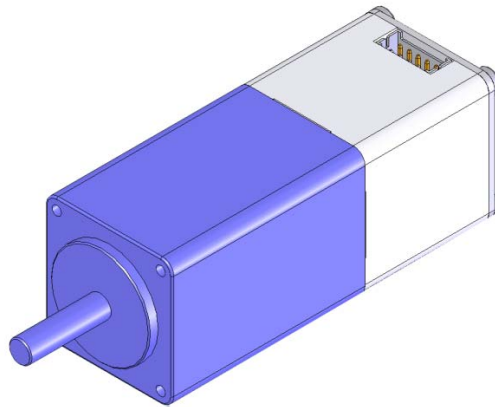


**Accuriss™28 Series**

Integrated Motor, Drive & Control System

USA28

**USER'S MANUAL**



©2010 USAutomation, Inc.  
All rights reserved

## **USAutomation™ Accuriss™ Series User's Guide**

This manual, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. The content of this manual is furnished for informational use only, is subject to change without notice and should not be construed as a commitment by USAutomation. USAutomation assumes no responsibility or liability for any errors or inaccuracies that may appear herein.

Except as permitted by such license, no part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of USAutomation.

USAutomation, Inc.  
2900 S. Gilbert Road, Ste 2  
Chandler, AZ 85286  
(714) 646-9785 Office  
[www.usautomation.com](http://www.usautomation.com)

## Contents

Contents.....	3
Revision Notes .....	4
Product Returns .....	5
Unpacking .....	5
Part Number Configuration .....	6
Dimensions and Connection Pin out .....	6
Mounting the USA28 .....	7
Mounting a Load to the USA28 .....	7
Speed Torque Curves .....	8
Accuriss Communications .....	10
Starter Kit .....	10
Command Set Table .....	13
Appendix A.....	52
Homing Algorithm in Detail.....	52
Appendix B.....	54
Device Response Packet.....	54
Command Response Modification .....	56
Appendix C .....	57
Heat Dissipation .....	57

## Revision Notes

Rev	Description	Date
1.1	Update Tables	
1.2	Corrections	
1.3	Added clarification of commands, Appendix A, B, C	
1.4	Added Accuriss Communications and Starter Kit, Corrected drawing	
1.5	Reorganize pages and Table of Contents	
1.6	Corrected errors in command set	
1.7	-Moved TOC forward -Added "Executing Commands" section -"Operand" > "Range" on command pages	12-08-09
1.8	Correct error in pin input/output configuration page 6	8-25-10

## Using This Manual

The Accuriss Series is a unique motor, drive and control all in one package. The USA28 is the 28mm version in that series and it is designed to be interactively controlled with a computer or other industrial controller. It may also be preprogrammed to run complex subroutines from programs stored in non-volatile RAM.

This manual provides the basic information necessary to unpack, set up, and configure the USA28. If additional information is required beyond what is presented here, please refer to the Support section of our website or contact USAutomation Applications Engineering.

## Product Returns

All returns for warranty or out-of-warranty repairs must first receive an RMA (Return Material Authorization) number. Please contact USAutomation Customer Service Department with information about the return and an RMA number will be issued if warranted.

Products returned to the factory will be examined and tested for failure mode and cause. USAutomation Customer Service will contact the customer with the repair cost if the required repair is out of warranty.

## Unpacking

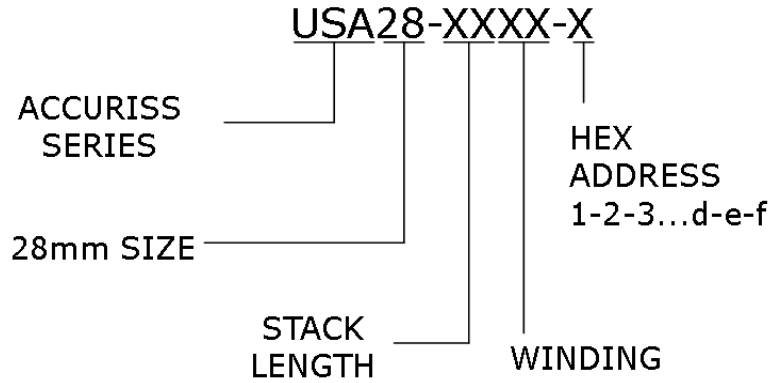
Carefully remove the USA28 from its shipping box and inspect the unit for any evidence of shipping damage. Report any damage immediately to USAutomation. Please save the shipping box for damage inspection or its use in returning product if necessary.

