typically refers to the communication of a Housecode and Device Code combination or the transmission of a function code. The format of these transmissions is:

```
                PC          Interface
                ==          =========

2 bytes     Header:Code
1 byte                      checksum
1 byte      Acknowledge
1 byte                      interface ready to receive
```

This format is typical of all transmissions between the PC and the interface with the difference being in the first transmission from the PC.

3.1.1. Header:Code.

The Header:Code combination is configured thus:

```
    Header: 7   6   5   4   3   2   1   0
            < Number of Dims>   1  F/A E/S
```

Where:

Number of Dims is a value between 0 and 22 identifying the number of dims to be transmitted (22 is equivalent to 100%)

Bit 2 is always set to '1' to ensure that the interface is able to maintain synchronization.

F/A defines whether the following byte is a function (1) or address (0).

E/S defines whether the following byte is an extended transmission (1) or a standard transmission (0).

```
Code:    7   6   5   4   3   2   1   0
Address: < Housecode >   <Device Code>
Function:< Housecode >   < Function  >
```

Note the function only operates for devices addressed with the same Housecode.

3.1.2. Interface Checksum and PC Acknowledge

When the interface receives a transmission from the PC, it will sum all of the bytes, and then return a byte checksum. If the checksum is correct, the PC should return a value of 0x00 to indicate that the transmission should take place. If however, the checksum is incorrect, then the PC should again attempt to transmit the Header:Code combination and await a new checksum.

3.1.3. Interface Ready to Receive.

Once the X-10 transmission has taken place (and this may be quite time consuming in the case of Dim or Bright commands) the interface will send 0x55 to the PC to indicate that it is in a 'ready' state.

3.1.4. Example.

PC	Interface	Description
0x04,0x66		Address A1
	0x6a	Checksum ((0x04 + 0x66)&0xff)
0x00		OK for transmission.
	0x55	Interface ready.
0x04,0x6e		Address A2
	0x72	Checksum ((0x04 + 0x6e)&0xff)
0x00		OK for transmission.
	0x55	Interface ready.
0x86,0x64		Function: A Dim 16/22*100%
	0xe0	Incorrect checksum.
0x86,0x64		Function re-transmission
	0xea	Checksum ((0x86 + 0x64)&0xff)
0x00		OK for transmission.
	0x55	Interface ready.

This transmission will address lamp modules A1 and A2, and then dim them by 72%. Note multiple addresses cannot be made across housecodes, i.e. A1, B2 Dim 72% is not valid, and would result in B2 being dimmed by 72%.

3.2. Extended X-10 Transmission.

Extended X-10 transmission is simply an extension of the protocol to allow two additional bytes of extended data to be transmitted. In this case, the protocol may be shown as:

```
            PC              Interface
            ==              =========
4 bytes     Header:Code:Data:Command
1 byte                      checksum
1 byte      Acknowledge
1 byte                      interface ready to receive
```

The header for an extended transmission is always:

```
    Header:     7   6   5   4   3   2   1   0
                0   0   0   0   0   1   1   1
```

Bits 7 to 3 are always zero because the dim level is not applicable to extended transmissions. Bit 2 must be set to '1' as in all PC header transmissions. Bit 1 is set to '1' as the extended transmission is always a function. Bit 0 is set to '1' to define an extended transmission rather than a standard transmission.

The code byte is:

```
    Code:      7   6   5   4   3   2   1   0
    Function:  < Housecode >   0   1   1   1
```

Again, the housecode must be the same as any previously addressed modules, and for extended data, the function code must be 0111.

Finally, the data and command bytes may take any value between 0x00 and 0xff. Note that the checksum is one byte and is defined as:

```
checksum = (header + code + data + command)&0xff
```

1.5 reception

4. X-10 Reception.

Whenever the interface begins to receive data from the power-line, it will immediately assert the serial ring (RI) signal to initiate the wake-up procedure for the PC. Once the data reception is complete, the interface will begin to poll the PC to upload its data buffer (maximum 10 bytes). If the PC does not respond, then the interface's data buffer will overrun, and additional data will not be stored within the buffer.

4.1. Interface Poll Signal.

In order to poll the PC, the interface will continually send:

```
    Poll:      7   6   5   4   3   2   1   0
    Value:     0   1   0   1   1   0   1   0        (0x5a)
```

This signal will be repeated once every second until the PC responds.

