Command and Control Reference Guide

M3-LS-3.4 Linear Smart Stage

This Command & Control Guide for the M3-LS-3.4 Linear Smart Stage explains how to connect it, set it up and communicate with it, through both the SPI and I2C interface buses.

New Scale Technologies

Table of Contents

1 Introduction

The M3-LS-3.4 Linear Smart Stage is an electromechanical device that moves and positions moderate payloads (up to 100 grams). Uniquely for such devices, the controller is built right into the stage, allowing a much more compact system and very simple integration. The patented piezoelectric SQUIGGLE® micro motor moves the carriage with 0.5 µm resolution over a 15-mm range, for precise, repeatable positioning. Absolute encoding removes the need to home the stage on power-up, eliminating errors and disruptions in processes and experiments. Precision ball bearings in a linear guide provide smooth, repeatable motion with excellent lateral stability. The device operates at 6V DC and communicates through a 3.3V SPI or I2C serial interface.

The M3-LS-3.4 is a complete closed-loop PID (proportional–integral–derivative) controller system comprising of a Squiggle® motor, Tracker NSE-5310 encoder with 0.5 micron resolution, driver IC and control electronics. All needed functionality and flexibility has been integrated into the M3-LS Linear Smart Stage, eliminating the need for users to develop their own PID controller system.

The commands, from the standard New Scale Technologies controller command set, provide simple Closed-Loop and Open-Loop control, while the internal processor in the M3-LS performs the PID processing, motor control and encoder monitoring.

Customers can "drop" the M3-LS into their system and run it immediately for testing and prototyping, using the M3 USB Interface and New Scale Pathway™ software. Having proven it, they can integrate the M3-LS into the final system, using an SPI or I2C interface.

2 Product Family

Primary Products	Part No.	Description
M3-LS-3.4-15	06224-3-0000	Linear Smart Stage with 15 mm of Travel
DK-M3-LS-3.4-15	06442-3-0000	Developer's Kit, M3-LS-3.4 Smart Stage. Inc: USB to M3-HDMI interface electronics, mounting hardware kit, and pathway software.

3 Terminology

Factory Limits

Factory Limits prevent the M3-LS carriage from exceeding the mechanical range of the device, causing damage. Even if Soft Limits are disabled, Factory Limits are always enforced, in both open- and closed-loop modes, and cannot be changed.

Soft Limits

Soft Limits may be set by the user to explicitly limit the travel range of the carriage. The carriage will not travel outside of these limits unless they are turned off (factory limits will still apply).

Absolute Encoding

The absolute position of the M3-LS carriage is determined on power-up, using an internal absolute position sensor and encoder. The home (zero) position is factory calibrated to about 0.5 mm from the reverse hard stop. The maximum forward position (15000 um for the M3-LS-3.4-15) is nominally about 0.5 mm from the forward hard stop. Travel beyond those limits (i.e. 0 – 15000 um, or more depending on stage type) is not allowed.

Relative Position

Following a power cycle, the reported position is the absolute position relative to the zero established at the factory, which is the reverse limit of travel. The user can set new zero position with the <07> command. Until the next power cycle (or next issuance of command <07>) the reported position will be relative to the where the stage was located when the zero command was sent. All move to target commands will be relative to that location. This user-settable zero position is not saved and must be reestablished after each power cycle. Note that command <07> actually toggles the zero position. That is, if sent a second time, the stage reports absolute position. If sent a third time, the stage's current location becomes the new zero position.

Closed-Loop Mode

Closed-Loop Mode is the default mode of operation on power-up, when the M3-LS system generates a position error for the current position relative to the target position. This error is then multiplied by the PID (Proportional-Integral-Derivative) parameters to achieve the set closed-loop speed and acceleration targets to reduce it to zero. The user need only tell the M3-LS system where to move, and the system does the rest. Closed loop commands include "Move to Target" and "Move Step".

Open-Loop Mode

In Open-Loop Mode, the M3-LS doesn't use the PID parameters or the encoder position to control how far or fast it moves (apart from enforcing travel limits). The motor can be given run/stop commands and will move at the currently selected open-loop speed setting. The M3-LS will internally issue a stop if the stage reaches one of its travel limits.

Maintenance Mode

Maintenance Mode is also normally on when in Closed-Loop Mode. If an external disturbance moves the M3-LS carriage, a position error will be generated, and the PID controller will take over to move it back to the target position. Maintenance Mode will be turned off, however, if a motor step command (<05…>) is received. This is an Open-Loop command that turns on the motor for a specified time interval. Maintenance mode will then be turned back on following the next Closed-Loop Mode command (e.g. <06…> or <08…>).

SPI Bus

The Serial Peripheral Interface Bus or SPI (pronounced "ess-pee-i" or "spy") bus is a synchronous serial data link standard, developed by Motorola, which operates in full duplex mode. Devices communicate in master/slave mode, in which the master device initiates the data frame. Multiple slave devices are allowed with individual slave select (chip select) lines. Sometimes SPI is called a "four wire" serial bus, contrasting with three, two, and one wire serial buses. The maximum clock rate that may be used for the M3-LS is 2 MHz.

Additional information on the SPI Interface is available at:

http://ww1.microchip.com/downloads/en/DeviceDoc/spi.pdf

The above document describes four polarity modes for SPI (0, 1, 2 or 3), and the M3-LS operates in mode 1. That is, the clock (SCK) is low in the idle state and the data output line (SDO) is set on the rising edge of the clock. The maximum clock rate that may be used for the M3-LS is 2 MHz.

I2C Bus

The I2C (Inter-Integrated Circuit) Bus, developed by Philips, is a multi-master, serial, single-ended bus that is used to attach low-speed peripherals to a motherboard, embedded system, or cellphone.

The I2C specifications and an application note are available at:

