M Servo Suite
Software Manual
V1.1

1 Revision History

Version	Author	Participator	Date	Changes
1.0	Austin		2013-7-19	Initial release
1.1	Frank	Austin	2015-05-08	Update new features in M Servo Suite

Contents

2 Introduction

Thank you for purchasing MOONS' M2 Series AC Servo products. With the excellent product properties, good usability and competitive price will make your applications more than expected.

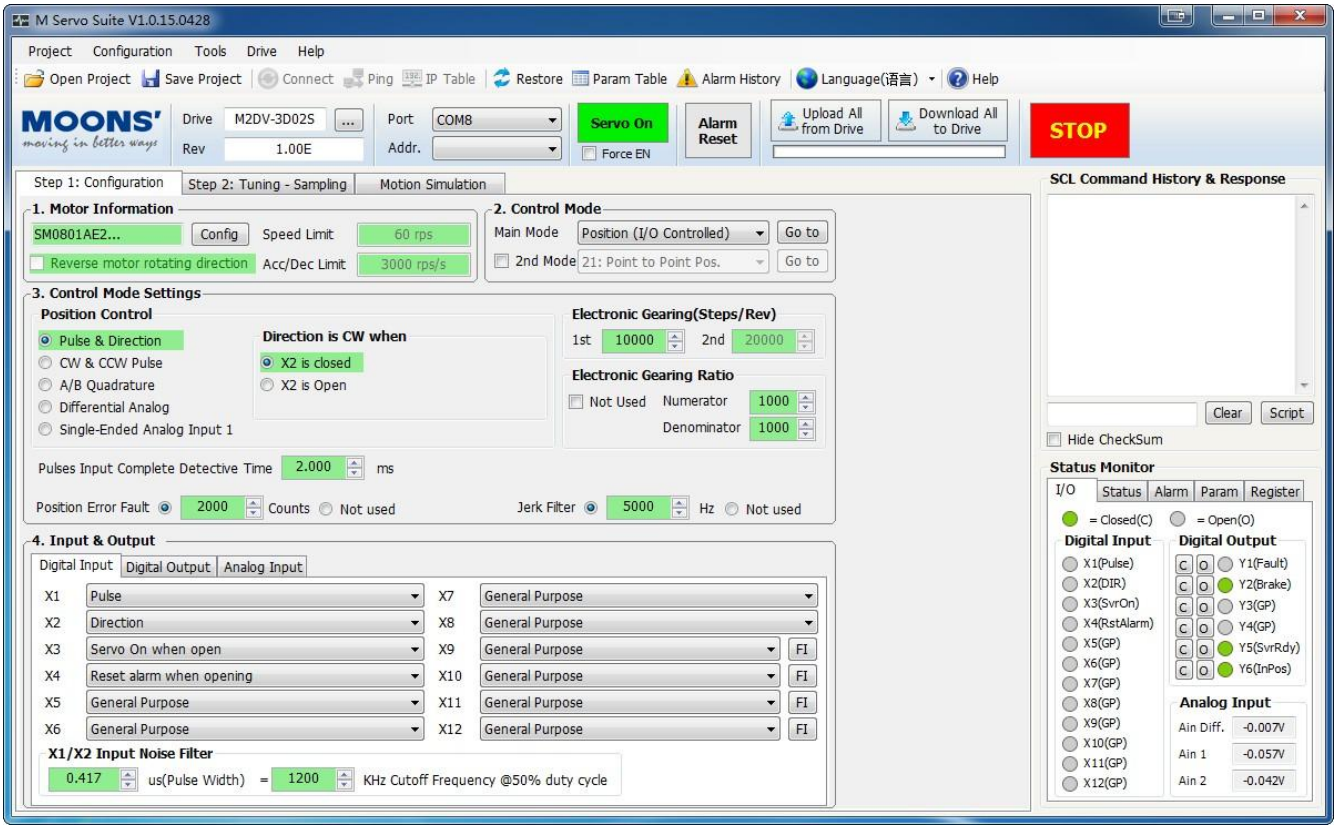
M2 Series, a new generation AC Servo system, with good response Frequency and setting time, is designed by AMP in US and AMA team in China.

- Frame size :40/60/80mm Servo Motor
- Rated output power: 50/100/200/400/750W
- Easy to use: Online Auto-tuning
- Advanced Anti-vibration: Two notch filters
- Stand Alone mode: Q Program, Position Table
- Various Industrial Field Bus: Modbus/RTU, CANopen, Ethernet

M2 is particularly suitable for High Speed/Torque/Accuracy, more safety and long-life applications such as: FA, semiconductor manufacturing equipment, SMT, PCB, LED, Packaging, and Food processing equipment, robot and Non-standard machinery.

2.1 M Servo Suite Overview

The M Servo Suite is a PC based software application to configure, perform servo tuning, program the Q programming, drive testing and evaluation of the servo product. This help explains how to install the M Servo Suite and how to configure and tune your servo system. For information regarding your specific hardware, such as wiring and mounting, please read the **M2 User Manual** and **M2 Quick Setup Manual** that came with the product.



The features of M Servo Suite include:

- Friendly Interface

- Easy setup within just three steps
- Drive setup and configuration
- Servo control gains Auto-tuning
- Servo tuning and sampling
- Built-in Q programmer
- Motion testing and monitoring
- Write and save SCL command scripts
- Online help integrated

If you get in trouble with using our Driver or software, or if you have any suggestions about our products and this manual, please call (86)400-820-9661 or fax to (8621)6296-8682. And also you can send an E-Mail to ama-support@moons.com.cn to let us know.

Support Operation System:

Microsoft XP (Service Pack 3), Windows 7/8, Vista with 32bit or 64 bit @1024X768

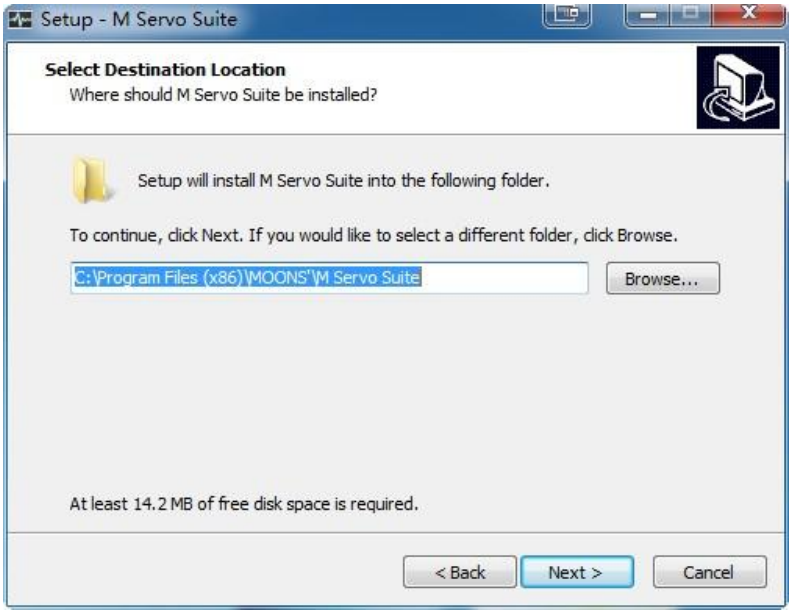
Microsoft .Net Framework 2.0

2.2 M Servo Suite Setup

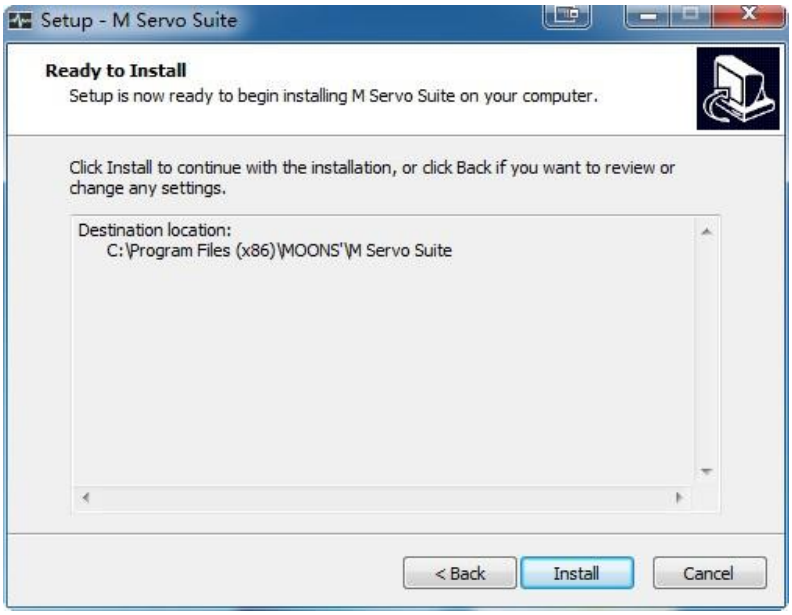
M Servo Suite can be download from MOONS' website http://www.moonsindustries.com/

Step 1: Open M servo Suite steup file for installation

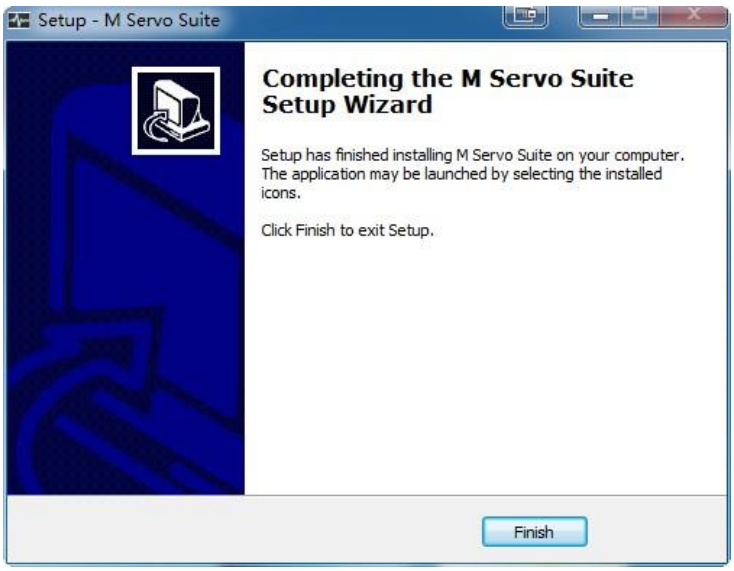
Step 2. Choose install file location



Step 3: Ready to install



Step 5: install Complte



2.3 Install M2 hardware Drive

M2 driver uses Mini USB for communication. It will be installed automatically when you install M Servo Suite software.

If you PC ask you to install the hardware drive when you connect M2 driver to your PC, you can find the drive file directly under the software installation file.

In default mode, the file location will be:

32bit system：C:\Program Files\MOONS'\M Servo Suite\Driver Installation Tool

64bit system：C:\Program Files (x86)\MOONS'\M Servo Suite\Driver Installation Tool

In this file, you can choose:

“x86”for 32 bits system

“x64” for 64 bits system

Based on your PC system, select correspondent file. Double click MCP2200DriverInstallationTool to install the drive

3 Use M Servo Suite software to connect driver

M Servo Suite offers two types of communication connection: **serial communication port** and Ethernet communication

3.1 Connecting Drive to M Servo Suite

Connect to M2 Drive via serial communication:

- Connect the drive to your PC COM port
- Launch M Servo Suite
- Switch to RS232 and select the COM port, see picture below
- Power up the drive
- M Servo Suite recognized the drive model and revision

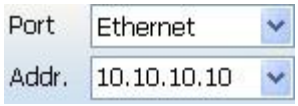
When launch M Servo Suite, the software will search all COM port available and load to the drop down list.

After established the connection between the drive and M Servo Suite, the software will switch the baud rate to 115200 bps, no matter what the baud rate is.

For Ethernet drive, the connection includes following steps

- Connect the drive and PC to your switch or router
- Launch M Servo Suite
- Switch to Ethernet and input IP Address, see picture below
- Power up the drive

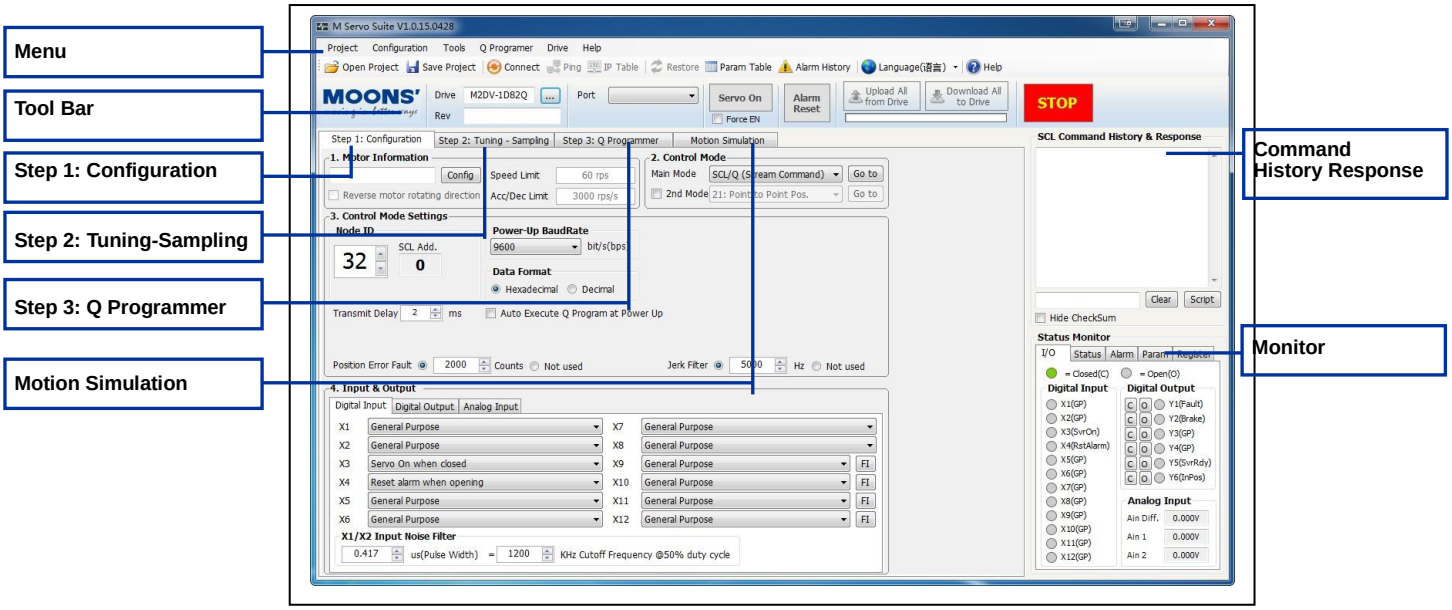
M Servo Suite will not detect the drive information automatically, you need to click "Upload" button in the main screen to get the drive model and revision.



3.2 User Interface

To launch the ***M Servo Suite,*** click windows menu: ***Start → Programs → MOONS' → M Servo Suite → M Servo Suite.***

The Main screen includes some sections, Menu, Tool Bar, Step 1: Configuration, Step 2: Tuning-Sampling, Step 3: Q Programmer (Only for –Q/-C Type) and Motion Simulation as shown below.



Menu

Main menu provides some frequently-used operations for Project, Configuration, Tools, Q Programmer, Drive and Help.

Tool Bar

Tool Bar is used to set the communication, Open Project, Save Project, Connect, Ping, IP Table, Restore, Parameter Table, Alarm History, Change Language, drive model, Servo status control, Alarm Reset, Upload & Download, Emergency stop.

Step 1: Configuration

This tab provides the drive configuration settings, such as 1 Motor Information, 2 Control mode, 3 Control mode settings, Input & Output.

Step 2: Tuning-Sampling

This tab provides the Auto-tuning and sampling settings, start sample and display sampling curve diagram.

Step 3: Q Programmer

This tab provides some functionality to program environment, test, save and download or upload the Q program. It is only for –Q and –C type.

Motion Simulation

This tab provides motion test, such as point to point motion, Jog, Homing etc...

SCL Terminal

The SCL Terminal allows you to send SCL commands to the drive.

Status Monitor

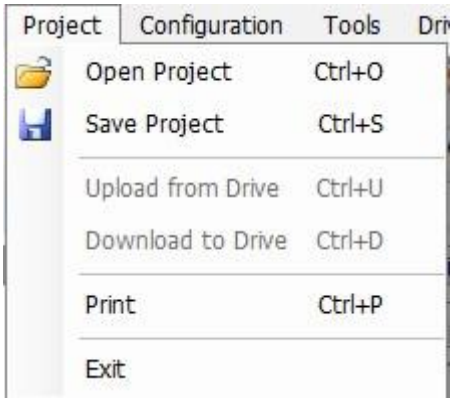
Status Monitor can display I/O status, Drive status, Alarm, Parameters and Register monitor.

3.3 Menu

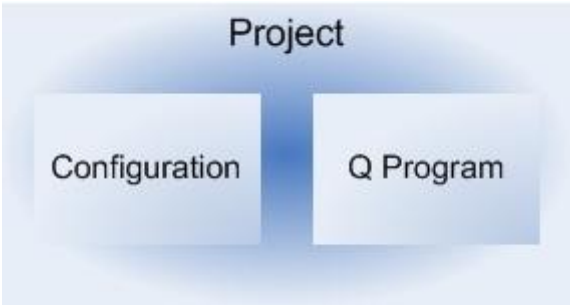
1st Stage Menu	2nd Stage Menu	Hot Key	Function
Project	Open	Ctrl+O	Open project file (.mdvprj format)
	Save	Ctrl+S	Save project file (.mdvprj format)
	Upload from Drive	Ctrl+U	Upload project from the drive
	Download to Drive	Ctrl+D	Download project to the drive
	Print	Ctrl+P	Print current project
	Exit		Exit M Servo Suite application
Config	Open Config	Ctrl+Shift+O	Open configuration file (.mdvcfg format)
	Save Config	Ctrl+Shift+S	Save configuration file (.mdvcfg format)
	Upload from Drive	Ctrl+Shift+U	Upload configuration from the drive
	Download to Drive	Ctrl+Shift+D	Download configuration to the drive
	Print	Ctrl+Shift+P	Print current configuration
Tools	Firmware Downloader		Upgrade the drive's firmware
	Calibration		Calibration for Non-Moons's motor
	Move Profile Calculator		Pilot motion profile based on target distance, velocity, acceleration/deceleration, etc.
	Export CANopen Parameters		Export CANopen Parameters to file
	CANopen Test Tool		Run CANopen Test Tool application (require pre-installation)
Q Program	Open Q Program		Open Q program file (.qpr format)
	Save Q Program		Save Q program file (.qpr format)
	Open Segment		Open Q segment file (.qsg format)
	Save Segment		Save Q segment file (.qsg format)
	Upload from Drive		Upload Q program from the drive
	Download to Drive		Download Q program to the drive
	Clear Q Program		Clear Q program
	Set Password		Set password to secure Q program
	Print Q Program		Print Q program
Drive	Connect	Ctrl+R	Connect or Re-connect to the drive
	Stop	Ctrl+F5	Emergency Stop
	Ping		Ping to the Ethernet drive
	IP Table		Edit user defined IP address through IP table No. 1to E
	Param Table		Display the Parameter Table
	Script		Run the SCL Script
	Option		Set Alarm, Regen, Communication and other options
	Restore		Configure the drive to Factory Default Setting
	Alarm History		Record drive's alarm history
Help	About		Get the software version
	Help Content		Open online help

3.3.1 Project

In project Menu, the M Servo Suite can allow you to upload and download both configurations and Q program. Driver's configurations and Q programs can save as project file (.mdprj) to your local disk. It can also download the project files to a different drive directly from the hard disk. In addition, it can also print out the detailed project files.



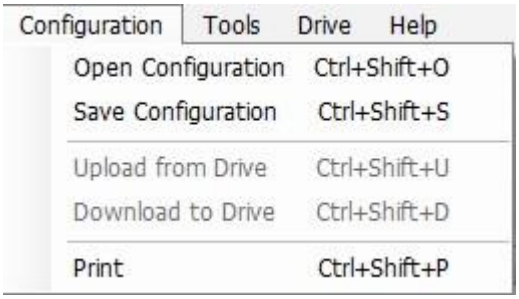
For the drive supporting Q Program capability, the project includes the configuration and Q program , see picture below:



For the drive without Q Program capablility, the project is the same as the configuration.

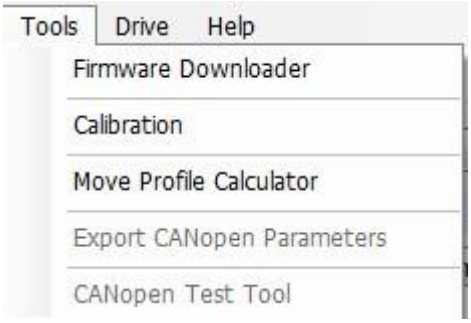
3.3.2 Configuration

In config Menu, the M Servo Suite allows you to upload and download configurations. It can also save as configuration file (.mdcfg) to your local disk and download configurations to a different drive directly from the hard disk. In addition, it can also print out the detailed configuration files.



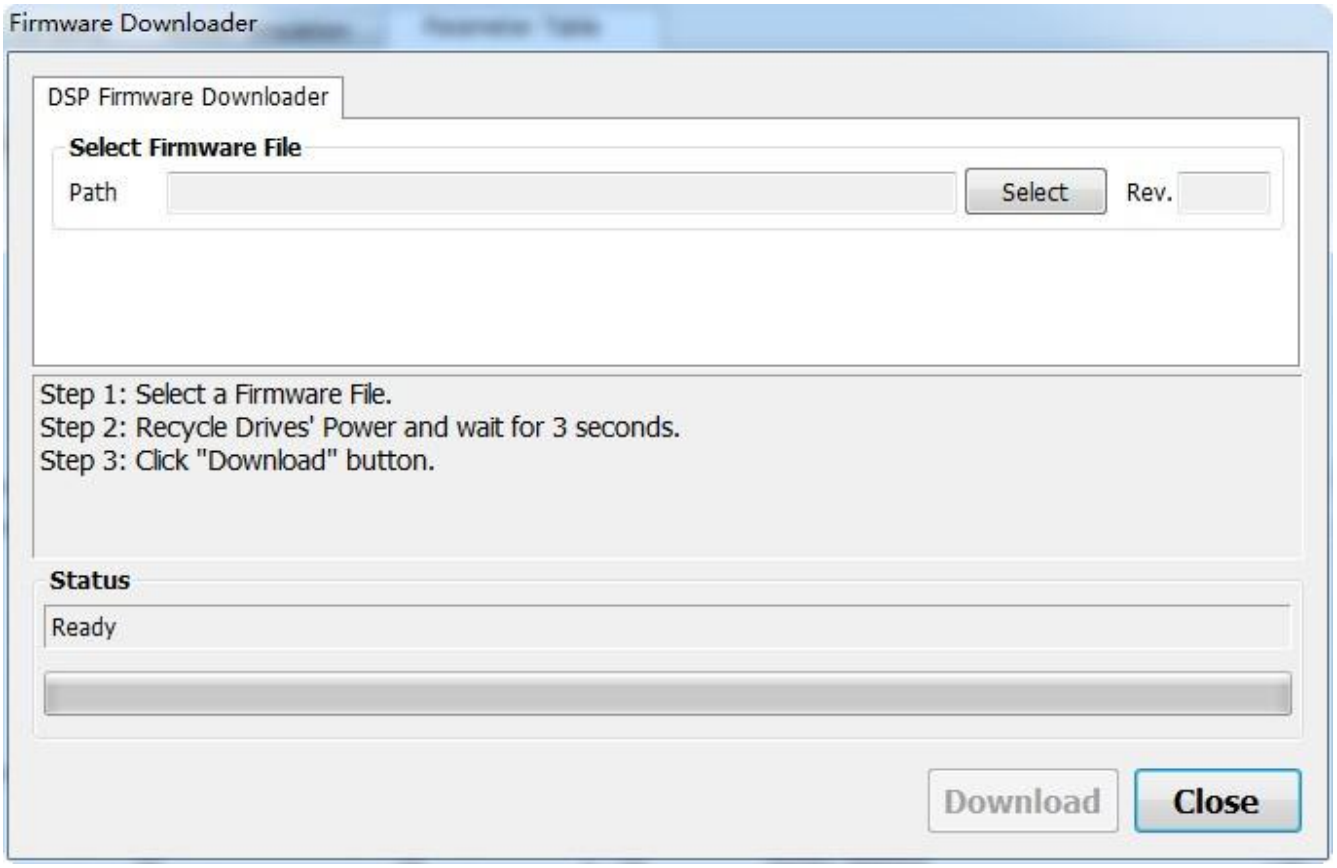
3.3.3 Tools

Tools includes Firmware Downloader, Calibration, Motion Profile calculator, Export CANopen Parameters and CANopen Test Tool, see picture below:



3.3.3.1 Firmware Downloader

Firmware Downloader is used to upgrade the drive firmware. Before upgrade please contact MOONS' to confirm that you get the proper latest firmware version to download.



Please follow the below sequence to do the firmware updates:

Step 1: Select a Firmware File

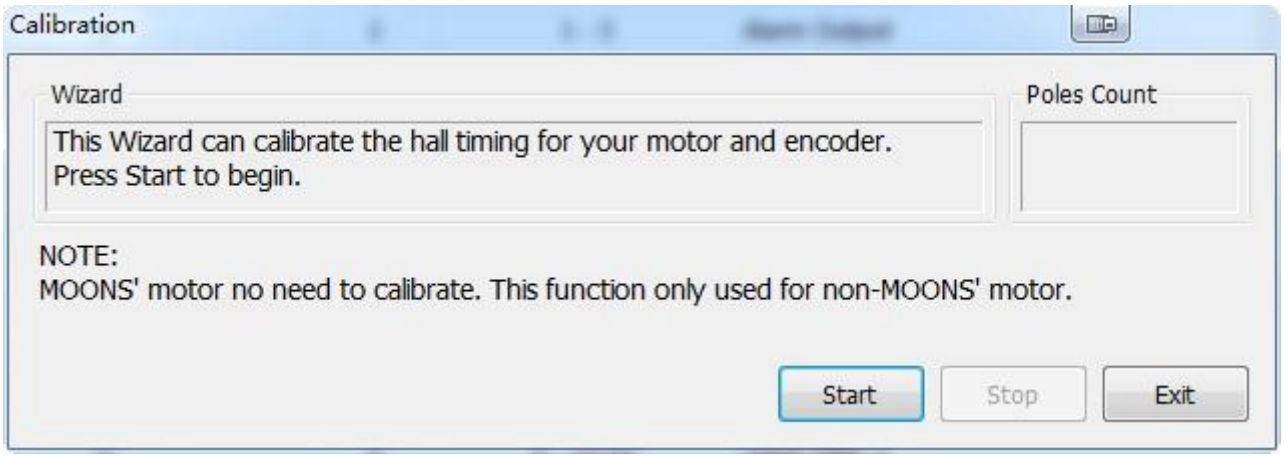
Step 2: Recycle Drive's Power and wait for 3 seconds

Step 3: Click "Download" button.

Note: MOONS' drives does not support multi axis networking firmware updates for RS485 field bus. You can only do the firmware updates for each single axis which must be offline from the network.

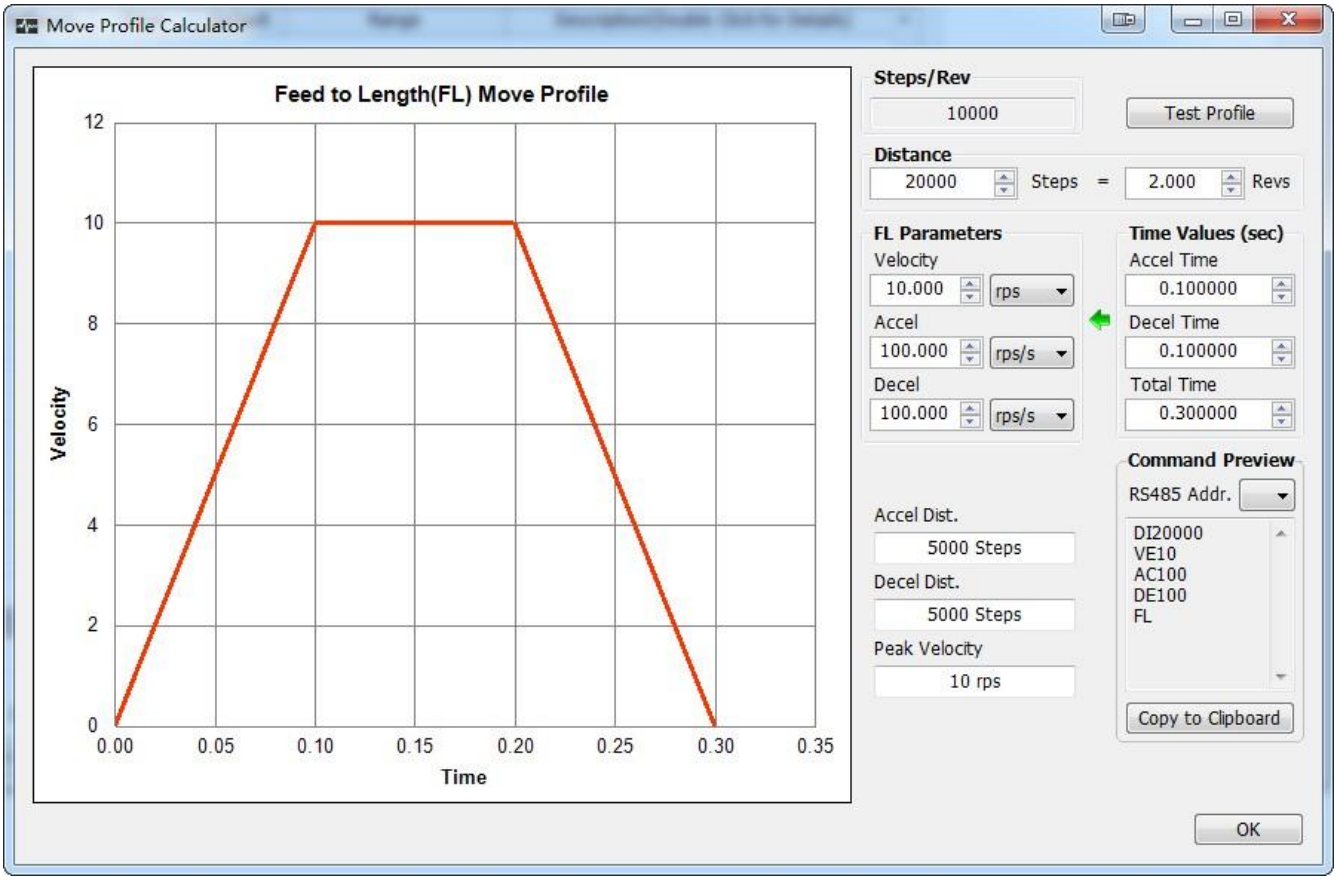
3.3.3.2 Calibration

This tool help you to calibrate the servo motor which is not made by MOONS'. In most cases, it can automatically detect your motor timing pattern and configure the drive settings for it.



3.3.3.3 Move Profile Calculator

Move Profile Calculator provides an excellent tool for the customer to simulate the assumed move profile. The motion parameters can convert between time and SCL parameters easily via click a button. When the drive is connected with the software, you can click "Test Profile" to try a move per your inputs.

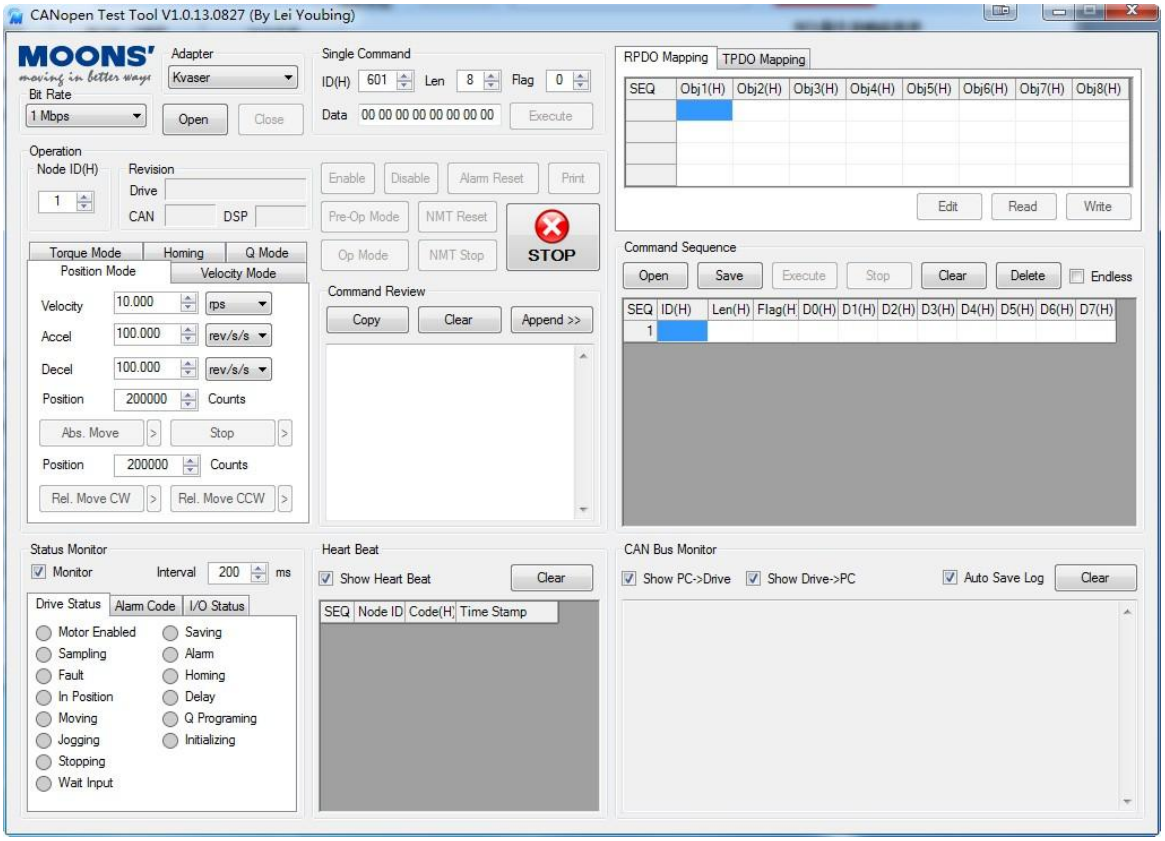


3.3.3.4 Export CANopen Parameters

After tuning is done. Export CANopen Parameters provide a tool to export the tuning parameters such as KP, KD, VP, VI and etc. and save these parameters to a text file with some specific data format which is easy for the customer to immigrate to their program. Below is a saved file example.

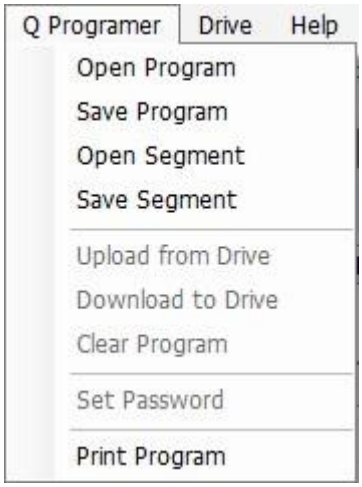
3.3.3.5 CANopen Test Tool

This provides a quick link to the installed CANopen Test Tool software.
If you have installed CANopen Test Tool, click this will launch “CANopen Test Tool” software.



3.3.4 Q program

If your drive is Q type, the Q program Menu can save driver's configurations as a configuration file (.qpr) to your local disk. It can also download Q program to a different drive directly from the hard disk. In addition, it can also print out the detailed configuration files.



3.3.5 Driver

Drive menu has the following functions:

Menu	Name	Hot keys	Description
Driver	Connect	Ctrl+R	Connect Drive
	Stop	Atl+F5	Emergency stop
	Ping		Ping for Ethernet driver
	Edit IP table		Edit IP table for Ethernet drive
	Parameter Table		Display parameter
	Script		Run script file
	Restore Factory default	Ctrl+Shift+D	Restore drive to factory default mode
	Restore tuning parameters		Restore drive's tuning default setting
	Alarm history	Ctrl+Shift+A	Check alarm history
	Misc. Settings		Settings for alarm mask, regen resistor, communication, and other settings

3.3.5.1 Connect

Connect M Servo Suite to the drive.

