AMT203 ABSOLUTE ENCODER
Demo Kit User Guide

PART NUMBER: AMT203 **DESCRIPTION: DEMO KIT USER GUIDE**

AMT-DMBD Demo Board

The AMT-DMBD Demo Board is designed for testing and configuring the AMT 203 Absolute Encoder over the 10 pin encoder connector using the SPI link. The board has a 2 line 16 character LCD display and 3 pushbuttons for use in standalone operation. In addition there is a USB 2.0 interface and drivers are available from WINDOWS UPDATE for XP86, XP64 and Vista (32 and 64 bit).

STARTUP

The USB connector provides power whether or not the driver is installed on the host PC. If the board is first plugged into the USB connector the AMT-DMBD will attempt to connect to an encoder and the LCD will show the current position in hexadecimal format (as shown in Figure 1). The AMT 203 connector is a 10 pin connector that has a polarization arrow on the topside as shown. This connector and the ABS encoder may be plugged in before or after the DEMO BOARD is powered up. The Activity LED (D1) will flash with a 1 second period when the board is powered. This signifies the internal timer interrupt is working and the program is running.

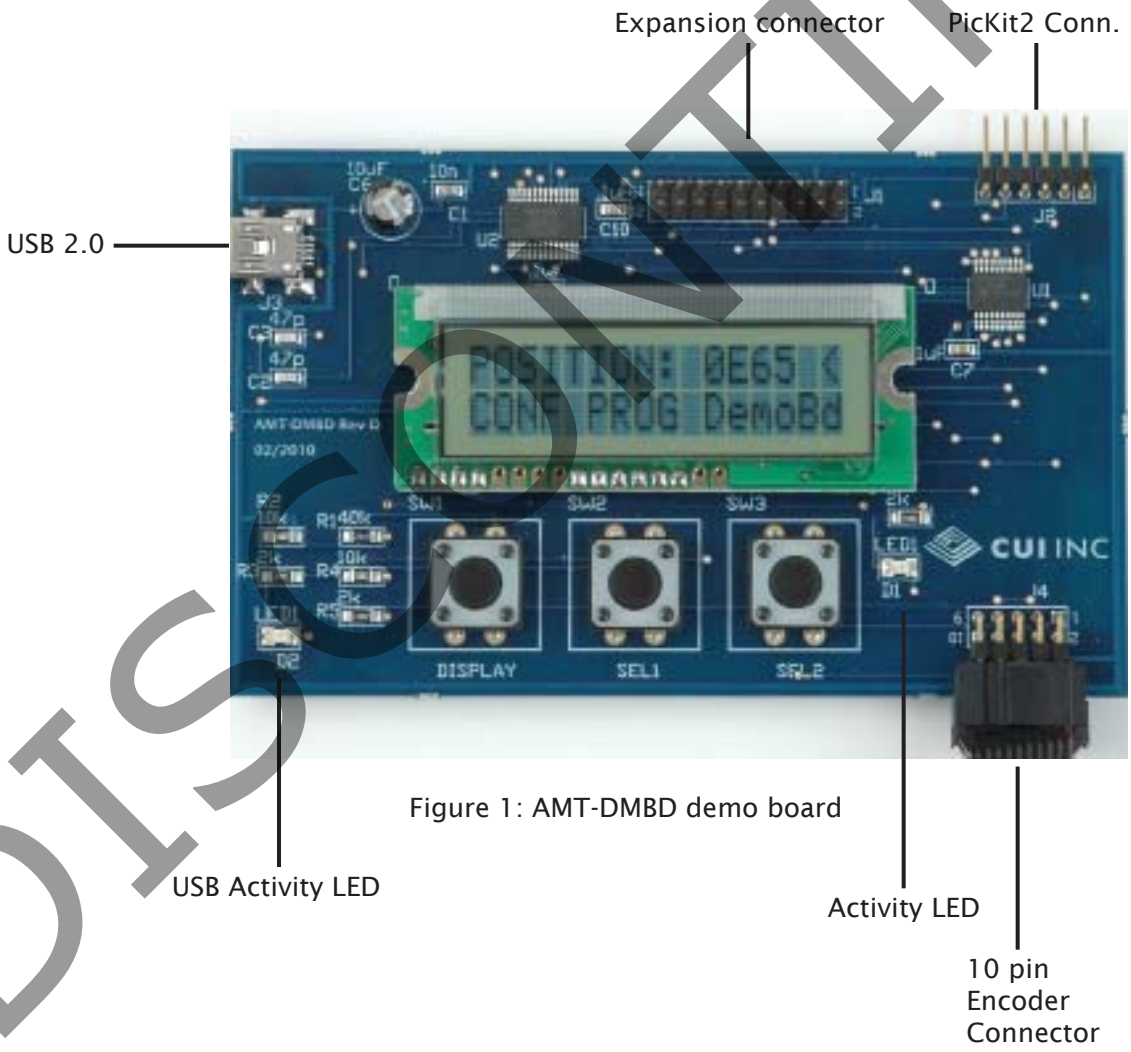


Figure 1: AMT-DMBD demo board

ABS ENCODER MENUS

This section discribes the use of the LCD Display and pushbuttons for the ABS encoder. Most operations can be performed from this interface, however the USB interface is more useful for automatic testing.

NO CONNECT

Figure 2 shows the display when there is no encoder attached. The version is shown as V11. The firmware can be updated using a Microchip PIC programmer such as the PICkit 2.

Figure 2: Startup with no encoder attached

POSITION: DISPLAY AND SELECTIONS

The POSITION: Display is shown in Figure 3. Pressing any of 3 buttons for 1/4 second will result in action. The following actions apply:

- **CONF**: go to the next Display Menu (page 3)