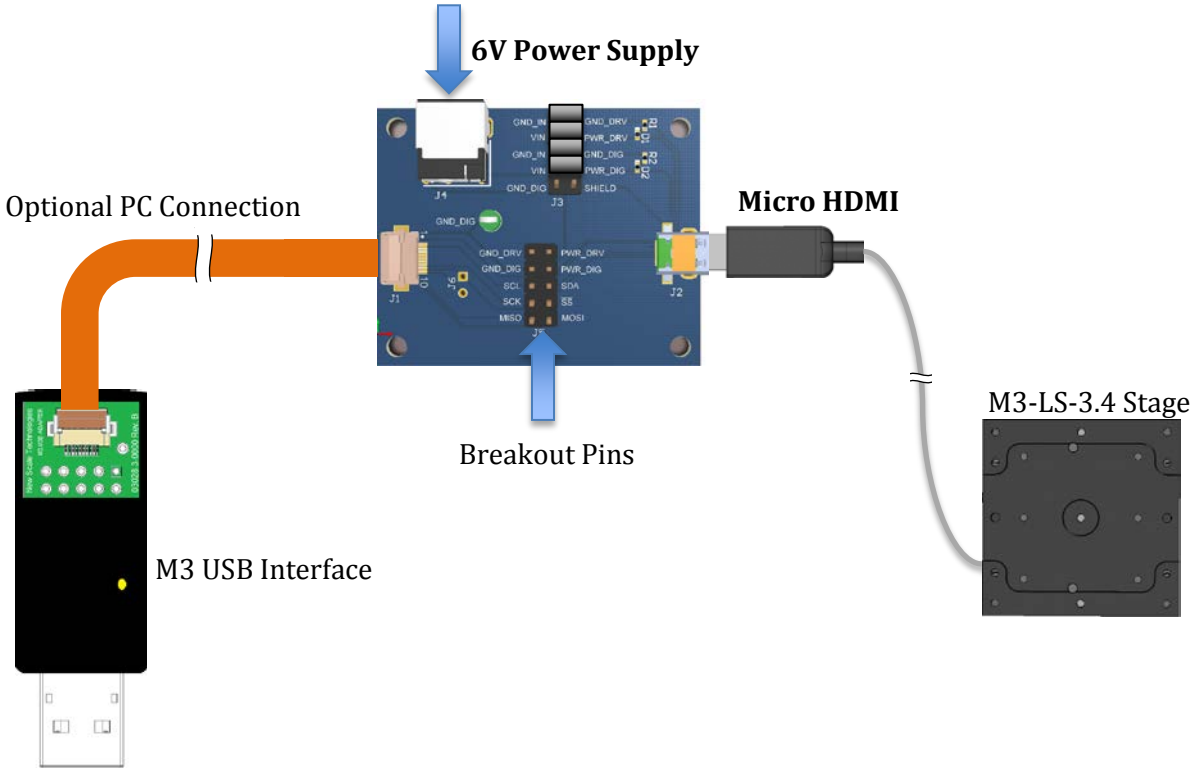
http://www.nxp.com/documents/user_manual/UM10204.pdf

http://www.nxp.com/documents/application_note/AN10216.pdf

4 Connecting the Smart Stage

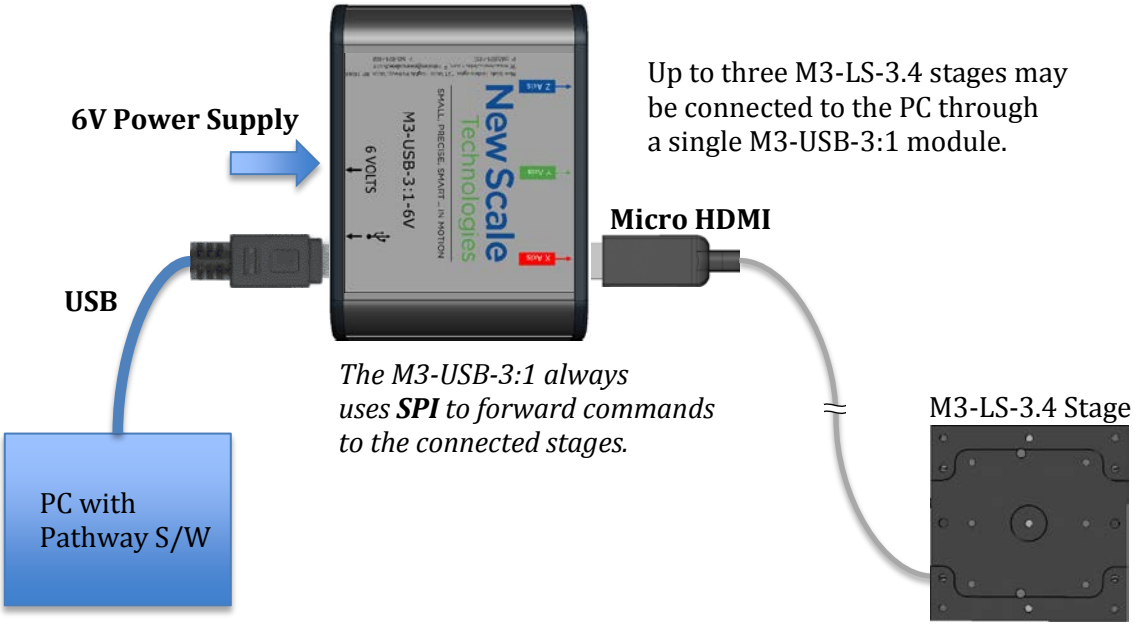
The M3-LS-3.4 has a micro HDMI plug which can connect to a single axis breakout board adapter to a USB multiplexer.

4.1 Connecting via the Micro HDMI Adapter



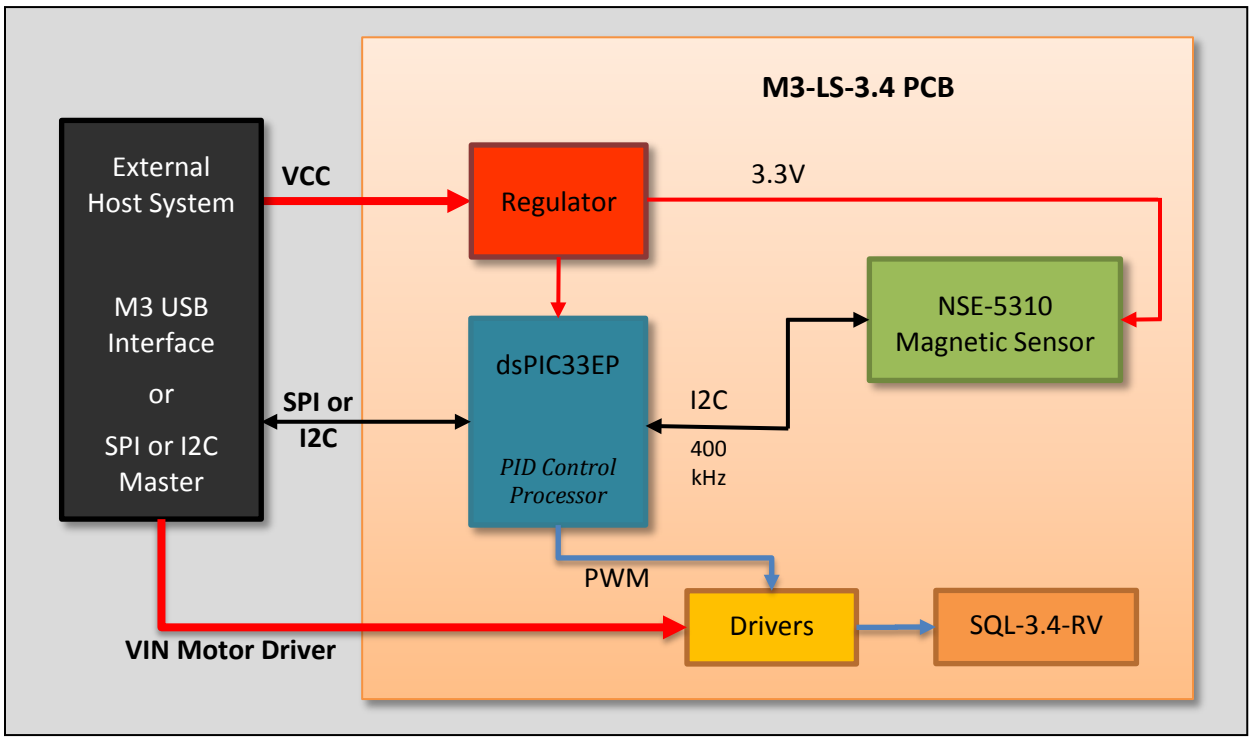
Connecting to the M3-LS-3.4 with an M3 USB Interface via a Breakout Board

