

A1 Series Servo Cylinder User Manual



Safety Information

IMPORTANT: Read this manual before installing and operating the Ultra Motion Servo Cylinder. Failure to read this section can result in personal harm or damage to the product.

Safety Disclaimer

The Servo Cylinder is intended to be a subcomponent of a larger piece of machinery or automated system. This section is not intended to provide the safety guidelines for the entire machine or system that the Servo Cylinder is installed into. It is the responsibility of the purchaser or system designer to assess the risks and safety requirements of the end application they are designing.

If the Servo Cylinder is to be used in a safety critical application, the purchaser or system designer must perform appropriate safety testing to ensure that the product meets the requirements and safety criteria for their application.

Ultra Motion has made all reasonable efforts to present accurate information in this document and is not responsible for unintentional oversights. If, at any time, the purchaser has questions or uncertainty about information in this manual, contact Ultra Motion to speak with an engineer.

Safety Warnings

- Once powered, the Servo Cylinder is capable of rapid motion and can produce large amounts of force. Always ensure that safe clearances from body parts and other equipment are maintained before applying power.
- The Servo Cylinder operates on low voltage (8 to 36 VDC recommended). Use caution when handling and working around the actuator to avoid electrical shock.
- Provide adequate cooling for the product.

Safety Notifications



As you read through the manual, you will notice certain safety notifications that indicate other important safety related information.

1.0 Introduction

What is the Servo Cylinder?

The Ultra Motion Servo Cylinder is a high performance linear servo system comprised of a robust rod-style actuator, configurable BLDC motion controller, and cutting edge multi-turn absolute feedback sensor (Phase Index). All components of the system have been engineered to seamlessly work together and provide a simple and extremely reliable linear positioning device.

In short, users provide power and a command input (either position or force) and the Servo Cylinder handles the rest. The built in actuator-specific controller can be set to work with many common command inputs and includes user-configurable firmware for defining many attributes including performance characteristics, software limits, output information, and much more.

Servo Cylinder advantages include:

- **Closed Loop position, force, and speed control directly out of the box**
- **Complete command language optimized for linear motion applications**
- **Absolute position all of the time – never requires homing**
- **Extreme fault tolerance**
- **Precision in rugged package – extremely long life**
- **Completely configurable**

Product Features

Advanced DSP Based Motion Control

- The onboard digital signal processor handles all user inputs, configuration, trajectory generation, Field-Oriented Control, and position control loop calculations.
- Phase Index is seamlessly integrated to provide reliable absolute position information of both the BLDC rotor and the absolute linear position of the actuator shaft.
- Field-Oriented Control of the BLDC motor increases efficiency, provides ideal motion at both low and high speeds, provides increased acceleration performance, and allows for direct user control of actuator force.

High-Performance Inverter Stage

- Modern automotive grade components allow for robust operation over a wide range of supply voltages (8 to 36 VDC).
- Low on-state resistance, high current capacity, and fast switching time means cool and efficient control over the load.
- High-speed current feedback from all three phases allows for maximum flexibility of Field-Oriented Control.

High-Speed Optically Isolated Digital Inputs

- Two high speed, optically isolated digital inputs for maximum noise rejection and signal integrity. The opto-isolators are used in 1-2 ms pulse, CW/CCW, A/B Quadrature, Step & Direction, four preset positions, and toggle operating modes.

Easy USB Configuration

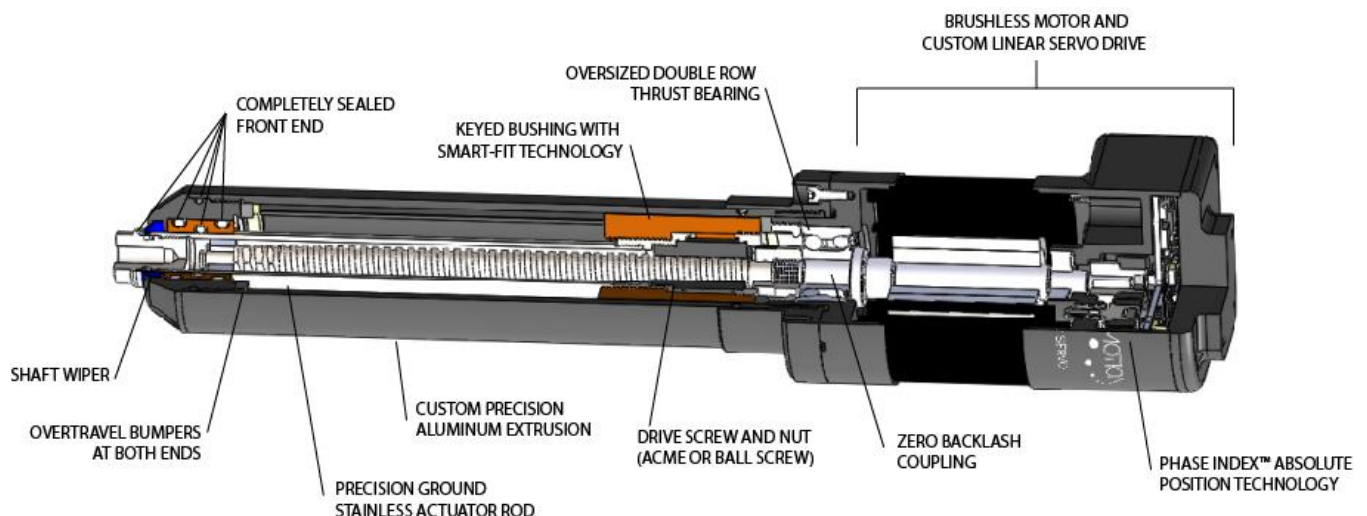
- Configuration of the Servo Cylinder is completed by editing a small configuration text file and saving any changes directly to the actuator.
- A small USB Mass Storage Device is maintained in the on-board, non-volatile memory of the microcontroller to provide fast, reliable access to all configuration parameters.

Flexible General Purpose Input/Outputs

The Servo Cylinder has two GPIO pins that can be configured as:

- Ground (sink up to 2 Amps)
- +5VDC (source up to 250 mA)
- General purpose limit switch outputs
- 1 kHz PWM waveform proportional to:
 - Absolute position
 - Velocity
 - Force
 - Bus voltage
 - Etc.
- Masked status word:
 - The GPIO pin will output a digital signal depending on the result of the Servo Cylinder's status word masked with a user defined bit mask.
 - This functionality provides a simple way to gather relevant diagnostic information from the Servo Cylinder.
 - See "Status Word Description" for more information

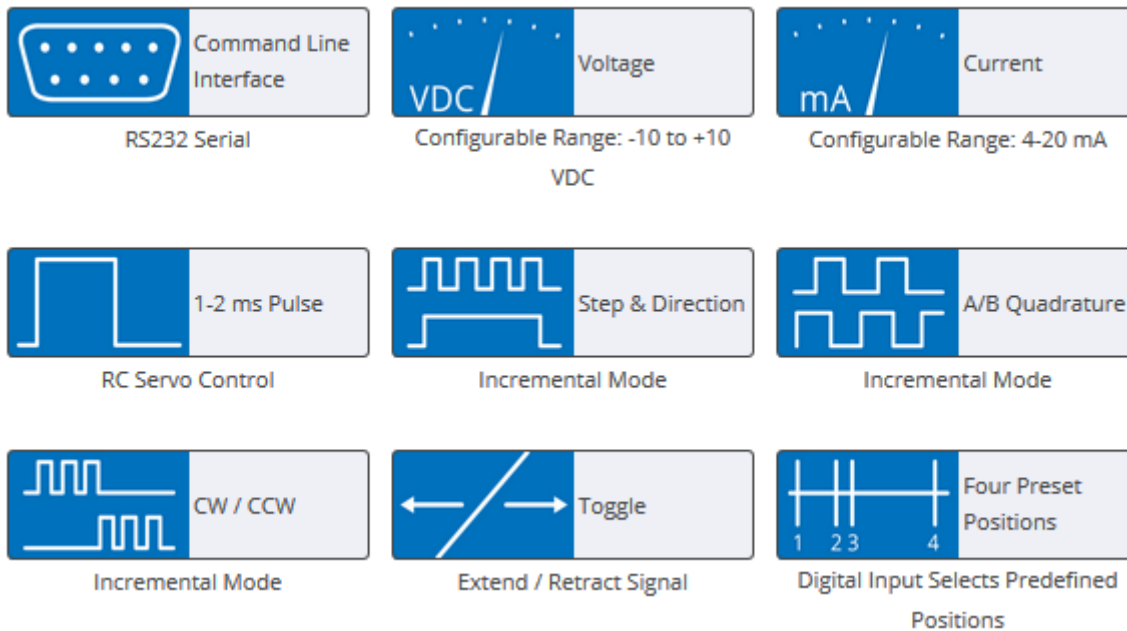
Robust Actuator Mechanics



Operating Modes

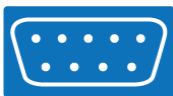
There are 5 distinct operating modes available for the Servo Cylinder. Within these 5 operating modes, there are a total of 9 command types. The icons below represent all the available command types.

IMPORTANT: The RS-232 port and any non-motion related serial commands from the command language can be used in ALL operating modes. Motion related serial commands from the serial command language are only available when in the command line interface operating mode.



There is no need to home the actuator in any of the operating modes

Command Line Interface (RS-232 Serial)



The Servo Cylinder RS-232 serial command line interface (CLI) offers the user complete control over all performance parameters, motion commands, and diagnostic information.

The Servo Cylinder can be controlled through a terminal program such as PuTTY, a PLC, or any program that can operate a serial communication port such as LabVIEW, MATLAB, Python, Arduino, etc.

The Servo Cylinder can be configured to operate in “Human mode” where detailed information regarding commands and asynchronous error messages are output to the terminal window. The Servo Cylinder can also be set to operate in “Machine Mode” where asynchronous messages are turned off, and checksums are sent to ensure communication integrity.

Proportional Mode

Proportional mode offers an extremely simple way to integrate an actuator into an application. The user simply supplies a signal proportional to the desired position, and the Servo Cylinder immediately responds.

Command Types:



Voltage (-10 to +10 VDC Range)



Current (4-20 mA Range)



PWM (1-2 ms Pulse Range)

There is no homing required so the user does not have to worry about intermittent power interruptions causing downtime or failure of their machine due to loss of position. Software definable end-of-stroke limits eliminate the need for limit switches, and the GPIO can be configured to indicate end of stroke, position, velocity, and more, to act as a drop-in replacement to existing systems.

The user simply configures the stroke range, and the corresponding proportional input range, and the Servo Cylinder is ready to use. For example, the user can adjust the configuration file to correlate 0 to 5 VDC to full stroke, or -10VDC to 10VDC to 1" to 3.250"

Incremental Mode

Incremental input mode allows users to realize the benefits of the Servo Cylinder without upgrading their existing control systems or PLCs.

Command Types:



Step & Direction



CW/CCW



Quadrature Signals

Incremental modes allow the Servo Cylinder to behave as a drop-in replacement to stepper motor systems while providing higher efficiency, increased torque at high speeds, no lost steps or stalling, and smoother operation in an integrated package.

As with all operating modes, there is no homing required. The configurable GPIO can output a bit when the software limit has been met, a PWM signal proportional to the absolute position of the Servo Cylinder, or the absolute position can be read directly over serial.

Two high speed optically isolated digital inputs reject noise and provide robust communication between PLCs and the Servo Cylinder.

Toggle Mode



The Servo Cylinder can be configured to mimic the simplicity of a brushed DC control system with the added performance benefits of a high efficiency, long life BLDC motor.

The actuator immediately responds to a user supplied extend/retract signal sent to the optically isolated digital inputs.

The absolute position capability of the Servo Cylinder eliminates the need for external limit switches and also allows for user configurable acceleration and max velocity for smooth, controlled motion.

Speed or Torque can be controlled by using one of the Servo Cylinder's analog inputs, or preset to a limit in the configuration text file.

Four Preset Position Mode



Four Preset position mode provides the user with an easy way to perform a variety of positioning tasks. The user selects one of four pre-configured position by toggling the state of the two optically isolated digital inputs (00,01,10,11), causing the actuator to move to the selected position using the Servo Cylinder's built-in trajectory generator.

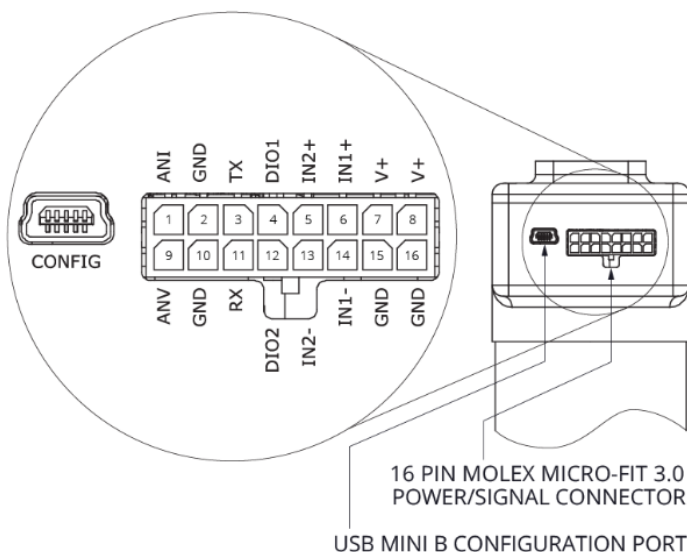
Homing is not required, which increases the ease of integration into machines and reduces complexity.

