Simple Step® Product Manual

Version: 5.0.8

Simple Step®

Motion Control Made Simple!®

Simple Step LLC

12 West Owassa Turnpike
Newton, New Jersey, 07860
Phone: (973) 948-2938
Fax: (973) 948-0182

http://www.simplestep.com
email: support@simplestep.com

This page intentionally left blank

TABLE OF CONTENTS

PREFACE

This Chapter describes the organization of the manual and contains a glossary of terms referenced in the manual.

Copyright and Trademarks

All brand and product names appearing in this manual are trademarks of their respective holders.

Simple Step and Motion Control made Simple! are registered trademarks of Simple Step L.L.C.

SSWin is a trademark of Simple Step L.L.C

References

1. Allegro MicroSystems, Inc. 115 Northeast Cutoff, Box 15036 115 Northeast Cutoff, Box 15036 Worcester, Massachusetts 01615-0036 (508) 853-5000, http://www.allegromicro.com/

2. U.S. Digital Corporation, 3800 NE 68th Street, Suite A3, Vancouver, WA 98661-1353 USA, Phone: (360) 696-246, Sales: (800) 736-0194, Fax: (360) 696-2469, http://www.usdigital.com

Contact Information

Office Hours

Monday - Thursday: 9:30am to 5:00pm (EST)

Friday: CLOSED

Telephone/Fax

Phone: 1-973-948-2938

Fax: +01+973+948+0182

Misc. Information

Established: 1997

D&B: 92-700-8029

Sic Code: 3625

Harmonized Code: 8537.10.9060

Postal address

12 West Owassa Turnpike

Newton, New Jersey 07860

Website

http://www.simplestep.com

Electronic mail

Support: support@simplestep.com

Sales: sales@simplestep.com

Organization of this Manual

Chapter 1	Describes Simple Step LLC and the Simple Step Product Line.
Chapter 2	Describes the theory of stepper motors and stepper motor controller boards.
Chapter 3	Contains step-by-step procedures for connecting and programming Simple Step controller boards.
Chapter 4	Provides an overview of controller board commands. Describes the power setting command.
Chapter 5	Describes standard commands that apply to all boards.
Chapter 6	Describes the real-time operating system and related commands.
Chapter 7	Contains standard commands that apply to all quadrature encoder boards.
Chapter 8	Describes commands that set transmission options for communication between the board and the host.
Chapter 9	Describes how to program the IEEPROM.
Chapter 10	Additional Commands for all ARM boards
Chapter 11	Describes commands for the Onboard ADC and SSADC-1x boards.
Chapter 12	Describes commands for the on board PWM.
Chapter 13	Provides troubleshooting tips.
Appendix A	Lists recommended power supplies and connectors.
Appendix B	Provides board pinout diagrams.
Appendix C	Provides board connector location diagrams.
Appendix D	Provides mechanical drawings with board dimensions and board mounting hole locations.
Appendix E	Provides instructions to create a bulkhead cable harness assembly.
Appendix F	Describes the command format and lists prefix characters and address settings.
Appendix G	Describes the board's response to a request for board capability information (the 'v' command).
Appendix H	Provides a summary of all commands.

Glossary

B	Beginning velocity.
CPR (N)	The number of cycles per revolution.
CR	Carriage return (0xD) or Enter key.
Cycle error	The difference between the actual shaft position and the position indicated by the encoder cycle count. An indication of cycle uniformity. The difference between an observed shaft angle which gives rise to one electrical cycle, and the nominal angular increment of 1/N of a revolution.
DAC	Digital to analog converter.
E	End velocity.
Index (CH I.)	The index output goes high once per revolution, coincident with the low states of channels A and B, nominally 1/4 of one cycle (90°e).
One Cycle (C)	360 electrical degrees (°e). Each cycle can be decoded into 1 or 4 codes, referred to as X1 or X4 resolution multiplication.
One Electrical Degree (°e)	1/360 of one cycle.
One Shaft Rotation	360 mechanical degrees, N cycles.
Position error	*See cycle error.*
Quadrature (Z)	The phase lag or lead between channels A and B in electrical degrees, nominally 90°e.
RTOS	Real time operating system. Developed for the University of Arizona to allow the Simple Step Controller Board to communicate and run commands while the motors are moving.
S	Slope.
SPS	Steps per second.
Symmetry	A measure of the relationship between (X) and (Y) in electrical degrees, nominally 180°e.
VPFD	Power-fail deselect voltage.

List of Figures

List of Tables

This page intentionally left blank.

CHAPTER 1: INTRODUCTION

This chapter provides an overview of Simple Step LLC and its product line.

Background

In 1997, Simple Step LLC introduced a revolutionary new stepper motion controller product line to the motion control industry. The product line is named Simple Step®.

Simple Step products have quietly been moving into many new and old motion control applications. Starting with a simple single-axis controller board named the SSCB, the Simple Step product line has grown to large, multi-axis systems.


NOTE

Early versions of Simple Step motor control boards were equipped with 8-bit X2 processors. Later versions (starting in 2003) are equipped with 16-bit Philips XA processors (equivalent to an X3 processor). Finally, a new product line (beginning with the SSNEMA7 motor control board) utilizes NXP 32-bit ARM processors. Refer to pages 28 and 117 for additional information.

The name Simple Step says it all. Unlike other systems that require purchase of separate controllers (translators) and motor drivers, Simple Step provides all of the necessary electronics on a single printed circuit board that is usually smaller than most controller boards alone. Motion commands are delivered to Simple Step products using unique RS232-compatible serial communications, allowing multiple Simple Step products to be used on either a single serial channel or on an industry-standard multi-drop RS422/485.

But don't let the Simple Step name fool you, since the performance is far from simple! The product line includes the following features:

- Proprietary motion algorithms yield smooth and efficient motor motion.
- All products allow both full-step and half-step motion.
- Home and limit inputs on each axis allow controlled motion with built-in safety limits.
- Many motion characteristics are programmable and allow more experienced users to develop custom motion profiles.
- Programming is carried out using simple ASCII commands.

The Simple Step Design

We designed the Simple Step Product line to create a simple motion control system that has few external components, is easily controlled, and allows multiple boards to be connected without the need to include a computer or programmable logic controller in the system.

We created the Simple Step motion control product line because customers do not need a complicated motion control system when their only requirement is to move a motor from one point to another. As a result, Simple Step products have been used to control a wide range of devices, ranging from simple automated lathes to highly sophisticated systems such as deep-sea robots.

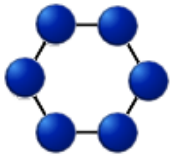
A fundamental difference in the design of our product line is that we use **distributed intelligence** instead of **centralized intelligence**. This means that every axis within the

network has its own microprocessor and that all of the axes in the system receive (or hear) a serial stream at exactly the same time. Since all of the axes are simultaneously parsing command strings from the host or user (as is done in high-performance mainframe computers), response times are fast even though the system is being controlled through a serial port.

A second fundamental difference in the design of our product line is that we are providing a **true network system** using RS232 connections. In contrast, most other serial designs employ either a **single-ended** control (one device per serial line) or an RS232 **ring network**.

NOTE

In a ring network, the transmitter line from the host is only connected to the receiver of the first motion control board. The transmitter line from that board is then connected to the receiver of the next motion control board, and the transmitter line from the last motion control board is connected to the receiver of the host. When communicating using a ring network, the host sends a command to the first board and that board forwards the command to the next board if the message is not for that board.



A Ring Network

Our use of a true network rather than a ring network avoids two problems inherent to a ring network:

- The communication lag time in a true network is much smaller than the communication lag time in a ring network. For example, it will only take 8 milliseconds for the host to receive a four-character response from any of the boards in a true network of up to 32 boards at 9600 baud.
- If one of the boards in the ring network fails, the entire network ceases to function.

The Simple Step Product Line

Overview

The Simple Step product line features a wide variety of stepper motor controller boards, quadrature encoder boards, and analog-to-digital conversion boards.

Stepper motor controller boards range from the SSMicroLC single-axis motion control board, which delivers up to 1.5 amps per motor winding, to multi-axis microstepping systems such as the SSXYZMicroMC, which delivers up to 3.125 amps per motor winding.

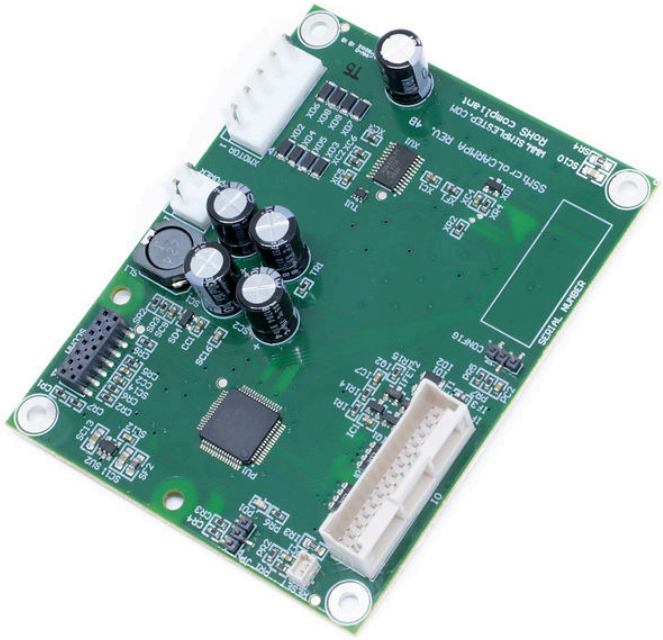


Figure 1 SSMicroLC Single-Axis Motion Control Board

Features

All Simple Step controller boards come with the following features:

- Operate on a single voltage (can be as low as 12.0VDC to as much as 50.0VDC).
- Translator/indexer and driver are included on a single board.
- Bipolar stepper motion control for all boards.
- Serial network system with software address switches (either Simple Step RS232 network or RS422/485), except for SSNEMA17 (which has only RS422/485).
- Each axis has its own 72MHz 32-bit ARM microprocessor.
- Each axis has separately adjustable current limiting using software commands.
- Each axis has dedicated home (infrared with on-board current limiting to 20ma@5VDC) and limit (microswitch) inputs.

- Each axis has software controlled I/O for general use (3 TTL I/O lines and 3 0-3.3VDC I/O lines).
- Each axis has 2 software-controlled 0.5 amp, diode clamped MOSFET output (only 1 available on SSNEMA17 board).
- All boards run in a simple ASCII command mode.
- All boards have a power LED which can display error codes.
- All boards come with a 5VDC and 3.3VDC switchers that are 85% efficient.
- Every purchase comes with FREE Simple Step for Windows software (which can be downloaded from the website).
- Every purchase comes with the FREE Simple Step ActiveX control (which can be downloaded from the website).
- The baud rate can be changed from 9.6K to 460.8K on all boards after power up is complete.
- All stepper motion controller boards have a software selection from full-step mode to 1/32 step mode.
- All microstepping motion controller boards can run to 1/32 of a step.
- All motion controller boards have a software programmable Running and Idle power modes.
- All boards have a speed range from 1 sps (step per second) to 30,000 sps in a linear acceleration/deceleration format. (All units now have the prescaler option installed, which allows sub-sps speeds).
- All boards include a new and unique linear motion algorithm.
- 2000-character IEEPROM on ALL units.
- All units now have a joystick input via the TTL I/O lines.

Summary of Product Features

Product Features	SSNEMA17	SS MicroLC	SS MicroMC	SSXY MicroLC	SSXY MicroMC	SSXYZ MicroLC	SSXYZ MicroMC
72MHz, 32 bit ARM Processor Features	X	X	X	X	X	X	X
1 axis	X	X	X				
2 axis				X	X		
3 axis						X	X
1.5 amps per phase	X	X		X		X	
3.125 amps per phase			X		X		X
6.25 amps per phase							
Microstepping to 1/32 step	X	X	X	X	X	X	X
Software Setting for Current control	X	X	X	X	X	X	X
Software Programmable Decay control	X	X	X	X	X	X	X
Software Idle Current Control (0-100% of FULL)	X	X	X	X	X	X	X
MOSFET Control per axis	1	2	2	2	2	2	2
User TTL Inputs per axis (0-5 VDC)	2	3	3	3	3	3	3
Option Connector (Pin Count)	0	3	3	3	3	3	3
Quadrature Encoder Interface per axis	STD	STD	STD	STD	STD	STD	STD
2000 Character IEEPROM per axis	STD	STD	STD	STD	STD	STD	STD
Optional 32-bit Motor Movement per axis	STD	STD	STD	STD	STD	STD	STD
Optional RTOS per axis	STD	STD	STD	STD	STD	STD	STD
12 bit ADC Input per axis	STD	STD	STD	STD	STD	STD	STD
PWM output per axis	STD	STD	STD	STD	STD	STD	STD

This page intentionally left blank

CHAPTER 2: THEORY

This chapter describes the theory of stepper motor operation and control with a Simple Step Controller Board.

Stepper Motor Basics

A stepper motor is a type of permanent magnet motor where the motor shaft moves a predefined fraction of a revolution (measured in degrees) in response to an electronic signal. For example, the shaft of a 1.8-degree stepper motor rotates 1.8 degrees per signal (or **step**), requiring 200 steps per revolution. The steps per revolution (**spr**, or **resolution** of the motor) for a particular stepper motor is determined by the number of coil winding pairs (or **phases**) and the rotor design for that motor.

The resolution of a stepper motor can be increased electronically by driving the motor in fractional steps (changing the **mode** of operation, or **microstepping**). For example, operating a 1.8-degree stepper motor in 1/2 step mode causes the motor to rotate 0.9 degrees (rather than 1.8 degrees) per step.


NOTE

Additional modes can be used to further increase the resolution of a stepper motor. For example, operating a 1.8-degree stepper motor in 1/16 step mode causes the motor to rotate only 0.1125 degrees rather than 1.8 degrees per step.

Microstepping Simple Step Controller Boards (all Bipolar stepper controllers) support up to six modes (full-step, 1/2 step, 1/4 step, 1/8 step, 1/16, and 1/32 step).

Stepper Motor Motion Control

Motion of a stepper motor shaft during a travel interval is controlled by a **motor translator**. The translator provides low-power electronic step signals to the **motor driver**, which provides sufficient electrical power to rotate the motor shaft. One step signal from the translator is required for each step taken by the motor.


NOTE

All Simple Step motor controller boards include both a motor translator and a motor driver.

The motor drivers supply current to the stepper motor windings, creating a magnetic field and providing the force to move the motor shaft and its connected load. The stepper motor driver circuit is a basic RL (resistor plus inductance) circuit, and acceleration of the motor shaft during the travel interval increases in proportion to the applied voltage.

Current supplied by the motor drivers is controlled so that sufficient force is provided to accelerate the stepper motor shaft at the required rate while preventing the motor from overheating.

Control of Voltage and Current to the Motor

Manufacturers of stepper motors base the maximum voltage and current ratings for a motor upon the resistance of the windings and upon heat dissipation characteristics.

Simple Step Controller Boards utilize **current switching** technology to control motor current, and the voltage that is applied to a stepper motor can be increased substantially over the manufacturers' specifications without damaging the motor.


IMPORTANT

Motor manufactures specify motor voltage. This is not the voltage to run the motor at. It is recommended that all motors be run from 15 to 50 volts (depending on controller and a higher voltage is recommended) when using Simple Step Controller Boards. ***Motor current limits must be correctly set before applying power to a stepper motor.*** *Otherwise, motor damage may occur.*


IMPORTANT

The motor current limits ***must*** *be correctly set before applying power to a stepper motor. Otherwise, motor damage may occur.*


IMPORTANT

The motor should ***NEVER*** *be disconnected from the board with power applied. This will void the warrantee.*


IMPORTANT

When the board is powered down, do not move the motor by hand since this could generate BEMF into the board which could reach voltages well over 100VDC back into the board.


NOTE

Refer to the appropriate Setting Maximum Current Limits section in Chapter 3 for instructions on setting motor current limits for the Simple Step Controller Board that you are using.

Allowable ranges for motor current settings when using various Simple Step controller boards are summarized below:

Board	Adjustment of motor current per phase	Allowable Current Adjustment Range (amps/phase)	Current Limit Adjustment	Idle Power Setting (amps/phase)	Current Decay Adjustment
SSMicroLC	Set by Software	0.100 to 1.50	Software page 59	0 to Set Point page 59	0-255 page 59
SSMicroMC (SSCB Replacement)	Set by Software	0.100 to 3.125	Software page 59	0 to Set Point page 59	0-255 page 59
SSXYMicroLC	Set by Software	0.100 to 1.50	Software page 59	0 to Set Point page 59	0-255 page 59
SSXYMicroMC (SSXYQE Replacement)	Set by Software	0.100 to 3.125	Software page 59	0 to Set Point page 59	0-255 page 59
SSXYZMicroLC	Set by Software	0.100 to 1.50	Software page 59	0 to Set Point page 59	0-255 page 59
SSXYZMicroMC (SSXYZ Replacement)	Set by Software	0.100 to 3.125	Software page 59	0 to Set Point page 59	0-255 page 59

SSNEMA17	Set by Software	0.100 to 1.50	Software page 59	0 to Set Point page 59	0-255 page 59

Cooling of Motor Drivers

A cooling fan should be used if any of the following are applicable:

- the power setting (parameter #1 of the 'P' command) is greater than **100**
- the idle mode (parameter #2 of the 'P' command) is greater than or equal to **50**
- the motor is run in a continuous motor movement mode

If a motor driver overheats, miss-stepping will occur and driver damage may occur. If a cooling fan is needed, a 24 to 37 CFM (cubic feet per minute) fan is recommended to be positioned near the board, at the side where the power supply is attached to the board.

NOTE

All Simple Step Controller Boards include thermal shutdown protection for the motor drivers.

In most cases, the motor driver will shut down if the junction temperature of the driver reaches 165 ºC and remain shut down until the temperature drops below the shutdown value. When the temperature drops by approximately 20 ºC, the driver will automatically reactivate.

NOTE

Several tests were performed on an SSMicro board to evaluate the thermal behavior. All tests were run at an ambient temperature of 25 ºC, an 'S' of 2, a 'B' of 100, an 'E' of 400, and half-step mode using the 'C+' command over a four hour period.

The test results indicate that fast-decay mode [DAC setting of zero (0)] tends to run the stepper motor drivers at a cooler temperature:

1) *The power was set to "P100,0,0" and the board was run without a fan. Measured driver temperature was 66.7 ºC.*
2) *The power was set to "P100,0,255" and the board was run without a fan. Measured driver temperature was 71.2 ºC, a 4.5 ºC increase in temperature.*
3) *The maximum current level was increased to "P150,0,0" and the board was run without a fan. Measured driver temperature increased over a four-hour period to 83.3 ºC. The driver shut down when the junction temperature reached 165 ºC and did not restart until the junction temperature dropped to 150 ºC. A 3-inch, 30 CFM AC fan was added to the side where the power supply is attached to the board, and the temperature decreased to 56.8 ºC.*
4) *The maximum current level was increased to "P255,0,0" and the board was run without the fan. The driver temperature increased to 160 ºC within 15 minutes. After the fan was activated, the driver temperature decreased to 72.1 ºC over a four hour period. When the power was changed to "P255,0,255" and the fan was activated, the driver temperature increased to 74.9 ºC.*

Short Circuit Protection

All Simple Step controller boards have short circuit protection on the outputs to the motor. As a result, if the motor connections are shorted to each other, the Simple Step controller board will not be destroyed.

IMPORTANT

This does NOT mean that you can connect any of the motor lines to either ground or the power supply directly. This will destroy the motor drivers and most likely other devices on the Simple Step Controller Board.

Microstepping Motor Current Control

Pulse width modulation (PWM) technology is used to control current in the Simple Step microstepping board motor drivers. PWM averages the on and off current pulses that are applied to the motor windings and yields a steady current flow.

When PWM is utilized, current applied to the motor windings is switched on and off at a rate that causes inductance of the motor to filter the applied waveform and average the motor current.

During **PWM off-times** (the period when current is switched off), the manner in which current is allowed to decay can be specified to optimize motor performance, reduce audible motor noise, increase step accuracy, and reduce power dissipation:

- **Slow current decay** (where motor windings are shorted during PWM off-time) yields less current ripple and should always be selected when possible, to minimize core losses and switching noise.
- **Fast current decay** (where motor voltage is reversed during PWM off-time) may be required when rapid changes of motor current are necessary in half-stepping and microstepping applications.
- **Mixed current decay** (where the decay starts in fast mode then changes to slow mode after a user-defined delay) minimizes ripple without compromising current control.

The **VPFD** parameter (Voltage applied to Percent Fast Decay) specifies the time that is spent in fast current decay. **ITRIP** is the desired load current.

Slow Current Decay Mode

When VPFD is **3.5V**, the device is in slow current-decay mode (the source drivers are disabled when the load current reaches ITRIP). During the fixed off time, the load inductance causes the current to recirculate through the motor winding, sink driver, ground clamp diode, and sense resistor (Figure 2).

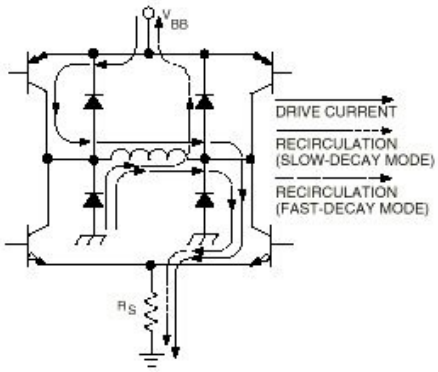


Figure 2 Load Current Paths

Slow-decay mode produces low ripple current for a given fixed off time (Figure 3). Low ripple current is desirable because the average current in the motor winding is more nearly equal to the desired reference value, resulting in increased motor performance in microstepping applications. For a given level of ripple current, slow-decay affords the lowest PWM frequency, which reduces heating in the motor and driver IC due to a corresponding decrease in hysteretic core losses and switching losses respectively. Slow-decay also has the advantage in that the PWM load current regulation can follow a more rapidly increasing reference before the PWM frequency drops into the audible range.

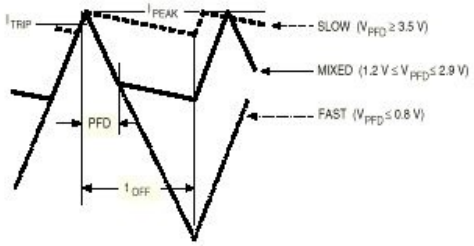


Figure 3 Current-Decay Waveforms

Mixed Current Decay Mode

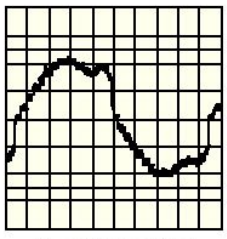
If VPFD is between **1.2V** to **2.9V**, the device will be in a mixed current decay mode, allowing the user to achieve good current regulation with a minimum amount of ripple current and motor/driver losses by selecting the minimum percentage of fast decay required for the application.

As in fast current decay mode, mixed decay starts with the sink and source drivers disabled after the load current reaches ITRIP. When the voltage at the RC terminal decays to a value below VPFD, the sink drivers are re-enabled, placing the device in slow current decay mode for the remainder of the fixed off time (Figure 3). The percentage of fast-decay (PFD) is user-determined, either by VPFD or with two external resistors.

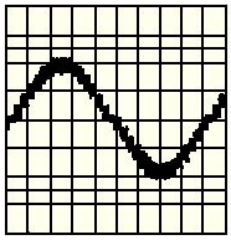
Fast Current Decay Mode

When VPFD is from **0.0V** to **1.8V**, the device is in fast current decay mode (both the sink and source drivers are disabled when the load current reaches ITRIP).

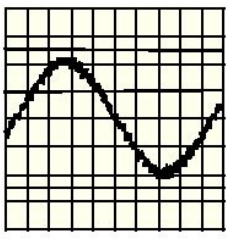
During the fixed off time, the load inductance causes the current to flow from ground to the load supply via the motor winding, ground-clamp and flyback diodes (Figure 2). Because the full motor supply voltage is across the load during fast decay recirculation, the rate of load current decay is rapid, producing a high ripple current for a given fixed off time (Figure 3). This rapid rate of decay allows good current regulation to be maintained at the cost of decreased average current accuracy or increased driver and motor losses.



A – Slow Decay Mode



B- Fast Decay Mode

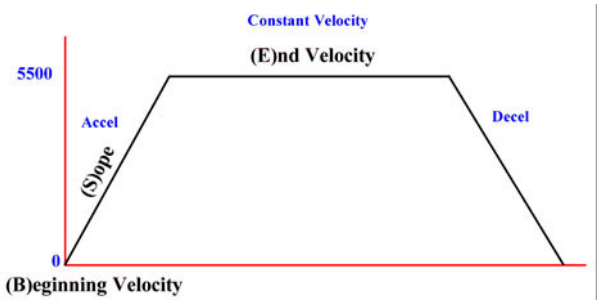


C- Mixed Decay Mode

Figure 4 Current Decay Modes

Acceleration and Deceleration Algorithms

During the travel interval, inertia of the stepper motor shaft, rotor, and load that is linked to the stepper motor shaft prevents the motor shaft from instantaneously reaching its target speed. In a similar manner, momentum of the shaft, rotor, and load prevents the motor shaft from instantaneously stopping at the end of travel. The velocity profile of the motor shaft during a typical travel interval is shown below:



where **(B)** is the beginning velocity (the startup speed)
(S) is the slope (increase or decrease in velocity per step)
(E) is the end velocity (maximum allowable speed)

Simple Step Controller Boards utilize unique acceleration and deceleration algorithms during the travel interval. These algorithms allow step rates in excess of 7,000 steps per second and stop the motor slowly to yield better position accuracy.

Beginning velocity **(B)** and end velocity **(E)** are first used to calculate the total acceleration (**or** deceleration) **(AD)** required during the travel interval:

AD = E – B

The acceleration or deceleration **(AD)** is then divided by acceleration/deceleration rate per step **(S)** to obtain the number of acceleration **or** deceleration steps **(**slope steps, or **SS)** needed during the travel interval:

SS = AD/S

The number of acceleration or deceleration steps **(SS)** needed during the travel interval is then multiplied by **2** to yield the total number of acceleration **and** deceleration steps **(TADS)** needed during the travel interval:

TADS = 2SS

The total number of acceleration **and** deceleration steps **(TADS)** are then subtracted from the total number of steps that are required for the travel interval **(**total steps to move, or **TSM)**. The result is stored as the constant velocity parameter for the travel interval.

Constant Velocity Parameter = TSM - TADS

NOTE

*If the total number of required steps for the travel interval **(TSM)** is less than the steps during acceleration and deceleration **(TADS)**, the end velocity **(E)** will never be achieved. In this case, the required value for **(S)** can be calculated:*

S = 2(E – B)/TSM

*If **(S) = 0**, then no acceleration or deceleration will occur.*

Simple Step Controller Boards are programmed using the **(B)**, **(S)**, and **(E)** values described above plus **(M)**, which is the desired position for the motor shaft at the end of its travel interval. The processor calculates the amount of travel based on the difference between the current motor position and **(M)**. If **(M)** is greater than the current position, the motor will be driven in the direction of the **limit** position. If **(M)** is less than the current position, the motor will be driven in the direction of the **home** position.


NOTE

*The **home** position is the initial position of the motor shaft. The **limit** position is the limit of motor shaft travel. Sensors may be utilized at both positions.*


NOTE

Refer to

Setting Beginning Velocity, End Velocity, and Slope on page 46.

Maximum Travel Distance

Maximum distance of a stepper motor travel interval is determined by the controller board. Most Simple Step Controller Boards allow up to +/-2,147,483,646 steps in one travel interval.

Simple Step Controller Boards are programmed to travel using either the **M** (Move Motor to an absolute position) command or the **R** (Move Motor to a relative position) command. Positioning from **0** to **+/-2147483646** is allowed for either of these commands. Refer to page 63 for the procedure to enter these commands.


NOTE

The RTOS position register is present in all boards, regardless of whether the RTOS option was purchased for the board. RTOS position register is a signed 32-bit register long integer type.

Control of Stepper Motor Shaft Travel Direction

The direction that a stepper motor shaft turns is determined by the direction in which power is applied to the motor winding pairs (the **phases**). This can be controlled either by the manner in which the phases are connected to the controller board or by the direction in which power is applied at the controller board.

Stepper motors typically contain two phases, referred to as **A** and **B**. Each phase can have a connector at each end of the winding as shown below (a **bipolar** motor), or it can have an additional connector at the center of the winding (a **unipolar** motor). Connectors for each phase are labeled with the phase notation (for example, **A**, **$\overline{A}$**, **B**, and **$\overline{B}$**).

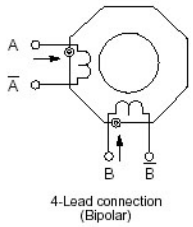


Figure 5 Sample Motor Connection

For all Simple Step controller boards:

- **A** is connected to **J5 pin 1**
- **$\overline{A}$** is connected to **J5 pin 2**
- **B** is connected to **J5 pin 3**
- **$\overline{B}$** is connected to **J5 pin 4**

To run the motor in the opposite direction for all commands, the motor can be connected in reverse or the direction of the **N** parameter can be changed on power-up.

NOTE

Refer to Connecting a Motor to the Board on page 39 for additional details.

Quadrature Encoder Interface

Use of a quadrature encoder allows direct feedback concerning the absolute direction and position of the part of the device that is being repositioned by the stepper motor. The encoder usually consists of a disk or strip with electronically detectable rulings (for example, scribed lines) at regular scaled intervals that create an electronic signal as each ruling passes under a sensor (such as a photocell).

Two detector channels are used. The detectors for the two channels are spaced so that when one ruling is directly under a detector (yielding a maximum signal), adjacent rulings are on either side of the second detector (yielding a maximum signal). This allows the position of the disk (or strip) and the direction of its rotation (or movement) to be instantaneously determined while in motion.

The diagram shown below represents the output from a two-channel quadrature encoder for a rotating shaft. Signals from channel B always precede the signals from channel A when rotation of the shaft is in a clockwise direction, and the reverse is true when rotation of the shaft is in a counterclockwise direction.

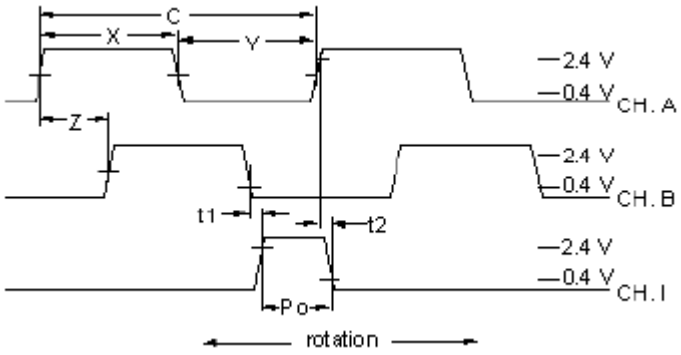


Figure 6 Timing Diagram

NOTE

The counting process for each channel is interrupt-driven so that the user can at any time request (even when the wheel is moving) the current quadrature reading. This allows for simultaneous counting and display of both channels.

Simple Step Product Versions and Their Features

Overview

Early versions of Simple Step boards (firmware versions 101 to 104) were equipped with 8-bit X2 processors. Latest versions of all boards (firmware versions 105 to 108, starting in 2003) are equipped with 16-bit Philips XA processors (equivalent to an X3 processor).