4.2 Connecting via the M3-USB-3:1 Multiplexer



Connecting to the M3-LS-3.4 via a USB 3:1 Multiplexer

4.3 M3-LS-3.4 System Diagram



For more details about the electrical connections, please see the integration guide.

5 Initial Set-up Example

This sequence of commands for an initial set-up provides an exercise that includes frequency calibration, setting soft travel limits, activating the soft limits, moving in forward steps to the forward soft limit and moving in reverse steps to the reverse soft limit.

Step	Description	Command Name	Command
1	M3-LS is mounted to user system and connected to the load. The M3-LS is powered on at this time.	N/A	N/A
2	Although the motor drive frequency is preset to a default value, it will vary with temperature and load. To maintain optimal performance, always issue a Calibrate Frequency command upon power up or if there is a significant change in ambient temperature or load. Note that this will move the carriage by up to 250 µm.	**Calibrate Frequency**	<87 5>
3	*This step is optional (only if you need to restrict travel).* Set soft travel limits based on the range of travel desired (default limits are 0 – 15 mm). Here we set the travel range to 10 mm (2 mm – 12mm). The forward limit is the number of 0.5 micron encoder counts corresponding to 12 mm distance from home = 12 / 0.0005 = 24000 (0x5DC0 HEX). 1. Set Forward Limit to 12000 (0x5DC0 HEX) 2. Set Reverse Limit to 2000 (0x00FA0 HEX) 3. Set Limit Active Window to 2 (0x0002 HEX) ***These settings are stored in volatile memory and are only remembered while power is on. Reinitialize them for each power cycle.*** ***Travel limits are always based on absolute position.***	**Set Soft Travel Limits** **Activate Soft Travel Limits**	<46 00001F40 00000FA0 0002> <47 1>
5	Verify travel range by moving forward to 4 mm and in reverse to 2mm. A "Move Step" command is most convenient here. As we are starting at 3mm (the center position, send move step forward commands to cover the desired range of travel. Any step size can be used but for this example, we will step forward in 50-micron increments. This corresponds to 100 encoder counts or 0x0064 HEX. If repeating the same step size, the size value may be omitted. The last transmitted step size is retained until the next power cycle or processor reset.	**Move Step Forward (and Set Step Size)** **Move Step Forward** **Move Step Reverse**	<06 1 00000064> <06 1> (Repeat…) <06 0> (Repeat…)

6 Communication Standards Supported

6.1 SPI Details

6.1.1 SPI format

- SCK (clock input) active high
  - Low level in the idle state
- /SS (slave select) active low, has 10K pull-up Resister
  - Low = Transmission and reception are enabled
  - High = SDO pin is no longer driven and tri-stated even if the module is in the middle of a transmission
- SDI (Data In)
  - 8-bit data input
- SDO (Data Out)
  - 8-bit data byte clocked out on low to high clock transition

The M3–LS uses SPI polarity mode #1. That is, the clock (SCK) must be low in the idle state and the data output line (SDO) must be set on the rising edge of the clock. The maximum clock rate that may be used for the M3-LS is 2 MHz.

6.1.2 SPI Operation Summary

1. Host drives the /SS pin low to enable transmission and reception of data
   - The slave select pin may be grounded to keep the SPI port active at all times
2. An ASCII command byte can then be clocked in by the host, one byte at a time
3. An 0x01 HEX will be clocked out by the M3-LS during reception of each byte
4. After an entire ASCII command has been sent, the host must continue to clock out 0x01
5. The M3-LS will in turn reply with 0x01 until the entire command has been interpreted, and a formatted reply message is done and is ready to be sent out.
   Please review the next section on timing
6. All reply messages from the M3-LS will be terminated with a carriage return
7. Once the host system receives the carriage return, the host can terminate clocking
8. Repeat SPI operation to send out another command

6.1.3 SPI Command Reply Timing

As stated in item 5 above, the M3-LS returns 0x01 until the command is completely processed, and the reply is loaded into an internal RAM buffer. Once a value other than 0x01 is received, the M3-LS is returning the contents of the RAM buffer

However, there can be a delay in the time it takes the M3-LS to move the next reply byte from the RAM buffer to the SPI shift register. That is, after each byte is shifted out through the SPI bus, an SPI interrupt will be pending. If the M3-LS is not already processing another interrupt, the SPI interrupt will be processed immediately. So, the typical time it takes the M3-LS to reload the SPI shift register is 6 µsec. But this is not always the case. Due to conflicts with other interrupts, this delay can be as long as 40 µsec. Therefore, when fetching a reply from the M3-LS, it is recommended that each SPI exchange be spaced by 60 µsec in order to guarantee that duplicate bytes are not received.

6.2 I2C Details

The I2C interface was not intended for long cable lengths, but for "IC-to-IC" communication within a circuit board, but it does work over cable lengths of less than 12 inches. Pull-up resistors and filter capacitors on the host system SCL (clock) and SDA (data) signals will increase reliability and resistance to noise and interference.

Refer to the I2C specifications for details and limitations of this bus architecture. The I2C specifications can be obtained here:

http://www.nxp.com/documents/user_manual/UM10204.pdf

http://www.nxp.com/documents/application_note/AN10216.pdf

The following is a link to the I2C implementation in the dsPIC33EP processor used in the M3-LS module.

http://ww1.microchip.com/downloads/en/DeviceDoc/70000195f.pdf

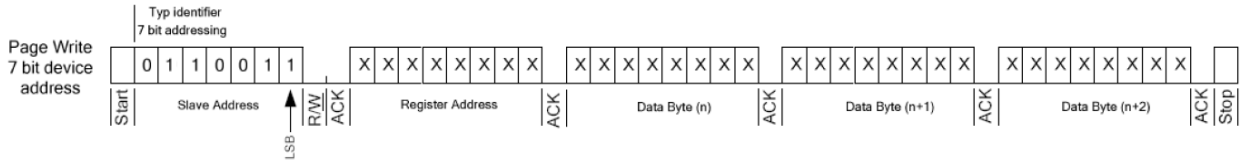
6.2.1 I2C Format

It is assumed that the host system is providing pull-up resistors for the SDA (data) and SCL (clock) lines of the I2C bus, and that the logic high will be 3.3V.