3.3.5.2 Ping

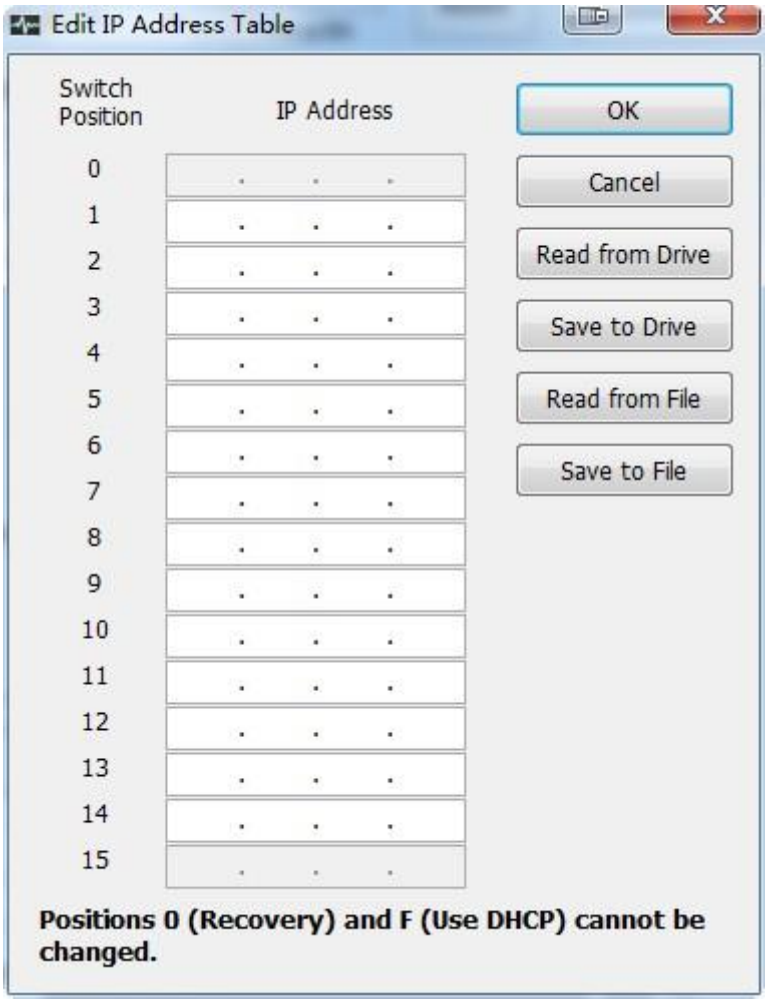
Ping Drive will verify your network configuration and ensure that the software can communicate with the drive. Click "Ping" button, the software will check drive's ARM build number and MAC ID,

3.3.5.3 IP Table

IP Table is used to edit the IP Address for the drives with Ethernet port.

For the Ethernet drive with rotary switch for IP address selection, you can input up to 14 IP address for the rotary switch position 1-E.

Note: When save the IP Address to the drive, the IP Address need to recycle the drive's power to make effect.



3.3.5.4 Parameter table

View parameter table setting values.

Step 1: Configuration | Step 2: Tuning - Sampling | Motion Simulation | Parameter Table

Open | Save | Print | Export | Upload from Drive | Download to Drive | Refresh

SEQ	Category	Command	Unit	Software	Drive	Default	Range	Description(Double Click for Details)
000	PID	KP		10000	10000	8000	0 - 32767	Global Gain 1
001	PID	KG		12000	12000	10000	0 - 32767	Global Gain 2
002	PID	KF		22500	22500	6000	0 - 32767	Proprotion Gain
003	PID	KD		16200	16200	2500	0 - 32767	Deriv Gain
004	PID	KV		25000	25000	8000	0 - 32767	Damping Gain
005	PID	KI		360	360	200	0 - 32767	Integrator Gain
006	PID	KK		22681	22681	0	0 - 32767	Inertia Feedforward Constant
007	PID	KJ		5000	5000	5000	0, 10 - 5000	Jerk Filter Frequency
008	PID	VP		15000	15000	15000	0 - 32767	Velocity Loop Proportional Gain
009	PID	VI		600	600	1000	0 - 32767	Velocity Loop Integral Gain
010	PID	KE		15000	15000	15000	0 - 32767	Deriv Filter Gain
011	PID	KC		20000	20000	25000	0 - 32767	PID Filter
012	Control Mode	CM		7	7	21	1-8,11,12,15-18,21,22,25	Main Control Mode
013	Control Mode	CN		21	21	21	1-6,8,11,12,15-18,21	Second Control Mode
014	Control Mode	PM		2	2	2	2, 5, 7, 8, 9	Power-up Mode
015	Control Mode	JM		1	1	1	1 - 2	Jog Mode
016	Current Config	GC	0.01A	0	0	0	-180 - 180	Current Command
017	Current Config	CC	A	1.800	1.800	0.500	0.000 - 1.800	Max Current
018	Current Config	CP	A	5.400	5.400	1.500	0.000 - 5.400	Peak Current
019	Current Config	HC	A	1.800	1.800	1.500	0.000 - 1.800	Current in Hard Stop Homing
020	Trajectory	VM	rps	60.000	60.000	60.000	0.025 - 100	Max Velocity
021	Trajectory	AM	rps/s	3000.000	3000.000	3000.000	0.167 - 5000	Max Accel
022	Trajectory	JS	rps	1.000	1.000	10.000	0.025 - 100	Jog Speed

3.3.5.5 Script

Use script to write SCL language and execute directly. Endless loop function allows you the run the script continuously, until stop is clicked.

If “Stop Monitor when executing” is checked, it will effectively decrease software delay on your PC

3.3.5.6 Restore Factory Default

Restore button will reset all the parameters on the drive to the default factory settings.

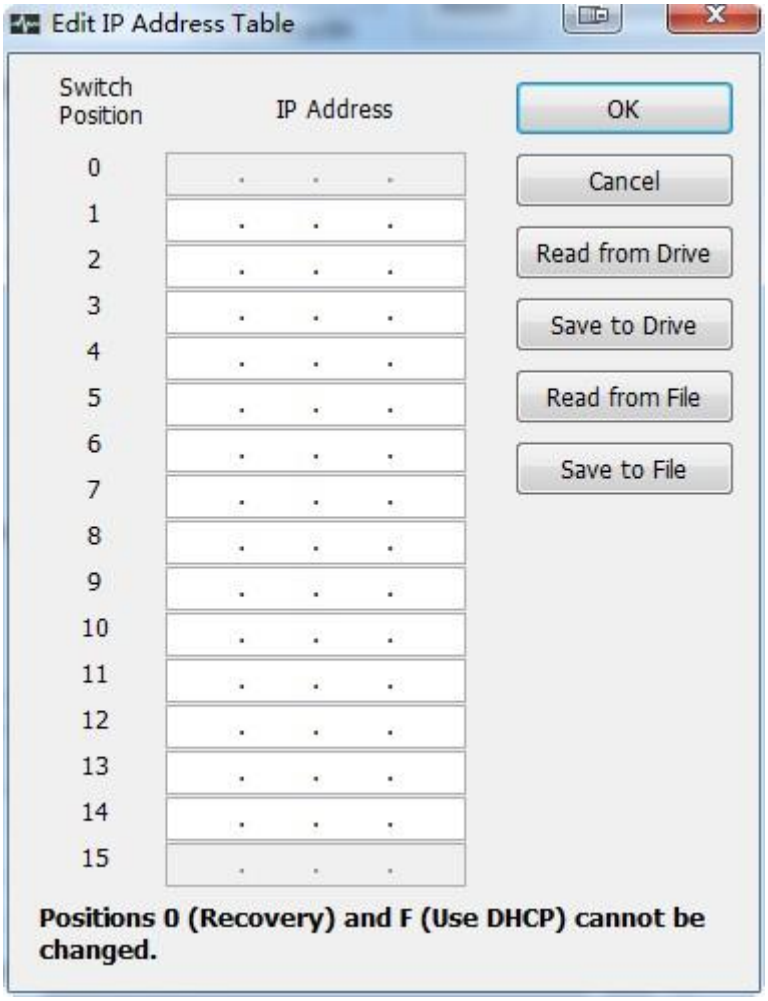
Note: This will erase all the parameters you have changed, so you may need to save them to a file first.

3.3.5.7 Restore Factory Default

Restore button will reset all the tuning parameters of the drive

3.3.5.8 Alarm History

MOONS' drive stores a log of previous alarm conditions. Each time there is an alarm, the drive stores the information of which alarms were triggered at this time. Since a fault may trigger more than one alarm condition, the drive stores all of them for reference. This information can then be extracted using M Servo Suite or the Host Language to help with drive and system problem solving. The drive stores up to 8 sets alarm conditions.

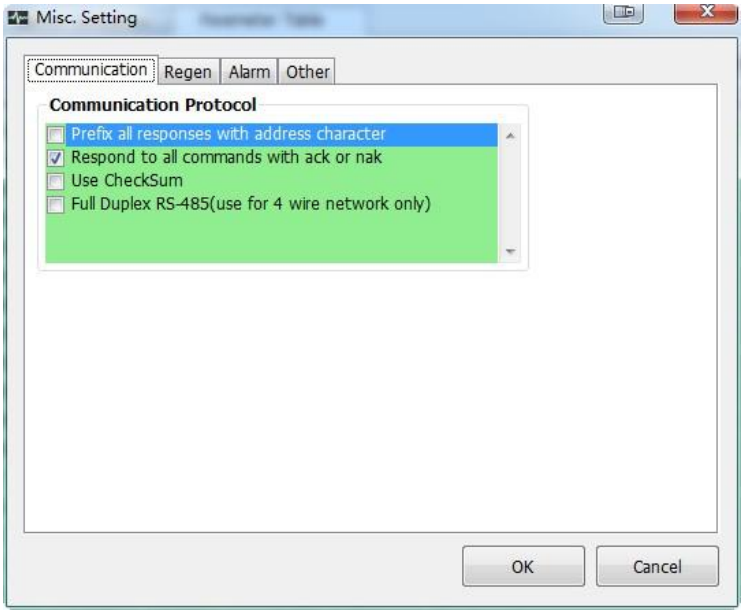


3.3.5.9 Misc. Setting

Set alarm mask, regeneration resistor, communication and other parameters

A. Communication

This page uses to setup the communication settings between the host controller and M2 Series AC Servo drive.



Prefix all responses with address character: Driver response to SCL command with address character prefix.

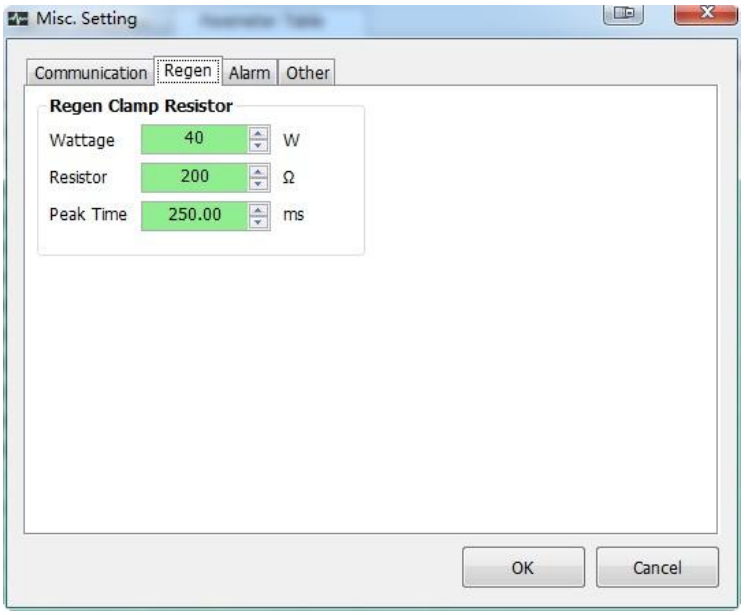
Respond to all commands with Ack or Nack: Respond to all commands with Ack or Nack

Use Checksum: Use Checksum during communication

Full Duplex RS-485: Only with 4 wire connection network

B. Regeneration Resistor

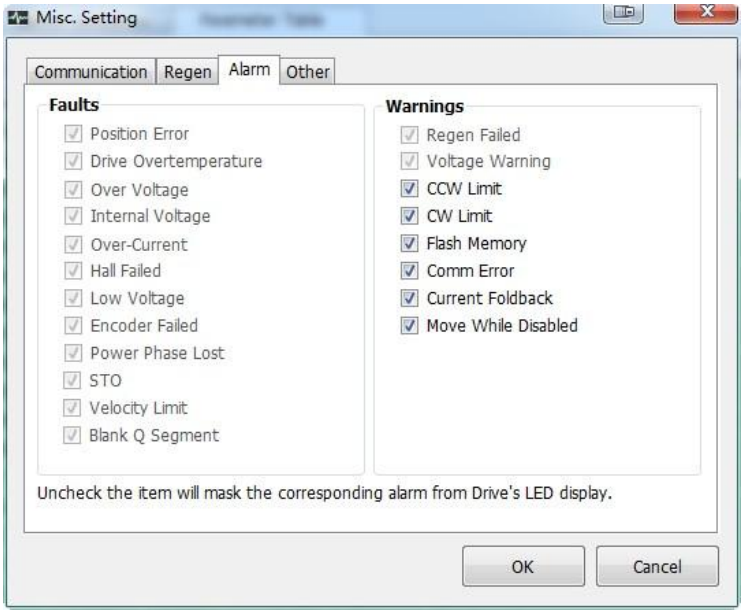
This page will help you to setup external regeneration resistor.



C. Alarm Menu

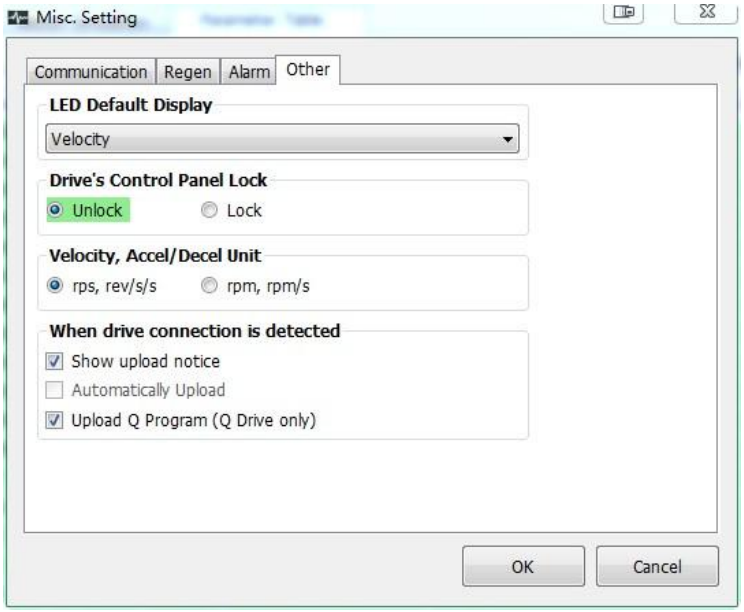
Sometimes you are puzzled and bored by some Alarms, and the Alarm is inessential for your application. In this case you can inhibit these Alarms.

Click "LED Flashing" button in the Menu, it will pop up a dialog.



Uncheck the alarms you want to inhibit, when drive encounter such alarms, the drive will not display the alarms by LED. However, the drive will still record them and stored to the alarm history for future examination.

D. Other



LED Default Display: Setting the default power-up LED Display of the drive.

Drive's Control Panel Lock: User cannot change any settings when the control panel is locked.

Velocity, Accel/Decel Unit: Units settings for velocity, acceleration and deceleration: rps，rev/s/s or rpm，rpm/s/s.

When drive is connected: Settings when connecting drive to the software controller.

3.3.6 Help

A. About

Click for the software version.

B. Help Content

Click for the software help.

3.3.7 Language

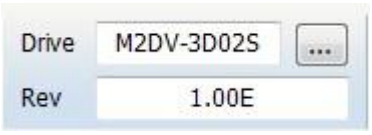
Language button has 2 language options. You can click one of them to shift the language between English and Chinese.

3.4 Tool Bar

The Tool Bar includes MOONS' Logo, Drive Model, Firmware Revision, Communication Settings, Servo Status control, Alarms, Upload & Download and Stop buttons.

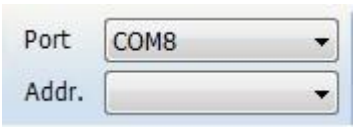
3.4.1 Drive Model

The Drive drop-down list shows all of the available M2 Series AC Servo drive model numbers.
The Revision window will display a drive's firmware version once the drive is properly connected to the PC and power is supplied.

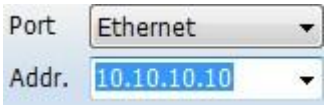


3.4.2 Communication Port

Choose the correspondent communication port for the drive before any drive configuration. For RS-485 drives, it allows you to choose the drive to connect with.



For Ethernet drives, the IP address need to be configurated.



3.4.3 Servo Status

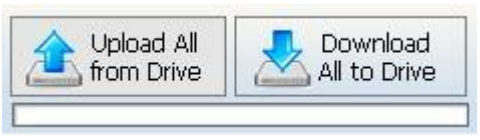
The servo enable switch uses to control the driver and motor status. When green color is shown, the drive is enabled.

Force enable allows you enable the drive when drive is connected regardless of the external enable switch status.
Alarm reset allows you to reset the alarm, when they occurs.
NOTE: Alarm can only be cleared when drive's warning or fault problems are solved.

3.4.4 Upload and download

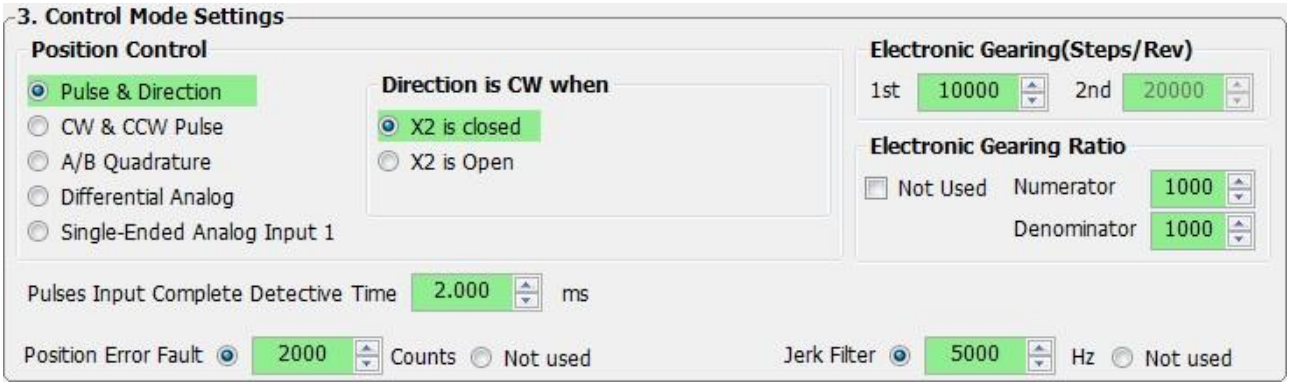
Upload lets you back-up the set up and tuning parameters from your dive into M Servo Suite software. This is useful if you want to make changes to a system that has already been tuned.

The Download command is used to restore settings from the M Servo Suite software to your drive. Use this if you make a change to a drive setting and want to transfer the information back to the drive.

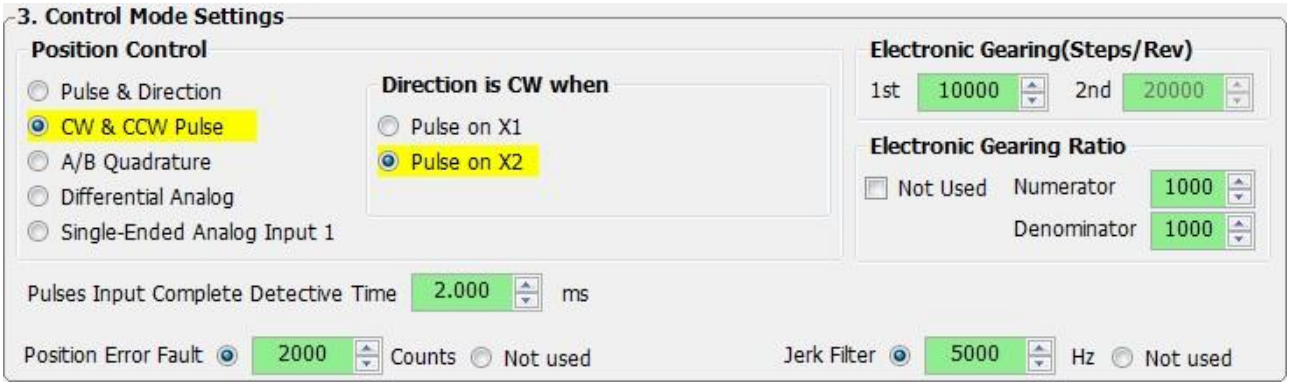


"Upload All from Drive" and "Download All to Drive" would upload or download whole project.

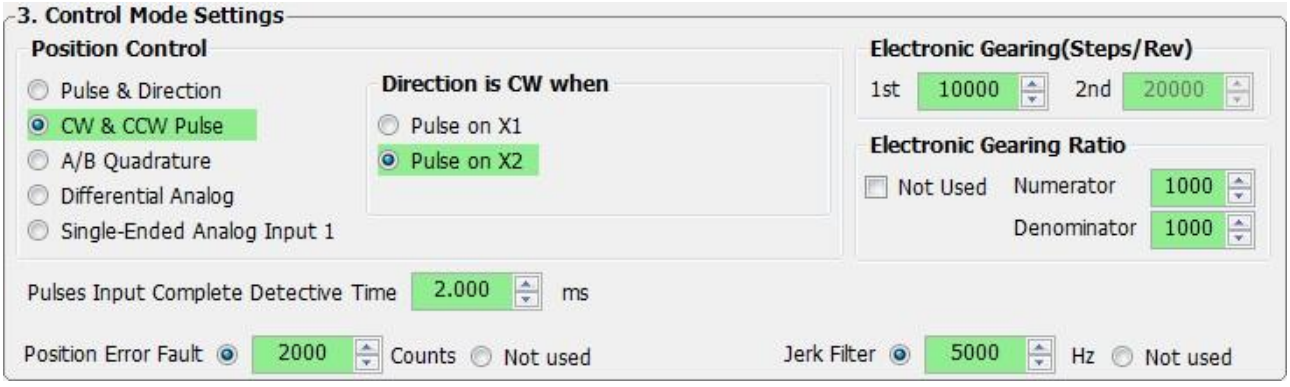
After perform an upload or download click, the background of each parameter will turn to Green color. This indicates the parameter in software and drive's setting is consistent. See below.



Then if a parameter is changed, the background of that parameter will change to Yellow color. This indicates the parameter in software and drive's setting is different. See below.



Then if a download is performed after that parameter changes, the background of that parameter will turn back to Green color. This indicates the parameter is downloaded successfully in which means the software and drive's setting is consistent. See below.



If the driver is not powered up or not connected to the software, or a single upload or download is not performed after the driver is powered up or connected to the software, the background color of parameter is transparent or white, in which means software and driver has not been synchronized (Upload or Download).

3.4.5 Stop

Stop drive's motion immediately.

4 Use M Servo Suite for Configuration

Steps for drive configuration with M servo Suite

Step 1: configuration

Configure motor information, control mode, as well as I/O settings

Step 2: Parameter tuning

Tuning the driver tuning parameters to fit your motion requirements

Step 3: Q programming

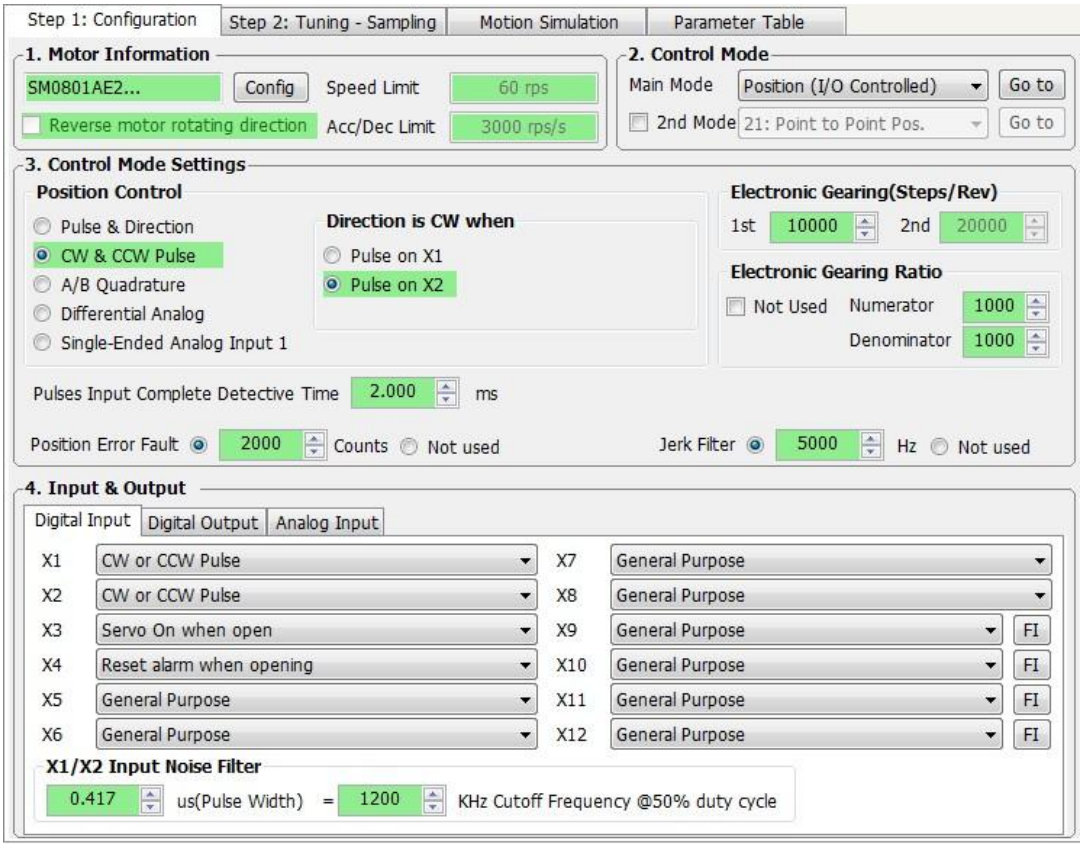
Q programming editing

Step 4: Motion simulation

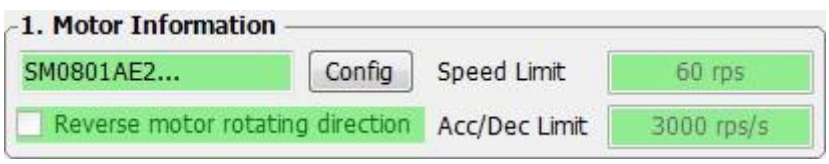
Use to simulate motion, including jog mode, P-to-P motion and homing mode.

4.1 Configuration

In this tab, you can configure drive's setting and control mode in details.



4.1.1 Motor Configuration

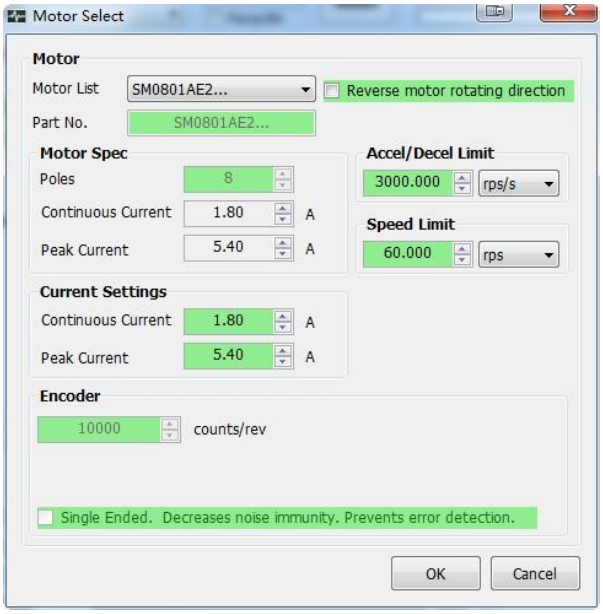


Click “config “into the details setting.

In this window different motors and maximum current, Speed limit, Acc/Dec Limit can be set.

Check box on “Reverse motor rotating direction” will reverse the default rotating direction of motor (A cycle

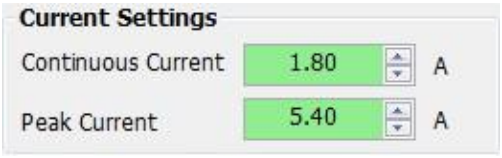
power on is necessary before the function is valid)



NOTE: Motor model number must be the same as the connected motor model number. Please DO NOT change current settings.

4.1.1.1 Current Settings

The drive current must be set to match the motor. First, determine the rated current for the motor according to the shipped M2 servo drive hardware manual.



If you are manually setting the current, type the value into the Maximum Current text box.

The M2 Servo drive provides a peak current momentarily. This will provide greater acceleration rates than would otherwise be possible. To assure reliable motor operation, the drive will automatically ramp the current down after one second so that the average current does not exceed the motor's rating. Never continuously operate a M2 servo motor above its rated current.

The peak current available varies from model to model, so check your product specifications before setting a value.

4.1.1.2 Maximum Speed

Here you can enter the maximum speed allowable in your application. If your maximum speed is set below the speed your command signal can demand, the final speed achieved will be the speed set in the Maximum Speed parameter.

Note: Maximum Speed works with Velocity Mode and Torque Mode Only.

In Pulse Input Mode these values will be limited in your controllers' software.

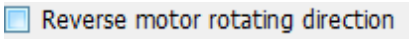
4.1.1.3 Maximum Acceleration/Deceleration Limit

This will set the maximum level of acceleration for the motor. Even if the command input tries to demand a

higher level of acceleration, the drive will only accelerate at the maximum set level.

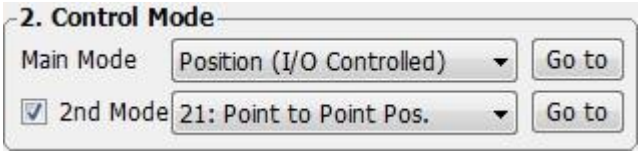
4.1.1.4 Reverse motor rotating direction

If this is checked, the motor rotating direction will be reversed without any other changes.



4.1.2 Control Mode Selection

For –S/- Q/- R/ type drivers, you can choose two types of control mode, based on your needs.



Main mode：Position（I/O Controlled）、Velocity（I/O Controlled）、SCL(Stream Command)、Torque、Position Table（-S type only）、SCL/Q(Stream Command)(-Q/-R type only)、Modbus（-R type only）。

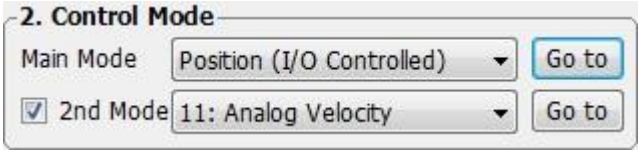
Second mode：SCL Commanded Torque、Analog Torque、Analog Torque+Dir、Analog Torque+R/S、Analog Torque+R/S+Dir、Analog Velocity、Analog Velocity+R/S、Fixed Velocity、Fixed Velocity+R/S、Fixed Velocity+CS、Fixed Velocity+R/S+CS、Point to Point Position。

Control mode switch

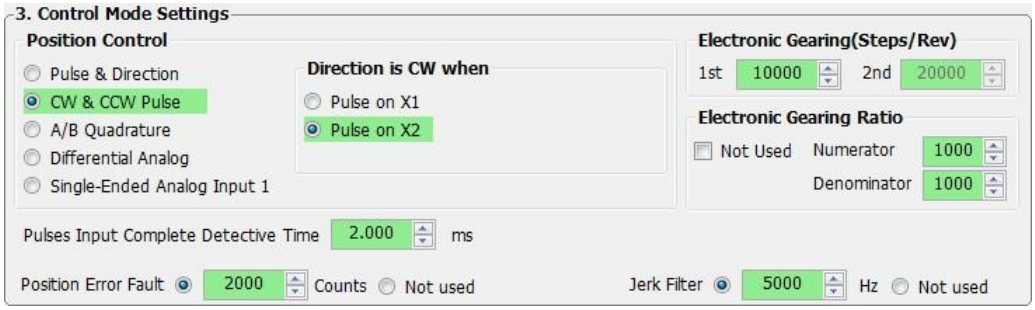
When second mode is allow, input X8 is used to switch between two control modes. In the software, you click “Go to” for mode selection.

For example:

1) Set main mode as Position（I/O Controlled）, and second mode ads Analog Velocity。



2) Click on “Go to” beside the main mode, the settings will be for the main mode

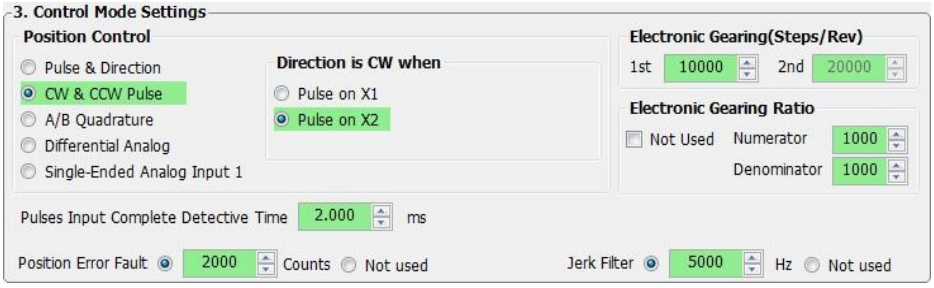


3) Click on “Go to” beside the 2nd mode, the settings will be for the secondary mode

4.1.3 Control Mode Configuration

4.1.3.1 Position Mode (I/O Controlled)

Position mode has five control inputs: Pulse & Direction, CW&CCW Pulse, A/B Quadrature, Differential Analog, and Single-Ended Analog Input 1.



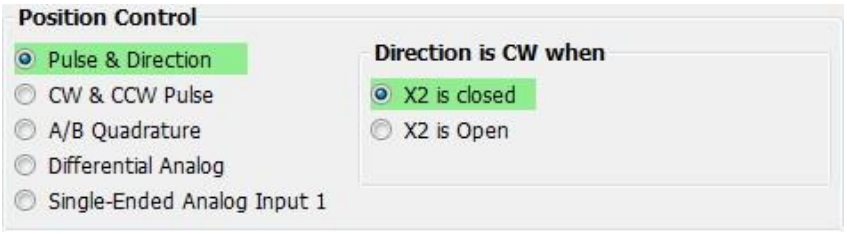
4.1.3.2 Position Control - Pulse Input

Pulse Input Mode is for systems whereby the position of the motor is determined by a digital input signal in the form of pulses.

The three modes available are:

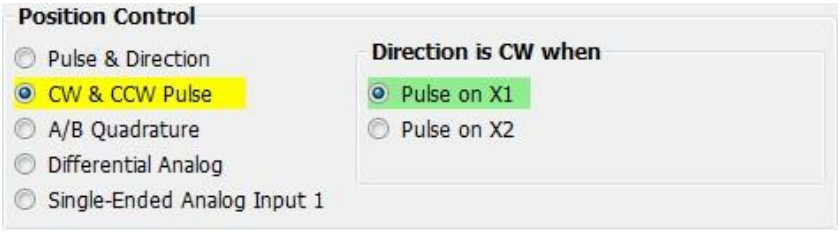
Pulse and Direction.

Accepts a signal such as that generated by a controller. With this mode the frequency of the pulses fed into one input determines the speed, the direction of rotation is determined by a signal fed into another input. You can configure whether X2 signal closed or open represents clockwise motion.



CW and CCW Pulse.

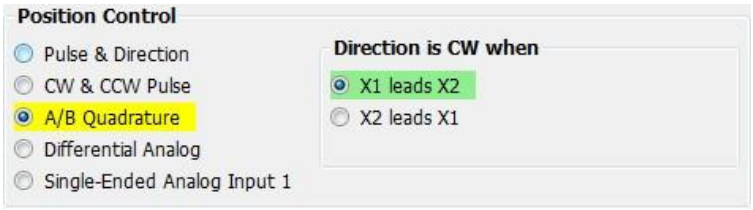
The motor will move CW or CCW depending on which input the pulse is fed into. The drive has two inputs allocated to this feature, pulses fed into one input will generate CW motion, and pulses fed into the other input will generate CCW motion.



A & B Quadrature.

Sometimes called “Slave Mode”. The motor will move according to signals that are fed to the drive from a

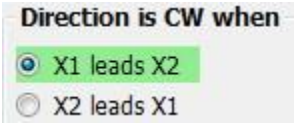
master encoder. This encoder can be mounted on a shaft on the machine or it can be another motor in the system. Using quadrature input mode it is possible for a number of motors to be “daisy chained” together with the encoder output signal from each drive being fed into the next.



For all the Pulse Input modes you will need to determine a value to enter into the Electronic Gearing Box. An explanation on how to do this is given in the next section.

Direction is CW when

CW direction is determined by the polarity of input X2 which requires to be set in priority.



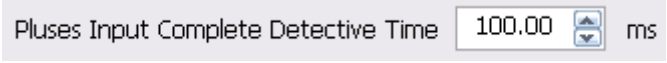
Jerk Filter

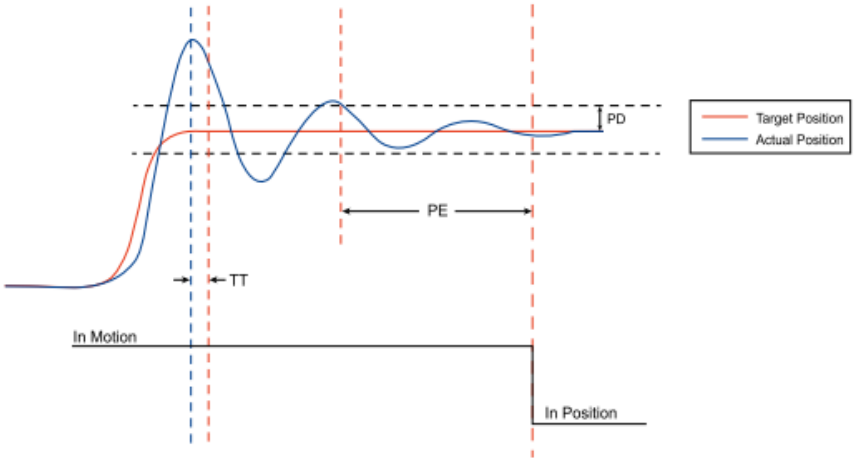
Jerk Filter technology on Step/Dir inputs will lower the mechanical transition between motor and equipment structure. The feature will bring smooth move on motor and significant lower the mechanical friction.

1) Smaller value gives more smooth performance.
2) Smoothing Filter technology will involve some time delay for reaction, however it doesn’t affect the positioning accuracy.

