copley controls

powered by analogic

Copley Basic Training

Contents

Introduction

Welcome! Thank you for taking the time to learn about Copley Controls and its products.

Copley Controls has been an industry leader on the field of motion control for 30+ years, our core line of products is servo controllers. In general terms, we can separate our servo controllers into two groups: panels & modules.

Panels are standalone drive units that have an enclosure, as you can see in the examples below:

Dual Axis Plus

Single Axis Plus

Single Axis ARM

Modules are usually smaller in size and are meant to be used by OEM's. Although Copley sells mounting boards for these drives (development kits board), most users of this line of products will lay out their own mounting board with help of our reference design documents.

Some examples of module drives:

NANO

ACK-HC

APV

Furthermore, we can subdivide the two groups into two additional groups, based on the main communication protocol they support (EtherCAT or CANopen).

Any Copley drive with an **E** as the second letter in their name model will be an EtherCAT drive such as: NES, XE2, XEC, XEL, AEV, BEL, BE2, AEM, AE2, IES, IEL, TEL, TE2, SE2, SE4, SEM, ME3, GEM.

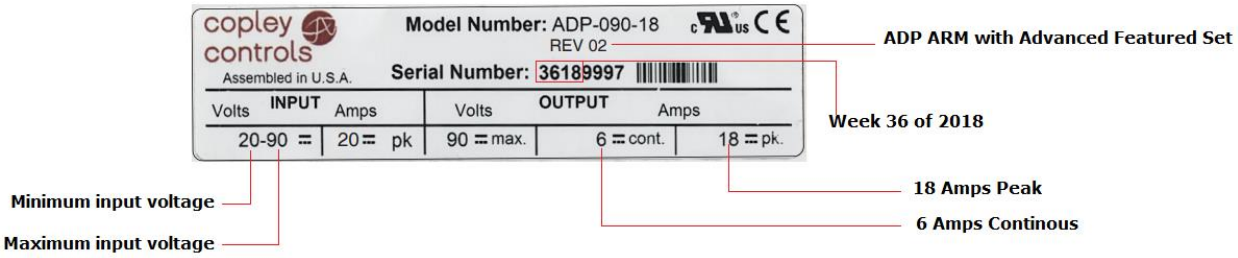
In general, any other Copley drive model that has any other letter as the second letter will be a CANopen drive such as: NPS, XPC, XP2, XPL, XTL, XSJ, APV, BPL, BP2, APM, AP2, ADP, ACJ, ACM, ACK, IPL, TP2, SP2, SP4, SPM, STX, STP, STM, STL, MP3, MP4, GPM.

Both EtherCAT and CANopen drives support serial communication RS-232.

Ratings Labels

See below examples of Copley Ratings labels:

Panel DSP & Arm Label



Module Label

Panel Plus Label

Hardware Connections

1-Powering the Drive

XEC-CK Connector Kit

Part #10-78871-01

2-Motor U,V,W & Feedback Connections

XEC-CK Connector Kit

Part #10-78871-01

STO JUMPER

Connector Kit Part #10-78871-01

3- Copley USB to RS-232 Adapter

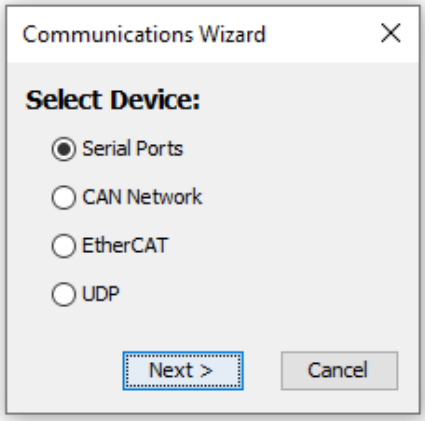
4-Use CME to enable the drive

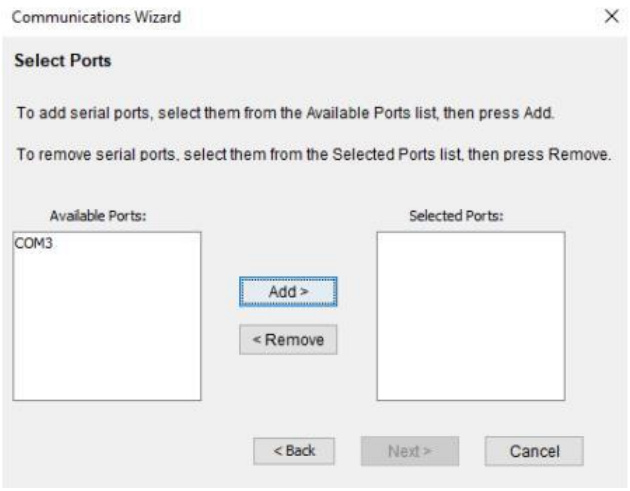
1- Connecting CME

A- Turn the power for the drive on.

B- Double-click the shortcut icon on desktop to start CME.

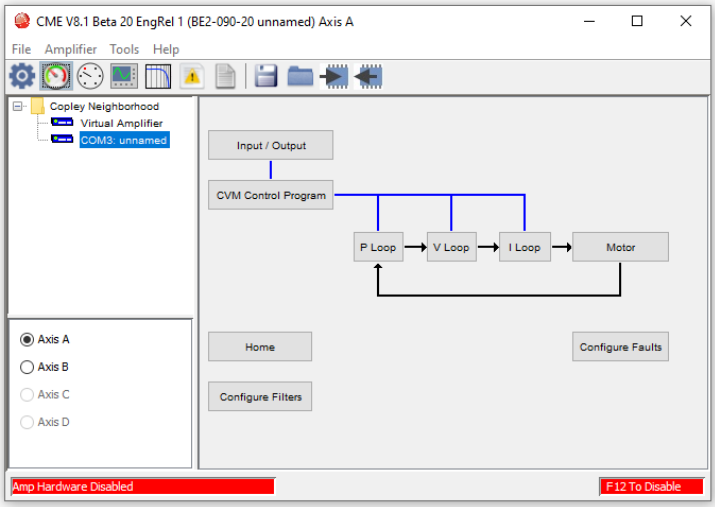
Note: If communications were set up already, the CME main screen opens. If communications were not set up, the communications wizard will be displayed as shown below:



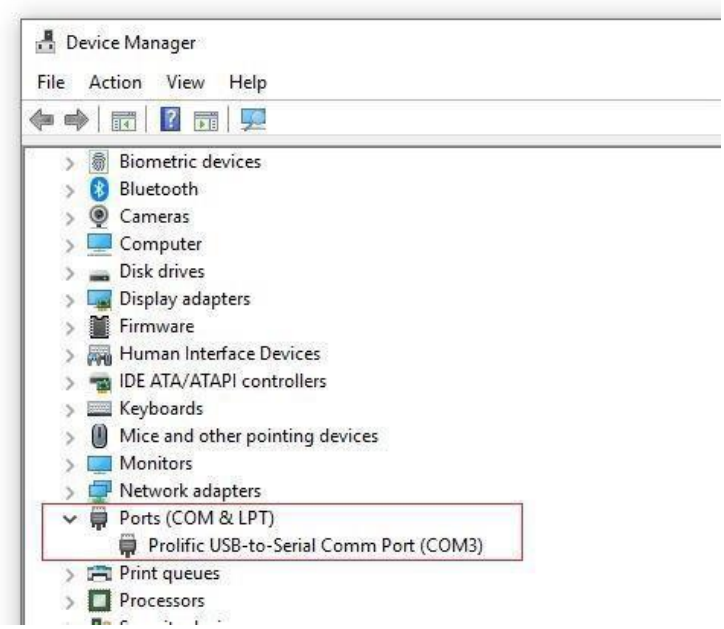


C- Select the port (Example port COM3), click add, Next and Finish.