- **PROG**: go to the menu to program the encoder (page 5)
- **DemoBd**: read eeprom settings in AMT-DMDB memory (page 6)

Figure 3: Startup with encoder attached

PART NUMBER: AMT203 **DESCRIPTION: DEMO KIT USER GUIDE**

ENCODER CONFIGURE MENU

Figures 4 and 5 show the Menus that allow for setting absolute position direction, incremental output type and resolution. The buttons function as follows:

- **NEXT**: go to the next Display Menu
- **CCW/CW**: set the encoder direction counter-clockwise or clockwise for absolute position (when viewed from front)
- **QAQB/STB**: QAQB for standard 90 deg. quadrature output or STB for counter output

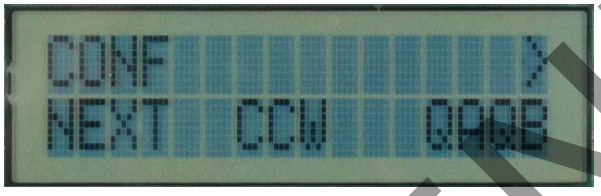


Figure 4: First encoder configure menu

- **NEXT**: go to the next Display Menu
- **B3-B0**: set the encoder resolution according to Resolution Table
- **B4**: internal sampling according to Resolution Table

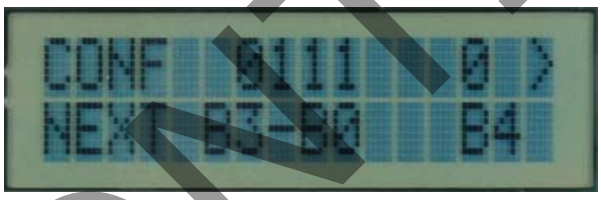


Figure 5: Second encoder configure menu

B3-B0		B4	
Demo Board Setting	Incremental Resolution (PPR)	*Sampling Interval (µs)	
		0 (normal mode)	1 (fast mode)
0000	96	50	13
0100	192	50	25
0001	200	50	13
0010	250	50	13
1000	384	100	50
0101	400	50	25
0110	500	50	25
0011	512	50	13
1001	768	50	100
1001	800	100	50
1010	1000	100	50
0111	1024	50	25

*Normally, bit B4 is set to 0 for stable, smooth tracking. If it is set to 1, the encoder will respond more quickly but not as smoothly.

Resolution Table

ENCODER PROGRAMMING MENU

Figure 6 shows the Menu that allows for setting the absolute zero position or loading configured settings. The buttons function as follows:

- **ZERO**: set current absolute position to zero
- **PROGRAM**: load all configured settings into encoder memory

Figure 6: Zero position and programming menu

Figure 7 displays the screen to confirm zero position has been set or programmed settings have been stored.