A HIGH to LOW transition on the SDA line while SCL is HIGH is the start condition for the bus.
A LOW to HIGH transition on the SDA line while SCL is HIGH is the stop condition.

Every byte put on the SDA line must indeed be 8-bits long. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first.

Data-transfer-with-acknowledge is obligatory. The master generates the acknowledge-related clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse. However, there is one exception. When the master is reading data from the slave, the master acknowledges all but the last byte to be read (i.e. the carriage return).



The I2C interface supports both read and write operations. The slave's 7-bit address is 33 (hexadecimal). However, since these bits are shifted to the left, the address value sent by the host is 66 (write) or 67 (read).

6.2.2 Sample I2C Commands

Since the M3 modules are based on the existing family of MC-3x00 controllers from New Scale Technologies, the supported ASCII commands all have the same syntax.

Details of every command are provided in a later section; however, these samples are provided to illustrate how the communication process works.

The M3 series module supports clock stretching and will automatically clock stretch (pull the SCLx bus line low) after each received data byte, allowing the software more time to move the data from the buffer. Clock stretching will also occur after the entire command has been received, and will be released only after the command has been interpreted and a formatted reply is ready to be sent.

Move to Target

Given a 0.5 um resolution encoder, a move to target command (08) to position 3000 um would consist of the following string (◊ represents the carriage return character).

Shown in ASCII:

<08 00001770>◊ Note: 1770 is hexadecimal notation for 6000 encoder counts (3000 um)

Sequence of byte values (in hexadecimal) that are written by the host to its I2C output register (corresponding ASCII value is shown underneath):

66 3C 30 38 20 30 30 30 30 31 37 37 30 3E 0D

< 0 8 0 0 0 0 1 7 7 0 > ◊ (corresponding ASCII value shown underneath)

After receiving the full command the M3-LS will hold the clock line (SCL) low until a reply is ready. To query the reply, the host would send an address byte with bit 0 high to indicate a read operation, by a series of byte reads, until a carriage return is detected (the byte values shown in italics are transmitted by the M3 module):

67 3C 30 38 3E 0D

< 0 8 > ◊

The reply should be read out to confirm that the command was received. When reading data back, the host must acknowledge all but the last byte (i.e. after receiving the carriage return character the host should release the data line).

Query Motor Status

The host may query position as well as motor status.
The query status command, shown in ASCII is: <19>◊

The sequence of byte values (in hexadecimal) that are written by the host to its I2C output register:

66 3C 31 39 3E 0D

< 1 9 > ◊

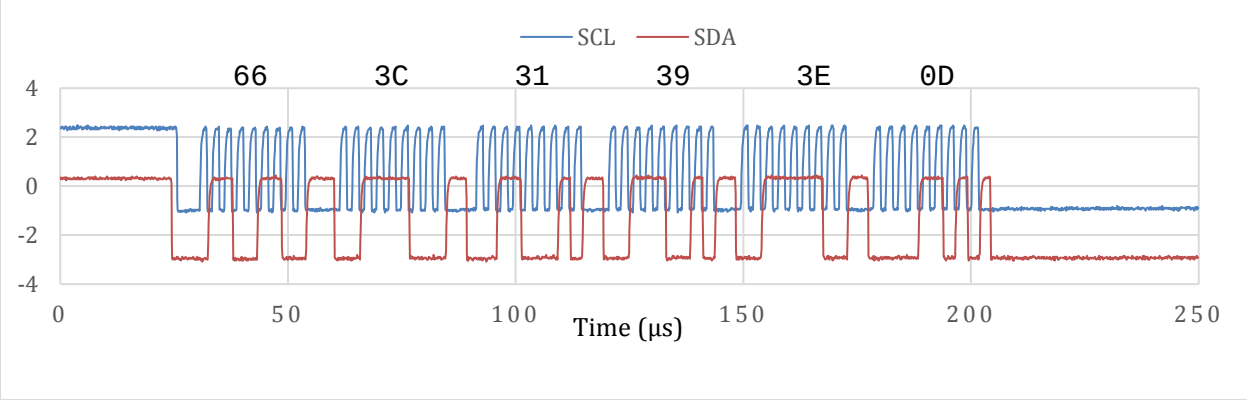
To query the reply, the host would send an address byte with bit 0 high to indicate a read operation by a series of byte reads until a carriage return is detected (the byte values shown in italics are transmitted by the M3 module):

67 3C 31 39 20 30 30 38 32 3E 0D

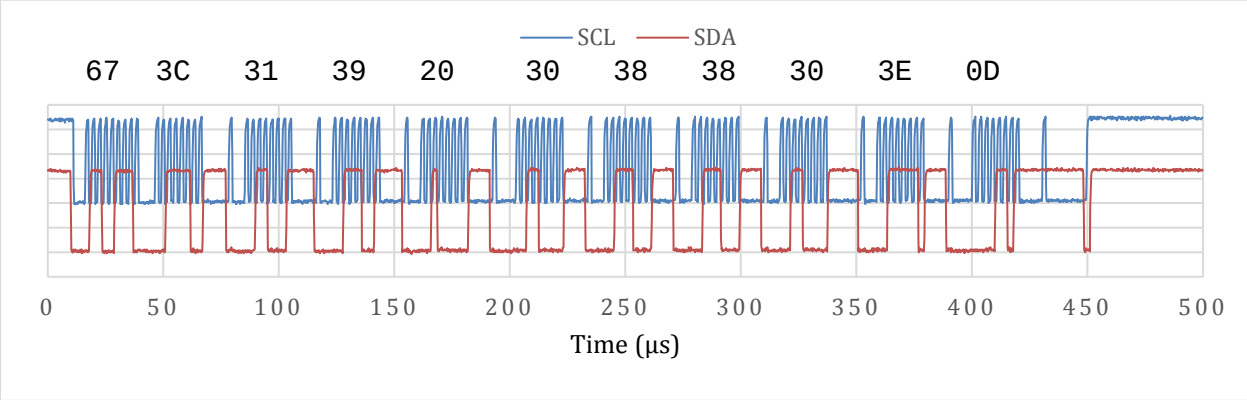
< 1 9 0 0 8 2 > ◊

The meaning of the value 16-bit status value returned by the M3 module is described in the section covering command 19.

Below are plots of the SDA and SCL lines as the host system transmits the query status command and then reads the reply.



Host Transmission of Command <19>



Host Query of Command Reply

Note that the interval between the 8th and 9th clock pulse on each received byte is due to the latency of the host's interrupt service routine. Once the byte is read by the host the 9th clock pulse is issued. Note that on the last byte (0D) the SDA line is released for that 9th pulse.