Once the communications wizard is setup, you should be able to see the drive like in the picture below.



Note: when using Copley USB to Serial adapter (Part# SER-USB-RJ11), communication should start working right away. In the rare event it doesn't, go to Windows/Device Manager/Ports:

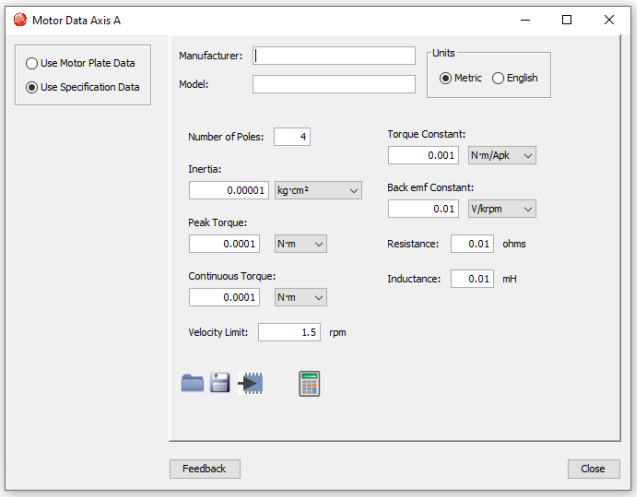


Select the device, mouse right click:

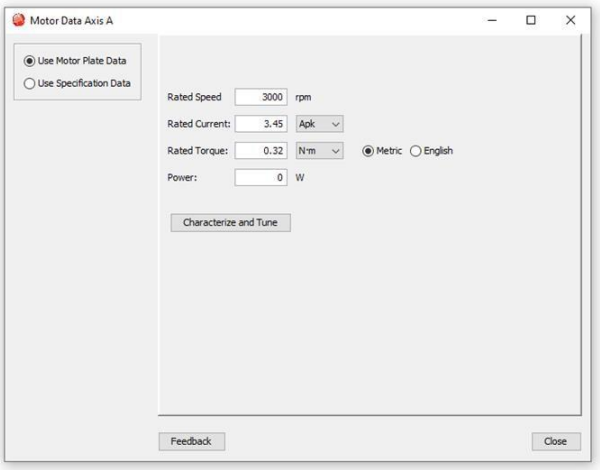
Now you should be able to see your drive just like the image on step C.

Motor Data and Characterizing

The next logical step to control a motor is to populate all the fields in the Motor screen, all this information can be found on the datasheet of the maker of the motor that you are trying to control:

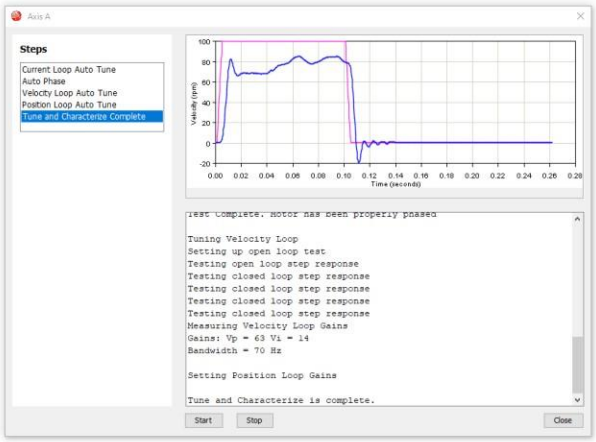


Sometimes, especially with older motors, the end user does not have access to the datasheet because they are not available anymore. Copley has developed a feature within CME that will help the end-user characterize a motor, when the user provides at least 3 out the 4 values requested. These values are usually available on most motor labels, see example below:

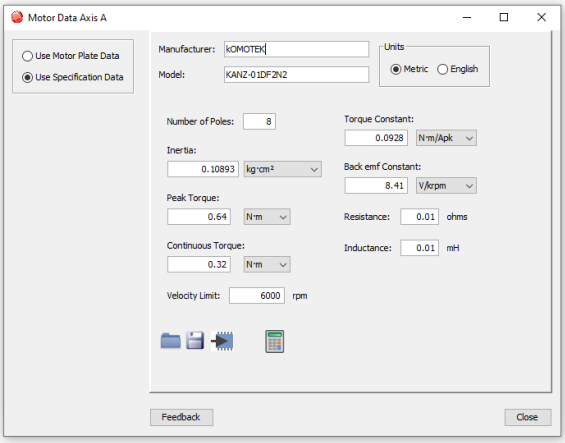


We fill-in the data displayed on the motor label above, then click on characterize and tune. On the next screen (Steps), press start to begin the process of characterizing & tuning. Follow the directions. Once you are done, you should see the screens below.

Characterization is done



Now you have data in the fields



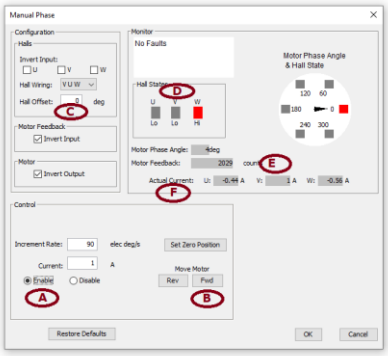
Note: the auto Characterizing algorithm in CME does not populate the fields for inductance and resistance, please use an LCR to measure both. Keep in mind the inductance value is very important for the tuning of the I loop.

No matters what approach was used to populate the motor data (Motor plate or Specification), press the calculator icon at the bottom of the page to apply calculated values derived from that data. Save the motor file pressing the floppy disk icon.

Manual & Auto Motor Phasing

Option 1 Manual Phasing

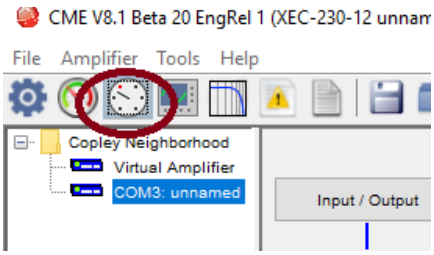
To manually phase the motor, go to main menu tools and then select manual phasing and then follow these steps:



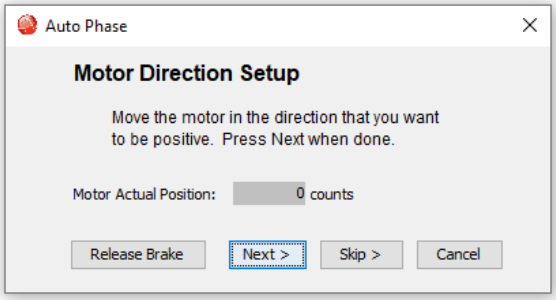
Step	Description
A	**Enable** a vector of current (black needle)
B	To **Move Motor** press and hold the forward button, you should see that the current vector rotates, and the halls (red squares) are getting decoded. Note that as we move forward with no load connected to the motor, it may appear to be a little lead from the red indicator (ahead of the black needle) and same thing should be true when press and hold reverse direction button.
C	The **Hall offset** is based on the mounting of the feedback device; therefore, you may see a little more lead in one direction and perhaps a lag in the in the other. End-users can use this screen to balance the forward & reverse between the motor phase angle and the hall states. Keep in mind most changes taking effect in the hall states happens in increments or decrease of 30°.
D	**Hall States**, we can see the hall states change, going on & off as we go forward or reverse.
E	**Motor feedback**, as we move we should see counts go up; if we are moving in the direction, we consider positive and down if negative or reverse. If this is not the case, we can invert the feedback adding a checkmark to the right option.
F	**Actual Current**, we can also monitor the current. In the picture example we commanded 1 amp and if we look at the current while we rotate the vector, all three phases will positive and negative 1 amp at the time. This feature could be very helpful while trying to troubleshoot motor wiring.

Option 2 Auto-Phasing

To Auto-Phase the motor, go to main menu Amplifier and then select Auto-Phasing or click the third icon from left to right on the main CME screen.



On the next screen you need to move the motor in the direction that you consider positive. Remember that if you have a brake, you need to press release brake before attempting to move the motor shaft.



Follow all the directions until the end and press ok to save the configuration.

Manual & Auto Current Loop Tuning

Option 1 Manual Tuning

The current loop tuning is dependent primarily on the inductance of the coils of the motor. The tuning of this loop will not produce any motion on the motor. To tune the I loop, we will utilize a simple procedure of adjusting its gains Cp & Ci

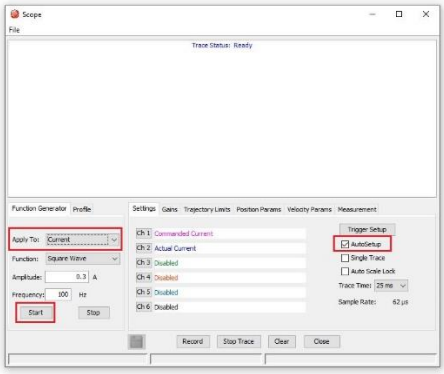
Gain	Description
Cp - Current loop proportional	The current error (the difference between the actual and the limited commanded current) is multiplied by this value. The primary effect of this gain is to increase bandwidth (or decrease the step-response time) as the gain is increased.
Ci - Current loop integral	The integral of the current error is multiplied by this value. Integral gain reduces the current error to zero over time. It controls the DC accuracy of the loop, or the flatness of the top of a square wave signal. The error integral is the accumulated sum of the current error value over time.

The following procedure can be applied to any motor rotary, linear or voice coil that is properly connected to a Copley drive.

Keep in mind that if the inductance value of the motor you are trying to tune is very low <200 µH, you could experience trouble with ripple currents or if it's < 100 µH CME, you could see a short-circuit detection by CME.

To begin, remember that effective tuning relies on good data. The appropriate values for resistance and inductance should be entered on the motor specification screen.

A-To manually tune the I loop, select the scope tool on the main CME screen:

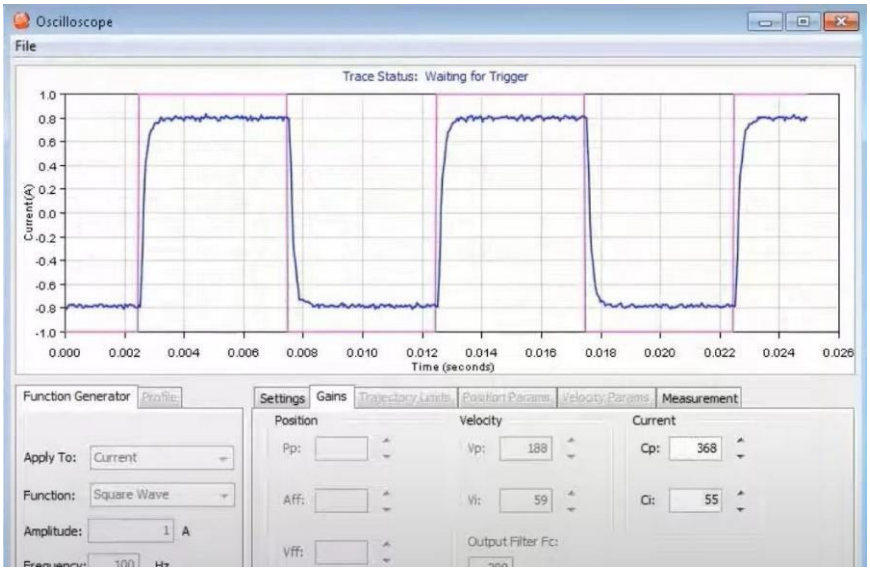


-Under function generator select apply to current.

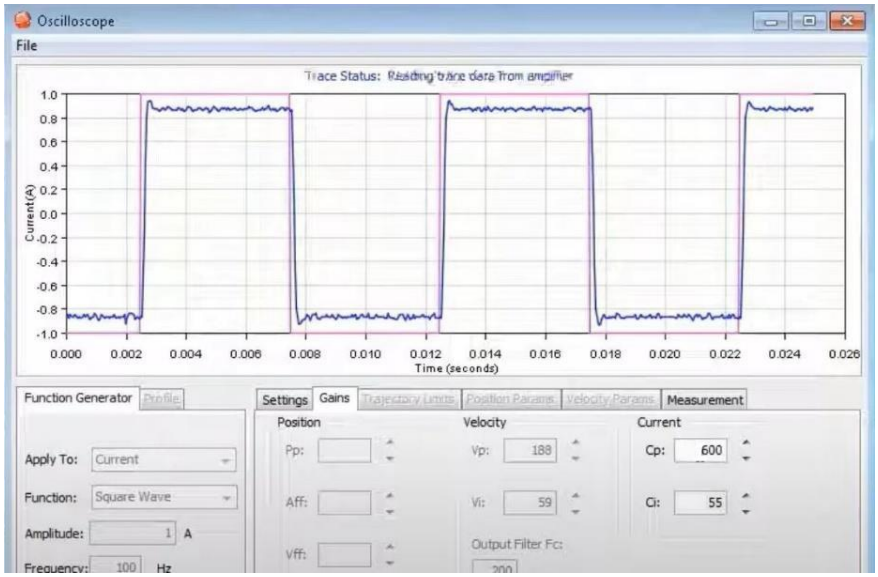
-Add a checkmark on Auto setup.

-Click Start.

B-Select the Gains Tab; in this new screen we will adjust both gains Cp & Ci to optimize the tunning values.