Pulse Input Complete Detective Time

Set a period of time, if the drive doesn’t recevie any pulse during this period of time, the target position is determined, the parameter is used for determine whether the motor is in position or not. See detailed information on TT command from Host Command Reference Manual.





4.1.3.3 Position Control - Analog

Positioning mode using an analog input causes the motor to position the motor relative to the analog input value.

Analog positioning allows you to move the motor a relative distance according to the value of an analog input. For example the below configuration would move the motor +/-8000 counts from its current position according to the voltage applied, e.g. a signal of +5 volts would move the motor 8000 counts clockwise. There is also option for an offset voltage and a dead-band. The offset can be used to offset the position in case the 0 volt signal from your analog command does not represent zero position on your application.

4.1.4 Velocity Mode (I/O Controlled)

Velocity mode means that the drive uses the command input signal to set the speed that the motor will run at.

Some operation are needed in velocity mode.

Velocity Control Type: Speed Only (without position error) or Position over time (With Position error che**ck)**

Speed Only: Only work at Velocity loop, Without Position loop. The Velocity loop gains, Velocity Proportional gain and Velocity Integral gain need to be set.

Position over Time: Operating at position loop with position error check.

Velocity Control By: Choose the velocity control by Fix speed or analog input.

Accel: Set the acceleration in velocity mode.

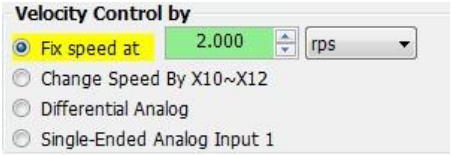
Decel: Set the deceleration in velocity mode.

Four Velocity Mode can be used;

1) Fix speed
2) Change Speed By X10~X12
3) Differential Analog
4) Single-Ended Analog Input 1

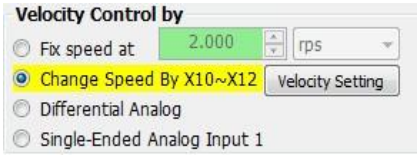
4.1.4.1 Fixed Speed

Motor will run at fixed speed, run/stop and direction are controlled by external input.

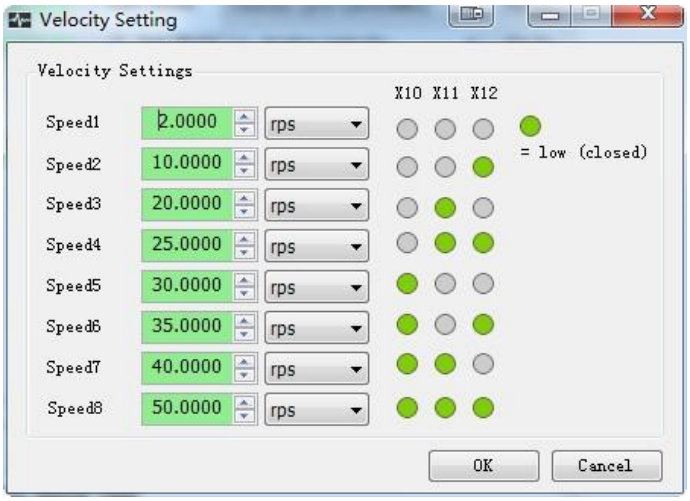


4.1.4.2 Change speed level by X10~X12

Change speed by X10~X12,



With different X10/X11/X12 settings, 8 different levels can be set. Click on ***Velocity Setting*** button to detailed settings.

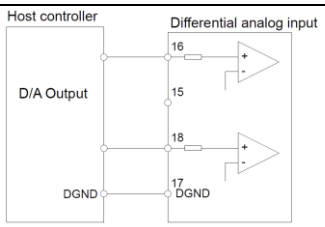


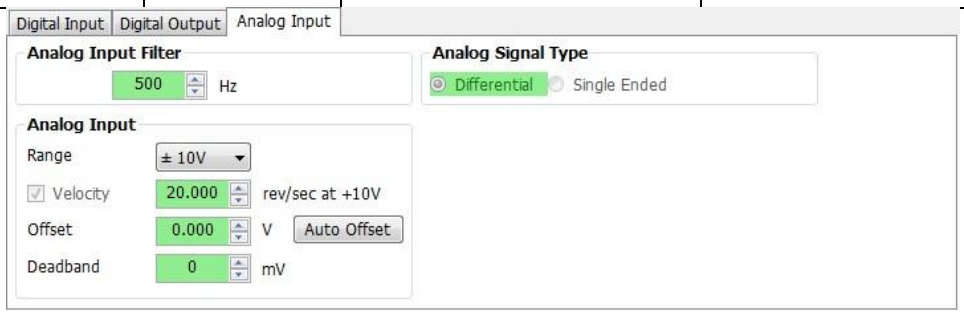
4.1.4.3 Analog Velocity Mode

Analog velocity mode has two input types, Differential Analog and Single-Ended Analog Input 1.

1) Differential Analog

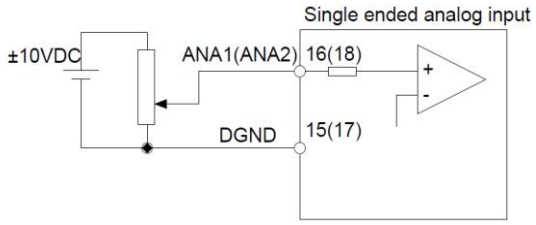
Use ANA1 and ANA2 to be a Differential analog input. Differential signal can improve the anti-interference ability.

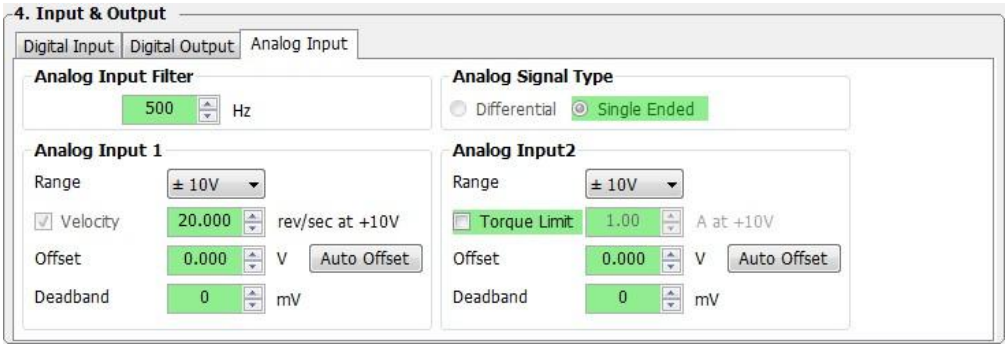
Input Type	Type	Pin NO.	Functions	
Analog Input	ANA1	16	Differential Analog Input	
	ANA2	18		
	DGND	15	Digital Ground	



2) Single-Ended Analog Input 1

ANA1 is a single-ended analog input for Analog velocity mode.



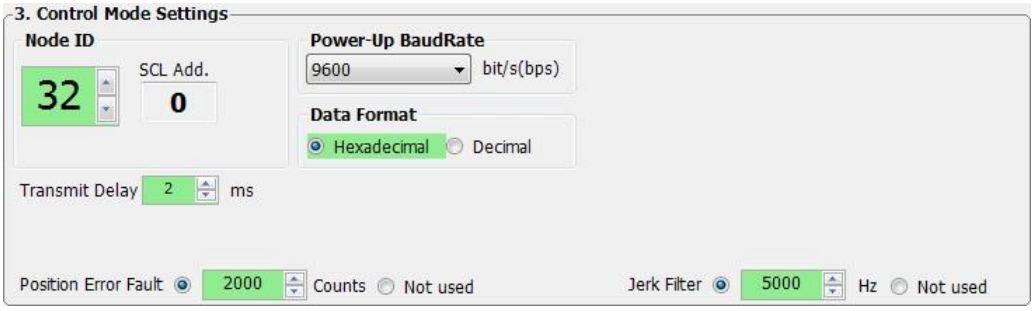


4.1.5 SCL /Q Mode (Stream Command/Stand Alone)

4.1.5.1 SCL

SCL or serial command language, was developed by MOONS to give users a simple way to control a motor drive via a serial port. This eliminates the need for separate motion controllers or to supply control signals, like Pulse & Direction, to your step and servo motor drives. It also provides an easy way to interface to a variety of other industrial devices like PLCs, industrial computers, and HMIs, which most often have standard or optional serial ports for communicating to other devices.

SCL is MOONS''s host command language for applications that require the drives to be sent instructions by a host controller in real time. With SCL, the drives can be operated in both RS-232 and RS-485 mode, the RS-485 option allows you to have multi-axis multi-drop applications with the drives “daisy chained” on one serial link. When this option is selected you will need to set an address for each drive you are working with. Refer to Setting the Address in the next section.



Node ID

In SCL mode with RS-485 communications you will need to set the address for each drive in your system. Simply select the address character and perform a download, in this way up to 32 drives can be connected together on a single serial link.

Transmit delay

This sets up the transmit delay for communications between host controller and the drive. This is highly necessary for 2 wire configurations for RS-485 communication. The host must disable its transmitter before it can receive data. This must be done quickly before a drive begins to answer a query.

Baud rate

At power up a drive will send its power-up packet detected after 1 second and the drive is configured for SCL

or Q operation (see PM command) the drive will set the baud rate according to the value stored in the Baud Rate NV parameter. This parameter will not effect immediately, it will only effect at next drive power up.

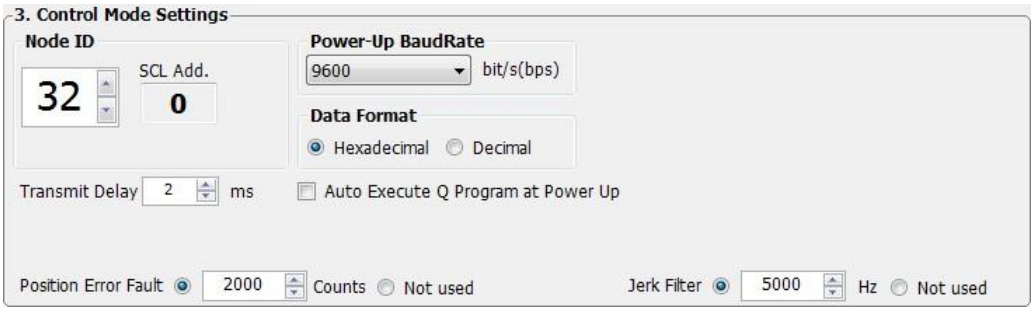
Data format

To setup data transmit type between Hexadecimal and Decimal.

4.1.5.2 Q Program

The use of SCL commands with MOONS' dates back many years. A few years ago a new control platform was created that expanded the use of SCL commands and allowed users to create stored programs with SCL commands. These programs could be saved in a drive's non-volatile memory, and the drive could run these programs stand-alone, or without a permanent connection to the host. This expansion of SCL's capabilities was called Q, and since that time MOONS' has continued to expand the offering of drives with the Q motion controller built in. By combining the ability to run a sophisticated, single-axis motion control program stand-alone and the ability to communicate serially to a host device, Q drives offer a high level of flexibility and functionality to the machine designer and system integrator. The characteristic as follows;

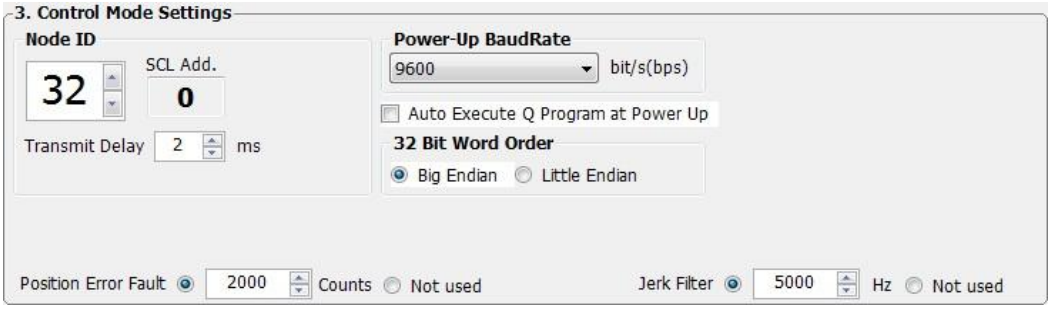
Single-Axis motion control
Stand Alone
Multi-task
Conditional Processing
Math Calculation
Data register manipulation



Auto Execute Q Program at Power Up

If this is checked, the drive will execute stored Q program from segment 1 automatically at power up.

4.1.6 Modbus/RTU



Node ID

In the network system, each drive requires a unique drive address. Only the drive with the matching address will responded to the host command. In Modbus network, address "0" is the broadcast address. It cannot be used for individual drive's address. Modbus RTU/ASCII can set drive address from 1 to 32. SCL address is ASCII code. The relationship between Modbus Node ID and SCL address is as below table.

Node ID	1	2	3	4	5	6	7	8
SCL Address	1	2	3	4	5	6	7	8
Node ID	9	10	11	12	13	14	15	16

SCL Address	9	:	;	<	=	>	?	@
Node ID	17	18	19	20	21	22	23	24
SCL Address	!	"	#	$	%	&	'	(
Node ID	25	26	27	28	29	30	31	32
SCL Address	)	*	+	,	-	.	/	0

Auto Execute Q Program at Power Up

If this is checked, the drive will execute stored Q program from segment 1 automatically at power up.

32 bit word order

Big-endian: The most significant byte (MSB) value is stored at the memory location with the lowest address; the next byte value in significance is stored at the following memory location and so on. This is akin to Left-to-Right reading in hexadecimal order.

Little-endian: The most significant byte (MSB) value is stored at the memory location with the highest address; the next byte value in significance is stored at the following memory location and so on. This is akin to Left-to-Right reading in hexadecimal order.

4.1.7 Torque Mode

When the drive is set up for Torque mode, it allows you to define the current that will be delivered and thus the torque generated by the motor and the direction it will rotate. In this mode the speed that the motor runs at will depend on the load applied to the motor.

WARNING - If the motor is not connected to the load or has no load applied, downloading this mode with a command signal may cause the motor to accelerate to high speed.

Torque mode has two control type, Analog and SCL Commanded.

4.1.7.1 Analog

Analog Torque mode has two analog input type, single-ended input 2 and differential.

1) Differential Analog Input

There are four settings that are required for getting the analog inputs to control the desired mode output:

1. Range – ±10V

2. Current– Establishes a gain value that scales the output to the input. For example in Current Mode (Torque mode), if “Current” is set to 1, a 10 volt input will apply 1 amps to the motor. A 2 volt input will apply 0.2 amps to the motor.

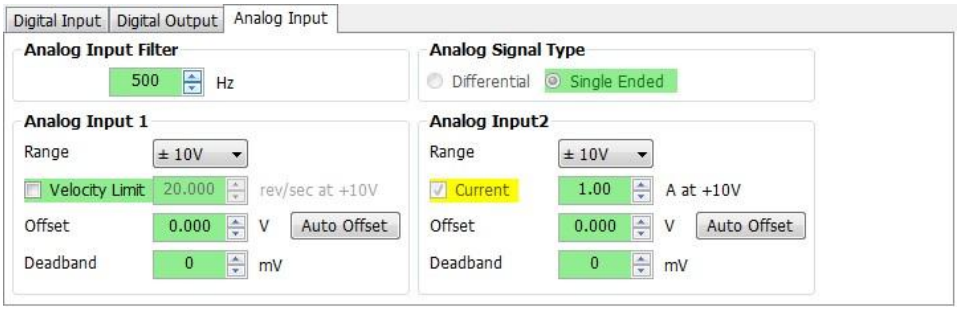
3. Offset – Sets an offset value to the input that can null out a voltage bias or it can shift the input voltage value as needed. Often in analog systems it is very difficult to get a true “0” value. Using the offset feature

allows adjusting out any unwanted offsets that disturb the desire for a true 0 volt input from an external controller. The “Auto Offset” function can automatically detect and correct voltage biases on the input. Click the button and follow the instruction to accomplish this task.

4. Dead band – Inserts a voltage region where the input is seen as “0”. Because of the sometime imprecise nature of analog signals and inputs there may be a need to create a “Dead” zone where the analog input has no effect on the output. This is normally needed around the “0” input. For example, when using a Joystick to operate the motor the user may not want any torque output when the Joystick is at its “Null” position. Most Joysticks are not that precise and may still output a small voltage, adding the dead band can eliminate the effect of the small voltage.

2) Single-ended Analog Input 2

ANA2 is a single-ended analog input for Analog Torque mode.

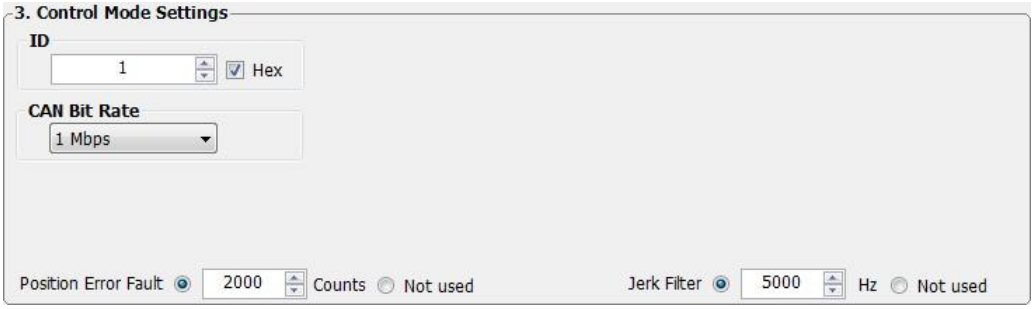


4.1.7.2 SCL Commanded

SCL Commanded control type need to send SCL command GC to control the motor’s output torque.

4.1.8 CANopen

CANopen is a communication field bus standardized by the CAN in Automation Group (CiA). MOONS' drives are compliant to CiA 301 and CiA 402 and use the CAN 2.0B passive physical layer. Detailed information on the MOONS’ CANopen implementation can be found on our website.



Node ID

In the CANopen network, each of the drive needs to have a unique NODE-ID. CANopen node ID address is represented are 7 bits binary numbers, range from 1~127 and in hexadecimal 0x01~0x7F.

CAN Bit Rated

MOONS’ CANopen drive can support 8 CAN communication bit rated.

CAN Bit Rated
1Mbps

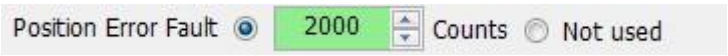
800Kbps
500Kbps
250Kbps
125Kbps
50Kbps
25Kbps
12.5Kbps

4.1.9 Positioning Error Fault & Electronic Gearing

4.1.9.1 Positioning Error Fault

Positioning error is the difference, in encoder counts, between the actual position and the commanded position of the motor. A small amount of positioning error is a normal part of a servo system. But sometimes the unexpected can happen. A wire might break, a sensor could fail or the motor may encounter a physical obstruction. You might even one day forget to set up and tune a drive before installing it into a system. In all of these cases, you'll want to know that something is wrong as soon as possible and without damaging anything. For this reason, the servo drives include a position error fault limit. Anytime the position error (as reported by the encoder) exceeds this limit, the drive cuts power to the motor.

You can set the fault limit to as little as 10 encoder counts, or as much as 32000. When you're first tuning the system, you should set this value high or Not Used so that the drive doesn't shut down as you experiment with tuning parameters. Once the drive is properly tuned and you know how much error to expect during normal operation, you can set an appropriate fault limit. For example: set Quick Tuner's scope to plot position error. Execute some aggressive sample moves, using the maximum speed and acceleration that you plan to use in your application. If the maximum position error is, say, 50 counts, then you could safely set the fault limit at 100.

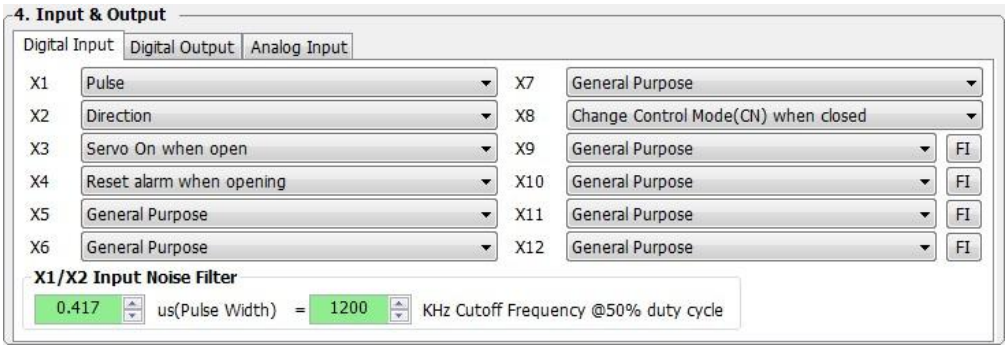


4.2 I/O Configuration

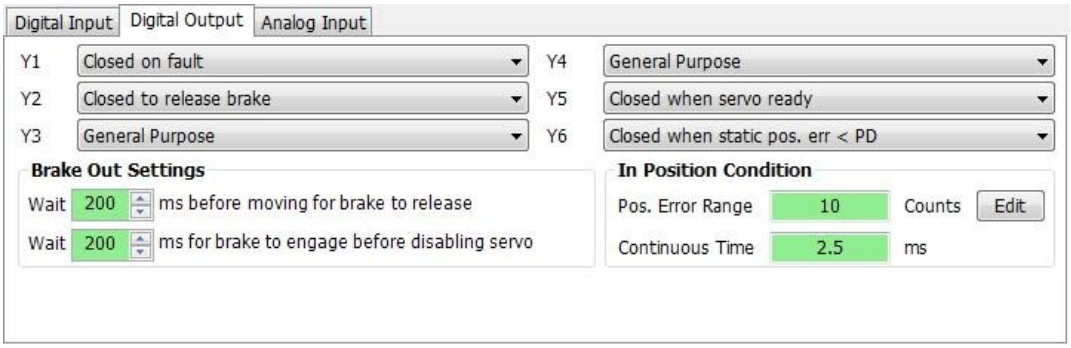
I/O Configuration includes Digital I/O configuration and Analog Input configuration.

4.2.1 Digital I/O Configuration

Digital I/O configuration is to configure the digital inputs(X) and digital outputs(Y).Please refer to the M2 User Manual for details.



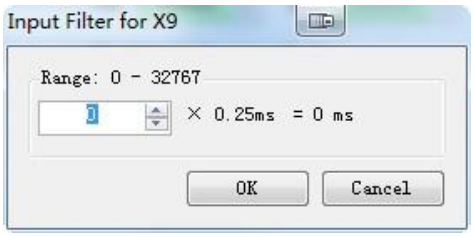
Digital input X



Digital output Y

4.2.1.1 FI Input filter

Applies a digital filter to the given input. The digital input must be at the same level for the time period specified by the FI command before the input state is updated. For example, if the time value is set to 100 the input must remain high for 100 processor cycles before high is updated as the input state. One processor cycle is 250µsec on M2 servo dirve. A value of “0” disables the filter.



4.2.1.2 Input Noise Filter

Input Noise Filter acts as a low-pass filter, rejecting noise above the specified frequency. Set the Pulse Width, the software will calculate the frequency.

4.2.2 I/O Functions

4.2.2.1 Input

Signal	Symbol	Pin NO.	Details
X1	X1+	3	This input has three functions:
			● Accept STEP pulse input such as STEP signals, CW pulse, A pulse in
			Position mode.
	X1-	4	● Run/Stop input in torque or velocity mode.
			● General purpose input.
X2	X2+	5	This input has three functions:
			● Accept STEP pulse input such as Direction signals, CCW pulse,B pulse in position mode.
	X2-	6	● Direction input in torque or velocity mode.
			● General purpose input.
X3	X3+	29	● Enable/Disable input.
	X3-	31	● General purpose input.
X4	X4+	35	● Alarm Reset Input, used to reset drive alarm.
	X4-	34	● General purpose input.
X5	X5+	8	● Limit Sensor Input.
	X5-	2	● General purpose input.
X6	X6+	9	● Limit Sensor Input.
	X6-	1	● General purpose input.
X7	X7+	39	● Gain Select Input in all control mode.
	X7-	38	● General purpose input.
X8	X8+	12	● Switch Control mode between main mode and second mode.
	X8-	32	● General purpose input.
X9	X9	26	● Dividing Switch, change the pulses per revolution for electronic Gearing.
			● General purpose input.
X10	X10	27	● Pulse Inhibited Input. Ignore the pulse input when this input is activated
			in position mode.
			● Speed Selecting Input 1 in change Speed mode.
			● General purpose input.
X11	X11	28	● Speed Selecting Input 2 in change Speed mode.
			● General purpose input.
X12	X12	30	● Speed Selecting Input 3 in change Speed mode.
			● General purpose input.

4.2.2.2 Output

Signal	Symbol	Pin NO.	Details
Y1	Y1+	37	This output has two functions: • Alarm Output. • General purpose output.
	Y1-	36	
Y2	Y2+	11	This output has two functions: • Motor brake control output. • General purpose output.
	Y2-	10	
Y3	Y3+	42	• Torque Reached Output.
	Y3-	33	• General purpose output.
Y4	Y4+	43	• Moving signal output, output signal when dynamic position error less than set value in position mode. • Velocity reach output. Output signal when actual speed is same as the target speed and the speed ripple less than ripple range.
	Y4-	33	• General purpose output.
Y5	Y5+	40	• Servo ready output. Output servo ready signal when the drive is ready to be controlled and without alarm.
	Y5-	41	• General purpose output.
Y6	Y6+	14	• In position signal output, output signal when in position, and the position error less than set value in position mode. • Tach out output. Tach output, produces pulses relative to the motor position with configurable resolution. • General purpose output.

4.2.3 Analog Input

4.2.3.1 Analog Input Filter

The analog input filter sets the frequency in Hertz of the roll off point of a single pole low pass filter. When using any of the Analog Input modes, this filter can be used to reduce the effects of analog noise on the mode of operation.

4.2.3.2 Analog Input Settings

1. Range –±10V.

2. Offset – Sets an offset value to the input that can null out a voltage bias or it can shift the input voltage value as needed. Often in analog systems it is very difficult to get a true “0” value. Using the offset feature allows adjusting out any unwanted offsets that disturb the desire for a true 0 volt input from an external controller. The “Auto Offset” function can automatically detect and correct voltage biases on the input. Click the button and follow the instruction to accomplish this task.

3. Dead band – Inserts a voltage region where the input is seen as “0”. Because of the sometime imprecise nature of analog signals and inputs there may be a need to create a “Dead” zone where the analog input has no effect on the output. This is normally needed around the “0” input. For example, when using a Joystick to operate the motor the user may not want any torque output when the Joystick is at its “Null” position. Most Joysticks are not that precise and may still output a small voltage, adding the dead band can eliminate the effect of the small voltage.

5 Step 2: Tuning - Sampling

Like most modern servo motors, ours employ sophisticated algorithms and electronics for controlling the torque, velocity and position of the motor and load.

Sensors are used to tell the drive what the motor is doing. That way, the drive can continuously alter the voltage and current applied to the motor until the motor does what you want. This is called “closed loop control.”

One of the loops controls the amount of current in the motor. This circuit requires no adjustment other than specifying the maximum current the motor can handle without overheating.

The PID loop compares the intended motor position to the actual motor position as reported by the encoder. The difference is called error, and the PID loop acts on this error in three ways: the Proportional term, the

Integral term and the Derivative term. Accelerate feedforward term is also added to achieve greater system control.

5.1 Servo Gain Parameters Tuning

Servo gain parameters tuning is used to optimize the servo system overall performance, reduce system response time. Servo tuning allows servo motor to execute host control command more precisely and maximize its system potential. Therefore, it is highly recommend tuning the servo system parameters before your system real operation. (Figure 6-1)

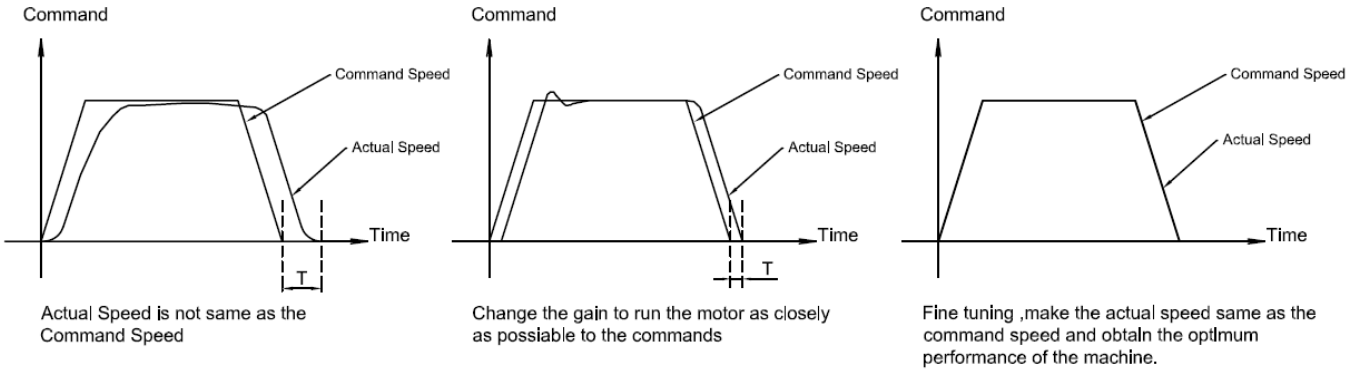


Figure 6-1

The PID loop compares the intended motor position to the actual motor position as reported by the encoder. The difference is called error, and the PID loop acts on this error in three ways: Global gain (KP), Integrator Gain (KI), Derivative gain (KD). In addition to the PID, M2 series drive add a number of extra gain terms to enable greater system control, including: position loop gain (KF), Damping gain(KV), Inertia feed forward gain constant(KK),Follow Factor(KL) ,Derivative filter gain(KE), and PID filter(KC).

In general terms, for high stiffness mechanical system increase servo gain parameters will improve its response time. On the other hand, for lower mechanical system stiffness increase servo gain parameters can potentially causing system vibrations, and in turn the system response time will be worse.

5.1.1 Gain Parameter Introduction

Global gain (KP): This parameter is the primary gain term for minimizing the position error. It defines the system stiffness. Larger KP value means higher stiffness, and fast response. However, if gain value is too high, it will leads to vibration. Value ranges from 6000 to 16000 is commonly used. In general cases please use default parameter values.

Position loop gain (KF): This parameter is the primary gain term for minimizing the position error. Increase of KF will increase stiffness and reduce in position time duration. However, it might cause system vibration if gain is too large.

Derivative gain (KD): This parameter is used to damp low speed oscillations, and increase system smoothness.

Integrator gain (KI): This parameter minimizes (or may even eliminate) position errors especially when motor is in holding position.

Damping gain (KV): KV minimizes the velocity error, and vibrations in position control mode.

Inertia Feedforward Constant (KK): KK improves acceleration control by compensating for the load inertia.

Follow Factor (KL): Higher value will reduce system noise, eliminate the overshoot, but it will reduce the system dynamic following performance. Lower value will raise system stiffness, but will cause system noise probably.

Derivative Filter Gain (KE): The differential control parameters filter frequency. The filter is a simple one-pole, low-pass filter intended for attenuating high frequency oscillations. The value is a constant that must be calculated from the desired roll off frequency.

PID Filter gain (KC): The servo control overall filter frequency. The filter is a simple one-pole, low-pass filter intended for attenuating high frequency oscillations. The value is a constant

that must be calculated from the desired roll off frequency.

Among all the parameter, changes for KP, KE, and KC are NOT recommended after system configuration. Therefore, parameter tunings are more based on KF, KD, KV, KI, KL and KK.

5.2 Use M Servo Suite to Auto-tuning.

M2 servo system can accomplish real time response to the dynamic feedback of the load and optimize parameters tuning gain automatically. This auto tuning function can greatly save debugging time and simplify the debugging procedure. These all can be done by the PC based software (M servo suite) in only a few minutes.

NOTE: Auto Tuning function can operate with load installed.

5.2.1 Step 1: Select Motor

Before using the auto tuning function, please make sure the motor configurations is correct.

1: In M servo Suite "**Configuration**" page-----"**Motor Information**" click on "**Config**" (Figure 6-2)

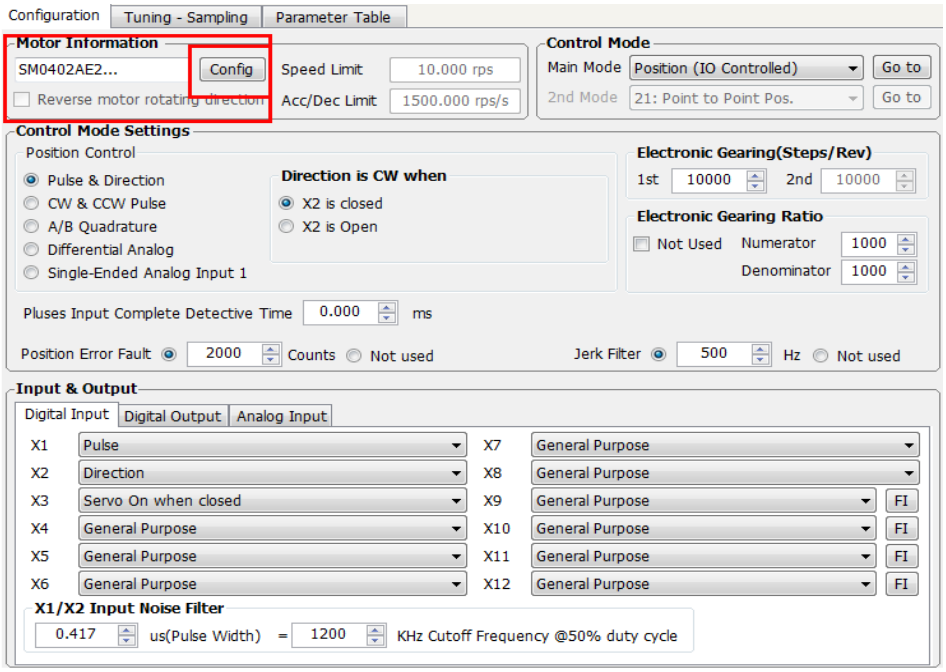


Figure 6-2

2: In the pop-up menu, click on motor list to choose the correspondent motor number (Figure 6-3) and click "**OK**"

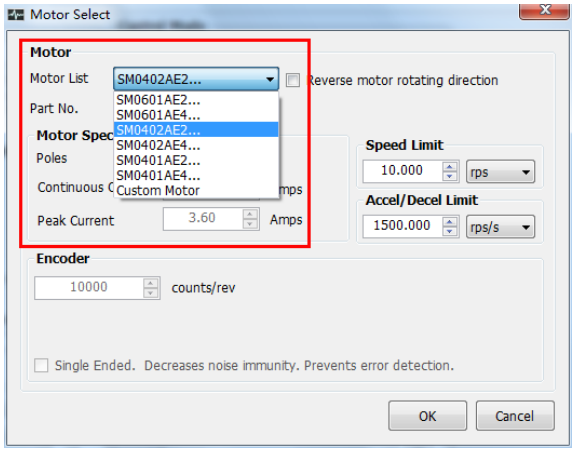


Figure 6-3

NOTE: Please refer to M2 Series AC servo User manual Chapter 2.3 Servo Motor Model Introduction for motor selection instructions.

5.2.2 Step 2: Software Position Limit setting

Software Position Limit function: it uses software to setup the position limits for software tuning functions. Position limit ensures that the motor will ONLY rotates between the CCW and CW limits, to prevent any damages and accidents to the system.

NOTE: The software Position Limit will ONLY be effective at current power-up operation, and it will not be saved at next drive power up. Therefore, Please DO NOT use it as really operation system position limit.

In M servo Suite "**Tuning- Sampling**" page, under "**Limit**" to setup software position limit. If software position limit is not required, Please click "**Clear Limit**" and then start "**Auto-Tune**". (Figure 6-4).