7 Command Conventions

- All commands must be preceded and ended with "<" and ">" characters (for example, <01>).
- All commands are terminated with a carriage return character.
- Brackets indicate optional values, but are not entered with the commands. For example, the command <04 D [TTTT]> could be entered: <04 1> or <04 1 1234> where 1 represents D and 1234 represents TTTT.
- Commands use 7-bit ASCII characters only.
- All hexadecimal (hex) numbers sent or received use only capital letters (e.g., 0xF4 = F4).

8 Commands

01: Read the Firmware Version

This command retrieves the version of the controller firmware.

Send to Controller	Receive from Controller	Usage
<01>	<01 A "VER X.X.X"…[INFO]>	**A** = 1 for M3 **VER X.X.X** is the controller firmware version.[INFO] is further M3 information text.

03: Halt the Motor

This command halts motor motion regardless of where the movement command was issued. If in closed-loop mode, the current position is now the target position.

Send to controller	Receive from Controller
<03>	<03>

04: Run the Motor

This command runs the motor. The motor will continue to move until command <03> is received or until an optional time value elapses. In closed-loop mode the motor will run according to PID, speed and acceleration settings.

Send to Controller	Receive from Controller	Usage
<04 D[TTTT]>	<04>	If **D** = 1, motor runs forward. If **D** = 0, motor runs in reverse. **TTTT** is an optional run duration, entered in 0.1-second units. (Hex format)

Examples

To…	Send to controller	Receive from Controller
Run the motor forward	<04 1>	<04>
Run the motor in reverse for two seconds	<04 0 0014>	<04>

05: Move the Motor in Timed Open-Loop Steps

This command sends one or more bursts of resonant pulses to the motor at 100 Hz (10 ms period) or at the period indicated by the **PPPP** parameter.

Send to Controller	Receive from Controller	Usage
<05[D[SSSS PPPP TTTT]>	<05>	If **D = 1**, motor runs forward. If **D = 0**, motor runs in reverse. **SSSS** is an optional number of steps to take.* **PPPP** is an optional step interval in 3.2 µsec units.* The step interval is the time between the start of each step. **TTTT** is optional step duration in 3.2 µsec units. The step duration is the time that the resonant pulses are active, which must be less than the step interval.

* 4-digit hex format. The µs unit value could change in future versions of the controller. It is recommended that you use command **<52>** to query actual units.

Example

To…	Send to Controller	Receive from Controller
Run the motor 100 steps forward with an interval of 20-ms and a duration of 10-ms per step	<05 1 0064 186A 0C35>	<05>

06: Move Closed-Loop Step

This command adds or subtracts the specified step size (in encoder counts) to the current target position, and then moves the motor to the new target at the previously defined speed.

Note: This command will return an illegal command error <24> if issued while the axis (motor) is in open-loop mode. See command <20> to select closed-loop mode.

Use command <40> to set the closed-loop speed.

Send to Controller	Receive from Controller	Usage
<06 D[SSSSSSSS]>	<06>	If **D = 1**, motor runs forward. If **D = 0**, motor runs in reverse. Set **D** to **N** to set the step size without moving the actuator. **SSSSSSSS** (HEX) is the number of encoder counts to advance/backup from the current target position. If not specified, the previously stored step size will be used.

Example

To…	Send to Controller	Receive from Controller
Run the motor forward 100 encoder counts from the target position	<06 1 00000064>	<06>

07: Toggle Absolute/Relative Position

This command toggles the relative or absolute position modes. If the M3 is currently in Absolute position mode, then the current reported position is set to 0.

Send to controller	Receive from Controller
<07>	<07>

08: Move to Target

This command sets a target position and moves the motor to that target position at the speed defined by command <40>.

Note: This command will return an illegal command error <24> if issued while the axis (motor) is in open-loop mode. See command <20> to select closed-loop mode.

Send to Controller	Receive from Controller	Usage
<08[TTTTTTTT]>	<08> or <08 TTTTTTTT>	**TTTTTTTT** is the target position in encoder counts. (HEX). If **TTTTTTTT** is omitted, then the last target will be returned.

Example

To…	Send to Controller	Receive from Controller
Set target position to 100 microns based on 0.5-micron resolution. 100/0.5 = 200 (0x000000C8 HEX)	<08 000000C8>	<08>
Set target position to -1000 microns based on 0.5-micron resolution. -1000/0.2 = -2000 (0xFFFFF830 HEX)	<08 FFFFF830>	<08>

09: Set the Open-Loop Mode Speed

This command sets the open-loop speed of the motor. This value is not saved to internal EEPROM. Use command <20> on to select open-loop mode.

Send to Controller	Receive from Controller	Usage
<09[XX]>	<09> or <09 XX>	**XX** is the speed command of the value 1 to 255 (0x01 to 0xFF) (HEX) where the speed is the ratio of (XX / 255) x 100%. If **XX** is omitted, then the current setting will be returned.

Example

To…	Send to Controller	Receive from Controller
Set the speed to (0 x 80 / 0 x FF) x 100% == (128 / 256) x 100% = 50%	<09 80>	<09>

10: View Closed-Loop Status and Position

This command is used to view the motor status and position.

Send to Controller	Receive from Controller	Usage
<10>	<10 SSSSSS PPPPPPPP EEEEEEEE>	**SSSSSS** is the motor status. (6-digit hex format, 24-bit unsigned integer) See "Motor Status Values," next. **PPPPPPPP** is the absolute position in encoder counts. (8-digit hex format, 32-bit signed integer) **EEEEEEEE** is the position error in encoder counts.** This value is the distance between where the motor is located and where it is supposed to be located.

Motor Status Values

Bit	Description	Values
0	Reserved	N/A
1	Motor direction	0 = Reverse 1 = Forward
2	Running	1 = Motor is running
3	Motor driver not responding	0 = If motor driver communication OK 1 = If motor driver not responding

Bit	Description	Values
4	Numbered burst mode	1 = Fixed number of bursts in progress
5	Timed run	1 = Timed free run in progress
6	Multiplexed axis	N/A
7	Host control Established	1=Command <01> received since last power cycle or reset.
8	Reserved	
9	Forward limit	1 = Forward travel limit reached
10	Reverse limit	1 = Reverse travel limit reached
11	Motor burst or amplitude mode	0 = Amplitude mode (always used in closed-loop mode). 1 = Burst mode (200 Hz)
12 – 14	Reserved	N/A
15	Background Job Active	1 = A background job is in progress 0 = No background job in progress.
16	Encoder Error	1 = A change in encoder position exceeded 1023 counts (0.5mm) in a 2 ms interval. This could cause a loss in absolute position. The error is cleared by sending <42 A> to reestablish the home position. 0 = No encoder error
17	Zero reference enabled	1=Reference enabled. If enabled, the M3-L will reset the tracker zero position when the home reference sensor indicates the required field threshold has been reached (used internally during execution of command <42 A>
18	Motor on target	1 = Encoder position is within the preset tolerance of the target (typically 2 encoder counts or 1 um).
19	Motor moving toward target	1 = Motor is moving toward a target position; appears after command <**08>** or move step command <06>. Once the target is reached, bit 19 is set to zero.
20	Maintenance mode enabled	1 = Controller will actively hold the last target position **Note**: If bits 20 and 21 are set 1 and bit 18 is set to 0, the controller is in the process of moving back toward the last targeted position.