A new Simple Step product line (beginning with the SSNEMA7 motor control board) utilizes NXP 72MHz 32-bit ARM processors. Refer to page 117 for additional information.

Differences Between Product Versions

The primary difference between earlier Simple Step boards (firmware version 101 to 104) and new boards with XA processors is the manner in which motor motion is controlled. Motion control for firmware version 101 to 104 products with and without X2 processors was non-linear, with a top end velocity of approximately 14,000 steps per second (sps). Motion control in current products with firmware version 105 and above is linear over the range of 1 to 30,000 sps.

Calculations for boards with firmware version 105 and above are the same as for older boards, except that acceleration and deceleration is always linear. Motor movement for boards is now linear, and the 'E' and 'B' values run in steps per seconds (sps) with the maximum 'S'lope value expanded from 100 to 200.

NOTE

A software switch allows new Simple Step motor control boards to use the old motions (see Powering up in the Old Motion Control State on page 46). This allows backwards compatibility with older software systems that cannot be updated to the new motion system.

Current limits to the time base only allow a speed range from 1 sps to 30,000 sps in 1 sps increments. Firmware versions 105.xx and above have the prescaler option installed, allowing the user to prescale the 1 to 30K sps range from 1:1 to 255:1.

New Simple Step motor control boards also have a 2000-character IEEPROM.

NOTE

New Simple Step motor control boards also include a software switch for turning the Real Time Operating System (RTOS) on or off. This allows the user to use new Simple Step boards in systems with older non-RTOS software.

Summary of Command Changes

1. 'E', 'B', and 'S' limits depend upon the processor, version, and whether RTOS is ON or OFF (for firmware version 106 and above):

Command	Simple Step Motor Control Board Processor			
	v101 to v104	**v106 to v108, RTOS ON**	**v106 to v108, RTOS OFF**	**v109 and above**
E	1-46250	1-15000	1-30000	1-30000
B	0-46000	1-13000	1-13000	1-2000
S	1-100	0-200	0-200	0-200

Initialization values of 'E', 'B' and 'S' at power-up for all firmware versions 105 and above are as follows: E = 3000, B = 300, and S = 10.

2. The continuous command (‘C’) now allows for speed updates within 1 step time after the <CR> delimiter is received.

3. The software has a 32-bit real-time counter that is updated at all times. This allows the *request current motor position* command (‘m’) to be requested in signed long integer format. To *display the current RTOS position register* command (‘opd’) is issued, the full 32-bit signed number is returned.

4. Relative moves do not destroy the current position counter variable.

5. The old RTOS motor movement commands could be stacked 1 deep (RTOS only units). Firmware version 105.xx and above now has an 8 level stack and can switch from one command to the next usually within 1 step time.

6. A new option command allows the user to determine the direction of the last motor movement. The board responds to an “oD” command with either a ‘-‘ or a ‘+’. A ‘-‘ indicates that the last motor movement was towards home and a ‘+’ indicates that the last motor movement was towards the limit (away from home).

7. If the *continuous motion* command (‘C) is issued when software limits are active and the slope is zero (0), the board will return a Parameter out of Range Error (‘!’).

8. If the *continuous motion* command (‘C’) is issued and then an ‘E’ update is issued, the board will not perform the update if any of the following conditions exist:

   a. Slope is set to zero (0).

   b. E and B are the same.

   c. The motor is currently in the deceleration mode (software limits active and tripped).

New Motor Time Calculations (Firmware version 105 and above)

Motor movement for firmware version 105 and above is now linear, and the ‘E’ and ‘B’ values run in steps per seconds (sps) with the maximum ‘S’lope value expanded from 100 to 200. Calculations are the same as for older motion system, except that the acceleration/deceleration is always linear.

Desired time to move the total distance in seconds = TDS

Total amount of steps to move = TS

Programmed rate (E) = PR

TS = 1000 steps

TDS = 3 seconds

PR = TS / TDS

1000 / 3 = 333.333 or 333 sps = PR = E

The above calculation does not take into account the slope, and if the slope is a very long the time it takes to move the total distance will vary.

Old Motor Time Calculations (Firmware version 101 to 104)

Timer Resolution = 0.5425347 µsec per increment (Standard, and X2 Option = 0.18084491 µsec).

For a constant velocity with no acceleration or deceleration slope, 'B' and 'E' values should be set to the same number. The time values are as follows:

46250 is equal to 69.987 µsec (base) per step and a 1 is equal to 25.091039 msec.

46250 x 0.5425347 µsec = 25.092229875 msec

25.02125 msec + 69.987 µsec = 25.16221688 msec

Every increment of the rate value equals 0.5425347 µsec. The formula to calculate other rates is as follows:

Desired time to move the total distance (in seconds) = TDS

Total amount of steps to move = TS

Programmed rate = PR

TS = 1000 steps

TDS = 3 seconds

BASE = 0.000069987

TIMER = 0.0000005425347

PR = 46250 - ({[1 / (TS / TDS)] – BASE} / TIMER)

PR = 46250 - ({ [1 / (1000 / 3)] - 0.000069987} / 0.0000005425347)

= 46250 – [(0.003 - 0.000069987) / 0.0000005425347]

= 46250 – [(0.002930013) / 0.0000005425347]

= 46250 - 5400.5999616 = 40849.4000384 = 40849

The above calculation does not take into account the slope. If the slope is very long, the time it takes to move the total distance will vary. Since the fastest value is 46,250, which represents 69.987 µsec, every decrement of the 46250 increases the step time by 0.5425347 µsec. The slowest number that can be given as an input value is 1.

SPS Calculations and Conversion Formulas (Firmware version 101 to 104)

Steps per second (sps) = 1 / [(46379 - User Value) * 0.0000005425347]

User Value = 46379 – [(1 / sps) / 0.0000005425347]

If the special order prescaler option is installed, the calculations change as follows:

PR = 46250 – {[(1 / (TS / TDS)][- BASE) / (TIMER*(S+1)]}

where S = the prescale value (0 to 255)

CHAPTER 3: SETTING UP A SIMPLE STEP CONTROLLER BOARD

This chapter describes procedures for connecting a Simple Step Controller Board to a power supply, installing the SSWin application, connecting the board to the host (PC), and operating the motor from the SSWin software application.

Power Requirements

Simple Step controller boards operate from a single power supply. Power requirements for the various Simple Step products are summarized below:

Board Type	Minimum Current (amps)	Maximum Current* (amps)	J1 Input Minimum Voltage (Tolerance) (VDC)	J1 Input Maximum Voltage (Tolerance) (VDC)
SSNEMA17	0.100	3.25	15.0 (-1.0%)	50.0 (+1.0%)
SSMicroLC	0.100	4.20	15.0 (-1.0%)	50.0 (+1.0%)
SSMicroMC	0.100	6.25	12.0 (-1.0%)	50.0 (+1.0%)
SSXYMicroLC	0.100	8.60	15.0 (-1.0%)	50.0 (+1.0%)
SSXYMicroMC	0.100	10.60	12.0 (-1.0%)	50.0 (+1.0%)
SSXYZMicroLC	0.100	12.06	15.0 (-1.0%)	50.0 (+1.0%)
SSXYZMicroMC	0.100	18.75	12.0 (-1.0%)	50.0 (+1.0%)

* All motors and outputs turned on

Table 1: Power Requirements for Each Simple Step Product

A well-regulated power supply with an allowable voltage tolerance of ± 1% is required. If a power supply with large ripple is used, this will affect overall motor performance. Typical current draw for a board with no motor or output lines activated is approximately 52 mA per axis.


NOTE

Maximum power supply current depends on the motor used, the current per phase of the motor, the stepping mode currently activated, and the output lines that are switched on.

For example, if a motor uses one ampere per phase, then the power supply must be capable of delivering at least 2.165 amperes when the motor is turned on, is in FULL step mode, is not moving, and all outputs are turned ON. In half-step or higher modes, however, two phases are usually never on at the same time, so the current draw would be approximately one ampere.

*Motor parameters (**E**, **B** and **S** values) also impact power requirements. If **B** is 100, **E** is 10,000, and **S** is 10, there is a very fast slope and the major portion of the motor movement will be at the **E** value, and in this case the current draw may only be about 75% of whatever stepping mode the motor is in. If the movements are running most of the time at the **E** value (65% or more), then a 1.5 ampere power supply may be sufficient. Finally, current limiting affects power requirements for a motor that is running.* *General Purpose Inputs/Outputs*

All Simple Step Controller Boards create their own 5 Volt DC logic supply using an on-board switcher allowing the use of just one (1) supply source. The switcher is 88% efficient, minimizing heat dissipation.

Connecting a Simple Step Controller Board to a Power Supply

1. Obtain a CE-approved power supply that meets requirements outlined in the previous section.


NOTE

Refer to page 137 for a list of recommended power supplies and connectors.

2. Connect the power supply connector to the **J1** connector on the controller board.


NOTE

***J1** (or POWER) is used as the power input connector in all Simple Step products. **Pin 1** of **J1** (designated by an arrow or the number **1** next to the **J1** connector) is the plus (+) or positive input terminal and J1, and **Pin 2** of **J1** is the negative (-) or ground input terminal.*

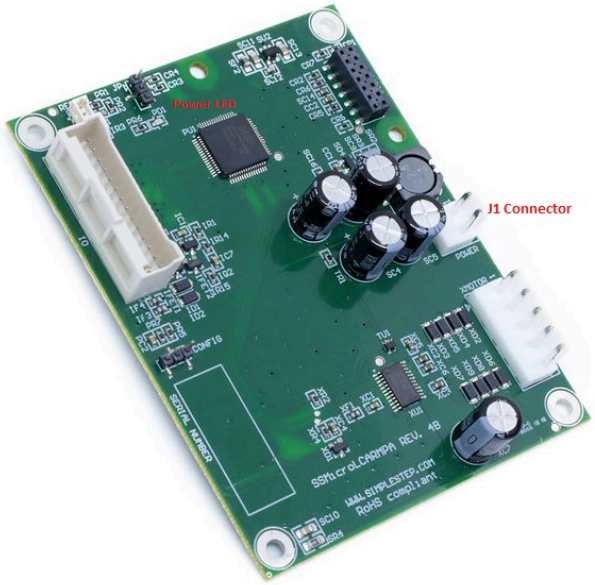


Figure 7 Location of the J1 Connector and Power LED (SSMicroLC Board Example)

3. Turn on the power supply to check operation. The Power LED on the board should illuminate.

NOTE

If the power LED on the board does not illuminate, turn off the power supply and check the connections.

4. Turn off the power supply before proceeding.

IMPORTANT

These are JST Sales polarized and locking style connectors. Please purchase the correct tool and pins needed for your application.

Installing the SSWin Application on the Host

1. Insert the Simple Step LLC CD-ROM into the hosts' CD-ROM drive.
2. If the installation does not start automatically:
   a. Click the **Start** button located on the bottom left side of the Windows task bar.
   b. Select **Run...**.
   c. Type in (or Browse) so that the text line reads "x:\setup.exe" where x is the CD-ROM drive letter.
   d. Click the **OK** button to start the setup procedure.
3. Follow the prompts presented by the installation wizard.
4. Once the installation is completed, remove the CD-ROM from the CD-ROM drive.

Connecting the Simple Step Controller Board to the Host

Communications between the host and the board are performed using either the (CBRS232) Simple Step RS232 Network with up to 16 Simple Step Boards or an (CBRS422-485) RS422/RS485 port that allows up to 32 Simple Step Controller Boards on one serial line.

1. Connect pin #1 of the J2 connector to digital ground.
2. Connect pin #2 of the J2 connector to receive input.
3. Connect pin #3 of the J2 connector to transmit output.

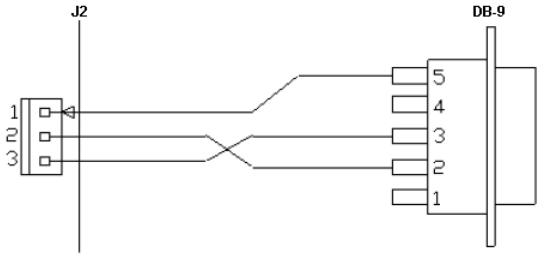


Figure 8 DB9 to J2 Communications Connection

DB-9	DB-25	Host Signal	CBSR232	Board Signal
5	7	Ground	1	Ground
2	3	Receive Input	2	Transmit Output
3	2	Transmit Output	3	Receive Input

Table 2 RS232 Network Connection

CBSR422 -485	Board Signal
1	Ground
2	Transmit +
3	Receive +
4	Transmit -
5	Receive -

Table 3 J2 Connection for RS422/485 Network

NOTE

CBRS422-485 boards now have a DIP switch that allows the choice of RS422 or RS485. The DIP switch also allows for RXD and TXD balance resistors to be enabled/disabled.

NOTE

Both the CBRS232 and CBRS422-485 now have cloned communication connectors. They are identical and allow for inter-board connections.

SSNEMA17 Board J1 Pin	SSNEMA17-4x Board Signal
1	MOTOR /B
2	MOTOR B
3	MOTOR /A
4	MOTOR A
5	MOTOR SHIELD
6	Power Ground
7	Power Ground
8	Power Ground
9	+15.0 to 50.0 VDC
10	+15.0 to 50.0 VDC
11	Transmit +
12	Receive +
13	Transmit -
14	Receive -

Table 4 SSNEMA17-4x J1 Connections for RS422/485 Network, Power and Motor

Setting the Simple Step Controller Board Network Address

A Simple Step Controller Board address are determined by a software command (see Page xx to read about setting the ARM configuration for the board address).

If there is more than one TYPE of board on the network, make sure that each board of the same type has a unique address.

For example: If you have one SSMicroMC, and three SSXYZMicroMC boards on the line, set the SSMicroMC, and one SSXYZMicroMC board to address zero (0). Set the address of one of the other SSXYZMicroMC board to one (1), and the address of the last SSXYZMicroMC board to two (2).

Starting SSWin

1. Select **Start** > **Programs** > **Simple Step for Windows** > **SSWin Program** from the Windows task bar. A dialog box appears prompting you to power up all boards before proceeding.

2. When all boards are powered up, click **OK**.

   The *Baud Rate Selection* dialog box is displayed.

3. Select the baud rate that was programmed at the factory when the board was ordered. If you are unsure of the baud rate, leave the default 57.6K setting and click **OK**.

4. SSWin scans for all Simple Step products on the serial network. Results are displayed on the right hand side of the main screen (e.g. three SSCB Units (now called SSMicroMC), addresses 0, 1, and 2 in the example below).

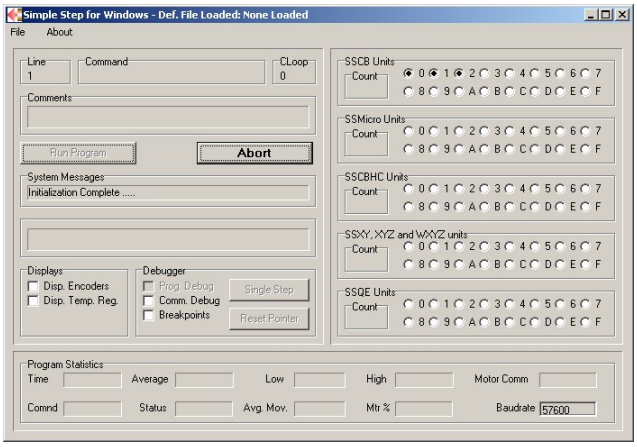


Figure 9 Simple Step for Windows Main Screen

Troubleshooting

If no boards were found (all counts are **0**), use the troubleshooting steps below to help determine the cause of the problem.

1. Wrong Communication Port is selected.

   a. Select **File** > **Configure System** from the SSWin menu bar. The *Configuration Dialog* box is displayed.

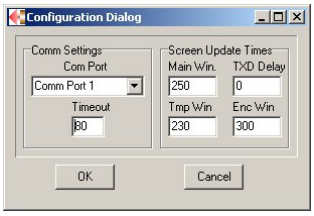


Figure 10 Configuration Dialog Box

b. Verify the port selection (e.g. Comm Port 1 in the example above).

NOTE

Most PCs use COMM1 but a few may use COMM 2. Neither port may be available on your PC if you are using a standard serial mouse and have a built in modem.

c. If the COMM port is incorrect, use the drop-down arrow to choose the correct port, click **OK** to save the new setting, and then restart the SSWin application.

2. Wrong baud rate was chosen.

a. Select **File** > **Terminal and Network Scanning** from the SSWin menu bar. The *Terminal and Network Scanning Dialog* box is displayed.

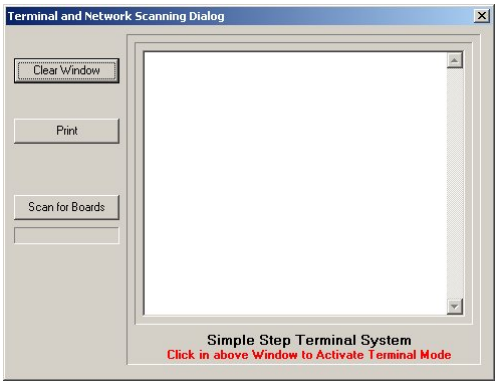


Figure 11 Terminal and Network Scanning Dialog Box

b. Click **Scan for Boards** and wait for the test to complete.

c. The actual baud rate is displayed. If no results are displayed proceed to step 3 below to check the wiring to the board.

3. Wiring is incorrect.

   a. Unplug the J2 connector from the Simple Step board and short pins #2 and #3 on the cable.

   b. Click inside the *Simple Step Terminal System* window (Figure 11) and type the letter 'A'.

   c. If another 'A' appears on the same line, then take out the short and swap pins #2 and #3 (either on the J2 connector or the DB-9 connector).

   d. Reconnect the J2 connector on the board and click **Scan for Boards**.

Entering Commands

All commands to Simple Step boards are prefixed with a network address string. Refer to the information on page 172 to find the prefix string needed for your particular Simple Step board.

NOTE

Network addressing can be disabled using the "on" command (see page 105).

Every command to a Simple Step board terminates with an Enter (Carriage Return - 0x0D).

1. Start SSWin.

NOTE

Make sure that the main screen (Figure 9) displays the correct number of boards for your network.

2. Examine the radio button next to each number and verify that the board addresses are as expected.

3. Select **File** > **Terminal and Network Scanning** from the SSWin menu bar. The *Terminal and Network Scanning Dialog* box is displayed (Figure 11).

4. Click inside the Simple Step Terminal System window area and type the command appropriate to your board type (the examples below assume that the board is at address 0):

Board	User Types	Board Responds
SSMicroMC	D0<Enter>	d0>
SSMicroLC	U0<Enter>	u0>
All Multi-axis and SSNEMA17 boards	X0<Enter>	x0>

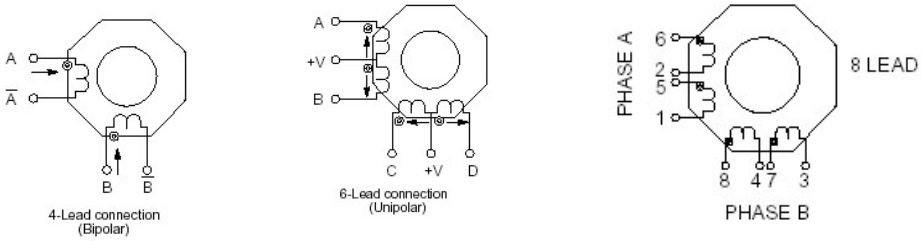
Setting Current Limits for Software Current Limiting Boards

The maximum current, idle current, and current decay settings for the **SSNEMA17**, **SSMicroLC**, **SSXYMicroLC and SSXYZMicroLC** boards are all set using the 'P'ower (or 'P'3 for the SSMicroMC, SSXYMicroMC and SSXYZMicroMC) command. These settings must be entered prior to any motor movement.

Calculate each of these parameters using the instructions on pages 59 and **Error! Bookmark not defined.**, and then submit the 'P'ower command using the SSWin Terminal Mode (see page 38).

Connecting a Motor to the Board

There are two different types of stepper motors, the old unipolar standard and the newer bipolar standard. Unipolar motors are less efficient than bipolar motors by as much as 50%.



All Motor connectors (except for the SSNEMA17) is a 5 pin. 0.156 center JST Sales header that is used on all boards. Pin 5 on all boards (except for the SSNEMA17) is used to connect a shield for the motor cable to help reduce radiated emissions.

Use the information below when connecting a stepper motor to a Simple Step board.

Board Type	Phase A	Phase A Prime	Phase B	Phase B Prime
SSMicroLC SSMicroMC	XMOTOR, Pin 1	XMOTOR, Pin 2	XMOTOR, Pin 3	XMOTOR, Pin 4
SSXYMicroLC SSXYMicroMC	XMOTOR / YMOTOR Pin 1	XMOTOR / YMOTOR Pin 2	XMOTOR / YMOTOR Pin 3	XMOTOR / YMOTOR Pin 4
SSXYZMicroLC SSXYZMicroMC	XMOTOR / YMOTOR / ZMOTOR Pin 1	XMOTOR / YMOTOR / ZMOTOR Pin 2	XMOTOR / YMOTOR / ZMOTOR Pin 3	XMOTOR / YMOTOR / ZMOTOR Pin 4
SSNEMA17	J1, Pin 4	J1, Pin 3	J1, Pin 2	J1, Pin 1

Table 5 Bipolar Motor Connections

Unipolar Connection (Full Copper mode, more torque, less speed):

XMOTOR, Pin 1 = A

XMOTOR, Pin 2 = B

XMOTOR, Pin 3 = C

XMOTOR, Pin 4 = D

Unipolar Connection (Half Copper mode, more speed, less torque):

XMOTOR, Pin 1 = A

XMOTOR, Pin 2 = V+ of #1 coil

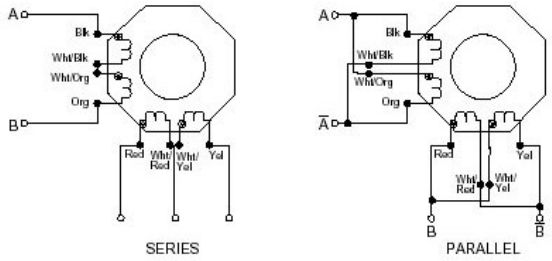
XMOTOR, Pin 3 = C

XMOTOR, Pin 4 = V+ of #2 coil

NOTE

All remaining wires for Unipolar motors connected to the Simple Step boards should be cut and/or taped to prevent accidental shorting to ground or boards.

Eight-leaded motors are considered hybrids. They can be wired in many ways (unipolar or bipolar) and in bipolar mode they can be wired either in series or parallel mode. Series mode gives you more torque and parallel mode gives you more speed.



4-Lead motor
(Bipolar)

The Home Sensor

Now that the power, communications, and motor connections are complete, it is time to decide if a home sensor and limit sensor are required.

Most motion applications have a specific start position or "home" position that defines all motion. To allow for homing, the Simple Step Controller Board drives a slotted optical sensor commonly used in motion applications (i.e. Omron's EE-SX1088-W1).

Sensor power drives the LED with an onboard 50ma current limiting resistor. Therefore, no external current limiting resistor is needed. This is an active high signal (default) that stops the motor when the home signal is detected.

A simple mechanical switch can also be used. The switch should have the common contact connected to XIO (YIO or ZIO) pin 7 (J2, Pin 7 for the SSNEMA17), the normally closed contact connected to XIO (YIO or ZIO) pin 6 (J2, Pin 8 for the SSNEMA17). When home is found, the switch will open the contact indicating "home". Software debouncing is performed on this input. A valid signal is detected when the signal is active for more than 3 µsec (microseconds).

Board	Connector Location
SSNEMA17	J2, Pin 8
SSMicroLC	IO, Pin 5
SSMicroMC	IO, Pin 5
SSXYMicroLC	XIO, Pin 5, YIO, Pin 5
SSXYMicroMC	XIO, Pin 5, YIO, Pin 5
SSXYZMicroLC	XIO, Pin 5, YIO, Pin 5, ZIO, Pin 5
SSXYZMicroMC	XIO, Pin 5, YIO, Pin 5, ZIO, Pin 5

Table 6 Home Sensor Connector Location

Recommended Home Sensors

The following optical sensors are recommended:

- OPB881T55: available from TT electronics' OPTEK Technology
- QVB11334: available from Mouser Electronics Part # 512-QVB11334 or Digi-Key Part # QVB1134QT-ND
- EE-SX1088-W1: available from Digi-Key Part # OR562-ND

NOTE

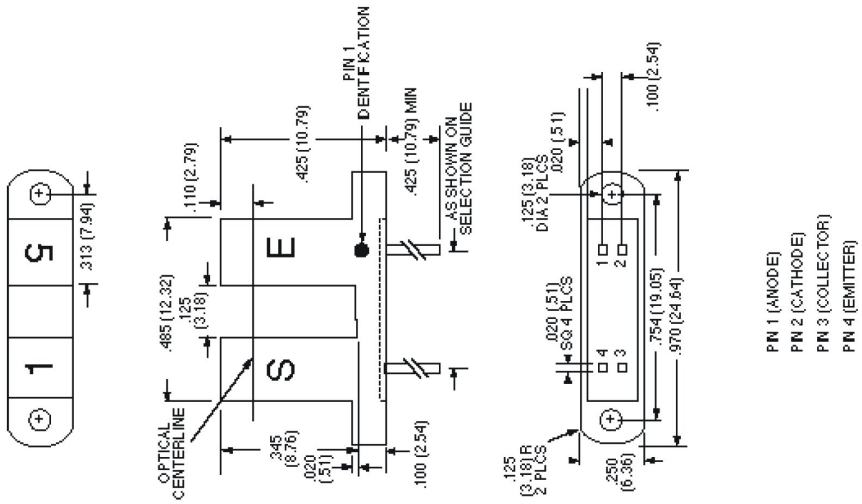
We recommend the EE-SX1088-W1 sensor because it comes with lead wires. The EE-SX1088-W1 sensor is more expensive than the other two sensors, but if the sensor will be wired by hand and not soldered to a PCB, you will save many hours of headaches with the wired version.

An optical sensor can be replaced with another type of sensor. The home sensor input is an active high signal (greater than 2.5 volts). If using a micro-switch for home, connect pin #6 of the Home Connector to the 'C'ommon connection and connect either pins #7 or #8 of the same connector to the 'NC' Normally Closed terminal.

If connecting a limit sensor that is also a micro-switch, connect pin #3 to the 'C'ommon and pin #4 to the 'NO' Normally Open terminal. Current Limiting of pin #5 of the Home Sensor Power is rated to 33ma at 5VDC (20ma at 3.3 VDC for the NEMA17).

NOTE

Software version 1.0.4 and higher allows the user to set the input logic level to whatever is needed for the home input (see Null (Zero/Home) Motor (N) *on page 66).*



Home Conn	Home Signal	Sensor Pin	Opto-Signal	Omron Wire
IO, Pin 1	Shunt (if shorted to IO, Pin 2 will supply 5.0VDC to the Home Sensor LED Drive (IO, Pin 7) up to 10ma			
IO, Pin 2	+5VDC Power (up to 10ma)			
IO, Pin 3	LED Power	1	Anode LED	Red
IO, Pin 4	LED Ground	2	Cathode LED	Black

IO, Pin 5	Sensor Output	3	Collector	White
IO, Pin 6	Sensor Ground	4	Emitter	Green

Figure 12 Home Sensor Hookup

Testing the Home Sensor

First, position the mechanism to the middle of its travel. Using the SSWin Terminal Mode (see page 38), type the *Get Input Port State* command (where ***pa*** is the prefix string for your board type):

***pa*I1<CR> (see page 82)**

The system should respond with a lower-case prefix character, then the board address followed by a greater than symbol ('>') followed by a 5 digit number (or 10 digit if you have the 32-bit option installed). The last digit of the number tells you the current state (logic level) of the home sensor. If it is a '0', then everything is fine.

If the response is a '1', you may have the sensor already blocked, wired incorrectly, or no power connected to the infrared LED within the sensor. Use a small screwdriver or flat piece of metal (DO NOT USE PAPER or CARDBOARD because the sensor can see through most porous materials) to block the sensor and run the test again. A '1' response should be received (for those using a microswitch for home the states may be reversed, since you will have the Normally Open connection instead of the Normally Closed).

NOTE

If you need to invert the active signals for the Home and/or Limit sensors, refer to the 'N' command description on page 66.

Next you need to determine which direction to move the motor to get it to travel to the home sensor. Using the SSWin Terminal Mode (see page 38), type the *Null Motor* command (where ***pa*** is the prefix string for your board type):

***pa*N+0HLS<CR> (see page 66)**

This initializes the motor WITHOUT movement and also tells the system to ignore the home and limit sensors for now.

NOTE

On all firmware versions 1.0.4.30 and greater, and if NO Home sensor is present, append an 'H' character to the end of the 'N' command string to disable the home sensor input.

Now set the motor speed for homing. Try setting the motor movement with the following commands:

***pa*E3000<CR> (see page 71)**

***pa*B100<CR> (see page 70)**

***pa*S3<CR> (see page 72)**

This initializes the motor movement with a standard slope.

Now type the *Move Motor* command:

***pa*M100 (see page 63)**

Place your finger on the *Enter* key. Depress the *Enter* key and watch the motor. If the motor moved towards home, then your initialization string is "***pa*N-1". If the motor moves away from home, then your initialization string is "***pa*N+1".

NOTE

The '1' tells the axis that you are going to initialize the motor, moving it until you reach the home sensor. If NO motor movement is necessary, then change the '1' at the end of the 'N' command string to a zero(0).

The next step is to type in the initialization string for your system:

***pa*N+1<CR>** or ***pa*N-1<CR>**

Watch the motor move towards the home sensor. As soon as the board sees the change of state from the home sensor, the board will stop all motion. If it does not, then turn power off immediately.

Possible problems are:

- The flag on the mechanism is not deep enough into the sensor to block the sensor (for those using microswitches, you may not be pushing the lever down far enough to trigger the switch). You will have to adjust the position of the sensor or bend the lever.
- The sensor ACTIVE state is inverted. If you need to invert the active signals for the Home and and/or Limit sensors, refer to the 'N' command description on page 66.

IMPORTANT

Never disconnect the motor while the motor is moving! This will destroy the motor driver. Never move the motor with NO power applied to the board.

The Limit Sensor

A limit sensor is an active low signal (default) that allows the user to stop the motor. Software debouncing is performed. A valid active signal should be longer than 3 µsec (microseconds).

The motor checks this input before every step is performed when the motor is moving away from home or when the 'R'elative command is issued and the optional parameter is set to 'Y'. If a change of state is detected from the limit sensor, the motor routine is aborted. The board calculates its current position and stores it into the current motor position variable. The motor is allowed to travel in the opposite direction, but cannot be moved any further away from home.

Most times the limit is set as the maximum movement allowed before the mechanism is damaged. The majority of customers use a microswitch, or one of the new Allegro Hall Effect Sensors. If a microswitch is used, then connect the common to one pin of the limit input and the Normally Open connection to the other Pin.

You can check the limit sensor to see if it is working properly in the same manner the home sensor was tested. Type the *Get Input Port State* command (where ***pa*** is the prefix string for your board type):

***pa*I3<CR> (see page 82)**

The system should respond with a lower-case prefix character, then the board address followed by a greater than symbol ('>') and then up to a 10-digit number. The last digit of the number tells you the current state (logic level) of the sensor. If it is deactivated it should come back with a '1', and if it is activated it should come back with as a '0'.

NOTE

If you need to invert the active signals for the Home and/or Limit sensors, refer to the 'N' command description on page 66.