A user defined input filter is available to reject noise.

Electrical Specifications

Electrical Interface

| PIN NUMBER | LABEL | FUNCTION |
|------------|-------|--------------------------------|
| 1 | ANI | Analog Current Input |
| 2 | GND | Ground |
| 3 | TX | RS-232 Transmit |
| 4 | DIO1 | General Purpose Input/Output 1 |
| 5 | IN2+ | Optically Isolated Input 2+ |
| 6 | IN1+ | Optically Isolated Input 1+ |
| 7 | V+ | Power |
| 8 | V+ | Power |
| 9 | ANV | Analog Voltage Input |
| 10 | GND | Ground |
| 11 | RX | RS-232 Receive |
| 12 | DIO2 | General Purpose Input/Output 2 |
| 13 | IN2- | Optically Isolated Input 2- |
| 14 | IN1- | Optically Isolated Input 1- |
| 15 | GND | Ground |
| 16 | GND | Ground |



Power Requirements

Input Voltage Range: 8-36VDC Recommended (7-48VDC Absolute Max)



Power Shunt: Power shunts are power dissipation devices that engage at a pre-defined voltage level to prevent surplus energy generated by the actuator from further increasing the bus voltage and damaging the Servo Cylinder or other sensitive equipment. Power shunts are always recommended, but especially important in the following applications:

- Applications that require hard/fast deceleration.
- Applications where the load is assisting the actuator's motion (spring, gravity in a vertical application, etc.)

Note: Power shunts must be wired in parallel with the output of the DC power supply. Power shunts must maintain connectivity to the actuator even in the event of a blown power supply fuse or engaged emergency switch. See the **power wiring diagram**

Power Supply:: A 250 Watt, unregulated power supply is recommended. Smaller supplies are suitable for less demanding applications.

Unregulated Power Supplies (Preferred): An unregulated power supply is preferred for servo applications because of its ability to supply bursts of energy during acceleration, and its ability to absorb energy during deceleration/back-driving events. The large smoothing capacitor of a properly sized unregulated supply maintains safe actuator voltage levels when absorbing/supplying large amounts of electrical energy. The use of an unregulated power supply does not eliminate the need for a power shunt.

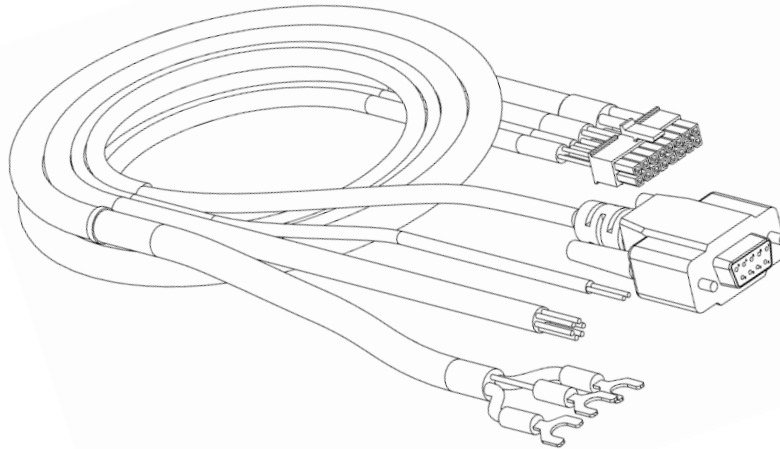
Regulated Power Supplies: Many popular benchtop power supplies are regulated, which are acceptable for applications that don't require high accelerations and cannot be back-driven. Regulated power supplies cannot absorb the energy created during deceleration or back-driving events as effectively as unregulated supplies, which can cause the supply voltage to quickly rise to unsafe levels. The power supply can also fall below required voltage levels due to high current demands during acceleration or high force events. To protect the Servo Cylinder from high voltages, a power shunt should be wired in parallel with the power supply.

Fuseable Link: An external, slow burning fuse should be used on the positive leg of the DC power supply to isolate the Servo Cylinder from the power supply in an overloading condition.

Emergency Stop Switch: The use of an emergency stop switch should always be considered by the system designer. The switch should be wired in series and provide a means to easily and rapidly cut power to the actuator during emergency situations. The system designer must fully understand the power-off behavior of the Servo Cylinder in their application before deciding to use an emergency off switch. For instance, cutting power in applications that are lifting a load vertically will result in backdriving of the Servo Cylinder.

Cable Requirements

You will need a Power/Signal Cable and a USB mini B cable to interface with the actuator. These cables can be purchased from Ultra Motion or you can provide your own.



Purchasing Cables from Ultra Motion:

USB CABLE: Use part number **CBL-USB** to purchase USB Mini B 6ft cable.

POWER/SIGNAL CABLE: Use the information below to specify a cable part number based on what functions you would like. A complete part number **CBL-S-AV-D** (with analog voltage) or **CBL-S-AC-D** (with analog current) will come with all wires populated.

All cables start with the base part number **CBL**. See base power cable section below for details.

Choose the additional functions as required:

-S (adds serial capability). See serial cable section below for details.

-AC or -AV (adds analog voltage or analog current capability). See analog cable section below for details.

-D (adds all digital capability including GPIO). See digital cable section below for details.

CABLE PART NUMBER EXAMPLE: A user planning to control the actuator with an analog voltage signal (-AV) who also wants access to the Servo Cylinder's GPIO (-D) would specify the following cable assembly part number: CBL-AV-D.

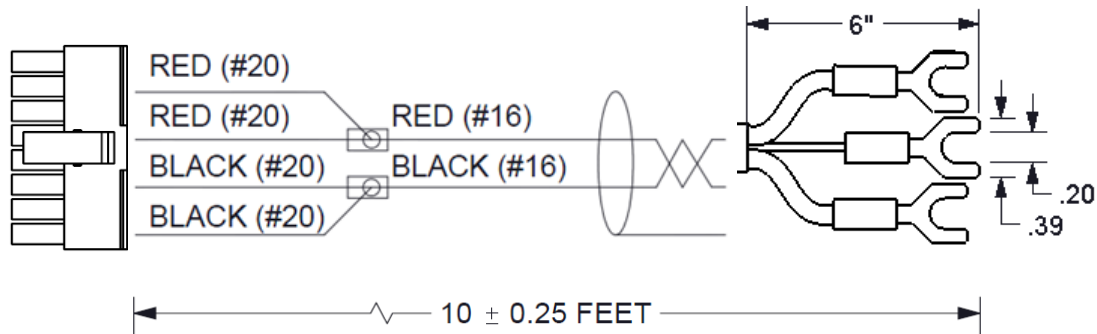
Please note that there is no need to double specify a cable if it is used in two functions. For example, if the user is interested in 1 to 2 ms pulse proportional mode (-D) and also GPIO functionality (-D), the full cable assembly part number is: CBL-D.

- **Base Power Cable (CBL)**

Cable Specification: (4) 20 AWG conductors are spliced to (2) 16 AWG conductors in a twisted, shielded cable terminated with 5 mm stud, crimp spade terminals.

Uses:

1. Supply power to the Servo Cylinder (Mandatory)

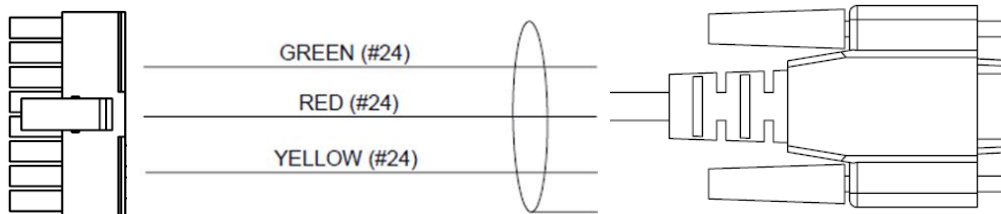


- **RS-232 Serial Cable (-S)**

Cable Specification: Shielded cable with (3) 24 AWG conductors terminated in an overmolded DB9 connector

Uses:

1. Command Line Interface Mode
2. RS-232 diagnostic information and data streaming
3. Updating firmware

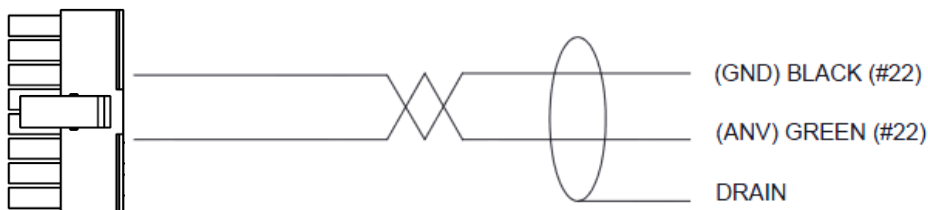


- **Analog Voltage Cable (-AV)**

Cable Specification: (2) 22 AWG conductors in a shielded twisted pair terminated with 6" flying leads.

Uses:

1. -10V to 10V Proportional Input Mode
2. Toggle Mode with -10V to +10V control of speed or force

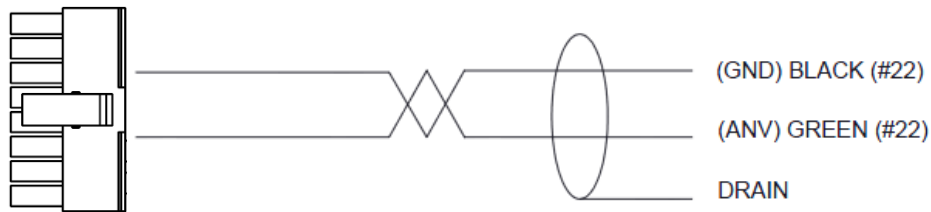


- **Analog Current Cable (-AC)**

Cable Specification: (2) 22 AWG conductors in a shielded twisted pair terminated with 6" flying leads.

Uses:

1. 4 to 20 mA Proportional Input Mode
2. Toggle Mode with 4 to 20 mA control of speed or force

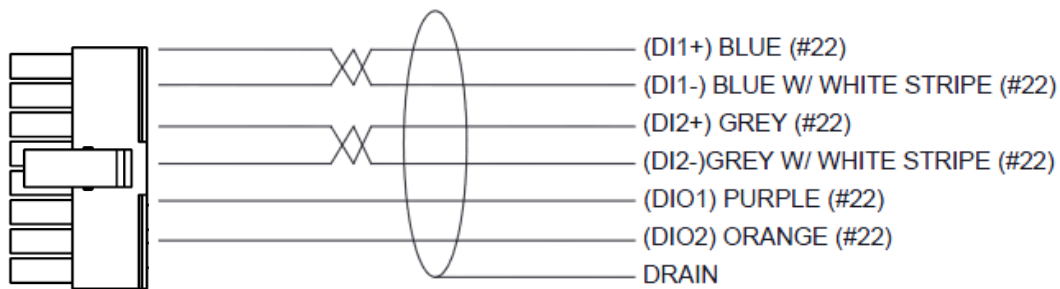


- **Digital Cable (-D)**

Cable Specification: (6) 22 AWG conductors in a shielded cable, terminated with 6" flying leads.

Uses:

1. 1 to 2 ms Pulse Proportional Input Mode
2. Incremental Modes
3. Four Preset Position Mode
4. Toggle Mode
5. General Purpose Input/Output functionality



Building Your Own Cable

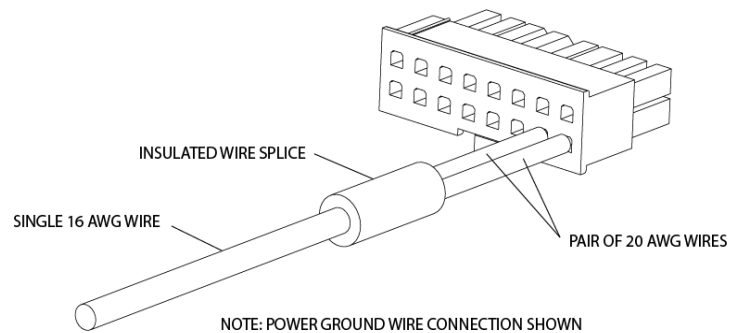
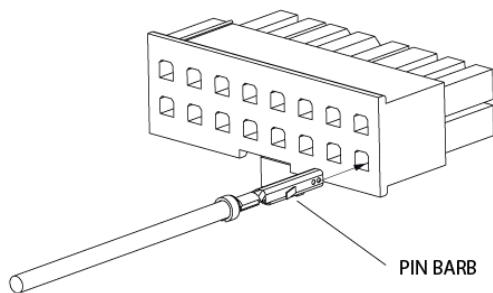
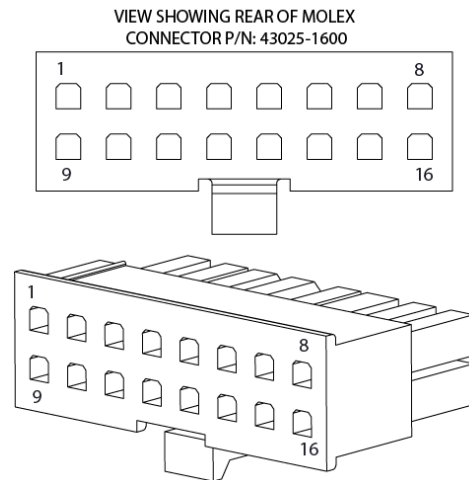
If you choose to create your own cable assembly, the following information will detail everything required to build your own.