Bit	Description	Values
21	Closed loop enabled	1 = Motion commands use the encoder for feedback
22	Motor accelerating	1 = The motor is accelerating to the desired velocity (set at the start of closed-loop motion) 0 = Required motor speed is reached, motor is decelerating, or motor is stopped
23	Stalled	1 = The position error exceeds the stall detection threshold while the motor is moving

19: Read Motor Flags from the Controller

This command reports internal flags used by the controller to monitor motor conditions.

Send to Controller	Receive from Controller	Usage
`<19>`	`<19 XXXX>`	**XXXX** (16-bit hex format) is the status flags returned. See "Motor Status Values" from command <10> (first 16 bits only).

20: View and Select the Open- or Closed-Loop Drive Modes

This command sets the drive mode for the M3. The M3 will always default to closed-loop mode on power up.

Send to Controller	Receive from Controller	Usage
`<20 X[ IIII]>`	`<20 X IIII>`	If **X** = 0, open-loop drive mode is selected. If **X** = 1, closed-loop drive mode is selected. If **X** = R, the controller will report the current drive mode. **IIII** - if specified, the closed-loop control interval (in 3.2 or 1.6 µsec units, use command <52> to query the timer units) **Notes**: The open-loop motion commands (<03>, <04> and <05> will work in closed-loop mode. Command <10> will continue to work in open-loop mode.

23: Illegal Command Format

If the controller returns <23>, the command was improperly formatted. Verify that the proper begin- and end-terminating characters are present.

24: Illegal Command

If the controller returns <24>, an illegal command was attempted.

40: Set the Closed-Loop Mode Speed

This command sets the closed-loop speed of the current axis (motor). Note that the controller works in 0.5-micron encoder units. It also works in increments of the closed-loop interval (time between each processing of the current encoder position) rather than in seconds. So, the host must convert the required velocity and acceleration into those units.

If <40> is sent with no parameter, it will report the current settings.

The default value preset in the factory is 4000 um/s.

These values can be saved to internal EEPROM so that they are the default values on power up by issuing command <74>.

Note: This command will return an illegal command error <24> if issued while the axis is in open-loop mode. See command <20> to select closed-loop mode.

Send to Controller	Receive from Controller	Usage
<40 SSSSSS CCCCCC AAAAAA IIII>	<40>	**SSSSSS** is the speed of the axis (motor) in encoder counts per interval.* **CCCCCC** is the cutoff (minimum) speed of the axis (motor) in encoder counts per interval.* **AAAAAA** is the acceleration/deceleration of the axis (motor) in encoder counts per interval.* **IIII** is the interval duration in closed-loop interval units.**

* 6-digit hex format. The last two hex digits are a fractional encoder count.

** 4-digit hex format.

Example

Consider the following settings (note that for slow speeds, < 1µm/s, the interval count may have to be increased:

Encoder Resolution	500 nm	(EncRes)
Closed Loop Interval	1000 µs	(Interval)
Interval Count	1	(IntervalCount)
Velocity	4000 µm/s	(Velocity)
Acceleration	20000 µm/s²	(Accel)
Cutoff Velocity	20 µm/s	(Cutoff)

```
SSSSSS = (Velocity/(EncRes/1000)) * 256 * (IntervalCount*(Interval/1000000))
CCCCCC = (  Cutoff/(EncRes/1000)) * 256 * (IntervalCount*(Interval/1000000))
```

```
AAAAAA = (  SSSSSS/(Velocity/Accel))    * (IntervalCount*(Interval/1000000))

SSSSSS = 2048 decimal or 800 hex
CCCCCC = 10   decimal or   A hex
AAAAAA = 10   decimal or   A hex
```

Therefore, the command to be sent is: **<40 000800 00000A 00000A 0001>**

*An Excel spread sheet is available in the **Examples** subdirectory of the flash drive (or downloaded zip file) containing the Pathway s/w. This spread sheet illustrates the above calculation and allows you to enter alternative velocity, acceleration and cutoff values.*

41: Set Position Error Thresholds and Stall Detection

This command is used to enable stalled motor detection and set position thresholds for the current axis (motor). By-default stall detection is not enabled on the M3 but may be used if necessary. Improperly setting stall detection parameters may result in false stall detects halting movement.

These values are saved to internal EEPROM.

Send to Controller	Receive from Controller	Usage
<41 X EEEEEE IIIIII>	<41 X EEEEEE IIIIII>	If **X** = 0, stalled motor will not be detected. If **X** = 1, stalled motor will be detected. If **X** = R, the controller will report the current stall mode. **EEEEEE** is an optional stall position error threshold in encoder counts.* This value sets the maximum allowable difference between where the motor is currently located and where it is supposed to be located. **IIIIII** is an optional stall incremental error threshold.* This value is used at the end of a move and during position hold and is the sum of encoder errors and the incremental coefficient. The incremental error threshold is used when the motor is close to the required target (within position error threshold) but is unable to reach the target.

*6-digit hex format.

Example

To…	Send to Controller	Receive from Controller
Enable stall detection with a position error threshold of 100 encoder counts	<41 1 000064>	<41 1 000064 002710>

43: View and Set Closed-Loop PID Coefficients

This command is used to view and set the closed-loop PID coefficients in the motor's EEPROM. Closed-loop PID coefficients are used in closed-loop mode to adjust the motor drive signal gain based on how far the motor position is from the target position.

These parameters are calibrated at New Scale Technologies for your M3 system stable operation but may be changed if needed. Adjusting these parameters can cause unstable operation if done improperly.

These values are saved to internal EEPROM.

Send to Controller	Receive from Controller	Usage
<43 PPPP IIII DDDD>	<43 PPPP IIII DDDD>	To view the current PID coefficients, use this command without any parameters. The following parameters are optional: **PPPP** = P coefficient.* **IIII** = I coefficient.* **DDDD** = D coefficient.

*4-digit hex value (16-bit).

Example

To…	Send to Controller	Receive from Controller
View the PID coefficients The coefficients in this example have the following decimal values: P = 300, I = 5, D = 0.	<43>	<43 012C 0005 0000>

46: View and Set Forward and Reverse Soft Limit Values

This command is used to view and set motor travel limits for the M3. Once the motor/stage reaches or exceeds the travel limit, it will be stopped and the appropriate limit flag will be set.

NOTE: Travel limits are always set based on the absolute position even if command <7> was used to set a new relative zero position. Factory limits are always active and cannot be changed. To activate travel limits, use command <47>

These values are saved to internal EEPROM.