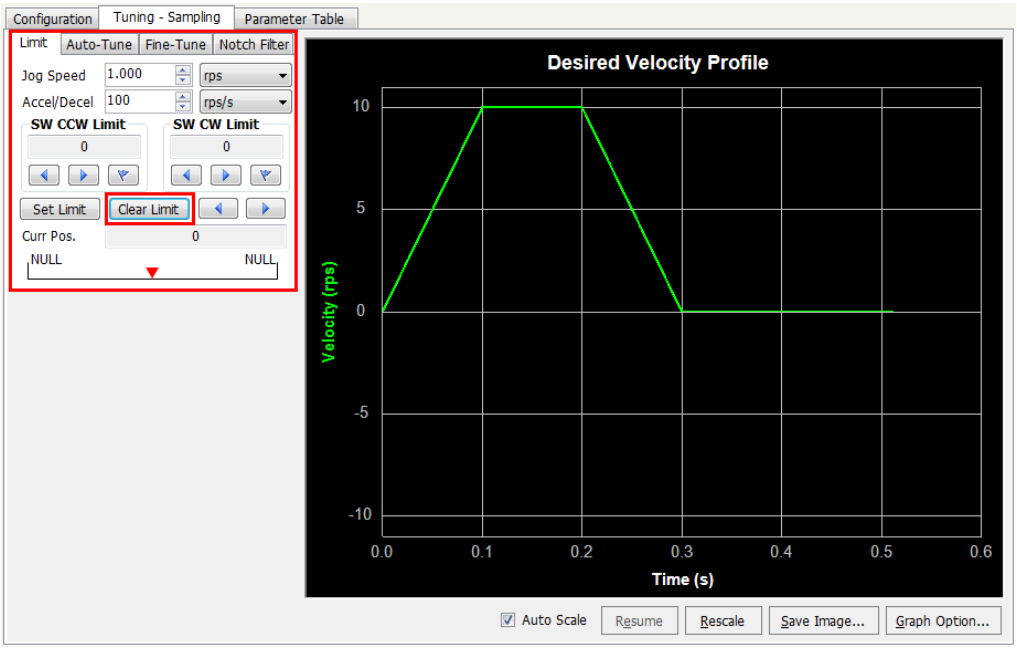


Figure 6-4

5.2.3 Setup Software position Limit

As shown in Figure 6-5:

A. Before limit setting, please set JOG speed, and acceleration and deceleration rate.
B. Set CCW limit (motor rotate in CCW direction)
C. Set CW limit (motor rotate in CCW direction)
D. Confirm or Cancel position limits set by step B and C

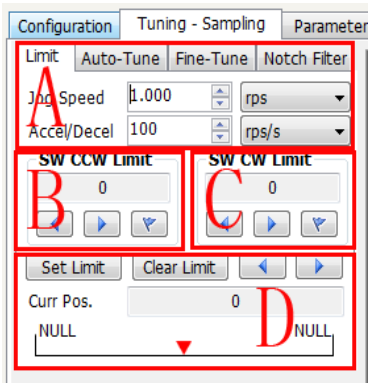
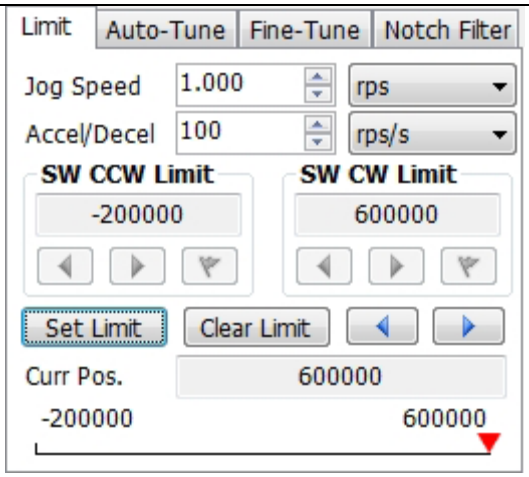


Figure 6-5

Detailed Steps for Software Position Limit

Step	Operation	Software
1	Make sure Servo is Enabled Click  or  to rotate motor in CCW or CW direction When target position reached, click  to setup	
2	Same process as above	
3	Confirm position limit Click on  **NOTE: CW limit must be larger than CCW limit.**	
4	Setting done	

5.2.4 Step 3 Auto-Tuning Function

After entering the "**Auto-Tune**" page, steps are as follows:

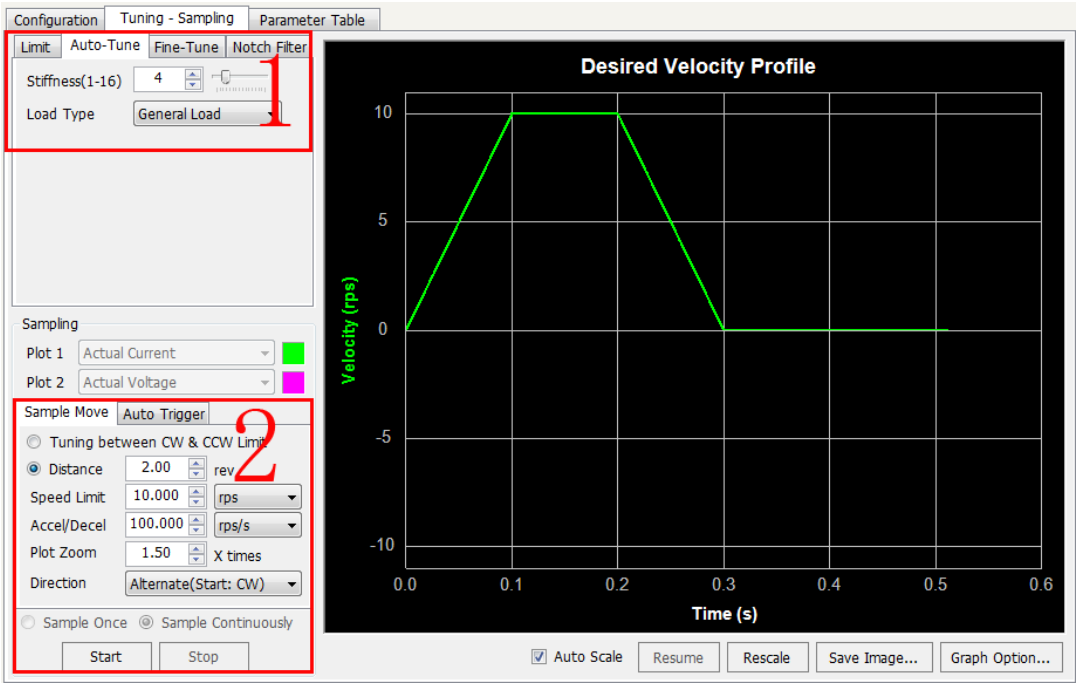
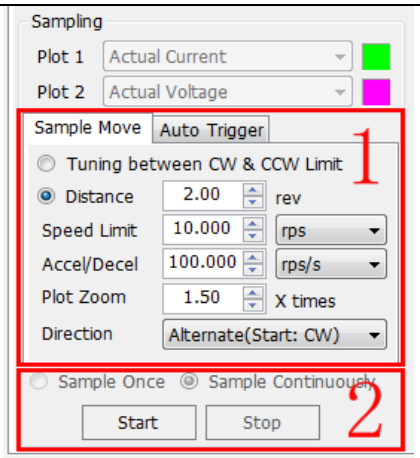
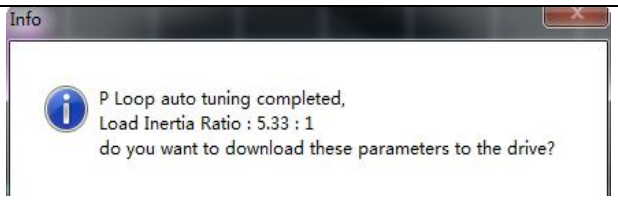


Figure 6-6 Auto Tuning

Operation steps：

1	Set Stiffness and Load type	
2	Set Auto Tuning distance, speed, acceleration and deceleration **NOTE:** 1) **If software position limit is set, please use "Tuning Between CW and CCW Limit** 2) **If no limit is required, please choose "Distance"(Please ensure software position limit is cleared)**	
3	Click start to start auto tuning function	
4	Finish Auto tuning, download parameters into the drive.	

NOTE: During the tuning process, there might be motor or load vibrations. It is normal for the operation, the system will correct itself.

For customized performance requirements, please use fine tuning functions.

5.3 Fine tuning

Based on the mechanical system and the use of servo motor, the following parameters can be tuned to improve the system performance:

- **Global gain (KP),**
- **Position loop gain (KF)**
- **Derivative gain (KD)**
- **Damping gain (KV)**
- **Integrator Gain (KI)**
- **Inertia feed forward gain constant (KK)**
- **Derivative filter gain (KE)**
- **PID filters (KC).**

Among all the parameter, changes for Global gain (**KP**), Derivative filter gain (**KE**) and PID filter (**KC**) are NOT recommended after system configuration. Therefore, parameter tunings are more based on Position loop gain (**KF**) Derivative gain (**KD**), Damping gain (**KV**), Integrator Gain (**KI**), Inertia feed forward gain constant (**KK**). Details are explained below.

5.3.1 Position loop gain (KF)

This parameter is the primary gain term for minimizing the position error. Increase of KF will increase stiffness and reduce in position time duration. However, it might cause vibration if gain value is too large. This is simplest part of the PID loop. The drive applies current to the motor in direct proportion to the error. Here's an example: if the motor were standing still, and you suddenly turned the shaft by hand, you'd want the drive to increase the motor current so that it goes back into position. The further you disturb the motor from its target position, the more the torque will increase. In general, Increase of KF will increase stiffness and reduce in position time duration. However, it might cause vibration if gain is too large.

As shown in Figure 6-7 below, if KF is small, motor position error will be high at all times (including acceleration, constant velocity, and deceleration).

Figure 6-7 when KF is small (KF =2000)

As shown in Figure 6-8 below, if KF value is appropriate, the position error during acceleration and deceleration will be settled very quickly, and position error at constant speed is between ±1

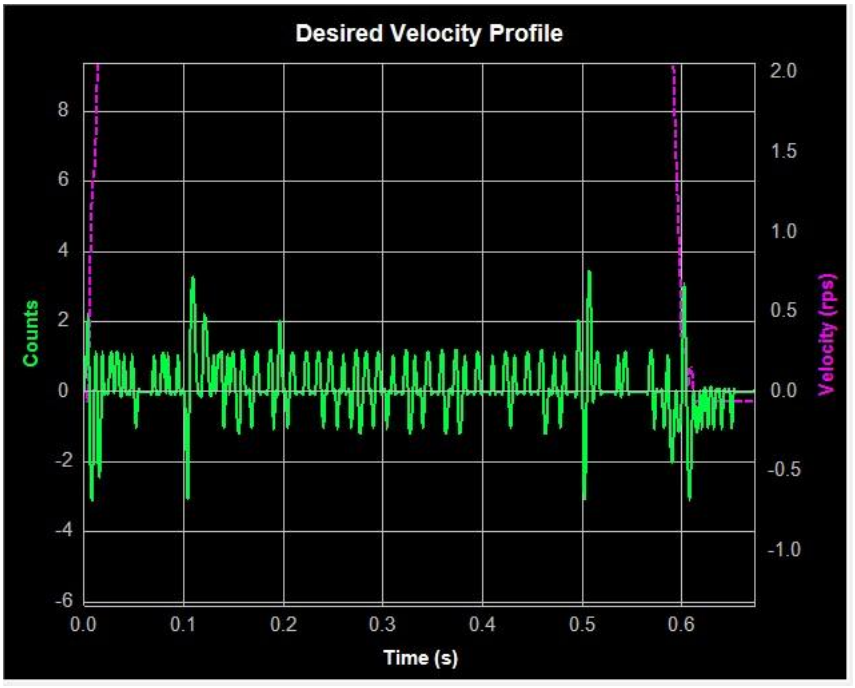


Figure 6-8 KF is appropriate KF = 16000

5.3.2 Integrator Gain (KI)

Parameter KF itself sometimes cannot give the best performance for position error, or it might requires a long time for position limit to settle. In these cases, the Item will keep adding up that error and continue to increase the torque until the motor truly returns to the target position.

As Figure 6-9 is shown, when KI is small, the system will require a long settling time for position error to reach steady (around 0) during acceleration and deceleration.

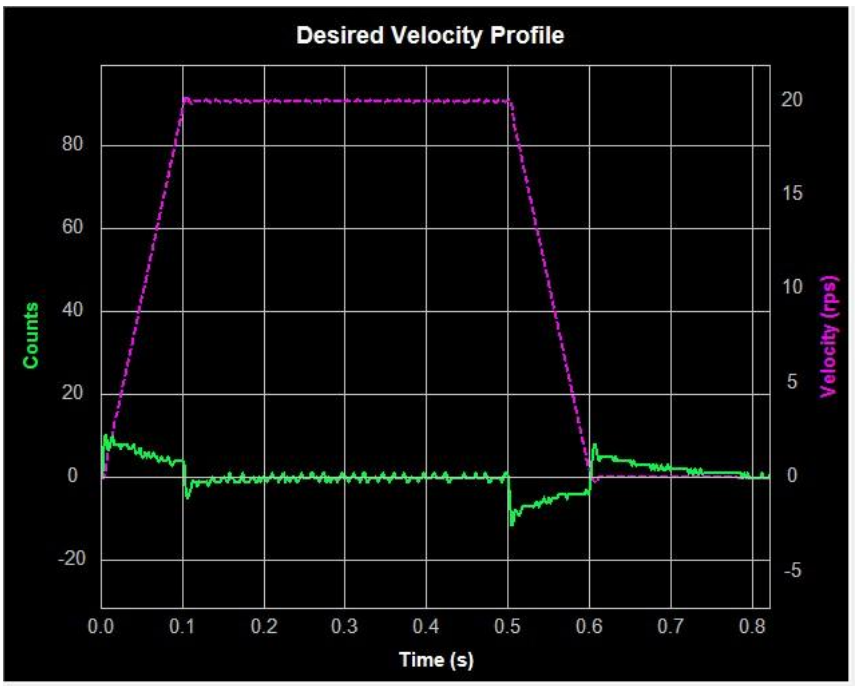


Figure 6- 9 KI =50 (too small)

As Figure 6-10 and 11-11 is shown, Increase KI can improve system response time, and reduce position error and settling time for position error during motor acceleration and deceleration.

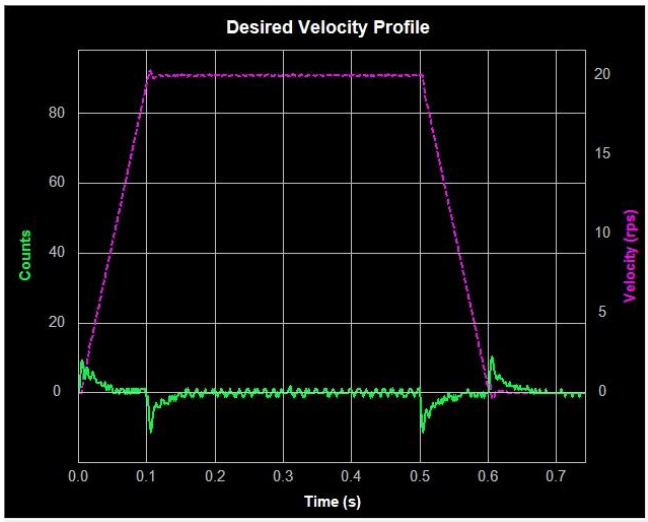


Figure 6-10 KI= 200

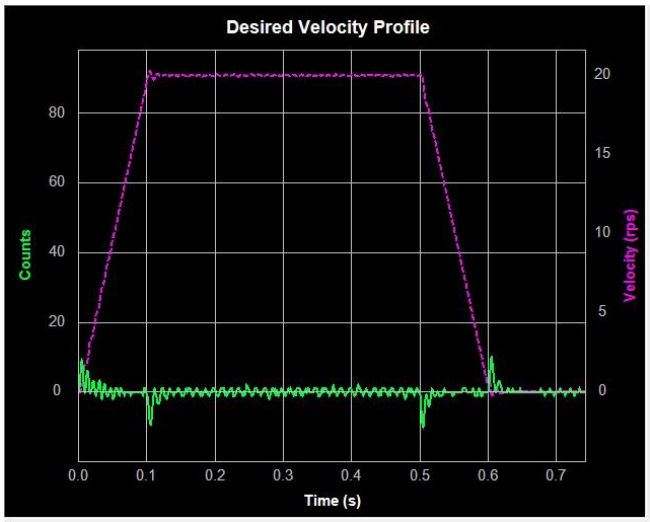


Figure 6-11 KI = 500

As Figure 6-12 is shown, if KI is too large, it will cause the whole servo system vibration and making unexpected noises. It in turn will also increase the position error, and the system might never settle.

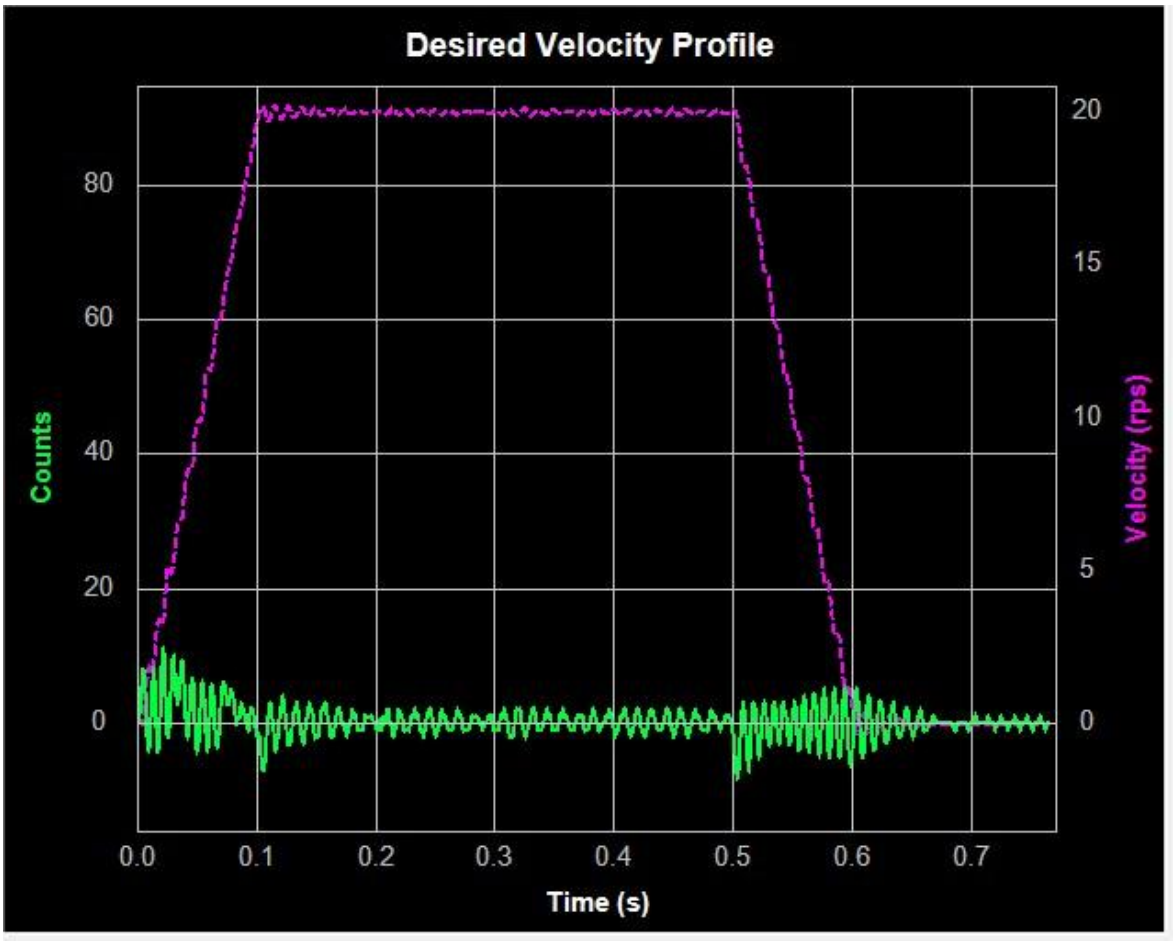


Figure 6-12 KI = 10000 (too large)

5.3.3 Damping gain (KV)

As the motor load inertia increases, the servo system will require higher damping gain to ensure low amount of position errors during constant speed motor rotation and motor in stop position.

When KV is too small, it low damping value will cause high motor position error fluctuations when motor is in constant running or stop in position. As Figure 6-13 is shown, the high position error value has occurred during constant moving and stopping. In turn it will cause motor vibration.

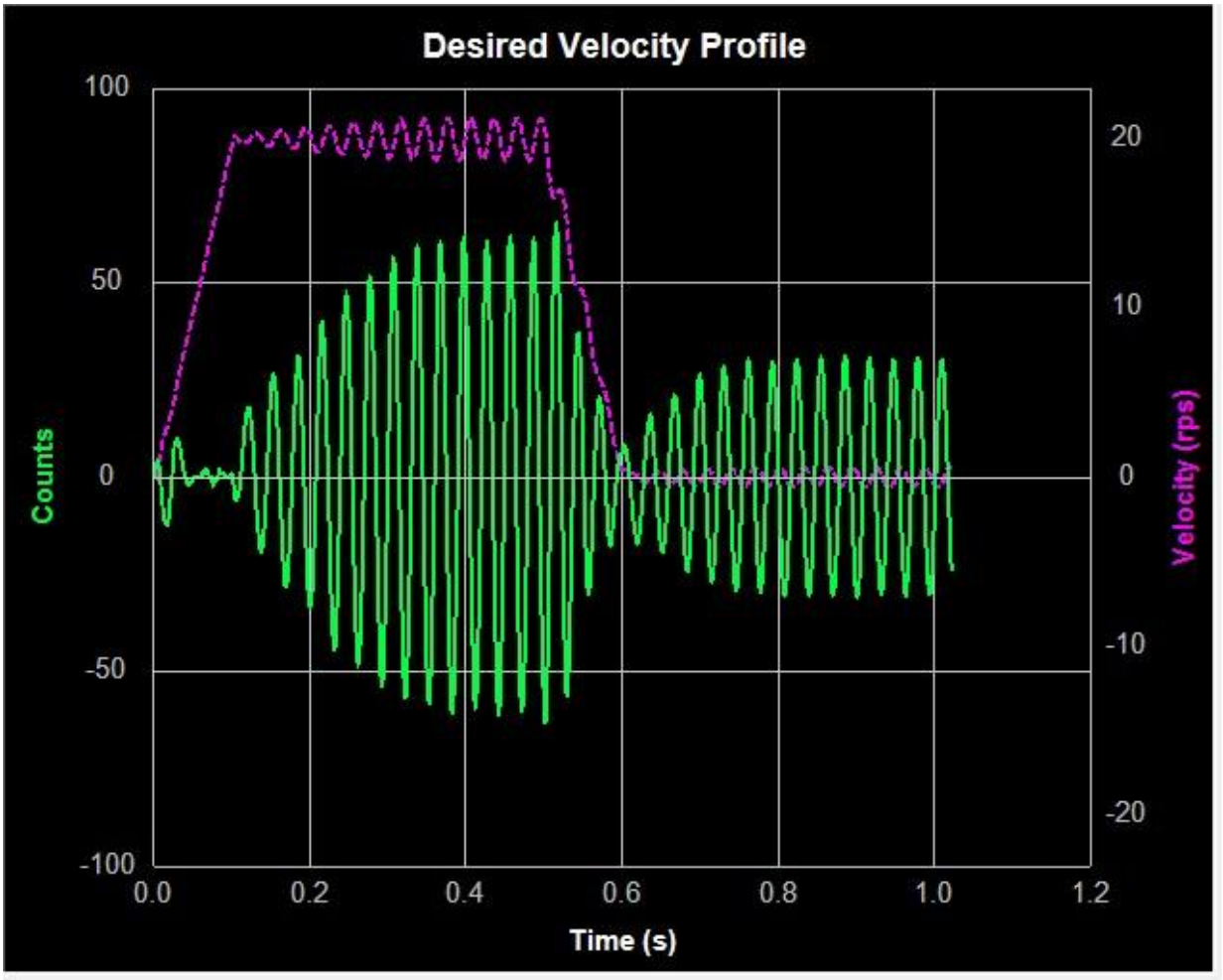


Figure 6-13 KV =5000 (too small)

As Figure 6-14 and 11-15 shown, the position error is reduced as KV increases.

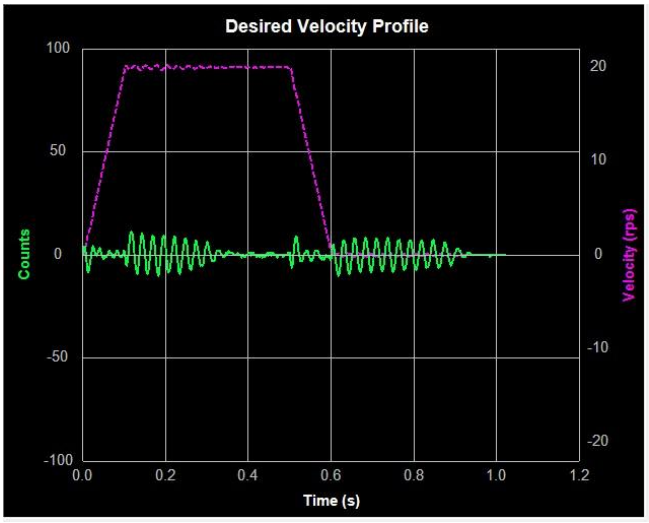


Figure 6-14 KV= 10000

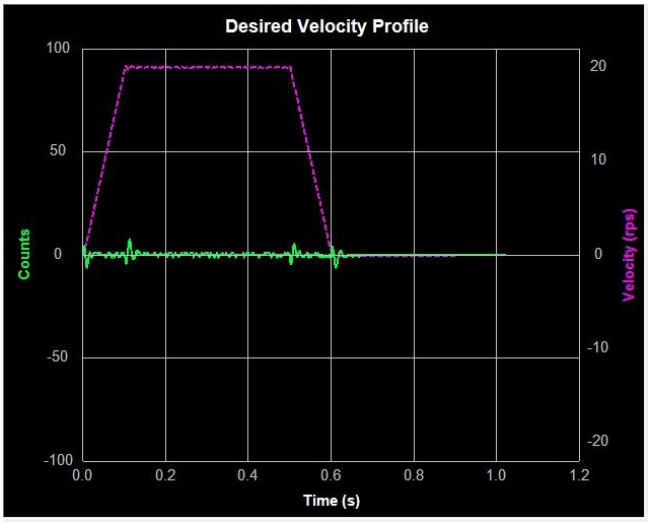


Figure 6-15 KV = 16000

When KV is too large, the strong damping gain will cause system vibration and noises during acceleration and deceleration. As shown in Figure 6-16 by yellow marker below:

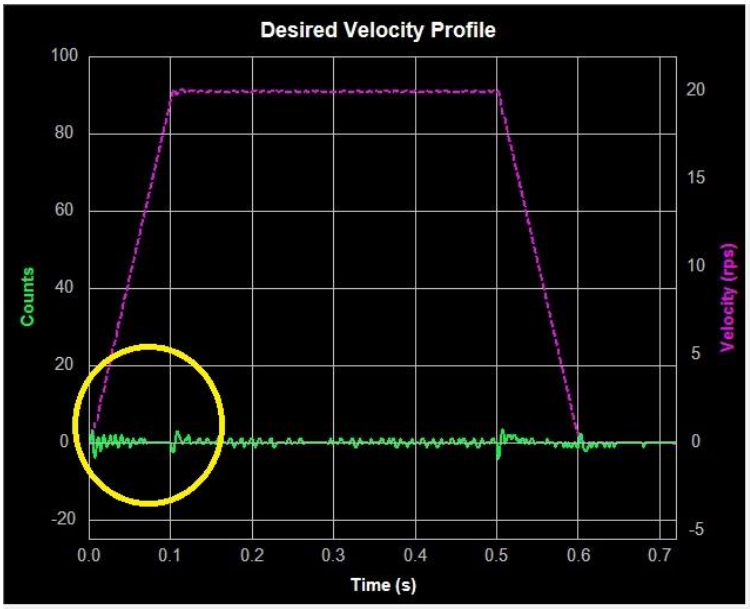


Figure 6-16 KV = 32000 (too large)

5.3.4 Derivative gain (KD)

PI controller the motor would overreact to small errors, creating ever larger errors, ultimately becoming unstable. If you knew what the motor was going to do before it did it, you could prevent this. If you are driving your car into your garage, most people will not fully hit the brake until the car is fully into the garage. Instead, most people slow down as they see the distance between them and their objective get smaller.

A motor drive can control a motor better if it examines the rate of change of the position error and includes that in its torque calculation. When KD is small, the system might not be able to settle quickly after the change of motion, as shown in Figure 6-17 below:

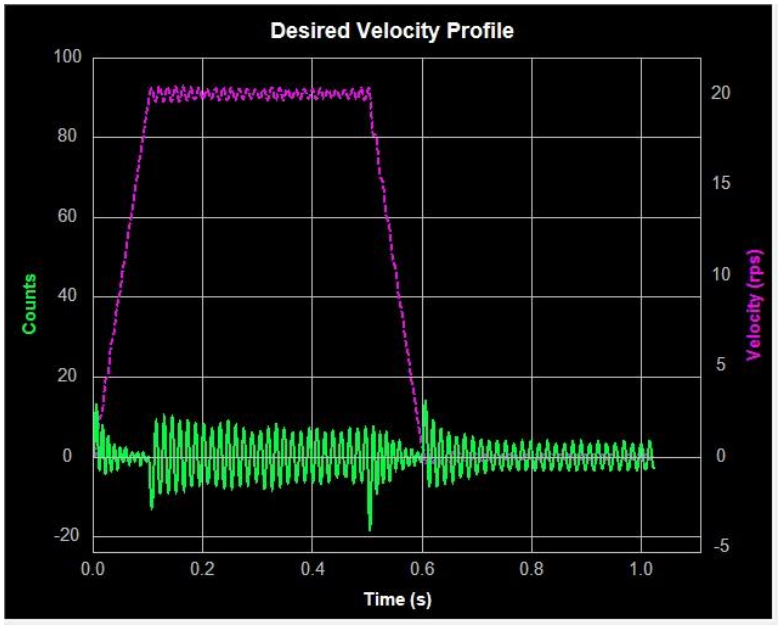


Figure 6-17 KD = 3000 (too small)

As KD increases, the system will require less time to settle, as shown in Figure 6-18 and 11-19 below:

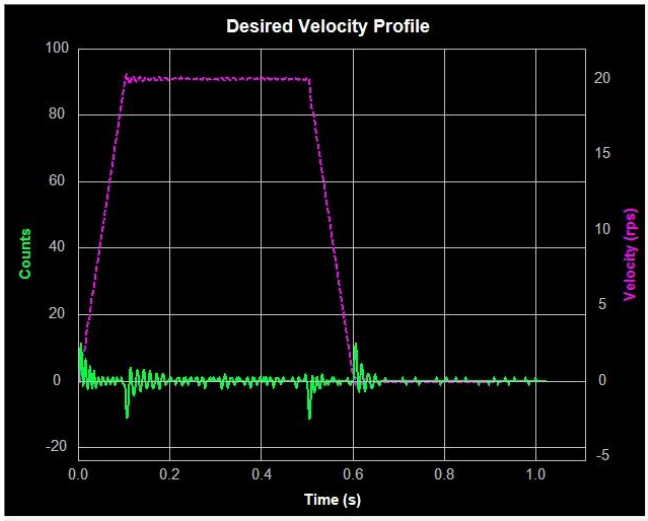


Figure 6-18 KD= 4000

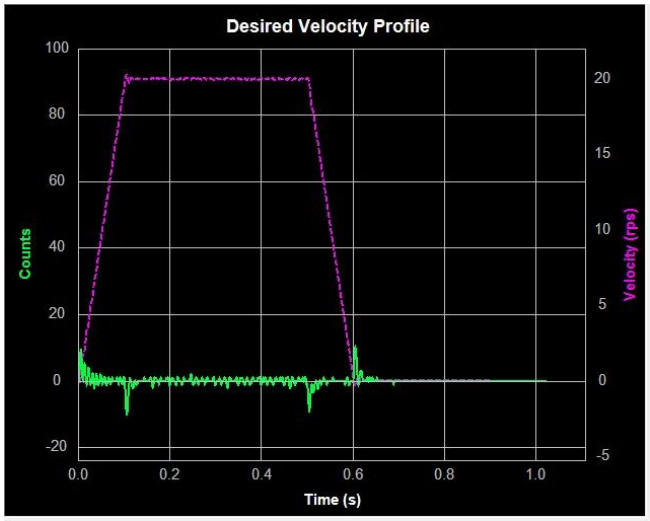


Figure 6-19 KD= 7000

When KD is too large, the system will become highly sensitive to the change of motion, it will potentially cause unexpected system vibrations and noises. As shown in Figure 6-20 below:

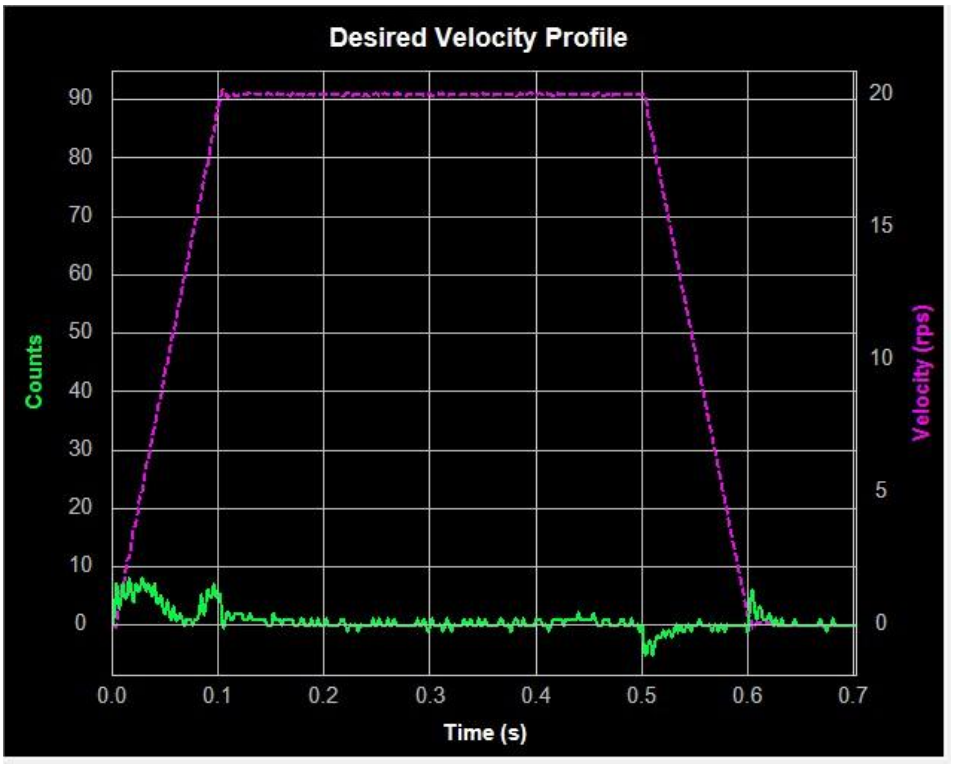


Figure 6-18 KD = 15000 (too large)

5.3.5 Inertia Feedforward Constant (KK)

With larger loads typically comes larger load Inertia. These larger inertias can be more easily accelerated or decelerated by anticipating the control system needs. The Inertia Feedforward term does this by adding an acceleration value to the control value, and reduces position error during acceleration and deceleration.

When KK is small, the feedforward constant will not be enough. It will cause bad effects on system dynamic performance during the acceleration and deceleration. In turn, the position error will be larger, and settling time for position error will be longer. As shown in Figure 6-21 below.

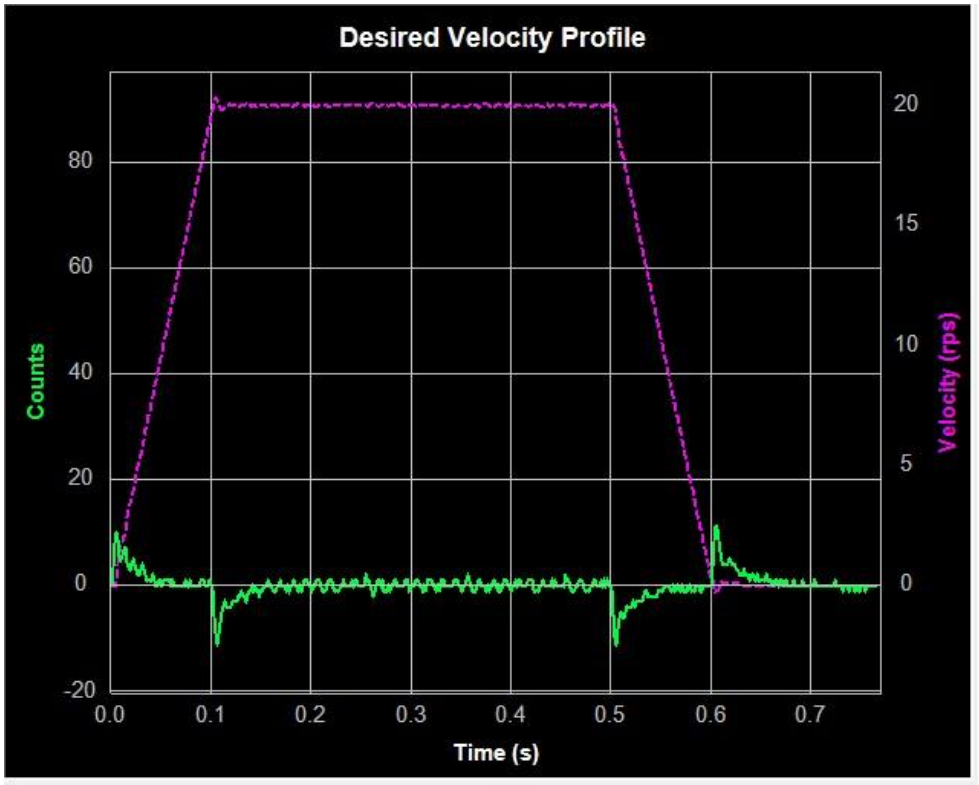


Figure 6-21 KK = 2000(too small)

As Figure 6-22 and 11-23 below is shown, as KK increases, the system dynamic performance improves, the position error during acceleration and deceleration reduces significantly.