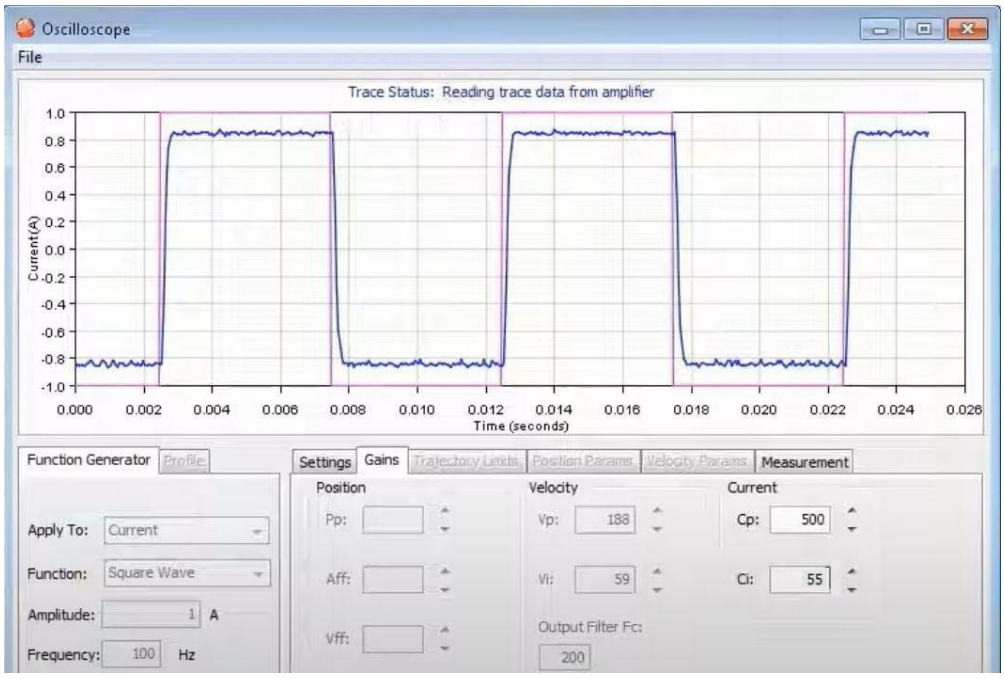


C-As you can see in the picture below, we doubled and rounded-up the value for Cp. The idea is to try to match the profile of the commanded current (magenta):



D- We will double the value of the Cp gain until we see an overshoot:

E- Then, we will cut the value of the gain to about half the overshooting value and move to tune the Ci gain. Note there are no "spikes" on top of the (blue) actual current square wave.

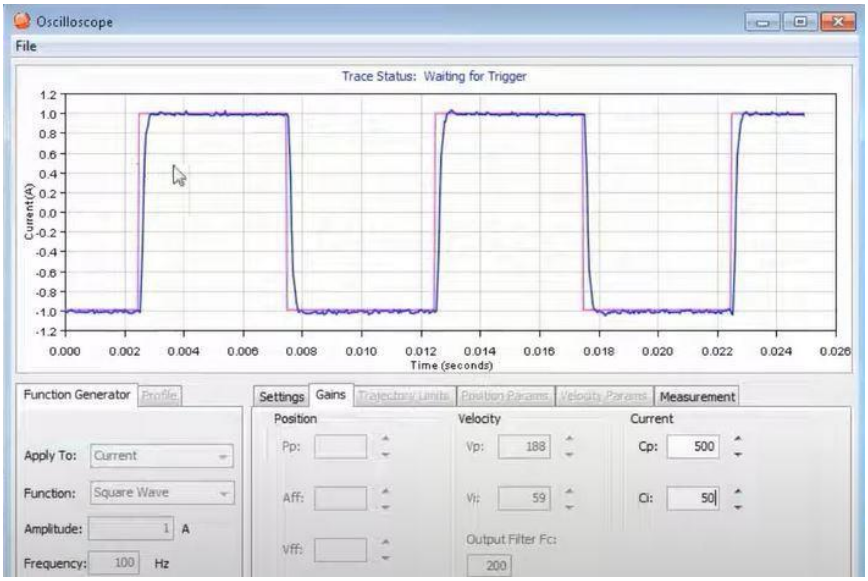


F- For the Ci gain tuning, we will follow the same procedure, increasing the value of the gain until we see an overshoot, then cutting that value in half.

Note: a good value to start the tuning of the Ci gain is about 10% of the value of the Cp, but in the picture below, notice we intentionally started with a large value to demonstrate how the square wave form behaves once we increased the Ci value.

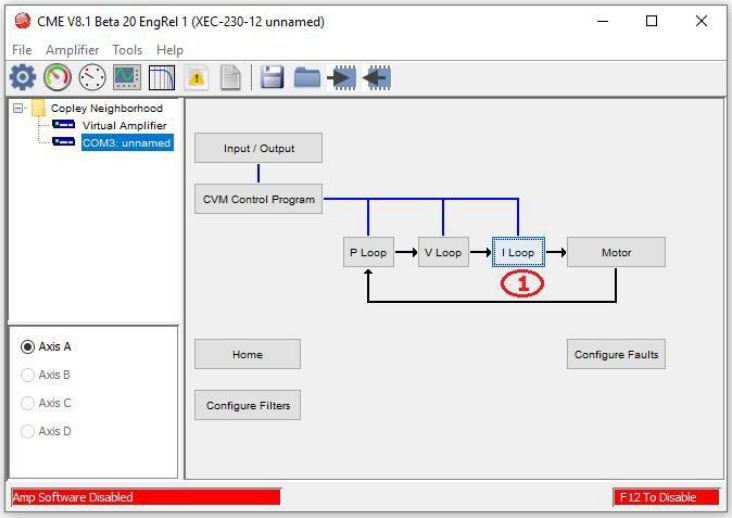
G- We decreased the value but there is still some overshoot:

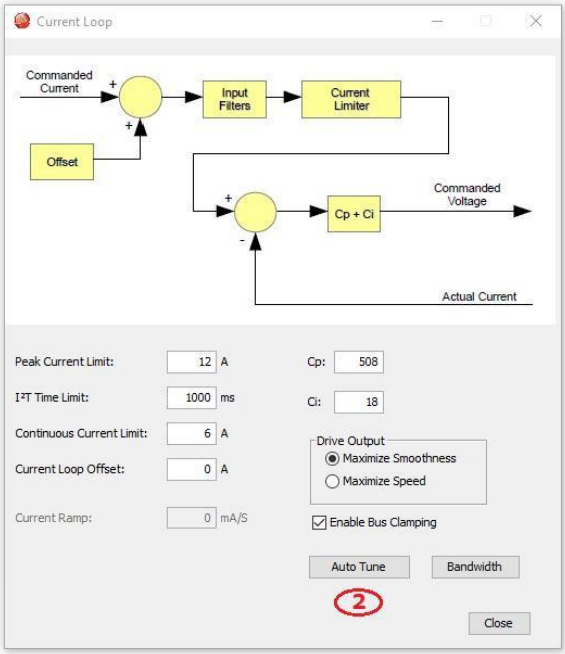
H- Then, we reduce the value further until both square waves, commanded current (magenta) and actual current (blue) match each other closely.



Option 2 Auto Tuning

To use the Auto tune tool, go to I loop and then Autotune, follow all the steps.