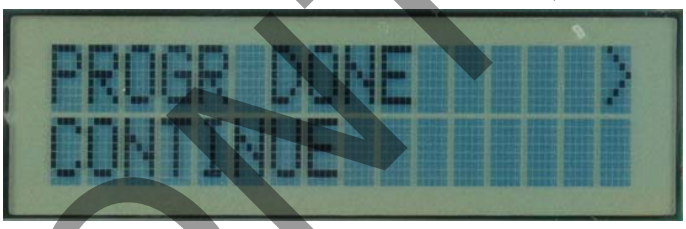


Figure 7: Programming confirmation

After completion of either the zero set or program function, the encoder must be power cycled for changes to be recognized by the encoder. This is done by resetting power to the demo board.

DEMO BOARD SETTING

Figures 8 and 9 show the AMT-DMBD board operational setting menus. The buttons function as follows:

- **NEXT**: go to the next menu
- **HEX/DEC**: select the position display to be HEX or DECIMAL. The encoder always sends binary data but the AMT-DMBD translates it to HEX or DECIMAL for display on the LCD.
- **NORM/FAST**: select the encoder reading method to be normal or FAST. For normal reading the full position is read, for FAST reading only the lower 8 bits are read and the AMT-DMBD board tracks the position. This setting is to allow a logic analyzer to view the read protocol on the SPI lines.

Figure 8: First demo board settings menu

- **EXIT**: return to the main menu
- **NEXT**: returns eeprom XX location value
- **XX**: displayed eeprom value (swith has no functionality)

Figure 9: Second demo board settings menu

AMT-DMBD USB OPERATION

SETUP

The AMT-DMBD board is powered by the USB which will work even when the drivers are not installed. To use the USB port for controlling and reading the ABS encoder, the USB drivers must be installed. When the drivers are installed the USB port behaves as a high speed serial port and is mapped to a com port under Windows.

INSTALLING DRIVERS

When the encoder is plugged into the PC the first time, Windows should attempt to load the driver. The drivers are on Windows update so you can allow Windows to get them from the network.

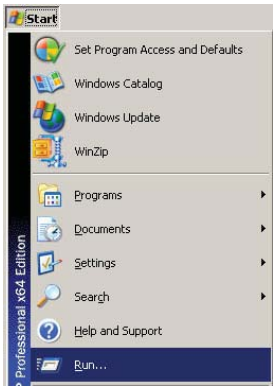
These are FTDI VCP drivers so they can also be downloaded and upgrades will be available from http://www.ftdichip.com/Drivers/VCP.htm

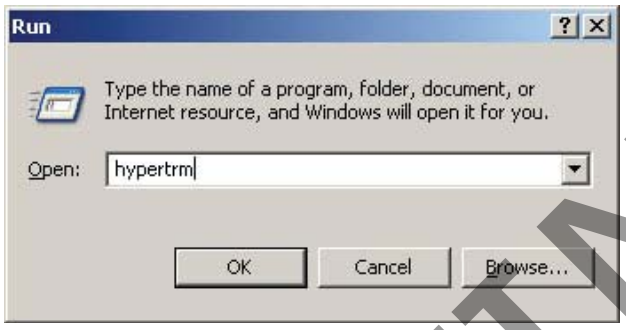
PART NUMBER: AMT203 **DESCRIPTION: DEMO KIT USER GUIDE**

RUNNING HYPERTERM TO TEST THE INTERFACE

Hyperterm is a common terminal emulator found on Windows XP, it is not on later Windows operating systems but there are substitutes or you can use TCL commands as described below.

A. Set up Hyperterm in the properties menu, for example set up a new session called “usbftdi.ht”.

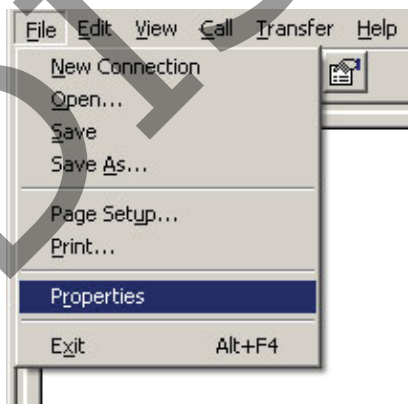


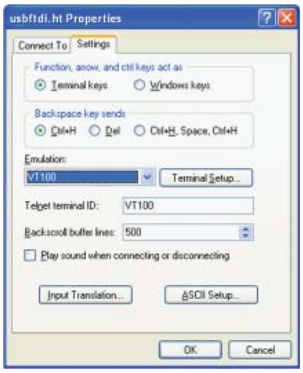


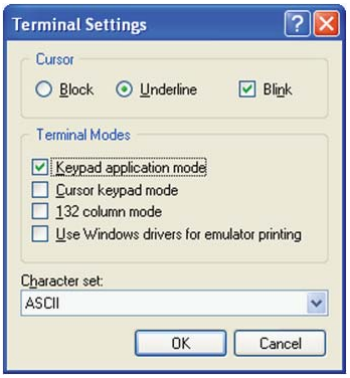
B. ”Connect to” set to the com port that shows up in device manager, for example COM1:.