The part numbers for the required Molex parts are listed below:

1. MOLEX Connector P/N: 43025-1600
2. MOLEX Crimping Tool P/N: 63819-000
3. MOLEX Crimp Pins (20-24 AWG): 43030-0008
4. MOLEX Pin Extraction Tool: 11-03-0043

The illustrations below provide the mating connector pinout and show proper orientation when inserting pins into a connector.

| PIN NUMBER | LABEL | FUNCTION |
|------------|-------|--------------------------------|
| 1 | ANI | Analog Current Input |
| 2 | GND | Ground |
| 3 | TX | RS-232 Transmit |
| 4 | DIO1 | General Purpose Input/Output 1 |
| 5 | IN2+ | Optically Isolated Input 2 (+) |
| 6 | IN1+ | Optically Isolated Input 1 (+) |
| 7 | V+ | Power |
| 8 | V+ | Power |
| 9 | ANV | Analog Voltage Input |
| 10 | GND | Ground |
| 11 | RX | RS-232 Receive |
| 12 | DIO2 | General Purpose Input/Output 2 |
| 13 | IN2- | Optically Isolated Input 2 (-) |
| 14 | IN1- | Optically Isolated Input 1 (-) |
| 15 | GND | Power Ground |
| 16 | GND | Power Ground |



Note: The pin barbs should always be on the side of pins as they are inserted into the socket



Note: The power wires to the Servo Cylinder require the use of 4 pins on the Molex connector. You will need to splice a pair of 20AWG wires into a single 16AWG wire for both the V+ and GND connections. Servo Cylinders **should not** be daisy chained together by their power pins.

Be sure to properly insulate your wire splices. The illustration above shows an example of a proper wire splice going into the two power GND pins

3.0 Operation

Getting Started

IMPORTANT: You must determine what control mode you wish to operate in before getting started. At a minimum, you must provide the Servo Cylinder with power and a command signal. If you need assistance in determining which control mode is right for your application, please contact one of our application engineers or review the control modes section.

Make sure you have read the Cabling Requirements section before getting started.

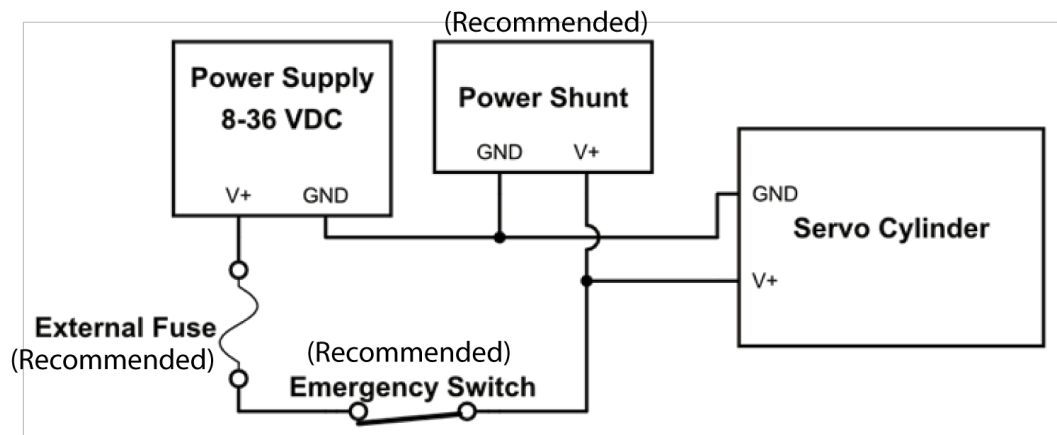
Connecting the Servo Cylinder

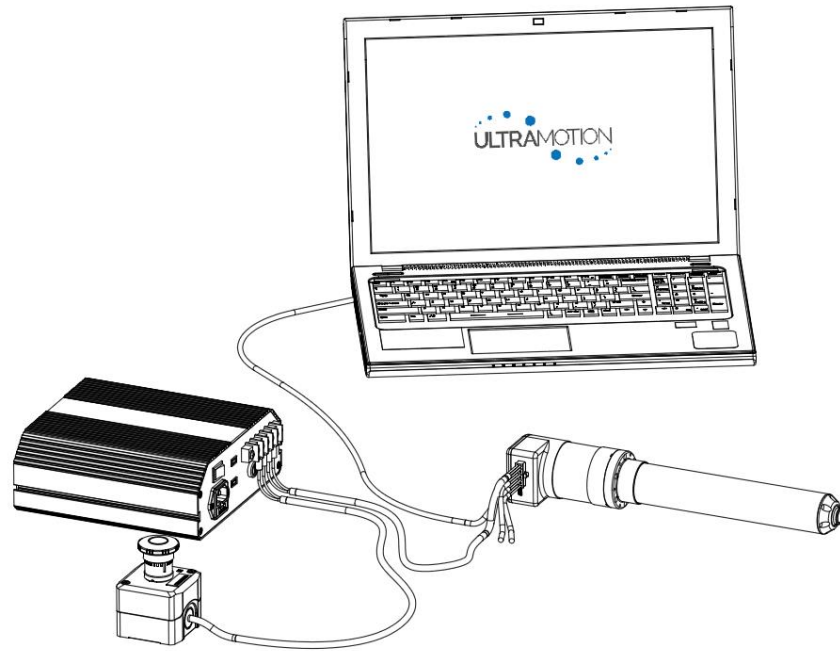
Wiring to a DC power source



DO NOT APPLY POWER TO THE SERVO CYLINDER UNTIL ALL WIRING IS COMPLETE.

Once you have your cabling, wire the power leads to a power supply as shown below. We recommend using a power shunt, external fuse, and emergency off switch. Please ensure you have properly designed your power system and addressed all safety concerns.





Note that the Ultra Motion power supply shown above has an internal DC side fuse, power shunt, and two external terminals for an emergency switch wired to properly maintain connections to the power shunt and protect the Servo Cylinder from back-emf in the event of a blown fuse, or engaged E-switch.

Wiring the Analog Inputs (Used for Proportional Mode with Voltage or Current Control Only)

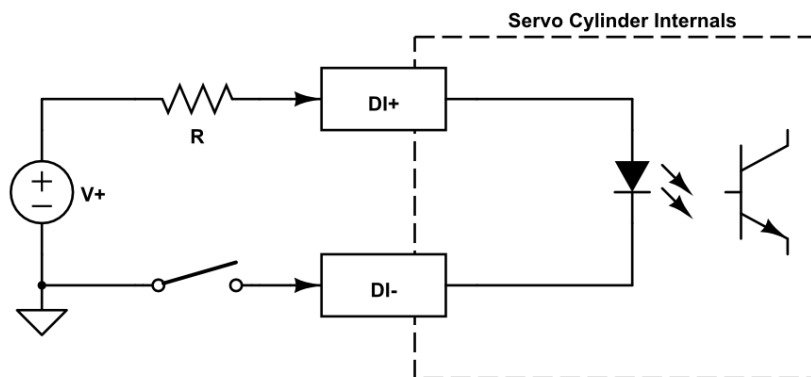
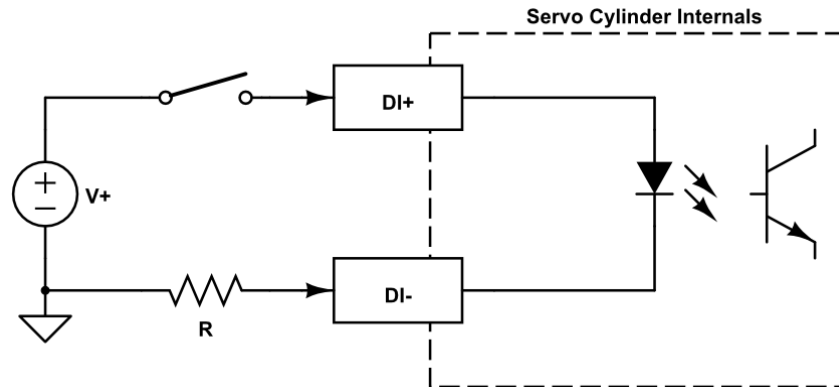
Care should be taken to ensure analog signals are properly shielded against EMI, and are routed separately from power wires and other sources of interference. When possible, a 100% foil shield should be used and the shield electrically connected to ground.

Wiring the Digital Inputs (Used for Proportional Mode with 1-2ms Pulse Control, Four Preset Position Mode, All Incremental Modes and access to the general purpose I/O)

The Servo Cylinder has two dedicated, optically isolated digital inputs. An opto-isolator transmits an electrical signal between two electrically isolated circuits via light. The use of optically isolated inputs allow for high data transfer rates, protection from high voltages, and very effective noise rejection.

For the highest level of noise rejection, the input signal and ground to the opto-isolator should be completely separate from the Servo Cylinder, but this is not a requirement. The +5V or ground of the optically isolated input can be supplied by one of the Servo Cylinder's on-board digital input/output pins.

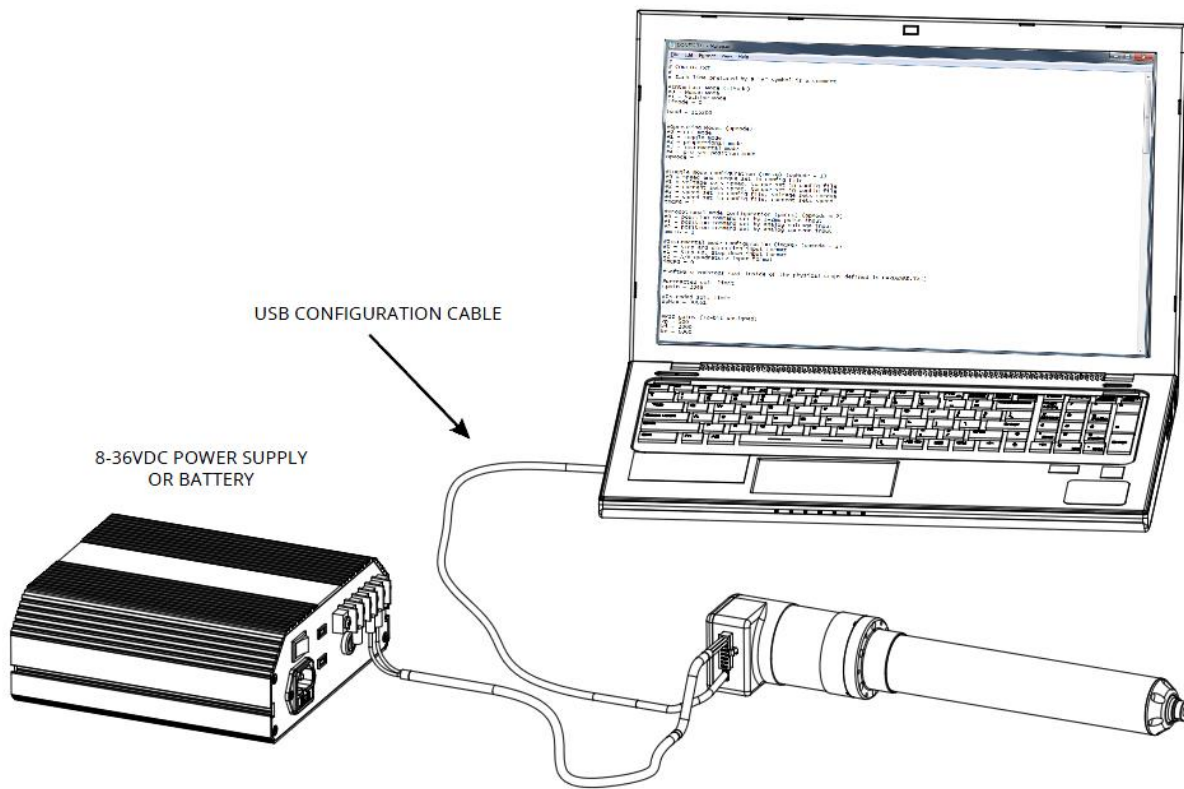
Suggested wiring for the optically isolated inputs is shown below for both high side and low side switching arrangements.



The Servo Cylinder's digital inputs can be driven by a +5VDC to +24VDC signal. Depending on the voltage, the suggested values for the external resistor (R) are as follows.

| Signal Voltage (V+) | Suggested Resistor (R) |
|---------------------|-------------------------------|
| +5 VDC | No external resistor required |
| +12 VDC | 430 Ohm Resistor |
| +24 VDC | 1.1 kOhm Resistor |

Configuring the Servo Cylinder



How the Configuration Process Works:

The Servo Cylinder has a small **10KB** section of non-volatile flash memory that is used to store configuration data. Upon startup, the Servo Cylinder's microprocessor looks for two files called **CONFIG.txt** (user defined parameters) and **HARDWARE.txt** (factory defined values). These files contain variables that determine how the actuator will behave.



WARNING: The Servo Cylinder's microcontroller needs both **HARDWARE.txt** and **CONFIG.txt** to operate. Deleting, removing, or renaming this file will result in a fault mode. We recommend creating a backup of these files to your local drive before starting to edit them.