Pin #	Description
IO, Pin 7	Limit Input Signal (10K Pullup, 5.0VDC max.)
IO, Pin 8	Limit Ground

Table 7 IO Connection Breakdown

Pin #	SSNEMA17 J2 Description
1	Limit LED Ground
2	Limit LED Power (3.3 VDC, 20 ma)
3	Limit Ground
4	Limit Input (22K Pullup, 0-5.0 VDC)
5	Home LED Ground
6	Home LED Power (3.3 VDC, 20 ma)
7	Home Ground
8	Home Input (22K Pullup, 0-5.0 VDC)

Table 8 SSNEMA17 Home and Limit Connection Breakdown

Setting Beginning Velocity, End Velocity, and Slope

1. Set E to 4000: ***pa***E4000<CR> (see page 71).
2. Set B to 400: ***pa***B400<CR> (see page 70).
3. Set S to 2: ***pa***S2<CR> (see page 72).
4. Set the current motor position WITHOUT Movement: ***pa***N+0<CR> or ***pa***N-0<CR> (see page 66).
5. Move the motor to position 1000: ***pa***M1000<CR> (see page 63) and watch how the motor moves.
   a. If the motor stalls on the start of the movement, decrease the 'B' by 100 and repeat the test from step 3.
   b. If the motor moves fine, but stalls when it hits the top velocity, decrease the 'E' by 500 and repeat the test from step 3.
   c. If the motor moves without stalling, start increasing the 'E' value by 100 and repeat the test from step 3. Continue increasing the 'E' value until the motor stalls at top velocity. This is the fastest you will be able to move the motor. Decrease the 'E' value by 100 and record that as your starting 'E' value.

Increasing Torque at Startup

If you need more torque at start-up, decrease the 'B'eginning velocity. This will allow you to increase the 'S' lope.

For Example: Your top velocity, 'E', is set to 10300. You find you need a 'B' of 300 to get the mass moving. If your 'S'lope is 2, it takes 'E'-'B' (10300 - 300 = 10000) divided by 'S' (10000/2 = 5000) or 5000 steps to accelerate the motor. Increase the 'S' to 15 or 16 to achieve a 1000 to 2000 step acceleration.

Most customers set the slope from 5 to 20 depending on the mass and motor inductance in conjunction with the motor voltage.

Powering up in the Old Motion Control State

ALL firmware versions 106.09 and above use a 'K' command that allows the axis to automatically run the EEPROM contents on power up (refer to page 109).

These firmware versions can switch back to the old Simple Step motor movement with the "o@LD" command. For example:

00001: K4<CR>

00004: o@LD<CR>

The above EEPROM contents will automatically run on power up and switch the v106 axis back to the old Simple Step motor movement.

Creating a Simple Test Program with SSWin

1. Select **File** > **Edit Def File** from the SSWin menu bar. The *Open* dialog box is displayed.

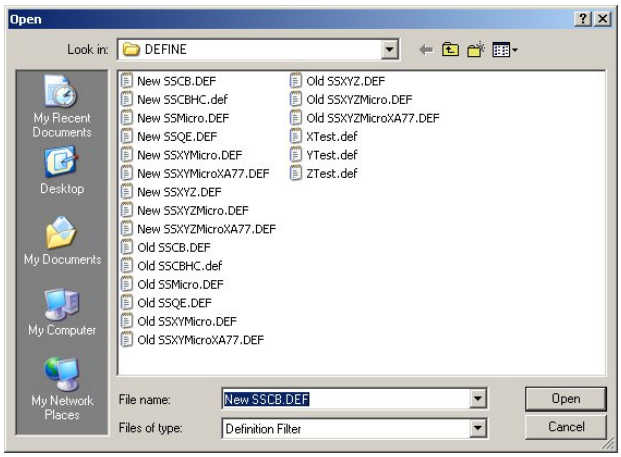


Figure 13 Open Dialog Box

2. Click **Cancel**. The *SSWin Definition Editor* dialog box is displayed.

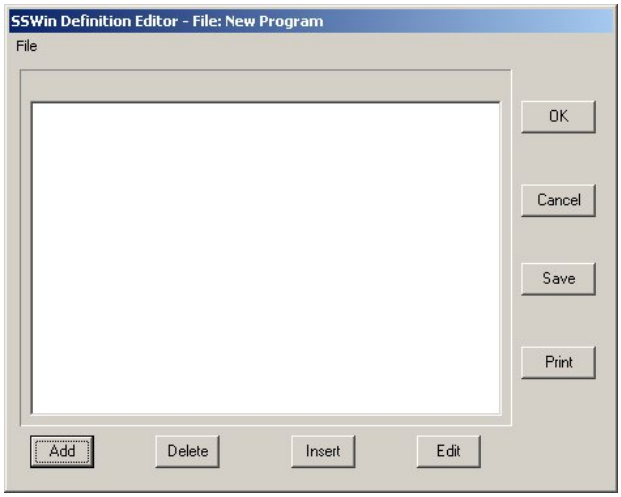


Figure 14 SSWin Definition Editor Dialog Box

3. Click **Add**. The *Please Select Command* dialog box is displayed.

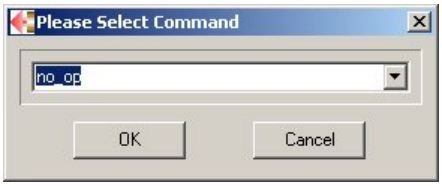


Figure 15 Please Select Command Dialog Box

4. Click the drop-down arrow to selected the desired command (e.g. select SSCB (this is an old reference which has been replaced by the SSMicroMC controller) and click **OK**.

   The *xxxx Command* dialog box is displayed, where *xxxx* is the selected command. The parameters that are available depend on the selected command.

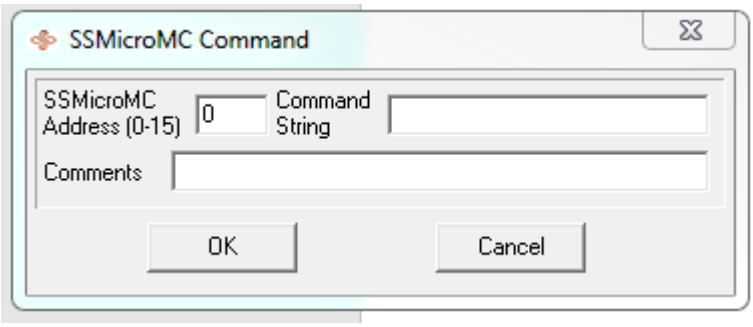


Figure 16 SSMicroMC Command Dialog Box

5. Enter the command parameters and optional comments and click **OK**. The command is added to the display.

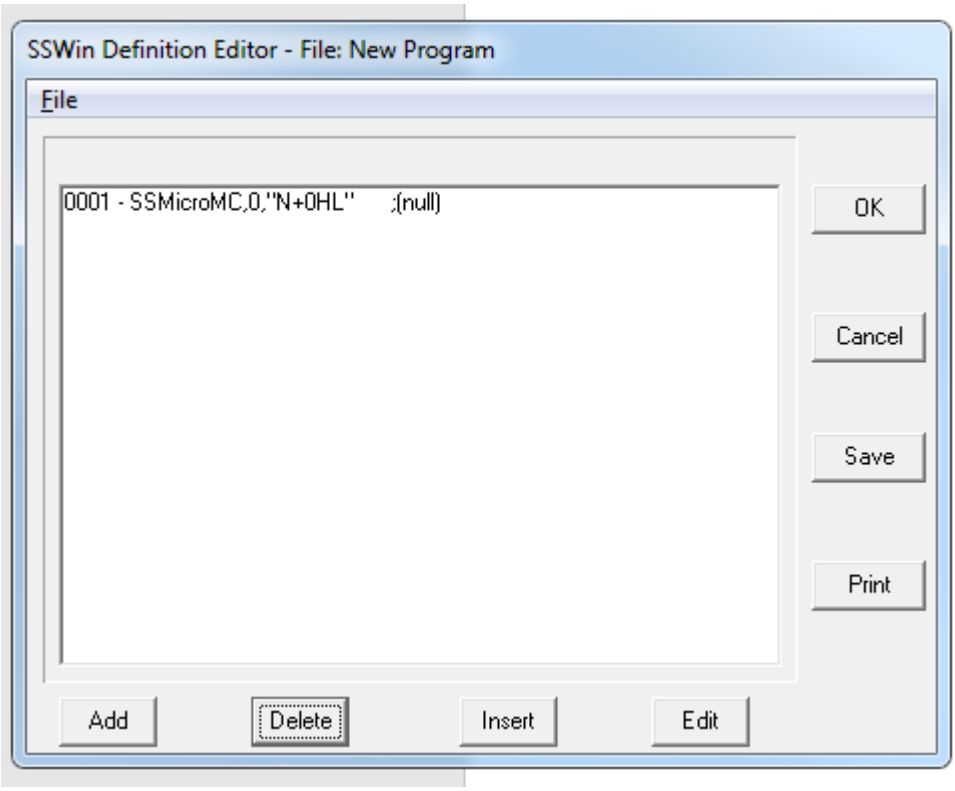


Figure 17 Sample Program Entry

6. Select the appropriate board type and create one of the sample programs below:

SSMicroMC:

001 - SSMicroMC,0,”N+0HL”

002 - SSMicroMC,0,”M1000”

003 - wait_SSMicroMC_done,0

004 - SSMicroMC,0,”M0”

005 - wait_SSMicroMC_done,0

006 - cloop,3,2

007 - end_program

All Multi Axis Boards (SSNEMA17, SSXYMicroLC, SSXYMicroMC, SSXYZMicroLC and SSXYZMicroMC):

001 - SSXYZMicroXX,0,"XN+0HL"

002 - SSXYZMicroXX,0,"YN+0HL"

003 - SSXYZMicroXX,0,"ZN+0HL"

004 - SSXYZMicroXX,0,"XM1000,YM1000,ZM1000"

005 - wait_ SSXYZMicroXX _done,0,A

006 - SSXYZMicroXX,0,"XM0,YM0,ZM0"

007 - wait_ SSXYZMicroXX _done,0,A

008 - cloop,3,4

009 - end_program

SSQE

001 - ssqe,1,"N+0"

002 - ssqe,1,"N+1"

003 - ssqe,1,"P"

004 - loop,3

005 - end_program

NOTE

*Click **Add** to insert a new command below the last line in the program. Click **Delete** to delete the currently highlighted command. Click **Insert** to insert a new command above the currently highlighted command. Click **Edit** to modify the currently highlighted command.*

7. Click **Save,** choose a file name for the new program, and click **Save** again.

8. Click **OK** to close editor.

9. Click **Run Program** to start the program.

   The SSMicroMC and programs move the motors from position 0 to position 3000 and then back to position 0 three (3) times and stop. The SSQE program allows you to move the encoders. You will see the encoder values change on the right side of the SSWin main screen.

10. Click the **Abort** button at any time to terminate the program.

User I/O

There are two (2) software controlled outputs. The first and primary output of ***each axis*** is the MOSFET output. This is part of the Home and Limit 8 pin connector on all motion control boards as shown in Table 7.

This output is diode clamped to allow the user to connect relays, solenoids, etc. without the worry of external components. This is a sink output, and the motor power is brought to the opposite pin. This drive can sink up to 500 millamps DC.

The breakdown for this drive varies on the board type as outlined in Table 9.

Pin #	SSNEMA17 (JR)	Pin #	SSMicroLC (IO)	SSMicroMC (IO)	SSXY MicroLC XIO, YIO	SSXY MicroMC XIO, YIO	SSXYZ MicroLC XIO, YIO, ZIO,	SSXYZ MicroMC XIO, YIO, ZIO,
1	User 1 (22K Pullup)	9	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)
2	Digital Ground	10	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground
3	User 2 (22K Pullup)	11	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)
4	Digital Ground	12	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground
5	User 1 (22K Pullup)	13	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)
6	Digital Ground	14	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground
7	Anlgin (0-3.3VDC input ONLY with no Pullup)	21	SPARE 1 (No Pullup, 0-3.3VDC ONLY)	SPARE 1 (No Pullup, 0-3.3VDC ONLY)	SPARE 1 (No Pullup, 0-3.3VDC ONLY)	SPARE 1 (No Pullup, 0-3.3VDC ONLY)	SPARE 1 (No Pullup, 0-3.3VDC ONLY)	SPARE 1 (No Pullup, 0-3.3VDC ONLY)
8	MOSFET Output 0 (500ma	22	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground
9	J1, Pin 9 or 10 Power (+)	23	SPARE 2 (No Pullup, 0-3.3VDC ONLY)	SPARE 2 (No Pullup, 0-3.3VDC ONLY)	SPARE 2 (No Pullup, 0-3.3VDC ONLY)	SPARE 2 (No Pullup, 0-3.3VDC ONLY)	SPARE 2 (No Pullup, 0-3.3VDC ONLY)	SPARE 2 (No Pullup, 0-3.3VDC ONLY)
		24	SPARE 3 (No Pullup, 0-3.3VDC ONLY)	SPARE 3 (No Pullup, 0-3.3VDC ONLY)	SPARE 3 (No Pullup, 0-3.3VDC ONLY)	SPARE 3 (No Pullup, 0-3.3VDC ONLY)	SPARE 3 (No Pullup, 0-3.3VDC ONLY)	SPARE 3 (No Pullup, 0-3.3VDC ONLY)
		25	MOSFET Output 0 (500ma)	MOSFET Output 0 (500ma)	MOSFET Output 0 (500ma)	MOSFET Output 0 (500ma)	MOSFET Output 0 (500ma)	MOSFET Output 0 (500ma)
		26	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1
		27	MOSFET	MOSFET	MOSFET	MOSFET	MOSFET	MOSFET

			Output 1 (500ma)	Output 1 (500ma)	Output 1 (500ma)	Output 1 (500ma)	Output 1 (500ma)	Output 1 (500ma)
		28	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1	POWER, Pin 1

Table 9 User I/O Connector Breakdown for all Boards

No hardware debouncing is performed and should be taken care of by the connected circuitry. Most of the User Inputs have 10K Pullup resistors on board. The only time this is not the case is User Input #3 for the boards that have User #3 Input. These are general purpose inputs with voltage input levels that are CMOS/TTL Input thresholds. 0 to 5VDC logic only is allowed. No diode clamping is performed. The “I4”, “I5” and “I6” commands allow the user to connect the inputs to sensors, switches, etc. and have it report them back to the host.

Most customers use these inputs to allow a multi-axis system to control each other. With the -IEE option, they can wait for a logic state to change and then run a predefined program internal to the system. Many automation companies use this feature.

Example

SSXYZMicroMC Board User Input #1 is connected to a PLC. The PLC holds the User Input #1 Line Low to tell the X-axis to start its program. In the X-axis program, the X-axis turns on its Output #1 which is connected to User Input #1 of the Y-axis. The Y-axis now starts its preprogrammed sequence. Since Output #1 has no pull-up on board, the user connected a resistor (22K, 5%, 1/4W) between the output and the option connector Pin 1 (+5 VDC Power from the on-board switcher).

General Purpose Inputs/Outputs

The general purpose input/output feature is only available on units that have USER I/O. This feature allows ALL the User Inputs and Outputs, including the I/O (SPARE 1-n connections), on the IO connector as programmable I/O lines. Either the 'I'nput (see page 82) or 'O'utput (see page 83) commands can be used with the same I/O parameter value.

NOTE

This does not include Home and Limit or the MOSFET control lines.

To use a line as an Output, issue the 'O'utput command with the port number parameter from Table 11. To change the line back to an Input line, first 'O'utput a '1' value to the line, and then use the 'I'nput command to read the line back. Otherwise, you will see the initialized or last state that was written to the line.

Pin	SS MicroLC IO	SS MicroMC IO	SSXY MicroLC XIO, YIO	SSXY MicroMC XIO, YIO	SSXYZ MicroLC XIO, YIO, ZIO	SSXYZ MicroMC XIO, YIO, ZIO
21	SPARE 1	SPARE 1	SPARE 1	SPARE 1	SPARE 1	SPARE 1
23	SPARE 2	SPARE 2	SPARE 2	SPARE 2	SPARE 2	SPARE 2
24	SPARE 3	SPARE 3	SPARE 3	SPARE 3	SPARE 3	SPARE 3
20	+3.3 VDC (50ma)	+3.3 VDC (50ma)	+5 VDC (50ma)	+5 VDC (50ma)	+3.3 VDC (20ma)	+3.3 VDC (20ma)
22	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground	Digital Ground

Table 10 IO Connector Breakdown for all Boards

General I/O Number	SSNEMA17	SS MicroLC	SS MicroMC	SSXY MicroLC	SSXY MicroMC	SSXYZ MicroLC	SSXYZ MicroMC
21	User 1	User 1	User 1	User 1	User 1	User 1	User 1
22	User 2	User 2	User 2	User 2	User 2	User 2	User 2
23		User 3	User 3	User 3	User 3	User 3	User 3
24		Spare 1	Spare 1	Spare 1	Spare 1	Spare 1	Spare 1
25		Spare 2	Spare 2	Spare 2	Spare 2	Spare 2	Spare 2
26		Spare 3	Spare 3	Spare 3	Spare 3	Spare 3	Spare 3

Table 11 General I/O Cross-reference Chart

Connecting a Quadrature Encoder Board

All Simple Step boards directly accommodate a quadrature encoder. There are four connections for each quadrature encoder. Two are for power to the sensors and two are for quadrature encoder channels A & B (see Table 12).

Channel	SSNEMA17	All other boards IO, XIO, YIO, ZIO
Encoder Channel A	ENC, Pin 1	IO, Pin 15
Encoder Channel B	ENC, Pin 2	IO, Pin 17
Index Input	ENC, Pin 3	IO, Pin 13 (Shared with USER 3)
+5.0 VDC (20ma)	ENC, Pin 4	IO, Pin 16
Ground	ENC, Pin 5	IO, Pin 18

Table 12 Encoder Sensor Connections

Refer to Chapter 7 for a description of quadrature encoder commands.

NOTE

U.S. Digital Corporation (refer to page 6 for contact information) is sugessted as a source for quadrature encoders that include an easily installed casing.

U.S. Digital Encoder E2-100-250-H is recommended for use with a standard 1.8 degree stepper. (Use the x4 mode, which allows 400 steps per revolution.)

This page intentionally left blank.

CHAPTER 4: POWER SETTING COMMANDS

This chapter describes commands that are used to set and check the maximum current, idle current, and decay currents. The format of the power setting command is dependent on the board type.

Command Format Used in Examples

Unless otherwise indicated all commands start with the board type prefix character followed by a one-digit board address/status (***pa*** in the examples) and end with a carriage return, 0xD (<CR> in the examples).

NOTE

See page 172 for board prefix characters.

NOTE

See page **Error! Bookmark not defined.** for board address format.

NOTE

See page 174 board status values.

Software Settable Current Limit Boards

Set Motor Power Mode (P)

Description: Sets the maximum current, idle current, and decay current per phase for boards that use software controlled current limiting.

Before the board starts any motor movement, it activates the motor to full power using the defined maximum current value. After the motor movement is complete, the board sets the motor to the user-defined idle current value.

All settings are performed via an 8-bit DAC. The voltage range for the DAC is from 0 volts (no power) to 5 volts DC. The maximum current setting is calculated using the board's maximum current per phase rating.

Parameters:

Parameter	Description	Values
Maximum	Maximum current in DAC increments (see below).	1 to 255 Default = 1
Idle	Idle current in DAC increments (see below).	0 to 255 Default = 0 (no power) Cannot exceed the maximum current value. Cannot exceed 0.5 amps.
Decay	Decay current in DAC increments where each DAC increment is 0.01953125 volts (5 volts DC / 256)	0 to 255 Default = 0 (no decay) • Slow decay > = 3.5 V • Mixed decay 1.2 V to 2.9 V • Fast decay < = 1.8 V

SSNEMA17, SSMicroLC, **SSXYMicroLC**, and **SSXYZMicroLC** full power is 1.5 amps:
Resolution per increment = 1.5 amps / 256 = 0.005859375 amps

SSMicroMC, **SSXYMicroMC**, and **SSXYZMicroMC** full power is 3.125 amps: Resolution per increment = 3.125 amps / 256 = 0.01220703125 amps

Example:

SSMicroLC, SSXYMicroLC, and SSXYZMicroLC:

paP128,17,0<CR>

Assumes that the motor that will be attached is rated at 0.75 amps per phase and that the selected idle current mode is 0.10 amps:

0.75 amps / 0.005859375 = DAC setting of 128
0.10 amps / 0.005859375 = DAC setting of 17

SSMicroMC, SSXYZMicroMC, and SSXYZMicroMC:

paP3,128,17,0<CR>

Assumes that the motor that will be attached is rated at 0.75 amps per phase and that the selected idle current mode is 0.10 amps:

0.75 amps / 0.01220703125 = DAC setting of 61.44 or 61

0.10 amps / 0.01220703125 = DAC setting of 8.192 or 8

NOTE

The MicroMC series can be purchased of 1 of 2 modes. The first mode uses the standard 'P' command settings as all other boards. Mode 2 will then use the Standard SSCB, SSXYQE and SSXYZ commands to control the power with the P0, P1 and P2 commands with the new P3 to set the software controlled Power settings. The old SSCB, SSXYQE and SSXYZ used hardware power setting and idle was preset to 25%. With the new P3 command now supplanting the hardware settings, the user now has the ability to set the Power and Idle current to whatever they need.

NOTE

When using quadrature encoder feedback (or when the mass that is being moved is very small), some idle current should be applied to the motor. Otherwise, when the motor stops and the magnetic fields collapse, the motor can burp from 1 to as much as 20 steps in any direction (usually in the direction the motor was moving). Adding a small idle current will stop this from occurring.

Get Motor Power Settings (p)

Description: Retrieves settings for maximum current per phase, idle mode current per phase, decay mode, and step mode.

Parameters:

Parameter	Description	Values
Setting	Determines which setting is returned.	0 = Maximum current 1 = Idle mode current 2 = Decay mode 3 = Step mode 4 = User Prescale 5 = System Prescale 6 = LED Flash sequence 7 = LED Flash Error count

Example: ***pa***p0<CR>
Asks for the Maximum current setting

Values in the response are separated by commas.

CHAPTER 5: STANDARD COMMANDS

This chapter describes commands for the following functions:

- Set and determine motor position, velocity, and slope
- Abort motor movement
- Set the prescaler option
- Set stepping mode
- Set motor timing
- Set motor software limits
- Set motor JOG control
- Perform a delay
- Get status and board capability information

Commands apply to all boards unless otherwise indicated.

Move Motor (M)

Description: Moves the motor to a new absolute position.

The Simple Step Controller Board sets the correct motor direction and automatically enables the motor to move to the new position. Once this has been achieved the Simple Step Controller Board automatically turns off power to the motor.

Parameters:

Parameter	Description	Values
Position	Absolute position.	± 2,147,483,646

Example: ***pa***M0<CR>

Get Current Motor Position (m)

Description: Retrieves the current motor position.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***m<CR>

Relative Motor Movement (R)

Description: Moves the motor the indicated number of steps relative to its current position.

This command allows the motor to go past its programmed steps of absolute travel, therefore it can overload the 32-bit motor position variable.

The RTOS display position register command (see page 86) can be used to get the true position value. This includes boards that do not have the RTOS option installed.

Parameters:

Parameter	Description	Values
Home sensor	Look for home sensor while moving. If found, stop motor and zero motor position counter.	Y = Look for sensor N = Don't look for sensor
Limit sensor (optional)	Look for limit sensor while moving. If found stop motor.	Y = Look for sensor (default) N= Don't look for sensor
Direction	Direction to move the motor.	+ = Away from home – = Towards home
Steps	Number or steps to move.	± 2,147,483,646

Example: ***pa***RYY+200<CR>

Continuous Motor Movement (C)

Description: Starts a continuous motion.

This command stays in effect until either a soft abort or hard abort command is received or the power is turned off.

If the RTOS option is installed, the motor speed can be changed while the motor is moving (accelerate or decelerate from its current setting using the defined 'S'lope value). If the user changes the 'E' nd velocity value, the 'B' eginning velocity value is recalculated to determine the offset difference between the old 'E' and 'B' values. This new value is stored in the 'B' register so that the current user-defined slope setting can be maintained and the motor is accelerated/decelerated to the new 'E' value. While this is occurring, the board status is 'u' (updating speed) until the requested 'E' value is obtained. The status then changes back to the normal 'b' (busy) state.

Parameters:

Parameter	Description	Values
Direction	Direction to move the motor.	+ = Away from home – = Towards home

Example: ***pa***C+<CR>

Null (Zero/Home) Motor (N)

Description: Moves the motor to the absolute 0 position by powering on the motor and then stepping in the specified direction until the home sensor is found.


NOTE

After power up, the 'N' command must be the first command given for all hardware controlled current limiting boards before any motor movement commands are processed.

Once the command to home the motor in a particular direction is performed, the system always moves towards home when a 0 direction movement command is issued. After the command is processed the Simple Step Controller Board turns off the motor power and zeros out the current position register.

The Null (Zero/Home) command performs the home motor movement based on the current Beginning, End, and Slope values.


NOTE

If a home sensor will not be used, the user can instruct the board to zero the current position register without moving the motor, "N+0". An 's' status is returned.


NOTE

If a quadrature encoder is connected for motion to an axis, the user SHOULD put the axis into a low power idle mode BEFORE the 'N' command is given.


NOTE

If there is no home sensor present, include the 'H' parameter to disable the home sensor input.

Parameters:

Parameter	Description	Values
Direction	Direction to home the motor.	+ = Clockwise – = Counter-clockwise
Initialization	Determines if the home sensor is used for initialization.	0 = Ignore the home sensor and initialize the board at its current position 1 = Use the home sensor and move the motor until the home sensor is found
Strip (optional)	Strips all leading zeros (0) off all numeric responses sent from the board.	S

Parameters:

Parameter	Description	Values
Disable home sensor (optional)	Disables the home sensor input. This allows the user to use this input as a standard user input.	H
Disable limit sensor (optional)	Disables the limit sensor input. This allows the user to use this input as a standard user input.	L
No acknowledgement (optional)	When this parameter is included, the board will not acknowledge commands sent from the host.	n
No status (optional)	When this parameter is included, no status characters are sent back to the host on acknowledgements.	a
Status (optional)	Allows the host to receive the current status from the board while the board is running.	s
Finished Movement (optional)	Allows the axis to send back a 'f'inished motor movement status automatically. **Note**: If several axes are running at the same time, and this parameter is active on more than one axis, there is a chance that a communication crash will occur.	c
Skip Prefix (optional)	Tells the board to skip the prefix string being sent back to the host on all responses.	h
Not Zero (optional)	Allows the board to not zero the current position register when the home sensor is tripped.	z
IEEPROM (optional)	Allows IEEPROM messages to be sent out when the board is running in IEEPROM mode.	e

Parameters:

Parameter	Description	Values
"10" Sensor Trip (optional)	This is a two-character parameter. The first is the TRIP value for the home sensor and the second is the TRIP value for the limit sensor. Both must be specified if this parameter is included. The original Simple Step Controller Boards only allowed an active high (1) to trigger the home sensor and an active low (0) to trigger the limit sensor. This new parameter allows a '0' or a '1' for the home sensor and a '0' or '1' for the limit input. *For example*: If you are using a microswitch for both sensors and you want both sensors to be active lows, the parameter syntax is "00".	11, 00, 10, or 01

Example:

paN+0<CR>
Direction to home the motor is clockwise, no motor movement.

paN-1<CR>
Direction to home the motor is counterclockwise, home the motor in that direction while looking for the home sensor.

paN+0S00<CR>
Set both sensors to active lows. No motor movement, home is in the clockwise direction.

paN+000S<CR>
Note the optional parameters can be in any position.

Set Absolute Motor Position (A)

Description: Sets the current motor position variable to the indicated value.

This parameter is normally set via the encoder position

Parameters:

Parameter	Description	Values
Position	Absolute motor position.	± 2,147,483,646

Example: ***pa***A100<CR>

Motor Hard Abort (*)

Description: Aborts any motor movement that is underway.

This command replaces the general motor abort command (Escape character). The user can now abort each motor individually on each board connected to the network.

The board calculates its current position and stores that value in the current motor position variable. The board responds with its current status as an acknowledgment.

Parameters:

Parameter	Description	Values
none		

Example: ***pa****<CR>

Motor Soft Abort (!)

Description: Performs the deceleration process and then stops the motor movement.

The board calculates its current position and stores that value in the current motor position variable. The board does NOT respond with its current status as an acknowledgment.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***!<CR>

Set Beginning Velocity (B)

Description: Sets the beginning velocity.

This parameter should be set according to motor load conditions. If this parameter is set too high, the motor will stall when starting to move.

Parameters:

Parameter	Description	Values
Velocity	Beginning velocity.	1 (slowest) to 13,000 (fastest) 300 = default

Example: ***pa***B1000<CR>


NOTE

See page

Setting Beginning Velocity, End Velocity, and Slope on page 46.

Get Beginning Velocity (b)

Description: Retrieves the current setting for beginning velocity.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***b<CR>

Set End Velocity (E)

Description: Sets the end (top or constant) velocity.

This parameter should be set according to motor load conditions. If this parameter is set too high, the motor will stall when trying to achieve top velocity.


NOTE

If the beginning and end velocities are the same, the motor will run at a constant velocity throughout the total motion with no slope (no acceleration or deceleration).

Parameters:

Parameter	Description	Values
Velocity	End velocity.	Full Step: 1 (slowest) to 20,000 (fastest) 1/2 to 1/32 : 1 (slowest) to 30,000 (fastest) 4000 = default

Example: ***pa***E3000<CR>


NOTE

See page

Setting Beginning Velocity, End Velocity, and Slope on page 46.

Get End Velocity (e)

Description: Retrieves the current setting for end velocity.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***e<CR>

Set Slope (S)

Description: Sets the slope (acceleration / deceleration) of the motor movement.

If the slope value is set too low, the system produces a very long acceleration / deceleration that is usually used when moving large loads. A setting that is too high will stall the motor when acceleration is being performed and start again when deceleration is being performed.

The link between beginning, end, and slope values allows the system to handle a wide range of dynamic loads. Beginning, end, and slope values that are high can move 65,534 steps in less than 9.4 seconds.

Parameters:

Parameter	Description	Values
Slope	Slope.	1 (slowest) to 2000 (fastest) 10 = default

Example: ***pa***S100<CR>

NOTE

See page

Setting Beginning Velocity, End Velocity, and Slope on page 46.

Get Slope (s)

Description: Retrieves the current setting for slope.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***s<CR>

Set Prescaler Option (r)

Description: This command stretches the current step time by the specified value. A value of one (1) tells the system to use an x1 clock value. A value of two (2) tells the system to use an x2 clock and so on, up to x255.

The prescaler option can slow the step rate down to very slow speeds, which can create a lot of heat.


NOTE

When a prescaler value > 1 is used, it is recommended that a fan be running from 25 to 37 CFM or higher to keep the board from overheating and shutting down the motor. Motor shutdown will occur if the motor driver IC reaches 165 °C. A fan must be used for speeds of 300 sps or less and a prescaler value of 1 or less.


NOTE

The prescaler value for SSNEMA17 boards cannot exceed 190 when the top velocity value is 1.

Parameters:

Parameter	Description	Values
Step Time	Clock value multiplier.	1 (default) to 255

Example: ***pa***r2<CR>

Set Motor to Half Step Mode (H)

Description: Changes the motor stepping mode to half step.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***H<CR>

NOTE

Applies to standard boards.