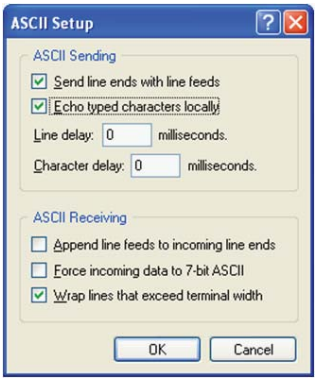
C. In the congifuration submenu, the Bits per second should be 115200m with 8 data bits, parity none, 1 stop bit, flow control none.

D. Select ‘Properties’ from the ‘File’ pull-down menu, click on the ‘Settings’ tab. Select ‘VT100’ in the ‘Emulation’ pull-down menu, click the ‘Terminal Setup…’ dialog button and click the ‘Keypad application mode’ box. Click on the ‘ASCII Setup…’ dialog button, click the boxes marked ‘Send line ends with line feeds’ and ‘Echo typed characters locally’:









TROUBLESHOOTING THE SERIAL CONNECTION

This latter issue where the serial port is in the “used” state is not unique to the AMT-DMBD drivers or the Demo Board, it is a Windows serial port problem, and we have supplied the “comdisable” utility to free up serial ports just in case. If you have that problem, please install the comdisable utility and either try using it with help screens or go online to Microsoft and look up instructions. Basically you have to disable those “used” ports then reboot to free them up.

Another way to disable and reenable the port is to use device manager to unload and reload the com port driver including the USB serial driver. This works for example if you exit Hyperterm abruptly and it will not start again because the port is “unavailable”. Once again this is because the COMM.db is set in the wrong state. Normally Hyperterm expects you to disconnect the port before exiting which will prevent this awkward setting.

RUNNING COMMANDS UNDER HYPERTERM

Once the Hyperterm is up you can type:
p<cr>
and you should see the position information, for example see Figure 6.

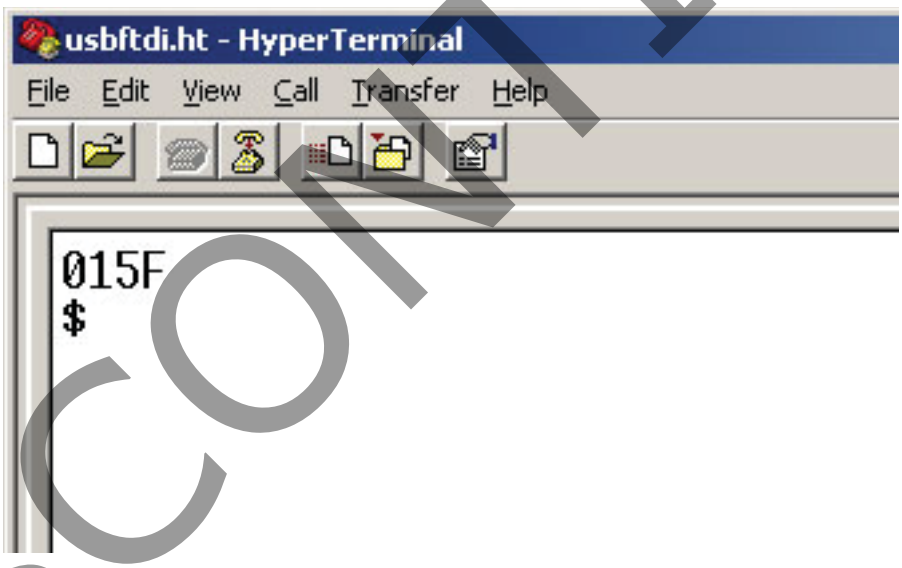


Figure 6: Hyperterminal reading position data

The AMT-DMBD board returns <cr><lf>$ so you see the $ prompt after each command is processed.