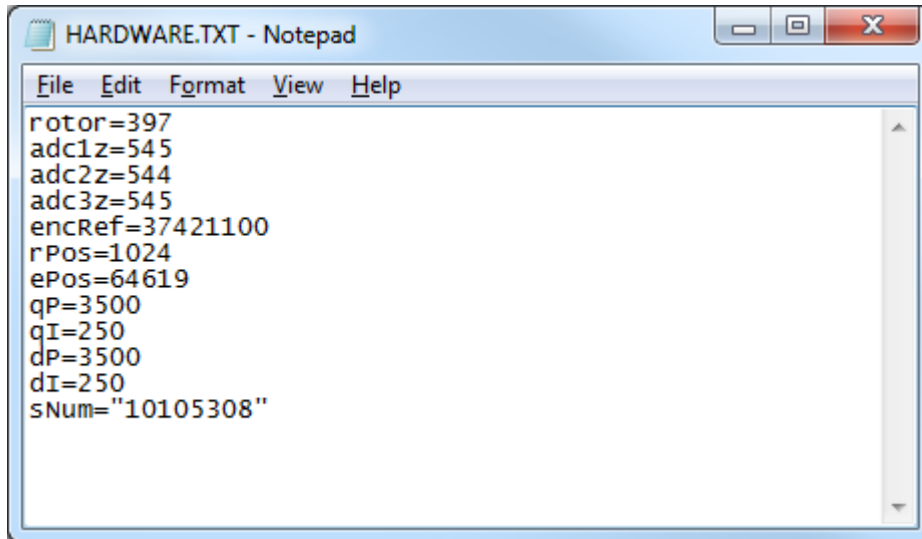
WARNING: Do not use the 10KB of flash memory to store backup copies of **CONFIG.txt** and **HARDWARE.txt**. Storing any files other than 1 **CONFIG.txt** and 1 **HARDWARE.txt** will use up the available storage space.

HARDWARE.txt

This file contains hardware specific information (such as FOC gains, sensor calibration information, etc.). This file is not intended to be edited by the user unless suggested by Ultra Motion engineers after a consultation. This

file is specific to each actuator and cannot be copied between multiple Servo Cylinders. If for some reason this file is lost, it can be automatically regenerated with a calibration routine command in the SCL language.

Example of a hardware.txt file:



```
rotor=397
adc1z=545
adc2z=544
adc3z=545
encRef=37421100
rPos=1024
ePos=64619
qP=3500
qI=250
dP=3500
dI=250
sNum="10105308"
```

CONFIG.txt

This file contains user settable variables that change how the actuator behaves (operating mode, retracted position limit, extended position limit, input ranges, etc.). CONFIG.TXT can be edited manually through any text file editor or generated using Ultra Motion's A1 Series configuration file generator. The configuration file generator offers a graphical interface and makes defining the configuration variables easier if you are not comfortable with manual configuration.

Once your configuration file is properly set for your application, this file can be copied across multiple Servo Cylinders. If you need to replace a Servo Cylinder, copy the existing CONFIG.txt file to the new Servo Cylinder and it is ready to go.

NOTE: The Servo Cylinder comes pre-configured to operate in SCL Mode. We recommend you configure the actuator for your application, but you can skip this step if you want to just run the actuator in SCL mode with default settings and travel limits.

Steps to complete for Manual Configuration of CONFIG.txt:

1. Use the provided USB cable to connect the Servo cylinder to a USB port on any Windows based computer.
2. Switch on your power supply. Note that you must power the Servo Cylinder before starting the configuration process. The USB cable is used only for data transmission and will not power the Servo Cylinder.
3. Once powered, the computer will automatically recognize the Servo Cylinder as a mass storage device.

Note: If the computer fails to automatically recognize the actuator, please contact Ultra Motion for support.

4. Browse to “my computer” and open the new storage drive labeled “Ultra Motion”.
5. Locate the text file “CONFIG.txt” and open it in a text editor.
6. Edit the variables to configure the actuator to behave as desired. Reference the following sections “Mandatory Configuration Variables for All Operating Modes”.
7. Based on your chosen operating mode, reference the relevant section for configuration variables specific to your control mode.
8. Save the file locally and then copy it to the Ultra Motion drive to REPLACE the config.txt file currently on the Servo Cylinder. If there is no CONFIG.txt file, just copy this new file onto the drive. Alternatively, you can download and save the website generated CONFIG.txt directly to the “Ultra Motion” drive.

Mandatory Configuration Variables for All Operating Modes

Several configuration variables are relevant to all operating modes and must be set for proper operation of the Servo Cylinder. The configuration variables that apply in all modes are the following:

opMode (Operating mode)

What it does: Sets the Servo Cylinder’s operating mode

Argument Type: Integer

Valid Range: 0 to 4 (See below for details)

0 = CLI mode

- Full control through Ultra Motion Serial Command Language
- Operates via RS-232 connection.
- Ultra Motion Serial Command Language includes a full command set for position moves, trajectory moves, acceleration, speed, torque, etc.

1 = toggle mode

- Extend/Retract control with adjustable acceleration, speed, and force.
- See tmCFG variable for additional configuration options.

2 = proportional mode

- Actuator moves proportional to the command signal.
- See pmCFG variable for additional configuration options.

3 = incremental mode

- Actuator moves as it receives each incremental command signal.
- See imCFG variable for additional configuration options.

4 = pre-set position mode

- Actuator has up to four preset positions.
- See Pos1, Pos2, Pos3, Pos4 variables for defining the positions.

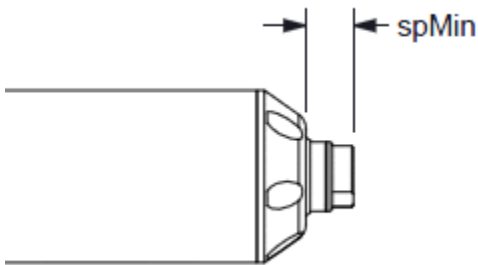
spMin (Software position minimum)

What it does: Sets the minimum allowable position of the Servo Cylinder. spMin can be treated as a retracted software limit switch. The value is expressed in phase index sensor counts and there are 1024 counts per revolution.

Argument Type: Integer

Valid Range: rPos to ePos (rPos and ePos are the physical travel limits which can be found in hardware.txt. rPos is always set to 1024)

Rules: $rPos \leq spMin < spMax$



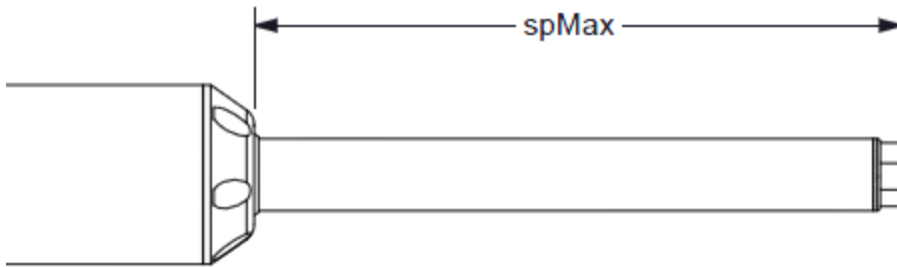
spMax (Software position maximum)

What it does: Sets the maximum allowable position of the Servo Cylinder. spMax can be treated as an extended software limit switch. The value is expressed in phase index sensor counts and there are 1024 counts per revolution.

Argument Type: Integer

Valid Range: rPos to ePos (rPos and ePos are the physical travel limits which can be found in hardware.txt. rPos is always set to 1024)

Rules: $spMin < spMax \leq ePos$



maxTorq (Max Torque)

What it does: Sets the limit for the current controlled by the Field Oriented Control loop, thereby limiting the maximum force produced by the Servo Cylinder. The value represents a percentage of force output whereas 32767 equals 100%. The relationship is linear with a slight offset do to unloaded running friction of the system.

Argument Type: Integer

Valid Range: 0 to 32767.

kp, ki, kd

What it does: These three values represent the gains for the proportional, integral, and derivative terms of the position PID control loop. They are to tune the response. See the section in the manual on PID tuning for more information about what these values represent.

Argument Type: Integer

Valid Range: See PID tuning section

Baud (Serial baud rate)

What it does: Sets the Baud rate for serial communication. Setting this is only necessary if you plan to use a serial connection to the actuator. Serial is accessible in all control modes and is used for streaming output information and diagnostics.

Argument Type: Integer

Valid Range: 2400 to 256,000

Notes: Default Baud rate is 115200. Lower baud rates are more tolerant to noise and cross talk.

Proportional Mode- Voltage Control Configuration:

The actuator position will be directly proportional to the value of the analog voltage supplied to the Servo Cylinder. You can set your input voltage range to be anywhere between -10 VDC to +10 VDC. Typical signal sources include PLC's, potentiometers, signal generators, etc.

pmCFG (Proportional Mode Configuration)

What it does: Sets behavior in proportional mode

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = position command set by 1-2ms pulse input

1 = position command set by analog voltage input

2 = position command set by analog current input

vMin/vMax

****Only applies when setting pmCFG=1**

What it does: Define the usable range of the -10 to +10 V input that will relate to the Servo Cylinder travel range (sMin to sMax).

Argument Type: Integer

Valid Range vMin: 0 to 65535

Valid Range vMax: 0 to 65535

Rules: vMin < vMax

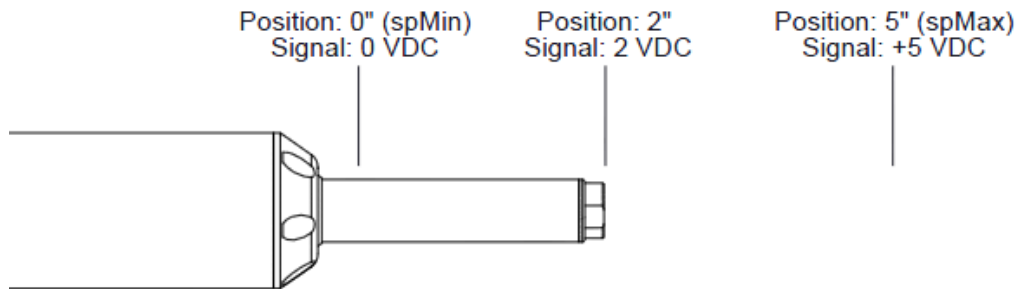
For optimal results the user should calibrate the minimum and maximum voltage range to each actuator.

vFilter (Voltage filter)

****Only applies when setting pmCFG=1**

What it does: The analog voltage input can be filtered in software to reduce the effect of noise from the transceiver. The larger the value the less filtering will occur, allowing for more dynamic movements. A smaller value will increase the level of filtering, leading to smoothed motion and less dithering.

Valid Range: 0 to 65535



Proportional Mode- Current Control Configuration:

The actuator position will be directly proportional to the value of the analog current supplied to the Servo Cylinder. You can set your input voltage range to be anywhere between 4-20 mA. Typical signal sources include PLC's, signal generators, etc.

pmCFG (Proportional Mode Configuration)

What it does: Sets behavior in proportional mode

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = position command set by 1-2ms pulse input

1 = position command set by analog voltage input

2 = position command set by analog current input

cMin/cMax (Current min/current max)

****Only applies when setting pmCFG=2**

What it does: Define the usable range of the 4 to 20 mA input that will relate to the Servo Cylinder travel range (sMin to sMax).

Pre-Requisite: opMode = analog current mode

Valid Range cMin: 0 to 65535

Valid Range cMax: 0 to 65535

Rules: cMin < cMax

For optimal results the user should calibrate the minimum and maximum voltage range to each actuator.

cFilter (Current filter)

****Only applies when setting pmCFG=1**

What it does: The analog current input can be filtered in software to reduce the effect of noise from the transceiver. The larger the value the less filtering will occur, allowing for more dynamic movements. A smaller value will increase the level of filtering, leading to more stable motion and less dithering.

Valid Range: 0 to 65535

Proportional Mode- 1-2ms Pulse Control Configuration:

The actuator position will be directly proportional to the value of the 1-2 ms pulse supplied to the Servo Cylinder. Typical signal sources include PLC's, RC Transceivers, etc.

pmCFG (Proportional Mode Configuration)

What it does: Sets behavior in proportional mode

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = position command set by 1-2ms pulse input

1 = position command set by analog voltage input

2 = position command set by analog current input

pMin/pMax (Pulse minimum and pulse maximum)

****Only applies when setting pmCFG=0**

What it does: Defines the usable range of 1-2 ms pulse inputs that will relate to the Servo Cylinder travel range (sMin to sMax).

0 = 0.954 ms

65535 = 2.046 ms

Argument Type: Integer

Valid Range pMin: 48000 to 70000

Valid Range pMax: 108000 to 132000

Example: To have a signal of 1.5 to 1.75 ms correspond to a position of sMin to sMax, the following values should be used for pMin and pMax.

$$pMin = \frac{(1.5 \text{ ms} - 0.954 \text{ ms})}{(2.046 \text{ ms} - 0.954 \text{ ms})} * 65535 \text{ counts} = 32,767$$

$$pMax = \frac{(1.75 \text{ ms} - 0.954 \text{ ms})}{(2.046 \text{ ms} - 0.954 \text{ ms})} * 65535 \text{ counts} = 47,770$$

pFilter (Pulse filter setting)

**Only applies when setting pmCFG=0

What it does: The 1-2 ms pulse input can be filtered in software to reduce the effect of noise from the transceiver. The larger the value the less filtering will occur, allowing for more dynamic movements. A smaller value will increase the level of filtering, leading to more stable motion and less dithering.