4.2. PC Response to the Poll Signal.

To terminate the interface's polling and initiate the data transfer, the PC

must send an acknowledgment to the interface's poll signal. This acknowledgment is:

Poll:	7	6	5	4	3	2	1	0	
Value:	1	1	0	0	0	0	1	1	(0xc3)

Notice that bit #2 of the PC transmission is not set, indicating that this cannot be the beginning of a transmission from the PC.

4.3. Interface Serial Data Buffer.

The buffer consists of 10 bytes defined as follows:

Byte	Function
0	Upload Buffer Size
1	Function / Address Mask
2	Data Byte #0
3	Data Byte #1
4	Data Byte #2
5	Data Byte #3
6	Data Byte #4
7	Data Byte #5
8	Data Byte #6
9	Data Byte #7

The interface will only upload the specified number of bytes within the buffer, and will not default to uploading 10 bytes in every transmission. The number of bytes to receive is thus specified in byte 0 of the transmission.

The function address mask indicates whether the following 8 bytes should be interpreted as an address or as a function. The position of the bit in the mask corresponds to the byte index within the data buffer. If the bit is set (1), the data byte is defined as a function, and if reset (0), the byte is an address.

The data bytes are in the same format as for the Code byte in the X-10 transmissions (i.e. Housecode:Device Code or Housecode:Function).

Note that once the data buffer has been uploaded, there is no acknowledgment from the PC to the interface as the contents of the serial data buffer will have been changed. This will not cause a problem as this is simply informing the PC of the external status, rather than controlling a device (as in the case of the PC transmission) which may have safety implications.

4.4. Dim or Bright.

After a dim or bright code, the PC will expect the following byte to be the change in brightness level. An X-10 module has 210 discrete brightness levels, and therefore this byte will be equivalent to a brightness change of n/210*100%.

4.5. Extended Code.

Extended code is processed in a similar way to Dim and Bright, except that the PC will expect two bytes, which are the Data and Command bytes.

4.6. Example.

PC	Interface	Description
	0x5a	Poll from interface.
0xc3		'PC Ready' Response from PC
	0x05	5 bytes to follow
	0x04	xxxx x100-> byte 0,1 addresses, 2 function
	0xe9	B6
	0xe5	B7
	0xe5	B Bright
	0x58	0x58/210 * 100%

This transmission will wake the computer, and then indicate that a transmission of length 5 bytes will occur, data bytes 0 and 1 are addresses and byte 2 is a bright function, which means that the following byte is the change in brightness level.

1.6 fast_macro_download

5. Fast Macro Download.

The interface contains a 42 byte buffer which contains macro codes. These macro codes are initiated upon the reception of a pre-defined address (i.e. B7), and the code specifies the transmissions that the interface should then make. Due to the shortage of bytes, the macro code is 'compressed' by grouping similar functions.

Note, any error in the function codes may result in the interface entering an endless loop and becoming 'locked-up', so steps should be taken to ensure that the code is correct prior to transmission.

If the interface detects that it has suffered a power-down situation, it will ring the PC and poll with a specialized code to indicate that the macros must be refreshed.

5.1. Power-fail Macro Download Poll Code.

> NOTE: I beleive that this is mainly for the CP10. The battery backed CM11 does send this poll after a power failure, but it responds to a setclock directive rather than the macro download. It waits till the resumption of power before it starts sending this byte. DBS, Jan 1, 1997

In order to poll the PC, the interface will continually send:

Poll:	7	6	5	4	3	2	1	0	
Value:	1	0	1	0	0	1	0	1	(0xa5)

This signal will be repeated once every second until the PC responds.

5.2. PC Response to Macro Download Poll Code.

To stop the polling, the PC must respond with:

PC Response:	7	6	5	4	3	2	1	0	
Value :	1	1	1	1	1	0	1	1	(0xfb)

Once this has been transmitted, the macro must be immediately downloaded. At this stage, the interface will wait until the 42 byte macro has been received before any X-10 transmissions can occur.

5.3. Macro Code (CM10).

Macro code is divided into individual macros, and functional groups within the macros. The only limit to the number of macros and groups is the number of available storage bytes.

Each macro begins with an initiator byte which details the Housecode and Device code that will cause the macro to start.

Following the initiator byte is the length of this current macro, and the functional trigger (ie On or Off functions). The length is defined by the lower 7 bits, and the functional trigger by the most significant bit. If the most significant bit is set, the functional trigger is 'On', and if reset, the functional trigger is 'Off'.