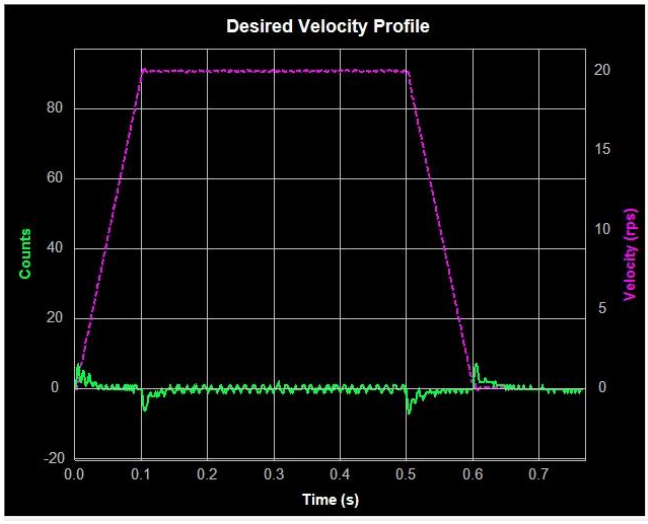


Figure 6-22 KK= 4000

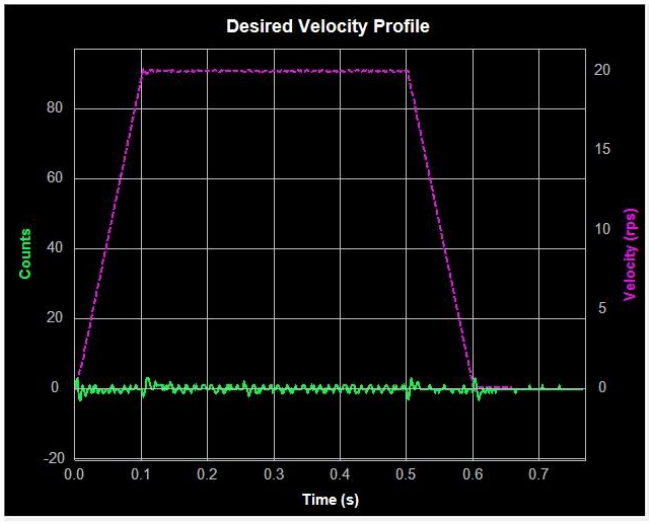


Figure 6-23 KK = 11000

When KK value is too large, the feedforward constant will cause the opposite effect. Therefore it will also decrease system dynamic performance, by increase position error and system settling time, as shown in Figure 6-24 below:

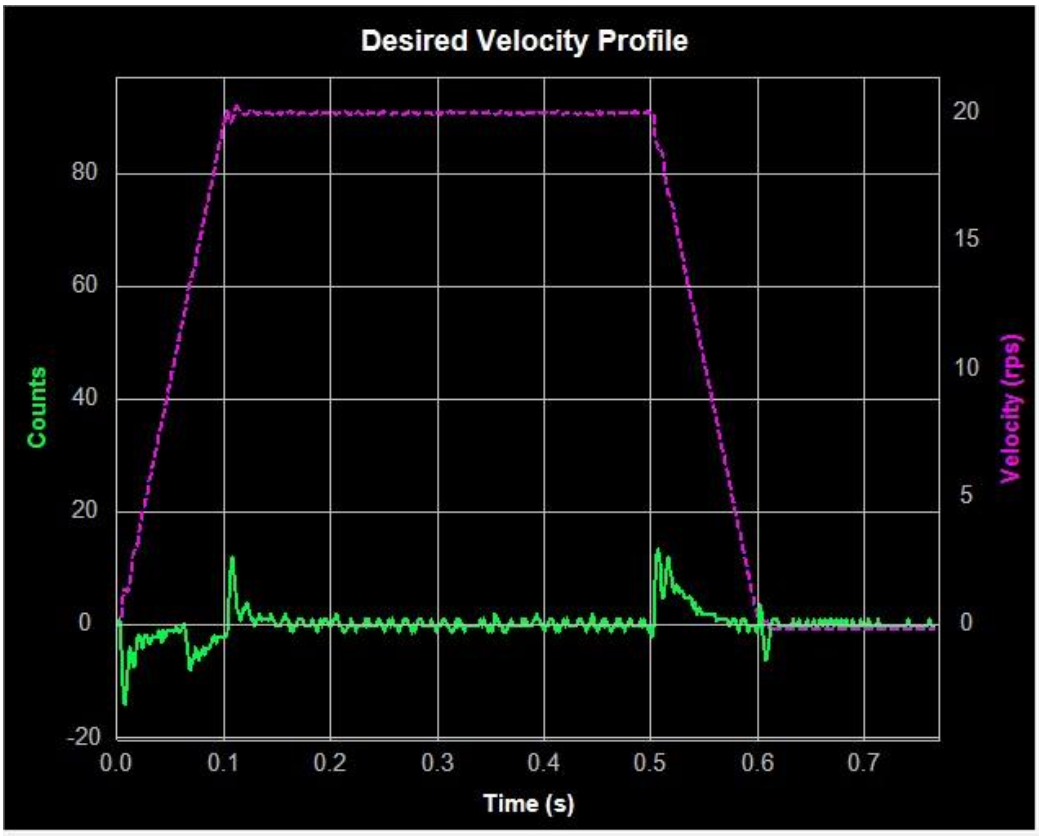


Figure 6-24 KK=19000 (too large)

5.3.6 Follow Factor（KL）

Higher value will reduce system noise, eliminate the overshoot, but it will reduce the system dynamic following performance. Lower value will raise system stiffness, but will cause system noise probably. As shown in Figure 6-25 and 1-26 below (Curve in green is Actual Speed, the purple one is Position error).

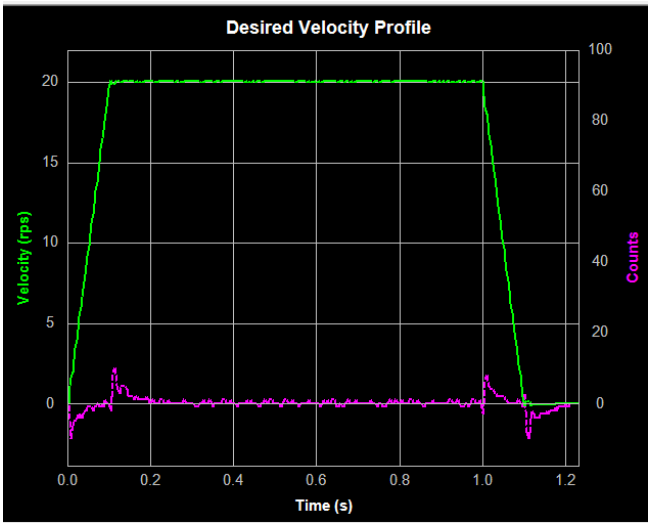


Figure 6-25 Follow Factor (KL) = 0

KL = 0，Good stiffness and less position error

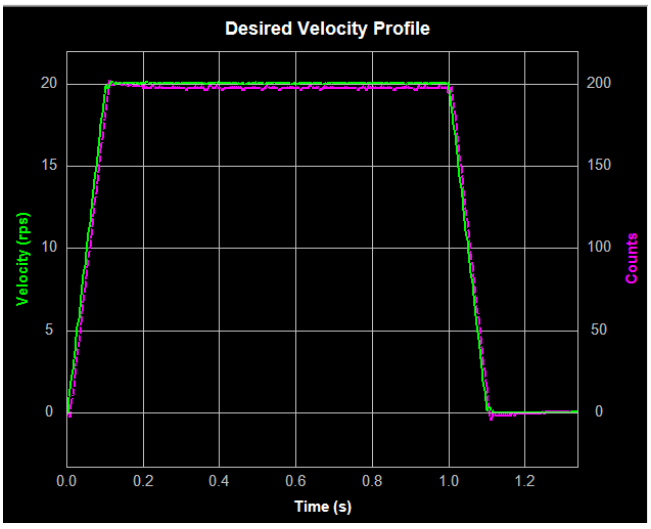


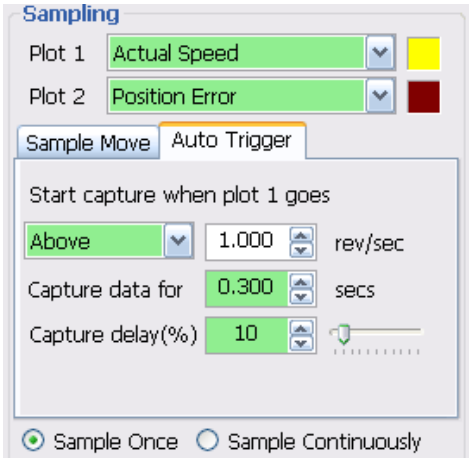
Figure 6-26 Follow Factor (KL) = 15000

KL=15000，The position error curve same as the Actual Speed

5.4 Using Auto Trigger Sampling

In cases where an external controller is used to perform move profiles, such as in the **Position Control Mode** using **Pulse & Direction** input, the **Auto Trigger** will allow the **Sampling** to collect data and display the move profile.
This sampling technique is different in that it is not triggered by the start of a move profile as the drive cannot know when the move is actually started (remember the controller is external). Instead the **Auto Trigger** waits for a predefined set of conditions to tell it when to start collecting the move profile data.

When using **Auto Trigger**, the primary effort is to select the conditions that will trigger the sampling. Begin by selecting the desired trigger value in the **Plot 1** list. This selection is what is monitored by the Auto Trigger, **Plot 2** is not monitored.



In the Auto Trigger tab the displayed text will indicate the value to be used and the conditions to trigger the capture of the selected value. In the example to the right, the capture will begin when **Actual Speed** is **Above 1.000 rev/sec**, the capture will **Capture data** for **0.300 seconds** and there will be a **10% Capture delay** from the beginning of the capture to the trigger point. The **Capture delay** allows viewing of the data prior to the trigger point so that a more complete profile can be observed.
When changing **Plot 1** to other selections notice that the conditions for the capture trigger will change with it. For example, when selecting **Position Error** the capture will look at **Counts** for determining the trigger point.
Sample Once: when the **Start** button is clicked the servo drive begins continuous collection of data. It will constantly check the data to see if the value meets the capture trigger conditions. At the same time Quick Tuner monitors the status of the servo drive to detect if the capture is complete.
When the capture is complete the data is displayed in the profile window.
Sample Continuously: when the **Start** button is clicked the capture is repeated each time the trigger condition is met until the **Stop** button is clicked. During continuous sampling the tuning gains can be changed at any time and will be updated automatically. This allows more dynamic adjustment of the gains for speeding up the tuning process

NOTE: When adjusting control loop gain values remember that the FF Term (KK) has no effect when operating in the Position – Pulse & Direction Control Mode.

6 Step 3: Q Programmer

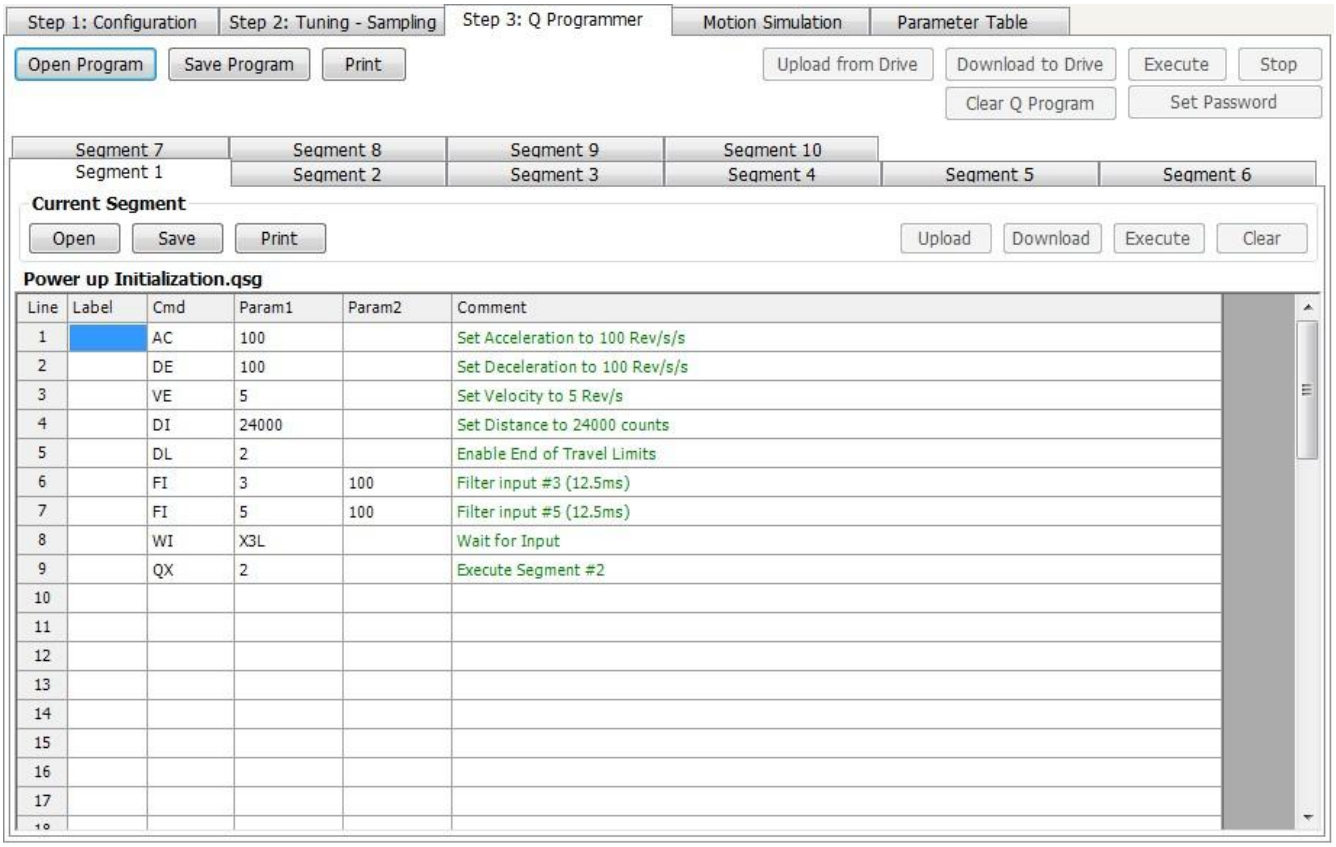
The use of SCL commands with MOONS' dates back many years. A few years ago a new control platform was created that expanded the use of SCL commands and allowed users to create stored programs with SCL commands. These programs could be saved in a drive's non-volatile memory, and the drive could run these programs stand-alone, or without a permanent connection to the host. This expansion of SCL's capabilities was called Q, and since that time MOONS' has continued to expand the offering of drives with the Q motion controller built in. By combining the ability to run a sophisticated, single-axis motion control program stand-alone and the ability to communicate serially to a host device, Q drives offer a high level of flexibility and functionality to the machine designer and system integrator. The characteristic as follows;
Single-Axis motion control

- Single-Axis motion control
- Stand Alone
- Multi-task
- Conditional Processing
- Math Calculation
- Data register manipulation
- Motion Simulation

A single Q program can have 10 individual segments, each segment can have maximum 62 lines of command.

6.1 Q programmer Page

The Q programmer pager is as follows:



Open Q program: Open Q program file from your computer disk

Save Q program: Save Q program file to your computer disk

Print: Print current Q program

Upload from Drive: Upload Q program from the drive.

Download to Drive: Download current Q program to the drive.

Clear Q Program: Clear current Q program.
Execute: Execute current Q program.
Stop: Stop the current running Q program
Set Password: Set Q program password. Password set for upload Q program from the drive. Wrong password entry will not able to upload from the drive. If you want to rest your password, please input default password “1234”, but it will clear up all stored Q program.
Auto Execute Q program at power up: check the box, drive will automatically execute segment 1 of the Q program at power up.

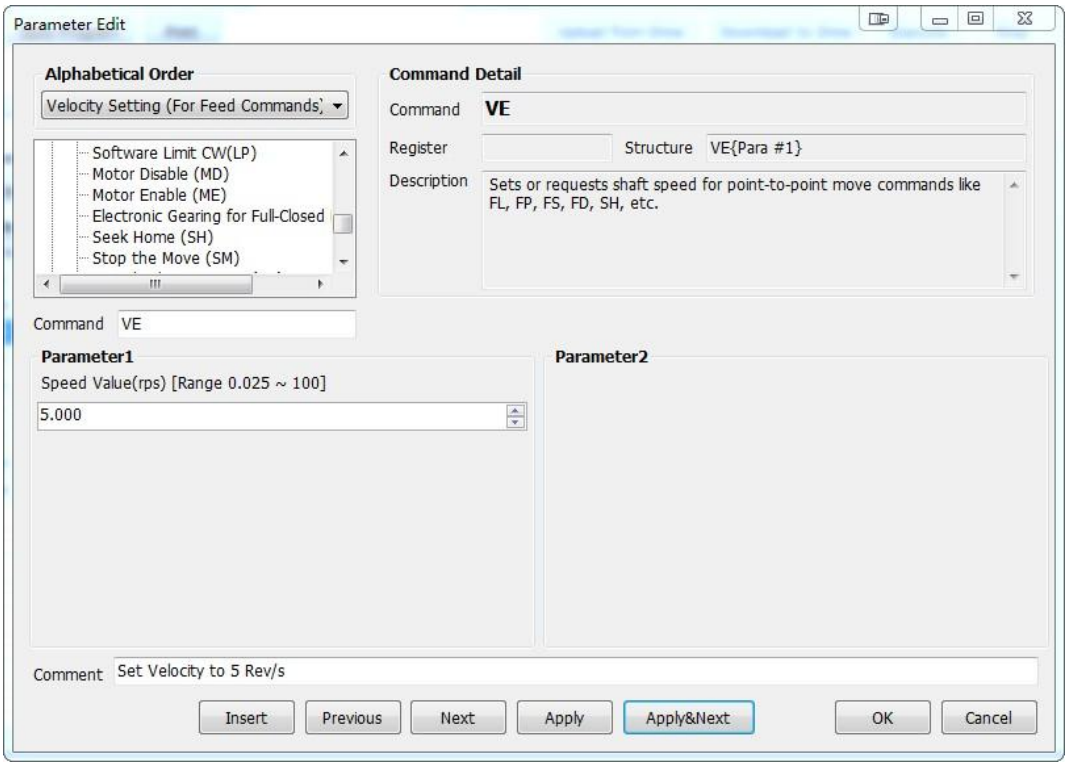
6.2 Current Segment

There are 10 Q segments within the Q program “current segment” page is used to edit the segment that is currently viewing.

Open Q segment: Open Q segment file from your computer disk
Save Q segment: Save Q segment file from your computer disk
Print: Print current Q segment
Upload from Drive: Upload Q segment from the drive.
Download from Drive: Download Q segment from the drive.
Execute: Execute current Q segment.
Stop: Stop current Q segment.

6.3 Command Editing

Click on any box from the Cmd list, and then click on the button, the Command editing page will pop up as follows:



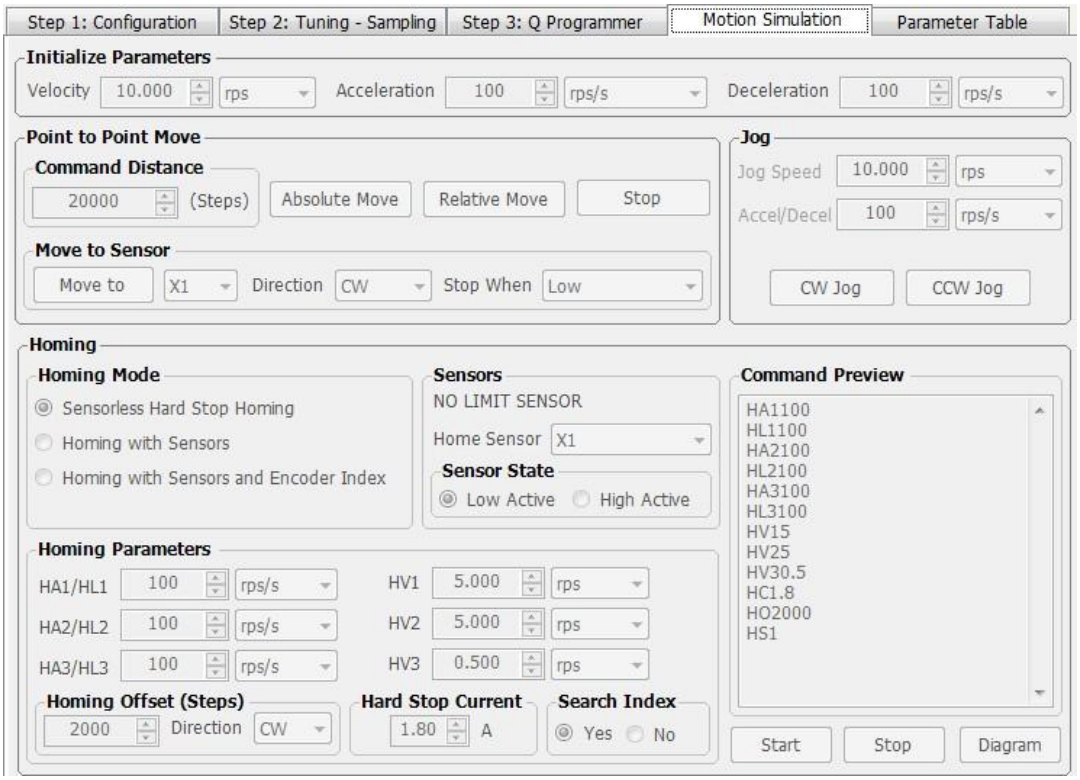
The Command list is on the left hand side of the window. In addition, you can also search the command by

alphabetical order by opening the list above the tree, or type in command name directly in the “command” box.
If command is found, the detailed command details will be shown on the right hand side of the window. The command value can be entered via parameter 1 or parameter 2 box based on command entry requirement. Comment allows you to write descriptions.

Insert:	Insert a blank line within the current Q segment.
Previous:	Moving up by one line within the current Q segment.
Next:	Moving down by one line within the current Q segment.
Apply:	Apply current command to the segment
Apply and Next:	Apply current command and move to the next line.
Ok:	Apply current command to the segment and quit.
Cancel:	Quit the command editing window without save the change.

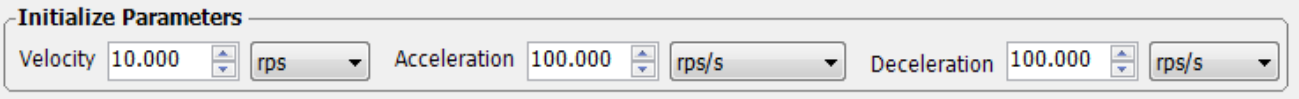
7 Motion Simulation

Motion simulation provides Point to Point Move, Jog and Homing simulation.



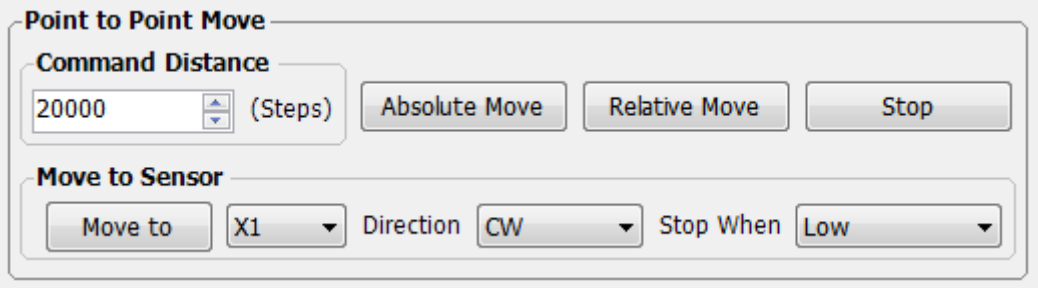
7.1 Initialize Parameters

Initialize the motion parameters velocity, acceleration and deceleration.



7.2 Point to Point Move

Point to Point Move allows you set Command distance and some motion conditions. Then click the Move button to do so.



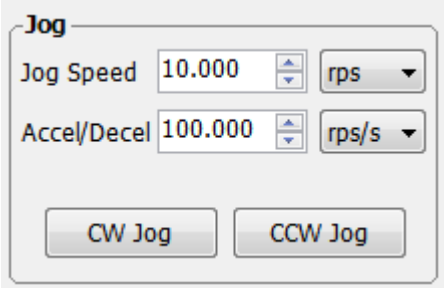
Absolute Move: Execute the absolute motion according to the set distance. The Absolute ZERO is the zero count of motor encoder.

Relative Move: Execute the relative motion according to the set distance.

Move to Sensor: Click the" Move to" after set the conditions.

7.3 Jog

The Jog allows you set Jog Speed, Jog acceleration/deceleration and launch Jog. Click down the CW Jog or CCW Jog button to start and mouse up to stop.



7.4 Homing

Homing allows you set homing mode, sensor state, search speed and acceleration/deceleration, offset, hard stop current etc... So that the drive homing until the homing sensor is active. Click "Start" to start home, also you can click "Stop" button to interrupt when homing.

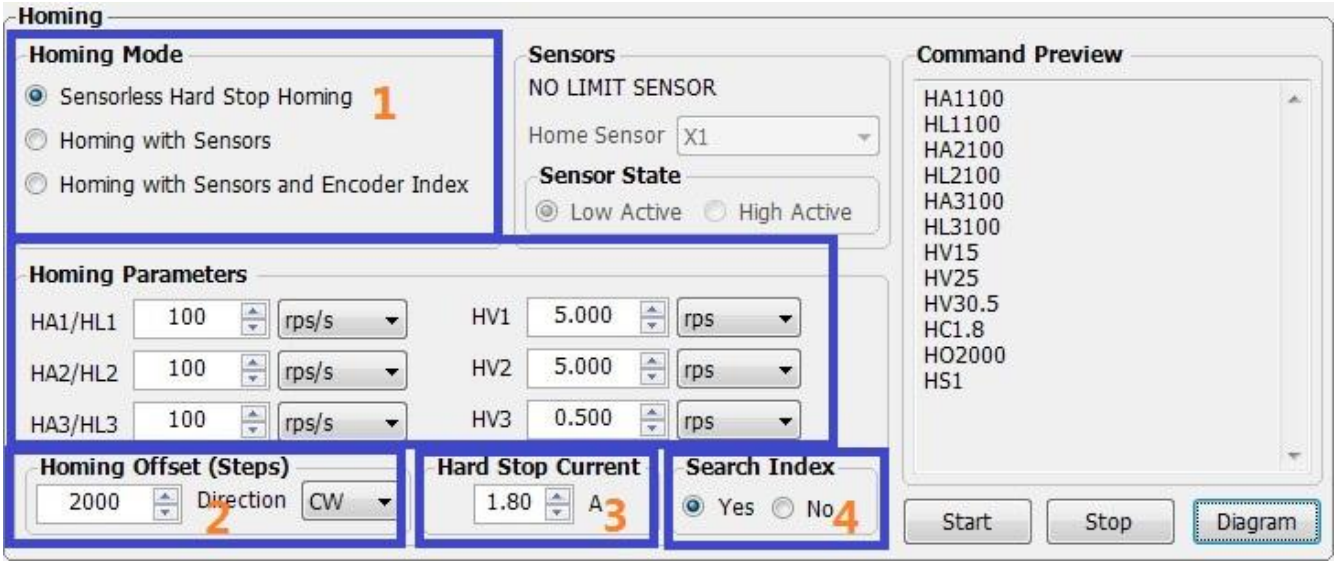
7.4.1 Homing Mode

There is three homing mode can be selected, Sensorless Hard Stop Homing, Homing with Sensors, Homing with Sensors and Encoder Index.

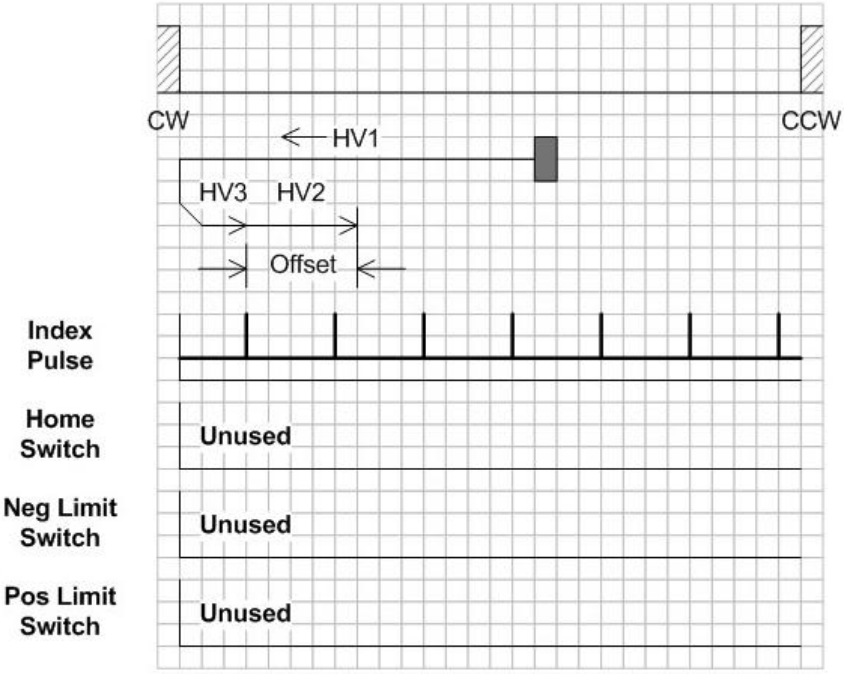
Click the ***"Diagram"*** button to get more details of each Homing mode.

7.4.1.1 Sensorless Hard Stop Homing

Sensorless Hard Stop Homing means homing without any homing sensors. The load will homing to a fixed mechaniccal end with set current condition.



The diagram of the whole homing process is shown as follows.



Searching mechanical end with HV1, The start direction comes from the sign of the HO command ("-" is CCW, no sign is CW). Motor stops while the actual current is equal to HC when reaching the mechanical end. And then;

If Check YES of Search Index

Motor runs opposite with HV3 to the first encoder index. After that motor move to HO--Homing offset with HV2.

Or Check NO of Search Index

Motor move to HO--Homing offset with HV2.

7.4.1.2 Homing with Sensors

Executes an Extended Homing command. Requires input number and condition for the home sensor. Speed is set by HV command, there are three velocity setting for different steps (see the detail as following description).

Acceleration and deceleration are set by HA (Homing Accel) and HL (Homing Decel). The start direction comes from the sign of the HO command ("-" is CCW, no sign is CW). Here following the description for each commands and motor motion.

HV1: Homing velocity for searching Limit Sensor and Home sensor.

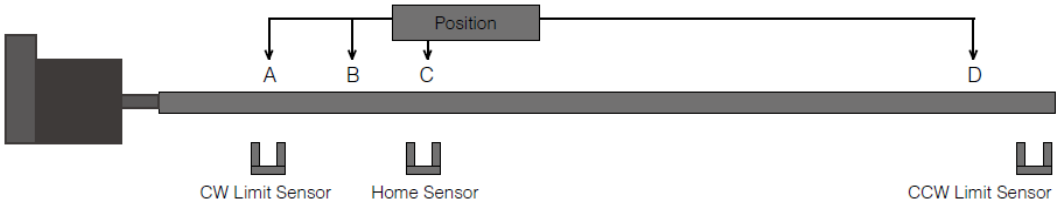
HV2: Homing velocity for moving the setting distance after (beyond) home sensor reached.

HV3: Homing velocity for return back to home sensor after setting distance moving finished.

HO: distance to move after home sensor reached.

Here shows an example of Extended Homing operation as following.

Condition: HO = 20000(no sign for CW direction), DL = 2



(1) When the motor is positioned at A (CW Limit Sensor triggered)

-The motor searches the home sensor at high speed specified by HV1 value, with HA1/HL1 for acceleration/deceleration. Once home sensor is reached, it will move to the distance specified by HO value with HV2 speed and HA2/HL2 acceleration/deceleration beyond home sensor in CCW direction. Finally, the motor approaches back with HV3 speed and HA3/HL3 acceleration/deceleration to home sensor.

(2) When the motor is positioned at B (Motor Stop between CW Limit Sensor and Home Sensor)

-The motor moves by CW direction to find CW limit sensor with HV1 speed and HA1/HL1 acceleration/deceleration.

-The CW limit sensor triggered and stopped.

-Then the motor moves the same as above (1).

(3) When the motor is positioned at C (Home Sensor triggered)

-The motor moves to the distance specified by HO value with HV2 speed and HA2/HL2 acceleration/deceleration beyond home sensor in CCW direction. Finally, the motor approaches back with HV3 speed and

HA3/HL3 acceleration/deceleration to home sensor.

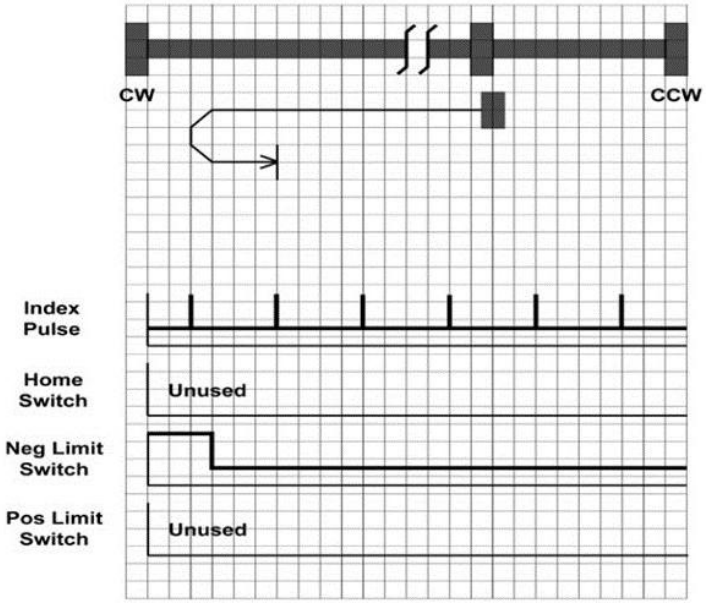
(4) When the motor is positioned at D (Motor Stop between Home Sensor and CCW Limit Sensor)

-The motor moves to home sensor with speed HV1 in CW direction.

-After Home Sensor triggered, the motor moves the same as above (3)

NOTE: If the HO value is negative, the motor will start at CCW direction.

7.4.1.3 Homing with Sensors and Encoder Index



As shown above, the initial direction of movement shall be CW if the CW limit switch is inactive (here: low), the home position shall be at the first index pulse to the CCW direction where the CW limit switch becomes inactive.

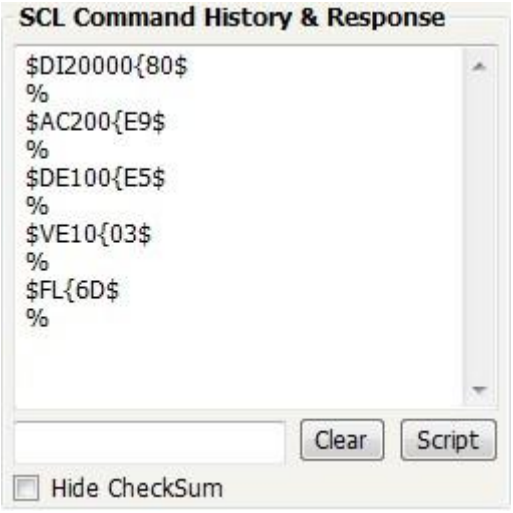
Firstly the motor moves in CW direction and stops when CW limit switch is triggered. Then it moves in CCW direction until index is firstly reached after the CW limit switch becomes inactive from active state. Velocity, acceleration and deceleration are set by VE, AC and DE respectively in the first move. Velocity, acceleration and deceleration are set by VC, AC and DE commands respectively in the second move. Index is masked until it moves in CCW direction and CW switch is changed to inactive state from active. DL command set the active signal state of limit switch, high level or low level.

7.4.2 Command Preview

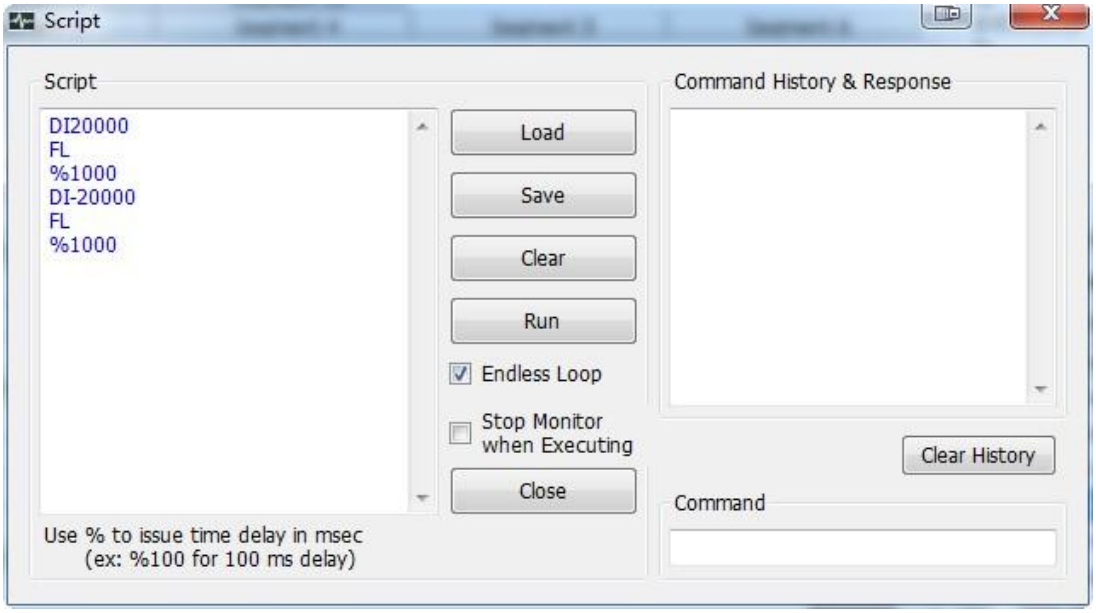
Command Preview will shows all the SCL commands which homing mode is selected.