Range: 0 to 65535

Incremental Modes Configuration (Step and Direction, A/B Quadrature, Step & Direction, CW/CCW)

imCFG (Incremental mode configuration)

What it does: Sets the behavior in incremental mode

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = Step and direction input format

1 = Step up, Step down input format

2 = A/B quadrature input format

stepSize

What it does: The value of stepSize details the amount of Phase Index steps taken per incremental command.

Argument Type: Integer

Valid Range: 1 to 65535

Command Line Interface Mode Configuration

The user will have complete control over the Servo Cylinder as well as detailed diagnostic information. Reference Appendix A for the full set of available serial commands.

Baud (Serial baud rate)

What it does: Sets the Baud rate for serial communication.

Argument Type: Integer

Valid Range: 2400 to 256,000

Notes: Default Baud rate is 115200. Lower baud rates are more tolerant to noise and cross talk.

ifMode (Interface mode)

What it does: Set the behavior of the Serial Command Line Interface to be either human mode or machine mode.

Argument Type: Integer

Valid Range: 0-1 (See below for details)

0 = Human Mode

- All commands are followed by an ack or nack
- Asynchronous error messages are sent to the serial prompt
- Detailed information regarding actuator commands is written to the serial prompt

1 = Machine Mode

- Checksum follows every command
- Asynchronous error messages are disabled
- No detailed information is sent to the serial prompt
- (\ "SI0 CC\" sets interface back to Human Mode)

Serial Command Mode allows for use of the Servo Cylinder's built-in trajectory generator when moving from one position to the other. To define the characteristics of the trajectory generator, the maximum speed and acceleration values must be set.

maxSpeed (Maximum speed)

What it does: Limits the maximum speed the Servo Cylinder will reach when executing trajectory moves.

Argument Type: Integer

Valid Range: 0 to 50,000,000

$$l = \text{screw lead}$$

$$\text{maxSpeed} \left(\frac{\text{in}}{\text{s}} \right) = \frac{SP * 1,250 * l}{2^{16} * 1,024}$$

Accel (Acceleration and deceleration)

What it does: Defines the acceleration and deceleration the Servo Cylinder will use when calculating motion profiles for trajectory moves. Note: the acceleration and deceleration must be equal to one another.

Argument Type: Integer

Range: 0 to 131000

$$\text{Accel} \left(\frac{\text{in}}{\text{s}^2} \right) = \frac{AC * 1,250^2 * l}{2^{16} * 1024}$$

Preset Position Mode Configuration

Four preset position mode is a simple way to move between up to four absolute positions with simple digital commands. The state of the Servo Cylinder's two optically isolated digital inputs corresponds to the absolute position command. Four preset position mode uses the maximum speed and acceleration values (maxSpeed, accel) to create trajectories when moving from one preset position to another. A user configurable time delay (posTime) is used to reject noise or invalid states sent to the actuator.

posTime

What it does: Defines the amount of time in milliseconds it takes for a state change on the optically isolated digital inputs to register as a valid command.

Argument Type: Integer

Valid Range: 0 to 65535

Pos1

What it does: Defines the absolute position that corresponds to the first state of the digital inputs. This position is active when digital Input 1 = 0 and digital Input 2 = 0

Argument Type: Integer

Valid Range: spMin to spMax

Pos2

What it does: Defines the absolute position that corresponds to the second state of the digital inputs. This position is active when digital Input 1 = 1 and digital Input 2 = 0

Argument Type: Integer

Valid Range: spMin to spMax

Pos3

What it does: Defines the absolute position that corresponds to the third state of the digital inputs. This position is active when digital Input 1 = 0 and digital Input 2 = 1

Argument Type: Integer

Valid Range: spMin to spMax

Pos4

What it does: Defines the absolute position that corresponds to the fourth state of the digital inputs. This position is active when digital Input 1 = 1 and digital Input 2 = 1

Argument Type: Integer

Valid Range: spMin to spMax

maxSpeed (Maximum speed)

What it does: Limits the maximum speed the Servo Cylinder will reach when executing trajectory moves.

Argument Type: Integer

Valid Range: 0 to 50,000,000

$$l = \text{screw lead}$$

$$\text{maxSpeed} \left(\frac{\text{in}}{\text{s}} \right) = \frac{SP * 1,250 * l}{2^{16} * 1,024}$$

Accel (Acceleration and deceleration)

What it does: Defines the acceleration and deceleration the Servo Cylinder will use when calculating motion profiles for trajectory moves. Note: the acceleration and deceleration must be equal to one another.

Argument Type: Integer

Range: 0 to 131000

$$Accel \left(\frac{in}{s^2} \right) = \frac{AC * 1,250^2 * l}{2^{16} * 1024}$$

stopMode

What it does: Defines the stopping behavior when stopping

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = snap to position

1 = decelerate at rate set by “accel” variable

2 = brake for 100 ms, then snap to position

Toggle Mode Configuration

tmCFG (Toggle mode configuration)

What it does: Sets behavior in toggle mode

Argument Type: Integer

Valid Range: 0 to 4 (See below for details)

0 = speed and torque set in config file (both torque and speed are static settings)

1 = voltage sets speed, torque set in config file (dynamically controllable torque)

2 = current sets speed, torque set in config file (dynamically controllable torque)

3 = speed set in config file, voltage sets torque (dynamically controllable speed)

4 = speed set in config file, current sets torque (dynamically controllable speed)

mtMin (Max torque minimum)

****Only applies when tmCFG = 1 or tmCFG = 2**

What it does: Since you are using an adjustable range to set the torque limit, this variable defines the lower limit for the maxTorque variable.

Argument Type: Integer

Valid Range: 0 to 32766

For Example: You want to use a 0-10V input to control torque in a range from mtMin to maxTorque

msMin

**Only applies when tmCFG = 1 or tmCFG = 2

What it does: Since you are using a range to define the speed, this defines the lower limit for the range. The upper limit is defined by maxSpeed.

Argument Type: Integer

Valid Range: 0 to 32766

For Example: You want to use a 0-10V input to control speed in a range from msMin to maxSpeed

maxSpeed (Maximum speed)

What it does: Limits the maximum speed the Servo Cylinder will reach when executing trajectory moves.

Argument Type: Integer

Valid Range: 0 to 50,000,000

$l = \text{screw lead}$

$$\text{maxSpeed} \left(\frac{\text{in}}{\text{s}} \right) = \frac{SP * 1,250 * l}{2^{16} * 1,024}$$

Accel (Acceleration and deceleration)

What it does: Defines the acceleration and deceleration the Servo Cylinder will use when calculating motion profiles for trajectory moves. Note: the acceleration and deceleration must be equal to one another.

Argument Type: Integer

Range: 0 to 131000

$$\text{Accel} \left(\frac{\text{in}}{\text{s}^2} \right) = \frac{AC * 1,250^2 * l}{2^{16} * 1024}$$

stopMode

What it does: Defines the stopping behavior when stopping

Argument Type: Integer

Valid Range: 0 to 2 (See below for details)

0 = snap to position

1 = decelerate at rate set by “accel” variable

2 = brake for 100 ms, then snap to position

Configuring Outputs and Diagnostics (Available in all Control Modes)

General Purpose Input/Outputs

The Servo Cylinder has two GPIO pins that can be configured as a high impedance input, or to output signal ground, +5 VDC, the result of a status word bit mask, or a PWM data source. The following configuration variables control the functionality of the Servo Cylinder’s GPIO. Note: Damage will occur if the +5VDC output is shorted to GND.

ioPin1

What it does: Sets the operating mode for digital input/output pin 1

Argument Type: Integer

Valid Range: 0 to 4 (See below for details)

0 = High speed digital input

1 = Output ground (sink up to 2 Amps)

2 = Output +5V (source up to 250 mA)

3 = Output the result of the masked status register (1 kHz update rate)

4 = Output 1 kHz PWM proportional to a variable set by **ioPWM1**

ioPin2

What it does: Sets the operating mode for digital input/output pin 2

Argument Type: Integer

Valid Range: 0 to -4 (See below for details)

0 = High speed digital input

1 = Output ground (sink up to 2 Amps)

2 = Output +5V (source up to 250 mA)

3 = Output the result of the masked status register (1 kHz update rate)

4 = Output 1 kHz PWM proportional to a variable set by **ioPWM2**

ioPWM1

Pre-requisite: ioPin1 is configured to output a 1 kHz PWM waveform (ioPin1 = 4)

What it does: Sets the variable to be output via a 1 kHz PWM waveform on digital input/output pin 1.

Argument Type: Integer

Valid Range: 0 to 7 (See below for details)

- 0** = Actuator Position (spMin to spMax)
- 1** = Actuator Velocity (-512 to 511)
- 2** = Actuator Torque (-32768 to 32767)
- 3** = Bus Voltage (0 to 1023)
- 4** = Commanded position (spMin to spMax)
- 5** = 1-2 ms pulse input channel 2 (pMin to pMax)
- 6** = Analog voltage input (vMin to vMax)
- 7** = Analog current input (cMin to cMax)

ioPWM2

Pre-requisite: ioPin2 is configured to output a 1 kHz PWM waveform (ioPin2 = 4)

What it does: Sets the variable to be output via a 1 kHz PWM waveform on digital input/output pin 2.

Argument Type: Integer

Valid Range: 0 to 7 (See below for details)

- 0** = Actuator Position (spMin to spMax)
- 1** = Actuator Velocity (-512 to 511)
- 2** = Actuator Torque (-32768 to 32767)
- 3** = Bus Voltage (0 to 1023)
- 4** = Commanded position (spMin to spMax)
- 5** = 1-2 ms pulse input channel 2 (pMin to pMax)
- 6** = Analog voltage input (vMin to vMax)
- 7** = Analog current input (cMin to cMax)

ioBit1

Pre-Requisite: ioPin1 is configured to output the result of the masked status word (ioPin1 = 3)

What it does: The value of ioBit1 will be compared to the value of the status word, and a high or low bit will be signaled by the digital input output pin depending on whether or not the two are equal. See **Status Word** for more detailed diagnostic information.

Argument Type: Unsigned long

Valid Range: 0 to 4294967295 (2^{32})

ioBit2

Pre-Requisite: ioPin2 is configured to output the result of the masked status word (ioPin2 = 3)

What it does: The value of ioBit2 will be compared to the value of the status word, and a high or low bit will be signaled by the digital input output pin depending on whether or not the two are equal. See **Status Word** for more detailed diagnostic information.

Argument Type: Unsigned long

Valid Range: 0 to 4294967295 (2^{32})

Data Streaming

Pre-Requisite: Serial connection is made to the actuator

The Servo Cylinder can report the values of up to three variables over serial at a defined rate. This is useful for debugging, graphing output data, and confirming Servo Cylinder performance. The following configuration variables control the serial data streaming functionality:

sFlag

What it does: Defines the start-up behavior of the data streaming functionality.

Argument Type: Integer

Valid Range: 0 to 1 (See below for details)

0 = data streaming off at startup

1 = data streaming on at power-up

sTime

What it does: Defines the rate at which data is streamed from the actuator. This will be limited by the pre-selected baud rate. 65535 is the minimum data transfer rate (~3.576 Hz), the maximum data transfer rate is a function of the baud rate.

Argument Type: Integer

Valid Range: 0 to 65535

| Baud Rate | Approximate Maximum Streaming Frequency |
|------------------|--|
| 9600 | |
| 19200 | |
| 38400 | |
| 57600 | |
| 115200 | |

Strx1

What it does: Defines the variable that will be streamed in column 1

Argument Type: Integer

Valid Range: 0 to 19 (See “Streaming variables” below for details)

Strx2

What it does: Defines the variable that will be streamed in column 2

Argument Type: Integer

Valid Range: 0 to 19 (See “Streaming variables” below for details)

Strx3

What it does: Defines the variable that will be streamed in column 3

Argument Type: Integer

Valid Range: 0 to 19 (See “Streaming variables” below for details)

Streaming Variables

- 0 = Column Off
- 1 = Position Feedback
- 2 = Position Setpoint
- 3 = Motor Current Feedback
- 4 = Motor Current Setpoint
- 5 = Motor Phase 1 Current
- 6 = Motor Phase 2 Current
- 7 = Motor Phase 3 Current
- 8 = Velocity at 1.25 kHz
- 9 = Velocity at 156.25 Hz
- 10 = Encoder Interval Timer

- 11 = Supply Voltage
- 12 = Supply Voltage Rate of Change
- 13 = Status Register
- 14 = 1-2 ms Pulse Input
- 15 = Filtered 1-2 ms Pulse Input
- 16 = Analog Voltage Input
- 17 = Filtered Analog Voltage Input
- 18 = Analog Current Input
- 19 = Filtered Analog Current Input
- 20 = Temperature Sensor Input
- 21 = Position Error
- 22 = Quadrature Input Errors
- 23 = 32-bit Counter with 100 μ s Increments

Status Word

Pre-Requisite: Serial connection is made to the actuator

The Servo Cylinder updates a 32-bit status word to provide the user with an indication of health. Several of the bits require a user defined threshold that defines when they are active and when they are inactive. The following configuration variables are used to define thresholds that trigger several of the status bits:

Several of the bits in the status word require a user defined threshold that defines when they trigger.

atTarg

What it does: Defines the position error threshold for status bit 6 “at target position”. The “at target position” bit will flip when the actuator position falls between setpoint \pm atTarg.