As mentioned previously, the macro is divided into functional groups, and each group has a byte indicating the length of the group before the macro is defined. This group length byte is exclusive of the function code.

The group is then made up of a common housecode (1 nibble), followed by a number of device codes (each takes 1 nibble) and finally a function code (1 nibble). If the function code falls on a byte boundary, then it is always the low nibble of the byte.

All unused bytes must take a value of 0x00.

5.3.1. Dimming and Brightening within a macro.

If the function is a bright or dim, then the next byte specifies the change in brightness level in 22 steps. Note if the most significant bit of this byte is set, the interface will send out enough bright commands to ensure that the associated lamps are at 100%, and then dim the lamp by the specified value.

5.3.2. Extended codes in macros.

Extended code cannot be grouped as for other functions, and consequently an extended code group would be defined as:

Byte	Description
0x01	Group length
0xa7	Housecode D (1010 = D), Extended code function
0x03	Device code 11 (0011 = 11)

```
    0xff    Data byte: 0xff
    0x55    Command byte: 0x55
```

5.3.3. Checksum.

Once the macro has been downloaded, the interface calculates the 1 byte checksum by summing all 42 bytes of the macro code (not including the PC macro download start byte) and returns the appropriate value. If the value is incorrect, the PC should again initiate the macro download by transmitting the PC macro download start byte.

5.3.4. Example.

```
    PC      Interface   Description
    ==      =========   ===========

            0xa5        Power-fail, macro poll.
    0xfb                Macro download start byte

    0x26                Initiator C1
    0x0a                Functional Trigger: 'Off'; Macro length: 10 bytes
    0x04                Group length: 4 nibbles
    0x66                Macro housecode, A, device 1
    0x2e                Devices 2 and 3
    0x04                Dim
    0x0b                Dim by 11/22*100% = 50%
    0x02                group length: 2 nibbles
    0x6a                Macro housecode, A, device 4
    0x02                Function: On

    0x26                Initiator C1
    0x8c                Functional Trigger: 'On'; Macro length: 12 bytes
    0x02                Group length: 2 nibbles
    0x66                Macro housecode, A, device 1
    0x02                Function: On
    0x03                Group length: 3 nibbles
    0x6e                Macro housecode, A, device 2
    0x42                Device 3, Function Dim (0100)
    0x06                Dim by 6/22*100% = 27%
    0x02                Group length: 2 nibbles
    0x6a                Macro housecode, A, device 4
    0x03                Function: Off

    0x00...             Remaining 20 bytes set to 0x00
            0x91        Macro checksum: 0x91
    0x00                Checksum correct
            0x55        Interface ready
```

5.4. EEPROM Code (CM11 and CP10).

The EEPROM code for the CM11 and CP10 contains both the downloaded timers and also the macro data. The timers are resolved into 'pseudo-macros' with the only difference being in the initiator (ie a timer as opposed to a macro code).

The EEPROM may be broken down into four categories:

Macro Offset
Timer Initiators
Macro Initiators
Macro Data.

5.4.1. Macro Offset.

The first two bytes of the EEPROM contain an offset to the macro initiator table. The macro initiator table is offset rather than the timers as the timers must be processed every minute, whereas the macros are only processed whenever an X-10 transmission event is detected.

5.4.2. Timer Initiator.

The timers reside from address 0x0002 in the EEPROM and are delimited by a 0xff at the end of the table. Each timer entry contains the following data:

Bit range	Description
71	Reserved
70 to 64	Day mask (bit 1 = Sunday, bit 7 = Saturday)
63 to 56	Start day range (bits 0 to 7)
55 to 48	Stop day range (bits 0 to 7)
47 to 44	Event start time x 120 minutes
43 to 40	Event stop time x 120 minutes
39	Start day range (bit 8)
38 to 32	Event start time (0 to 120 minutes, bits 0 to 6)
31	Stop day range (bit 8)
30 to 24	Event stop time (0 to 120 minutes, bits 0 to 6)
23 to 20	Start event macro pointer (bits 8 to 11)
19 to 16	Stop event macro pointer (bits 8 to 11)
15 to 8	Start event macro pointer (bits 0 to 7)
7 to 0	Stop event macro pointer (bits 0 to 7)

5.4.3. Macro Initiator.

The macro initiators are configured thus:

Bit range	Description
23 to 20	Initiator house code
19 to 16	Initiator device code
15	Initiator function (’1’ = on, ’0’ = off)
14 to 12	Reserved
11 to 0	Macro pointer (bits 0 to 11)

5.4.4. Macro data.

Macro data starts with a timer offset (0 for instant to 240 for 4 hours). Following the timer offset is the number of elements within the macro (1 to 255). This is followed by the macro elements themselves:

The macro elements are configured as follows:

Basic command:

Bit range	Description
23 to 20	Command house code
19 to 16	Command function
15 to 0	X10 format device bitmap

Bright or dim commands:

Bit range	Description
31 to 28	Command house code
27 to 24	Command function
23 to 8	X10 format device bitmap
7	Brighten first ('1') or simply dim ('0')
6 to 5	Reserved
4 to 0	Dim value (ranging from 0 to 22)

Extended data commands:

Bit range	Description
47 to 44	Command house code
43 to 40	Command function
39 to 24	X10 format device bitmap
23 to 0	Extended code data

Note: The timer offsets are relative to the previous timer value.

5.4.5. EEPROM Data Transfer.

The EEPROM is downloaded to the interface in blocks of 19 bytes. The first byte is the macro download initiator command byte (0xfb), followed by two bytes containing the actual EEPROM address (this does not need to be sequential, although it must not cross the 16 bit page boundary). 16 bytes of EEPROM data follows the EEPROM address.

Once the interface has received the EEPROM data, it will return a checksum. If the checksum is correct, the PC will acknowledge (0x00) and after the data has been programmed into the EEPROM, the interface will return a 'ready' command (0x55) to indicate that it is available to process PC requests.

5.4.6. Example.

PC	Interface	Description
0xfb		EEPROM download start byte (first block of data)
0x00		EEPROM address 0x0000 (lo byte)
0x00		(high byte)

```
0x00
0x0c                EEPROM offset to macro initiators 0x000c

0x3e                Day mask x 0111110 (.FTWTM.)
0x00                Start day [0..7]
0x6d                Stop day [0..7]
0x49                (Event start time, Event stop time) x 120 minutes
0x00                Start day range [8], Event start time [0..6]
0x80                Stop day range [8], Event stop time [0..6]
0x00                Start macro pointer [8..11], Stop macro pointer [8..11]
0x1d                Start macro pointer [0..7]
0x22                Stop macro pointer [0..7]

                    Summary:    Start day: 0x000    (Jan 1)
                            Stop day: 0x16d (Dec 31)
                            Start time: 4 x 120mins = 08:00
                            Stop time: 9 x 120mins = 18:00
                            Start macro pointer: 0x01d
                            Stop macro pointer: 0x022

0xff                Timer table delimiter

0x6a                Macro initiator house and device code (A4)
0x80                Macro function (On)
0x11                Macro pointer (0x011)

0xff

        0xb8        Checksum from the interface

0x00                Checksum correct

        0x55        Programming complete

0xfb                Second block of data

0x00
0x10                EEPROM start address

0xff                Macro table delimiter

0x00                Macro: instant
0x01                1 element
0x64                House code A, function Dim
0x00
0x40                Bitmap: device #1
0x0b                Dim level 11/22 = 50%

0x0f                Macro: delayed by 15 minutes
0x01                1 element
0x64                House code A, function Dim
0x00
0x40                Bitmap: device #1
0x80                Brighten to 100%

0x00                Macro: instant
```

```
    0x01                1 element
    0x62                House code A, function On

            0x56        Checksum from the interface

    0x00                Checksum correct

            0x55        Programming complete

    0xfb                Third block of data

    0x00
    0x20                EEPROM start address

    0x00
    0x04                Bitmap: device #3

    0x00                Macro: instant
    0x01                1 element
    0x63                House code A, function Off
    0x00
    0x04                Bitmap: device #3

    0x00                Zero pad for remainder of the data stream
    0x00
    0x00
    0x00
    0x00
    0x00
    0x00
    0x00
    0x00

            0x8c        Checksum from the interface
    0x00                Checksum correct

            0x55        Programming complete
```

1.7 serial_ring_disable

```
6.   Serial Ring Disable

If may be required, for the sake of 'trouble-shooting' to disable the
serial ring (RI) signal, although undesirable as macros held within the
computer will not operate, nor will the computer be able to track the
system status.

The following protocol will allow the serial ring (RI) signal to be enabled
and disabled:

Enable Ring:

    PC      Interface   Description
    ==      =========   ===========
```

```
    0xeb                Enable the ring signal
            0xeb        Checksum
    0x00                Checksum correct
            0x55        Interface ready
```

Disable Ring:

```
    PC      Interface   Description
    ==      =========   ===========

    0xdb                Disable the ring signal
            0xdb        Checksum
    0x00                Checksum correct
            0x55        Interface ready
```

The default state of the serial ring (RI) signal after a power on reset is enabled.