2- Manual Velocity Loop Tuning

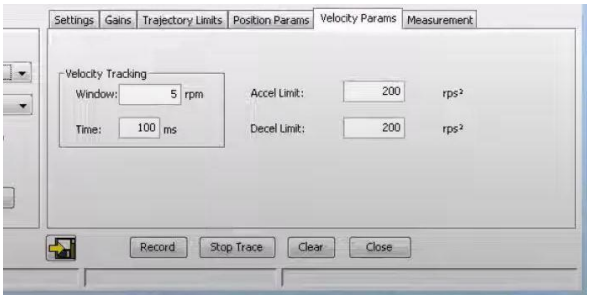
Before attempting to manually tune the velocity loop, please make sure the motor is free to move, and is properly phased and the current loop is tuned. To tune the V loop, we will utilize a simple procedure of adjusting its gains Vp & Vi:

Gain	Description
Vp - Velocity loop proportional	The velocity error (the difference between the actual and the limited commanded velocity) is multiplied by this gain. The primary effect of this gain is to increase bandwidth (or decrease the step-response time) as the gain is increased.
Vi - Velocity loop integral	The integral of the velocity error is multiplied by this value. Integral gain reduces the velocity error to zero over time. It controls the DC accuracy of the loop, or the flatness of the top of a square wave signal. The error integral is the accumulated sum of the velocity error value over time.

To manually tune the V loop, select the scope tool on the main CME screen:

A-Under function generator select apply to Velocity.

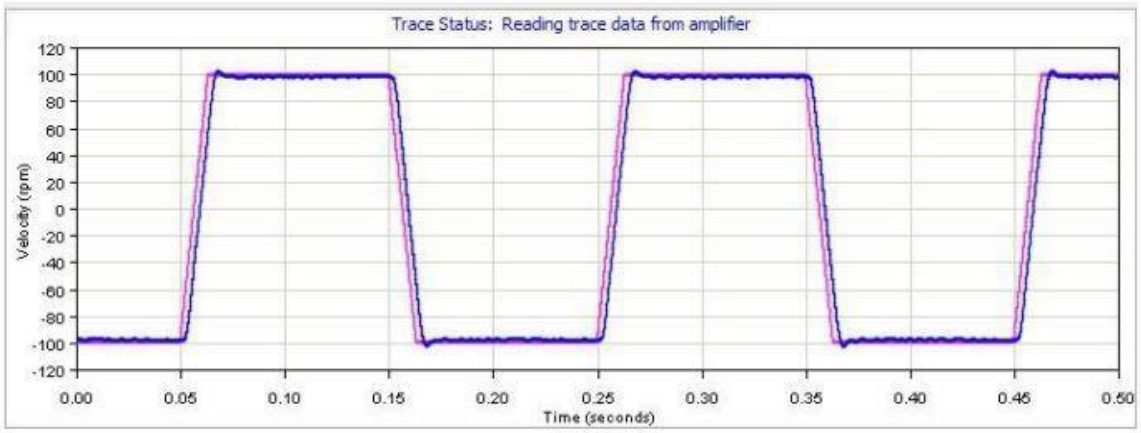
B-Click on the Velocity Parameters Tab and make sure the values for acceleration and deceleration are not too aggressive.



C-Click back to the settings tab and make sure the checkmark on Auto setup is on.

D-Click Start.

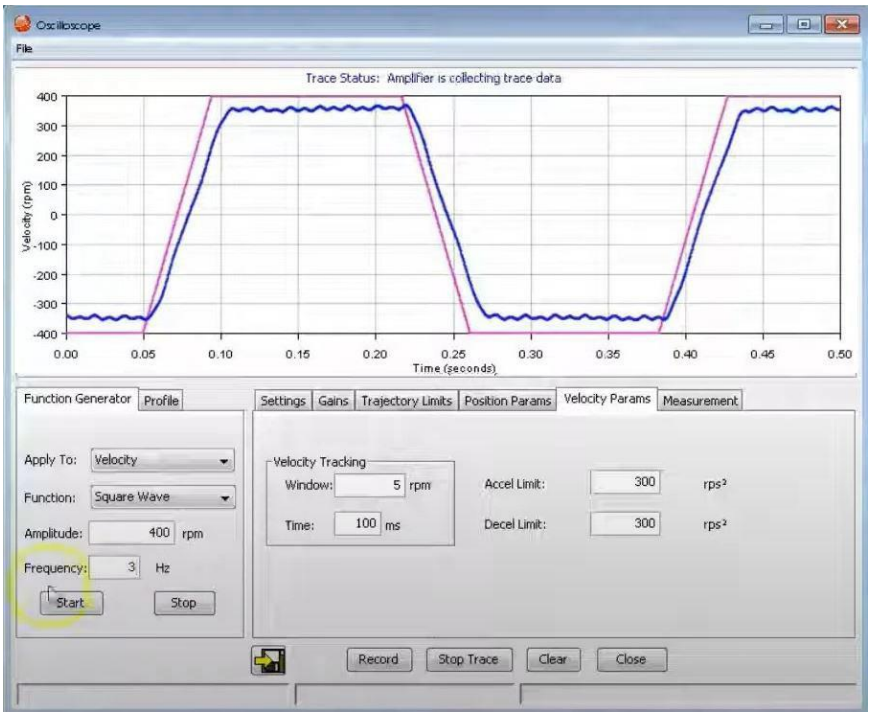
E-Just like the tuning of the current loop, the idea here is to closely match the square waves of commanded velocity (magenta) with the actual velocity (blue). The picture below depicts and optimal tunning of the V loop:



However, there could be instances where the tuning of the V loop is not straight forward. In the picture below, we don't have a trapezoidal shape for our wave.

In this case, we need to increase our acceleration & deceleration values from 200 rps^2 to 300 rps^2. Also reduce the value of the amplitude and frequency, see picture below.

The implemented changes will yield wave forms that closely match the commanded and actual velocity profiles.



F-We can still adjust the gains Vp & Vi to achieve a much precise tuning:

3- Manual Position Loop Tuning

Before attempting to tune the position loop please make sure you motor is free to move, it's properly phased, and the current & velocity loop are tuned.

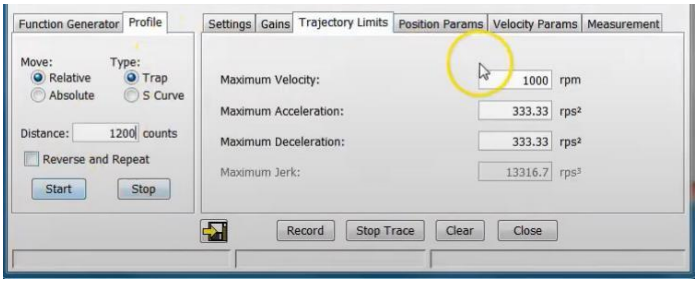
A-Select the Profile Tab and select a distance you want to move; in this example we are moving 1200 counts (about a quarter of a full turn of the motor shaft).

B-Click on the trajectory limits Tab and make sure the values for maximum velocity, acceleration and deceleration are not too aggressive (Picture 1).