Please observe the following guidelines for handling and mounting of your USA28:

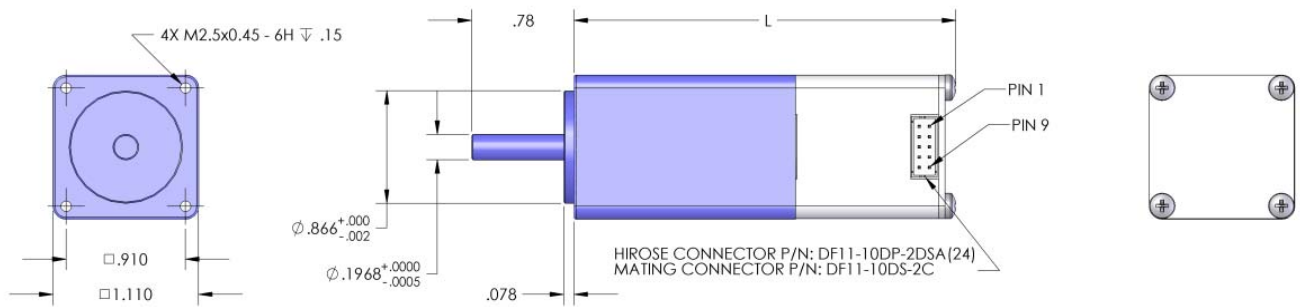
- Do not drop the unit on any hard surface or subject it to any impact loads. Dropping the unit or other impact loads may result in bearing damage or misalignment.
- Do not drill holes into the motor. Drilling holes into the unit can generate particles and machining forces that may affect the operation of the unit. USAutomation can supply the USA28 with modifications to your drawing. Please contact the factory for a quote.
- Do not expose the USA28 to mist, spray or submersion in liquids.
- Do not disassemble the USA28. Unauthorized adjustments may alter the specifications and void the product warranty.

## Part Number Configuration

The part number for the Microstage USA28 is determined as follows:



## Dimensions and Connection Pin out



Part #	"L" Dimension (in.)
USA28-2102-X	2.21
USA28-2202-X	2.72
USA28-2202-X	3.96

MOTOR PINOUT	
Pin #	Function
1	+9 to +30 VDC (+V)
2	Ground (-V)
3	I/O #1 (Input #1 or Output #2)
4	Input (input 3)
5	RS485 B
6	I/O #2 (Input #2 or Output #1)
7	RS485 A
8	Opto-sensor LED (Power Out)
9	Input (Input 4)
10	N/C

## **Mounting the USA28**

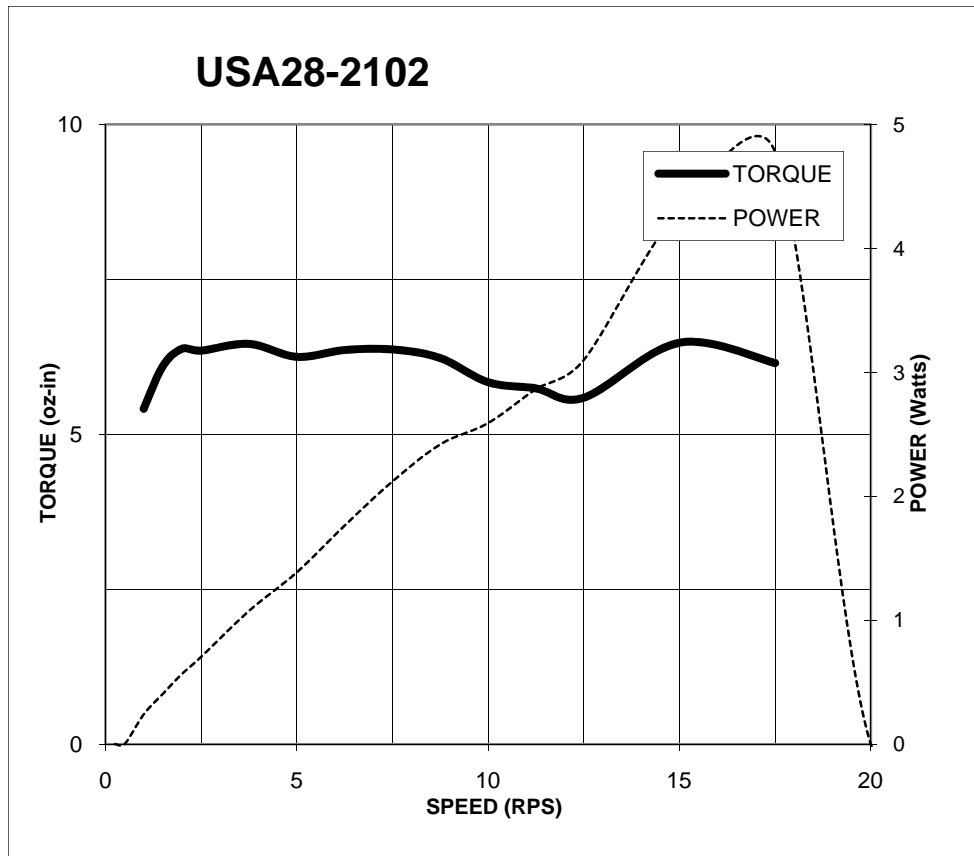
Threaded mounting holes are located on the front of the Accuriss 28. There are four threaded holes for M2.5 X .045 screws. See Dimensions section for exact locations.