1.8 eeprom_address

7. EEPROM Address.

This command is purely intended for the CM11 and CP10.

When the interface receives a fast macro initiator, or when a timer event is processed, it will immediately perform an asynchronous transmission of the EEPROM address that is subsequently processed.

The command is of the form:

```
    0x5b        EEPROM address transmission
    0xhh        High byte of macro EEPROM address
    0xll        Low byte of macro EEPROM address
```

This transmission is a one time transmission, and requires no hand-shaking as the interface may not be connected to the PC.

1.9 set_clock

8. Set Interface Clock.

This command is purely intended for the CM11 and CP10.

The PC can set the interface clock with an unsolicited transmission at any time. In addition, once the interface detects the absence of power, it will request the current time from the PC when the PC is available as follows:

CM11:

For a CM11, the time request is from the interface is: 0xa5.

The PC must then respond with the following transmission

    Note: The bit range is backwards from what you'd expect in serial
    communications. Bit 55-48 is actually the first byte transmitted,
    etc. To make matters worse, the bit orientation is correct within
    the bit range, IE bits 4-7 of byte 6 _IS_ the monitored house code.
    Further, bits 0 and 1 of byte 6 appear to be flipped. I get a
    "monitor status clear" if bit 0 is set.
    The original docs had bit 23 as part of current hours AND day.
    DBS Jan 1, 1997

Bit range	Description	
55 to 48	timer download header (0x9b)	(byte 0)
47 to 40	Current time (seconds)	(byte 1)
39 to 32	Current time (minutes ranging from 0 to 119)	(byte 2)
31 to 24	Current time (hours/2, ranging from 0 to 11)	(byte 3)
23 to 16	Current year day (bits 0 to 7)	(byte 4+.1)
15	Current year day (bit 8)	(byte 5-.1)
14 to 8	Day mask (SMTWTFS)	(6...)
7 to 4	Monitored house code	
3	Reserved	
2	Battery timer clear flag	
1	Monitored status clear flag	
0	Timer purge flag	

CP10:

For a CP10, the time request is from the interface is: 0xa6.

The PC must then respond with the following transmission

Bit range	Description
63 to 56	Timer download header (0x7b)
55 to 48	Current time (seconds)
47 to 40	Current time (minutes ranging from 0 to 119)
39 to 32	Current time (hours/2, ranging from 0 to 11)
31 to 24	Current year day (bits 0 to 7)
23	Current year day (bit 8)
22 to 16	Day mask (SMTWTFS)
15 to 12	Monitored house code
11	Reserved
10	Battery timer clear flag
9	Monitored status clear flag
8	Timer purge flag
7 to 4	Power strip house code
3 to 0	Power strip device code

1.10 status_request

9. Status Request.

This command is purely intended for the CM11 and CP10.

The PC can request the current status from the interface at any time as follows:

CM11:

For a CM11, the status request is: 0x8b.

The status request is immediately followed by:

Note: This is really interesting. The btye order is reversed per the note in section 8. The last 3 bytes are each mapped to show a 1 in the bit position if the unit with value equating to the nibble (section 1) is set. Low byte comes first, hi byte second.
Example: if unit 1 is on, the nibble = 6, so the mask would show 00...0100000
Note also that the hi bit of byte 6 must be multiplied by 256 and added to the decimal value of byte 5 (+1) to find the Julian date.
DBS Jan 1, 1997

Bit range	Description	
111 to 96	Battery timer (set to 0xffff on reset)	(byte 0-1)
95 to 88	Current time (seconds)	(byte 2)
87 to 80	Current time (minutes ranging from 0 to 119)	(bute 3)
79 to 72	Current time (hours/2, ranging from 0 to 11)	(byte 4)
71 to 63	Current year day (MSB bit 63)	(byte 5+)
62 to 56	Day mask (SMTWTFS)	(byte 6-)
55 to 52	Monitored house code	(bute 7 lo)
51 to 48	Firmware revision level 0 to 15	(byte 7 hi)
47 to 32	Currently addressed monitored devices	(bute 8-9)
31 to 16	On / Off status of the monitored devices	(byte 10-11)
15 to 0	Dim status of the monitored devices	(bute 12-13)

CP10:

For a CP10, the status request is: 0x6b.