Send to Controller	Receive from Controller	Usage
<46 FFFFFFFF RRRRRRRR WWWW>	<46 FFFFFFFF RRRRRRRR WWWW>	**FFFFFFFF** is optional the forward limit*, in encoder units. **RRRRRRRR** is the optional reverse limit*, in encoder units. **WWWW** is the optional distance** (in encoder units) before each travel limit position at which the limit flag will be remain asserted once it has been tripped. This value appears on the Limit Active window.

* Signed 32-bit hex format.

** Unsigned 16-bit hex format.

Example

To…	Send to Controller	Receive from Controller
Query the current travel limit.	<46>	<46 00002EE0 00000000 0004>
Set the forward and reverse travel limits to 1000 µm and 500 µm, assuming a 500 nm encoder. This also sets the Limit Active window to 1 µm.	<46 000007D0 000003E8 0002>	<46 000007D0 000003E8 0002>

47: Viewing, activating and deactivating soft limits

This command is used to view, activate and deactivate motor travel limits. This command can disable the soft limits but the factory limits will remain active in both open- and closed-loop modes at all times. To define travel limits, use command <46…>.

These values are saved to internal EEPROM.

Send to Controller	Receive from Controller	Usage
<47 X>	<47 X>	If **X = 0**, soft limits are not activated. If **X = 1**, soft limits are activated.

Example

To…	Send to Controller	Receive from Controller
Query the current travel limits enabled flag.	<47>	<47 1>

52: Viewing time interval units

This command is used to view M3 time interval units, which are applied to open-loop timed step command values.

Send to Controller	Receive from Controller	Usage
<52>	<52 XXX UUUU>	**XXX** is the interval value. **UUUU** is the interval units (e.g., a reply of <52 1.6 usec> means that each controller timer interval is in 1.6-µs units, so a period of 2-ms would be 1250).

54: Getting/Setting the Baud Rate Selection

This command is used to get or set the M3-LS baud rate selection. There are five possible selections as shown in the table below. When the selection is changed, it takes affect following the next power cycle.

If the baud rate setting of the M3-LS is not known, the query command may be sent via the I^2C lines (e.g. via an M3 USB Interface that has been configured for I^2C communication). ***At this time the UART communication mode is not supported.***

Send to Controller	Receive from Controller	Usage
<54 1>	<54 1 XX>	Used to query the baud rate. **XX** is the current baud rate selection: 00 19,200 01 38,400 02 57,600 03 115,200 04 250,000
<54 1 03>	<54 1 03>	Sets to M3 controller UART port to a baud rate of 115,200, which takes effect following the next power cycle.

58: Suppress/Enable Writes to "EEPROM"

Normally if a non-volatile parameter (such as stall threshold, PID coefficient, etc.) is changed, the portion of the M3-LS processor's F-RAM (Ferroelectric RAM memory) that is reserved for parameter storage is updated).

The original M3 modules used the processor's own flash memory for non-volatile parameters. This command was created to allow the host to prevent the flash memory from being written (only a local RAM cache would be updated if a non-volatile parameter was changed). It was added because

any update to processor flash memory will stall the processor for several ms (i.e. making it unresponsive to SPI or I2C).

However, since the M3-LS employs an off-chip F-RAM device for non-volatile parameter storage, processor stalls are no longer an issue. But support remains for this command and it functions identically to the original M3 modules.

This command allows the host to:

- Get/Set the current EEPROM write suppression mode.
- Get the RAM cache modified state (whether or not has it changed since the EEPROM was last written/read)
- Start an update to the EEPROM. But this will only be performed if the RAM cache has been modified.

Send to Controller	Receive from Controller	Usage
`<58 XX[ Y]>`	`<58 XX Y>`	**XX** is the sub-command 01 Get/Set EEPROM Mode 02 Get RAM Cache Modified State 03 Start EEPROM update **Y** value depends on sub-command. 01 0=Normal mode, 1=Suppress Mode 02 0=Not modified, 1=Modified 03 0=Not updated, 1=Update in Progress

Example

To…	Send to Controller	Receive from Controller
Suppress automatic EEPROM writes. Note that once turned off, automatic EEPROM writes will remain off until turned back on or power is cycled.	`<58 01 1>`	`<58 01 1>`
Enable automatic EEPROM writes.	`<58 01 0>`	`<58 01 0>`
Query EEPROM write suppression mode	`<58 01>`	`<58 01 Y>` Where Y is 0 for normal mode or 1 for suppression mode

Query EEPROM's RAM cache modified state The RAM cache is a copy of the data in the EEPROM area except when EEPROM writes are turned off. Then it can be different.	<58 02>	<58 02 Y> Where Y is 0 for unmodified or 1 for modified.
Force the EEPROM to be written immediately even if automatic writes are turned off. However, if the RAM cache has not been modified, the write to the EEPROM area will not be performed.	<58 03>	<58 03 Y> Where Y is 0 for write not being performed or 1 for write in progress.

74: Save Closed-Loop Speed Parameters to EEPROM

This command saves current run speed, cutoff speed and acceleration to EEPROM. As indicated in command 58, if EEPROM writes are turned off this command will only affect the RAM cache.

Send to Controller	Receive from Controller
<74>	<74>

87: Run Frequency Calibration

This command is used to optimize the Squiggle® motor resonant drive frequency by, on command, sweeping over a range of frequencies, centered at the specified period, and settling on the frequency at which the best motor performance was detected.

This command needs to be run at every power-up or more often in environments where the temperature is changing. When issuing an automatic frequency-calibration sweep command, the carriage may typically move 250 microns during the frequency sweep. If in closed-loop mode, the smart stage will return to the current target position automatically.

It is best to run this command when system is idle and in the forward direction for the M3-LS-3.4 Linear Smart Stage.

Send to Controller	Receive from Controller	Usage
<87 D[XX]>	<87 D XX FFFF>	**D** is the calibration movement direction and calibration type. **Bit 0:** 0 = calibration is run in reverse direction (towards M3-LS home) 1 = calibration is run in forward direction (away from M3-LS home)

		Bit 1: 1 **=** incremental frequency calibration sweep only; 0 = Frequency calibration sweep only. **Bit 2:** **1 =** Automatic Frequency calibration sweep followed by an incremental frequency calibration sweep. If Bit 2 is set, Bit 1 will be ignored. **Bit 3 to 7:** N/A, Set to 0 **XX** is optional frequency offset register. Lower nibble typically set to 5 but can be range from 0 to 7. This should be left at the factory setting. **FFFF** internal factory flags