Argument Type: Integer

Valid Range: 0 to 4095

ovErrP

What it does: Defines the position error threshold (status bit 7) is active when the actuator error exceeds the user defined maximum.

Argument Type: Integer

Valid Range: 0 to 65535

ovSpeed

What it does: Defines the speed over threshold (status bit 8) is active when the actuator speed exceeds the user defined maximum.

Argument Type: Integer

Valid Range: 0 to 127

ovTorq

What it does: Torque Over Threshold (status bit 9) is active when the actuator torque exceeds the user defined maximum.

Argument Type: Integer

Valid Range: 0 to 32767

posGrtr

What it does: Position greater than threshold is a bit that defines what portion of the stroke the actuator is currently within. For example, setting posGrtr = 35000 will cause an active bit if the actuator position is 35001 and an inactive bit if the actuator position is 34999.

Argument Type: Integer

Valid Range: 0 to 65535

posLess

What it does: Position less than threshold is a bit that defines what portion of the stroke the actuator is currently within. For example, setting posless = 5000 will cause an active bit if the actuator position is 4999 and an inactive bit if the actuator position is 5001.

Argument Type: Integer

Valid Range: 0 to 65535

Using the Servo Cylinder



IT IS A GOOD IDEA TO RUN THE SERVO CYLINDER UNLOADED ON A BENCHTOP TO GAIN FAMILIARITY WITH THE PRODUCT BEFORE USING IT TO DRIVE ANY LOAD.

Once you have configured your Servo Cylinder. You are ready start sending a command signal.

Tuning Performance

PID Basics

The PID circuit is often utilized as a control loop feedback controller and is very commonly used for many forms of servo circuits. The letters making up the acronym PID correspond to Proportional (P), Integral (I), and Derivative (D), which represents the three control settings of a PID circuit. The purpose of any servo circuit is to hold the system at a predetermined value (set point) for long periods of time. The PID circuit actively controls the system so as to hold it at the set point by generating an error signal that is essentially the difference between the set point and the current value. The three controls relate to the time-dependent error signal; at its simplest, this can be thought of as follows: Proportional is dependent upon the present error, Integral is dependent upon the accumulation of past error, and Derivative is the prediction of future error. The results of each of the controls are then fed into a weighted sum, which then adjusts the output of the circuit, $u(t)$. This output is fed into a control device, its value is fed back into the circuit, and the process is allowed to actively stabilize the circuit's output to reach and hold at the set point value. The block diagram below illustrates very simply the action of a PID circuit. One or more of the controls can be utilized in any servo circuit depending on system demand and requirement (i.e., P, I, PI, PD, or PID).

Proportional Response

The proportional component depends only on the difference between the set point and the process variable. This difference is referred to as the Error term. The *proportional gain* (K_c) determines the ratio of output response to the error signal. For instance, if the error term has a magnitude of 10, a proportional gain of 5 would produce a proportional response of 50. In general, increasing the proportional gain will increase the speed of the control system response. However, if the proportional gain is too large, the process variable will begin to oscillate. If K_c is increased further, the oscillations will become larger and the system will become unstable and may even oscillate out of control.

Integral Response

The integral component sums the error term over time. The result is that even a small error term will cause the integral component to increase slowly. The integral response will continually increase over time unless the error is zero, so the effect is to drive the Steady-State error to zero. Steady-State error is the final difference between the process variable and set point. A phenomenon called integral windup results when integral action saturates a controller without the controller driving the error signal toward zero.

Derivative Response

The derivative component causes the output to decrease if the process variable is increasing rapidly. The derivative response is proportional to the rate of change of the process variable. Increasing the *derivative time* (T_d) parameter will cause the control system to react more strongly to changes in the error term and will increase the speed of the overall control system response. Most practical control systems use very small derivative time (T_d), because the Derivative Response is highly sensitive to noise in the process variable signal. If

the sensor feedback signal is noisy or if the control loop rate is too slow, the derivative response can make the control system unstable

| Parameter Increased | Rise Time | Overshoot | Settling Time | Steady-State Error | Stability |
|---------------------|----------------|----------------|----------------|------------------------|----------------------------|
| K_p | Decrease | Increase | Small Change | Decrease | Degrade |
| K_i | Decrease | Increase | Increase | Decrease Significantly | Degrade |
| K_d | Minor Decrease | Minor Decrease | Minor Decrease | No Effect | Improve (for small K_d) |

Tuning

In general the gains of P, I, and D will need to be adjusted by the user in order to best servo the system. While there is not a static set of rules for what the values should be for any specific system, following the general procedures should help in tuning a circuit to match one's system and environment. In general a PID circuit will typically overshoot the SP value slightly and then quickly damp out to reach the SP value.

Manual tuning of the gain settings is the simplest method for setting the PID controls. However, this procedure is done actively (the PID controller turned on and properly attached to the system) and requires some amount of experience to fully integrate. To tune your PID controller manually, first the integral and derivative gains are set to zero. Increase the proportional gain until you observe oscillation in the output. Your proportional gain should then be set to roughly half this value. After the proportional gain is set, increase the integral gain until any offset is corrected for on a time scale appropriate for your system. If you increase this gain too much, you will observe significant overshoot of the SP value and instability in the circuit. Once the integral gain is set, the derivative gain can then be increased. Derivative gain will reduce overshoot and damp the system quickly to the SP value. If you increase the derivative gain too much, you will see large overshoot (due to the circuit being too slow to respond). By playing with the gain settings, you can maximize the performance of your PID circuit, resulting in a circuit that quickly responds to changes in the system and effectively damps out oscillation about the SP value.

| Control Type | K_p | K_i | K_d |
|--------------|------------|---------------|-------------|
| P | $0.50 K_u$ | - | - |
| PI | $0.45 K_u$ | $1.2 K_p/P_u$ | - |
| PID | $0.60 K_u$ | $2 K_p/P_u$ | $K_p P_u/8$ |

While manual tuning can be very effective at setting a PID circuit for your specific system, it does require some amount of experience and understanding of PID circuits and response. The Ziegler-Nichols method for PID tuning offers a bit more structured guide to setting PID values. Again, you'll want to set the integral and derivative gain to zero. Increase the proportional gain until the circuit starts to oscillate. We will call this gain level K_u . The oscillation will have a period of P_u . Gains for various control circuits are then given below in the chart.

Updating the Firmware

Introduction: The firmware on the Servo Cylinder is field upgradeable through a serial connection. The update process uses a windows based utility called PC Loader.exe to load the new version of the firmware onto your Servo Cylinder. Contact Ultra Motion to obtain a copy of firmware and the windows update utility before getting started.

Pre-requisites:

- A cable assembly with a minimum of a power and serial connection.
- A USB mini-B cable
- The PC Loader.exe utility.
- The firmware .HEX file.

Update Instructions:

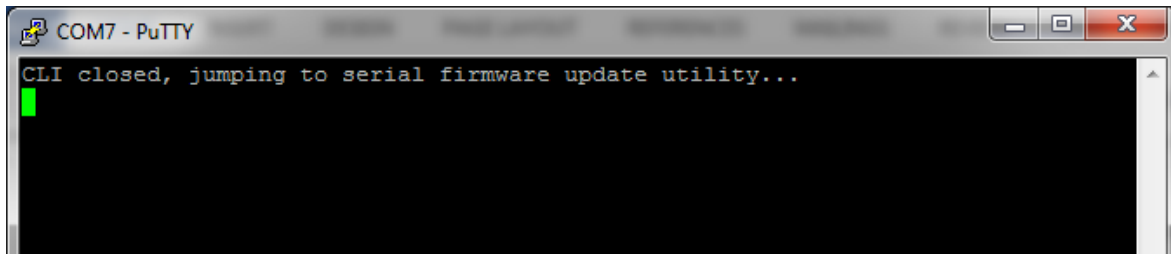


WARNING: While updating your firmware the Servo Cylinder will be put into an inoperable state. Make sure there is no load applied to the Servo Cylinder before starting this process. It is a good practice to remove the Servo Cylinder from the machine or installed location before initiating the update.

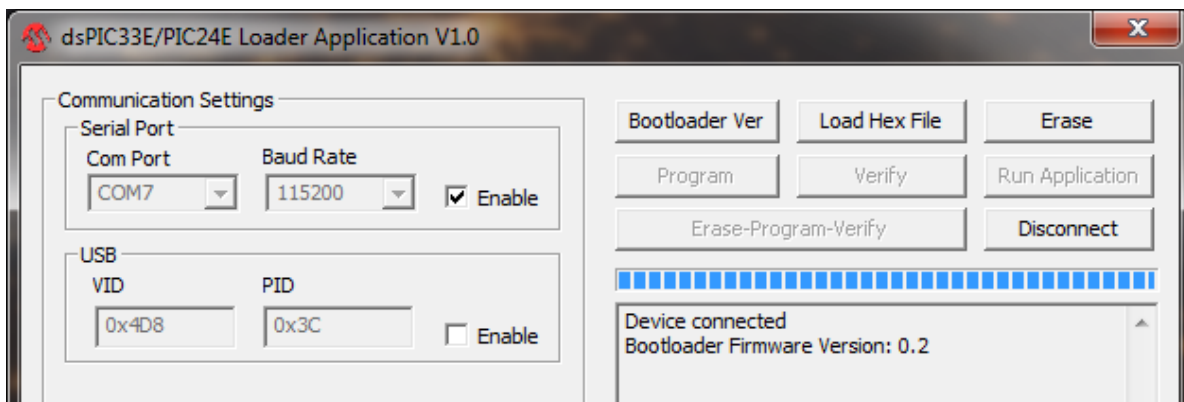
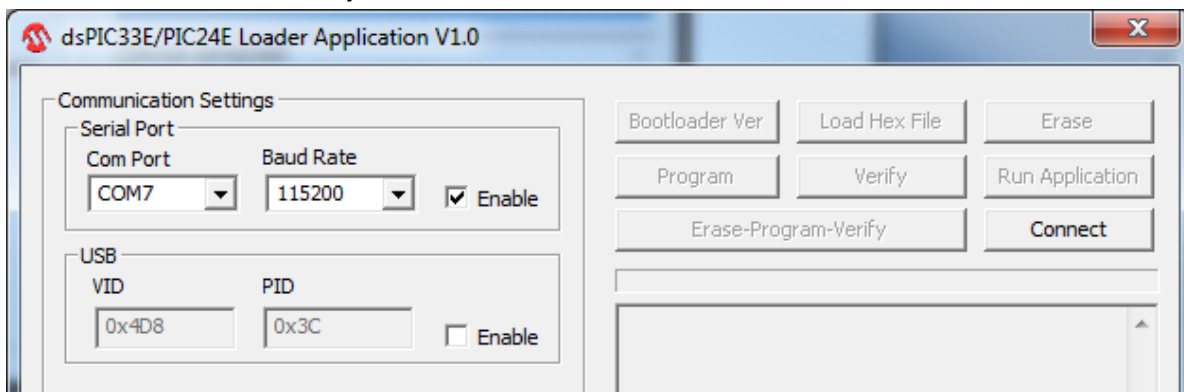


WARNING: Do not unplug or power cycle the Servo Cylinder until the update process has completed.

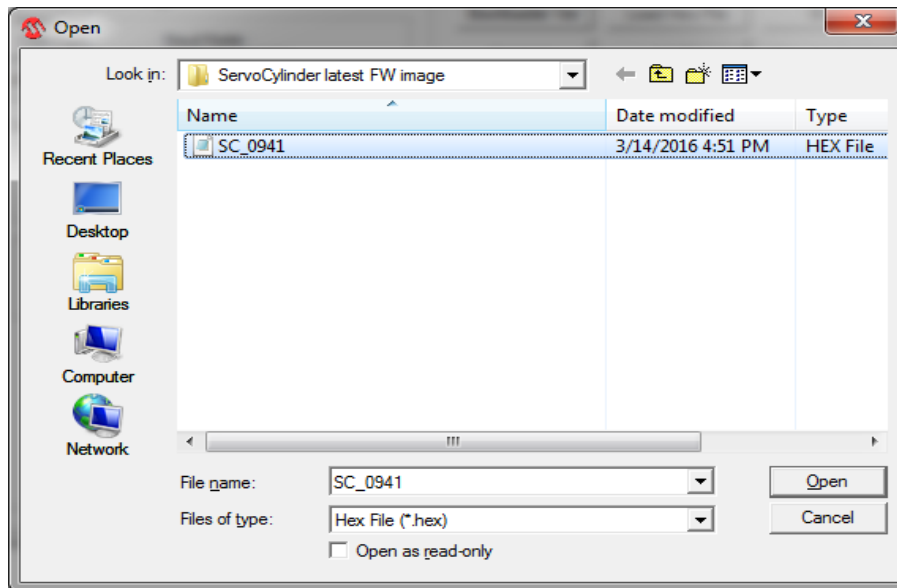
1. Connect the Servo Cylinder to the serial port on a windows computer.
2. Connect the “config” port on the Servo Cylinder to a USB port on your windows computer with the USB mini-B cable.
3. Connect the Servo Cylinder to a power supply and turn on power.
4. On the windows computer, browse to the drive labeled “Ultra Motion”.
5. Copy the Config.txt and Hardware.txt files on the drive and paste them in a location of your choice on the computer to create a backup.
6. Determine which COM port you are connected to:
 - a. Open Windows Device Manager.
 - b. Expand the port section.
 - c. Determine which of the listed ports is connected to your Servo Cylinder.
7. Open a serial terminal program such as Putty. Establish a serial connection (select appropriate COM port and baud rate). The default baud rate is 115200. If you have changed this in your config.txt file, please use whatever value is listed there.
8. Press Enter to ensure that the CLI prompt “>” is printed to the terminal.
9. With a serial connection established, type serial command “ZU321” to enter update mode. The STATUS LED on the Servo Cylinder will flash either blue or blue/green. The image below is what you should see in the terminal window after entering the ZU321 command.