Set Motor to Full Step Mode (F)

Description: Changes the motor stepping mode to full step.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***F<CR>

NOTE

Applies to standard boards.

Set Microstepping Mode (H)

Description: Changes the stepping mode for microstepping boards.

Parameters:

Parameter	Description	Values	Step count with a 1.8 degree stepper
Mode	Stepping mode	0: Full Step mode (1/1)	200 Steps/Rev.
		1: Half Step mode (1/2) default *on power up*	400 Steps/Rev.
		2: Quarter Step mode (1/4)	800 Steps/Rev.
		3: Eighth Step mode (1/8)	1600 Steps/Rev.
		4: Sixteenth Step mode (1/16)	3200 Steps/Rev.
		5: Thirty-second Step mode (1/32)	6400 Steps/Rev.

Example: ***pa****H0<CR>*

NOTE

Only applies to microstepping boards.

Set Motor Settling Timer (oP)

Description: Changes the motor delay timer after motion is completed.

This allows for a settling time before power is set to its programmed idle mode (either Full, 1/4, or Off on standard boards).

Parameters:

Parameter	Description	Values
Settling Timer	Delay timer.	4 (default at power up) = 4 milliseconds delay. 0 to 65534 motor stays on at full power in 1ms increments. 0 turns off the timer

Example: ***pa***oP2<CR>

NOTE

Does not apply to boards with firmware version 101.xx to 104.xx.

Set Motor JOG control (oj)

Description: Puts the Simple Step Controller Board into JOG mode.

This command uses two inputs (USER 1 and USER 2) of each axis to activate the motion towards home (USER 1) and away from home (USER 2). The two inputs are normally connected to a microswitch (normally open) type joystick.

One company that sells industrial/amusement type of joysticks is Happ Controls (http://www.happcontrols.com/). Look for universal joysticks in the amusement section. These joysticks are inexpensive (under $20.00 USD in single quantity) and can take a lot of punishment.

One terminal of the microswitch (terminal marker NO) is connected to the USER 1 input and the microswitch common ground is connected to the ground connection of the axis (usually the next pin in line). USER #2 input is connected to the other microswitch terminal (normally marked NO) and the common ground of that microswitch is connected to the ground connection of the board (usually the next pin next to USER 2).

NOTE

The only valid commands when an axis is in JOG mode are "ojN<CR>" to disable JOG mode if no motor movement is present, or the abort commands "*<CR> "or "!<CR>". All other commands will cause unpredictable results.

Parameters:

Parameter	Description	Values
JOG Mode	Enable or disable JOG mode.	Y = Enable JOG Mode N = Disable JOG Mode S = Set JOG Mode parameters **Note**: Remaining parameters only apply when JOG Mode is 'S'.
b	Beginning velocity	1 (slowest) to 12,000 (fastest) 300 = default
e	End velocity	1 (slowest) to 15,000 (fastest) 4000 = default
s	Slope	1 (slowest) to 200 (fastest) 10 = default
ll	Lower software limit	0 to 2,147,483,646
lh	Upper software limit	0 to 2,147,483,646

Example:

paojY<CR>
Enable JOG Mode.

paojS100,4000,1,0,65534<CR>
Give the axis the following jog settings:

- Beginning velocity = 100
- End velocity = 4000
- Slope = 1
- Lower software limit = 0
- Upper software limit = 65534.

NOTE

A new status has been added that allows the user to know if an axis is in JOG mode. The new status is a 'j' character that is sent back to the host instead of the '>' ready (see page 174).

Get Status ()

Description: Retrieves the last status value that was sent to the host.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***<CR>


NOTE

See status values on page 174.

Perform Delay (d)

Description: Performs a delay. NOTE: Value #3 uses the processor clock timebase for the delay function.

Parameters:

Parameter	Description	Values
Delay option	Tells the delay routine what to do.	0 = Standard delay routine based on a 1 msec timebase 1 = Not used anymore 2 = Not used anymore 3 = Super timebase delay. This is processor clock dependent.
Timer	Timer value.	1 to 65534 or if #3 1 to 2147483646

Example: ***pa***d1,255<CR>


NOTE

Parameters are separated by a comma.

Get Board Information (v)

Description: Allows the user to determine the board capabilities.

Parameters:

Parameter	Description	Values
Information	Information type.	0 = Primary software version number 1 = Board Type 2 = Board options #1 3 = CPU Type 4 = Sub-version software number 5 = Board options #2 6 = Board revision level 7 = Board options #3 8 =Motor driver type 9 = Firmware test version number A = Firmware compile date B = Firmware compile time C = Board options #4 D = IAP version number E = IAP Sub-version number F = Internal IEEPROM flash size 10 = Customer Number 11 = Board option #5 12 = Code version 13 = Code sub-version 14 = Build number 15 = Core number 16 = Processor Mode 17 = System ICC 18 = ARM Code 19 = Little Endian value 1A = Global ms timer 1B = Board options #6 1C = Board options #7 1D = Board options #8 1E = Board options #9 1F = Board code type 20 = Not Used 21 = Board Type 22 = Board Counter 23 = Core Voltage Input 24 = Last Motor Move value 25 = RTOS System Count 26 = Watchdog Count

		27 = Watchdog CRC1 28 = Watchdog CRC2 29 = Watchdog Error Count 2A = Axis system clock 2B = Serial TXD delay (ms) 2C = Perform a Watchdog Reset E6 = Moving To position E7 = QE Error Count E8 = Motor Tasker Switch E9 = Motor Parameter count EA = Status register value EE = Motor Shutdown value EF = Moto Running value 140 = Current Per Phase value 141 = Max Stepping Mode

Example: ***pa***v3<*CR*>
Requests the board's CPU type


NOTE

See response values on page 176.

Zero Previous and Current Status (z)

Description: Resets the current and previous status of the axes to ready (a '>' character).

Parameters:

Parameter	Description	Values
none		

Example: ***pa***z<CR>

Input Port State (I)

Description: Retrieves the current condition of an input port.

The response back is either a zero (low) or one (high).


NOTE

Refer to *General Purpose Inputs/Outputs* on page 54 for information on general I/O commands that allow using the lines as ALL inputs, ALL outputs, or a mixture.

Parameters:

Parameter	Description	Values
Port Number	Input port number (see Table 13).	1 to 6

Example: ***pa***I3<*CR*>

Input Port Number	SS NEMA17	SS MicroLC	SS MicroMC	SSXY MicroLC	SSXY MicroMC	SSXYZ MicroLC	SSXYZ MicroMC
1	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)	Home Sensor (10K Pullup)
2	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used
3	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)	Limit Sensor (10K Pullup)
4	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)	User 1 (10K Pullup)
5	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)	User 2 (10K Pullup)
6		User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)	User 3 (10K Pullup)

Table 13 Input Command Breakdown for All Boards

Set Output Control Line (O)

Description: Sets the state of either of the two user controlled output port lines.


NOTE

Refer to *General Purpose Inputs/Outputs* on page 54 for information on general I/O commands that allow using the lines as ALL inputs, ALL outputs or a mixture.

Parameters:

Parameter	Description	Values
Port Number	Output port number (see Table 14).	0 or 1
State	Port setting.	0 = Low 1 = High

Example: ***pa***O,1<CR>

Output Port Number	SSNEMA17	SS MicroLC	SS MicroMC	SSXY MicroLC	SSXY MicroMC	SSXYZ MicroLC	SSXYZ MicroMC
0	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output
1		500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output	500 ma MOSFET Output

Table 14 Output Command Breakdown for All Boards


NOTE

All outputs are Open Collector.


NOTE

All boards except the SSNEMA17 have the outputs are diode clamped for relay control. The SSNEMA17 has no diode clamp so one must be installed to prevent damage to the controller.

CHAPTER 6: RTOS COMMANDS

This chapter describes the Real-Time Operating System (RTOS) and associated commands. These commands are only applicable to boards with the RTOS option installed.

The Real Time Operating System (RTOS)

The RTOS (Real-Time Operating Software) is a unique operating system that was developed for the University of Arizona. It allows the Simple Step Controller Board to communicate and run commands while a motor is moving.

What makes this software so unique is that it allows the host (when the Continuous command is used) to change the speed of the motor (with acceleration/ deceleration) while the motor is moving. The RTOS allows the user to send 'E' and 'S' values while the motor is moving.

The system stacks (layers) the motor movement commands with their associated 'E', 'B', and 'S' values to 8 deep. When the host requests a motor movement, all the standard communications apply, except that after the motor movement command is implemented and the host requests an axis status, the axis returns one of three (3) responses:

- The board is 'u'pdating the current speed to new value
- The motor is 'b'usy with the current motion command
- The motor is 'w'aiting for the current motor move command to complete and the stack to be unloaded

The host can also run other commands while the motor is moving such as checking Input Ports, Output Ports, current motor position, etc.

Restricted RTOS commands

If a motor command is given, and the status response is either 'u', or 'b', the host can request another move command, and the parser will accept as much information as it can and wait for the current motor move to complete. Commands that wait for the last motor movement are listed below.

Command	Description
M	Move motor to absolute position
C	Continuous motor move
R	Move motor relative
N	Initialize motor/board
A	Set motor position to another value without moving motor
P	Motor power command
d	Delay command
H	Half step mode or stepping mode
F	Full step mode

Table 15 Restricted RTOS Commands while Motor is Moving

Set Abort Mode (oa)

Description: The oa command allows the user to choose whether waiting commands are processed or aborted when an abort command is given.

By default, the abort commands ('*' or '!') terminate both current and waiting commands. If the 'oaY' command is issued only the current command is aborted, waiting commands are allowed to run.


NOTE

This command should be given as part of the initialization process, NOT while commands are in process.

Parameters:

Parameter	Description	Values
Abort Mode	Abort mode setting.	Y = Allow waiting command to run N = Abort waiting commands (default)

Example: ***pa***oaY<CR>

Set/Get RTOS Position Register (op)

Description: This command instructs the Simple Step Controller Board to capture, display, or reset the global 32-bit RTOS position counter.

The RTOS updates a 32-bit motor position counter when the motor is in motion. This register is a 32-bit register that allows the user to capture the current position of any axis.

This command is linked with the standard position commands ('m', 'A', etc.).

When the home signal is activated, both the current motor position and RTOS position are reset to zero.

Parameters:

Parameter	Description	Values
Capture	Capture current RTOS position and save in capture register.	C
Display current	Display (in binary format) the current capture register.	D
Display RTOS	Display (in binary format) the current RTOS position register.	d
Reset	Reset RTOS position counter to 0.	R
Set	Set local current motor position using the current RTOS position.	s

Example: ***pa***opd<CR>

The "opd" command can be used even if the RTOS option is not installed. Other parameters for the "op" command are restricted to boards running the RTOS.

Set Software Limits (os)

Description: Sets motor software limits when running in Continuous ("C+" or "C-" command) mode. **NOTE:** On newer boards (ARM) this command can be used for C, R and M commands.

The positions are positive values only that allow the user to set lower and upper motion limits. The Simple Step Controller Board calculates the deceleration values that are currently set, and starts deceleration of the motor automatically so that the motor stops at the current set point. The deceleration point is re-calculated every time the 'E' command is issued while the motor is moving.

Parameters:

Parameter	Description	Values
Limit control	Set or display limit control.	y or Y = Turn on limit control n or N = Turn off limit control L = Display lower limit set point l = Display lower limit with deceleration calculation U = Display upper limit set point u = Display upper limit with deceleration calculation
Lower limit	Lower motion limit Only applies if parameter 1 = "Y" OR "y"	-2,147,483,646 to 2,147,483,646
Upper limit	Upper motion limit Only applies if parameter 1 = "Y" or "y" Note: A comma separates the lower and upper limits	-2,147,483,646 to 2,147,483,646

Example:

paosy100,50000<CR>
Set lower limit to 100 and upper limit to 50,000

paosn<CR>
Turn off limit control.


NOTE

This command is only available if the RTOS option is installed.

IMPORTANT

The lower limit must always be smaller then the upper limit.

(T)rip

Description: Allows the user to use any of the USER Inputs to stop motor movement.

Activation of this command for motor abort (similar to home and limit inputs) is NOT dependent on motor direction.

When this command is activated and a trip occurs, all motor movement aborts (*). All command processing proceeds as normal.

Parameters:

Parameter	Description	Values
Enable Trip	Enables or disables the trip option.	Y = Enable N = Disable
Input Port	Specifies the User Input port to use for the trip command. Only applies if Enable Trip = 'Y'	See Table 13
Input Port State	Specifies the state that activates the trip command. Only applies if Enable Trip = 'Y'	0 = Low 1 = High

Example: ***pa***TY4,1<CR>

NOTE

This command can only be used if the RTOS option is Enabled.

(I) Motor Interlock

Description: Allows the user to specify an input and input state which will lock out all motor movement if the Interlock is enabled.

Activation of this command for motor abort (similar to home and limit inputs) is NOT dependent on motor direction.

When this command is activated and a Interlock condition occurs, all motor movement aborts (*). All command processing proceeds as normal.

Parameters:

Parameter	Description	Values
'p'	Set Interlock Input Port Selection. See Note below.	4 – 9
's'	Set Interlock Active State	0 or 1
'E'	Enable Interlock System	
'e'	Set Interlock to Active (Sent by the Host to perform an Interlock)	
'd'	Clear Interlock Active state (Sent from the Host to clear Interlock state from 'e' command)	
'D'	Set Interlock delay value in milliseconds	
'T'	Set Interlock timer in milliseconds	
't'	Set Interlock time that determines how long an interlock must be active before accepted as on.	
'A'	Enable/Disables the Interlock Auto-correction mode.	

Example: ***pa***lp0<CR>

NOTE

Input ports are 4 - User 1
5 - User 2
6 - User 3
7 - Spare 1
8 - Spare 2
9 - Spare 3

#(V)oltage readings command

Description: Allows the user to see the POWER input voltage in floating point display.

Parameters:

Type	Description	Parameters
Display Power	Displays the current input Power reading	#VD
Display maximum	Displays the maximum reading recorded since powering up.	#VX
Display minimum	Displays the minimum reading recorded since powering up.	#VM
Display maximum running	Displays the maximum reading recorded while running the motor.	#Vx
Display minimum running	Displays the minimum reading recorded while running the motor.	#Vm
Reset all recordings	Resets all recorded values with the current Power input voltage.	#Vr

Example:

pa#Vx<CR>
Displays the maximum reading collected while the motor was running in double precision floating point.

pa#VD<CR>
Displays the current input voltage on the POWER connector (Pin 1) in double precision floating point.

o(T)emperature readings command

Description: Allows the user to see the temperature of each axis near the motor drive.

Parameters:

Type	Description	Parameters
Display Temperature	Displays the current axis temperature reading in degrees F	oTD
Display Temperature maximum	Displays the maximum reading recorded since powering up.	oTX
Display Temperature minimum	Displays the minimum reading recorded since powering up.	oTM
Reset all recordings	Resets all recorded values with the current temperature.	oTR

Example:

paoTX<CR>
Displays the maximum temperature reading collected since power up.

paoTD<CR>
Displays the current axis temperature reading in degrees F.

This page intentionally left blank.

CHAPTER 7: QUADRATURE ENCODER BOARD COMMANDS

This chapter describes commands that are specific to boards with a quadrature encoder interface:

All ARM Boards

Initialize the Axis Home Direction (qN)

Description: Initializes the axis for home in the clockwise or counterclockwise direction.

This command initializes the quadrature encoder interface:

- If '+' is specified all CW movement is decremented and all CCW movement is incremented.
- If '-' is specified all CCW movement is decremented and all CW movement is incremented.

The command allows the user to change the directional count of the encoder interface.

Parameters:

Parameter	Description	Values
Direction	Direction to initialize the axis	+ = Initialize the axis for Home direction in the CW direction - = Initialize the axis for Home direction in the CCW direction

Example: ***pa***qN+<*CR*>

Set Quadrature Encoder Mode (qm)

Description: Sets the quadrature encoder to a specific control mode.

- **O**ff: turns the quadrature encoder interface off (default on power up).
- **C**ount ONLY: puts the quadrature encoder interface into *count only* mode. This is just like the standard SSQE mode.
- **D**etect error: puts the quadrature encoder interface into *count with error detection* mode. The system performs as if in *count only* mode, unless the *motor movement tolerance* setting (see page 98) is trigged. Then the system aborts the motor movement and responds with an 'e'ncoder error.
- **A**uto-correction: puts the quadrature encoder interface into *auto-correction* mode. The system performs as if in *count with error detection* mode when running the motor. When the motor movement is complete, or if the motor movement was aborted because of an error, the system checks the motor position. If the correct motor position was not attained, the system applies additional motion commands until the correct position is achieved or the *auto-retry counter* is exhausted (see page 99).

  If the motor is not in motion, the *auto-correction* mode "servos" the current idle position when the *auto-correction tolerance* value is triggered.

Parameters:

Parameter	Description	Values
Mode	Quadrature encoder mode	O = Off mode (default at power up) C = Count only mode D = Detect error mode A = Auto-correction mode

Example: ***pa***qmD<*CR*>

NOTE See additional examples later in this section.

Closed Loop Feedback Control

The *auto-correction* and *count with error detection* modes assume that the stepping mode and the encoder tone wheel are at a one to one (1:1) ratio. The current software only performs small compensation (software two's compliment divider and quadrature encoder modes) for encoder tone wheel and stepping modes that are not 1:1.

For example: If the wheel is 100 CPR and the encoder counting mode is x4, you will receive 400 pulses per revolution. A 1.8 degree stepper motor in half step mode will produce 400 steps per revolution, which is a 1:1 ratio.

If the motor is switched to full step mode, this will produce 200 steps per revolution. This is a 2:1 ratio in which the current software (if the divider is set to 1), will compensate for and produce 200 steps per revolution.

If however you used a 150 CPR tone wheel, then an x1 counting mode would give you 150 pulses per revolution and x4 counting mode would give you 600 pulses per revolution. This cannot be compensated for on a standard 1.8 degree stepper/encoder ratio.

Since the *auto-correction* mode is dependent on knowing where the motor is going, *auto-correction* mode CANNOT be enabled prior to performing the motor axis "N+1" or "N-1" command. Only *off*, *count only,* and *count with error detection* modes are allowed.

If an encoder error occurs in *count with error detection* mode, or if the system is unable to correct an error in *auto-correction* mode, the system aborts the motor movement. No additional motor movements can be performed until the system has been re-initialized with the 'N' command followed by the 'E' command, or until the axis is given the correct position with the 'A' command. This clears the motor error and allows standard motor control.

Enable or disable the Quadrature Encoder Axis (q)

Description: Enables or disables the quadrature encoder interface.

Parameters:

Parameter	Description	Values
Interface	Enable or disable the interface.	E = Enable D = Disable

Example: ***pa***qE<*CR*>

Set QE Interface Count Mode (qM)

Description: Sets the count mode to either x1, x2 or x4.

The default power up state is x4, which allows a 4 times count instead of the usual 1 times counting mode.

Parameters:

Parameter	Description	Values
Mode	Counting mode.	0 = x1 1 = x4 2 = x2 3 = Clock (B) / Direction (A)

Example: ***pa***qM0<*CR*>

For example: if you have a 1.8 degree stepper running in half step mode (400 steps per revolution), the encoder has a 100 PPR (pulses per revolution) tone wheel. With the interface in x1 mode, this gives you 100 PPR output, but if the x4 mode was used it would give you 400 PPR, which is equal to the stepper in half step mode. It is easier to find a 100 PRR wheel over a 400 PPR wheel.

Get the Current Quadrature Encoder Position (qP)

Description: Displays the current encoder position in signed long integer format.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***qP<*CR*>

Set the Quadrature Encoder Multiplier Register (qp)

Description: Sets the encoder position multiplier register to the specified value. This will allow the user to perfectly scale the encoder to the mechanical control system. (Default = 1.0)

Parameters:

Parameter	Description	Values
Floating point	Multiplier	1.0 to xxxxx.xxxxx

Example: ***pa***qp<*CR*>

Set the Quadrature Encoder Divider Register (qd) (NOT USED ANYMORE)

Description: Sets the encoder position divider register to the specified value. Not used anymore but the command will be excepted. The divider values are 1 to 65534.

Set Motor Movement Tolerance (qT)

Description: Sets the motor movement tolerance threshold.

This threshold is used to detect errors while the motor is moving if *count with error detection* mode or *auto-correction* mode are enabled.

The motor movement is automatically aborted if the difference between the encoder position and the calculated motor position is greater than this threshold setting.

If *count with error detection* mode is active, an '**e**'ncoder error status is returned. If *auto-correction* mode is enabled, the system applies additional motion commands until the correct position is achieved or the *auto-retry counter* is exhausted.

Parameters:

Parameter	Description	Values
Tolerance	Tolerance value	0 to 65534 15 = default at power up

Set Auto-correction Tolerance (qa)

Description: Sets the auto-correction tolerance threshold.

This threshold is used to correct errors for a stopped motor. If at any time the difference between the encoder position and the calculated motor position is determined to be greater than this threshold setting the

system applies motion commands until the motor is back in position or the *auto-retry counter* is exhausted (see *Example 4 - Auto-correction side effect on page* 101).

Parameters:

Parameter	Description	Values
Tolerance	Tolerance value	0 to 65534 1 = default at power up

Set Auto-correction Retry Counter (qr)

Description: Sets the auto-correction retry counter, the maximum number of times an auto-correction movement should be attempted.

NOTE

This command will not be executed if the motor is running.

Parameters:

Parameter	Description	Values
Counter	Counter value	1 to 200 5 = default at power up

Examples Using Encoder Control Modes

Example 1 - Standard motor movement - *count with error checking* mode

1. Put each axis into idle power mode.
2. Home each axis as normal.
3. Send either the "**qN+**" or the "**qN-**" command to each axis to zero the counter.
4. Send a "**qT5**" command to each axis to set the standard motion tolerance to 5 steps.
5. Give each axis a "**qmD**" to enable count with error detection.
6. Give each axis the "**qE**" to enable the encoder.
7. Move the motor from home to point B (example: M10000 to move to position 10,000). When the motor is moving, stall the motor.
8. Ask for the current status of the axis (example for a SSXYZMicroLC board at address 0: X0<CR>). Response will be "x0e"<CR>.

Example 2 - Standard motor movement - *count only* mode

1. Put each axis into idle power mode.
2. Home each axis as normal.
3. Send either the "**qN+**" or the "**qN-**" command to each axis to zero the counter.
4. Send a "**qmC**" command to each axis to enable the "count only" mode.
5. Give each axis the "**qE**" to enable the encoder.
6. Move the motor from Home to point B (example: M10000 to move to position 10,000).
7. Ask for the current position status of the axis (example for a SSXYZMicroLC board at address 0: X0m<CR>). Response will be "x0>10000"<CR>.
8. Ask for the current encoder position status of the axis (example for a SSXYZMicroLC board at address 0:X0qP<CR>). Response will be "x0>10000"<CR>.

Example 3 - Standard motor movement - auto-correction mode

1. Put each axis into idle power mode.
2. Home each axis as normal.
3. Send either the "**qN+**" or the "**qN-**" command to each axis to zero the counter.
4. Send a "**qa5**" command to each axis to set the standard motion tolerance to 5 steps.
5. Send a "**qmA**" to enable auto-correction mode.
6. Give each axis the "**qE**" to enable the encoder.

7. Move the motor from Home to point B (example: M10000 to move to position 10,000). When the motor is moving, stall the motor.

8. The motor will stop and then restart to try to get to position 10,000 5 more times before erroring out.

Example 4 - Auto-correction side effect

Auto-correction mode has a side effect that customers may find useful. The system will servo at the current motor position just like a DC Servo running a PID loop. If the motor movement has completed and the user moves the motor shaft, the axis will automatically correct for the error. This would be useful in for example a pipette arm that a user accidentally hit when walking past the machine.

When the system is first powered up, the user should check to see if the encoders are connected correctly and that the board is receiving counts. Using the "**qN+**" or the "**qN-**" with the "**qE**" to enable the encoder, the user can use the "**qP**" command to see what the encoder value is when the motor is moved.

1. Put each axis into idle power mode.

2. Give the axis an "**N+0HL**" command (zero counter and ignore home and limit with no motor movement).

3. Send a "**qN+**" to initialize the quadrature encoder module.

4. Send a "**qmC**" to set for count only mode.

5. Send a "**qE**" to enable the encoder.

6. Move the motor to position 1000 ("**M1000**").

7. Ask for the current Quadrature encoder position ("**qP**").

If the number is negative, the initialization command for the Quadrature encoder module needs to be changed from a "**qN+**" to a "**qN-**". If the count is correct, you are ready to start running the board. If the count is wrong, check your wiring to make sure that both channels A and B are connected correctly and that there is power getting to the sensor array. If the count is a multiple of the CPR value of the purchased encoder, then either change the counting mode of the quadrature encoder interface (x1, x2 or x4) and/or use the built-in scaler to get the proper counts. Remember that the CPR value must be in multiples of the stepping value (degrees per step) to achieve a 1:1 stepper to encoder ratio:

1- 100 CPR encoder and a 1.8 degree stepper:

- Full Step = 200 steps per revolution
- Half Step = 400 steps per revolution
- x1 mode = 100 counts per revolution
- x2 mode = 200 counters per revolution
- x4 mode = 400 counts per revolution

This page intentionally left blank.

CHAPTER 8: COMMUNICATION COMMANDS

This chapter describes commands that affect the transmission speed, numeric format, and network configuration for communication between the host and Simple Step Controller board.

Set Communications Baud Rate (c)

Description: Changes the transmission speed.

The default baud rate is typically 57.6K baud on power up, although another rate may have been specified.

Parameters:

Parameter	Description	Values
Baud rate	Communications baud rate	0 = 9,600 1 = 19,200 2 = 38,400 3 = 57,600 4 = 115,200 5 = 230,400 6 = 480,600

Example: ***pa***c2<CR>
Changes the baud rate to 38.4K.

Before the baud rate is changed, an acknowledgement is sent that includes the current baud rate (e.g. a "d0" response from an SSMicroMC board acknowledges that the current baud rate is 19.2K).

Set Binary Transfer Option (ob)

Description: Turns binary transfer mode ON or OFF.

If binary mode is enabled, numeric parameters are transmitted in binary (rather than ASCII) format.

Binary mode saves significant processing time on the board and shortens the communication string length.

If a numeric value has a range of 0-255 (8 bit), then one (1) byte is expected. If the parameter has a range from 0-65535 (16 bit), then two (2) bytes are expected, and if the parameter is 32-bit, then four (4) bytes are expected.

Parameters:

Parameter	Description	Values
Binary mode	Binary communications option.	Y = Enable binary communications N = Disable binary communications

Example: ***pa***bY<CR>

NOTE

This command is a special order option.

Set Networking (on)

Description: Turns networking ON or OFF.

NOTE

This new option is included with software version 1.0.4.005 (sub-version 005) or greater.

By default, communications are performed using the Simple Step RS232 network with up to 16 Simple Step boards on one serial line. A board's address is determined by the open/closed condition of switches 1 through 4 (see page **Error! Bookmark not defined.**). Commands include a board type prefix ('p' in the example) and the board's address ('a' in the example). When networking is turned off the board's address character is eliminated from the command sequence.

Parameters:

Parameter	Description	Values
Networking mode	Networking option.	Y = Enable networking (default) N = Disable networking

Example: ***pa***onY<CR>

Set Radix Number (or)

Description: Selects decimal or hexadecimal mode.

All numeric entries and responses are in the selected mode.

Parameters:

Parameter	Description	Values
Radix number	Determines whether base 10 or base 16 are used for numeric values.	D = Decimal, base 10 (default) H = Hexadecimal, base 16

Example: ***pa***orH<CR>

Set Communication Type (oC)

Description: Determines whether an RS232 port or RS422/485 port is being used.

When an RS422/485 communications interface is ordered with the board, firmware version 105.xx and above can speed up the communication delay by 250 µsec per serial string response.

On power up, all firmware v105 and above assume that an RS232 serial communications network is available (except for SSNEMA17).

Parameters:

Parameter	Description	Values
Communication type	Communication type	Y = RS422/485 communications N = RS232 communications

Example: *pa*oCY<CR>

CHAPTER 9: IEEPROM COMMANDS

This chapter describes commands that can be used to program IEEPROM memory. It also describes the power-up configuration area that is available on all boards.

Process on power up

If the firmware finds a 'K' (run EEPROM) command followed by the address at location 1 in the EEPROM, firmware automatically runs the EEPROM contentsExample:

- 00001: K4<CR>
- 00004: o@LD<CR>

The "o@LD" command tells the system to initialize the axis for the old Simple Step movement control. On power up the firmware sees the K4 command and starts running the program. If no 'K' command is present, the system initializes as normal.

On power up, the board checks the EEPROM for either an ERASE or PROGRAMMED contents marker. If the EEPROM has data that is invalid, the system automatically erases the contents of the EEPROM. An erase delays the power up by an additional 5 seconds.

Program EEPROM (Q)

Description: Initiates programming mode.

The board sends back an “address:” string to signify that it is ready for the program. Each time a command string, terminated by a <CR> is entered, the board responds with the “address:” for the next programming command.

An escape key, 0x1B, followed by a carriage return <CR> terminates programming mode. The system confirms program termination with a status of ‘c’ to acknowledge programming is complete.

When programming is complete, the system stores a 0xAA character at the first location (0x0000) in memory to indicate that a program is stored.

Parameters:

Parameter	Description	Values
Starting address	Address where first instruction should be stored.	1 to EEPROM size

Example: ***pa***Q1<CR>
Start EEPROM programming mode, store first instruction at address 1.

Sample programming exchange:

Board Responds	User Enters	Function
	D0Q1	Initiates programming mode starting at address location 1 (assumes board type is SSQE (‘Q’) and board address is 0.
00001:	N+0	Initializes the board, without moving the motor, so that home is in the CW direction
00005:	E6000	Sets the end velocity to 6000 sps
00012:	B500	Sets the beginning velocity to 500 sps
00019:	S5	Sets the slope to 5
00022:	M9000	Moves the motor to position 9000
00028:	M0	Moves the motor to home
00031:	j22,3	Loop: Jump to address 22 and perform the commands at location 22 and 28 three times before continuing
00037:	M1000	Moves the motor to position 1000
00043:	M0	Moves the motor to home
00046:	j37,2	Loop: Jump to address 37 and perform the commands at location 37 and 43 two times before continuing
00052:	ESC	Terminates programming mode

d0c		Acknowledges program completion

Run EEPROM Contents (K)

Description: Runs a stored EEPROM program.

All boards will also flash the processor LED in a 500 ms on/off cycle to show that it is running a IEEPROM program.

The other way to execute a stored program is to issue the 'K' command via the communications interface. The Simple Step Controller Board responds:

- "d0r" if it is running the program
- "d0>" if there is no program residing at the specified address or there is an error in the program

Program execution continues until a 0x00 is found in EEPROM storage.

Parameters:

Parameter	Description	Values
Address	Starting address for the stored program.	1 to EEPROM size

Example: ***pa***K4<CR>
Run EEPROM contents starting at address 4 until a 0x00 is found.

Jump to EEPROM address (J)

Description: Programs a jump to the specified EEPROM address. Program execution continues from that location.

Parameters:

Parameter	Description	Values
Address	Jump to address.	1 to EEPROM size

Example: J10<CR>
Jump to EEPROM address 10 and start execution there.