8 SCL Terminal

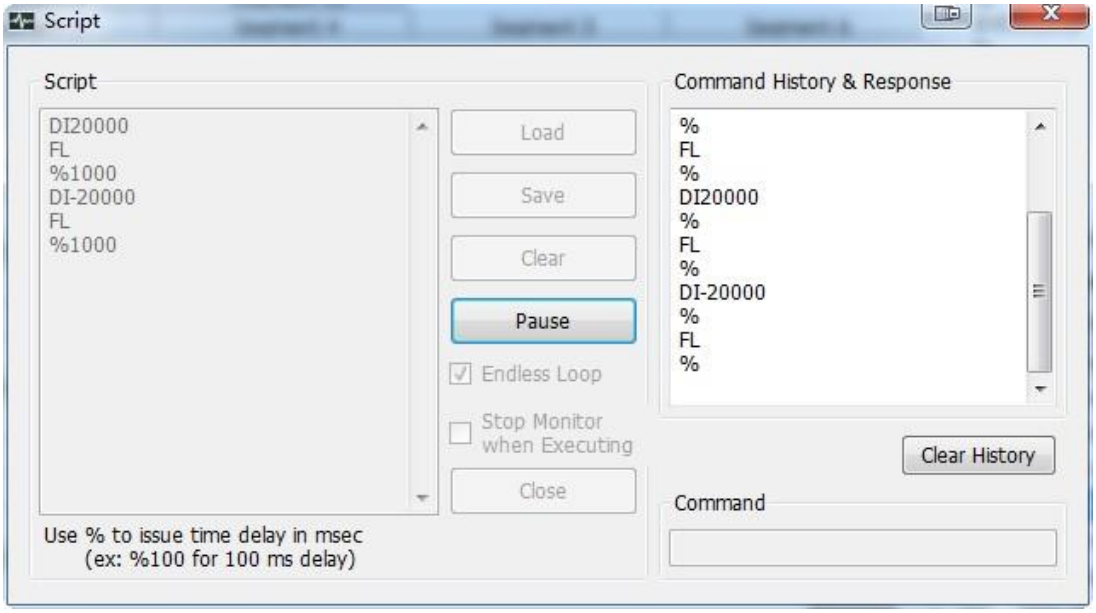
The SCL Terminal allows you to send SCL commands to the drive, regardless of the Operating Mode. The terminal is also useful as a commissioning tool, allowing you to test your drive and SCL without having to launch a separate application.



In SCL terminal window, there is a “Script” button, click on the button, the Script window shows up. See below.



Edit a SCL command script and check “Endless Loop” box, click Run will perform to run SCL commands in looping. Click pause will stop the running.

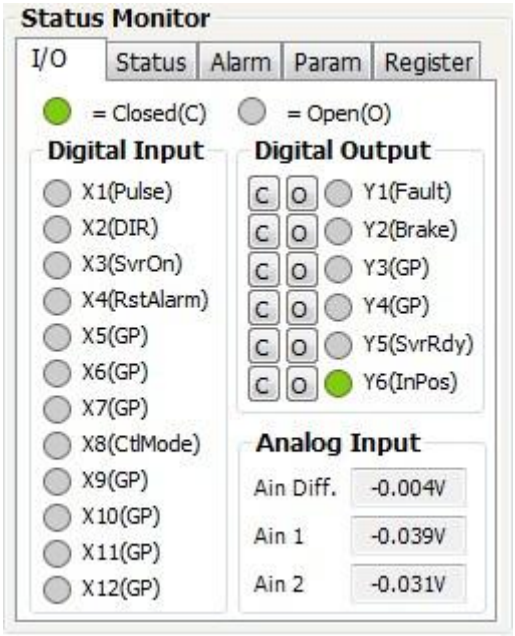


Note: if you check box on "Stop Monitor when Executing", the software will stop background status monitoring. This will make the script run more efficiently and in time.

9 Status Monitor

Status Monitor can display I/O status, Drive status, Alarm, Parameters and Register monitor.

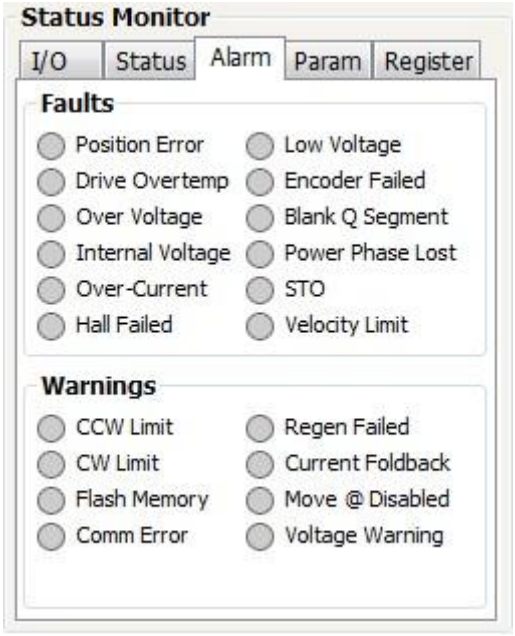
9.1 I/O Monitor



It shows the Digital Input status, measures the analog input value and be able to control the digital output status.

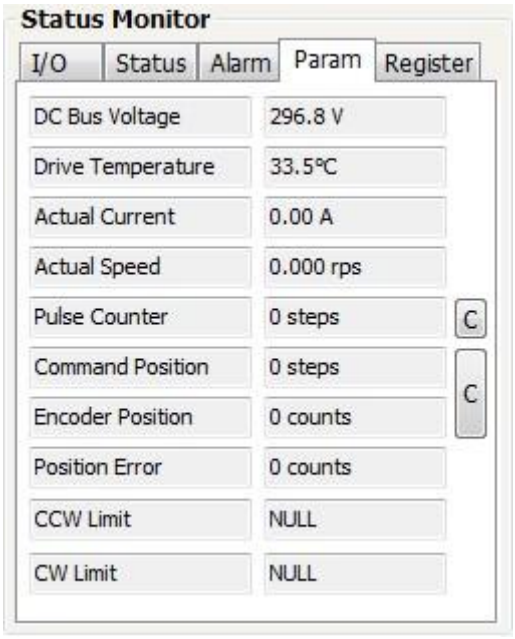
9.2 Drive Status Monitor

9.3 Alarm Monitor

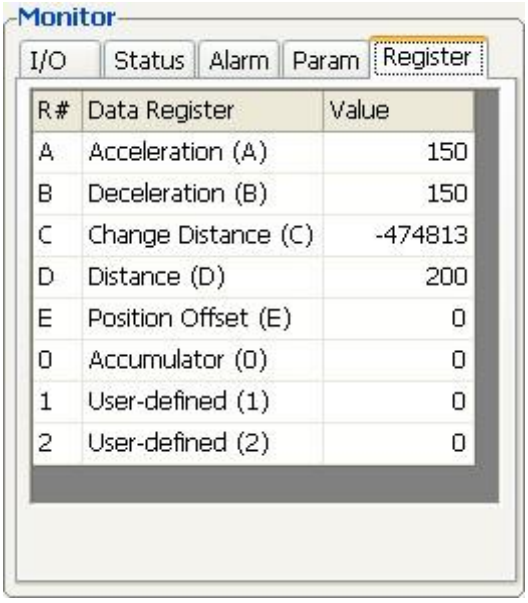


There are two categories of alarm, faults and warnings.
A faults alarm will be indicated in red color flag.
A warning alarm will be indicated in yellow color flag.

9.4 Drive Parameter Monitor



9.5 Register Monitor



10 Appendix A: SCL Reference

SCL or Serial Command Language, was developed to give users a simple way to control a motor drive via serial port. This eliminates the need for separate motion controllers to supply control signals, like Pulse & Direction or +/-10V signals, to your step and servo motor drives. It also provides an easy way to interface to a variety of other industrial devices like PLCs and HMIs, which most often have standard or optional serial ports for communicating to other devices.

NOTE: For more details about SCL command, please click here or download latest Host Command Reference manual from our website www.moonsindustries.com/Products/Drives.This document may be changed without notification to the customers.

10.1 Commands

There are two types of host commands available: buffered and immediate. Buffered commands are loaded into and executed out of the drive's volatile command buffer, also known as the *queue*. Immediate commands are not buffered: when received by the drive they are executed immediately.

10.1.1 Buffered Commands

After being loaded into the command buffer of a drive, buffered commands are executed one at a time. (See "Multi-tasking in Q Drives" below for an exception to this rule). If you send two buffered commands to the drive in succession, like an FL (Feed to Length) command followed by an SS (Send String) command, the SS command sits in the command buffer and waits to execute until the FL command is completed. The command buffer can be filled up with commands for sequential execution without the host controller needing to wait for a specific command to execute before sending the next command. Special buffer commands, like PS (Pause) and CT (Continue), enable the buffer to be loaded and to pause execution until the desired time.

Stored Programs in Q Drives

Stored Q Programs, created with the *Q Programmer* application software, are created by using only buffered commands.

Multi-tasking in Q Drives

Multi-tasking allows for an exception to the "one at a time" rule of buffered commands. The multi-tasking feature of a Q drive allows you to initiate a move command (FL, FP, CJ, FS, etc.) and proceed to execute other commands without waiting for the move command to finish.

10.1.2 Immediate Commands

Immediate commands are executed right away, running in parallel with a buffered command if necessary. For example, this allows you to check the remaining space in the buffer using the BS (Buffer Status) command, or the immediate status of digital inputs using the IS (Input Status) command, while the drive is processing other commands. Immediate commands are designed to access the drive at any time.

MOONS' recommends waiting for an appropriate Ack/Nack response from the drive before sending subsequent commands. This adds limited overhead but ensures that the drive has received and executed the current command, preventing many common communication errors. If the Ack/Nack functionality cannot be used in the application for any reason, the user should allow a 10ms delay between commands to allow the drive sufficient time to receive and act on the last command sent.

This approach allows a host controller to get information from the drive at a high rate, most often for checking drive status or motor position.

10.2 Using Commands

The basic structure of a command packet from the host to the drive is always a text string followed by a carriage return (no line feed required). The text string is always composed of the command itself, followed by any parameters used by the command. The carriage return denotes the end of transmission to the drive. Here is the basic syntax.

YXXAB<cr>

In the syntax above, "Y" symbolizes the drive's RS-485 address, and is only required when using RS-

485 networking. “XX” symbolizes the command itself, which is always composed of two capital letters. “A” symbolizes the first of two possible parameters, and “B” symbolizes the second. Parameters 1 and 2 vary in length, can be letters or numbers, and are often optional. The “<cr>” symbolizes the carriage return which terminates the command string. How the carriage return is generated in your application will depend on your host software.

Once a drive receives the <cr> it will determine whether or not it understood the preceding characters as

a valid command. If it did understand the command the drive will either execute or buffer the command. If Ack/ Nack is turned on (see PR command), the drive will also send an Acknowledge character (Ack) back to the host. The Ack for an executed command is % (percent sign), and for a buffered command is * (asterisk).

It is always recommended that the user program wait for an ACK/NACK character before subsequent commands are sent. If the ACK/NACK functionality cannot be used in the application, a 10ms delay is recommended between non-motion commands.

If the drive did not understand the command it will do nothing. If Ack/Nack is turned on a Nack will be sent, which is signified by a ? (question mark). The Nack is usually accompanied by a numerical code that indicates a particular error. To see a list of these errors see the PR command details in the Appendix.

Responses from the drive will be sent with a similar syntax to the associated SCL command.

YXX=A<cr>

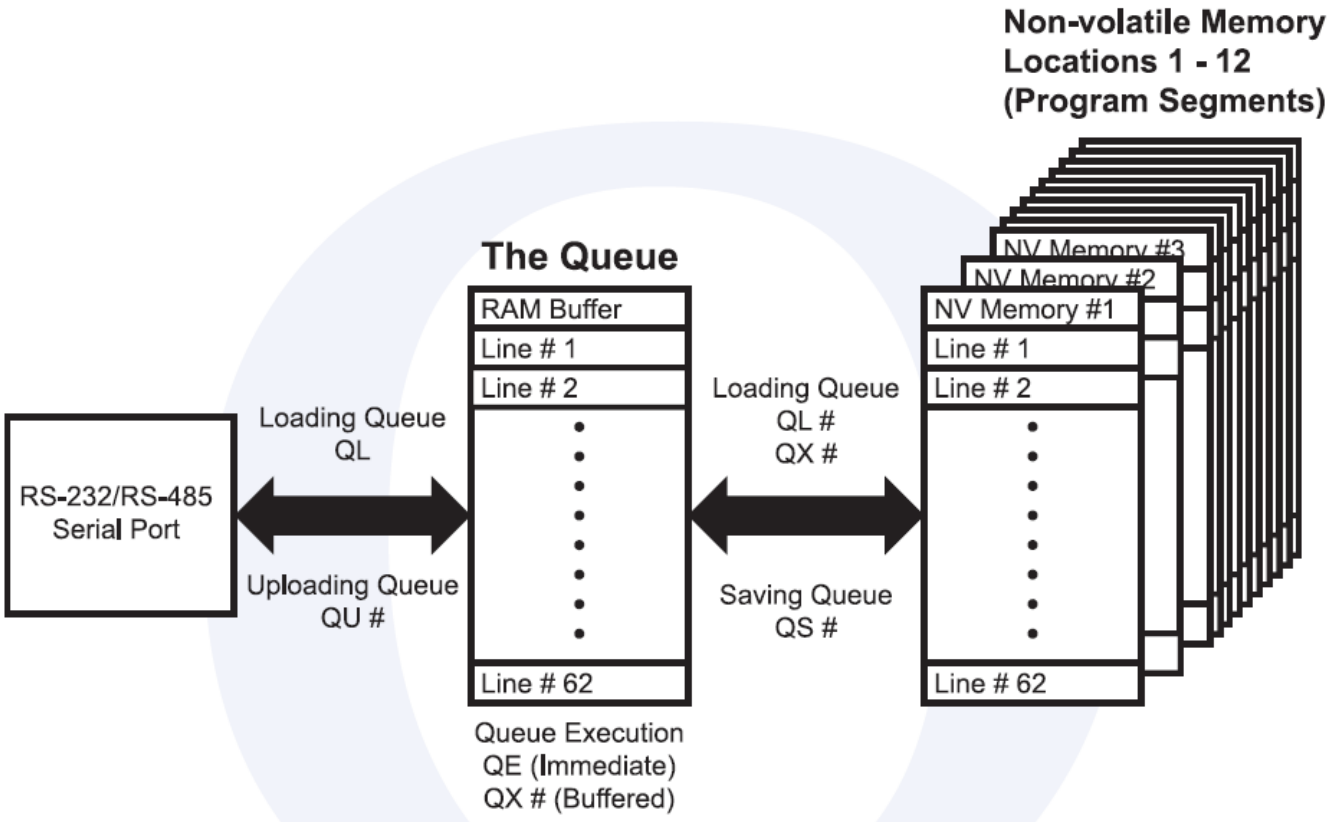
In the syntax above, “Y” symbolizes the drive’s RS-485 address, and is only present when using RS-

485 networking. “XX” symbolizes the command itself, which is always composed of two capital letters. “A” symbolizes the requested data, and may be presented in either Decimal or Hexadecimal format (see the IF command). The “<cr>” symbolizes the carriage return which terminates the response string.

10.2.1 Commands in Q drives

Q drives have additional functionality because commands can also be composed into a stored program that the Q drive can run stand-alone. The syntax for commands stored in a Q program is the same as if the commands were being sent directly from the host, or “XXAB”. Q Programmer software is used to create stored Q programs and can be downloaded for free from www.moons.com.cn.

The diagram below shows how commands sent from the host’s serial port interact with the volatile command buffer (AKA the Queue), and the drive’s non-volatile program memory storage. Loading and Uploading the Queue contents via the serial port are done with the QL and QU commands, respectively. Similarly, the Queue’s contents can be loaded from NV memory using the QL and QX commands, and can be saved to NV memory with the QS command. Finally, commands currently in the Queue can be executed with the QE or QX command.



The *Q Programmer* software automates many of the functions shown in the diagram above.

10.2.2 SCL Utility software

The SCL Utility software is an excellent application for familiarizing yourself with host commands. SCL Utility can be downloaded for free from www.moonsindustries.com

To send commands to your drive from SCL Utility simply type a command in the Command Line and press the ENTER key to send it. (Remember that all commands are capital letters so pressing the Caps Lock key first is a good tip). Pressing the ENTER key while in SCL Utility does two things: it terminates the command with a carriage return and automatically sends the entire string. Try the example sequence below. In this example, note that <ENTER> means press the ENTER key on your keyboard, which is the same as terminating the command with a carriage return.

IMPORTANT: We recommend practicing with SCL commands with no load attached to the motor shaft. You want the motor shaft to spin freely during startup to avoid damaging mechanical components in your system.

AC25<ENTER>	Set accel rate to 25 rev/sec/sec.
DE25<ENTER>	Set decel rate to 25 rev/sec/sec
VE5<ENTER>	Set velocity to 5 rev/sec
FL20000<ENTER>	Move the motor 20000 steps in the CW direction.

If your motor didn't move after sending the FL20000 check the LEDs on your drive to see if there is an error present. If so send the AR command (AR<ENTER>) to clear the alarm. If after clearing the alarm you see a solid green LED it means the drive is disabled. Enable the drive by sending the ME command (ME<ENTER>) and verify that the you see a steady, flashing green LED. Then try the above sequence again.

Here is another sample sequence you can try.

JA10<ENTER>	Set jog accel rate to 10 rev/sec/sec
JL10<ENTER>	Set jog decel rate to 10 rev/sec/sec
JS1<ENTER>	Set jog speed to 1 rev/sec
CJ<ENTER>	Commence jogging

CS-1<ENTER> Change jog speed to 1 rev/sec in CCW direction

SJ<ENTER> Stop jogging

In the above sequence notice that the motor ramps to the new speed set by CS. This ramp is affected by the JA and JL commands. Try the same sequence above with different JA, JL, JS, and CS values to see how the motion of the motor shaft is affected.

10.3 Command Summary

This section contains a set of tables that list all of the Host Commands available with your drive. In each table there are a number of columns that give information about each command.

- "Command" shows the command's two-letter Command Code.
- "Description" shows the name of each command.
- "NV" designates which commands are Non-volatile: that is, which commands are saved in non-volatile memory when the SA (Save) command is sent to the drive. Note that certain commands (PA, PB, PC, PI, and PM) save their parameter data to non-volatile memory immediately upon execution, and need not be followed by an SA command.
- "Write only" or "Read only" is checked when a command is not both Read/Write compatible.
- "Immediate" designates an immediate command (all other commands are buffered).
- "Compatibility" shows which drives use each of the commands.

The different categories for these tables - Motion, Servo, Configuration, I/O, Communications, Q Program, Register - are set up to aid you in finding particular commands quickly.

- "Motion" commands have to do with the actual shaft rotation of the step or servo motor.
- "Servo" commands cover servo tuning parameters, enabling / disabling the motor, and filter setup.
- "Configuration" commands pertain to setting up the drive and motor for your application, including tuning parameters for your servo drive, step resolution and anti-resonance parameters for your step motor drive, etc.
- "I/O" commands are used to control and configure the inputs and outputs of the drive.
- "Communications" commands have to do with the configuration of the drive's serial ports.
- "Q Program" commands deal with programming functions when creating stored programs for your Q drive.
- "Register" commands deal with data registers. Many of these commands are only compatible with Q drives.

10.3.1 Motion Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
AC	Accel Rate	·				All drives
AM	Accel Max	·				All drives
CJ	Commence Jogging		·			All drives
DC	Distance for FC, FM, FO, FY	·				All drives
DE	Decel Rate	·				All drives
DI	Distance or Position	·				All drives
ED	Encoder Direction	·				Servos and steppers with encoder feedback
EF	Encoder Function	·				Servos and steppers with encoder feedback
EG	Electronic Gearing	·				All drives
EH	Extended Homing		·			All Step-Servo drives and M2 Servo drives
EI	Input Noise Filter	·				All drives
EP	Encoder Position					Servos and steppers with encoder feedback

FC	Feed to Length with Speed Change		·			All drives
FD	Feed to Double Sensor		·			All drives
FE	Follow Encoder		·			All drives
FH	Find Home		·			All Step-Servo drives and M2 Servo drives
FL	Feed to Length		·			All drives
FM	Feed to Sensor with Mask Dist.		·			All drives
FO	Feed to Length & Set Output		·			All drives
FP	Feed to Position		·			All drives
FS	Feed to Sensor		·			All drives
FY	Feed to Sensor with Safety Dist.		·			All drives
HA	Homing Acceleration	·				All Step-Servo drives and M2 Servo drives
HC	Hard Stop Current	·				All Step-Servo drives
HL	Homing Deceleration	·				All Step-Servo drives and M2 Servo drives
HO	Homing Offset	·				All Step-Servo drives and M2 Servo drives
HS	Hard Stop Homing		·			All Step-Servo drives
HV	Homing Velocity	·				All Step-Servo drives and M2 Servo drives
HW	Hand Wheel		·			All drives
JA	Jog Accel/Decel rate	·				All drives
JC	Velocity mode second speed	·				All drives
JD	Jog Disable		·			All drives
JE	Jog Enable		·			All drives

JL	Jog Decel rate	·				All drives
JM	Jog Mode	·				Al drives (see JM command)
JS	Jog Speed	·				All drives
MD	Motor Disable		·			All drives
ME	Motor Enable		·			All drives
MR	Micro step Resolution	·				Stepper drives only
PA	Power-up Accel Current	·				STM stepper drives only
SD	Set Direction	·				STM stepper drives with Flex I/O only
SH	Seek Home		·			All drives
SJ	Stop Jogging		·		·	All drives
SM	Stop the Move		·			Q drives only
SP	Set Absolute Position					All drives
ST	Stop Motion		·		·	All drives
VC	Velocity for Speed Change (FC)	·				All drives
VE	Velocity Setting (For Feed Commands)	·				All drives
VM	Velocity Max	·				All drives
WM	Wait on Move		·			Q drives only
WP	Wait on Position		·			Q drives only

10.3.2 Servo Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
CN	Second Control Mode	·				M2 servo drives only
CO	Node ID/ IP Address Series Number	·				M2 servo drives only
CP	Change Peak Current	·				Servo drives only
DD	Default Display Item of LEDs	·				M2 servo drives only
DS	Switching Electronic Gearing	·				M2 servo drives only
EN	Numerator of Electronic Gearing Ratio	·				M2 servo drives only
EP	Encoder Position					Servo drives only
EU	Denominator of Electronic Gearing Ratio	·				M2 servo drives only
FA	Function of the Single-ended Analog Input	·				M2 servo drives only
GC	Current Command	·			·	Servo drives only
GG	Controller Global Gain Selection	·				M2 servo drives only
IC	Immediate Current Command			·	·	Servo drives only
IE	Immediate Encoder Position			·	·	Servo drives only
IQ	Immediate Actual Current			·	·	Servo drives only
IX	Immediate Position Error			·	·	Servo drives only
JC	Eight Jog Velocities	·				M2 servo drives only
KC	Overall Servo Filter	·				Servo drives only
KD	Differential Constant	·				Servo drives only
KE	Differential Filter	·				Servo drives only
KF	Velocity Feedforward Constant	·				Servo drives only
KI	Integrator Constant	·				Servo drives only
KJ	Jerk Filter Frequency	·				SV7 Servo drives only
KK	Inertia Feedforward Constant	·				Servo drives only
KP	Proportional Constant	·				Servo drives only
KV	Velocity Feedback Constant	·				Servo drives only
MS	Control Mode Selection	·				M2 servo drives only
PF	Position Fault	·				Servo drives, drives with encoder feedback
PH	Inhibition of the pulse command	·				M2 servo drives only
PK	Parameter Lock	·				M2 servo drives only
PL	Position Limit	·				Servo drives only
PP	Power-Up Peak Current	·				Servo drives only
PV	Second Electronic Gearing	·				M2 servo drives only
TV	Torque Ripple	·				M2 servo drives only
VI	Velocity Integrator Constant	·				Servo drives only
VP	Velocity Mode Proportional Constant	·				Servo drives only
VR	Velocity Ripple	·				M2 servo drives only

10.3.3 Configuration Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
AL	Alarm Code			•	•	All drives
AR	Alarm Reset		•		•	All drives
BD	Brake Disengage Delay time	•				All drives
BE	Brake Engage Delay time	•				All drives
BS	Buffer Status			•	•	All drives
CA	Change Acceleration Current	•				STM stepper drives only
CC	Change Current	•				All drives
CD	Idle Current Delay	•				Stepper drives only
CF	Anti-resonance Filter Frequency	•				Stepper drives only
CG	Anti-resonance Filter Gain	•				Stepper drives only
CI	Change Idle Current	•				Stepper drives only
CM	Control mode	•				All drives
CP	Change peak current	•				Servo drives only
DA	Define Address	•				All drives
DL	Define Limits	•				All drives
DP	Dumping Power	•				SS drives only
DR	Data Register for Capture		•			Q servo drives only
ED	Encoder Direction	•				Servo drives, drives with encoder feedback
ER	Encoder or Resolution	•				Servo drives, drives with encoder feedback
HG	4th Harmonic Filter Gain	•				Stepper drives only
HP	4th Harmonic Filter Phase	•				Stepper drives only
IA	Immediate Analog			•	•	All drives
ID	immediate Distance			•	•	All drives
IE	Immediate Encoder			•	•	Servo drives, drives with encoder feedback
IF	Immediate Format	•			•	All drives
IQ	Immediate Current			•	•	Servo drives only
IP	Immediate Position			•	•	All drives
IT	Immediate Temperature			•	•	All drives
IU	Immediate Voltage			•	•	All drives
IV	Immediate Velocity			•	•	All drives
LP	Software Limit CW					All Step-Servo drives and M2 Servo drives
LM	Software Limit CCW					All Step-Servo drives and M2 Servo drives
LV	Low Voltage Threshold	•				All drives
MD	Motor Disable				•	All drives
ME	Motor Enable				•	All drives
MN	Model Number			•	•	All drives
MO	Motion Output	•				All drives
MR	Micro step Resolution	•				All drives (deprecated - see EG

MV	Model & Revision			•	•	All drives except Blu servos
OF	On Fault		•			Q drives only
OI	On Input		•			Q drives only
OP	Option Board	·		•	•	All drives
PA	Power-up Acceleration Current	·				
PC	Power up Current	·				All drives
PD	In Position Counts	·				All Step-Servo drives and M2 Servo drives
PE	In Position Timing	·				All Step-Servo drives and M2 Servo drives
PF	Position Fault	·				Servo drives, drives with encoder
PI	Power up Idle Current	·				Stepper drives only
PL	In Position Limit	·				Servo drives only
PM	Power up Mode	·				All drives
PP	Power up peak current	·				Servo drives only
PW	Pass Word		•			Q drives only
RE	Restart / Reset		•		•	All drives
RL	Register Load				•	All drives
RS	Request Status			•	•	All drives
RV	Revision Level			•	•	All drives
SA	Save all NV Parameters		•			All drives
SC	Status Code			•	•	
SD	Set Direction	·				STM stepper drives with Flex I/O
SF	Step Filter Frequency	·				Stepper drives only
SI	Enable Input usage	·				All drives
SK	Stop & Kill		•		•	All drives
TT	Pulse Complete Timing	·				All Step-Servo drives and M2 Servo drives
ZC	Regen Resistor Continuous Wattage	·				BLuAC5 and STAC6 drives only
ZR	Regen Resistor Value	·				BLuAC5 and STAC6 drives only
ZT	Regen Resistor Peak Time	·				BLuAC5 and STAC6 drives only

10.3.4 I/O Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
AD	Analog Deadband	·				All stepper drives and SV servo drives
AF	Analog Filter	·				All drives
AG	Analog Velocity Gain	·				All stepper drives and SV servo drives
AI	Alarm Input usage	·				All drives
AN	Analog Torque Gain	·				All Step-Servo drives and M2 Servo drives
AO	Alarm Output usage	·				All drives
AP	Analog Position Gain	·				All drives
AS	Analog Scaling	·				All stepper drives and SV servo drives
AT	Analog Threshold	·				All drives

AV	Analog Offset	·				All drives
AZ	Analog Zero (Auto Zero)		·			All drives
BD	Brake Disengage Delay time	·				All drives
BE	Brake Engage Delay time	·				All drives
BO	Brake Output usage	·				All drives
DL	Define Limits	·				All drives
EI	Input Noise Filter	·				All drives
FI	Filter Input	·				All drives **(Note: not NV on Blu servos)**
FX	Filter Selected Inputs					Blu, STAC5, STAC6, SVAC3
IH	Immediate High Output		·		·	All drives
IL	Immediate Low Output		·		·	All drives
IO	Output Status				·	All drives
IS	Input Status request			·	·	All drives
MO	Motion Output	·				All drives
OI	On Input		·			Q drives only
SI	Enable Input usage	·				All drives
SO	Set Output		•			All drives
TI	Test Input		•			Q drives only
TO	Tach Output	·				TSM drives only
WI	Wait on Input		•			All drives

10.3.5 Communications Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
BR	Baud Rate	•				All drives
BS	Buffer Status				•	All drives
CE	Communications Error				•	All drives
IF	Immediate Format	•			•	All drives
PB	Power up Baud Rate	•				All drives
PR	Protocol	•				All drives
TD	Transmit Delay	•				All drives

10.3.6 Q Program Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
AX	Alarm Reset		•			All drives
MT	Multi-Tasking					Q drives only
NO	No Operation		•			Q drives only
OF	On Fault		•			Q drives only
OI	On Input		•			Q drives only
PS	Pause		•			All drives
QC	Queue Call		•			Q drives only
QD	Queue Delete		•			Q drives only
QE	Queue Execute		•		•	Q drives only

QG	Queue Goto		•			Q drives only
QJ	Queue Jump		•			Q drives only
QK	Queue Kill		•			Q drives only
QL	Queue Load		•		•	Q drives only
QR	Queue Repeat		•			Q drives only
QS	Queue Save		•		•	Q drives only
QU	Queue Upload			•	•	Q drives only
QX	Queue Load & Execute		•			Q drives only
SM	Stop Move		•			Q drives only
SS	Send String		•			All drives
TI	Test Input		•			Q drives only
WD	Wait Delay using Data Register		•			Q drives only
WI	Wait for Input		•			All drives
WM	Wait for Move to complete		•			Q drives only
WP	Wait for Position in complex move		•			Q drives only
WT	Wait Time		•			Q drives only

10.3.7 Register Commands

Command	Description	NV	write only	read only	Immediate	Compatibility
CR	Compare Register		•			Q drives only
DR	Data Register for Capture		•			Q drives only
RC	Register Counter		•			Q drives only
RD	Register Decrement		•			Q drives only
RI	Register Increment		•			Q drives only
RL	Register Load				•	Q drives only
RM	Register Move		•			Q drives only
RR	Register Read		•			Q drives only
RU	Register Upload		•		•	
RW	Register Write		•			Q drives only
RX	Register Load					Q drives only
R+	Register Addition		•			Q drives only
R-	Register Subtraction		•			Q drives only
R*	Register Multiplication		•			Q drives only
R/	Register Division		•			Q drives only
R&	Register Logical AND		•			Q drives only
R|	Register Logical OR		•			Q drives only
TR	Test Register		•			Q drives only
TS	Time Stamp read		•			Q drives only

10.4 Host Command Reference

11 Appendix B: Q Programmer Reference

The use of SCL commands with MOONS' dates back many years. A few years ago a new control platform

was created that expanded the use of SCL commands and allowed users to create stored programs with SCL commands. These programs could be saved in a drive's non-volatile memory, and the drive could run these programs stand-alone, or without a permanent connection to the host. This expansion of SCL's capabilities was called Q, and since that time MOONS' has continued to expand the offering of drives with the Q motion controller built in. By combining the ability to run a sophisticated, single-axis motion control program stand-alone and the ability to communicate serially to a host device, Q drives offer a high level of flexibility and functionality to the machine designer and system integrator. The characteristic as follows;

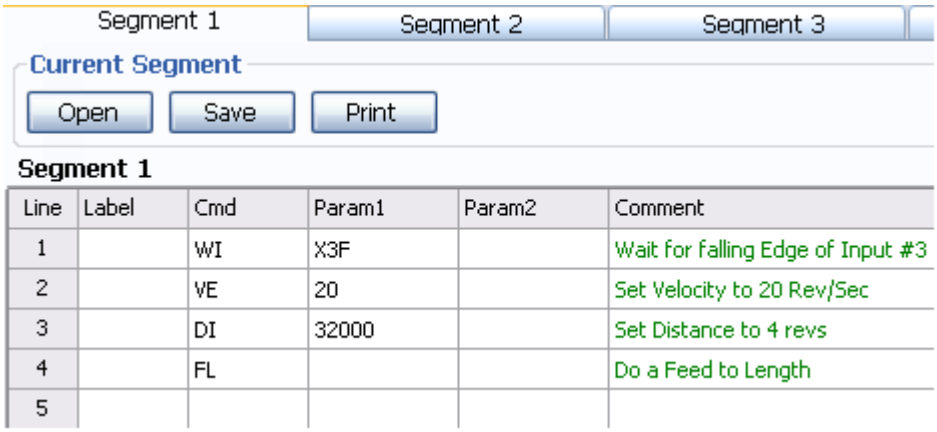
Single-Axis motion control
Stand Alone
Multi-task
Conditional Processing
Math Calculation
Data register manipulation

11.1 Sample Command Sequences

What follows are sequences of commands that give examples of how to create motion and logic within a program. All of the commands in this section are buffered-type commands.

11.1.1 Feed to Length

The FL (Feed to Length) command is used for relative or incremental moves. When executed, the motor will move a fixed distance, using linear acceleration and deceleration ramps and a maximum velocity. These move parameters are set using the DI (Distance), AC (Acceleration), DE (Deceleration), and VE (Velocity) commands. The direction of the move is determined by the sign of the DI parameter. "DI32000" is 32000 counts in the CW direction, whereas "DI-32000" is 32000 counts in the CCW direction.

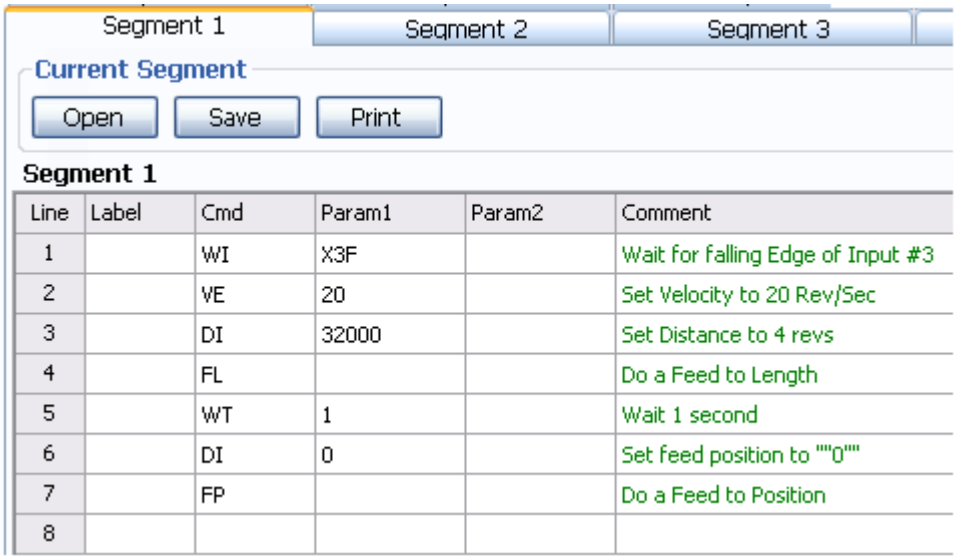


Line	Label	Cmd	Param1	Param2	Comment
1		WI	X3F		Wait for falling Edge of Input #3
2		VE	20		Set Velocity to 20 Rev/Sec
3		DI	32000		Set Distance to 4 revs
4		FL			Do a Feed to Length
5					

Here is a sample sequence showing a move of 80000 counts, with a velocity of 20 rps, and accel/decel rates of 500 rps/s. The FL command initiates the move. Also, the order of the commands is not significant, except that any changes to the move parameters must be done before the FL command.

11.1.2 Feed to Position

The FP (Feed to Position) command is used for absolute moves. When executed, the motor will move to a position, with linear acceleration and deceleration ramps and a maximum velocity, based on the internal motor position of the drive. The move parameters are set using the AC, DE, VE and DI commands. In the case of the FP command, the DI command sets the motor position, not the relative move distance.


Segment 1 | Segment 2 | Segment 3

Current Segment

Open | Save | Print

Segment 1

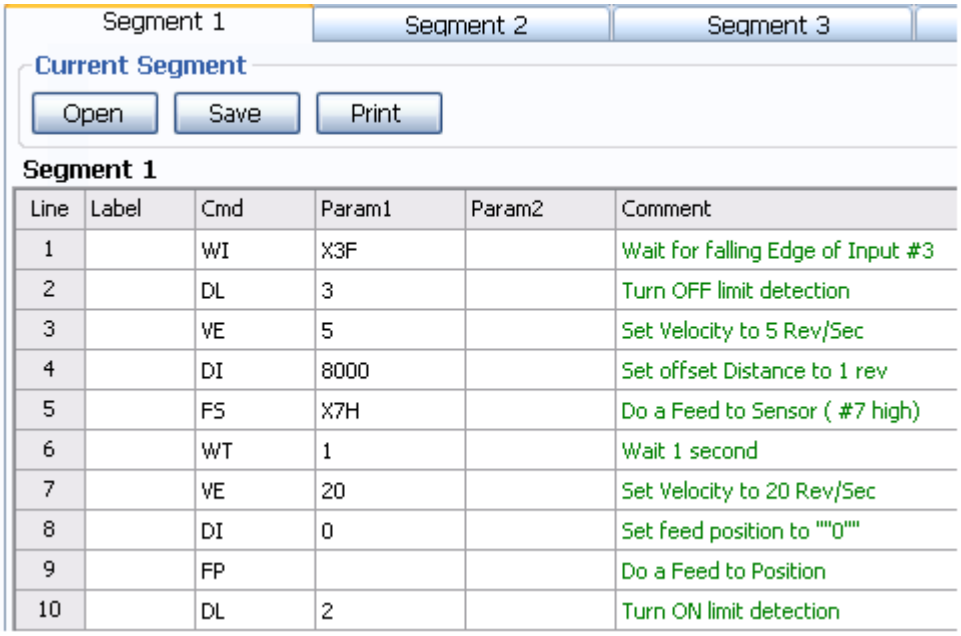
Line	Label	Cmd	Param1	Param2	Comment
1		WI	X3F		Wait for falling Edge of Input #3
2		VE	20		Set Velocity to 20 Rev/Sec
3		DI	32000		Set Distance to 4 revs
4		FL			Do a Feed to Length
5		WT	1		Wait 1 second
6		DI	0		Set feed position to ""0""
7		FP			Do a Feed to Position
8					

Here is a sample sequence showing a move to motor position 32000 counts (motor may move CW or CCW depending on the actual motor position before the start of the move), with a velocity of 20 rps and accel/decel rates of 500 rps/s.