## **Mounting a Load to the USA28**

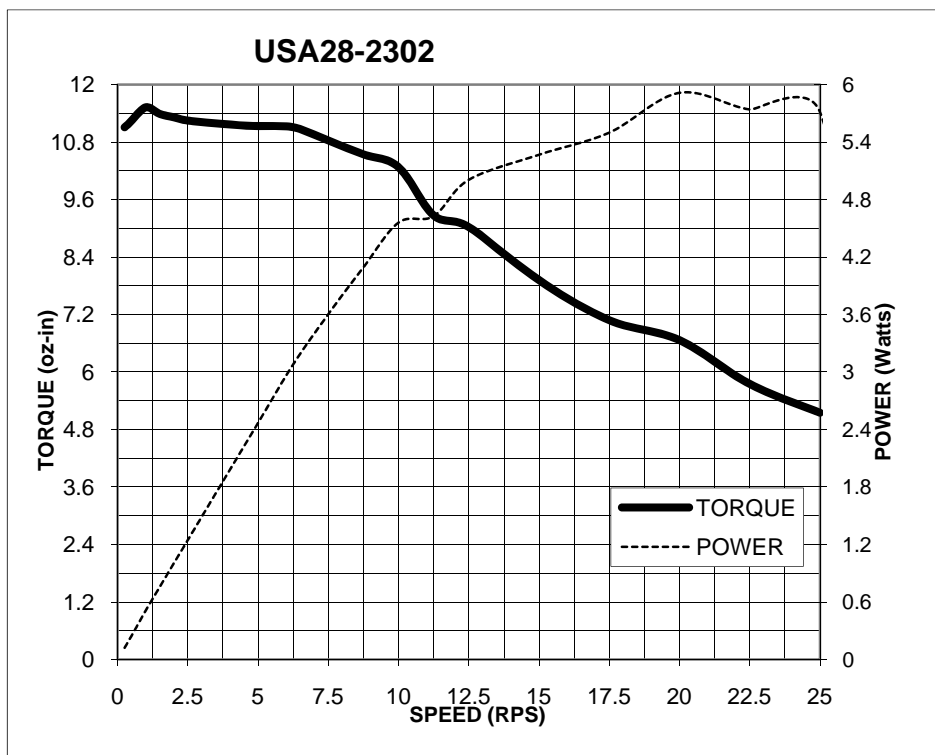
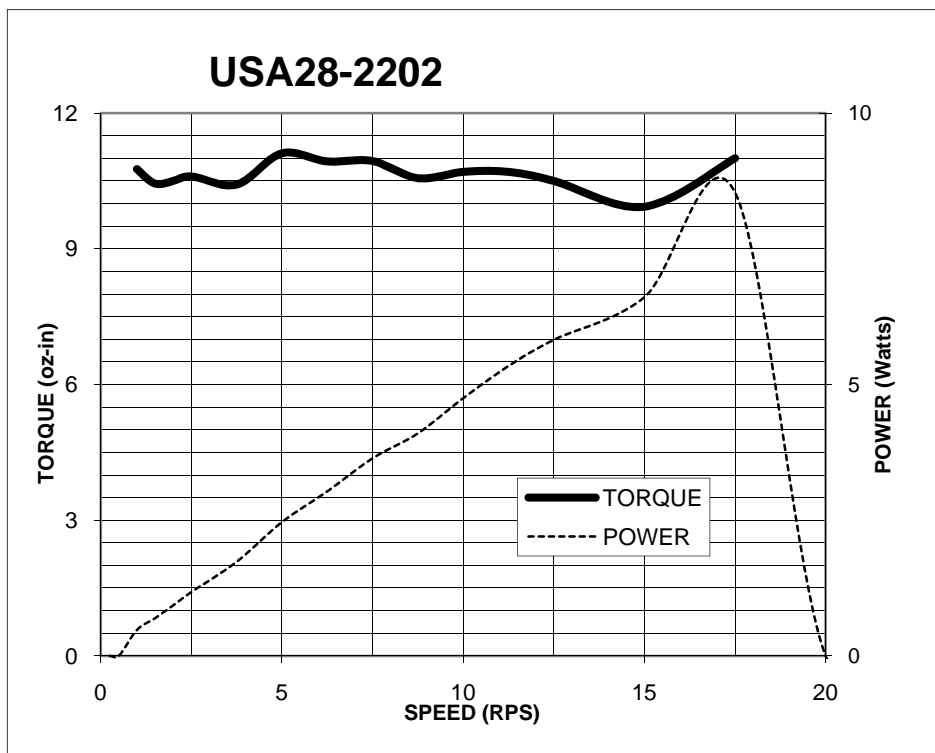
The Accuriss 28 has 5mm shaft for mounting the load (see dimensions page). Care should be used in isolating the motor bearing from thrust and axial loads. It is not advisable to couple loads to the Accuriss shaft with a solid coupling as that may impose radial loads on the motor bearings which will shorten the life of the motor. A flexible coupling, such as those made by Helical ([www.heli-cal.com](http://www.heli-cal.com)), should be used.