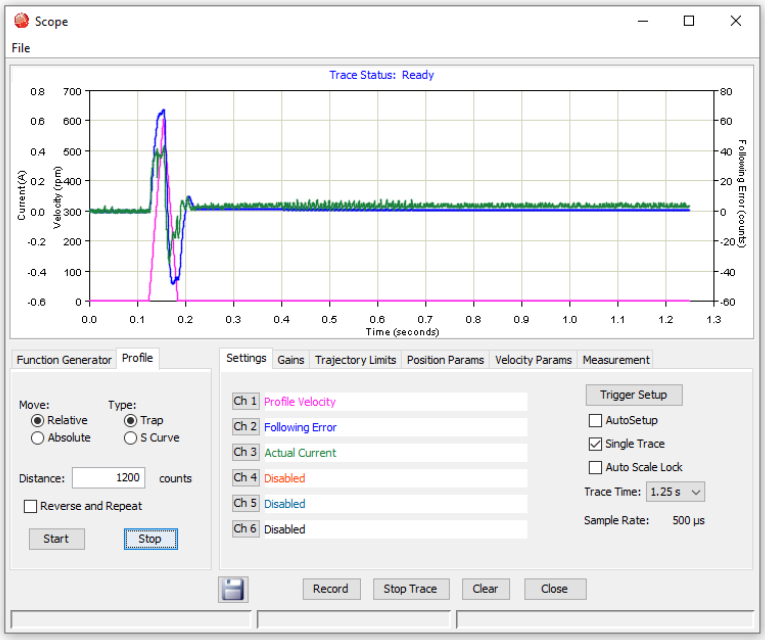
C-Go back to the Settings tab, make sure Auto Setup checkbox is checked.

D-Add actual current to the third channel of the scope (Picture).

Picture 1



Picture 2



In the picture above, we can see how much current we utilize: 0.4 Amps to make 1200 counts move. We also see that we have about 10 counts of following error.

4- CVM program

The CVM indexer is a program that runs on a drive as a standalone program. CVM has a GUI to interact with the drive and runs in the background as a low priority, saving processing capacity.

To effectively work with CVM, it is a good idea to have a basic understanding of two Copley documents: Indexer User Guide and ASCII Programmers Guide. Both documents are available in the help section of CME.

In the Indexer User Guide, we can find the functions to be used in our exercise today. These functions are Java Beans, which reside on the computer and add functionality to the CVM indexer.

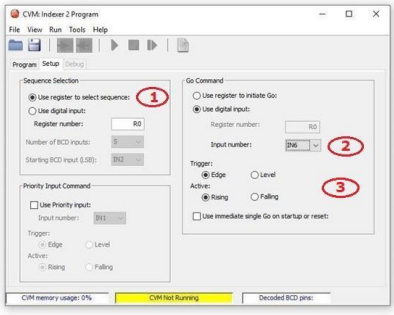
In the ASCII guide we can see the I command, which is used to read and write an index or register.

This is an extremally simple exercise and is intended to demonstrate to the new user some basic capabilities of CME such us move, wait for time delay, and conditional jump.

Exercise Setup

A-Open CME, select the Setup Tab and select the following

options:

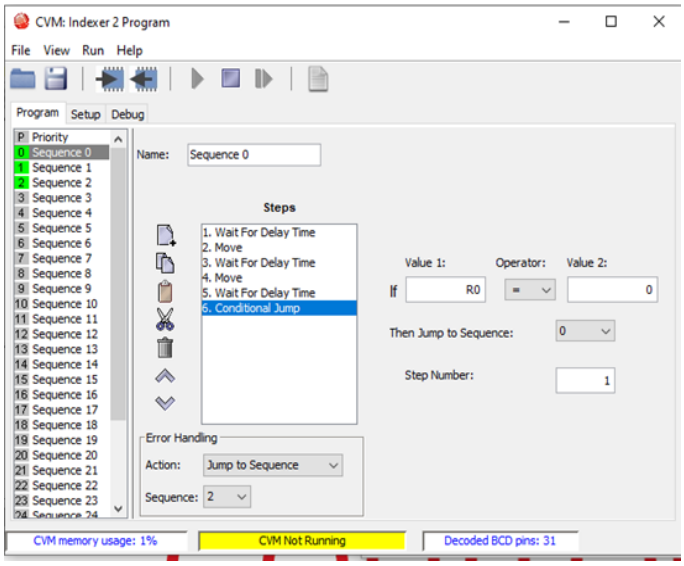


1-Use a register we will change the value)

2-Select and Input as a Go Command.

3-Select the trigger in this case the transition from low to hi of IN6 (rising edge)

First Program



-Select the program tab and click Sequence 0.

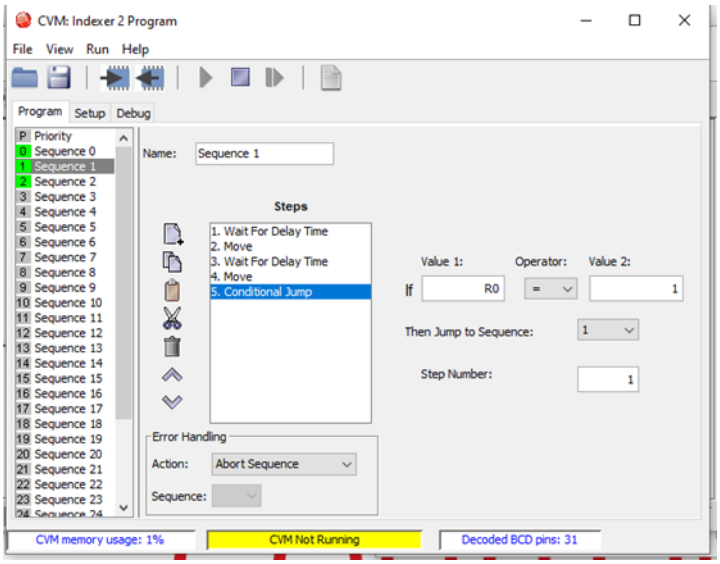
- Click the icon Add new step.

Wait For delay= X ms
Move= Distance X counts
Wait For delay= x ms
Move= Distance x counts

Conditional Jump= If R0 is = to 0 then execute sequence 0 step 1.

I hope you can see how with this small program we created a loop. Also notice that in the error handling section we are telling the program that if an error occurs while trying to move, we have a rule in place, jumping to sequence 2.

Second Program



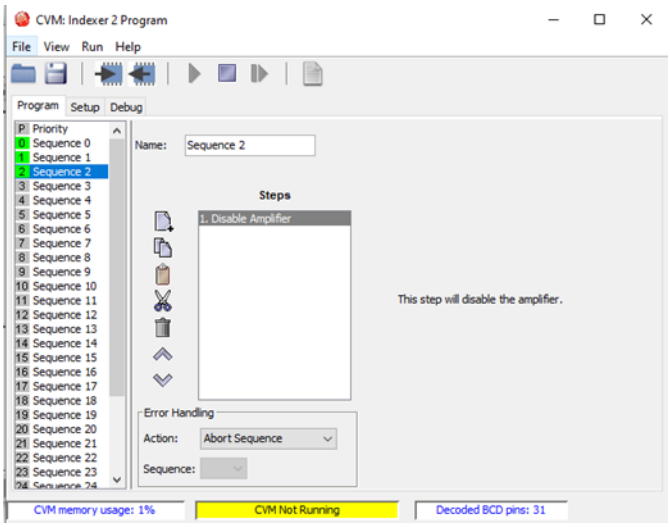
-On the program tab and click Sequence 1.