Another command to keep in mind when using absolute moves is the SP (Set Position) command. This command allows you to zero the motor position at any time, by entering “SP0”, or to set the motor position to another value. The parameter in the SP command is encoder counts. For example with a 2000 line encoder on the motor, an “SP5000” command would set the current motor position to 2.5 revolutions CW from the zero position.

11.1.3 Feed to Sensor

The FS (Feed to Sensor) command causes the motor to move at a fixed velocity until an input changes state. When the designated input changes state the motor decelerates to a stop. The parameters of the move are set by the AC, DE, VE and DI commands. In an FS command, the DI command sets both the distance in which the motor should stop after the input changes state and the direction of the move. Parameters for the FS command are the input number (0-7) and the input state the drive should look for: H (high), L (low), R (rising edge), or F (falling edge).


Segment 1 | Segment 2 | Segment 3

Current Segment

Open | Save | Print

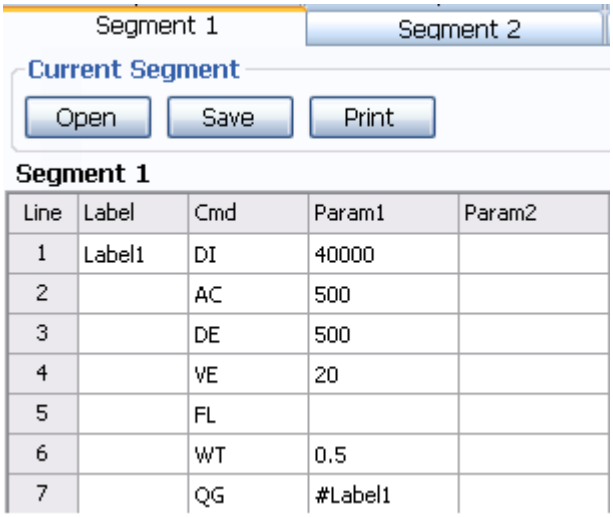
Segment 1

Line	Label	Cmd	Param1	Param2	Comment
1		WI	X3F		Wait for falling Edge of Input #3
2		DL	3		Turn OFF limit detection
3		VE	5		Set Velocity to 5 Rev/Sec
4		DI	8000		Set offset Distance to 1 rev
5		FS	X7H		Do a Feed to Sensor (#7 high)
6		WT	1		Wait 1 second
7		VE	20		Set Velocity to 20 Rev/Sec
8		DI	0		Set feed position to ""0""
9		FP			Do a Feed to Position
10		DL	2		Turn ON limit detection

Above is an example where the motor will move in the clockwise direction, starting off with an acceleration rate of 500 rps/s and a maximum speed of 5 rps, until drive input X7 goes high, at which point the drive will use the distance set in the DI command (8000 counts) and the deceleration rate set in the DE command (500 rps/s) to bring the motor to a stop.

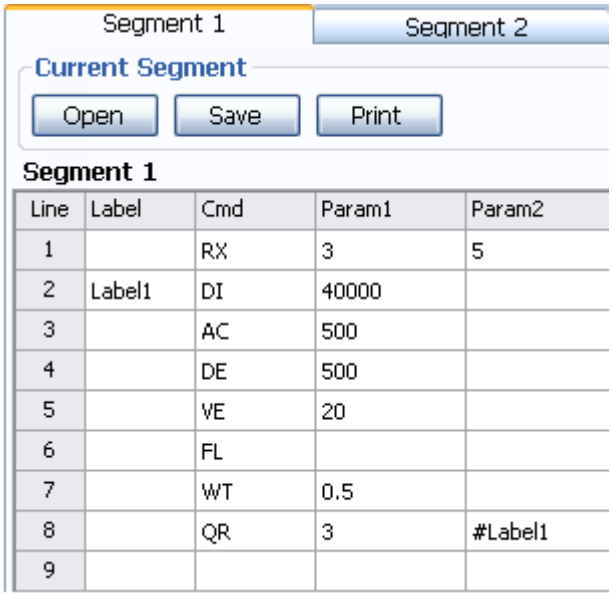
11.1.4 Looping

There are two ways to accomplish looping, or repeat loops, within a pro-gram. The first method accomplishes an infinite loop and uses the QG (Queue Goto) command. The parameter for this command is a line number in the segment, and whenever the sequence gets to the QG command the segment will jump to the designated line.



Line	Label	Cmd	Param1	Param2
1	Label1	DI	40000	
2		AC	500	
3		DE	500	
4		VE	20	
5		FL		
6		WT	0.5	
7		QG	#Label1	

In the example to above, the sequence contains an FL command, with related parameter commands ahead of it (AC, DE, DI, VE). After the FL command is a WT (Wait Time) command with a time of 0.5 seconds, and then a QG command that points to line 1. This sequence will loop forever now, with the segment always starting at line one after it executes the QG command.



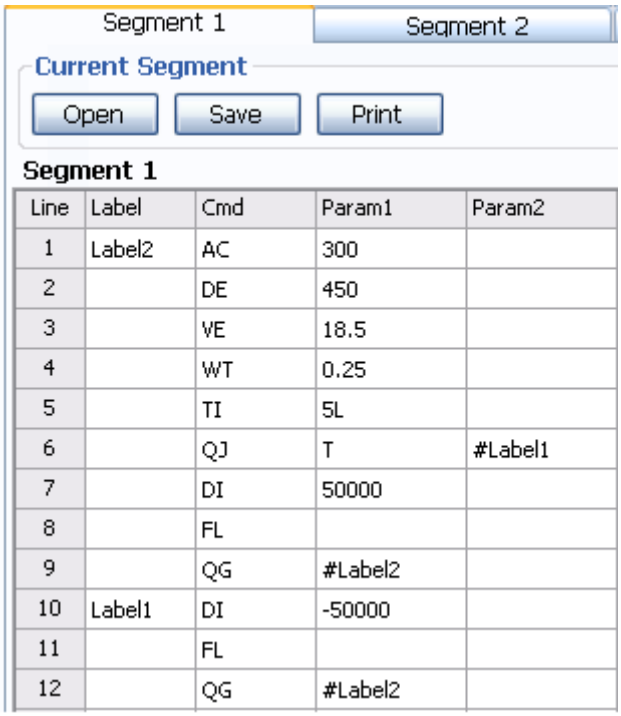
Line	Label	Cmd	Param1	Param2
1		RX	3	5
2	Label1	DI	40000	
3		AC	500	
4		DE	500	
5		VE	20	
6		FL		
7		WT	0.5	
8		QR	3	#Label1
9				

The second method for looping utilizes the QR (Queue Repeat) command. It works by jumping to a given segment line for the number of times indicated in a user-defined data register. Any user-defined data register will work. In the example to the right, the QG command from the previous example has been replaced with the QR command, and parameters have been added. In this sequence the segment will jump to line 2 for the

number of times indicated in register 3. Notice on line 1 of the segment that data register 3 has been loaded (using the RX command) with the value 5. Therefore, the FL command in this example (as well as the DI, AC, DE, VE and WT commands) will repeat five times.

11.1.5 Branching

Branching in a program is done using the QJ (Queue Jump) command. Branching is different than looping in that a branch (or jump) is done based on a tested condition. The QJ command will always work in conjunction with one other command: TI (Test Input), TR (Test Register), or CR (Compare Register).



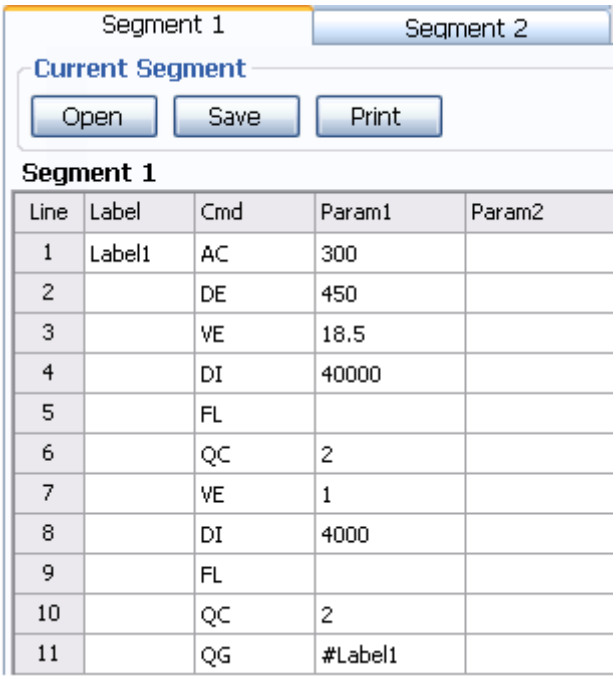
Segment 1

Line	Label	Cmd	Param1	Param2
1	Label2	AC	300	
2		DE	450	
3		VE	18.5	
4		WT	0.25	
5		TI	5L	
6		QJ	T	#Label1
7		DI	50000	
8		FL		
9		QG	#Label2	
10	Label1	DI	-50000	
11		FL		
12		QG	#Label2	

Let's say we have an application with two possible moves. We always want to make a CW move, unless input X5 is low in which case we want to make a CCW move. In this example we set all of the move parameters except distance at the top of the segment. We set accel to 300 rps/s, decel to 450 rps/s, and velocity to 18.5 rps. There is a WT (Wait Time) of 0.25 seconds so that we may have a noticeable delay between moves. Then, we test input X5 to see if it's low using the TI (Test Input) command. If it is true (i.e. input X5 is low), we branch (using QJ) to line 10, set the distance to -50000 counts and make a CCW move. Otherwise the program proceeds to line 7, sets the distance to 50000 counts and makes the CW move. To keep from doing the CCW move right after the CW move, and to repeat the segment forever QG commands are placed after each FL command.

11.1.6 Calling

Calling is similar to using sub-routines. The QC (Queue Call) command allows us to exit a segment, execute another segment, and then return to the original segment to the line where the "call" was initiated. This is useful when we have a sequence of commands that is used over and over within a program. Rather than repeatedly pro-gram these commands into our segment(s), we locate the frequently-used sequence in its own segment, and then call that segment whenever we need to.


Segment 1 | Segment 2

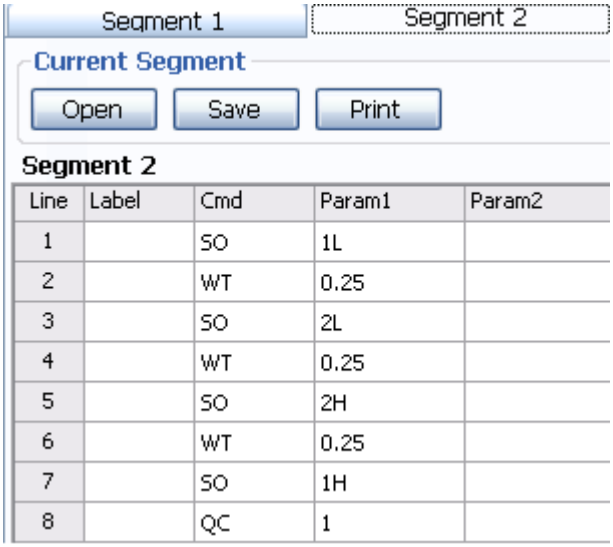
Current Segment

Open | Save | Print

Segment 1

Line	Label	Cmd	Param1	Param2
1	Label1	AC	300	
2		DE	450	
3		VE	18.5	
4		DI	40000	
5		FL		
6		QC	2	
7		VE	1	
8		DI	4000	
9		FL		
10		QC	2	
11		QG	#Label1	

In this example we are making two distinct moves (FL), one fast move and one slow move. After each move we'd like to turn 2 outputs on and off. To accomplish this using the QC command, we must program two segments. In this example, segment 1 is the primary (or calling) segment, and in it we program the two distinct FL commands. We are using the same accel and decel rates for the two moves, but the velocities and distances change. After each move we'd like to set outputs Y1 and Y2 on then off, and rather than entering the necessary commands to do this after each FL command in segment 1, we place the commands in segment 2 and then use the QC command to call it.


Segment 1 | Segment 2

Current Segment

Open | Save | Print

Segment 2

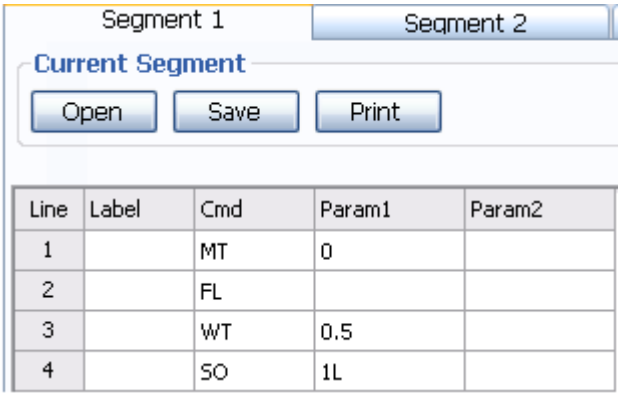
Line	Label	Cmd	Param1	Param2
1		SO	1L	
2		WT	0.25	
3		SO	2L	
4		WT	0.25	
5		SO	2H	
6		WT	0.25	
7		SO	1H	
8		QC	1	

In segment 2 we place the desired SO (Set Output) commands that turn output Y1 on, then output Y2 on, then output Y2 off and finally output Y1 off. Notice we've also placed WT (Wait Time) commands of 0.25 seconds between each SO command to make the changing output states more noticeable. Segments 1 and 2 work in condition when segment 1 reaches its first QC command (with the parameter "2" indicating segment 2). At this moment the program calls segment 2 to execute its sequence of commands. Notice at the end of the sequence in segment 2 we've placed a QC command with no parameter. A QC command with no parameter means return to the original, calling line and segment. So what happens then is the program returns to segment 1, completes the second move, calls segment 2 again, returns to segment 1 once more,

and then starts the process over by looping to line 1 (“QG1”).

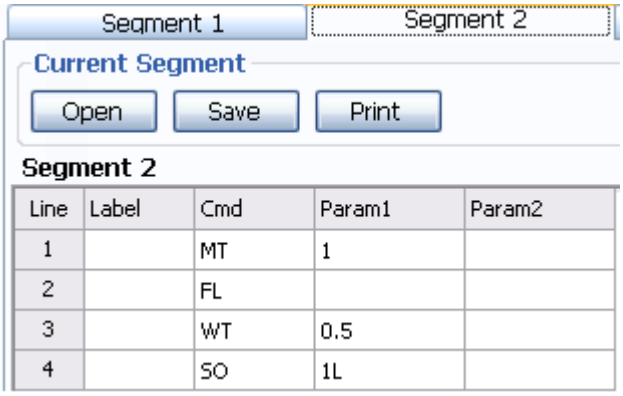
11.1.7 Multi-tasking

The multi-tasking feature of Q drives allows you to initiate a move command (FL, FP, CJ, FS, etc.) and proceed to execute other commands without waiting for the move command to finish. Without multi-tasking (or more accurately with multi-tasking turned off), a Q drive always executes commands in succession by waiting for the completion of a particular command before moving on to the next command. In the case of move commands, this means waiting for the move to finish before executing subsequent commands. For example, if you have an FL command (Feed to Length - incremental move) followed by an SO command (Set Output), the drive will wait to finish the motor move before setting the drive’s digital output.
With multi-tasking turned on, a Q drive initiates a move command and then immediately proceeds to execute subsequent commands. For example, doing the same FL and SO commands as above, but this time with multi-tasking turned on, the drive will initiate the move command and immediately proceed to execute the set output command without waiting for the move command to finish. Multi-tasking is turned on and off with the MT command. “MT1” turns multi-tasking on, and “MT0” turns it off.
To illustrate the use of the MT command some more, here are a couple of sample command sequences.



Line	Label	Cmd	Param1	Param2
1		MT	0	
2		FL		
3		WT	0.5	
4		SO	1L	

In the above command sequence to the right, notice that multi-tasking is turned off, “MT0”. When this sequence is executed by a drive, the FL (Feed to Length) incremental move will complete before the drive waits 0.5 seconds (WT0.50) and then sets output 1 low (SOY1L).



Line	Label	Cmd	Param1	Param2
1		MT	1	
2		FL		
3		WT	0.5	
4		SO	1L	

In the above command sequence to the right, notice that multi-tasking is turned on, “MT1”. When this sequence is executed by the drive, the drive will not wait for the FL command to complete before executing the WT and SO commands. In other words, the drive will initiate the FL command, then wait 0.50 seconds, and then set output 1 low. If the last distance set by the DI command is sufficiently long, the drive’s output 1 will be set low before the FL command has completed.
This example is actually quite basic, even though it illustrates the function of multi-tasking well. If you try these sequences with your drive, make sure the last DI command is sufficiently large enough to see a

noticeable difference in when the drive sets the output.

NOTE: Because it is physically impossible for a motor to make two moves at the same time, move commands are always blocked even with Multi-tasking turned on. For example, if you have Multi-tasking turned on and the program has two move commands in a row, the drive will wait to execute the second move command until the first move command is finished.

12 Appendix C: CANopen Reference

12.1 CANopen Communication

CANopen is a communication protocol and device profile specification for embedded systems used in automation. In terms of the OSI model, CANopen implements the layers above and including the network layer. The CANopen standard consists of an addressing scheme, several small communication protocols and an application layer defined by a device profile. The communication protocols have support for network management, device monitoring and communication between nodes, including a simple transport layer for message segmentation/desegmentation. The lower level protocol implementing the data link and physical layers is usually Controller Area Network (CAN)

The basic CANopen device and communication profiles are given in the CiA 301 specification released by CAN in Automation. [1] Profiles for more specialized devices are built on top of this basic profile, and are specified in numerous other standards released by CAN in Automation, such as CiA 401[2] for I/O-modules and CiA 402[3] for motion control.

12.2 Why CANopen

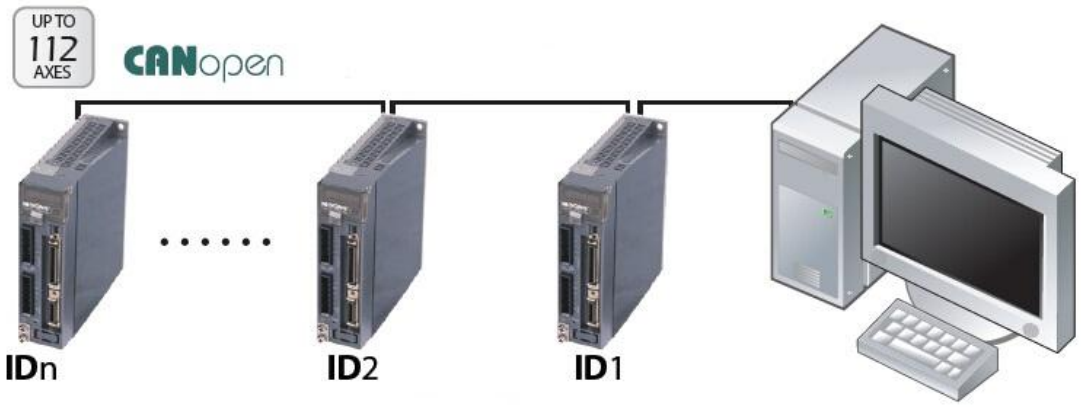
Multi-axis Control

Up to 127 axis can be supported via CANopen, and the maximum communication baud rate is up to 1Mbps.

A further advantage with CAN is the Multi-Master Capability. This means that each user on the bus has the same access rights. The access authorization alone controls the users among one another via the priority of the communication objects and their identifiers (arbitration). This allows direct communication between the individual users without a time-consuming "detour" over a central master.

Easy to Wiring

A shielded twisted pair cable is be used as the bus cable. Less cable will cause less error, reduce the wiring cost, labor cost, whilst maintaining availability and minimizing cost.



12.3 CANopen Example Programs

12.3.1 Profile Position Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID DLC Data

$0603 $8 $2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on

$0603 $8 $2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 ‘Operation Enabled
**** Set to Profile Position Mode ****
$0603 $8 $2F $60 $60 $00 $01 $00 $00 $00 ‘Set to Profile Position Mode
**** Set Motion Parameters ****
$0603 $8 $23 $81 $60 $00 $F0 $00 $00 $00 ‘Set Profile Velocity to 1 rps
$0603 $8 $23 $83 $60 $00 $58 $02 $00 $00 ‘Set Acceleration to 100 rps/s
$0603 $8 $23 $84 $60 $00 $58 $02 $00 $00 ‘Set Deceleration to 100 rps/s
Single Move Absolute
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603 $8 $2B $40 $60 $00 $1F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 ‘Clear New Set Point Bit
Single Move Relative
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603 $8 $2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit
Multiple Move, Stopping between Moves
$0603 $8 $23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603 $8 $2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit
$0603 $8 $23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603 $8 $2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit
Multiple Move, Continuous Motion
$0603 $8 $23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603 $8 $2B $40 $60 $00 $5F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $4F $02 $00 $00 ‘Clear New Set Point Bit
$0603 $8 $23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603 $8 $2B $40 $60 $00 $5F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $4F $02 $00 $00 ‘Clear New Set Point Bit
Multiple Move, Immediate Change in Motion
$0603 $8 $23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603 $8 $2B $40 $60 $00 $7F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $6F $02 $00 $00 ‘Clear New Set Point Bit
$0603 $8 $23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603 $8 $23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603 $8 $2B $40 $60 $00 $7F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603 $8 $2B $40 $60 $00 $6F $02 $00 $00 ‘Clear New Set Point Bit

12.3.2 Profile Velocity Mode

**** Enable Motor Power - CiA 402 State Machine ****
ID DLC Data
$0603 $8 $2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603 $8 $2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603 $8 $2B $40 $60 $00 $0F $01 $00 $00 ‘Operation Enabled; Motion Halted
**** Set to Profile Velocity Mode ****
$0603 $8 $2F $60 $60 $00 $03 $00 $00 $00 ‘Set to Profile Velocity Mode
**** Set Motion Parameters ****
$0603 $8 $23 $FF $60 $00 $F0 $00 $00 $00 ‘Set Target Velocity to 1 rps
$0603 $8 $23 $83 $60 $00 $58 $02 $00 $00 ‘Set Acceleration to 100 rps/s
$0603 $8 $23 $84 $60 $00 $58 $02 $00 $00 ‘Set Deceleration to 100 rps/s
**** Start/Stop Motion ****
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 ‘Motion Starts
$0603 $8 $23 $FF $60 $00 $60 $09 $00 $00 ‘Change Target Velocity to 10 rps
$0603 $8 $2B $40 $60 $00 $0F $01 $00 $00 ‘Motion Halts

12.3.3 Homing Mode

**** Enable Motor Power - CiA 402 State Machine ****
ID DLC Data
$0603 $8 $2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603 $8 $2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 ‘Operation Enabled
**** Set to Homing Mode ****
$0603 $8 $2F $60 $60 $00 $06 $00 $00 $00 ‘Set to Homing Mode
$0603 $8 $2F $98 $60 $00 $13 $00 $00 $00 ‘Set Homing Method to 19
**** Set Motion Parameters ****
$0603 $8 $23 $9A $60 $00 $58 $02 $00 $00 ‘Set Homing Acceleration to 100rps/s
$0603 $8 $23 $99 $60 $01 $F0 $00 $00 $00 ‘Set Homing Velocity (Search for Switch) to 1rps
$0603 $8 $23 $99 $60 $02 $78 $00 $00 $00 ‘Set Index Velocity (Search for Index or Zero) to 0.5rps
$0603 $8 $23 $7C $60 $00 $40 $9C $00 $00 ‘Set Homing Offset to 40000 Steps
$0603 $8 $2F $01 $70 $00 $03 $00 $00 $00 ‘Set Homing Switch to Input 3
**** Start/Stop Homing ****
$0603 $8 $2B $40 $60 $00 $1F $00 $00 $00 ‘Homing Starts
$0603 $8 $2B $40 $60 $00 $1F $01 $00 $00 ‘Homing Stops

12.3.4 Normal Q Mode

**** Enable Motor Power - CiA 402 State Machine ****
ID DLC Data
$0603 $8 $2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603 $8 $2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 ‘Operation Enabled

**** Set to Normal Q Mode ****
$0603 $8 $2F $60 $60 $00 $FF $00 $00 $00 'Set to Normal Q Mode
$0603 $8 $2F $07 $70 $00 $01 $00 $00 $00 'Set Q Segment Number to 1
**** Start/Stop Q Program ****
$0603 $8 $2B $40 $60 $00 $1F $00 $00 $00 'Q Program Starts
$0603 $8 $2B $40 $60 $00 $1F $01 $00 $00 'Q Program Halts

12.3.5 Sync Q Mode

**** Enable Motor Power - CiA 402 State Machine ****
ID DLC Data
$0603 $8 $2B $40 $60 $00 $06 $00 $00 $00 'Ready to Switch on
$0603 $8 $2B $40 $60 $00 $07 $00 $00 $00 'Switched on
$0603 $8 $2B $40 $60 $00 $0F $00 $00 $00 'Operation Enabled
**** Set to Sync Q Mode ****
$0603 $8 $2F $60 $60 $00 $FE $00 $00 $00 'Set to Sync Q Mode
$0603 $8 $2F $07 $70 $00 $01 $00 $00 $00 'Set Q Segment Number to 1
$0603 $8 $23 $05 $10 $00 $80 $00 $00 $00 'Set Sync Pulse to 0x80
**** Start/Stop Q Program ****
$80 $0 'Q Program Starts
$0603 $8 $2B $40 $60 $00 $0F $01 $00 $00 'Q Program Halts

12.3.6 PDO Mapping

****Mapping TPDO2 ****
$0000 $2 $80 $03 'Return back to "PreOperation" Mode
$0603 $8 $23 $01 $18 $01 $80 $02 $00 $80 'Turn off the TPDO2
$0603 $8 $2F $01 $1A $00 $00 $00 $00 $00 'Set Number of Mapped objects to zero
$0603 $8 $23 $01 $1A $01 $10 $00 $41 $61 'Map object1 (0x6041) to TPDO2 subindex1.
$0603 $8 $23 $01 $1A $02 $20 $00 $0A $70 'Map object2 (0x700A) to TPDO2 subindex2.
$0603 $8 $2F $01 $1A $00 $02 $00 $00 $00 'Set Number of total Mapped objects to two
$0603 $8 $23 $01 $18 $01 $80 $02 $00 $00 'Turn on the TPDO2

12.4 Downloads

Eds Download	Link
CANopen User Manual	Link

13 Appendix D: Modbus/RTU Reference

The Modbus products of MOONS' are based on serial communication bus with Modbus/RTU.

Modbus communication protocol is a kind of industrial field bus communication protocol, which is the application layer on the OSI 7packet transport protocol. It defines a device controller which can identify the frame structure and information. It is independent of the physical medium and can be used over various networks.

Since Modbus is a master/slave protocol, that means only one node is a master and the others is slave node .Each device intended to communicate using Modbus is given a unique address. In serial networks,

only the node assigned as the Master may initiate a command.

A Modbus command contains the Modbus address of the device it is intended for. Only the intended device will act on the command, even though other devices might receive it (an exception is specific broadcast able commands sent to node 0 which are acted on but not acknowledged). All Modbus commands contain checksum information, to allow the recipient to detect transmission errors. The basic Modbus commands can instruct an RTU to change the value in one of its registers, control or read an I/O port, and command the device to send back one or more values contained in its registers.

13.1 Communication Address

In the network system, each drive requires a unique drive address. Only the drive with the matching address will responded to the host command. In Modbus network, address “0” is the broadcast address. It cannot be used for individual drive’s address. Modbus RTU/ASCII can set drive address from 1 to 31.

13.2 Data Encode

Big-endian: The most significant byte (MSB) value is stored at the memory location with the lowest address; the next byte value in significance is stored at the following memory location and so on. This is akin to Left-to-Right reading in hexadecimal order.

For example: To store a 32bit data 0x12345678 into register address 40031 and 40032. 0x1234 will be defined as MSB, and 0x5678 as LSB. With big-endian system

Register 40031 = 0x1234

Register 40032 = 0x5678

When transfer 0x12345678, the first word will be 0x1234, and the second word will be 0x5678

Little-endian: The most significant byte (MSB) value is stored at the memory location with the highest address; the next byte value in significance is stored at the following memory location and so on. This is akin to Left-to-Right reading in hexadecimal order.

For example: To store a 32bit data 0x12345678 into register address 40031 and 40032. 0x5678 will be defined as MSB, and 0x1234 as LSB. With little-endian system

Register 40031 = 0x5678

Register 40032 = 0x1234

When transfer 0x12345678, the first words will be 0x5678, and the second words will be 0x1234

PR defines data transfer type.

13.3 Communication Baud Rate & Protocol

M2 Series AC Servo has a fixed communication data framing: 8, N, 1. Date bits: 8, parity checking: none, stop bit: 1.

BR and **PB** defines the communication baud rate.

In serial communication, the change of baud rate will NOT effect immediately, it will ONLY effects at next power up of the drive.

1 = 9600bps

2 = 19200bps

3 = 38400bps

4 = 57600bps

5 = 115200bps

13.4 Function Code

MOONS drives currently support following Modbus function code:

1) 0x03: Read holding registers
2) 0x04: Read input registers
3) 0x06: Write single registers
4) 0x10: Write multiple registers

13.4.1 Function Code 0X03, Reading Multiple Holding Registers

If we want to read encoder's actual position command to drive Node ID 1, the data address for encoder's actual position is register 40005. If the register value is in decimal numbers it will be 250000, and the transfer method is P-75 (PR) = 5, for big-endian transfer.

Communication details are as follows:

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	03H	1		Function Code	03H	1
Starting Data Address (Register 40005)	00H(High) 04H(Low)	2		Number of Data （In Byte）	04	1
Number of Data （In word）	00(High) 02(Low)	2		Content of Starting Data Address 40005	00H(High) 26H(Low)	2
CRC Check Low	85	1		Content of second Data Address 40006	25H(High) A0(Low)	2
CRC Check High	CA	1		CRC Check Low	01H	1
				CRC Check High	10H	1

Host Sending: 01 03 00 04 00 02 85 CA

Drive Reply: 01 03 04 00 26 25 A0 01 10

If error is occurred, drive reply format: 01 83 XX CRC_L CRC_H

Where XX = 01 ： Function code 03 unsupported

XX = 02 ： Incorrect reading on driving address or numbers

XX = 03 ： Reading register address out of range

XX = 04 ： Reading failure

13.4.2 Function Code 0x06, Writing Single Register

If we want to set motor rotary velocity 12.5 rps to drive node ID 11, the corresponding address is register 40030. The write in data value for the register will be 12.5 x 240 = 3000. In hexadecimal number, it is 12CH.

Communication Details are as follows:

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	0BH	1		Slave Address	0BH	1
Function Code	06H	1		Function Code	06H	1
Starting Data Address (Register 40030)	00H(High) 1DH(Low)	2		Starting Data Address (Register 40030)	00H(High) 1DH(Low)	2
Content of Data	01(High) 2C(Low)	2		Content of Data	01(High) 2C(Low)	2
CRC Check Low	19	1		CRC Check Low	19	1
CRC Check High	2B	1		CRC Check High	2B	1

Host Sending: 0B 06 00 1D 01 2C 19 2B

Drive Reply: 0B 06 00 1D 01 2C 19 2B

If error is occurred, drive reply format: 01 86 XX CRC_L CRC_H

Where XX = 01 ： Function code 06 unsupported

XX = 02 ： Incorrect writing on driving address or number

XX = 03 ： Writing register address out of range

XX = 04 ： Writing failure

13.4.3 Function Code 0X10, Writing Multiple Registers

If we writing target distance 30000 into drive NODE-ID 10, the correspondent register address will be 40031. Transfer into hexadecimal, it is 7530h.

Communication Details are as follows:

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	0AH	1		Slave Address	0AH	1
Function Code	10H	1		Function Code	10H	1
Starting Data Address (Register 40031)	00H(High) 1EH(Low)	2		Starting Data Address (Register 40031)	00H(High) 1EH(Low)	2
Number of Data （In word）	00H(High) 02H(Low)	2		Number of Data （In word）	00H(High) 02H(Low)	2
Number of Data （In byte）	04H	1		CRC Check Low	20	1
Content of first Data address	00(High) 00(Low)	2		CRC Check High	B5	1
Content of second Data address	75H(High) 30H(Low)	2				
CRC Check Low	70	1				
CRC Check High	8F	1				

Host Sending： 0A 10 00 1E 00 02 04 00 75 30 70 8F

Drive Reply： 0A 10 00 1E 00 02 20 B5

If error is occurred, drive reply format: 01 90 XX CRC_L CRC_H

Where XX = 01 ： Function code 10 unsupported

XX = 02 ： Incorrect reading on driving address or number

XX = 03 ： Reading register address out of range

XX = 04 ： Reading failure

13.5 Modbus/RTU Data Frame

Modbus RTU is a master and slave communication system. The CRC checking code includes from drive's address bits to data bits. This standard data framing are as follows:

Address	Function Code	Data	CRC

Based on data transfer status, there can be two types of response code:

Normal Modbus response:

Response function code = request function code

Modbus error response:

Response function code = request function code + 0x80

Providing an error code to indicate the error reasoning.

13.6 Modbus Register Table

Modbus Register Table				
Register	**Access**	**Data Type**	**SCL Command**	**Map Register**
40001	Read	SHORT	Alarm Code (AL)	f
40002	Read	SHORT	Status Code (SC)	s
40003	Read	SHORT	Immediate Expanded Inputs (IS)	y
40004	Read	SHORT	Driver Board Inputs (ISX)	i
40005..6	Read	LONG	Encoder Position (IE, EP)	e
40007..8	Read	LONG	Immediate Absolute Position	l
40009..10	Write	LONG	Absolute Position Command	P
40011	Read	SHORT	Immediate Actual Velocity (IV0)	v
40012	Read	SHORT	Immediate Target Velocity (IV1)	w
40013	Read	SHORT	Immediate Drive Temperature (IT)	t
40014	Read	SHORT	Immediate Bus Voltage (IU)	u
40015..16	Read	LONG	Immediate Position Error (IX)	x
40017	Read	SHORT	Immediate Analog Input Value (IA)	a
40018	Read	SHORT	Q Program Line Number	b
40019	Read	SHORT	Immediate Current Command (IC)	c
40020..21	Read	LONG	Relative Distance (ID)	d
40022..23	Read	LONG	Sensor Position	g
40024	Read	SHORT	Condition Code	h
40025	Read	SHORT	Analog Input 1 (IA1)	j

40026	Read	SHORT	Analog Input 2 (IA2)	k
40027	Read	SHORT	Command Mode (CM)	m
40028	R/W	SHORT	Point-to-Point Acceleration (AC)	A
40029	R/W	SHORT	Point-to-Point Deceleration (DE)	B
40030	R/W	SHORT	Velocity (VE)	V
40031..32	R/W	LONG	Point-to-Point Distance (DI)	D
40033..34	R/W	LONG	Change Distance (DC)	C
40035	R/W	SHORT	Change Velocity (VC)	U
40036	Read	SHORT	Velocity Move State	n
40037	Read	SHORT	Point-to-Point Move State	o
40038	Read	SHORT	Q Program Segment Number	p
40039	Read	SHORT	Average Clamp Power (regen)	r
40040	Read	SHORT	Phase Error	z
40041..42	R/W	LONG	Position Offset	E
40043	R/W	SHORT	Miscellaneous Flags	F
40044	R/W	SHORT	Current Command (GC)	G
40045..46	R/W	LONG	Input Counter	I
40047	R/W	SHORT	Jog Accel (JA)	
40048	R/W	SHORT	Jog Decel (JL)	
40049	R/W	SHORT	Jog Velocity (JS)	J
40050	R/W	SHORT	Accel/Decel Current (CA)	
40051	R/W	SHORT	Running Current (CC)	N
40052	R/W	SHORT	Idle Current (CI)	
40053	R/W	SHORT	Steps per Revolution	R
40054	R/W	SHORT	Pulse Counter	S
40055	R/W	SHORT	Time Stamp	W
40056	R/W	SHORT	Analog Position Gain (AP)	X
40057	R/W	SHORT	Analog Threshold (AT)	Y
40058	R/W	SHORT	Analog Offset (AV	Z
40059..60	R/W	LONG	Accumulator	0
40061..62	R/W	LONG	User Defined	1
40063..64	R/W	LONG	User Defined	2

40065..66	R/W	LONG	User Defined	3
40067..68	R/W	LONG	User Defined	4
40069..70	R/W	LONG	User Defined	5
40071..72	R/W	LONG	User Defined	6
40073..74	R/W	LONG	User Defined	7
40075..76	R/W	LONG	User Defined	8
40077..78	R/W	LONG	User Defined	9
40079..80	R/W	LONG	User Defined	:
40081..82	R/W	LONG	User Defined	;
40083..84	R/W	LONG	User Defined	<
40085..86	R/W	LONG	User Defined	=
40087..88	R/W	LONG	User Defined	>
40089..90	R/W	LONG	User Defined	?
40091..92	R/W	LONG	User Defined	@
40093..94	R/W	LONG	User Defined	[
40095..96	R/W	LONG	User Defined	\
40097..98	R/W	LONG	User Defined	]
40099..100	R/W	LONG	User Defined	^
40101..102	R/W	LONG	User Defined	_
400103..104	R/W	LONG	User Defined	`
40105	R/W	SHORT	Brake Release Delay	
40106	R/W	SHORT	Brake Engage Delay	
40107	R/W	SHORT	Idle Current Delay	
40108	R/W	SHORT	Hyperbolic Smoothing Gain	
40109	R/W	SHORT	Hyperbolic Smoothing Phase	
40110	R/W	SHORT	Analog Filter Gain	
40111..124			(reserved)	
40125	R/W	SHORT	Command Opcode	
40126	R/W	SHORT	Parameter 1	
40127	R/W	SHORT	Parameter 2	
40128	R/W	SHORT	Parameter 3	
40129	R/W	SHORT	Parameter 4	

40130	R/W	SHORT	Parameter 5	

13.7 Command Opcode description

Register 40125 is defined as command Opcode, when following command is entered into register, the drive will execute the corresponding operation.