Jump to EEPROM address *n* times (j)

Description: Programs a jump to the specified EEPROM address. Program execution continues from that location until the jump command is encountered again. The jump is repeated the indicated number of times before going to the next location in memory (see example on page 109).

Parameters:

Parameter	Description	Values
Address	Program's starting address.	1 to EEPROM size
Loop count	Number of times to perform the jump.	Loop range is from 1 to 255 (firmware version 106.40 now allows up to 65000 loops).

Example: j31,3<CR>
Jump to EEPROM address 31 and start execution there. Repeat this 3 times.

Erase EEPROM Contents (Z)

Description: Sets EEPROM address 0 to 0x00, indicating there are no valid programs stored in the EEPROM.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***Z<CR>

Unerase EEPROM Contents (u)

Description: Places a checksum value at address location 0 to allow recovery of stored EEPROM information.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***u<CR>

NOTE

This command is not supported in firmware versions 105.xx and above.

Abort Current IEEPROM Program (* or !)

Description: Aborts any currently running motor and EEPROM program.

Parameters:

Parameter	Description	Values
none		

Example: ***pa****<CR>

Dump EEPROM Contents (D)

Description: Initiates dump mode for the board.

All information, starting from the user specified address until a 0x00 delimiter is found, is uploaded to the host. The dump display is in the format “address:command” as shown in the example on page 108.

The dump terminates with a ‘c’ status indicator.

Parameters:

Parameter	Description	Values
Address	Starting address.	1 to EEPROM size

Example: ***pa***D12<CR>
Dump EEPROM contents starting at address 12, continue the dump until a 0x00 is found

Check Input Line and then jump to EEPROM address (L)

Description: Programs a conditional jump based on the state of the input channel.

Parameters:

Parameter	Description	Values
Input port	Input port to be checked.	See Table 13 on page 82.
Input port state	Input port state.	0 = Low 1= High
Address	Address to jump to if the input channel MATCHES the specified state.	0 = Perform the next sequential command 1 to EEPROM size
Address	Address to jump to if the input channel DOES NOT MATCH the specified state.	0 = Perform the next sequential command 1 to EEPROM size

Example: L4,0,10,100 <CR>
Check input channel 4 (User 1). If it is low (0) continue execution at address 10, otherwise continue execution at address 100.

Wait for Motion to Complete (W)

Description: Waits for motion to complete before going onto the next command in the IEEPROM.

This is most useful when the RTOS option is installed. Without this command the RTOS stacks the motion commands and allows the program to continue.

Parameters:

Parameter	Description	Values
none		

Example: W<CR>

Call Subroutine (k)

Description: Calls a subroutine in another portion of memory.

The subroutine should end with an 'i' command (return from call) so that the program returns to the next instruction after the call subroutine command


NOTE

Nested subroutine calls are not allowed.

Parameters:

Parameter	Description	Values
Address	Starting address for the subroutine	1 to EEPROM size

Example: k2<CR>
Run EEPROM contents starting at address 2.

Return From Subroutine (i)

Description: Returns from a subroutine call. Program execution continues where it left off before the jump to the subroutine.

Parameters:

Parameter	Description	Values
none		

Example: i<CR>

Text Display (t)

Description: Adds a text message to IEEPROM memory. The message is transferred to the host via the serial port.

Parameters:

Parameter	Description	Values
Text	Text string to be stored.	

Example: ***pa***tHello World/0x0A/0x0D<CR>

In the above example, a "/0x0A/0x0D" (Linefeed and Carriage Return) is the delimiter for the string. The <CR> is the delimiter for the EEPROM command and is not part of the text string.

The forward slash character ('/') is used to indicate that a hexadecimal value follows. The format for the special sequence is "/0x0hh" where the "hh" is a hexadecimal value from 00 to FF. This allows the user to enter control characters that are below the 0x20 hexadecimal space (' ') character.

Toggle Output Port Line (^)

Description: Toggles an output control line for fast switching speeds.

This command activates the output port to the specified port value, delays and then returns the output port back to the non-activation state.

This command can also be used for inter-axis control of other axes with the trigger commands that are explained below.

Parameters:

Parameter	Description	Values
Output Port	Output port number.	See Table 14 on page 83.
State	Output port activation state.	0 = Low 1 = High
Delay	Output port activation state delay	1-65534 milliseconds

Example: ***pa***^1,0,1300<CR>
Toggle output #1 from it current state to a 0 value, delay 1300 milliseconds and then reset back to 1.

Wait for Input Port Trigger (<)

Description: Waits for a change of state on an input port and then waits for the de-activation before jumping to the user specified address.


NOTE

Similar to a bit check command but in real-time.

Parameters:

Parameter	Description	Values
Input Port	Input port number.	See Table 13 on page 82.
State	Input port activation state.	0 = Low 1 = High
Address	Jump to address	0 = Perform the next sequential command 1 to EEPROM size

Example: ***pa***<4,0,100<CR>
Wait for Input #4 (User 1) to go low, then high and then jump to EEPROM address 100.

Wait for Input Port Change (=)

Description: Waits for a change of state on an input port and then jumps to the user specified address.

NOTE

Similar to a bit check command but in real-time.

Parameters:

Parameter	Description	Values
Input Port	Input port number.	See Table 13 on page 82.
State	Input port activation state.	0 = Low 1 = High
Address	Jump to address	0 = Perform the next sequential command 1 to EEPROM size

Example:

pa=4,0,100<CR>
Wait for Input #4 (User 1) to go low and then jump to EEPROM address 100.

CHAPTER 10: ADDITIONAL COMMANDS FOR ALL ARM BOARDS

All products of the Simple Step motor control boards utilizes NXP ARM 32-bit Cortic-M4 processors running at 72MHz and utilizes software settings for all configuration parameters.

Some features of the new ARM boards are the following:

- Power-up configuration registers allow the user to program power-up values such as board address, baudrate, communications prefix characters (both TXD and RXD), RTOS active or disabled, etc.
- An axis configuration storage area is used every time the board powers up for the first time. This area allows the user to configure the motor axis (power setting, top velocity, start velocity, slope, etc.).
- Features that are options for the XA product line are standard. These features include RTOS, 32 motor movement, etc.

This chapter describes the additional commands that are used to configure the Simple Step ARM boards, which is the first of these new products. Additional commands to display ARM boards and motor status information are also described. ***You can also program the ARM configuration by downloading from our websites Support, Download page the ARM Configuration utility.***

Programming ARM Power-up Configuration Parameters

The ARM boards have a programmable area in memory that can be used to store power-up configuration register values. This memory area has two (2) sections:

- The main section that stores communication and display values
- The motor section that stores motor power settings

When the board powers up, it checks the JP (JPX, JPY, JPZ) connector to see if Pin 3 is shorted to Pin 4. If the pin is shorted, the board ignores the stored configuration parameters and uses the default values.

If default settings are used, the LED flashes off/on for 300ms cycle time until power is recycled. After power-up the configuration settings can be reprogrammed.

Commands used to program the configuration areas are described starting on the next page.

NOTE

The default value for board prefix ‘X’ and default board address ‘0’ are used in all the examples.

NOTE

Both the board and axis configuration memory registers **share** the same flash memory so when the #CE or #AE commands are given, it will erase **both** configuration values and return them to their default states.

Setting the Main Configuration Registers

A series of #CS commands is used to set the main configuration registers:

Command	Description	Values	See Also Page
#CS0,<n>	Set radix number	D = Decimal, base 10 H = Hexadecimal, base 16	105
#CS1,<n>	Set the board's address	1-digit address: 0 to 15 2-digit address: 0 to 256	See #CS7 below
#CS2,<n>	Set communication baud rate	0 = 9,600 1 = 19,200 2 = 38,400 3 = 57,600 (default) 4 = 115,200 5 = 230,400 6 = 460,800	103
#CS3,<n>	Set LED display	0 = OFF 1 = On 2 = Short continuous	
#CS4,<n>	Set the receive and transmit prefix characters	Receive character (A-Z) Transmit character (a-z)	
#CS5,<n>	Enable or disable RTOS	E = Enable D = Disable	84
#CS7,<n>	Enable or disable extended addressing	E = Enable (use 2 digits for the board's address - allows the network to have up to 256 controllers on 1 communication line) D = Disable (use 1 digit)	35

Example: ***X0***#CS2,2<CR>
Sets the baud rate to 38.4K

Displaying the Main Configuration Register Values

Enter the ***X0*#CD<CR>** command to display the main configuration register values. The board responds with the following information:

Line	Displayed
1	CRC marker (This number should always read 21930)
2	Checksum value (65535=Not programmed, 0-65534 are programmed values)
3	Radix value (16 for HEX or 10 for DECIMAL)
4	Board address in DECIMAL (0-255)
5	Baud rate (0=9600, 1=19200, 2=38400, 3=57600, 4=115200, 5=230400, 6=480600)
6	LED startup value
7	Communications prefix character (TXD in decimal format)
8	Communications prefix character (RXD in decimal format)
9	RTOS active on power up (1=Enabled, 0=Disabled)
10	NOT USED
11	Extended Address enabled (1=Enabled, 0=Disabled)
12	-2147483648 (Indicates there are no more parameters to be displayed).

Each line is appended with <prefix><address><CR>

Storing Main Configuration Register Values in Memory

Follow the steps below to save information stored in the main configuration registers to memory.

1. ***X0*#CE<CR>**

   Erase the configuration area. The erase command must be issued before the area can be reprogrammed

   
   NOTE

   Both the main and the motor configuration areas of memory are erased.

2. ***X0*#CP<CR>**

   Store the main configuration register settings in memory.

Setting the Motor Configuration Registers

A series of #AS commands can be used to set the motor power-up configuration registers:

Command	Description	Values	See Also Page
#AS0,<n>	Set beginning velocity	0 to 13000	70
#AS1,<n>	Set end velocity	1 to 20000	71
#AS2,<n>	Set slope value	0= OFF 1 to 200	72
#AS3,<n>	Set low software limit	-2147483647 to 2147483647	87
#AS4,<n>	Set high software limit	-2147483647 to 2147483647	87
#AS5,<c>	Enable or disable software limit control	E= Enable D= Disable	87
#AS6,<n>	Set JOG lower software limit	-2147483647 to 2147483647	77
#AS7,<n>	Set JOG upper software limit	-2147483647 to 2147483647	77
#AS8,<n>	Set JOG beginning velocity	0 to 13000	77
#AS9,<n>	Set JOG end velocity	1 to 20000	77
#AS10,<n>	Set JOG slope	0= OFF 1 to 200	77
#AS11,<n>	Set JOG HOME input channel	1 to 4	77
#AS12,<n>	Set JOG LIMIT input channel	1 to 4	77
#AS13,<n>	Set motor maximum current	1 to 255	59
#AS14,<n>	Set motor idle current	0= OFF 1 to 255	59
#AS15,<n>	Set motor decay current (N/A)	0 to 255	59
#AS16,<n>	Set motor power off delay	0 to 65534 (incremental value of 1ms per increment)	76
#AS17,<n>	Set prescaler option	0 to 255	73
#AS18,<n>	Set motor stepping mode	0= FULL 1= 1/2 2= 1/4 3= 1/8 4= 1/16	75
#AS19,<c>	Change the motor movement from linear to old style motor movements	E= Enable (linear) D= Disable (old style)	46

Example: ***X0***#AS0,5000<CR>
Set the beginning velocity to 5000

Displaying Motor Configuration Register Values

Enter the ***X0*#AD<CR>** command to display the motor power-up configuration register values. The board responds with the following information.

Line	Displayed
1	CRC marker (This number should always read 21930)
2	Checksum value (65535=Not programmed, 0-65534 are programmed values)
3	Beginning velocity
4	End velocity
5	Slope
6	Low soft limit (only available if RTOS is enabled)
7	High soft limit (only available if RTOS is enabled)
8	Soft limit active
9	JOG low limit
10	JOG high limit
11	JOG beginning velocity
12	JOG end velocity
13	JOG slope
14	JOG Home input channel
15	JOG Limit input channel
16	Motor current setting
17	Idle current setting
18	Decay setting (N/A)
19	Motor power down delay value (in 10ms increments)
20	Prescaler value
21	Motor stepping mode (0=FULL, 1=Half step, 2=1/4 step, 3=1/8 step, 4=1/16 step)
22	Linear motor movement flag (1-Linear motor movement, 0=Old motor movement)
23	-2147483648 (Indicates there are no more parameters to be displayed).

Each line is appended with <prefix><address><CR>

Storing Motor Configuration Register Values in Memory

Follow the steps below to save information stored in the motor configuration registers to memory.

1. ***X0*#AE<CR>**

   Erase the configuration area. The erase command must be issued before the area can be reprogrammed

   
   NOTE

   Both the main and the motor configuration areas of memory are erased.

2. ***X0*#AP<CR>**

   Store the motor configuration register settings in memory.

Displaying the ARM Board Revision and Serial Number

Enter the ***X0*#BD<CR>** command to display the board revision and serial number. The board responds with the following information.

Line	Displayed
1	CRC marker (This number should always read 21930)
2	Checksum value (65535=Not programmed, 0-65534 are programmed values)
3	Board revision level
4	Board serial number
5	IEEPROM memory size
6	FACTORY SETTING
7	-2147483648 (Indicates there are no more parameters to be displayed).

Each line is appended with <prefix><address><CR>

This page intentionally left blank.

CHAPTER 11: ADC COMMANDS

This chapter describes Analog to Digital Converter add-on board commands. The built in 12-bit successive approximation analog to digital converter only has 1 ADC channel that can be used. This is SPARE 1 input. This input has no filtering or pull-ups on the board. The user can either create their own interface or use the SSADC-1x-4x interface board. Input is from 0 – 3.3VDC maximum. The SSADC-1x-4x board can be switched from 0-3.3VDC input to 0-5.0VDC input with 1K or 10K RC filter (all switch selectable).

Initialize ADC System (o@aI)

Description: Initializes the system and changes the general I/O lines to ADC inputs (1 channel ONLY and a SSADC-1A-4x must be installed).

This command must be the first command issued after power up, before any ADC readings take place (uppercase i).

Parameters:

Parameter	Description	Values
none		

Example: ***pa***o@aI<CR>

ADC System Check (o@a?)

Description: Retrieves initialization status.

The axis responds with the board prefix, board address, status and ‘T’rue or ‘F’alse. If ‘F’alse is returned, the board must be initialized, ‘I’, before any ADC operations are performed.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***o@a?<CR>

Deactivate ADC System (o@aD)

Description: Deactivates the ADC system and changes the I/O lines back to general I/O.

Parameters:

Parameter	Description	Values
none		

Example: ***pa***o@aD<CR>

Read ADC (o@aR)

Description: Reads the specified channel.

The axis responds with the board prefix, board address, status and a number from 0 to 4095.

Parameters:

Parameter	Description	Values
Channel	Channel to read.	0 or 1

Example: ***pa***o@aR0<CR>
Read and display Channel 0 data

Set ADC Collection Parameters (o@aS)

Description: Sets parameters used to collect ADC information.

Parameters:

Parameter	Description	Values
Channel	0 or 1	Set channel to change
Parameter	Parameter to set.	a = Sets the average data collection for all read ADC commands issued by the host. On power up the default is 1 (1 sample). The range is from 1 to 254. Every time an ADC read is performed, it collects and averages the raw data based on this value. z = Zero offset of all raw data before any averaging is performed. The parameter range is from 0 (default) to 65534 n = Set a noise mask to the raw data before the zero offset and averaging is performed. The range for this command is from 0 to FFFF (65534, the default on power-up).
Value	Parameter setting	Specific to each parameter, see ranges above.

Example: ***pa***o@aSn0,FFFF<CR> *Set channel 0 noise mask to 0xFFFF.*

pao@aSa1,2<CR> *Set channel 1 average sampling to 2 samples per ADC Read.*

pao@aSz0,0<CR> *Set channel 0 zero offset to 0.*

Typical ADC Read sequence

1. 32-bit data sum register is set to 0.
2. ADC Channel is chosen and saved as the currently selected channel active.

3. If the new ADC channel does not equal the current ADC channel and the board is a Revision A board, then the axis delays by the ‘d’ variable.

4. Collect 12 bits of data from ADC and store into temporary data variable.

5. The temporary data is now ANDed with the noise mask and the result stored back into the temporary data register.

6. If temporary data is less than the zero-offset variable, then the data is set to the zero-offset value. If the data is greater than or equal to the zero-offset value, then the zero offset value is subtracted from the raw data.

7. The temporary data is now added to the 32-bit sum register.

8. If the average sample value is greater than 1, then we loop to step #4 until all of the samples are collected.

9. We divide the 32 -bit average sum register with the average register setting and return that value back to the host.

IO Connector	Description
Pin 21	ADC Channel 0 Input
Pin 22	ADC Ground

Table 16 ADC Input Connectors

This page intentionally left blank.

CHAPTER 12: PWM COMMANDS

This chapter describes how to enable, disable and control the built in PWM system.

There can be up to 3 channels that can be used. Each channel can have its own rate. The first channel has 3 I/O lines that can be set as well as the second channel. Channel 3 only has 1 I/O that can be used as shown below (depending on board type).

Channel Group 0:
Output 0, Output 1 and User 1
Channel Group 1:
User 2, User 3 and Spare 2
Channel Group 2:
Spare 3

Each IO that is converted to a PWM output can have its own duty cycle which can be changed at any time. Each Channel Group can have its own rate but can only be programmed once after power up. If a new rate is required then power must be recycled or all that channel groups PWM outputs be changed back to an standard I/O.

Output 0 and Output 1 are open collector output only I/O that can run up to 0.5 amps per channel and up to 50VDC

User 1, User 2, User 3, Spare 2 and Spare 3 are 0 to 3.3VDC output that has up to 1ma output current.

When the user disables the PWM, all output only lines go back to being output only and set to their respective disabled state. All I/O lines that are converted back to an I/O, will be reset to an input line.

Convert I/O to PWM (pEc,ddddd,pppp)

Description: Converts the I/O port to a PWM port.

This command must be the first command issued after power up before any PWM duty cycle can be issued. If that channels group's rate has already been set, then any subsequent commands to that channels group rate will be ignored.

Parameters:

Parameter	Description	Values
c	Channel	0 – Output 0 1 – Output 1 2 – User 1 3 – User 2 4 – User 3 5 – Spare 2 6 – Spare 3
ddddd	PWM rate	100 – 30000Hz
pppp	Initial PWM duty cycle	0 - ((72000000 / PWM rate) - 1)

Example: ***pa*** pE1,10000,0 <CR>

Convert PWM back to I/O (pGc)

Description: Converts the PWM port to an output or I/O port.

Parameters:

Parameter	Description	Values
c	Channel	0 – Output 0 1 – Output 1 2 – User 1 3 – User 2 4 – User 3 5 – Spare 2 6 – Spare 3

Example: ***pa*** pG1 <CR>

Change PWM Duty Cycle (pFc,pppp)

Description: Changes the duty cycle of the channel

This command can be issued at any time after the above command is issued.

Parameters:

Parameter	Description	Values
c	Channel	0 – Output 0 1 – Output 1 2 – User 1 3 – User 2 4 – User 3 5 – Spare 2 6 – Spare 3
pppp	New PWM duty cycle	0 - ((72000000 / PWM rate) - 1)

Example: ***pa*** pF0,3000<CR>

Display current PWM rate (pHc)

Description: Displays the current channels PWM rate.

Parameters:

Parameter	Description	Values
c	Channel	0 – Output 0 1 – Output 1 2 – User 1 3 – User 2 4 – User 3 5 – Spare 2 6 – Spare 3

Example: ***pa*** pH3 <CR>

This page intentionally left blank.

CHAPTER 13: TROUBLESHOOTING

This chapter describes procedures for troubleshooting Simple Step boards.

The Windows HyperTerminal program can be used to troubleshoot the connection between the Simple Step board and the host (PC).

NOTE

A new version of HyperTerminal is included on the CD-ROM that comes with Simple Step for Windows. You can also download a free copy of HyperTerminal Private Edition *from the Hilgraeve's Web site (www.hilgraeve.com). If your HyperTerminal program is dated before 1999, an upgrade is necessary.*

Make the Connection

Basic operation of the Simple Step board needs only a power supply (see *Connecting a Simple Step Controller Board to a Power Supply on* 32) and a communications connection between J2 and the host (see *Connecting the Simple Step Controller Board to the Host* on page 34).

NOTE

No motors, home, and/or limit sensors are required to run the communications test.

Verify the Communication Settings

1. Select **Start** > **Programs** > **Simple Step for Windows** > **Simple Step Boards** from the Windows task bar (see page 34 for instructions to install SSWin).

   This starts HyperTerminal and loads the parameter file that was configured to allow communications with your Simple Step board. Use the settings in the figures below to verify proper configuration of the HyperTerminal connection.

   NOTE

   Change the settings if necessary. If you change a setting, click the **Disconnect** button and then click the **Call** button to reinitialize HyperTerminal with the new setting.

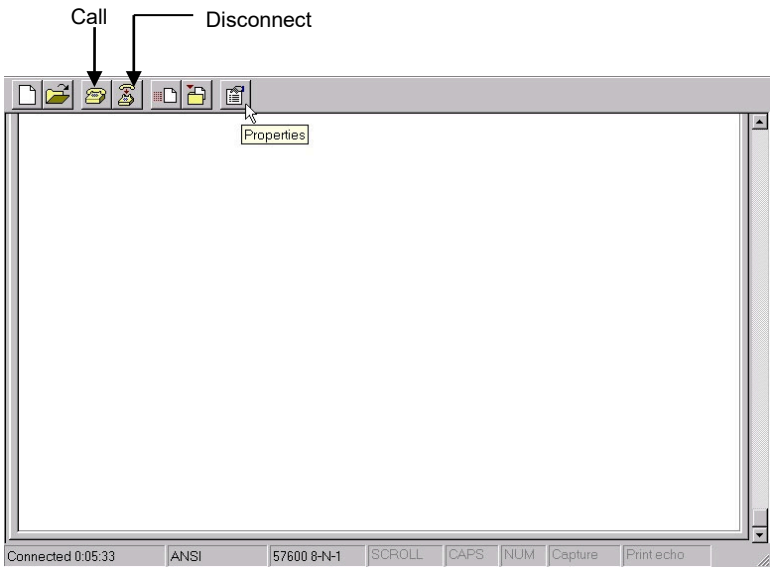


Figure 18 HyperTerminal Main Window

2. Click the **Properties** button and examine the *Connect To* settings.

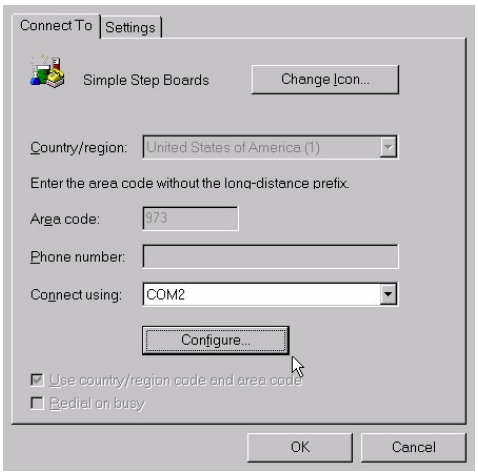


Figure 19 Connect To Tab

3. Verify that the correct COM port is displayed in the *Connect using* text box. This should be the COM port on the host that is connected to the board's J2 connector.

4. Click the **Configure…** button and examine the *Port Settings*.

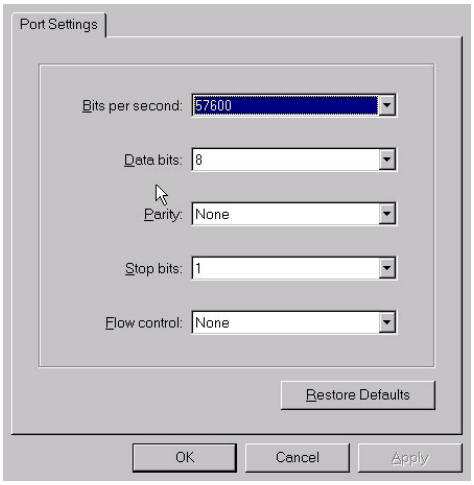


Figure 20 Port Settings Tab Dialog Box

NOTE

Bits per second will vary depending on the baud rate specified at the time of purchase. Baud rate is usually 57.6K, but can be changed at anytime after initialization of the board.

5. Close the *Port Settings* dialog box.

6. Display the *Settings* tab and verify ANSI emulation is selected.

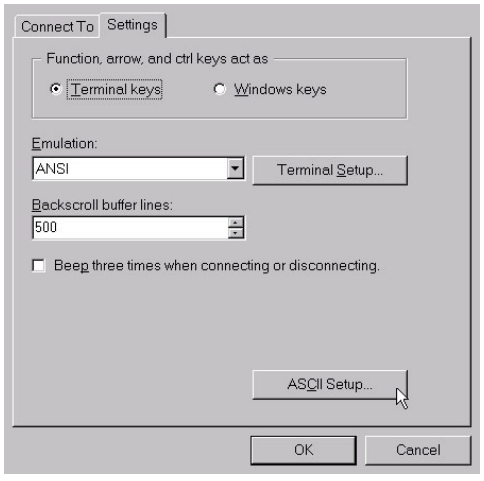


Figure 21 Settings Tab

7. Click the **ASCII Setup…** button and verify the correct ASCII Sending settings.

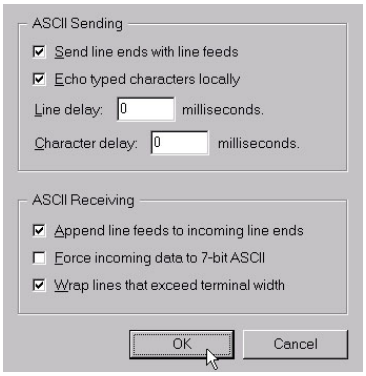


Figure 22 ASCII Sending Settings

Debug the Connection

If the HyperTerminal settings are correct and the board is still not communicating, try the tests described below.

Test 1:

a. Disconnect the J2 connector on the Simple Step board and short pins 2 and 3 together.

b. Type the letter ‘a’.

You should see another letter ‘a’ appear next to the one you just typed: “aa”.

c. Now type the letter ‘b’.

You should see a second ‘b’ echoed on the screen. Your display should look like this: “aabb”.

If you do not see “aabb”, then one of two (2) things is wrong:

- The cable connecting the PC to the Simple Step board is wired incorrectly (see page 34).
- The communications port setup is incorrect. Most PCs use COM1 but a few may use COM2. Neither port may be available on your PC if you are using a standard serial mouse and have a modem built into the PC.

Test 2:

a. Unshort the J2 connector.

b. Type the letter ‘a’.

c. This time only one ‘a’ should be displayed. If ‘a’ is not displayed, there is a short in the communications cable.

Test 3:

d. Connect the communications cable back to the board's J2 connector.

e. Apply power and using a voltmeter, connect pin #1 to ground, and measure the voltage at pins #2 and #3.

- Pin 2, host RxD to board TxD should read 0.0 +/- 0.1 volt DC
- Pin 3, host TxD to board RxD should read -5.0 to -12.0 +/- 0.5 volts DC

If you measure the pin #3 voltage on pin #2, then your TxD and RxD lines are swapped.

Test 4

a. Check the DIP Switch setting to determine the board address (see page **Error! Bookmark not defined.**). The SSNEMA17 and SSXYMicro boards use an software programmable address setting (see page 117).

b. Turn on the board's power supply.

c. Type the command that is appropriate to your board type:

Board	User Types	Board Responds
SSMicroMC	D0<Enter>	d0>
SSMicroLC	U0<Enter>	u0>
SSXYMicroLC SSXYMicroMC SSXYZMicroLC SSXYZMicroMC	X0<Enter>	x0>

Mac Users

The Mac uses the RS423 standard, which has the same electrical characteristics as RS232. RS423 allows higher baud rates and longer line lengths by decreasing the +/- 12 volts signals to a +/- 5 volt level. An RS423 connection is usually acceptable, but an RS423 to RS232 converter may be required.

This page intentionally left blank.