-Click the icon  Add new step
Wait For delay= 1000ms
Move= Distance 5000 counts
Wait For delay= 1000ms
Move= Distance -5000 counts

Conditional Jump= If R0 is = to 1 then execute sequence 1 step 1.

Notice the error handler, it will abort immediately in case an error occurs while moving.

Third Program

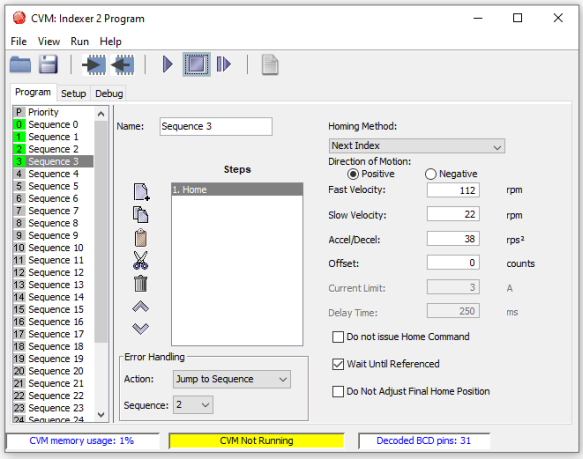


-On the program tab and click Sequence 2.

-Click the icon  Add new step

Sequence 2 will be a separate disable function

Fourth Program



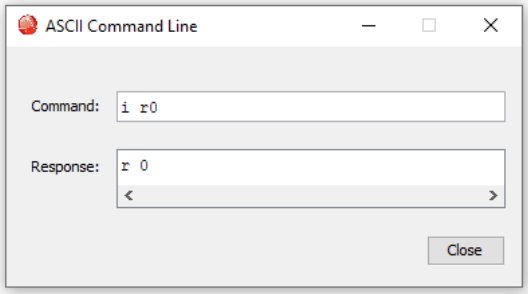
-On the program tab and click Sequence 3.

-Click the icon  Add new step

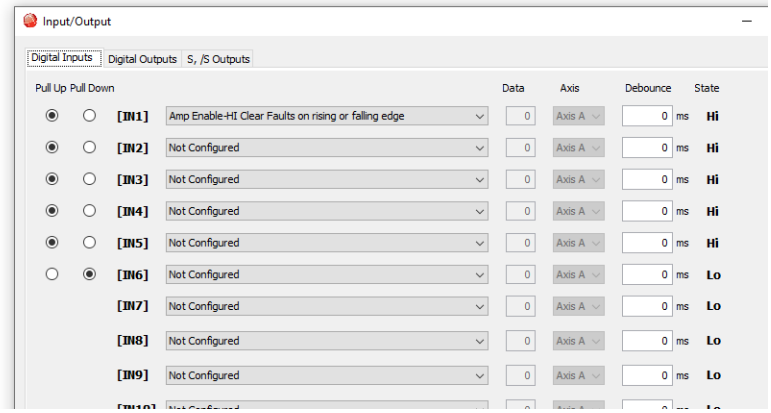
-Sequence 3 will be a homing function

Now we will use the ASCII command line get the current value of R0 to do this we will use the i command which can be used to read and write to CVM 32 registers.

Let's get the value of **R0: i r0** return value should be 0, remember that this value is a condition for sequence 0



Now that we know that R0=0 we can trigger from IN6 from Low to Hi



The loop we programmed for sequence 0 will start.

Now let's change the value of register 0 to 1 let's type in the ASCII command line I r0 1 the motion programmed in sequence 0 will stop and the next time we transition IN6 from low to Hi it will trigger sequence 1 because our condition in that program is that R0 must be = to 1.

Now if change the value of R0 to a value that is not equal to 0 or 1. I can still trigger the homing routine that we associate to IN6 transitioning from low to hi (the only requirement here is the transition).

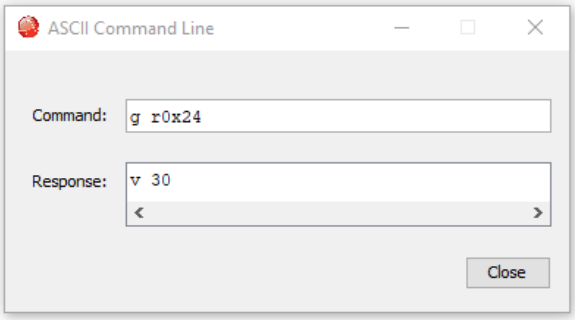
5- ASCII command line

The American Standard Code for Information Interchange is a character encoding protocol in which a host PC uses ASCII characters to send commands to a device and then receives responses back from that device.

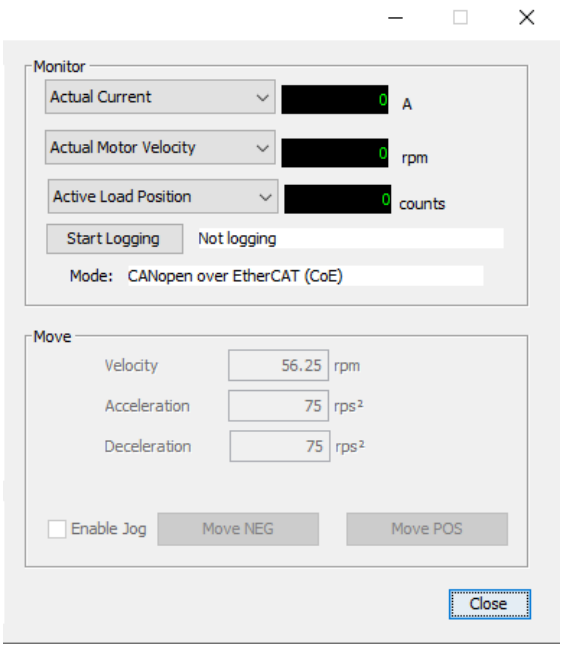
Code	Command	Description
s	Set	Set a value of a parameter in ram or flash.
g	Get	Read the value of a parameter in ram or flash.
c	Copy	Copy the value of a parameter from ram to flash or flash to ram.
r	Reset	Reset the drive.
t	Trajectory	Trajectory generator command.
i	Register	Read or write the value of a CVM program register.

ASCII control over the serial rs-232 allow us to make moves, enable and disable the drive, turn on and off specific features.