### Speed Torque Curves

The base motor which comes with the USA28 is a 1.8°/step 2-phase hybrid step motor. Full torque will be available from the motor when used with a drive which has a rated output of at least 0.67 amps per phase. Here are representative speed/torque curves for the three different lengths of motors available. Individual applications may differ in motor performance:







## Accuriss Communications

Communications to the Accuriss is possible with any RS485 compatible communication device and any commonly available terminal program. Most computers have either an RS232 or USB port (sometimes both). USAutomation offers software drivers for both on our web site or in our Starter Kit. Commercially available USB to RS485 and RS232 to RS485 converters will allow communications with the Accuriss from most computers using only a compatible terminal program.

Also available as a software tool is the Accuriss Terminal software, which is available as a no charge download from our web site, and it is included in our Starter Kit.

## Communication Specifications

Electrical Characteristics	In conformance with EIA-485 Use a twisted pair cable (TIA/EIA-568B CAT5e or higher is recommended) and keep the total wiring distance including extension to 50 m (164 ft.) or less.
Baud rate	Selectable from 9600 bps, 19200 bps, 38400 bps (9600 is default)
Physical layer	Asynchronous mode (8 bits, 1 stop bit, no parity)

## Starter Kit

The Accuriss Starter Kit is recommended for any first time user of the Accuriss 28. This kit includes a USB to RS485 converter, an I/O break out board, two input switches, and a CD that includes catalogs, manuals, and software.

---

## Executing Commands

In order to execute a command or a series of commands, the following syntax should be used:

```
[/][Device Address][Command][Value]...{[Command][Value]}[R][<CR>]
```

Where,

/ = the start character. Data following the “/” character will be loaded into the communication buffer.

Device Address = the device address. (Factory Preset)

Command = the command to be executed.

Value = a valid value for the command to be executed (position value, speed, etc).

Note: Up to fourteen (14) Commands and Values can be in a single command string.

R = Execute the command(s) in the communication buffer already received .

<CR> = Carriage Return+Line Feed (CR+LF) and indicates that the command string is complete.

Example 1:

```
/1A1000R<CR>
```

“/” is the start of character. Lets Accuriss know that a command string is coming.

“1” is the device address. (This is preset at the factory)

“A” instructs motor to move to an absolute position.

“1000 sets the absolute position to +1000 steps from the zero position.

“R” instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” tells Accuriss that the command string is complete and causes the command(s) to be executed since the R was included in the command string.

Example 2:

Command string 1: /1A2000A0<CR>

Command string 2: /1R<CR>

For command string 1:

“/” is the start of character. Lets Accuriss know that a command string is coming.

“1” is the device address. (This is preset at the factory)

The first “A” instructs motor to move to an absolute position.

“2000” sets the absolute position to +2000 steps relative to the zero position.

The second “A” instructs motor to move to an absolute position.

“0” sets the absolute position to the zero position.

“<CR>” tells Accuriss that the command string is complete. Nothing is executed at this time because the R command was not included in the command string.

For command string 2:

“/” is the start of character. Lets Accuriss know that a command string is coming.

“1” is the device address. (This is preset at the factory)

“R” instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” tells Accuriss that the command string is complete and causes both of absolute position the commands to be executed since an R command is included in the command string.