APPENDIX A: RECOMMENDED POWER SUPPLIES AND CONNECTORS

Board	Description	Digi-Key
SSMicroLC	3.5 AMP, 24 VOLT	RPS-120-40
SSMicroMC	6.75 AMP, 24 VOLT	9925340024
SSXYMicroLC	6.5 AMP, 24 VOLT	PMT-24V150W1AA
SSXYMicroMC	13.0 AMP, 24 VOLT	SP-320-24
SSXYZMicroLC	9.5 AMP, 24 VOLT	TXL 230-24S
SSXYZMicroMC	19.5 AMP, 24 VOLT	DRB480241

Table 17 Recommended Power Supplies

Board	Conn.	Manufac./PN
SSMicroLC	POWER	JST Sales / VHR-2N
SSMicroLC	MOTOR	JST Sales / VHR-5N
SSMicroMC	POWER	JST Sales / VHR-2N
SSMicroMC	MOTOR	JST Sales / VHR-5N
SSXYMicroLC	POWER	JST Sales / VHR-2N
SSXYMicroLC	XMOTOR, YMOTOR	JST Sales / VHR-5N
SSXYMicroMC	POWER	JST Sales / VHR-2N
SSXYMicroMC	XMOTOR, YMOTOR	JST Sales / VHR-5N
SSXYZMicroLC	POWER	JST Sales / VHR-2N
SSXYZMicroLC	XMOTOR, YMOTOR, ZMOTOR	JST Sales / VHR-5N
SSXYZMicroMC	POWER	Phoenix Contact / 1754449
SSXYZMicroMC	XMOTOR, YMOTOR, ZMOTOR	JST Sales / VHR-5N

Table 18 Mating Connector Guide Outline

Board	Conn.	Manufac./PN
All Boards	IO	Molex / 501646-2800
CBRS422-485-4x	SCOMM1, SCOMM2	Molex / 22-01-2057
CBRS232-4x	SCOMM1, SCOMM2	Molex / 22-01-2037

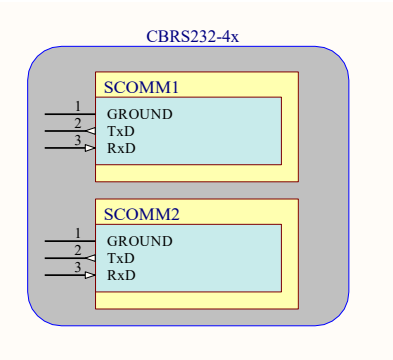
Table 19 Mating Connector Guide Outline

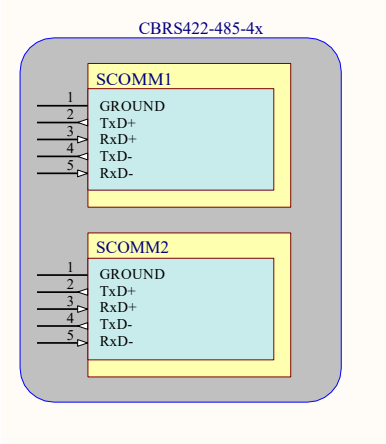
MOLEX CONTACTS: MOLEX KK# / 08-50-0114

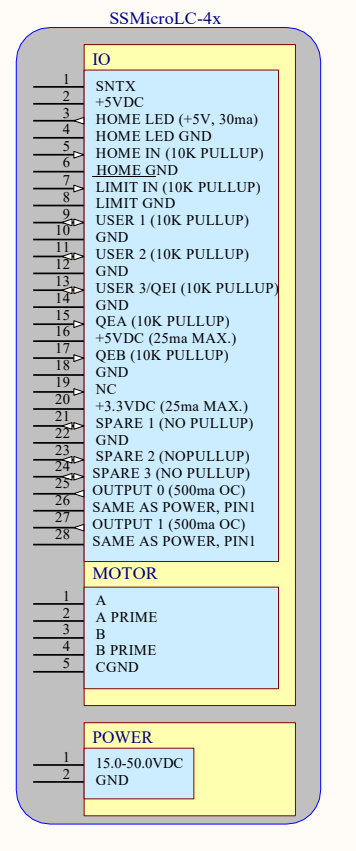
Connector	Mating Connector Part No.
J1	501646-1400
J2	51021-0800
JR	51021-0900
ENC	51021-0500

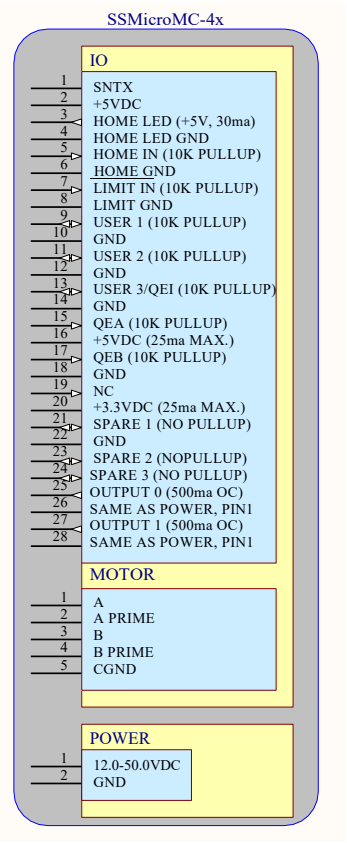
Table 20 Connector Breakdown – SSNEMA17 Boards

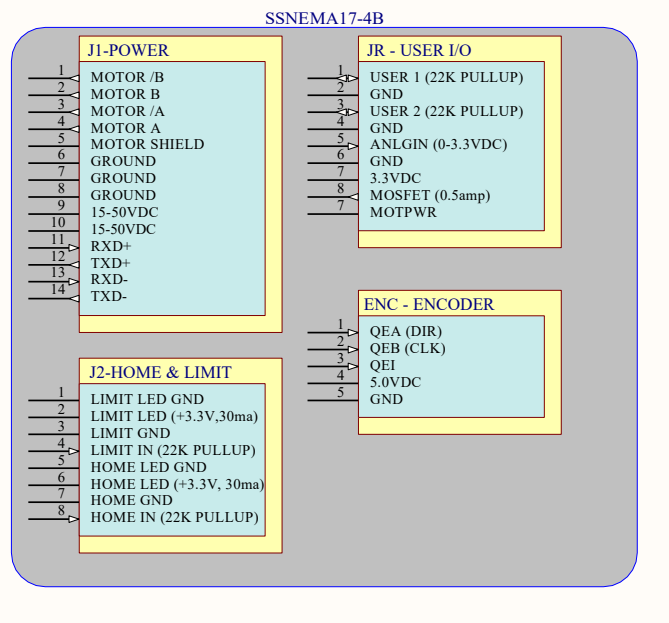
APPENDIX B: BOARD PINOUTS

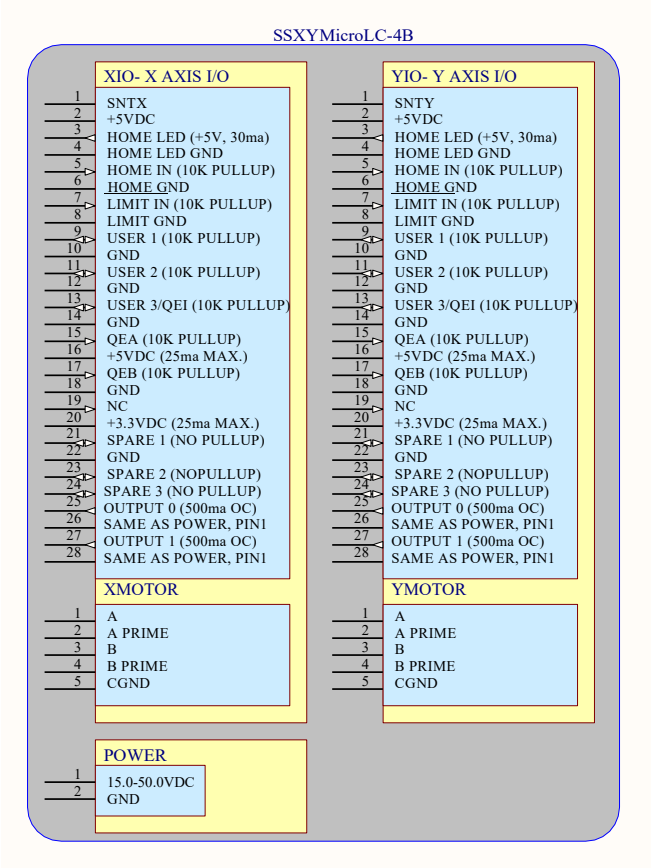


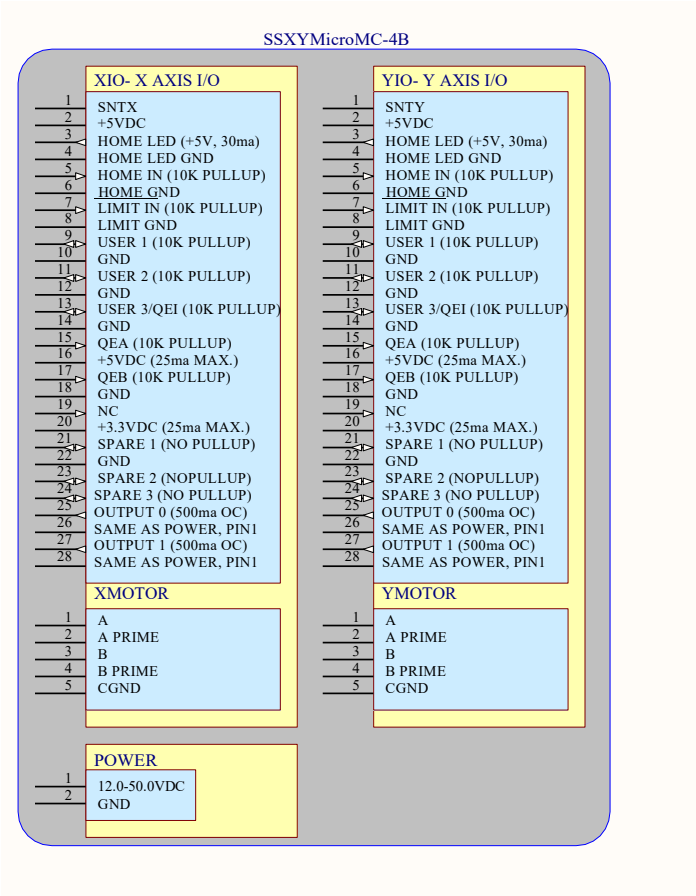


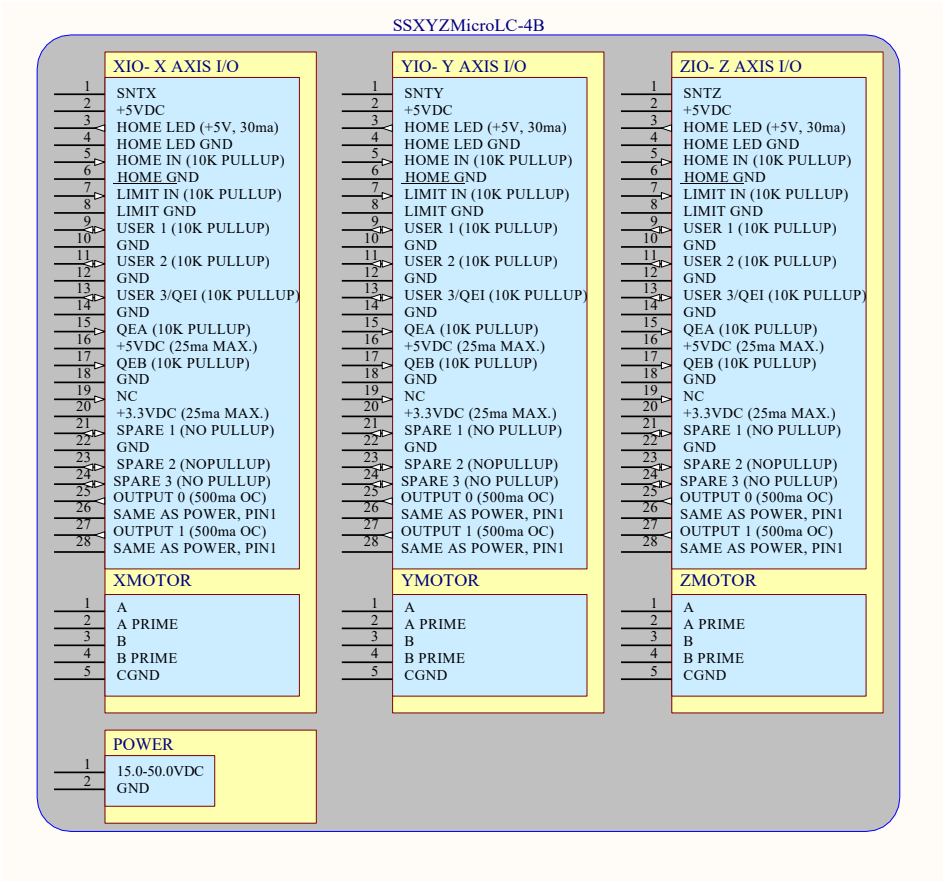

SSMicroLC-4x
IO
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
MOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 15.0-50.0VDC
2 GND

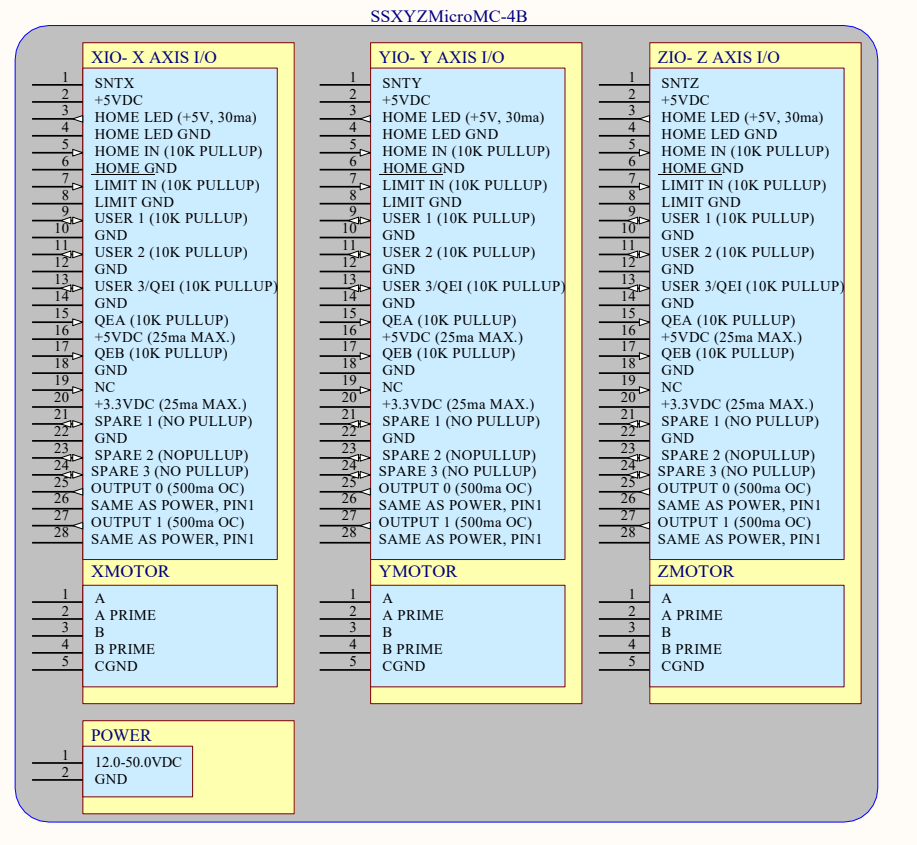

SSMicroMC-4x
IO
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
MOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 12.0-50.0VDC
2 GND


SSNEMA17-4B
J1-POWER
1 MOTOR /B
2 MOTOR B
3 MOTOR /A
4 MOTOR A
5 MOTOR SHIELD
6 GROUND
7 GROUND
8 GROUND
9 15-50VDC
10 15-50VDC
11 RXD+
12 TXD+
13 RXD-
14 TXD-
JR - USER I/O
1 USER 1 (22K PULLUP)
2 GND
3 USER 2 (22K PULLUP)
4 GND
5 ANLGIN (0-3.3VDC)
6 GND
7 3.3VDC
8 MOSFET (0.5amp)
7 MOTPWR
ENC - ENCODER
1 QEA (DIR)
2 QEB (CLK)
3 QEI
4 5.0VDC
5 GND
J2-HOME & LIMIT
1 LIMIT LED GND
2 LIMIT LED (+3.3V,30ma)
3 LIMIT GND
4 LIMIT IN (22K PULLUP)
5 HOME LED GND
6 HOME LED (+3.3V, 30ma)
7 HOME GND
8 HOME IN (22K PULLUP)


SSXYMicroLC-4B
XIO- X AXIS I/O
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
XMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
YIO- Y AXIS I/O
1 SNTY
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
YMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 15.0-50.0VDC
2 GND


SSXYMicroMC-4B
XIO- X AXIS I/O
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
YIO- Y AXIS I/O
1 SNTY
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
XMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
YMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 12.0-50.0VDC
2 GND


SSXYZMicroLC-4B
XIO- X AXIS I/O
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
XMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
YIO- Y AXIS I/O
1 SNTY
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
YMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
ZIO- Z AXIS I/O
1 SNTZ
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
ZMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 15.0-50.0VDC
2 GND


SSXYZMicroMC-4B
XIO- X AXIS I/O
1 SNTX
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
XMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
YIO- Y AXIS I/O
1 SNTY
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
YMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
ZIO- Z AXIS I/O
1 SNTZ
2 +5VDC
3 HOME LED (+5V, 30ma)
4 HOME LED GND
5 HOME IN (10K PULLUP)
6 HOME GND
7 LIMIT IN (10K PULLUP)
8 LIMIT GND
9 USER 1 (10K PULLUP)
10 GND
11 USER 2 (10K PULLUP)
12 GND
13 USER 3/QEI (10K PULLUP)
14 GND
15 QEA (10K PULLUP)
16 +5VDC (25ma MAX.)
17 QEB (10K PULLUP)
18 GND
19 NC
20 +3.3VDC (25ma MAX.)
21 SPARE 1 (NO PULLUP)
22 GND
23 SPARE 2 (NOPULLUP)
24 SPARE 3 (NO PULLUP)
25 OUTPUT 0 (500ma OC)
26 SAME AS POWER, PIN1
27 OUTPUT 1 (500ma OC)
28 SAME AS POWER, PIN1
ZMOTOR
1 A
2 A PRIME
3 B
4 B PRIME
5 CGND
POWER
1 12.0-50.0VDC
2 GND

APPENDIX C: BOARD CONNECTOR LOCATIONS

SSMicroLC Board Connector Locations

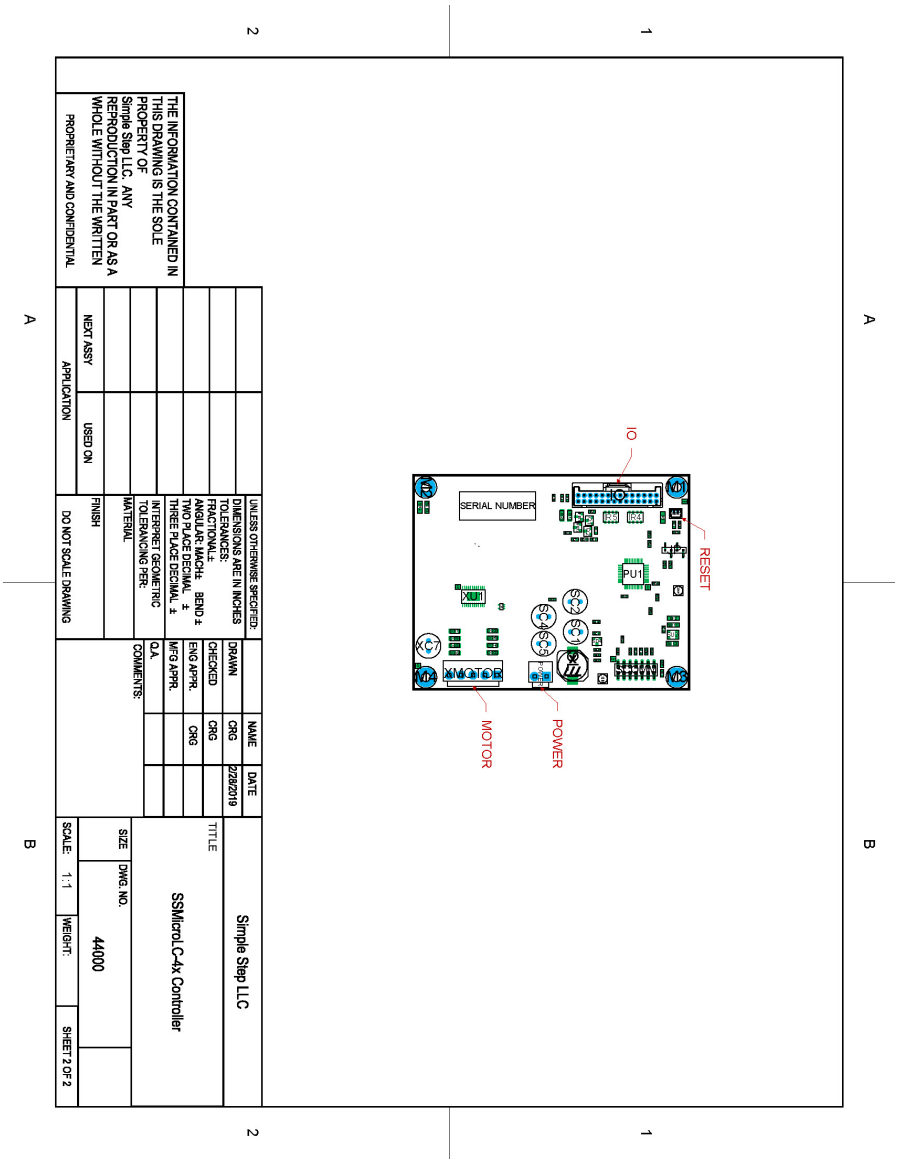


Figure 23 SSMicroLC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSMicroMC Board Connector Locations

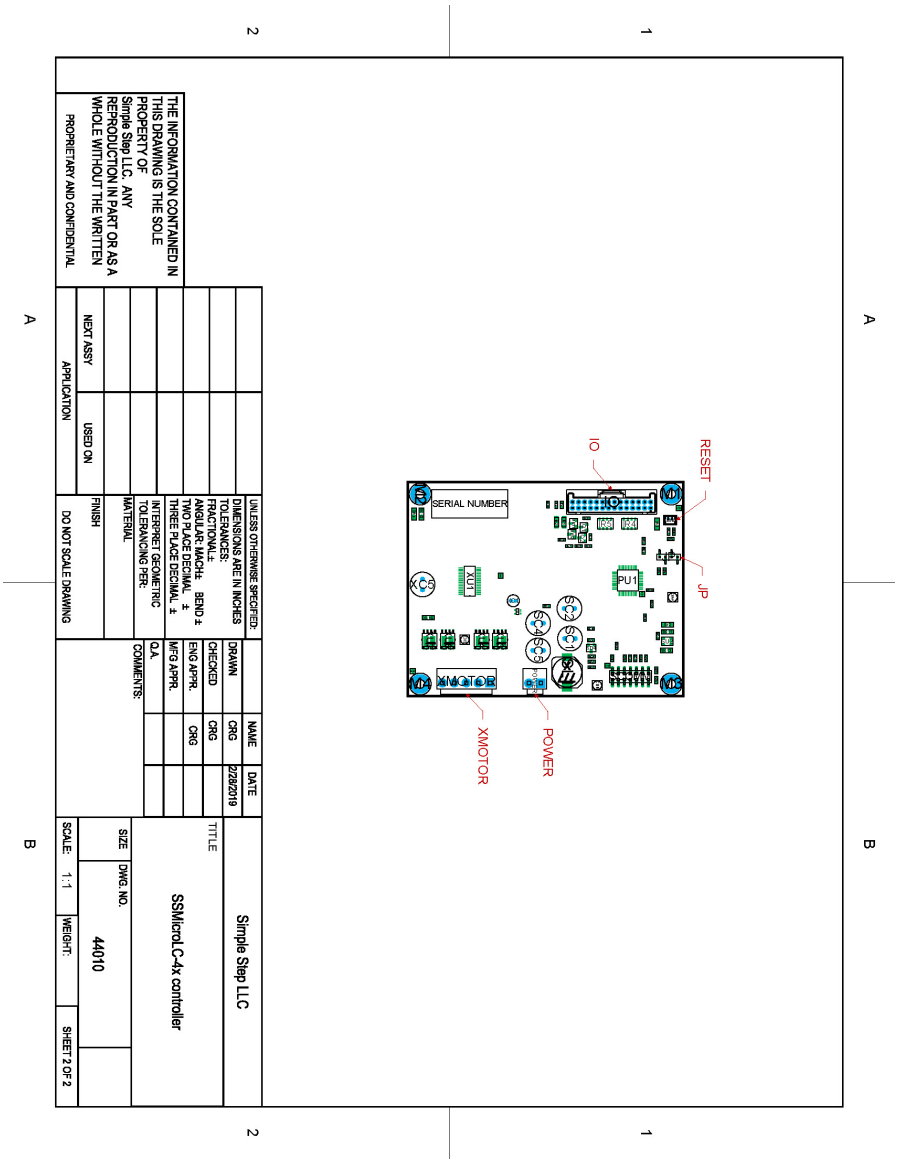


Figure 24 SSMicroMC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSXYMicroLC Board Connector Locations

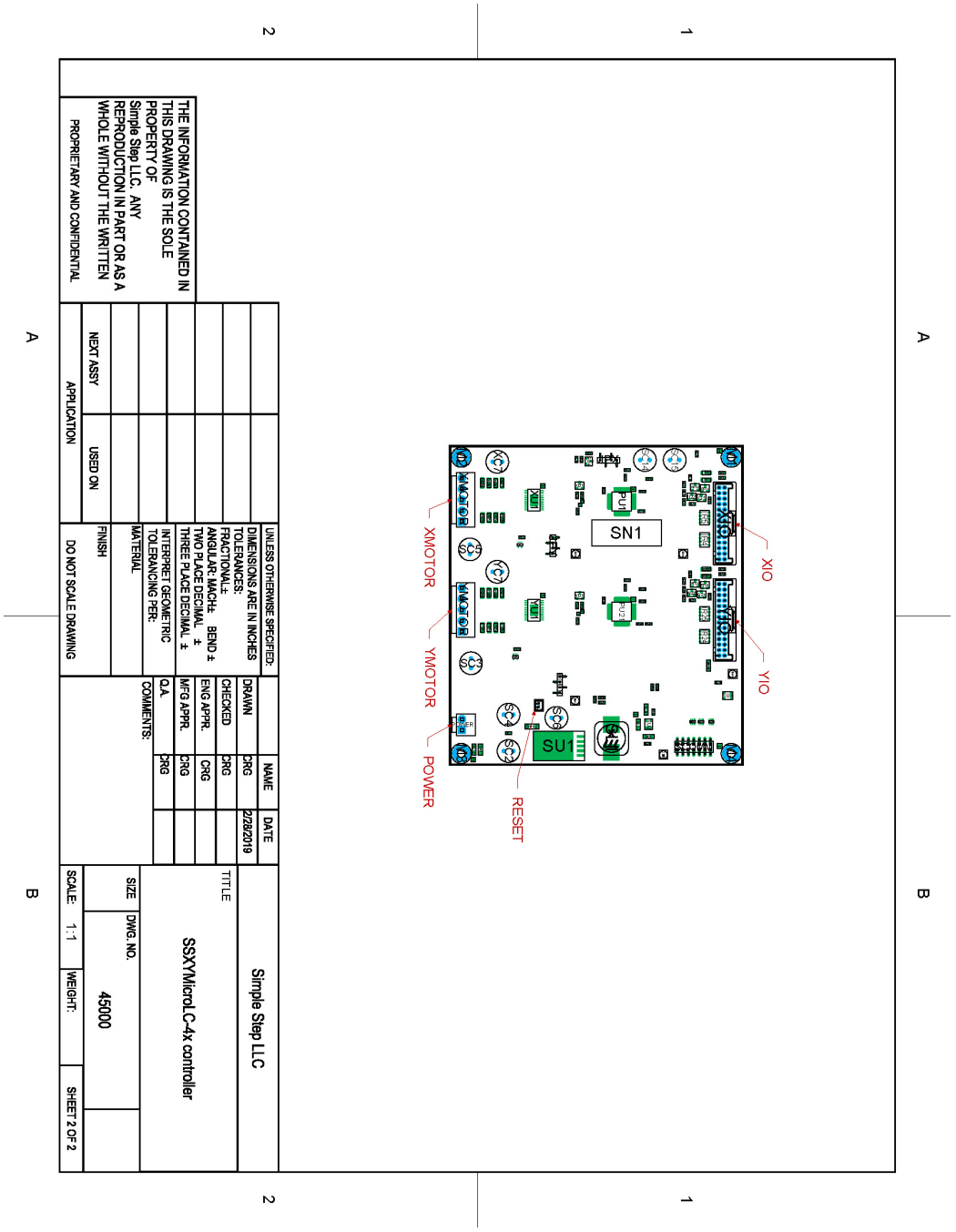


Figure 25 SSXYMicroLC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSXYMicroMC Board Connector Locations

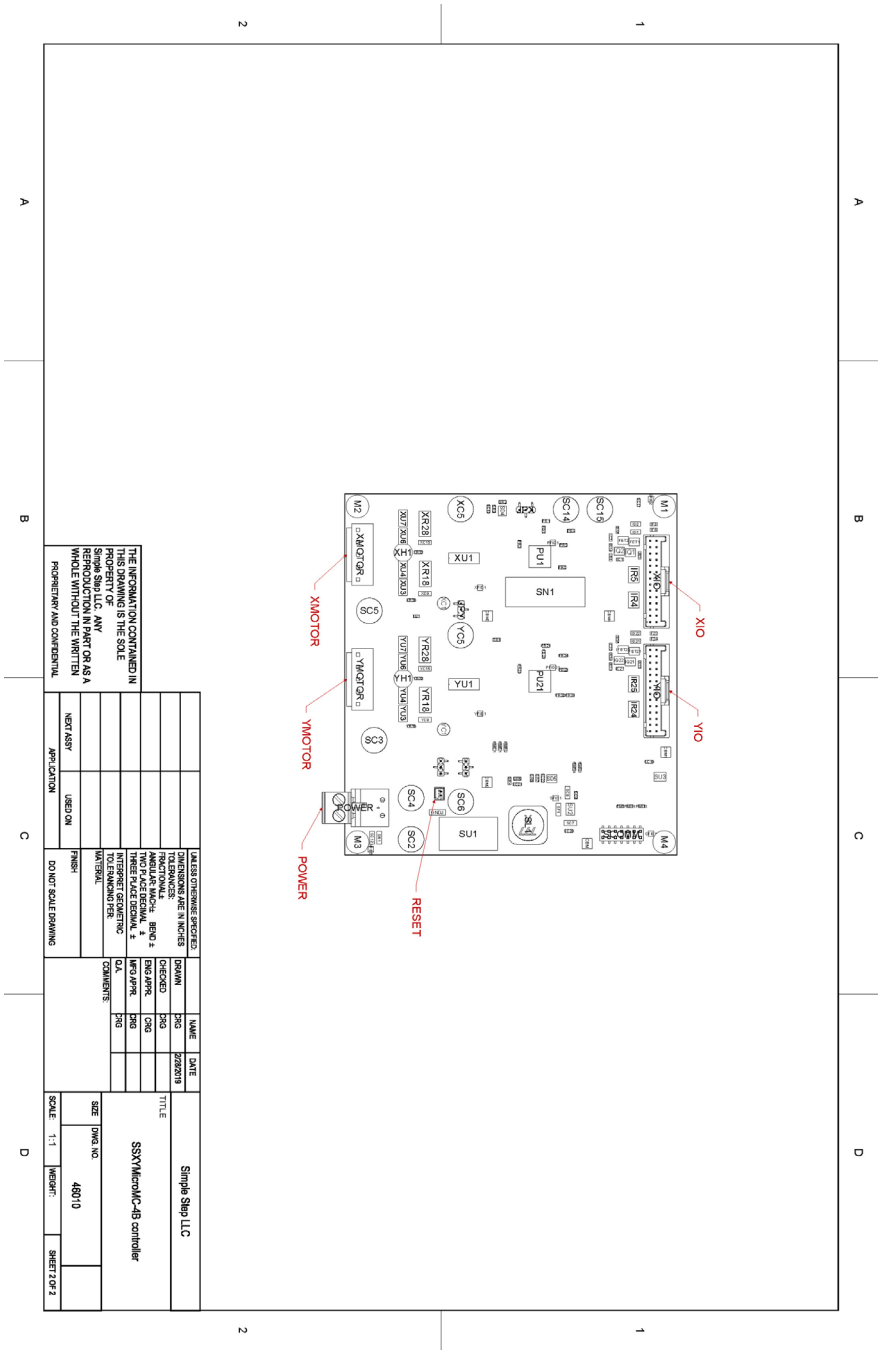


Figure 26 SSXYMicroMC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSXYZMicroLC Board Connector Locations

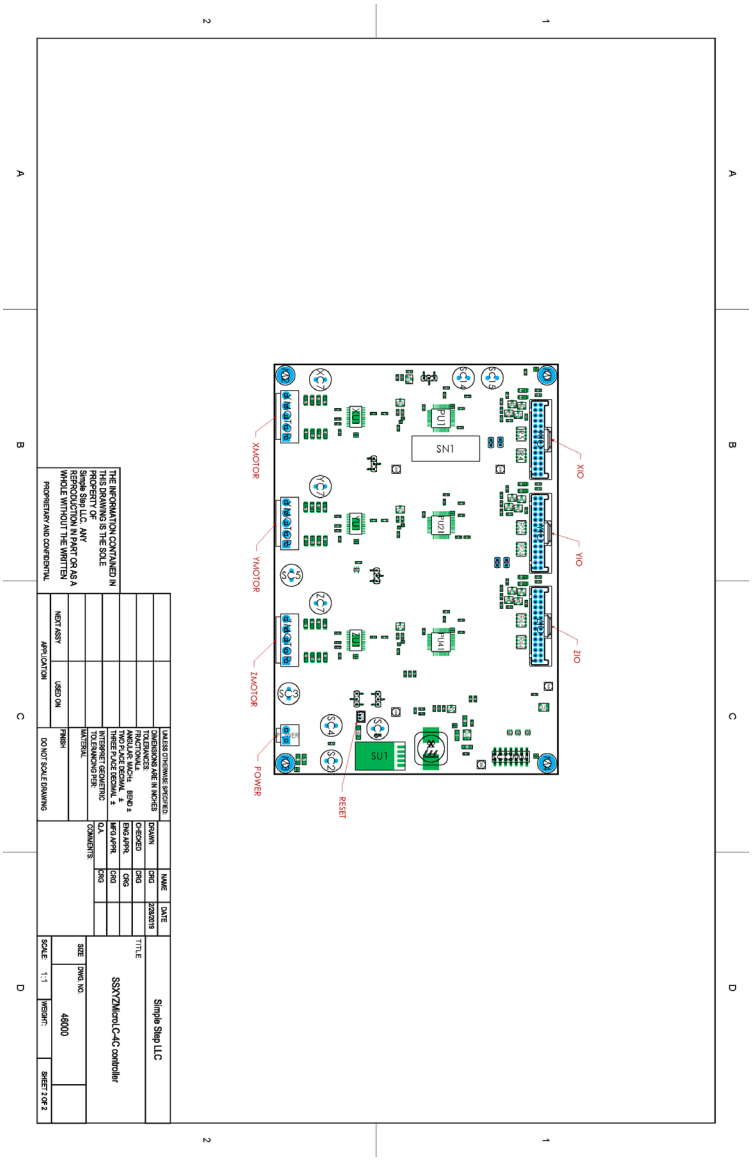


Figure 27 SSXYZMicroLC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSXYZMicroMC Board Connector Locations

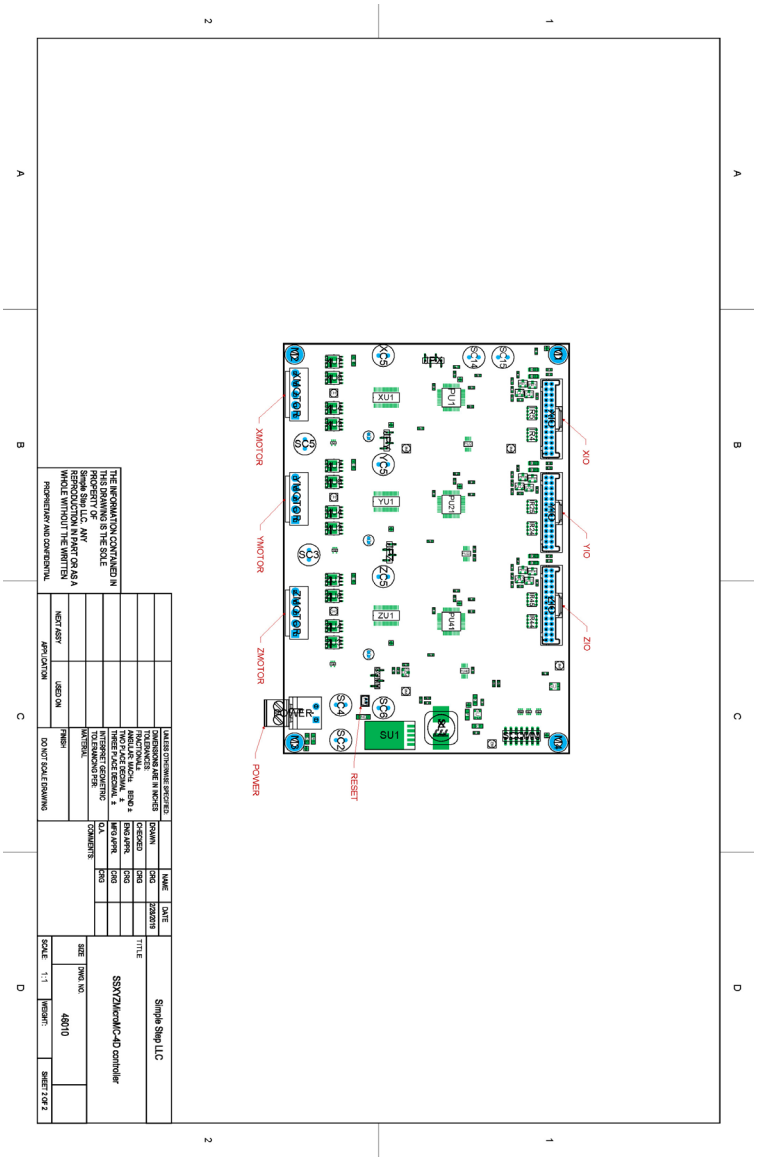


Figure 28 SSXYZMicroMC Board Connector Locations

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

In the above diagram, all pin #1 connectors are square. A one (1) or arrow (Δ) are silk screened onto the board next to the pin #1 connectors.