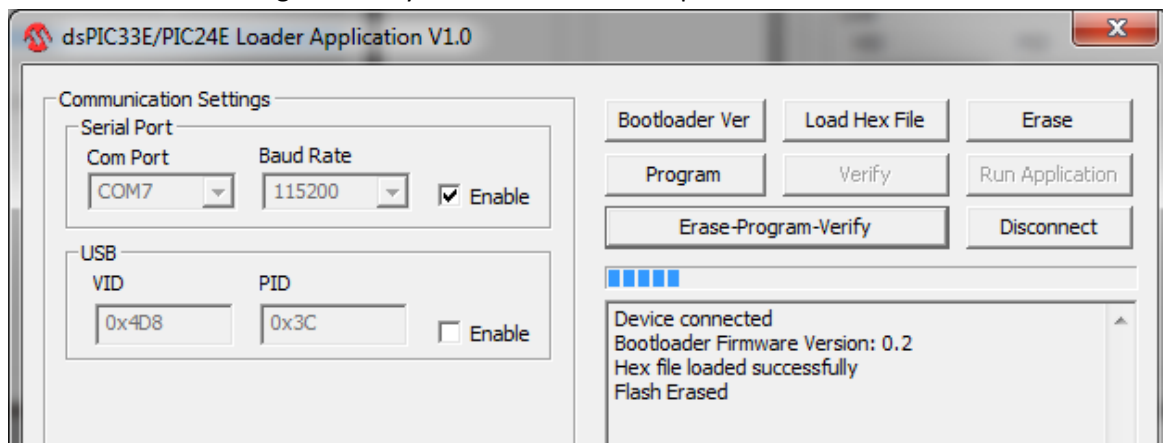
10. Once the updater has been started (indicated by flashing blue or blue/green STATUS LED or the message shown above), close the terminal emulator (Putty or similar) to free the PC's COM port for the firmware update utility.
11. Launch the PC Loader.exe program.
12. Select the appropriate COM port. The baud rate for the firmware updater is fixed at 115200.
13. Hit the "connect" button as shown below to establish connection. Once connected you will see confirmation. **It must say Bootloader Firmware Version 0.2** to indicate a successful connection.



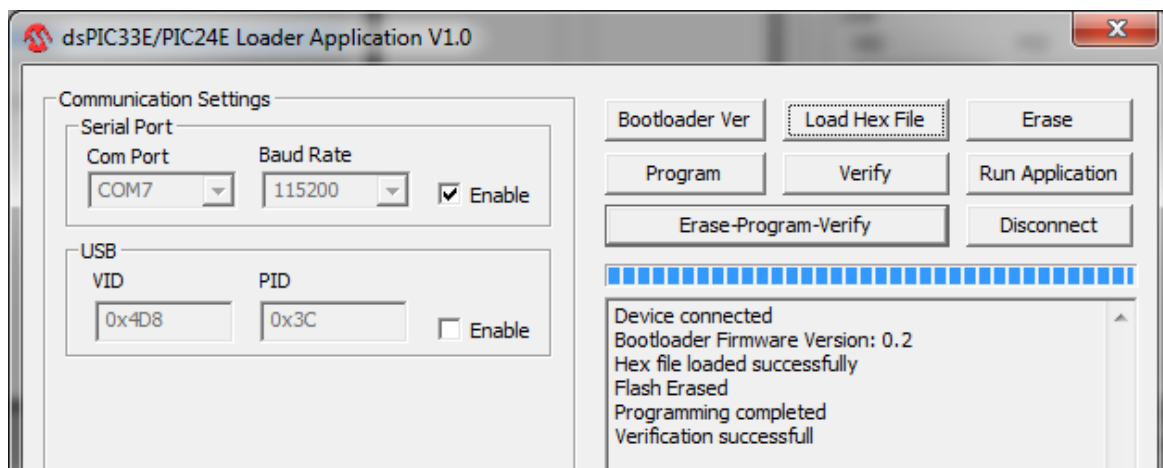
14. Once connected, click "Load Hex File" and navigate to the firmware file you would like to use for the update (Intel .hex file format). Click "Open"



15. Click on the “Erase-Program-Verify” button to start the update.



16. Once complete. You will see “programming complete” and “verification successful”. See screenshot below.



17. Click “Run Application” to run the new firmware image. Since neither the config.txt or the hardware.txt file are present (both have been overwritten by the firmware update), the Servo Cylinder will start in Safe Mode with the motor disabled and the STATUS LED solid red.
18. Close the PC_Loader application.
19. With the USB cable connected, browse to the drive labeled “Ultra Motion” on the windows computer.
20. Paste the previously backed up hardware.txt and config.txt files onto the “Ultra Motion” drive.
The firmware update will leave an empty config.txt on the drive. Overwrite this empty file with the backed up version of config.txt.
21. Restart Servo Cylinder. This can be done by cycling the power, or issuing the command “ZR321” to the CLI. After restart, the STATUS LED should be solid green.
22. If the STATUS LED is red after restart, establish a serial connection via putty or similar program. Power cycle the actuator and note the error message(s) displayed in the terminal window. Contact Ultra Motion for assistance troubleshooting any error.

Troubleshooting

LED Status Light Definitions

The “Status” LED on the rear cover will provide visual indications of the Status.

SOLID GREEN LED: Normal Operation

SOLID YELLOW LED: Actuator is outside of software travel limits

SOLID RED LED: Critical error in config.txt and/or hardware.txt. Establish a serial connection and reboot the Servo Cylinder. Upon power up, the error will be displayed in the command window. You can manually fix the error or try replacing the CONFIG or HARDWARE files with a backup copy

SOLID BLUE: Critical System Error – Try cycling the power to the Servo Cylinder. If the solid blue light persists, please contact Ultra Motion for technical support

SINGLE RED FLASH: Indicates a fault has occurred. Each flash indicates a new unique instance of the fault. The flash will happen right when the fault occurs. The actuator will continue to operate after a fault has occurred and you monitor/read must read the status register via a serial connection to get details about the fault. See Appendix X for a list of errors.

Clearing Faults

Use serial command ZF to reset the status register

Appendix A: CLI Command Reference Command List

| | |
|--|----|
| Read Commands..... | 44 |
| AN – Read Analog Inputs | 44 |
| AP – Read Actuator Position..... | 44 |
| BV – Read Bus Voltage..... | 44 |
| DI – Read Digital Input..... | 44 |
| FS – Read Firmware Version and Serial Number | 44 |
| HE – Display the Help Table..... | 44 |
| QP – Read Position Setpoint..... | 44 |
| RC – Read 1-2 ms Pulse Input..... | 45 |
| SR – Read Status Register | 45 |
| TQ – Read Motor Torque..... | 45 |
| VL – Read Actuator Velocity | 45 |
| Set Commands..... | 46 |
| AC – Set Acceleration for Trajectory Moves..... | 46 |
| CG – Coast On..... | 46 |
| CS – Coast Off | 46 |
| D1 – Set Digital I/O Pin 1 Mode | 46 |
| D2 – Set Digital I/O Pin 2 Mode..... | 46 |
| H1 – Set the Source for PWM Duty Cycle on I/O pin1 | 47 |
| H2 – Set the Source for PWM Duty Cycle on I/O pin2 | 47 |
| IM – Set Serial Interface Mode..... | 48 |
| KD - Set Derivative Gain for Position PID Filter | 48 |
| KI - Set Integral Gain for Position PID Filter..... | 48 |
| KP – Set Proportional Gain for Position PID Filter | 48 |
| MT – Set Limit for Maximum Commanded Torque..... | 48 |
| PA – Set Absolute Target Position | 48 |
| PC – Set Target Position to Current Position..... | 49 |
| PO – Set Target Position Offset | 49 |
| SP – Set Maximum Speed for Trajectory Moves | 49 |

| | |
|---|-------------------------------------|
| W1 – Set Status Register Bit Mask for I/O pin1..... | 49 |
| W2 – Set Status Register Bit Mask for I/O pin2..... | 50 |
| System Commands | 51 |
| ZC – Run Calibration Routine..... | 51 |
| ZD– Read Bridge Driver DIAG Register | 51 |
| ZF – Clear Latching Fault Flags in Status Register..... | 51 |
| ZR – Reset Actuator | 51 |
| ZU – Jump to RS-232 Firmware Updater | 51 |
| Trajectory Commands | 52 |
| T1 – Trajectory Move to User Defined Pos1..... | 52 |
| T2 – Trajectory Move to User Defined Pos2..... | 52 |
| T3 – Trajectory Move to User Defined Pos3..... | 52 |
| T4 – Trajectory Move to User Defined Pos4..... | 52 |
| TA – Trajectory Move to Absolute Position..... | 52 |
| TE – Trajectory Move to Extended Position (sMax) | 52 |
| TK – Interrupt the current trajectory | 52 |
| TM – Trajectory Move to Midpoint ($sMin + [sMax - sMin]/2$) | 53 |
| TO – Trajectory Move to Offset (incremental trajectory move) | 53 |
| TR – Trajectory Move to Retracted Position (sMin) | 53 |
| Serial Data Streaming | 54 |
| X1 – Set Data Stream Column 1..... | 54 |
| X2 – Set Data Stream Column 2..... | 54 |
| X3 – Set Data Stream Column 3..... | 54 |
| XC – Clear Timer | 54 |
| XG – Turn Data Streaming On..... | 55 |
| XR – Display Data Stream Setup | 55 |
| XS – Turn Data Streaming Off..... | 55 |
| XT – Set Data Stream Time Interval..... | 55 |
| XT = 2344 \approx 100 Hz XT = 23438 \approx 10 Hz..... | Error! Bookmark not defined. |

Read Commands

AN – Read Analog Inputs

Argument Type: None, 1 to 3

Description: Request the value of the Servo Cylinder's voltage input, current input, or built-in temperature sensor. Sending this command with no argument will return the value for all three analog inputs. Returned values will be unsigned integers with the following conversions:

AP – Read Actuator Position

Argument Type: None

Description: The AP command will output the absolute position of the Servo Cylinder. One revolution of the Servo Cylinder's screw is equivalent to 1,024 counts of Phase Index.

BV – Read Bus Voltage

Argument Type: None

Description: Calling BV will return a value from 0 to 1023 (0 Volts to 50 Volts), which represents the value of the Servo Cylinder supply voltage.

DI – Read Digital Input

Argument Type: None, 1 to 4

Description: Request the value of the Servo Cylinder's digital inputs.

DI1 = value of OPTO1

DI2 = value of OPTO2

DI3 = value of DIO1

DI4 = value of DIO2

Sending this command with no argument will return the value for all of the digital inputs.

FS – Read Firmware Version and Serial Number

Argument Type: None

Description: Request the firmware version currently running on the Servo Cylinder as well as the unique product serial number.

HE – Display the Help Table

Argument Type: None

Description: Sending this command will print out a list of all Serial CLI commands.

QP – Read Position Setpoint

Argument Type: None

Description: Request the position setpoint currently being commanded by the chosen operating mode.

RC – Read 1-2 ms Pulse Input

Argument Type: None

Description: Request the value of the 1-2 ms pulse being read by the Servo Cylinder. This command is only valid in pulse mode.

SR – Read Status Register

Argument Type: None, 0-31

Description: Sending the SR command without an argument will return the value of the status register in hex format. Calling the SR command followed by a value from 0 to 31 will return the value of the corresponding status register bit.

For example:

Input: SR 4

Output: bit4: "Emergency over-volt" = false

Input: SR

Output: Status Register: 0x000610A7

Refer to **Appendix C: Status Register** for detailed information regarding the status word.

TQ – Read Motor Torque

Argument Type: None

Description: Request the transformed three phase current feedback, which is a measure of motor torque (-32768 to 32767).

VL – Read Actuator Velocity

Argument Type: None

Description: Request the velocity of the actuator as measured by the change in Phase Index over a time interval. Negative values represent motion towards the retracted direction; positive values represent motion towards the extended position.

Set Commands

AC – Set Acceleration for Trajectory Moves

Argument Type: Integer (0 to 65535)

Definition: Set the acceleration/deceleration value to be used in the Servo Cylinder's built in trajectory generator. Sending this command without an argument will return the current value of AC.

$$l = \text{screw lead}$$
$$\text{Accel} \left(\frac{\text{in}}{\text{s}^2} \right) = \frac{AC * 1,250^2 * l}{2^{16} * 1024}$$

CG – Coast On

Argument Type: Integer, 321

Description: Disable the Servo Cylinder.

Notes: Enabling Coast electrically decouples the motor from the power stage, removing all power from the motor. Enabling coast is not recommended for applications in which backdriving is a possibility. **Care must be taken when issuing this command!**

CS – Coast Off

Argument Type: Integer, 321

Description: Return power to the BLDC motor allowing for normal operation of the Servo Cylinder.