## Command Set

Command Set Table

Command	Type	Description	Page
A	Motion	Absolute move in positive direction	14
P	Motion	Index move in positive direction	15
D	Motion	Index move in negative direction	16
Z	Motion	Homes motor	17
z	Motion	Set current position to commanded number	19
f	I/O	Set home flag polarity	20
F	System Control	Change direction of rotation considered positive	21
V	Motion Variable	Set velocity	22
L	Motion Variable	Set acceleration rate	23
g	Program Control	Set beginning loop marker	24
G	Program Control	Set number of loops	25
H	Program Control	Halt command and wait for condition	26
S	Program Control	Skip next command and branch on condition	27
s	Program Control	Store program to address	28
e	Program Control	Execute stored program	29
R	Program Control	Run command string	30
X	Program Control	Repeat running of command string	31
m	System Control	Set maximum current while moving (%)	32
h	System Control	Set maximum hold current (%)	33
j	System Control	Set microstep resolution	34
n	System Control	Set Accuriss mode	35
b	Communication	Set baud rate	36
M	Program Control	Wait for X milliseconds	37
J	I/O	Turn Accuriss outputs on or off	38
T	System Status	Terminate current command	39
?0	System Status	Return current position	40
?2	System Status	Return current maximum speed	41
?4	System Status	Return status of inputs	42
?6	System Status	Return current microstep size	43
?9	System Control	Erase all commands in EEPROM	44
&	System Status	Return current firmware version	45
Q	System Status	Return Accuriss current status	46

**Command:** A     **Absolute move in positive direction**

**Type:** Motion Command

**Syntax:** An

**Range:** n = 0 to 2,147,483,648

**Description:** Move motor to a positive absolute position relative to the 0 (zero) position. Once at a given absolute position, executing the same command will not move the motor again since the absolute position has already been reached.

**Example:** /1A1000R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“A1000” ***Instructs motor to move to an absolute position of positive 1000 steps from the 0 (zero) position***

“R” Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** P      **Index move in positive direction**

**Type:** Motion Command

**Syntax:** Pn

**Range:** 0 to 2,147,483,648 steps

**Description:** Move motor in a **positive** direction relative to the current position. The distance to be moved is n steps. A distance n of zero will cause the motor to continuously in the positive direction relative to the current position. The speed of the motion will be the speed last set by the "V" (velocity) command. (Note: An endless move can be terminated by the T command)

**Example:** /1P1000R<CR>

"/" Is the start character. Lets Accuriss know that a command string will follow.

"1" Is the device address, (this is preset at the factory).

**"P1000"** **Instructs motor to move to a position 1000 steps in the positive direction from its current position**

"R" Instructs Accuriss to run the command(s) that are in the communication buffer.

"<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** D      **Index move in negative direction**

**Type:** Motion Command

**Syntax:** Dn

**Range:** n = 0 to 2,147,483,648 steps

**Description:** Move motor in a negative direction relative to the current position. The distance to be moved is n steps. A distance n of zero will cause the motor to continuously in the negative direction relative to the current position. The speed of the motion will be the speed last set by the "V" (velocity) command. (Note: An endless move can be terminated by the T command)

**Example:** /1D1000R<CR>

"/" Is the start character. Lets Accuriss know that a command string will follow.

"1" Is the device address, (this is preset at the factory).

**"D1000"** *Instructs motor to move to a position 1000 steps in the negative direction from its current position*

"R" Instructs Accuriss to run the command(s) that are in the communication buffer.

"<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.



**Command:** Z      **Move to home sensor**

**Type:** Motion Command

**Syntax:** Zn

**Range:** n = 0 to 2,147,483,648

**Description:** The “Z” command is used to move the motor to a known position by seeking a home sensor. When issued, the motor will turn toward the 0 position until the home opto-sensor is interrupted. If the home sensor is already interrupted, the motor will back off of the sensor in the opposite direction and then come back in until the home sensor is re-interrupted. The motor position counter is set to zero. The homing motion is done at the current speed “V”. The maximum number of steps allowed to go towards home without encountering the sensor is defined by the Z command operand (n) + 400 steps. The maximum number of steps away from home (while sensor is interrupted) is 10000 steps.

To set up the home flags:

First ensure that a positive move e.g. /1P100R moves away from home and the home flag. If motor does not go away from home, flip the connections to only one of the windings of the stepper.

The Default condition expects the output of the Home flag to be low when away from the home sensor (as is the case in an opto). If the Home flag is high when away from the home sensor (as in the case of a “Normally Open” switch) then issue the command /1f1R to reverse the polarity that is expected of the home flag.