SSNEMA17 Board Connector Locations

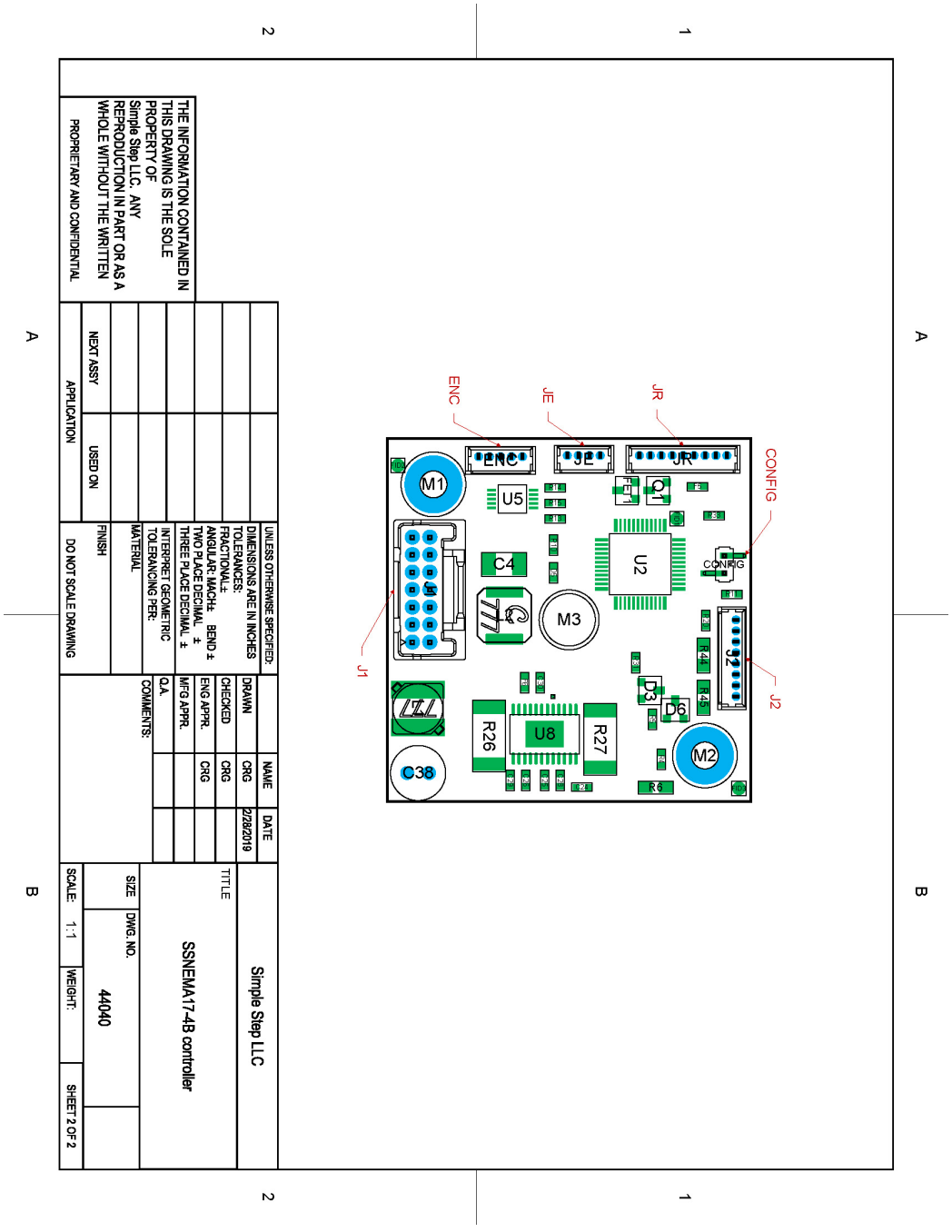


Figure 29 SSNEMA17-4B Board Connector Locations

APPENDIX D: MECHANICAL DRAWINGS

SSMicroLC Board Mounting Hole Outline

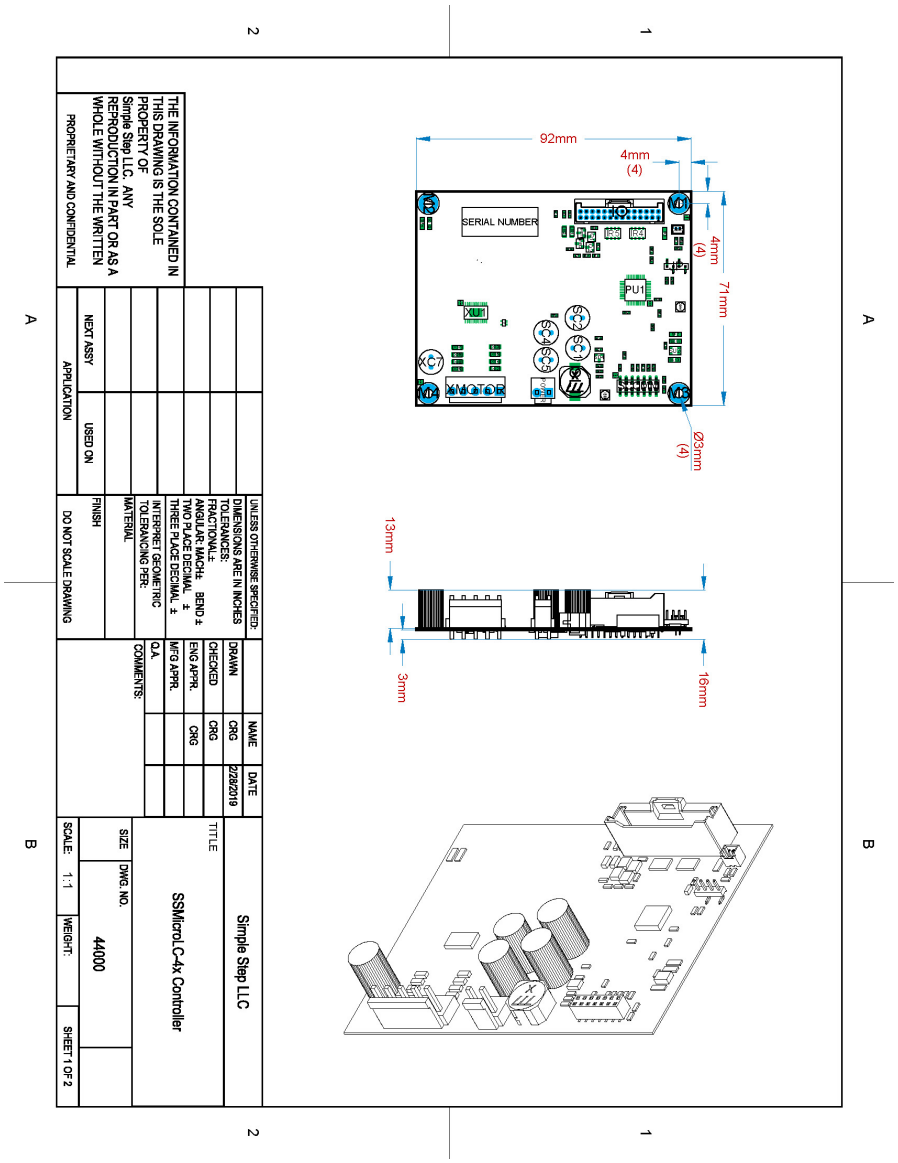


Figure 30 SSMicroLC Board Mounting Hole Outline

92 mm x 71mm x 16mm

SSMicroMC Board Mounting Hole Outline

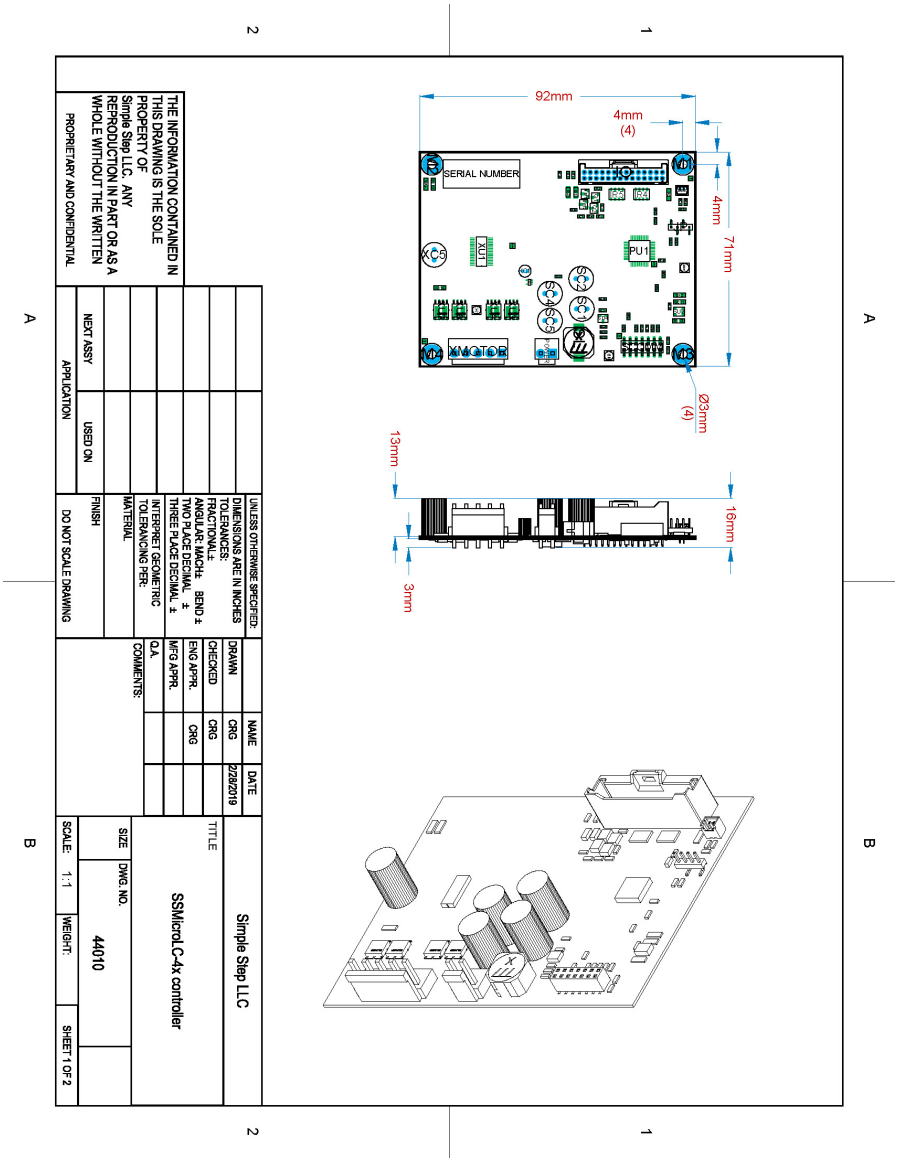


Figure 31 SSMicroMC Board Mounting Hole Outline

92 mm x 71mm x 16mm

SSXYMicroLC Board Mounting Hole Outline

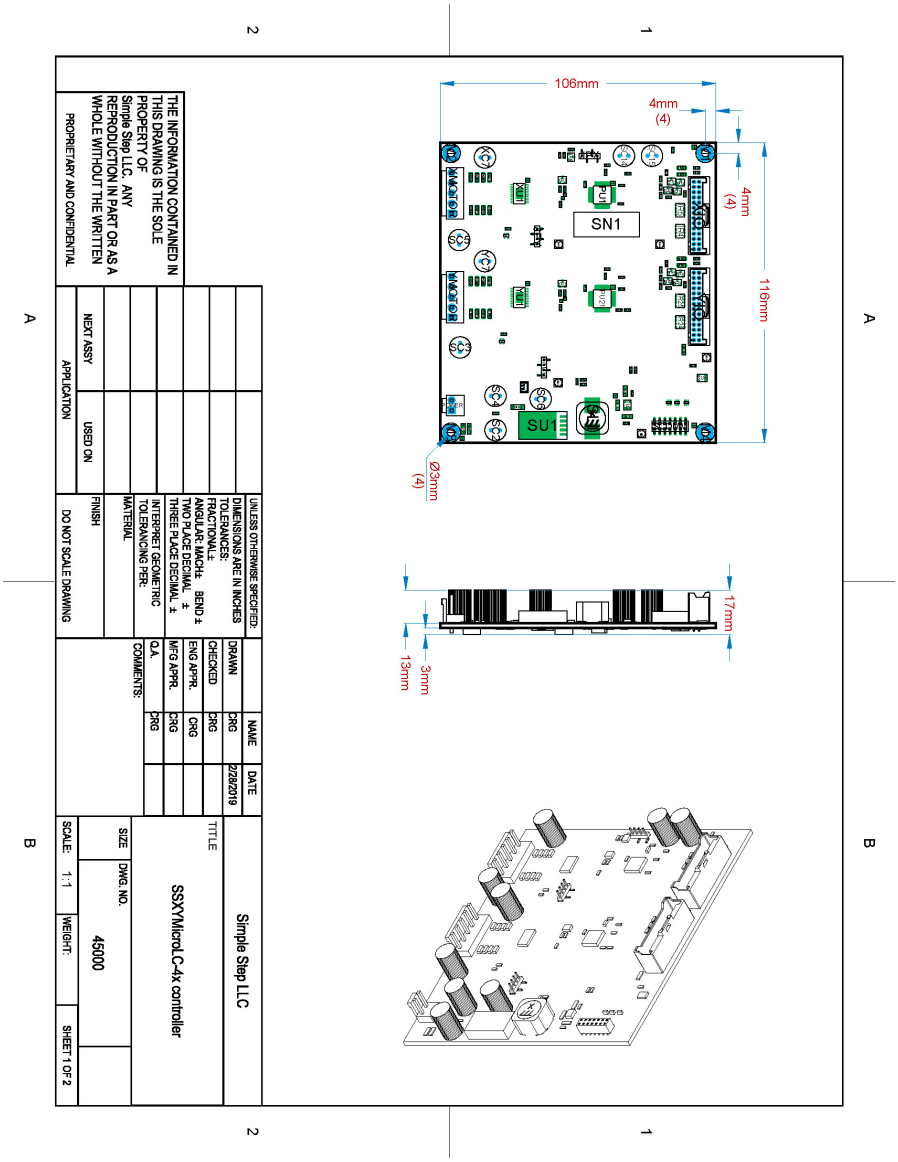


Figure 32 SSXYMicroLC Board Mounting Hole Outline

116mm x 106mm x 17mm

SSXYMicroMC Board Mounting Hole Outline

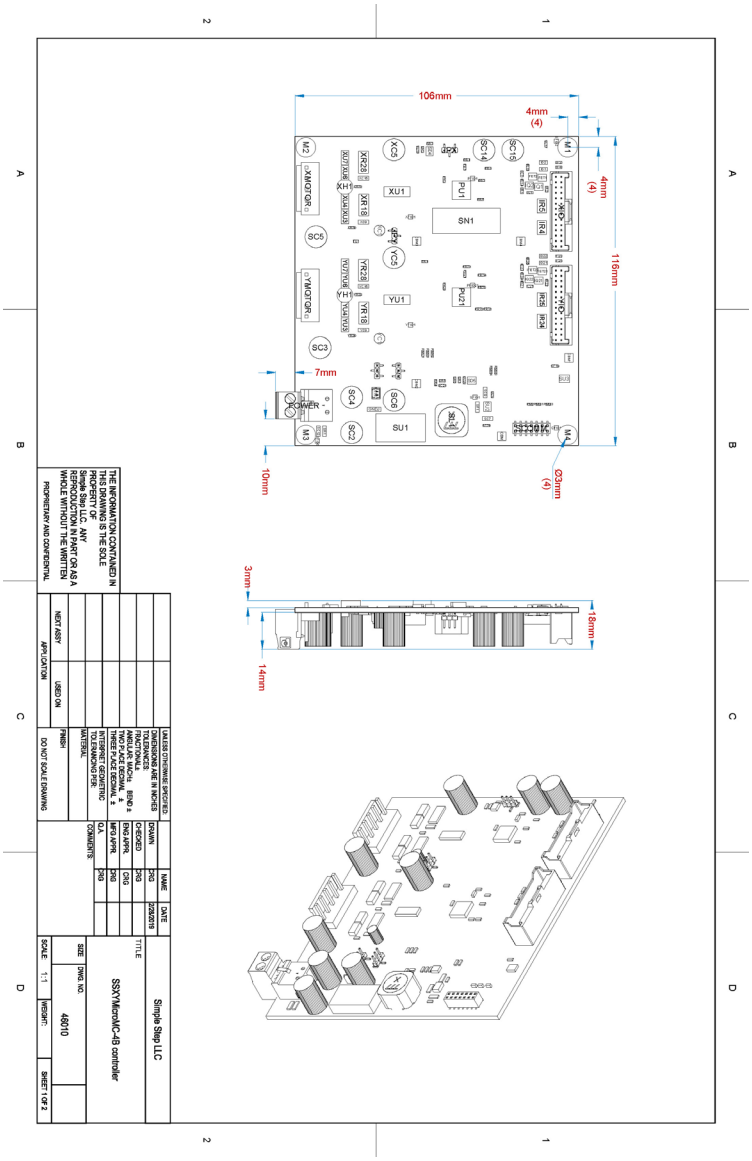


Figure 33 SSXYMicroMC Board Mounting Hole Outline

116mm x 106mm x 17mm

SSXYZMicroLC Board Mounting Hole Outline

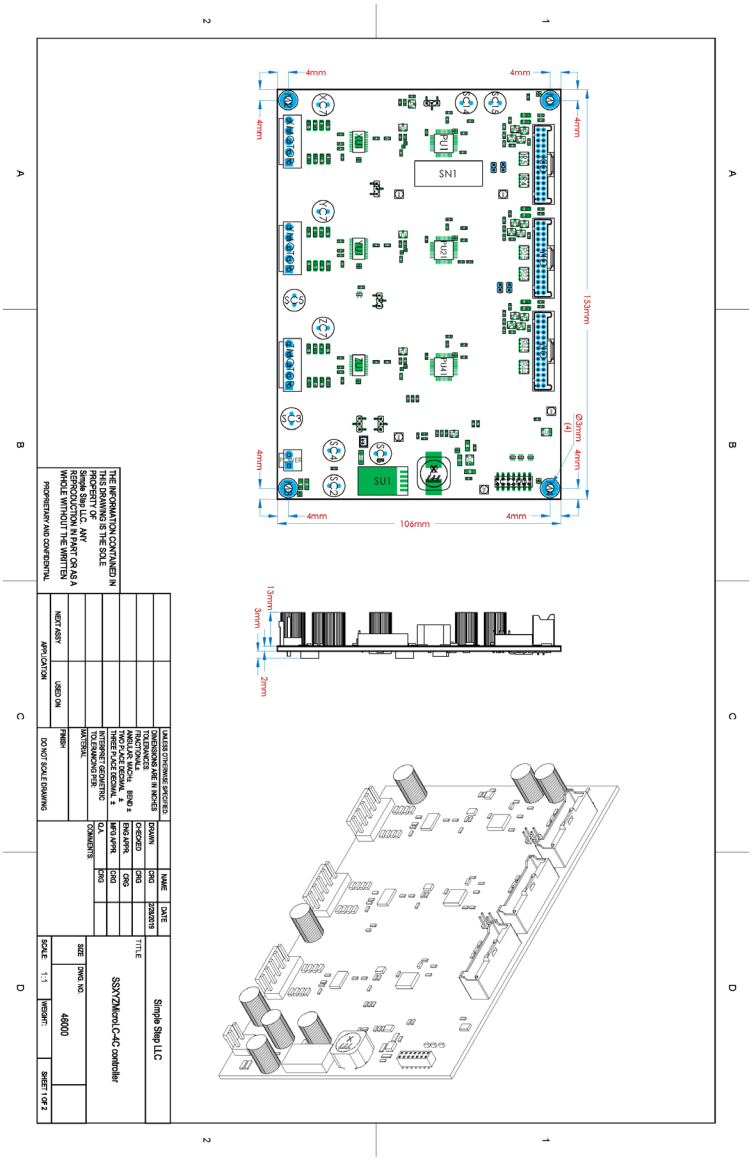


Figure 34 SSXYZMicroLC Board Mounting Hole Outline

153mm x 106mm x 18mm

SSXYZMicroMC Board Mounting Hole Outline

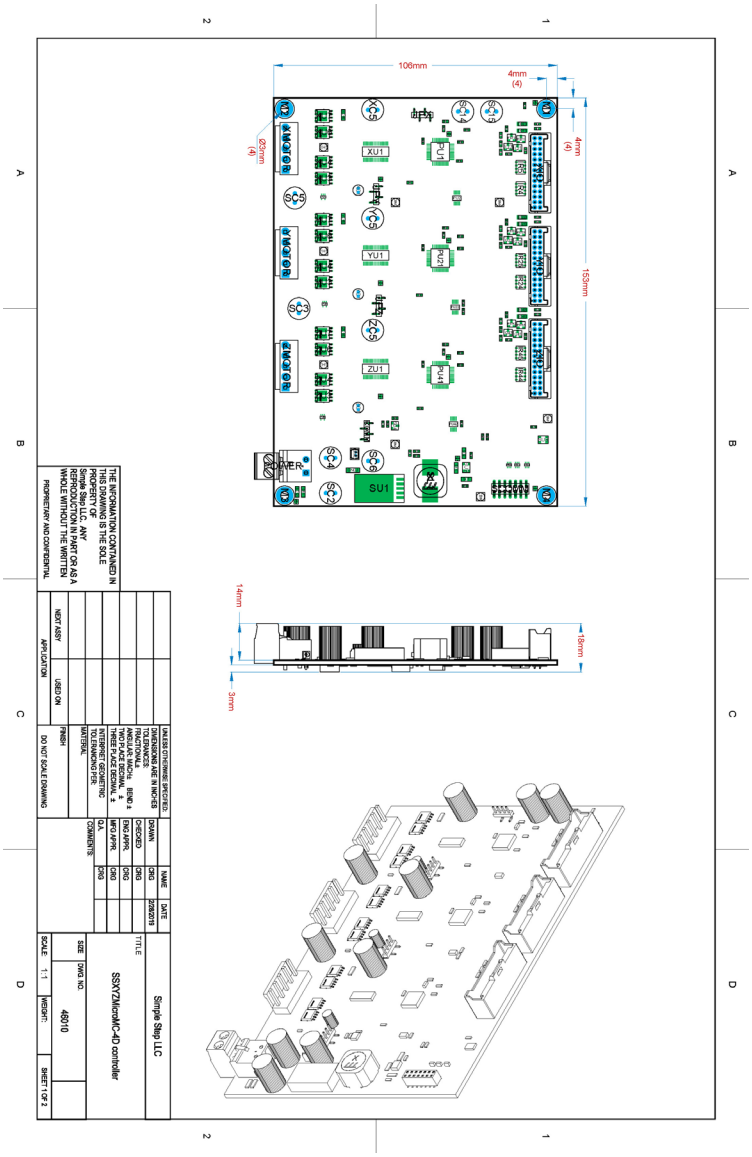


Figure 35 SSXYZMicroMC Board Mounting Hole Outline

153mm x 106mm x 18mm

SSNEMA17 Board Mounting Hole Outline

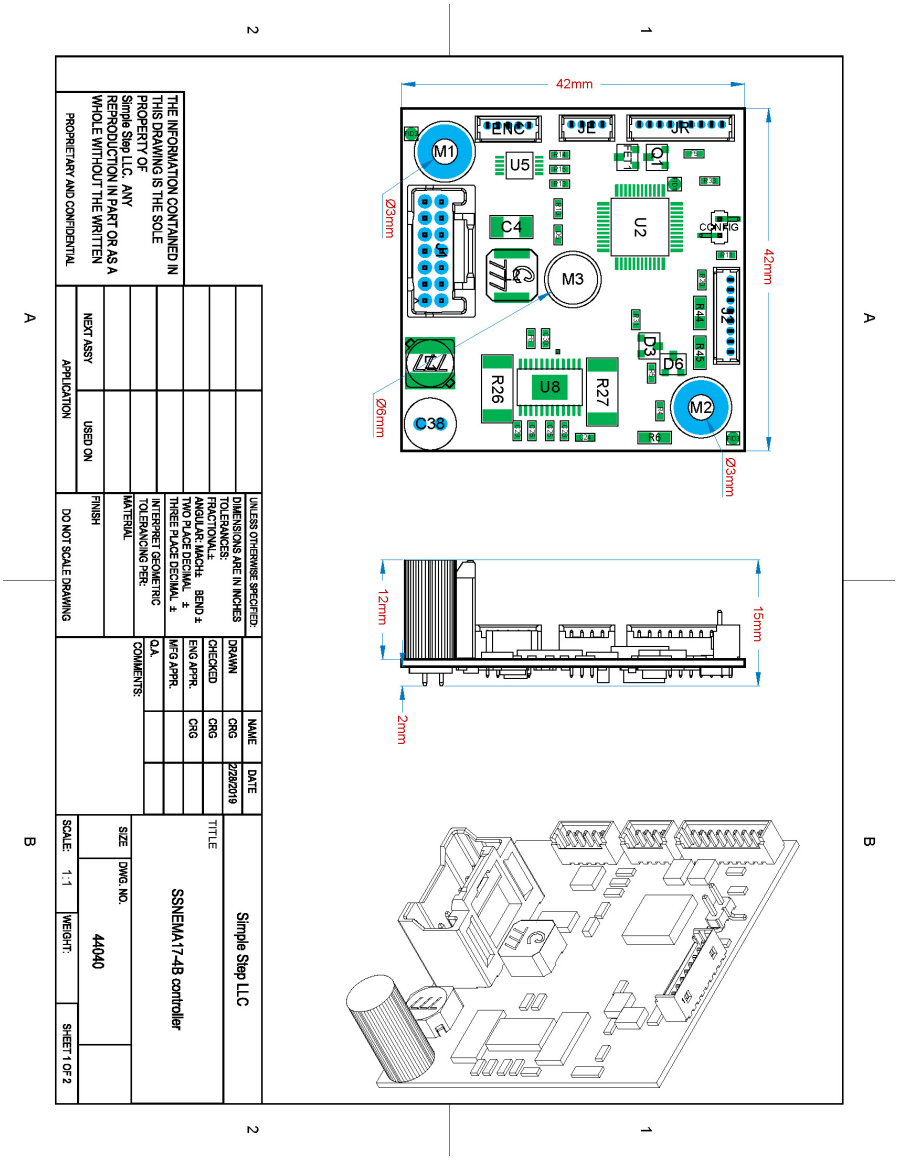


Figure 36 SSNEMA17 Board Mounting Hole Outline

42mm x 42mm x 15mm

APPENDIX E: BULKHEAD CABLE HARNESS ASSEMBLY

Instructions to Create a 1-Axis Motor Cable with Bulkhead Connector

Parts List

4pcs	4-40 Nut
1pc	4-40 Lock Washer
4pcs	4-40x1½" Screw
3pcs	Solder Lugs (Mouser part #534-7314)
3pcs	3 inch long, Black, 22 AWG stranded wire
2pcs	9 ½" 22 AWG, 4 Conductor Shielded Cable (Mouser Part #566-9418-xxx)
1pc	9 ½" 18 AWG, 4 Conductor Shielded Cable (Mouser part #602-2404C-xxx)
2pcs	36" 22 AWG, 4 Conductor Shielded Cable (Mouser Part #566-9418-xxx)
1pc	36" 18 AWG, 4 Conductor Shielded Cable (Mouser part #602-2404C-xxx)
2pcs	Molex, 4 Pin, 0.156 Housing (Molex #09-50-8043)
1pc	Molex, 8 Pin, 0.100 Housing (Molex #22-01-3087)
1pc	AMP Panel Mount Connector (AMP #206036-1)
1pc	AMP Panel Mount Connector Mate (AMP #206037-1)
1pc	AMP Cable Crimp (AMP #206322-1)
8pcs	AMP Pin Connectors (20-24 AWG) (AMP #66400-1)
8pcs	AMP Socket Connectors (20-24 AWG) (AMP #66399-1)
4pcs	AMP Pin Connectors (18-14 AWG) (AMP #66361-2)
4pcs	AMP Socket Connectors (18-24 AWG) (AMP #66360-2)
8pcs	Molex 0.100 contacts (22 – 30 AWG) (Molex #08-50-0114)
4pcs	Molex 0.156 contacts (18 – 24 AWG) (Molex #08-50-0106)
1pc	Digi-Key Home Sensor OR562-ND

