AXE 4844 Protocol

Alltrax AXE/DCX serial communications protocol.
The following notes are excerpts from the source code for the controller.

The controller is a slave, it only speaks if spoken to, and only then if it understands the command, checksum is correct and so on. You'll have to poll the controllers RAM for data. You can write memory locations in flash and eeprom, but all it will do is cause us both headaches. I've identified the commands in this protocol so you don't have to tinker around trying to find them, but do us both a favour and leave flash/eeprom alone. Call Alltrax for support.

Serial protocol is RS-232, 9600 baud, 8 data bits, no parity, 1 stop bit, no handshaking. Protocol only uses a source/target I.D. and checksum to validate incoming information, it wasn't meant to live on a network. The controller understands 6 commands, as detailed below, but really all you should be doing is reading RAM. The target processor in the AXE line is a PIC16F873, in the DCX it is a PIC16F876.

AXE communications Spec. All references relative to motor controller.

Communications Packet Protocol

Receive Packet Format

Name	***Size***	***Definition***
source/target	1 byte	Windows App is 5h, controller is Bh
command	1 byte	command function to controller
data1	1 byte	
data2	1 byte	
data3	1 byte	
data4	1 byte	
checksum	1 byte	OVERFLOWED SUM OF PREVIOUS SIX BYTES

Command byte definition;

This is the instruction set for communicating with the
controller.
00h = read byte of program (flash) memory
01h = write byte of program (flash) memory
02h = read byte of EEPROM memory
03h = write byte of EEPROM
04h = read 3 sequential bytes of RAM
05h = reset
04h = READ RAM - This command fetches three sequential bytes of information from RAM (variable) memory and sends it to the Windows host. variable locations are organized in order to minimize the number of read cycles needed to refresh data.

Read RAM Packet Format (sent by PC)

source/target	: 5Bh (windows app is host, controller is target)
command	: 04h (read three RAM bytes)
data1	: ADDRESS, in hex.
data2	: DON'T CARE
data3	: DON'T CARE
data4	: DON'T CARE
checksum	: OVERFLOWED SUM OF PREVIOUS SIX BYTES

After decoding the packet, the controller shall assemble the following information and reply to the Windows Host:
Response Format (sent by controller)

source/target	: B5h
command	: 04h
data1	: ADDRESS (start address of three sequential bytes)
data2	: data @ ADDRESS
data3	: data @ ADDRESS + 1
data4	: data @ ADDRESS + 2
checksum	: overflowed sum of 6 previous bytes

Reset Command:
05h = RESET: This command causes the program to goto the START location and begin anew. The complete 7 byte protocol must be sent, but all
data is dummy.

Data Addresses:
The following are RAM locations you can read to get measurement data. High and low bytes are in successive locations, thus you can fetch both high/low byte with one read cycle.

```
THROT_POS    : EQU 0x20 - 8 bit Throttle position, 00h = off,  FFh = wide open
DIODE_TEMP_L : EQU 0x2C
DIODE_TEMP_H : EQU 0x2D - Internal Controller temperature
BATT_VOLT_L : EQU 0x39 - 0.1025 volts per bit.
BATT_VOLT_H : EQU 0x3A
```

```
SHUTDOWN      : EQU 0x3B - Error status, any bit set, NO PWM output.
OUTPUT_CURR_L : EQU 0x60 - IOUT = 1A/BIT, 0 - 1023A
OUTPUT_CURR_H : EQU 0x61 - high byte
```

Battery current may be calculated as a percentage of output current, assuming continuous motor current. Convert THROT_POS to decimal, then
Ibattery = (THROT_POS / 256) * OUTPUT_CURR_H,L

Temperature is measured with a Kelvin sensor, temp = 2.73V @ 0C Thus there is an offset at 0C = 559 counts. Temperature slope = 2.048 counts / deg C (positive tempco.)