Issue the command e.g. /1Z100000R or /1f1Z100000R as necessary.

Homing should be done at a slow speed, especially if homing to a narrow Index pulse on an encoder, which may be missed at high speeds.

Opto and flag should be set up to be unambiguous, i.e. when motor is all the way at one end of travel, flag should cut the opto, when at other end of travel flag should not cut opto. There should only be one black to white transition possible in the whole range of travel. Home can be done to an opto (N1 Mode) or Index Pulse (N2 Mode). See Appendix A

**Example:** /1Z1000R<CR>

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “Z1000” **Instructs motor to move toward a zero position a maximum of 1000 steps**
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** z      **Set current position counter**

**Type:** Motion Command

**Syntax:** zn

**Range:** n = 0 to 2,147,483,648 steps

**Description:** Set the position counter to a new value without moving.

**Example:** /1z1000R<CR>

"/" Is the start character. Lets Accuriss know that a command string will follow.

"1" Is the device address, (this is preset at the factory).

**"z1000"** ***Sets the current position to positive 1000 steps***

"R" Instructs Accuriss to run the command(s) that are in the communication buffer.

"<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** f      **Home Flag Input Level**

**Type:** I/O Command

**Syntax:** fn

**Range:** n= 0: Opto signal is TTL high when at/on the opto (Normally open)  
1: Opto signal is TTL high when away from the opto (Normally closed)

**Initial Value:** 0 (normally open)

**Description:** This command sets the home flag polarity. The default value is 0 (zero).

**Example:** /1f1R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.  
“1” Is the device address, (this is preset at the factory).  
“f1” **Sets the home flag position sensor to Normally closed**  
“R” Instructs Accuriss to run the command that it has just receive  
“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** F      **Set Positive Direction**

**Type:** System Control Command

**Syntax:** Fn

**Range:** n = 0: Motor rotation away from home is the positive direction  
          1: Motor rotation toward home is the positive direction

**Description:** Change direction of rotation considered positive. (Note: this command should only be used once after the power is on.)

**Example:** /1F1R<CR>

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “F1” **Sets the direction to rotate toward home to positive**
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** V      **Running Velocity**

**Type:** Motion Variable Command

**Syntax:** Vn

**Range:** n = 1 to 16,777,216 microsteps per second

**Description:** In position mode, this command is used to set the maximum (or slew) speed of the motor. Units are microsteps per second. The microstep step angle should be taken into account when issuing this command, the default is 1/8<sup>th</sup> step per full step of the motor.

**Example:** /1V1600R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“V1600” ***Sets maximum velocity to 1 revolution per second. (For a 200 step/rev motor and a microstep resolution of 1/8<sup>th</sup>)***

“R” Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** L      **Acceleration Rate**

**Type:** Motion Variable Command

**Syntax:** Ln

**Range:** n = 0 to 5000 microsteps/sec<sup>2</sup>

**Description:** Sets the acceleration to a value

**Example:** /1L1000R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

**“L1000” Sets acceleration to 1000 microsteps/sec<sup>2</sup>**

“R” Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** g      **Begin Loop Block**

**Type:** Program Control Command

**Syntax:** g

**Range:** N/A

**Description:** This command signifies the beginning of a loop

**Example:** /1gP1000M1000D1000M500G10R<CR>

- "/" Is the start character. Lets Accuriss know that a command string will follow.
- "1" Is the device address, (this is preset at the factory).
- "g"** ***Informs Accuriss a loop is beginning***
- "P1000" Move 1000 steps in positive direction
- "M1000" Wait 1000 milliseconds
- "D1000" Move 1000 steps in negative direction
- "M1000" Wait 1000 milliseconds
- "G10" Repeat all commands between the "G10" command and the "g" command loop 10 times
- "R" Instructs Accuriss to run the command(s) that are in the communication buffer.
- "<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.



**Command:** G End Loop Block

**Type:** Program Control Command

**Syntax:** Gn

**Range:** n = 0 to 30000

**Description:** This command signifies the end of a loop and defines the number of loops to be performed. Loops can be nested up to four levels. A value of "0" (zero) causes an infinite loop. (A "T" command will terminate an infinite loop) If no value is given, the Accuriss will use "0" (zero) as the assumed number.

**Example:** /1gP1000M1000D1000M500G10R<CR>

"/" Is the start character. Lets Accuriss know that a command string will follow.

"1" Is the device address, (this is preset at the factory).