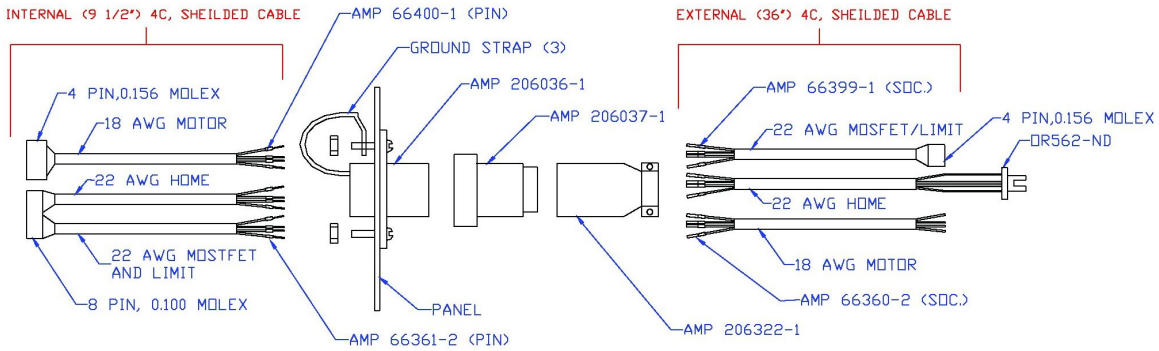


Figure 37 1-Axis Cable Assembly

Internal Home Sensor Cable

1. Obtain one 9 1/2” 22 AWG, 4 Conductor Shielded Cable (Mouser part #566-9418-xxx).

2. Strip 1 1/4” of insulation off both ends of the cable.

3. Cut the ground shield wire off both ends.

4. Strip 1/8” of insulation off all wire ends.

5. Crimp AMP Pin Connectors (part# 66400-1) onto all four wires at one end of the cable.

6. Crimp Molex 0.100 contacts (part #08-50-0114) onto the four wires at the opposite end of the cable.

7. Insert the Molex 0.100 contacts into the Molex 8-Pin 0.100 Housing (part #22-01-3087):

    a. Insert the red wire into position #5

    b. Insert the white wire into position #6

    c. Insert the green wire into position in #7

    d. Insert the black wire into position #8

8. Insert the AMP Pin Connectors into the AMP Panel Mount Connector (part # 206036-1):

    a. Insert the red wire into position #15

    b. Insert the black wire into position #16

    c. Insert the white wire into position #14

    d. Insert the green wire into position #10

Internal MOSFET/Limit Cable

1. Obtain one 9 1/2” 22 AWG, 4 Conductor Shielded Cable (Mouser Part #566-9418-xxx).

2. Strip 1 1/4” of insulation off both ends of the cable.

3. Cut the ground shield wire off both ends.

4. Strip 1/8” of insulation off all wire ends.

5. Crimp AMP Pin Connectors (part# 66400-1) onto all four wires at one end of the cable.

6. Crimp Molex 0.100 contacts (part #08-50-0114) onto the four wires at the opposite end of the cable.

7. Insert the Molex 0.100 contacts into the Molex 8-Pin 0.100 Housing (part #22-01-3087):

    a. Insert the red wire into position #1

    b. Insert the black wire into position #2

    c. Insert the white wire into position #3

    d. Insert the green wire into position #4

8. Insert the AMP Pin Connectors into the AMP Panel Mount Connector (part # 206036-1):

    a. Insert the green wire into position #13

    b. Insert the white wire into position #12

    c. Insert the red wire into position in #11

    d. Insert the black wire into position #7

Ground Straps

1. Obtain three 3” black 22 AWG stranded wires.
2. Strip 1/8” of insulation from one end of each wire.
3. Strip 1/2” of insulation from the opposite end of the wires.
4. Crimp AMP Pin Connectors (part# 66400-1) onto the 1/8” stripped ends.
5. Solder Mouser Solder Lugs (part #534-7314) onto the opposite ends.
6. Insert the AMP Pin Connectors into the AMP Panel Mount Connector (part # 206036-1):
   a. Position #9
   b. Position #6
   c. Position #5
7. Connect one Mouser Solder Lug (part #534-7314) onto one screw holding the AMP Panel Mount Connector (part # 206036-1) and bolt down.
8. Perform step 7 for the remaining two ground straps.

Internal Motor Harness

1. Obtain one 9 ½" 18 AWG, 4 Conductor Shielded Cable (Mouser part #602-2404C-xxx)

2. Strip 1 1/4" of insulation off both ends of the cable.

3. Cut the ground shield wires off both ends.

4. Strip 1/8" of insulation off all wires ends.

5. Crimp AMP Pin Connectors (part #66361-2) onto all four wires at one end of the cable.

6. Crimp Molex 0.156 contacts (part #08-50-0106) onto the wires at the opposite end of the cable.

7. Insert the Molex 0.156 contacts into a Molex 4-Pin 0.156 Housing (part # 09-50-8043):

    a. Insert the white wire into position #1

    b. Insert the green wire into position #2

    c. Insert the black wire into position #3

    d. Insert the red wire into position #4

8. Insert the AMP Pin Connectors into the AMP Panel Mount Connector (part # 206036-1):

    a. Insert the green wire into position #1

    b. Insert the white wire into position #2

    c. Insert the red wire into position #3

    d. Insert the black wire into position #4

External Motor Harness

1. Obtain one 36” length 18 AWG 4 Conductor Shielded Cable (Mouser part #602-2404C-xxx).
2. Strip 1 1/4” of insulation from one end of the cable.
3. Strip 1/8” of insulation from all wires at the stripped end.
4. Crimp the AMP Socket Connectors (part# 66360-2) onto all wires including the ground straps.
5. Insert the AMP Socket Connectors into the AMP Panel Mount Connector Mate (part #206037-1):
    a. Insert the green wire into position #1
    b. Insert the white wire into position #2
    c. Insert the red wire into position #3
    d. Insert the black wire into position #4
6. Obtain the AMP Cable Crimp (part #206322-1) and insert the opposite end of the cable through the housing.

NOTE

All the wires will be inserted through this housing. When all cable connections are complete, the locking housing will be screwed onto the AMP Panel Mount Connector.

7. Strip 1 1/4” of insulation from that end of the cable.
8. Cut the ground strap off on that end of the cable.
9. Strip 1/8” off insulation off of all four wires.
10. Connect the white wire to motor phase A
11. Connect the green wire to motor phase B
12. Connect the black wire to motor phase C
13. Connect the red wire to motor phase D

Home Sensor Cable

1. Obtain one 36” length 22 AWG 4 Conductor Shielded Cable (Mouser part#566-9418-xxx).

2. Strip 1 1/4” of insulation from one end of the cable.

3. Strip 1/8” of insulation from all four wires at the stripped end.

4. Crimp AMP Socket Connectors (part #66399-1) onto each wire.

5. Insert the AMP Socket Connectors into the AMP Panel Mount Connector Mate (part #206037-1):

    a. Insert the black wire into position #16

    b. Insert the red wire into position #15

    c. Insert the white wire into position #14

    d. Insert the green wire into position #10

    e. Insert the ground strap into position #6

6. Pass the opposite end of the cable through the AMP Cable Crimp as was done for the external motor harness.

7. Strip 1 1/4” of insulation from that end of the cable.

8. Strip 1/8” of insulation from all four wires.

9. Cut off the ground strap.

10. Obtain the Digi-Key Home Sensor (OR562-ND) and solder like wire colors to each other.

External MOSFET/Limit Input Cable

1. Obtain one 36” length 22 AWG 4 Conductor Shielded Cable (Mouser part #566-9418-xxx).
2. Strip 1 1/4” of insulation of one end of cable.
3. Strip 1/8” of insulation off all four wires.
4. Crimp AMP Socket Connectors (part #66399-1) onto each wire.
5. Insert the AMP Socket Connectors into the AMP Panel Mount Connector Mate (part #206037-1):
    a. Insert the green wire into position #13
    b. Insert the white wire into position #12
    c. Insert the red wire into position #11
    d. Insert the black wire into position #7
    e. Insert the ground strap into position #9
6. Pass the opposite end of the cable through the AMP Cable Crimp as was done for the home sensor cable.
7. Strip 1 1/4” of insulation from that end of the cable.
8. Strip 1/8” of insulation from the four wires.
9. Crimp Molex 0.156 contacts (part #08-50-0106) onto the four wires.
10. Insert the Molex 0.156 contacts into a Molex 4-Pin 0.156 Housing (part # 09-50-8043):
    a. Insert the red wire into position #1
    b. Insert the black wire into position #2
    c. Insert the white wire into position #3
    d. Insert the green wire into position #4

Completing the Cable Assembly

1. Tightly screw the AMP harness clamp onto the AMP Panel Mount Connector Mate (part #206037-1)
2. Insert the AMP clamp (from the AMP Cable Crimp part #206322-1 kit) and tighten the two screws.

AMP Pin	Wire Color	Wire Size	SSCB Conn.	SSXYZ Conn.	Conn. Pin Number	Signal Description
1	Green	18 AWG	J3	J3,J7,J11	2	MOTOR B
2	White	18 AWG	J3	J3,J7,J11	1	MOTOR A
3	Red	18 AWG	J3	J3,J7,J11	4	MOTOR D
4	Black	18 AWG	J3	J3,J7,J11	3	MOTOR C
5	Cable Gnd	22 AWG	Strap to Chassis	Strap to Chassis		CHASSIS GROUND
6	Cable Gnd	22 AWG	Strap to Chassis	Strap to Chassis		CHASSIS GROUND
7	Black	22 AWG	J5	J5,J9,J13	2	FET SINK CONTROL
8						
9	Cable Gnd	22 AWG	Strap to Chassis	Strap to Chassis		CHASSIS GROUND
10	Green	22 AWG	J2	J4,J8,J12	4	HOME SENSOR GROUND
11	Red	22 AWG	J5	J5,J9,J13	1	FET MOTOR POWER
12	White	22 AWG	J6	J6,J10,J14	1	LIMIT INPUT
13	Green	22 AWG	J6	J6,J10,J14	2	LIMIT GROUND
14	White	22 AWG	J2	J4,J8,J12	2	HOME SENSOR OUTPUT
15	Red	22 AWG	J2	J4,J8,J12	1	HOME SENSOR POWER
16	Black	22 AWG	J2	J4,J8,J12	3	HOME SENSOR GROUND

Table 21 AMP Series 1 Motor Harness Breakout

Type	Description	Manufacture	Part Number	Rep.	Rep. Part Number
Home Sensor	Opto Sensor	Omron	OR562	Digi-Key	OR562-ND
Motor Conn.	16 pin Conn.	AMP	206036-1	Mouser	571-2060361
Motor Conn. Pins	Pins (AWG 18-22)	AMP	66591-1	Mouser	571-665911
Motor Conn. Mate	16 pin Conn. Mate	AMP	206037-1	Mouser	571-2060371
Motor Conn. Mate Pins	Pins (AWG 18-16)	AMP	206322-1	Mouser	571-665921
Motor Conn. Mate Pins	Pins (AWG 24-20)	AMP	66101-4	Mouser	571-665921
Motor Conn. Mate Clamp	Cable Clamp	AMP	66399-4	Mouser	571-2063221
4 Conductor Cable - 100' Roll	18 AWG Cable	Belden	9418	Mouser	566-9418-100
4 Conductor Cable - 100' Roll	22 AWG Cable	Alpha	2404C	Mouser	602-2404C-100

Table 22 AMP Series 1 Motor Harness Part Numbers

APPENDIX F: COMMAND FORMATS

Commands from the host have the following format:

Byte 1:	Board type
Byte 2:	Board address
Bytes 3 to n-1:	Command string
Byte n:	Carriage return (0x0D)

For example “D1I3<CR>”

Responses from the boards have the following format:

Byte 1:	Board type
Byte 2:	Board address
Byte 3:	Board status
Bytes 4 to n-1:	Response string
Byte n:	Carriage return (0x0D)

For example “d1>00001”

NOTE

If a command contains multiple numeric parameters, parameters are separated by a comma.

Board Type Prefix Characters

The board type determines a command's prefix character:

Board Type	Prefix character from the host	Prefix response from the board
SSMicroMC	D	d
SSMicroLC	U	u
SSNEMA17	X	x
All multi-axis controllers	X, Y, or Z	x, y, or z
Global	G	No response

Table 23 Board Type - Byte 1 in Command Requests

Simultaneous Movement and Global Commands

If you have a 4-axis controller system based on 1- SSXYZMicroLC Board at address 0 and 1-SSMicroMC Board at address 0. The following commands can be issued to move all the axes to absolute position 100:

- D0M100<CR>
- X0M100<CR>
- Y0M100<CR>
- Z0M100<CR>

This command sequence requires the controlling computer system to send individual commands to each axis and wait for an acknowledgment from each board, precluding moving motors at the same time. A special command syntax was created so the controlling system could send the following command instead:

- D0M100,X0M100,Y0M100,Z0M100<CR>

Each axis parses the command, accepts its command, and ignores the rest of the command string. All motors wait for the last board to acknowledge the command and then start their movement at approximately the same time (100-400us).

The 'G'lobal command can also be used to perform the same operation:

- G0M100<CR>

NOTE

With the previous command syntax certain axis can be controlled. The global command can only be used to control all axes.

WARNING

If a syntax error is found in one of the axis commands, that axis responds immediately with an error status. When the 'G'lobal command is used NO ERROR is sent back.

Additional global command examples are shown below:

- "DGN+0"<CR> Initializes all SSMicroMC Boards, without moving the motor, so that home is in the CW direction.

- "XGM1000"<CR> Moves all 'X' axes to absolute position 1000.
- "G0M101<CR> Tells all boards that are at address 0 to move to position 101.

Board Status Characters

All commands that are sent to the Simple Step Controller Board are acknowledged with a status value. Commands are acknowledged within 1ms (typically 0.5ms with an operating baud rate of 57.6K). (The only exception to this rule is the motor abort command with deceleration "!" and in non-RTOS environment).

Status	Description
>	System is ready for another command.
s	Motor homed, but not moved ('N' command).
j	Jog command active.
#	Command syntax error.
!	Command parameter out of range.
a	Motor abort in progress.
b	Motor is running.
w	Motor command stack full.
f	Motor command complete.
u	Updating 'C' command 'E' value.
%	Command ignored. Motor running.
h	Axis has not been initialized.
l	(Small L) RTOS soft limit (upper or lower) has been reached.
H	Motor at Home position, cannot move past home.
L	Motor at Limit position, cannot move past limit.
d	Delay command complete.
^	IAP (In Application Programming) mode is active.
+	IAP CRC error found.
r	Currently running in IEEPROM program.
c	Finished IEEPROM Programming.
n	Cannot program, IEEPROM is not erased.
e	Encoder error motion aborted
o	Opto-Encoder error motion aborted.
B	v109.xx Board configuration data not erased.
A	v109.xx Axis configuration data not erased.
p	v109.xx Customer configuration data not erased.
i	V109.xx IEEPROM Call stack error.
E	V109.xx Watchdog error occurred.

Table 24 Status Characters - Byte 3 in Command Responses

Unless a board has the RTOS option installed, or the status command was activated using the "Ns" command, all motor movement commands disable the command parser so that no other commands can be executed during motor movements.

Motor operation complete can be determined by sending a status request, <CR>. The board responds with an ">" status indicating that the motor operation is complete and the board is ready for the next command sequence. If no response is received, the board is assumed to be busy.

NOTE

The board does not send unsolicited responses to the host

APPENDIX G: REQUEST FOR BOARD CAPABILITIES: THE (V) COMMAND

The 'v' command allows the user to determine board capabilities. When the 'v' command is issued it includes a parameter that indicates the type of information being requested.

Parameter	Requested Information
0	Primary software version number
1	Board Type
2	Board options #1
3	CPU Type
4	Sub-version software number
5	Board options #2
6	Board revision level
7	Board options #3
8	Motor driver type
9	Firmware test version number
A	Firmware compile date
B	Firmware compile time
C	Board options #4
D	IAP version number
E	IAP Sub-version number
F	Internal IEEPROM flash size

Table 25 Parameter options for the (v) Command

Board Response

The board responds with the requested information.

Primary Software Version Number

A five-digit number with the decimal points excluded. For example, version 1.4.0 is returned as 00140.

Board Type

Response	Meaning
62	SSMicroLC single axis low current motion controller
63	SSXYMicroLC dual axis low current motion controller
64	SSXYZMicroLC triple axis low current motion controller
67	SSMicroMC single axis medium current motion controller
68	SSXYMicroMC dual axis medium current motion controller
69	SSXYZMicroMC triple axis medium current motion controller
72	SSMicroHC single axis high current motion controller
73	SSXYMicroHC dual axis high current motion controller
74	SSXYZMicroHC triple axis high current motion controller
82	SSNEMA17 single axis low current motion controller
83	SSNEMA23 single axis medium current motion controller
84	SSNEMA34 single axis high current motion controller
86	SSBLDC single axis high current BLDC motion controller
87	CB-Laser single axis medium current BLDC motion controller and Communications boards with USB and Bluetooth interface.
255	Test Unit

Table 26 Board Type in (v) Command Response

Board Options #1

Response	Meaning (Bit Flags)
Bit 0	Version 1.0.3 Software - Board Revision A
Bit 1	Version 1.0.3 Software - Board Revision B
Bit 2	Version 1.0.3 Software - Board Revision C
Bit 3	Version 1.0.3 Software - Board Revision D
Bit 4	Version 1.0.3 Software - Board Revision E
Bit 5	32 Bit Software (All Versions)
Bit 6	2K Internal EEPROM Board (All Versions)
Bit 7	X2 Processor
Bit 8	Prescaler Option
Bit 9	QE Interface
Bit 10	High Current Interface
Bit 11	Velocity Control
Bit 12	Not Used
Bit 13	MAX3100 Interface

Bit 14	Analog to Digital Converter Interface
Bit 15	Serial EEPROM (AT25CXX)

Table 27 Board Options #1 in (v2) Command Response

CPU Type

Response	Meaning
0x012	LPC1317
0x013	LPC1347
0x014	LPC1763
0x015	LPC1766
0x016	LPC1767
0x017	LPC1549
0x018	LPC1547
0x019	LPC1517

Table 28 CPU Type in (v) Command Response

Board Options #2

Response	Meaning (Bit Flags)
Bit 0	Dual Channel 8 bit DAC Installed on board
Bit 1	Microstepping Software Installed
Bit 2	Four Channel 8 bit DAC Installed on board
Bit 3	Internal Quadrature Interface Installed on board
Bit 4	Global Input TRIP Software Installed on Board
Bit 5	RTOS Software Installed
Bit 6	General 1ms Global Timer Software Installed
Bit 7	Delimiter Timer Software Installed
Bit 8	Interrupt Driven Serial Software Installed #1
Bit 9	Serial I2C Interface
Bit 10	Serial SPI Interface
Bit 11	Serial Microwire Interface
Bit 12	LM75 Interface
Bit 13	Binary Xfer Interface
Bit 14	BCD Branch Controller
Bit 15	Optical Counter

Table 29 Board Options #2 in (v5) Command Response

Board Revision Level

Response	Meaning
30	Board Revision 4A
31	Board Revision 4B
32	Board Revision 4C
33	Board Revision 4E
34	Board Revision 4F
35	Board Revision 4G
36	Board Revision 4H
37	Board Revision 4I
38	Board Revision 4J
39	Board Revision 4K
40	Board Revision 4L

Table 30 Board Revision Level in (v) Command Response

Board Options #3

Response	Meaning (Bit Flags)
Bit 0	MTU Interface
Bit 1	Sleep Mode
Bit 2	Motion Done Status
Bit 3	Communication Test
Bit 4	Step Delay
Bit 5	Motor Busy Line
Bit 6	6 bit Board Address
Bit 7	ATMI Interface
Bit 8	Sync. Motion
Bit 9	Jog Interface
Bit 10	32 bit ASCII Interface
Bit 11	Software Limit Update
Bit 12	Focus Motor Control
Bit 13	Zoom Motor Control
Bit 14	Serial #2 ISR
Bit 15	Linear Movement Interface

Table 31 Board Options #3 in (v7) Command Response

Motor Driver Type

Response	Meaning (Bit Flags)

0x0000	No Motor Driver
0x1300	A3992 Motor Driver
0x1400	A3985 Motor Driver
0x1500	A3938 Motor Driver
0x1600	A3936 Motor Driver
0x1700	A3930 Motor Driver

Table 32 Motor Driver Type in (v) Command Response

Board Options #4

Response	Meaning (Bit Flags)
Bit 0	IAP Interface
Bit 1	SPI Interface
Bit 2	DC Motor Control
Bit 3	Dual Channel 12 bit ADC Interface
Bit 4	SSQE Binary Mode Interface
Bit 5	Not Used
Bit 6	Sanyo Denki Servo Amp Driver
Bit 7	LTC1661 DAC control
Bit 8	FAST crystal installed onboard
Bit 9	PID control
Bit 10	Auto-correction
Bit 11	Overdrive control
Bit 12	Unsigned Long Integers
Bit 13	Position Trigger
Bit 14	Encoder Learn
Bit 15	Deceleration offset

Table 33 Board Options #4 in (vC) Command Response

Board Options #5

Response	Meaning (Bit Flags)
Bit 0	Watchdog module Installed and enabled
Bit 1	S-Curve module installed
Bit 2	Slope torque control module installed
Bit 3	USB 2.0 interface installed
Bit 4	SPI multi-processor communications module installed

Bit 5	IAP programming module installed
Bit 6	SSP multi-processor communications module installed
Bit 7	High speed multi-processor communications module installed
Bit 8	Not used
Bit 9	Not used
Bit 10	Internal ADC module installed
Bit 11	SPI Quadrature Encoder module installed
Bit 12	Servo Motor module installed
Bit 13	Resonance Elimination module installed
Bit 14	Power conservation module installed
Bit 15	Short circuit protection module installed

Table 345 Board Options #5 in (v11) Command Response

Board Options #6

Response	Meaning (Bit Flags)
Bit 0	Velocity control module installed
Bit 1	PID log module installed
Bit 2	LM35D temp sensor module installed
Bit 3	Hardware Current Simulator module installed
Bit 4	AT24CXX IEEPROM module installed
Bit 5	PID Positional control module installed
Bit 6	Fan Control module installed
Bit 7	DSPIC33 Encoder module installed
Bit 8	Scaled Encoder module installed
Bit 9	Interrupt HOME module installed
Bit 10	Interrupt LIMIT module installed
Bit 11	Command recode mode module installed
Bit 12	Scaled ADC readings module installed
Bit 13	Coordinated Motion module installed
Bit 14	AT24DF041 FLASH module installed
Bit 15	Board Voltage Monitor module installed

Table 356 Board Options #6 in (v1B) Command Response

Board Options #7

Response	Meaning (Bit Flags)
Bit 0	Motor Current Monitor module installed
Bit 1	High Level Movement module installed
Bit 2	Output Flasher module installed
Bit 3	Test Logger module installed
Bit 4	Serial 1 ISR module installed
Bit 5	Bluetooth Serial module installed
Bit 6	GM Code Converter module installed
Bit 7	Trip Record mode module installed
Bit 8	Dual Serial Link module installed
Bit 9	LMT86DCK Temp Sensor module installed
Bit 10	Secondary Encoder module installed
Bit 11	SSQE Record Mode module installed
Bit 12	Motor Interlock System module installed
Bit 13	Speed Adjust Master module installed
Bit 14	Speed Adjust Slave module installed
Bit 15	Use DMA Controller module installed

Table 367 Board Options #7 in (v1C) Command Response

Board Options #8

Response	Meaning (Bit Flags)
Bit 0	Acceleration Assist module installed
Bit 1	Hardware Current Divider module installed

Table 378 Board Options #8 in (v1D) Command Response

APPENDIX H: COMMAND SUMMARY

Command	Description	See page	All MicroMC	*All* MicroLC	SSNEMA17	Notes
Power Setting Commands						
P	Set power settings (software settable)	59		X	X	
p	Check power settings (software settable)	60		X	X	
Standard Commands						
M	Move motor to absolute position	63	X	X	X	
m	Ask for current motor position	63	X	X	X	
R	Perform relative motor movement	64	X	X	X	
C	Perform continuous motor movement	65	X	X	X	
N	Null (zero/home) the motor	66	X	X	X	
A	Set motor absolute position	69	X	X	X	
*	Abort the motor- hard	69	X	X	X	
!	Abort the motor- soft	69	X	X	X	
B	Set beginning velocity	70	X	X	X	
b	Ask for beginning velocity	70	X	X	X	
E	Set end velocity	71	X	X	X	
e	Ask for end velocity	71	X	X	X	
S	Set slope	72	X	X	X	
s	Ask for slope	72	X	X	X	
r	Set prescaler option	73	X	X	X	

Command	Description	See page	All MicroMC	All MicroLC	SSNEMA17	Notes
H	Set motor to half step	74	X	X	X	Will not control Gecko driver amp
F	Set motor to full step	74	X	X	X	Will not control Gecko driver amp
H	Set motor microstepping mode	75		X		
oP	Set settling timer	76	X	X	X	Only valid in the v101 to v104
X	Set timebase	76	X	X	X	Only valid in the v101 to v104 Requires X2 option
oj	Enable or disable JOG mode	77	X	X	X	
<CR>	Get Status	79	X	X	X	
d	Perform a delay	79	X	X	X	
v	Get board capabilities	80	X	X	X	
z	Zero previous and current status	81	X	X	X	
I	Get Input Port State	82	X	X	X	
O	Set Output Control Line	83	X	X		
RTOS Commands						
oa	Set abort mode	85	X	X	X	Requires RTOS
op	Capture/display/reset RTOS position register	86	X	X	X	Requires RTOS (except for “opd”)
os	Set software limits for continuous mode	87	X	X	X	Requires RTOS
T	Enable or disable the trip option	88	X	X	X	Requires RTOS
New System Monitoring Commands						
#VD	Displays current “Power” input voltage	91	X	X	X	

Command	Description	See page	All MicroMC	*All* MicroLC	SSNEMA17	Notes
#VX	Displays maximum “Power” input voltage recorded since power on.	91	X	X	X	
#VM	Displays minimum “Power” input voltage recorded since power on.	91	X	X	X	
#Vx	Displays maximum “Running Power” input voltage recorded since power on.	91	X	X	X	
#Vm	Displays minimum “Running Power” input voltage recorded since power on.	91	X	X	X	
#Vr	Resets ALL maximum and minimum values to the current voltage reading on the POWER connectors	91	X	X	X	Only available on version 300.9.11 or above
oTD	Displays current axis temperature reading in degrees F	91	X	X	X	
oTX	Displays maximum temperature sensor recorded since power on.	91	X	X	X	Only available on version 300.9.11 or above
oTM	Displays minimum temperature sensor recorded since power on.	91	X	X	X	Only available on version 300.9.11 or above
oTR	Resets the maximum and minimum temperature readings to the current temperature reading.	91	X	X	X	Only available on version 300.9.11 or above
Quadrature Encoder Board Commands						
qN	Initialize the Axis Home Direction	94	X	X	X	
qmp	Set Quadrature Encoder Mode	95	X	X	X	
qE or qD	Enable or disable the Quadrature Encoder Axis	97	X	X	X	
qMp	Set QE Interface Count Mode	97	X	X	X	

Command	Description	See page	All MicroMC	*All* MicroLC	SSNEMA17	Notes
qP	Get the Current Quadrature Encoder Position	98	X	X	X	
qd	Set the Quadrature Encoder Divider Register	98	X	X	X	
qT	Set Motor Movement Tolerance	98	X	X	X	
qa	Set Auto-correction Tolerance	98	X	X	X	
qr	Set Auto-correction Retry Counter	99	X	X	X	
Communication Commands						
c	Set Communications Baud Rate	103	X	X	X	
ob	Set Binary Transfer Option	104	X	X	X	Special order option
on	Set Networking	105	X	X	X	
or	Set Radix Number	105	X	X	X	
oC	Set Communication Type	106	X	X	X	
IEEPROM Commands						
Q	Program EEPROM	108	X	X	X	
K	Run EEPROM Contents	109	X	X	X	
J	Jump to EEPROM address	110	X	X	X	
i	Jump to EEPROM address *n* times	110	X	X	X	
Z	Erase EEPROM Contents	110	X	X	X	
* or !	Abort current IEEPROM Program	111	X	X	X	
D	Dump EEPROM Contents	111	X	X	X	
L	Check Input Line and then jump to EEPROM address	112	X	X	X	
W	Wait for motion complete	112	X	X	X	

Command	Description	See page	All MicroMC	*All* MicroLC	SSNEMA17	Notes
k	Call subroutine	113	X	X	X	
i	Return from subroutine	113	X	X	X	
t	Text Display	113	X	X	X	Only valid in the v105 and up
^	Toggle Output Port Line	115	X	X	X	
<	Wait for Input Port Trigger	115	X	X	X	
=	Wait for Input Port Change	115	X	X	X	
ADC Commands						
o@aI	Initialize ADC System	124	X	X		Requires ADC Add-on
o@a?	ADC System Check	124	X	X		Requires ADC Add-on
o@aR	Read ADC	124	X	X		Requires ADC Add-on
o@aS	Set ADC Collection Parameters	125	X	X		Requires ADC Add-on

Command	Description	See page	All MicroMC	*All* MicroLC	SSNEMA17	Notes
Additional ARM board Configuration Commands						
#CS0 to #CS1	Set main power-up registers	117		X	X	
#CD	Display main power-up registers	119		X	X	
#CE	Erase configuration area of memory	119		X	X	
#CP	Store main power-up registers in memory	119		X	X	
#AS0 to #AS19	Set motor power-up registers	120		X	X	
#AD	Display motor power-up registers	121		X	X	
#AE	Erase configuration area of memory	122		X	X	
#AP	Store motor power-up registers in memory	122		X	X	
#BD	Display board information	122		X	X	
Miscellaneous						
o@LD	Switch back to old Simple Step motor movement	46	X	X	X	

Table 389 Command Summary

INDEX

Q

R

S

T

X