The status request is immediately followed by:

Bit range	Description
119 to 104	Battery timer (set to 0xffff on reset)
103 to 96	Current time (seconds)
95 to 88	Current time (minutes ranging from 0 to 119)
87 to 80	Current time (hours/2, ranging from 0 to 11)
79 to 71	Current year day
70 to 64	Day mask (SMTWTFS)
63 to 60	Monitored house code
59 to 56	Firmware revision level 0 to 15
55 to 48	Power strip house and device code

```
47 to 32    Currently addressed monitored devices
31 to 16    On / Off status of the monitored devices
15 to 0     Dim status of the monitored devices
```

1.11 power-up_timer

```
10.  Power-up Timer.

This command is purely intended for the CP10.

The interface contains a power-up timer that will turn on the remote
controlled sockets once it elapses on the assumption that the computer has
failed to boot-up.  If it receives a message ('Relay Open' or 'Relay
Close', see item 7) from the computer before the timer elapses, then the
time-out is canceled and the sockets configured in accordance with the
message.

The power-up timer is the fifth byte of the six byte transmission for the
scheduled ring, and it is split into two nibbles.  The upper nibble is a
reload value and the lower nibble is the actual timer.  Each timer tick is
2 seconds, so the maximum timer value is 30 seconds.

10.1.    Transmission Protocol

The PC can define the delay after which the power strip will turn the
controllable outlets on and off after detecting the PC turning on and off.

Bit range   Description
=========   ===========

55 to 48    Power-up timer download header (0xcb)
47 to 40    Reserved (0x00)
39 to 32    Reserved (0x00)
31 to 24    Reserved (0x00)
23 to 16    Reserved (0x00)
15 to 12    Power-up time-out (multiples of 2 seconds, range = 0 to 30s)
11 to 8     Reserved (0x0)
7 to 4      Power-down time-out (multiples of 2 seconds, range = 0 to 30s)
3 to 0      Reserved (0x0)

The interface will respond with a checksum excluding the header.  If
correct the PC should respond with 0x00, or download the correct value
again.  The interface will terminate the transfer with 0x55 indicating that
it is ready to communicate with the PC.

11.  Relay Control.

This command is purely intended for the CP10.

The power-strip contains a relay that controls four extension sockets.
These sockets are controllable via the PC with the following commands:

Close Relay (sockets on):

    PC      Interface   Description
```

```
    ==      =========   ===========

    0xab                Close the relay
            0xab        Checksum
    0x00                Checksum correct
            0x55        Interface ready

Open Relay (sockets off):

    PC      Interface   Description
    ==      =========   ===========

    0xbb                Open the relay
            0xbb        Checksum
    0x00                Checksum correct
            0x55        Interface ready
```

1.12 relay_control

```
11.  Relay Control.

This command is purely intended for the CP10.

The power-strip contains a relay that controls four extension sockets.
These sockets are controllable via the PC with the following commands:

Close Relay (sockets on):

    PC      Interface   Description
    ==      =========   ===========

    0xab                Close the relay
            0xab        Checksum
    0x00                Checksum correct
            0x55        Interface ready

Open Relay (sockets off):

    PC      Interface   Description
    ==      =========   ===========

    0xbb                Open the relay
            0xbb        Checksum
    0x00                Checksum correct
            0x55        Interface ready
```

1.13 input_filter_fail

```
12.  Input Filter Fail.

This command is purely intended for the CP10.
```

The power-strip contains an input filter and electrical surge protection that is monitored by the microcontroller. If this protection should become compromised (i.e. resulting from a lightening strike) the interface will attempt to wake the computer with a 'filter-fail poll'.

This poll signal takes the form:

Poll:	7	6	5	4	3	2	1	0	
Value:	1	1	1	1	0	0	1	1	(0xf3)

The poll signal will be repeated to the PC every second until the PC responds with the default poll response:

PC Response:	7	6	5	4	3	2	1	0	
Value:	1	1	1	1	0	0	1	1	(0xf3)