"g" Informs Accuriss a loop is beginning

"P1000" Move 1000 steps in positive direction

"M1000" Wait 1000 milliseconds

"D1000" Move 1000 steps in negative direction

"M1000" Wait 1000 milliseconds

**"G10"** *Repeat all commands between the "G10" command and the "g" command loop 10 times*

"R" Instructs Accuriss to run the command(s) that are in the communication buffer.

"<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** H      **Halt Command String**

**Type:** Program Control Command

**Syntax:** Hn

**Range:** n = See below

**Description:** Halt the current command string and wait for a specific switch input condition to become true. If an edge is detect is desired, a look for Low and a look for High can be placed adjacent to each other, i.e. H01H11, to define a rising edge detect. A halted operation can also be resumed by entering "/1R". An "H" command with no following number will wait for switch # 2 to close (low).

01	Wait for low input 1 (switch # 1)
11	Wait for high input 1 (switch # 1)
02	Wait for low input 2 (switch # 2)
12	Wait for high input 2 (switch # 2)
03	Wait for low input 3 (switch # 3)
13	Wait for high input 3 (switch # 3)
04	Wait for low input 4 (switch # 4)
14	Wait for high input 4 (switch # 4)

**Example:** /1gH01P1000G10R<CR>

"/" Is the start character. Lets Accuriss know that a command string will follow.  
 "1" Is the device address, (this is preset at the factory).  
 "g" Informs Accuriss a loop is beginning  
**"H01"** **Wait for switch "1" to go low then execute move**  
 "P1000" Move 1000 steps in positive direction  
 "G10" Loop 10 times  
 "R" Instructs Accuriss to run the command(s) that are in the communication buffer.  
 "<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** S      **Skip Command**

**Type:** Program Control Command

**Syntax:** Sn

**Range:** n = See below

**Description:** Skip next command depending on status of a specific switch input condition. Program branching to a complex subroutine can be implemented by making the next instruction a stored string execution. Loops can be escaped by branching to a stored string with no commands.

01	Skip next instruction if input 1 (switch # 1) is low
11	Skip next instruction if input 1 (switch # 1) is high
02	Skip next instruction if input 2 (switch # 2) is low
12	Skip next instruction if input 2 (switch # 2) is high
03	Skip next instruction if input 3 (switch # 3) is low
13	Skip next instruction if input 3 (switch # 3) is high
04	Skip next instruction if input 4 (switch # 4) is low
14	Skip next instruction if input 4 (switch # 4) is high

**Example:** /1gS02P1000G10R<CR>

- "/" Is the start character. Lets Accuriss know that a command string will follow.
- "1" Is the device address, (this is preset at the factory).
- "g" Informs Accuriss a loop is beginning
- "S02" **Skip loop if switch # 2 is low**
- "P1000" Move 1000 steps in positive direction
- "G10" Loop 10 times
- "R" Instructs Accuriss to run the command(s) that are in the communication buffer.
- "<CR>" Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** **s**      **Store Command String**

**Type:**            Program Control Command

**Syntax:**        sn

**Range:**          n = 0 to 15

**Description:** Stores the command string that follows to program storage locations 0 to 15. Program 0 is executed on start up. Up to fourteen (14) commands per string maximum are allowed.

**Note:** It takes approximately 1 second for the command string to be written to the EEPROM.

**Example:**      /1s1gP1000M1000G10R<CR>

“/”            Is the start character. Lets Accuriss know that a command string will follow.

“1”            Is the device address, (this is preset at the factory).

“s1”          **Tells Accuriss to store the command string that follows to storage location # 1**

“g”            Informs Accuriss a loop is beginning

“P1000”      Move 1000 steps in positive direction

“M1000”      Wait 1000 msec

“G10”        Loop 10 times

“R”            Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>”      Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** e      **Execute Stored Command String**

**Type:** Program Control Command

**Syntax:** en

**Range:** n = 0 to 15

**Description:** Executes the command string stored in program locations 0 to 15. Program 0 is executed on start up. Up to fourteen (14) commands per string maximum are allowed.

**Example:** /1e1R<CR>

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “e1” **Instructs Accuriss to execute the command string in storage location # 1**
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

---

**Command:** R      **Run Command String**

**Type:**            Program Control Command

**Syntax:**         R

**Range:**           None

**Description:** Run the command string that is currently in the execution buffer

**Example:**        /1R<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“R”                ***Instructs Accuriss to run the command that is currently in the buffer***

“<CR>”           Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

---

**Command:** X      **Repeat Command String**

**Type:**            Program Control Command

**Syntax:**         X

**Range:**           None

**Description:** Repeat running the command string that is currently in the execution buffer

**Example:**        /1X<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“X”                ***Repeats all of previous string still in buffer***

“<CR>”           Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** m      **Run Current**

**Type:**            System Control Command

**Syntax:**         mn

**Range:**           n = 0 to 100 % of rated current

**Initial Value:** 25

**Description:** Sets maximum current applied to the motor while the unit is moving

**Example:**        /1m15R<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“m15”            ***Informs Accuriss to set run current at 15% of maximum .7 amps (0 .05 amps)***

“R”                Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>”          Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.