1) SCL Command Encoding Table

SCL Command Encoding Table

Function	SCL	Opcode	Parameter1	Parameter2	Parameter3	Parameter4	Parameter5
Alarm Reset	AX	0xBA	×	×	×	×	×
Start Jogging	CJ	0x96	×	×	×	×	×
Stop Jogging	SJ	0xD8	×	×	×	×	×
Encoder Function	EF	0xD6	0,1,2 or 6	×	×	×	×
Encoder Position	EP	0x98	Position	×	×	×	×
Feed to Double Sensor	FD	0x69	I/O Point 1	Condition 1	I/O Point 2	Condition 2	×
Follow Encoder	FE	0xCC	I/O Point	Condition	×	×	×
Feed to Length	FL	0x66	×	×	×	×	×
Feed to Sensor with Mask Distance	FM	0x6A	I/O Point	Condition	×	×	×
Feed and Set Output	FO	0x68	I/O Point	Condition	×	×	×
Feed to Position	FP	0x67	×	×	×	×	×
Feed to Sensor	FS	0x6B	I/O Point	Condition	×	×	×
Feed to Sensor with Safety Distance	FY	0x6C	I/O Point	Condition	×	×	×
Jog Disable	JD	0xA3	×	×	×	×	×
Jog Enable	JE	0xA2	×	×	×	×	×
Motor Disable	MD	0x9E	×	×	×	×	×
Motor Enable	ME	0x9F	×	×	×	×	×
Seek Home	SH	0x6E	I/O Point	Condition	×	×	×
Set Position	SP	0xA5	Position	×	×	×	×
Filter Input	FI	0xC0	I/O Point	Filter Time	×	×	×
Filter Select Inputs	FX	0xD3	×	×	×	×	×
Step Filter Freq	SF	0x06	Freq	×	×	×	×
Analog Deadband	AD	0xD2	0.001 V	×	×	×	×
Alarm Reset Input	AI	0x46	Function ('1'.. .'3')	I/O Point	×	×	×
Alarm Output	AO	0x47	Function ('1'..'3')	I/O Point	×	×	×

Analog Scaling	AS	0xD1	×	×	×	×	×
Define Limits	DL	0x42	1..3	×	×	×	×
Set Output	SO	0x8B	I/O Point	Condition	×	×	×
Wait for Input	WI	0x70	×	×	×	×	×
Queue Load & Execute	QX	0x78	1..12	×	×	×	×
Wait Time	WT	0x6F	0.01 sec	×	×	×	×
Stop Move, Kill Buffer	SK	0xE1	×	×	×	×	×
Stop Move, Kill Buffer, Normal Decel	SKD	0xE2	×	×	×	×	×

For more detailed command functions description, please refer to **Host Command Reference manual.**

2) Digital I/O Function Selection and I/O Status

Character	hex code	
'0'	0x30	Index of encode
'1'	0x31	input 1 or output 1
'2'	0x32	input 2 or output 2
'3'	0x33	input 3 or output 3
'4'	0x34	input 4 or output 4
'L'	0x4C	low state (closed)
'H'	0x48	high state (open)
'R'	0x52	rising edge
'F'	0x46	falling edge

13.8 Modbus/RTU Applications

13.8.1 Position Control

1. Target Profile Planning

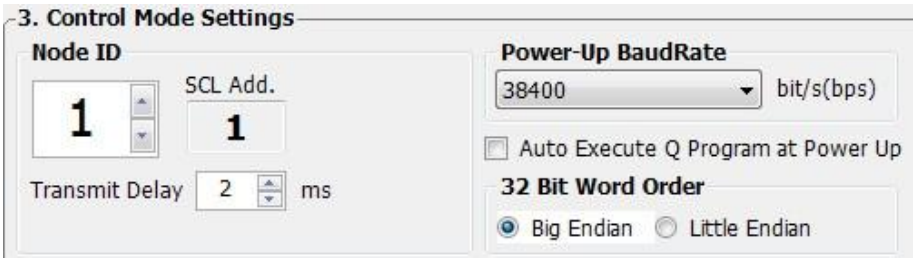
SCL command	Target Value	Unit	Register Address	Dec (in Hex)	Description
AC Acceleration	100	rps/s	40028	600(258h)	The unit for register 40028 is 1/6 rps2, when target acceleration is 100rps/s, the value will be 600
DE Deceleration	200	rps/s	40029	1200(4B0h)	The unit for register 40029 is 1/6 rps2, when target acceleration is 200rps/s, the value will be 1200
VE Velocity	10	rps	40030	2400(960h)	The unit for register 40030 is 1/240 rps. When target velocity is 10rps, the value will be 2400.
DI Distance	20000	counts	40031~40032	20000(4E20h)	The target distance will be 20000 counts

2. Drive Setting

Parameter	Function
P-75 (PR) = 5	Big-endian data transfer

P-77 (BR) = 3	communication baud rate 38400bps
P-78 (DA) = 1	Communication address 1
P-14 (PM) = 8	Power up mode as Modbus/RTU

Use M servo suite software for configurations:



3. Sending Command

First Step:

Set acceleration register 40028 = 258h, deceleration register 40029 = 4B0h, velocity register 40030 = 960h, and target position 40031~40032 = 4E20h.

Host Sending: 01 10 00 1B 00 05 0A 02 58 04 B0 09 60 00 00 4E 20 24 3B

Rive Respond: 01 10 00 1B 00 05 70 0D

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	10H	1		Function Code	10H	1
Starting Data Address (Register 40028)	00H(High) 1BH(Low)	2		Starting Data Address (Register 40028)	00H(High) 1BH(Low)	2
Number of Data （In word）	00H(High) 05H(Low)	2		Number of Data （In word）	00H(High) 05H(Low)	2
Number of Data （In byte）	0AH	1		CRC Check Low	70	1
Content of first Data address 40028	02(High) 58(Low)	2		CRC Check High	0D	1
Content of second Data address 40029	04H(High) B0H(Low)	2				
Content of third Data address 40030	09H(High) 60H(Low)	2				
Content of fourth Data address 40031	00H(High) 00H(Low)	2				
Content of fifth Data address 40032	4EH(High) 20H(Low)	2				
CRC Check Low	24	1				
CRC Check High	3B	1				

Second Step: Point to Point Motion Command

Command Opcode describes register 40125's control code. From the SCL code list shows that for point to point position motion, it requires to write data 0x66 to register 40125.

SCL Command Encoding Table							
Function	SCL	Opcode	Parameter 1	Parameter	Parameter	Parameter	Parameter
Feed to Length	FL	0x66	×	×	×	×	×

Host Sending: 01 06 00 7C 00 66 C8 38

Drive Reply: 01 06 00 7C 00 66 C8 38

Listed As Below:

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	06H	1		Function Code	06H	1
Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2		Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2
Content of Data	00(High) 66(Low)	2		Content of Data	00(High) 66(Low)	2
CRC Check Low	C8	1		CRC Check Low	C8	1
CRC Check High	38	1		CRC Check High	38	1

13.8.2 Velocity Mode

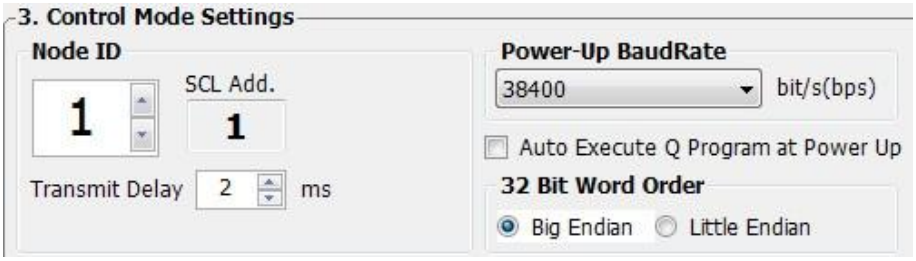
1. Velocity Mode Parameters

SCL Command	Target Value	Unit	Register Address	Write Value Dec(Hex)	Description
JA Jog Acceleration	100	rps/s	40047	600(258h)	The unit for register 40028 is $\frac{1}{6}$rps^2, when target acceleration is 100rps/s, the value will be 600
JL Jog Deceleration	200	rps/s	40048	1200(4B0h)	The unit for register 40029 is $\frac{1}{6}$rps^2, when target deceleration is 200rps/s, the value will be 1200
JS Jog Speed	10	rps	40049	2400(960)	The unit for register 40049 is $\frac{1}{240}$rps, when target velocity is 10rps, the value will be 2400

2. Drive Setting

Parameter	Function
P-75 (PR) = 5	Big-endian data transfer
P-77 (BR) = 3	communication baud rate 38400bps
P-78 (DA) = 1	Communication address 1
P-14 (PM) = 8	Power up mode as Modbus/RTU

Use M servo suite software for configurations:



First Step:

Set velocity mode acceleration register as 40047 = 258h, deceleration register as 40048 = 4B0h, and velocity register 40049 = 960h.

Host Sending: 01 10 00 2E 00 03 06 02 58 04 B0 09 60 A0 9F

Drive Reply: 01 10 00 2E 00 03 E0 01

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	10H	1		Function Code	10H	1
Starting Data Address (Register 40047)	00H(High) 2EH(Low)	2		Starting Data Address (Register 40047)	00H(High) 2EH(Low)	2
Number of Data （In word）	00H(High) 03H(Low)	2		Number of Data （In word）	00H(High) 03H(Low)	2
Number of Data （In byte）	06H	1		CRC Check Low	E0	1
Content of first Data address 40047	02(High) 58(Low)	2		CRC Check High	01	1
Content of second Data address 40048	04H(High) B0H(Low)	2				
Content of third Data address 40049	09H(High) 60H(Low)	2				
CRC Check Low	A0	1				
CRC Check High	9F	1				

Second Step: Command for Executing Point to Point Motion

Command Opcode describes register 40125's control code. From the SCL code list shows that for JOG mode, it requires to write data 0x96 to register 40125 to start, and sending 0xD8 to register 40125 to stop.

SCL Command Encoding Table							
Function	SCL	Opcode	Parameter 1	Parameter	Parameter	Parameter	Parameter
Start Jogging	CJ	0x96	×	×	×	×	×
Stop Jogging	SJ	0xD8	×	×	×	×	×

Start

Host Sending: 01 06 00 7C 00 96 C8 7C

Drive Reply: 01 06 00 7C 00 96 C8 7C

Stop

Host Sending: 01 06 00 7C 00 D8 48 48

Drive Reply: 01 06 00 7C 00 D8 48 48

Stat Message

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	06H	1		Function Code	06H	1
Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2		Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2
Content of Data	00(High) 96(Low)	2		Content of Data	00(High) 96(Low)	2
CRC Check Low	C8	1		CRC Check Low	C8	1
CRC Check High	7C	1		CRC Check High	7C	1

Stop Message

Command Message（Master）				Response Message（slave）		
Function	Data	Number of Bytes		Function	Data	Number of Bytes
Slave Address	01H	1		Slave Address	01H	1
Function Code	06H	1		Function Code	06H	1
Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2		Starting Data Address (Register 40125)	00H(High) 7CH(Low)	2
Content of Data	00(High) D8(Low)	2		Content of Data	00(High) D8(Low)	2
CRC Check Low	48	1		CRC Check Low	48	1
CRC Check High	48	1		CRC Check High	48	1

14 Position Table Mode

Position table mode allows Point-to-Point linear motion and Rotary motion without any external pulse input. Instead, position table mode uses Input port X7~X12 to configure different positions command. Input X4 is the trigger for motion.

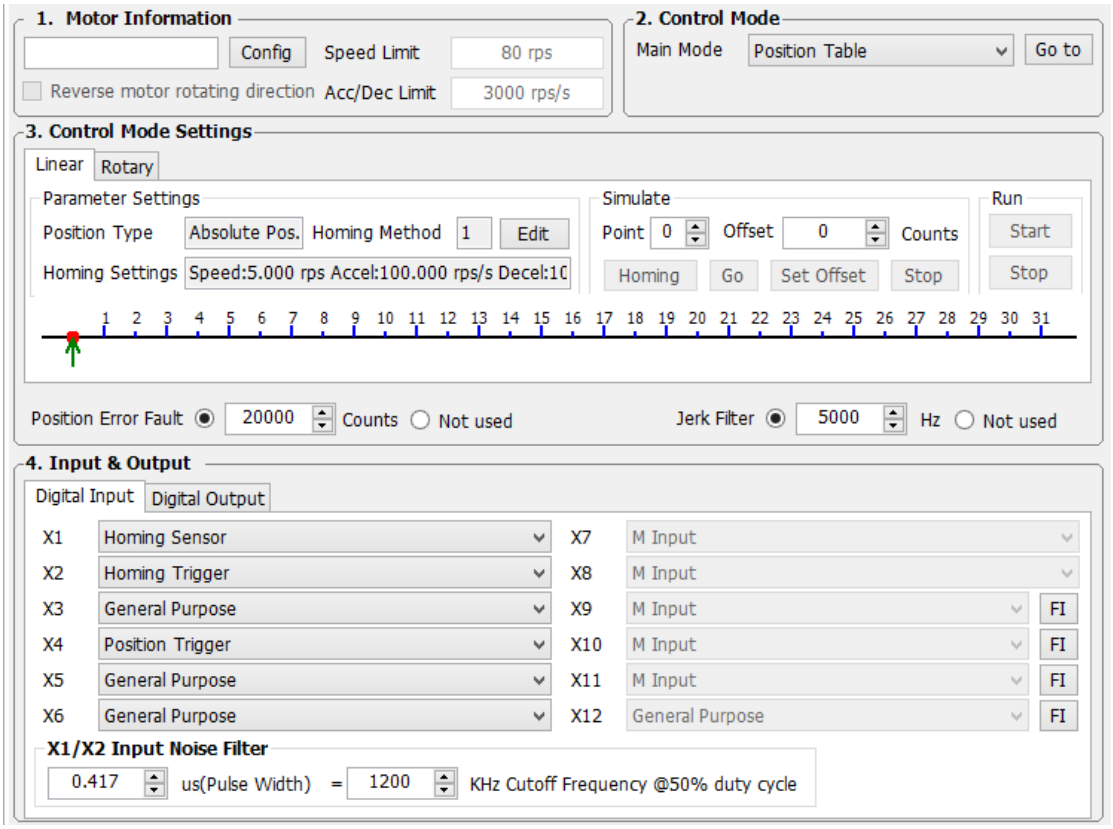


Figure 14.1 Position table mode

NOTE: Only -S type M2 series servo drive supports position table mode

14.1 Linear motion

Linear motion for position table mode can set up to 63 positions (not include homing position). Detailed software setting as follows:

14.1.1 Linear Motion Software Configuration

1) Open M Servo Suite, connect the driver with software(refer to software manual for details)

2) Select “position table” control mode from”step1: configuration”------“2. Control mode”
As shown in Figure 14.1 Select Position Table

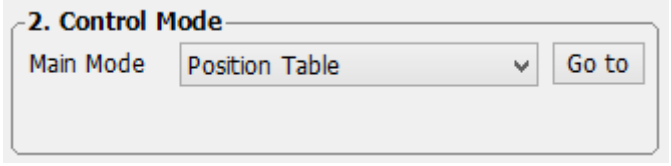


Figure 14.1 Select Position Table

3) Select linear motion from “3. Control mode setting” as show in Figure 14.2 Linear motion setting.

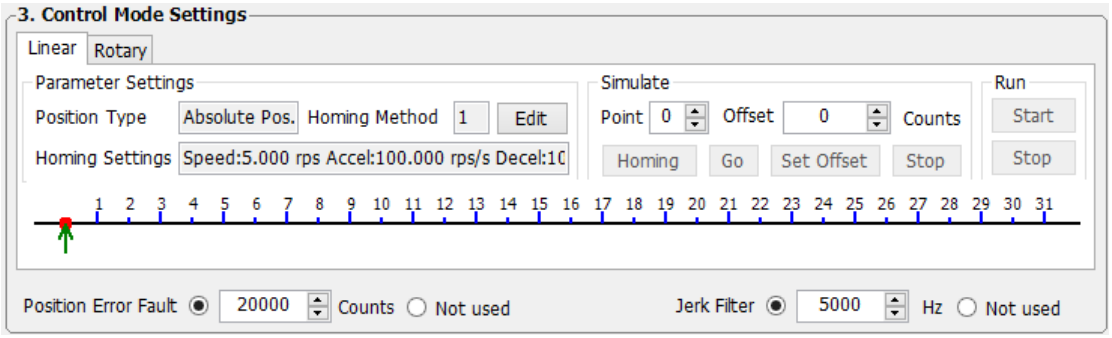


Figure 14.2 Linear motion setting

4) Click edit for detailed motion configurations, as shown in Figure 14.3 Linear motion configuration.

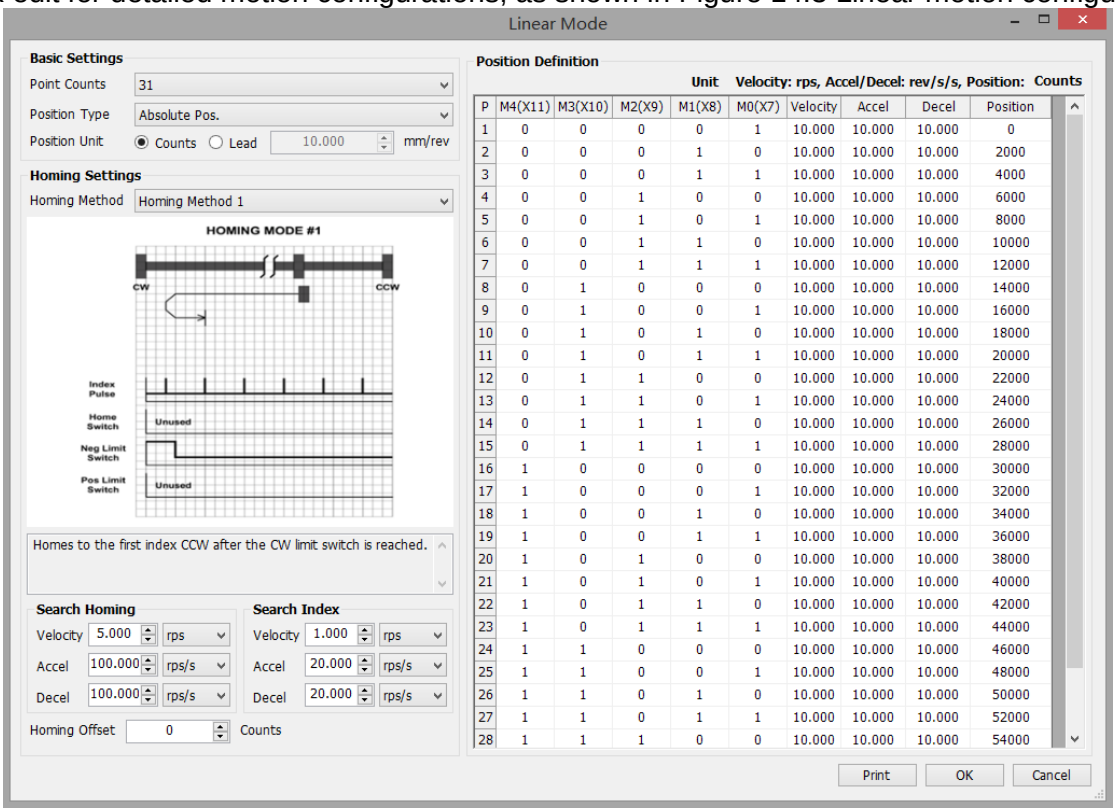


Figure 14.3 Linear motion configuration

14.1.2 Basic Configuration

Point Counts: Select the number of position points, For M series servo drives, there are four selections: 7、15、31、63 number of position points.

Position type: There are two types for point-to-point motion: Relative position; and absolute position. Example shown in Figure 14.4 Relative position VS Absolute position:

Set P1 position for 5revs，P2 position for 10revs，the difference between Relative position and absolute position are as shown in below:

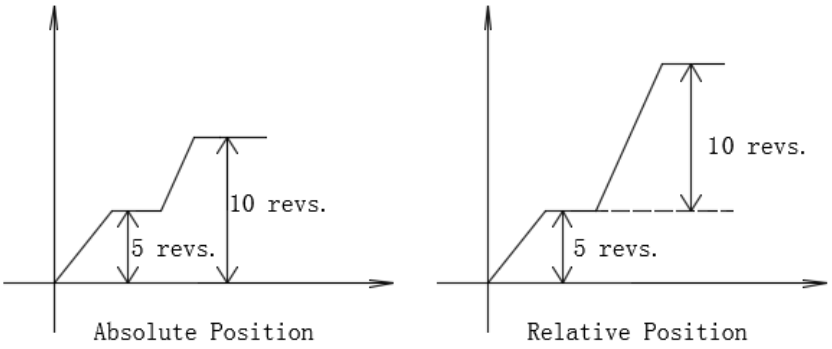


Figure 14.4 Relative position VS Absolute position

Position Unit: Set position point Units.

Counts: It represents the number pulse from encoder output. For position table mode, one motor revolution is 10000 pulse counts.

Lead: It represents the distance for one motor revolution. Unit: mm/rev.

14.1.3 Homing settings:

Homing Method: There are 12 types to homing available.

Search homing: This feature sets the velocity, acceleration and deceleration for search homing switch.

Search Index: This feature sets the velocity, acceleration and deceleration for search motor encoder index signal after the homing switch is reached.

Homing Offset: After homing process is finished, this sets the offset value from the homing position.

14.1.4 Print

Click on “Print” to print out the configurations table, as shown in Figure 14.5 Print Position Table configuration below:

Position Table Configuration. ©Shanghai AMP & MOONS' Automation Co., Ltd.
Linear Mode
Point Counts: 31
Position Type: Absolute Pos.
Homing Method: Homing Method 1
Search Homing
Velocity: 5 rev/s Accel: 100 rev/s/s Decel: 100 rev/s/s
Search Index
Velocity: 5 rev/s Accel: 100 rev/s/s Decel: 100 rev/s/s
Homing Offset: 0

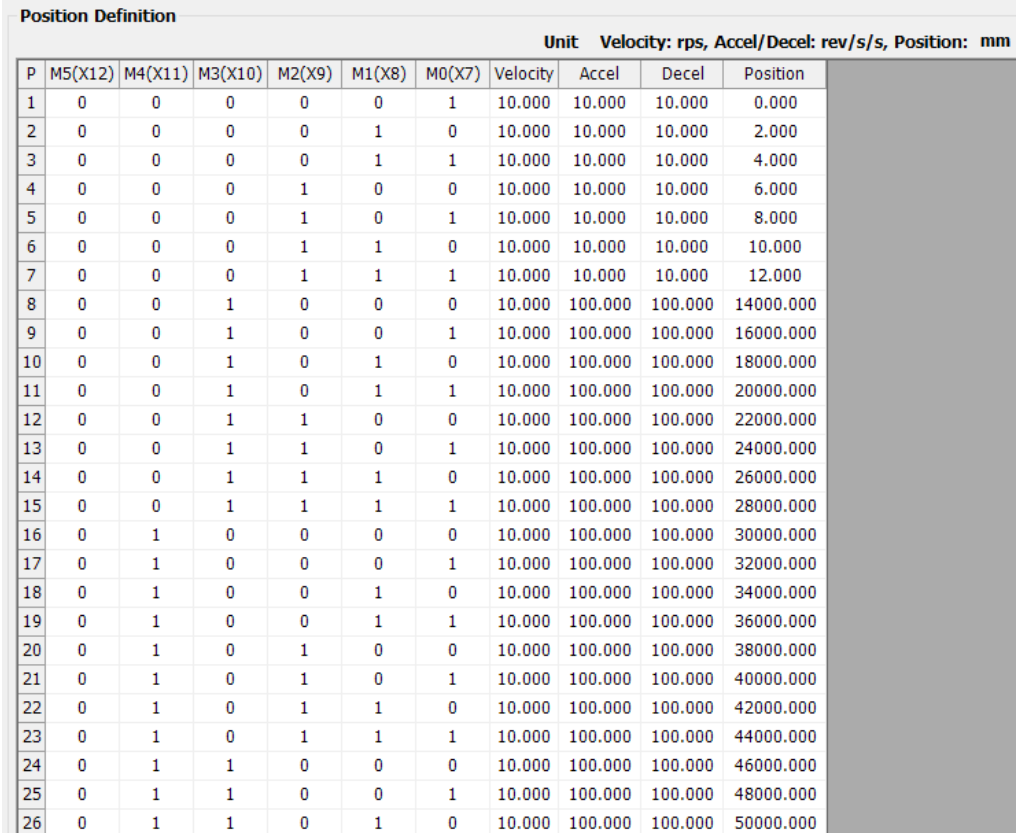
Position Definition

				Unit	Velocity: rps, Accel/Decel: rev/s/s, Position:			
P	M4(X11)	M3(X10)	M2(X9)	M1(X8)	M0(X7)	Velocity	Accel	Decel	Position
1	0	0	0	0	1	10.000	10.000	10.000	0
2	0	0	0	1	0	10.000	10.000	10.000	2000
3	0	0	0	1	1	10.000	10.000	10.000	4000
4	0	0	1	0	0	10.000	10.000	10.000	6000
5	0	0	1	0	1	10.000	10.000	10.000	8000
6	0	0	1	1	0	10.000	10.000	10.000	10000
7	0	0	1	1	1	10.000	10.000	10.000	12000
8	0	1	0	0	0	10.000	10.000	10.000	14000
9	0	1	0	0	1	10.000	10.000	10.000	16000
10	0	1	0	1	0	10.000	10.000	10.000	18000
11	0	1	0	1	1	10.000	10.000	10.000	20000
12	0	1	1	0	0	10.000	10.000	10.000	22000
13	0	1	1	0	1	10.000	10.000	10.000	24000
14	0	1	1	1	0	10.000	10.000	10.000	26000

Figure 14.5 Print Position Table configuration

14.1.5 Position Definition

Position Definition shows the detailed configurations for each position point, including velocity, acceleration and deceleration, position. In this table, it also shows the input condition (X7~X12) to trigger each position.



Position Definition

Unit Velocity: rps, Accel/Decel: rev/s/s, Position: mm

P	M5(X12)	M4(X11)	M3(X10)	M2(X9)	M1(X8)	M0(X7)	Velocity	Accel	Decel	Position
1	0	0	0	0	0	1	10.000	10.000	10.000	0.000
2	0	0	0	0	1	0	10.000	10.000	10.000	2.000
3	0	0	0	0	1	1	10.000	10.000	10.000	4.000
4	0	0	0	1	0	0	10.000	10.000	10.000	6.000
5	0	0	0	1	0	1	10.000	10.000	10.000	8.000
6	0	0	0	1	1	0	10.000	10.000	10.000	10.000
7	0	0	0	1	1	1	10.000	10.000	10.000	12.000
8	0	0	1	0	0	0	10.000	100.000	100.000	14000.000
9	0	0	1	0	0	1	10.000	100.000	100.000	16000.000
10	0	0	1	0	1	0	10.000	100.000	100.000	18000.000
11	0	0	1	0	1	1	10.000	100.000	100.000	20000.000
12	0	0	1	1	0	0	10.000	100.000	100.000	22000.000
13	0	0	1	1	0	1	10.000	100.000	100.000	24000.000
14	0	0	1	1	1	0	10.000	100.000	100.000	26000.000
15	0	0	1	1	1	1	10.000	100.000	100.000	28000.000
16	0	1	0	0	0	0	10.000	100.000	100.000	30000.000
17	0	1	0	0	0	1	10.000	100.000	100.000	32000.000
18	0	1	0	0	1	0	10.000	100.000	100.000	34000.000
19	0	1	0	0	1	1	10.000	100.000	100.000	36000.000
20	0	1	0	1	0	0	10.000	100.000	100.000	38000.000
21	0	1	0	1	0	1	10.000	100.000	100.000	40000.000
22	0	1	0	1	1	0	10.000	100.000	100.000	42000.000
23	0	1	0	1	1	1	10.000	100.000	100.000	44000.000
24	0	1	1	0	0	0	10.000	100.000	100.000	46000.000
25	0	1	1	0	0	1	10.000	100.000	100.000	48000.000
26	0	1	1	0	1	0	10.000	100.000	100.000	50000.000

Figure 14.6 Position definition table

M0(X7) ~ M5(X12) status：'0' means input is closed；'1' means input is Open。

After the homing process，motor will move to corresponding position which selected by input M0(X7) ~ M5(X12), and triggered by X4 (position trigger) when it changes from 'open' to 'close'.

5) Click 'OK' to finish linear mode settings
6) Click 'Download to Drive' the set the drive
7) Close the software turn off the power, and restart both drive and software for running position table mode.

14.1.6 Simulate

After the configuration process, simulate function can verify the settings simulate the motions.

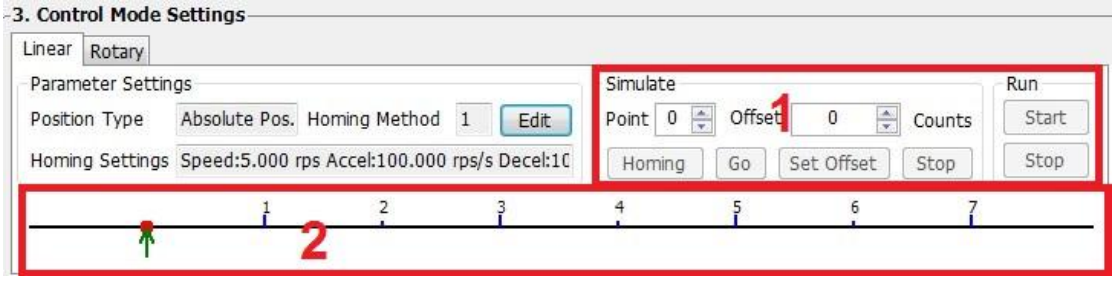


Figure 14.7 Linear motion Simulate

Homing: Click ‘homing’ to start homing process.

Go: Set the position point by changing the value in point box, and click ‘go’ button to start the motion. In Figure 14.7 Linear motion Simulate, green arrow in box ② shows the load position in real time.

Set Offset: Confirm offset position, change this value will change the position in position table

Stop: Stop current motion immediately

14.1.7 Linear motion input definition

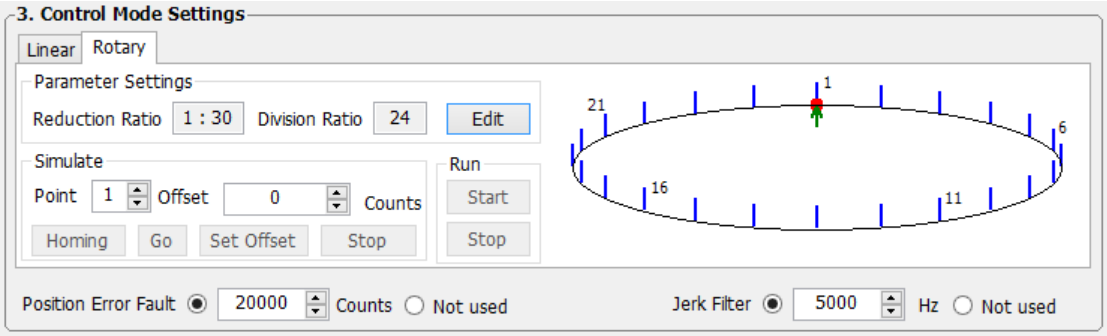
Input	Function	Description
X1	Homing Sensor	Homing sensor switch
X2	Homing Trigger	Triggering homing process
X3	General Purpose	General purpose
	Servo On When Closed	Enable the motor drive when input closed
	Servo On When Open	Enable the motor drive when input open
X4	Position Trigger	It is a trigger signal. When Input X4 changes from open to close, motor will move to the position selected by switch M0(X7) ~ M5(X12)
X5	General Purpose	General purpose
	CW Limit Sensor	Set CW position limit, please refer to M2 user manual chapter 7.1.3, CW/CCW limit for more details
X6	General Purpose	General purpose
	CCW Limit Sensor	Set CCW position limit, please refer to M2 user manual chapter 7.1.3, CW/CCW limit for more details
X7~X12	M Input	Position point input

14.2 Rotary motion

Rotary motion is highly suitable for dividing plate applications, system gearing reduction ratio can also be set based on the application. Settings such as number of division per revolution, motion profiles and homing profiles can also be set.

After the configuration. Input X4 is the motion trigger, the load will rotate according to set direction. Each trigger signal will turn the load by one single rotary point based on the settings.

14.2.1 Rotary motion software configuration



Edit: Click on 'Edit" to enter detailed configuration page, as shown in Figure 14.1 Detail configuration for rotary motion below

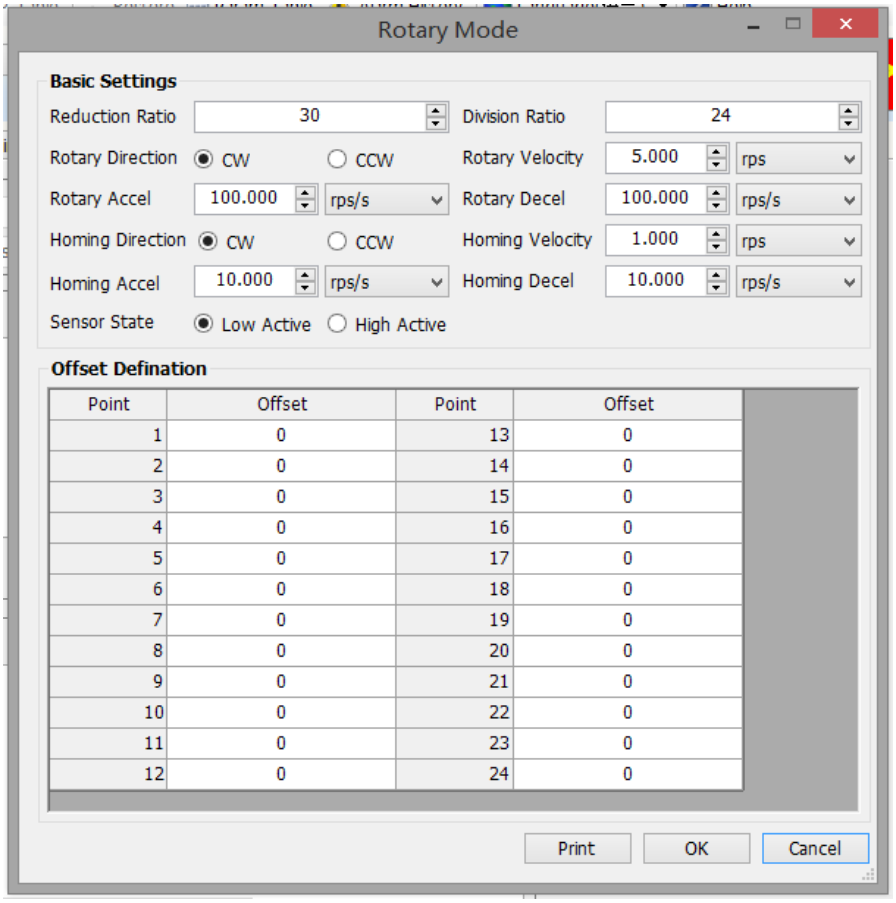


Figure 14.1 Detail configuration for rotary motion

Reduction ratio: Set mechanical gear box ratio

Division Ratio: Divide one revolution into numbers of point with equal distance

Rotary direction: Select the direction for rotary motion

Rotary velocity, rotary acceleration, rotary deceleration: Set motor rotary velocity, rotary acceleration, and rotary deceleration values

NOTE: the rotary are set based on Motor velocity/acceleration/deceleration. For actual system speed, please refer to ratio calculation shown below:

System speed = Motor Speed × Reduction ratio

Homing direction: Set homing direction

Homing velocity, Homing acceleration, Homing deceleration: To set motor homing velocity, homing acceleration, and homing deceleration values

NOTE: the rotary are set based on Motor velocity/acceleration/deceleration. For actual system speed, please refer to ratio calculation shown below:

System speed = Motor Speed × Reduction ratio

Sensor State: Set homing sensor type: low active, high active

Offset definition: Set position offset for each position point, for minor tunings.

14.2.2 Rotary motion input definition

Input	Function	Description
X1	Homing Sensor	Homing sensor switch
X2	Homing Trigger	Triggering homing process
X3	General Purpose	General purpose
	Servo On When Closed	Enable the motor drive when input closed
	Servo On When Open	Enable the motor drive when input open
X4	Position Trigger	It is a trigger signal. When Input X4 change from open to close, the load will move one single rotary point according to the position configuration