The D1 LED turns on when the first character is received by the AMT-DMBD board, and the LCD display shows “USB Active” and the buttons go inactive.

Any AMT-DMBD serial command can be given from the Hyperterminal. However the USB/Serial port is intended for automation and after initial bringup of the serial port the user is encouraged to use [the supplied] TCL procedures or create serial access procedures using Vbasic, C or some other language.

PART NUMBER: AMT203 **DESCRIPTION: DEMO KIT USER GUIDE**

STEPPER CONTROLS IN EXPANSION CONNECTOR

There are two signals used for stepper control: rc2 is used for stepping and rc3 is used for direction. rc2 is pulsed high for 30 microseconds per step. rc3 is initially low but can be set high by the “z” command or set low by the “Z” command.

The GND and VDD signals in the Expansion Connector can be used to power the opto isolated input circuit on a stepper driver.

AMT-DMBD USER COMMANDS

- **p**: read Position.
- **d**: set display radix 10 (decimal)
- **h**: set display radix 16 (hex) This is the default.
- **0**: (zero) set position to zero.
- **y**: set fast read
- **Y**: set no fast read
- **k**: set encoder position increasing CW
- **K**: set encoder position increasing CCW
- **l**: (small L) go to LCD mode (exit the serial control mode)
- **eNN01**: read encoder eeprom location NN (hex 00:FF)
- **uNNHH**: write user data location NN (hex 00:FF) data HH (hex). The user should only modify locations between 00 and 7f hex. Writing to other locations will render the encoder inoperative.
- **x**: step. This produces a 30 microsecond high pulse on rc2.
- **z**: stepper driver direction high. This sets rc3 high.
- **Z**: stepper driver direction low. This sets rc3 low.
- **t**: turn off <lf>$ part of prompt. This command has no reply and it is used to simplify automation, for example when using the TCL procedures to talk to the USB/Serial port.

TCL AUTOMATION SCRIPTS

INSTALLATION

The TCL installer is included on the flash drive, as ActiveTcl8.5...exe This should be run to install the TCL interpreter. After the TCL interpreter is installed, there will be a desktop icon Tclsh8.5, which can be double-clicked to start a TCL shell. The COMM package needs to be added. To do this simply start a TCL shell and type:

% teacup install comm

The teacup utility will go out over the internet, get the COMM package and install it. It will respond with the version number if all is well. The package is permanently installed so it is not necessary to repeat this installation.

EXAMPLE #1: RUNNING GETPOS

⚠ You must exit HyperTerminal (Hyperterm) before running the following command examples. Be sure to use the "Safely Remove Hardware" option in the task bar to disconnect the USB line.

There is a folder on the Installation CD called TCL_ABS_work. Copy this folder to the desktop. Start a TCL shell using the Tclsh85 shortcut and type the following:

% source abs_proc.tcl
% set com [StartCom [last_serial_port]]
% GetPos $com

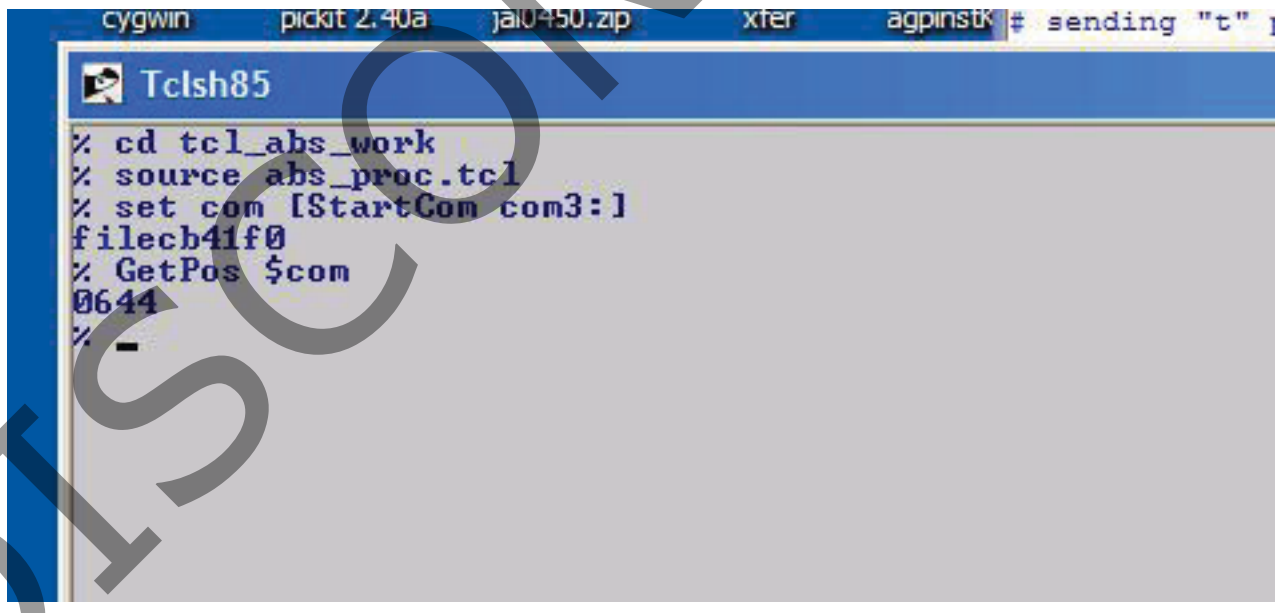


Figure 7: GetPos example

Figure 7 shows the results of running the command sequence above. Note that if the COM port is other than com3 there will be an error message instead of the file name when setting the com variable. We recommend using [last_serial_port] instead of com3.

PART NUMBER: AMT203 **DESCRIPTION: DEMO KIT USER GUIDE**

EXAMPLE #2: RUNNING A STEPPER DRIVER

TCL is very useful for coding small procedures to communicate with the demo board. A simple loop can be written to step and take position measurements. There are only a couple tHings to remember when communicating with the demo board: send the "t" command to turn off the prompt and <lf> characters, and be aware there is a <cr> sent back from the demo board after each command is received so you have to read a variable.

```
# Step takes a number and steps that many increments with delta msec between
proc step {numsteps delta cm}        {
        for {set i 0}    {$1 < $numsteps}     {incr i 1}        {
                puts $cm "x"
                gets $cm yy
                after $delta
        }
}

# GetCSDPos takes a filename, numsamp, steps_per_samp, delta (step time in ms), settles
proc GetCSDPos {cm fn ns sps del stl}        {
        set opt1 [open "$fn" "w"]
        for {set i 0}    {$1 < $ns}     {incr i 1}        {
                after $st1
                puts -nonewline $opt1 [GetPos $cm]
                puts -nonewline $opt1 ", "
                step $sps $del $cm
        }
        close $opt1
}
```

The two example procedures above are included in the abs_proc.tcl file supplied with the demo board. The "step" procedure takes 3 arguments: the number of step pulses, the time between steps in msec, and the com name. For example:
%step 10 8 $com

The "GetCSDPos procedure takes 6 arguments: the filename to create, the number of CSD samples, the steps per sample, the time between steps, the settle time before taking the sample. This procedure creates a file in the working directory. For example:

% GetCSDPos $com myfile.txt 200 10 8 16

Figure 8 shows the TCLSH screen with the commands being run, the output file is shown below, the CSD is shown. To output decimal data, the “d” command could be used, for example:

% puts $com “d”
% gets $com dd

Then the GetCSDPos command would get Decimal data. Notice the gets command follows the puts command and just reads the com port to clear it.