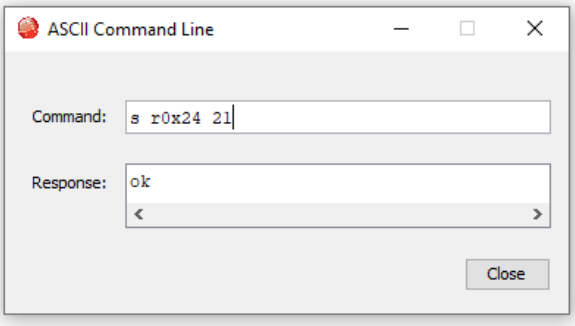
A- For example, to know the mode of operations that currently our drive is we can type:



We received as return a value of 30 that means that the drive is in CANopen mode:

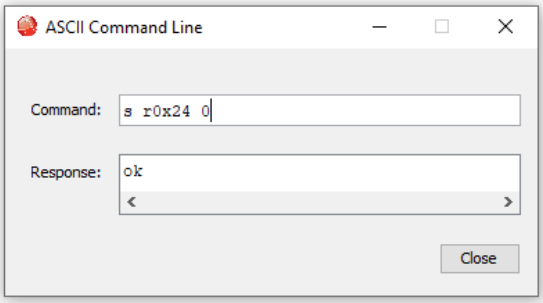


To change the mode of operations from CoE to Software programmed:

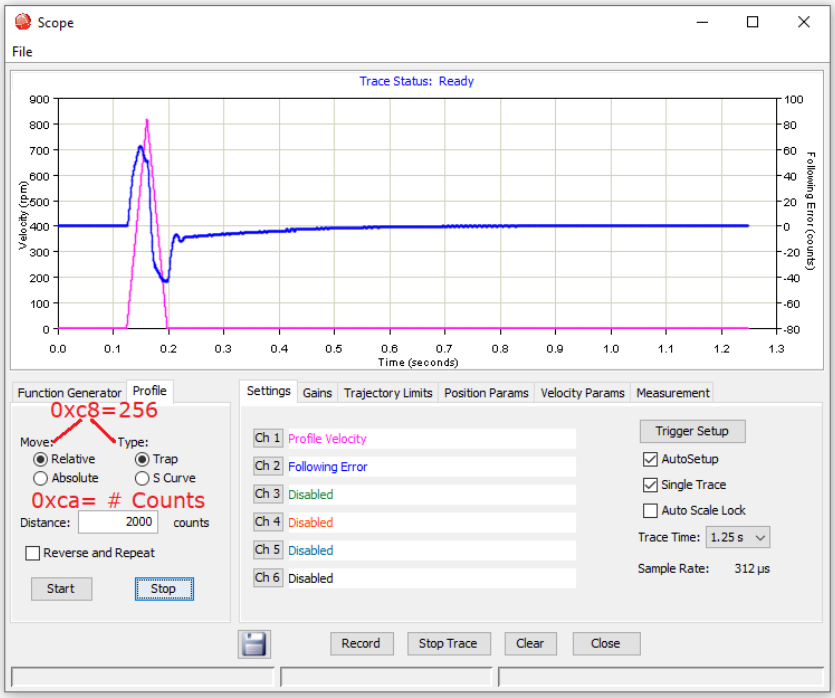


S= Set
R=Ram
0x24=desired state
21= Trajectory generator drives position loop

To disable the drive, we can write the value of 0 to 0x24, obviously to re-enable the drive all we need to do is write a value of 21 to 0x24.



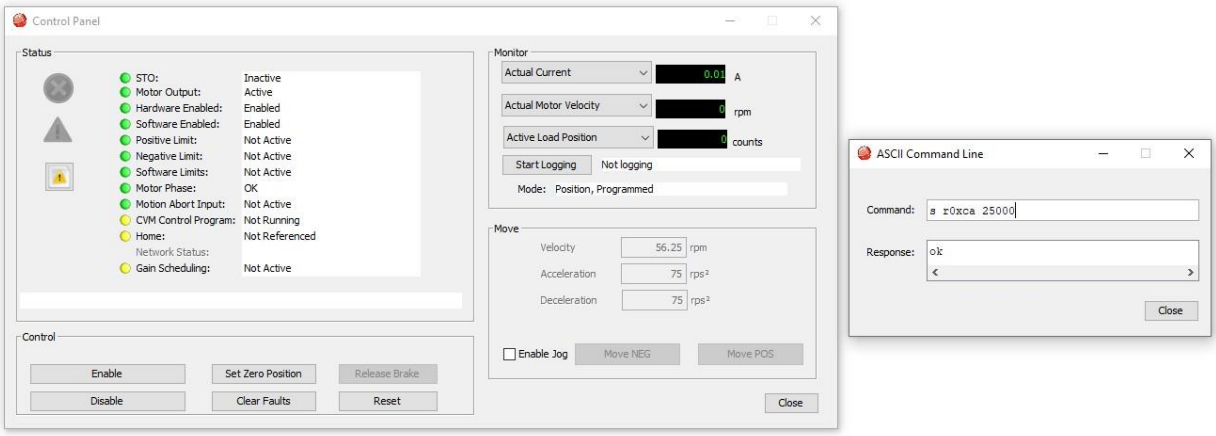
Because we are in position mode (21) we can make moves using ASCII commands, If we look at the ASCII Programmers Guide, we will find all the parameters associated available to be used under position mode.



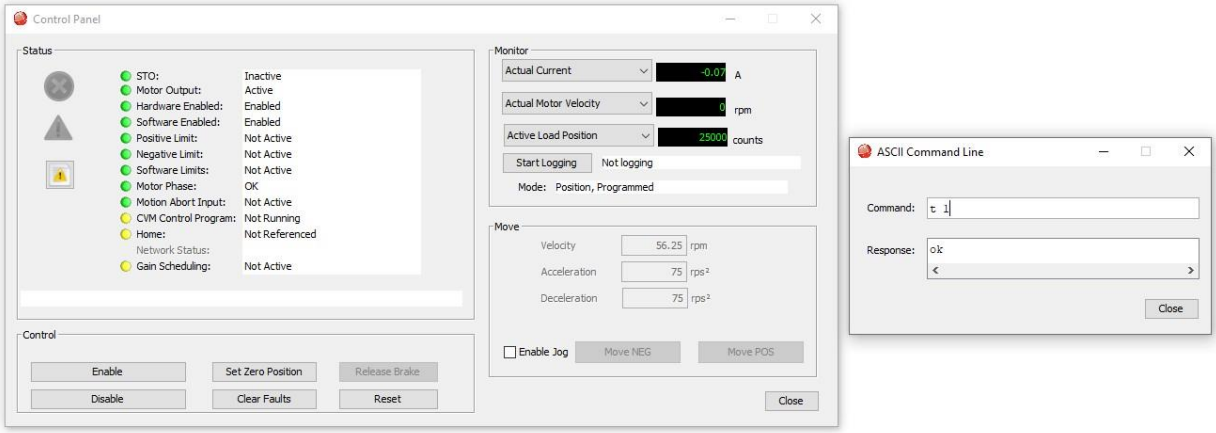
If we read parameter 0xc8 we will get back a value of 256 which means that it will perform a Relative move with a trapezoidal profile.

Parameter 0xca is Position command and the units must be in counts we can program any specific position in units of counts that we want.

For example, we can make a move of X number of counts just typing in the ASCII command line



Then issue the t 1 command to execute



We could use ASCII to modify any parameter available in CME.

See Page 28 Guide

6- What to Order When Purchasing Copley

All Copley servo drives have a datasheet full of technical data related to the drive, at the end of all datasheets we have a section called **Ordering Guide**.

This section talks about specific connectors and accessories related to the drive in the datasheet. In general terms any Copley drive is going to need the following accessories:

- 1 USB to serial RS-232 adapter.

-1 Connector kit specific to the drive (CK).

Depending in the communication protocol used:

-CANopen (NK) keep in mind that for CANopen the end user will need a CAN card, Copley recommends the use of CAN-USB-01

-If Ethernet is going to be used a network Kit (NK) is going to be need it.