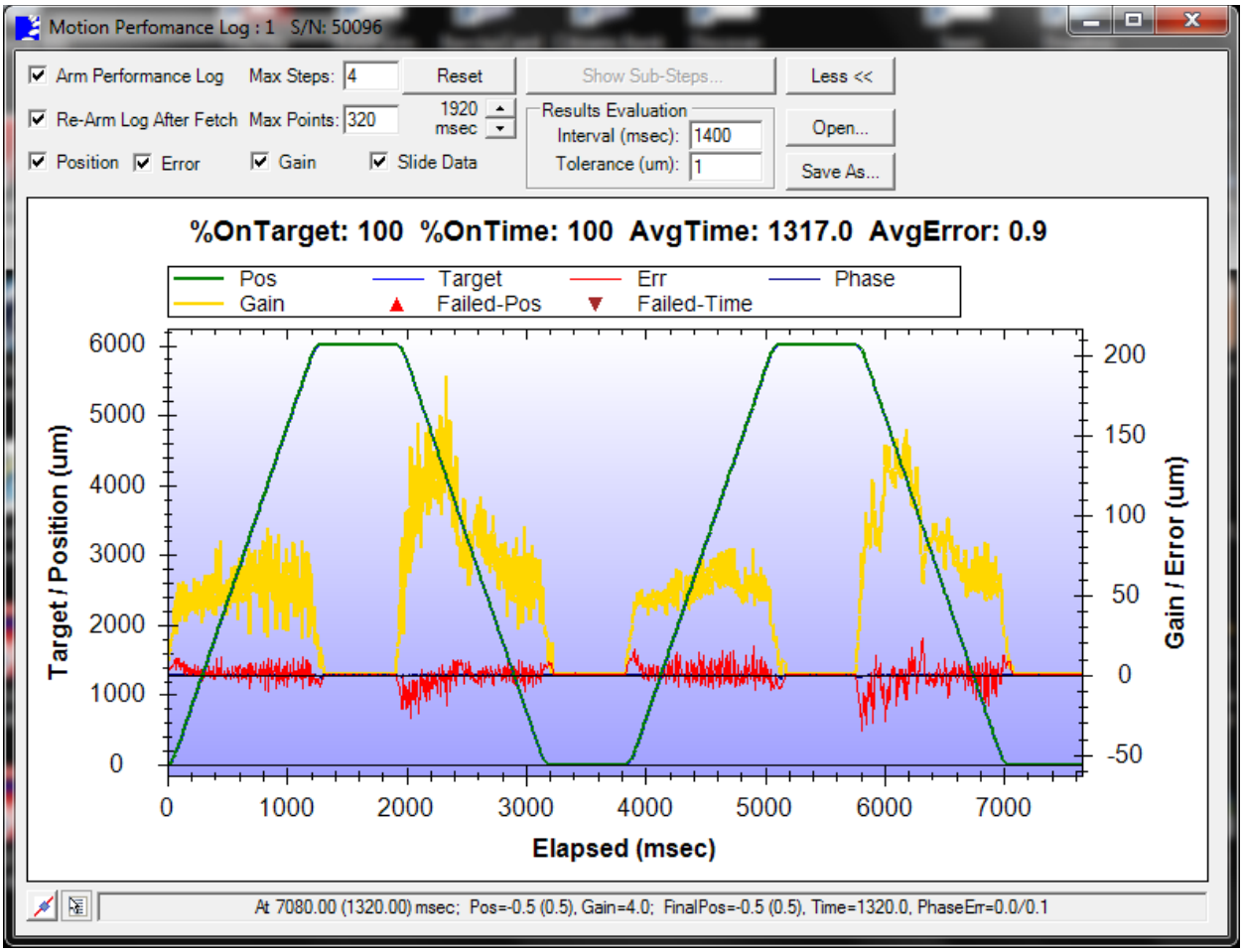
Example

To…	Send to Controller	Receive from Controller
Do an automatic frequency sweep in a forward direction.	`<87 5>`	`<87 4 75 0000>`

A9: Position Log Control and Query

To assess the step response of the M3-LS, a position log feature was added.

This log feature of the M3-LS can be configured from the New Scale Pathway™ Software (Version 2.7.4 or later) and the results may be displayed. The following figure illustrates two cycles of full travel motion. In the first cycle, the KP coefficient was 400. In the second cycle, KP was 150. In the latter case, there is less variation in the motor gain though the following error increases. If there were spots of high friction and/or load, they would be clearly visible in this plot.



Plot of Multiple Position Log Queries

Position Log Commands

Send to Controller	Receive from Controller	Usage
<A9>	<A9 X CCCC AAAA s>	*To Query Log Status* **X** 0=Unarmed, 1=Armed (and will start recording at start of next move) or armed and logging in progress (see CCCC) **CCCC** 4 digit hexadecimal number of data points collected so far (0000 on power-up) **AAAA** 4 digit hexadecimal number of possible data points (i.e. the size of the array) **s** 1 digit number (0-9) indicating the number of closed-loop intervals to skip (i.e. to not record a position value) between each recorded value. This is used to extend the time over which the log is active. If armed, it will switch to disarmed when the maximum

		possible data points are collected. In the case of the M3-LS that will be 640-ms after collection starts because its array size is 320 (0140 hex) and each closed-loop interval is 2-ms. In future versions, the maximum number of data points may increase so it is important to use the value (represented by AAAA) returned by the controller.
<A9 1>	<A9 1 CCCC AAAA s>	*To Arm Position Log* Once armed, the controller will start recording position, target and motor gain data at the start of the next move.
<A9 0>	<A9 1 CCCC AAAA s>	*To Disarm Position Log*
<A9 S s>	<A9 1 CCCC AAAA s>	*To Set Skip Value* s is the required 1 digit skip value (0-9). The default value is zero, which means the log duration is 640-ms (320 x 2ms/CL interval). The max value of 9 would make the log duration 6400-ms (320 x 10 x 2-ms) since only 1 point out of every 10 is being logged. Again, the skip feature is available in f/w V4.4.3 or later.
<AA 0 0000>	<AA 0 0000 DDDD DDDD … DDDD>	*To Query Position Data* 0000 is the 4 hex digit array offset. Since only 16 points are returned at a time, the offset would be incremented by 16 (10 hex) until all data points have been returned. DDDD is the 4 hex digit position value at that time interval (in encoder counts).
<AA 1 0000>	<AA 1 0000 GG GG GG GG … GG>	*To Query Motor Gain Data* 0000 same as above. GG is the 2 hex digit motor gain value generated at that time interval (00 = off, FF = max gain).
<AA 2 0000>	<AA 2 0000 EEEE EEEE … EEEE>	*To Query Position Error Data* 0000 same as above. EEEE is the 4-hex-digit position-error value, calculated that time interval (where Error = SetPoint – Position).

Revision History			
Date	Rev.	Description	Author
9 May 2017	1	M3-LS-3.4 Command & Control Reference Guide	TLG
3 Jul 2017	2	Corrected connector spelling error	TLG
30 Aug 2017	3	Updated M3 USB 3:1 Label	TLG
29 Sep 2017	4	Label Corrections	SRF

M3-LS-3.4 Linear Smart Stage Command & Control Guide

http://www.newscaletech.com (585) 924-4450 NSTservice@newscaletech.com

New Scale Technologies