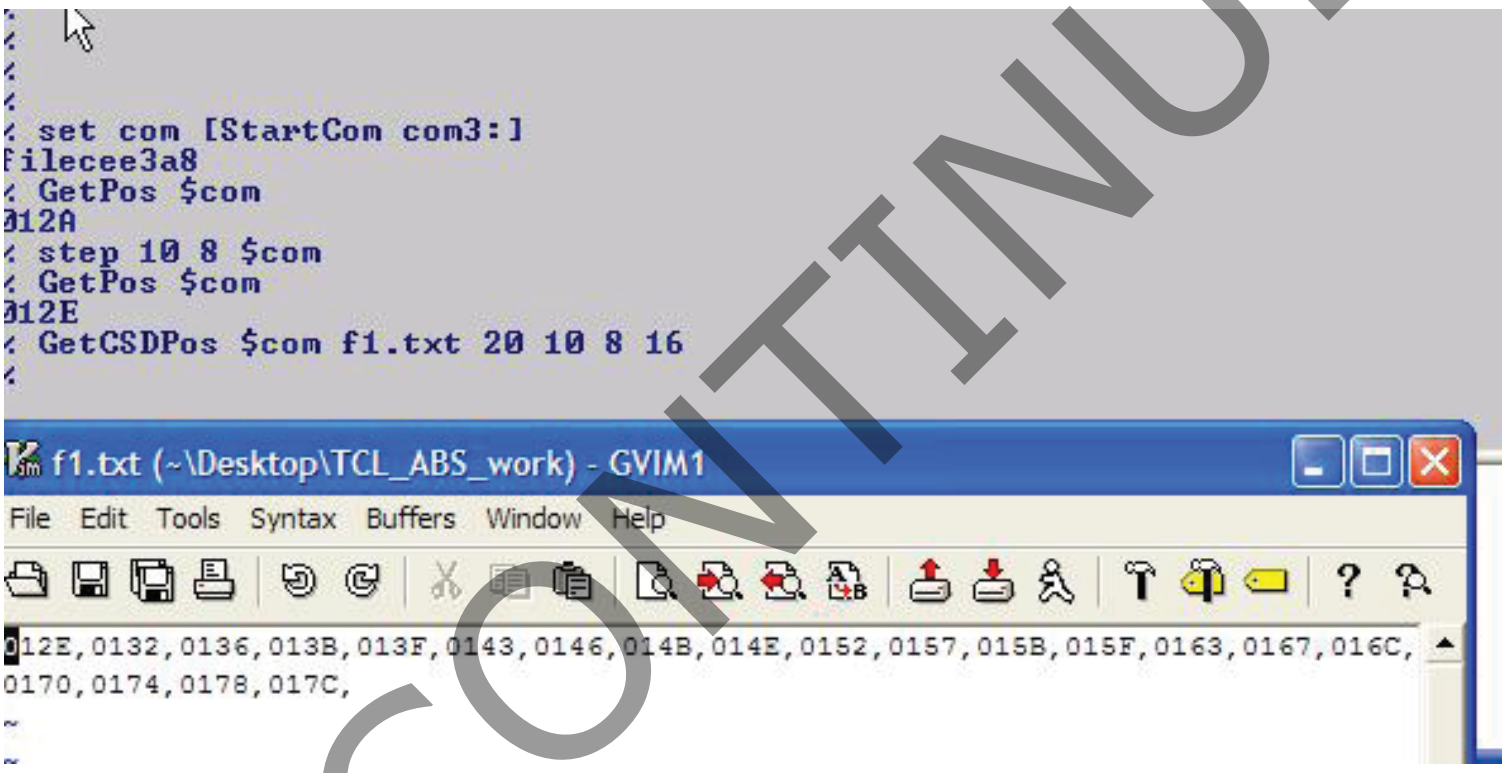


Figure 8: GetCSDPos example

CONNECTOR SCHEMATICS

The Encoder Connector is a 10 pin connector located at the bottom right edge of the board. Pin names are as shown and it is designed to match the ABS encoder connector.

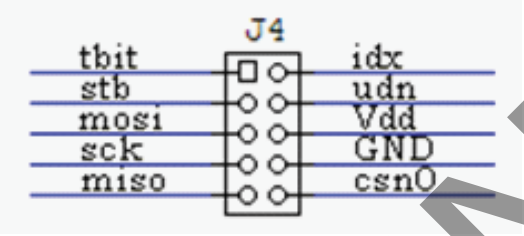


Figure 9: Encoder connector

The Expansion Connector is a 20 pin connector along the top edge of the board. The RC signals are visible in Figure 10, these are used by some commands for stepper control signals. The VDD supply is 5 volts but there is not much current available, it is only intended to drive the OPTO interface in a stepper driver.

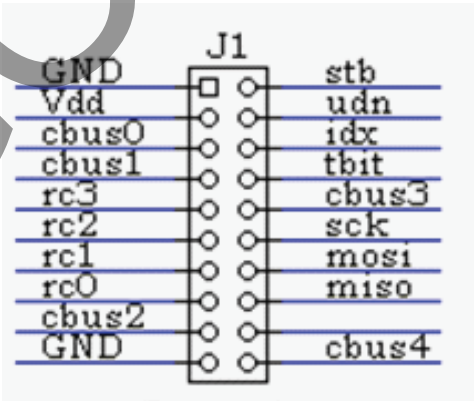


Figure 10: Expansion connector