Notes: Re-energizing the motor will cause the Servo Cylinder to move to the current commanded position. **Care must be taken when disabling Coast!**

D1 – Set Digital I/O Pin 1 Mode

Argument Type: Integer, 0 to 4

Description: Set the function of the Servo Cylinder's digital input/output pin 1

- 0 = Input (high impedance)
- 1 = Output: Ground (sink up to 2A)
- 2 = Output: +5 Volts (source up to 250 mA)
- 3 = Output: Status Bit (See **Outputting Status Bits**)
- 4 = Output: 1 kHz PWM (See **H1 – Set the Source for PWM duty Cycle**)

Sending this command without an argument will return the current value of D1.

D2 – Set Digital I/O Pin 2 Mode

Argument Type: Integer, 0 to 4

Description: Set the function of the Servo Cylinder's digital input/output pin 2

- 0 = Input (high impedance)
- 1 = Output: Ground (sink up to 2A)
- 2 = Output: +5 Volts (source up to 250 mA)
- 3 = Output: Status Bit (See **Outputting Status Bits**)
- 4 = Output: 1 kHz PWM (See **H2 – Set the Source for PWM duty Cycle**)

Sending this command without an argument will return the current value of D2.

H1 – Set the Source for PWM Duty Cycle on I/O pin1

Argument Type: Integer, 0 to 7

Pre-Requisite: Digital Input/Output Pin 1 is set to output a 1kHz PWM waveform (See **D1**)

Description: The value assigned to H1 represents the source for the PWM duty cycle. All source ranges are converted to the PWM duty cycle range of 0 to 60,000

- 0 = Actuator Position (sMin to sMax)
- 1 = Actuator Velocity (-512 to 511)
- 2 = Actuator Torque (-32768 to 32767)
- 3 = Bus Voltage (0 to 1023)
- 4 = Commanded Position (sMin to sMax)
- 5 = 1-2 ms pulse input channel (pMin to pMax)
- 6 = Analog voltage input (vMin to vMax)
- 7 = Analog current input (cMin to cMax)

Sending this command without an argument will return the current value of H1.

H2 – Set the Source for PWM Duty Cycle on I/O pin2

Argument Type: Integer, 0 to 7

Pre-Requisite: Digital Input/Output Pin 2 is set to output a 1kHz PWM waveform (See **D2**)

Description: The value assigned to H2 represents the source for the PWM duty cycle. All source ranges are converted to the PWM duty cycle range of 0 to 60,000

- 0 = Actuator Position (sMin to sMax)
- 1 = Actuator Velocity (-512 to 511)
- 2 = Actuator Torque (-32768 to 32767)
- 3 = Bus Voltage (0 to 1023)
- 4 = Commanded Position (sMin to sMax)
- 5 = 1-2 ms pulse input channel (pMin to pMax)
- 6 = Analog voltage input (vMin to vMax)
- 7 = Analog current input (cMin to cMax)

Sending this command without an argument will return the current value of H2.

IM – Set Serial Interface Mode

Argument Type: integer

Description: The serial interface can be set to operate in “Human Mode” or “Machine Mode”

- 0 = Human Mode
 - All commands are followed by an ack or nack
 - Asynchronous error messages are sent to the serial prompt
 - Detailed information regarding actuator commands is written to the serial prompt
- 1 = Machine Mode
 - Checksum follows every command
 - Asynchronous error messages are disabled
 - No detailed information is sent to the serial prompt
 - (\“SI0 CC\” sets interface back to Human Mode)

KD - Set Derivative Gain for Position PID Filter

Argument Type: Integer (0 to 32767)

Description: Coefficient for derivative portion of the position PID filter. Sending this command without an argument will return the current value of KD.

KI - Set Integral Gain for Position PID Filter

Argument Type: Integer (0 to 32767)

Description: Coefficient for integral portion of the position PID filter. Sending this command without an argument will return the current value of KI.

KP – Set Proportional Gain for Position PID Filter

Argument Type: Integer (0 to 32767)

Description: Coefficient for proportional portion of the position PID filter. Sending this command without an argument will return the current value of KP.

MT – Set Limit for Maximum Commanded Torque

Argument Type: Integer (0 to 32767)

Description: Vary the maximum allowable force produced by the Servo Cylinder. MT works by limiting the maximum commanded current in the Field Oriented Control loop. Sending this command without an argument will return the current value of MT.

PA – Set Absolute Target Position

Argument Type: None, Integer (sMin to sMax)

Description: Sending this command without an argument will return the current value of PA. Sending this command with a valid argument between sMin and sMax will result in an immediate move to the requested position.

The PA command directly writes the position register of the actuator, bypassing all trajectory generation. This leads to a full acceleration, full speed move to the target position. Users will typically use this command for maximum dynamic performance or in conjunction with their own trajectory generator.

PC – Set Target Position to Current Position

Argument Type: None

Description: Set the current position as the Servo Cylinder's setpoint, causing the actuator to hold in place.

PO – Set Target Position Offset

Argument Type: Integer

Description: Move a relative distance from the Servo Cylinder's current position. This command will not execute if the requested offset move will send the actuator beyond sMin or sMax.

The PO command directly writes the position register of the actuator, bypassing all trajectory generation. This leads to a full acceleration, full speed move to the target position. Users will typically use this command for maximum dynamic performance or in conjunction with their own trajectory generator.

SP – Set Maximum Speed for Trajectory Moves

Argument Type: Integer (0 to 50000000)

Definition: Set the max speed value to be used in the Servo Cylinder's built in trajectory generator. Sending this command without an argument will return the current value of SP.

$$l = \text{screw lead}$$

$$\text{maxSpeed} \left(\frac{\text{in}}{\text{s}} \right) = \frac{SP * 1,250 * l}{2^{16} * 1,024}$$

W1 – Set Status Register Bit Mask for I/O pin1

Argument Type: unsigned long

Pre-Requisite: Digital Input/Output Pin 1 is set to output status bit (See **D1**)

Description: The value of DIO1 will be 0V when the result of a bitwise AND between the status register and the bit mask is equal to zero, and +5V when the result of the operation is a non-zero number.

Example uses include limit switch outputs, over-voltage indication, active trajectory indication, etc. See

Appendix C: Status Register for detailed information regarding the status word. Sending this command without an argument will return the current value of W1.

W2 – Set Status Register Bit Mask for I/O pin2

Argument Type: unsigned long

Pre-Requisite: Digital Input/Output Pin 2 is set to output status bit (See **D2**)

Description: The value of DIO2 will be 0V when the result of a bitwise AND between the status register and the bit mask is equal to zero, and +5V when the result of the operation is a non-zero number.

Example uses include limit switch outputs, over-voltage indication, active trajectory indication, etc. See **Appendix C: Status Register** for detailed information regarding the status word. Sending this command without an argument will return the current value of W2.

System Commands

ZC – Run Calibration Routine

Argument Type: Integer (“ZC321”)

Description: This command should only be sent after consulting with Ultra Motion engineers. The Servo Cylinder must be free to move and be at a position at least one revolution from either hard stop. The CONFIG.TXT file should be saved to a local location. After calibration, the HARDWARE.TXT file is written and the Servo Cylinder is reset, at this time CONFIG.TXT should be reloaded on to the Servo Cylinder.

ZD– Read Bridge Driver DIAG Register

Argument Type: None

Description: Read the detailed bridge driver diagnostic information from the Servo Cylinder.

ZF – Clear Latching Fault Flags in Status Register

Argument Type: None

Description: Clear all latching fault flags in the status register.

ZR – Reset Actuator

Argument Type: Integer (“ZR321”)

Description: Power cycle the actuator. This must be done to enact changes to the configuration file.

ZU – Jump to RS-232 Firmware Updater

Argument Type: Integer (“ZU321”)

Description: Suspend normal operation of the Servo Cylinder and launch the on-board bootloader to update the firmware. The actuator should be removed/disconnected from the machine to prevent potential damages and must be disconnected from USB.

Trajectory Commands

Trajectory commands are used to smoothly move the actuator to a desired position using the user configurable max speed and acceleration.

T1 – Trajectory Move to User Defined Pos1

Argument Type: None

Description: Move to Pos1 with user defined speed and acceleration (Pos1 defined in CONFIG.TXT).

T2 – Trajectory Move to User Defined Pos2

Argument Type: None

Description: Move to Pos2 with user defined speed and acceleration (Pos2 defined in CONFIG.TXT).

T3 – Trajectory Move to User Defined Pos3

Argument Type: None

Description: Move to Pos3 with user defined speed and acceleration (Pos3 defined in CONFIG.TXT).

T4 – Trajectory Move to User Defined Pos4

Argument Type: None

Description: Move to Pos4 with user defined speed and acceleration (Pos4 defined in CONFIG.TXT).

TA – Trajectory Move to Absolute Position

Argument Type: Integer (Smin to Smax)

Description: Sending this command with a valid argument between sMin and sMax will result in a trajectory move to the requested position.

Example: Move to absolute position 25,000 with acceleration value 2,000 and speed 200,000

AC2000

SP200000

TA25000

TE – Trajectory Move to Extended Position (sMax)

Argument Type: None

Description: Move to sMax with user defined speed and acceleration.

TK – Interrupt the current trajectory

Argument Type: None

Description: Halt the current trajectory before completion.

TM – Trajectory Move to Midpoint ($sMin + [sMax - sMin]/2$)

Argument Type: None

Description: Move to midpoint with user defined speed and acceleration.

TO – Trajectory Move to Offset (incremental trajectory move)

Argument Type: Integer

Description: Move a relative distance from the Servo Cylinder's current position using the trajectory generator. This command will not execute if the requested offset move will send the actuator beyond $sMin$ or $sMax$.

TR – Trajectory Move to Retracted Position ($sMin$)

Argument Type: None

Description: Move to $sMin$ with user defined speed and acceleration.

Serial Data Streaming

X1 – Set Data Stream Column 1

X2 – Set Data Stream Column 2

X3 – Set Data Stream Column 3

Argument Type: Integer (0 to 23)

Description: Define what variable each of the three columns are reporting

- 0 = Column Off
- 1 = Position Feedback
- 2 = Position Setpoint
- 3 = Motor Current Feedback
- 4 = Motor Current Setpoint
- 5 = Motor Phase 1 Current
- 6 = Motor Phase 2 Current
- 7 = Motor Phase 3 Current
- 8 = Velocity at 1.25 kHz
- 9 = Velocity at 156.25 Hz
- 10 = Encoder Interval Timer
- 11 = Supply Voltage
- 12 = Supply Voltage Rate of Change
- 13 = Status Register
- 14 = 1-2 ms Pulse Input
- 15 = Filtered 1-2 ms Pulse Input
- 16 = Analog Voltage Input
- 17 = Filtered Analog Voltage Input
- 18 = Analog Current Input
- 19 = Filtered Analog Current Input
- 20 = Temperature Sensor Input
- 21 = Position Error
- 22 = Quadrature Input Errors
- 23 = 32-bit Counter with 100 μ s Increments

Reference **Appendix D: Streaming Variables** for more detailed information.

XC – Clear Timer

Argument Type: None

Description: Reset the 32 bit timer used for synchronizing data to a time interval.

XG – Turn Data Streaming On

Argument Type: None

Description: Initiate the data streaming.

XR – Display Data Stream Setup

Argument Type: None

Description: Display the current stream setup including what variables are being streamed, streaming frequency.

XS – Turn Data Streaming Off

Argument Type: None

Description: Stop the data streaming.

XT – Set Data Stream Time Interval

Argument Type: Integer (1 to 65535)

Description: Define the rate at which data will be sent to the serial prompt. The minimum frequency is 3.576 Hz (XT = 65535); the maximum frequency depends on the Baud rate.

$$\text{Stream Rate (Hz)} = \frac{234375}{XT}$$

$$XT = 2344 \approx 100 \text{ Hz}$$

$$XT = 23438 \approx 10 \text{ Hz}$$

Appendix C: Status Register

The 32-bit status register is defined as follows:

- Bit 0 = State of optically isolated input 1
- Bit 1 = State of optically isolated input 2
- Bit 2 = Temperature sensor > ovTemp
- Bit 3 = Excessive position error detected, MOTOR OFF
- Bit 4 = Over-voltage on Vbus, dynamic brake enabled
- Bit 5 = Motor Control Enabled
- Bit 6 = Trajectory Active
- Bit 7 = Direction (extend)
- Bit 8 = Position < spMin
- Bit 9 = Position > spMax
- Bit 10 = Command input below min value, capped at min
- Bit 11 = Command input above max value, capped at max
- Bit 12 = Position Error < atTarg
- Bit 13 = Position error > ovErrP
- Bit 14 = Speed > ovSpeed
- Bit 15 = Torque > ovTorq
- Bit 16 = Position > posGrtr
- Bit 17 = Position < posLess
- Bit 18 = RS-232 Connected
- Bit 19 = USB Connected
- Bit 20 = Retracted physical stop has been hit
- Bit 21 = Extended physical stop has been hit
- Bit 22 = Supply voltage has been too low
- Bit 23 = Supply voltage has been too high
- Bit 24 = Bridge drove fault condition was reported
- Bit 25 = Over-current detected on motor phase
- Bit 26 = 1 to 2 ms pulse input error
- Bit 27 = A/B Quadrature input error
- Bit 28 = File system has been modified
- Bit 29 = HARDWARE.TXT file contains errors
- Bit 30 = CONFIG.TXT file contains some errors
- Bit 31 Critical error(s) in configuration file, MOTOR OFF