**Command:** h      **Hold Current**

**Type:** System Control Command

**Syntax:** hn

**Range:** n = 0 to 50 % or rated current

**Initial Value:** 10

**Description:** Sets maximum holding current while unit is not moving. This value should be set as low as possible to prevent the motor from overheating.

**Example:** /1h50R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“h50” ***Informs Accuriss to set hold current at 50% of maximum .7 amps ( 0.35 amps)***

“R” Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** j      **Microstep Resolution**

**Type:** System Control Command

**Syntax:** jn

**Range:** n = 1: Full stepping  
          2: 2 microsteps per full step  
          4: 4 microsteps per full step  
          8: 8 microsteps per full step

**Initial Value:** 8 microsteps per full step

**Description:** Sets the number of microsteps per full motor step  
(i.e. 200 step/rev \* 8 microsteps = 1600 steps per rev)

**Example:** /1j4R<CR>

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “j4” ***Informs Accuriss to set 4 microsteps per full step***
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** n      **Set Modes**

**Type:** System Control Command

**Syntax:** nn

**Range:** n = 0 to 4095. See table below

**Description:** Set Modes – The combination of Binary Bits specified enables additional operating modes for the Accuriss.

Bit 0	/1n1R: Enables the Pulse Jog Mode. Jog distance is given by “B” command, and velocity is given by “V” command. The switch inputs becomes Jog Inputs
Bit 1	/1n2R: Enables hardware limits. The level of the limits is set by the “f” command.
Bit 2	/1n4R: Enables continuous jog mode. Continuously run motor while an input switch is closed, at a velocity set by the “V” command. Note: Moves below zero position are possible. If this is undesirable, use the “z” command to define a zero position that insures that a number underflow will not occur.
Bit 3 thru Bit 11	Reserved

**Example:** /1n4R<CR>

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “n” **Informs Accuriss a “n” command is coming**
- “4” **Enables the Continuous Jog Mode (Bit 2)**
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** **b**      **Baud Rate**

**Type:**            Communication Command

**Syntax:**        **bn**

**Range:**          n = 9600: 9600 bps  
                         19200: 19200 bps  
                         38400: 38400 bps

**Initial Value:** 9600

**Description:** Sets the baud rate. This command is usually stored in program zero, which is executed on power up

**Example:**      /1**b**192000R<CR>

“/”            Is the start character. Lets Accuriss know that a command string will follow.

“1”            Is the device address, (this is preset at the factory).

“**b19200**”    **Sets the baud rate to 19200 bps**

“R”            Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>”      Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** M      **Wait for Specified Time**

**Type:**            Program Control Command

**Syntax:**        Mn

**Range:**          n = 0 to 30000 msec

**Description:** M causes the command string execution to wait for the indicated time before proceeding to the next command.

**Example:** /1M10000R<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

**“M10000” Sets wait time of 10000 msec (10 seconds)**

“R”                Instructs Accuriss to run the command(s) that are in the communication buffer.

“<CR>”        Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** J      **Output Control**

**Type:** I/O Command

**Syntax:** Jn

**Range:** n = 0: No outputs ON (Binary 00)  
          1: Output 1 ON and output 2 OFF (Binary 01)  
          2: Output 1 OFF and output 2 ON (Binary 10)  
          3: Output 1 ON and output 2 ON (Binary 11)

**Initial Value:** 0

**Description:** Turn the system outputs ON or OFF. Enter as a 2 bit binary value.

**Example:** /1J3R<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.  
“1” Is the device address, (this is preset at the factory).  
“J3” **Turns both driver outputs ON**  
“R” Instructs Accuriss to run the command(s) that are in the communication buffer.  
“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** T      **Terminate Command or Loop**

**Type:** System Control Command

**Syntax:** T

**Range:** None

**Description:** Terminate current command or loop.

**Example:** /1T<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“**T**” **Terminates the current command string**

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

---

**Command:** ?0    **Report Current Commanded Position**

**Type:**            System Status Command

**Syntax:**         ?0

**Range:**           0

**Description:** Returns the current position of the commanded motor.

**Example:**        /1?0<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“?0”              **Report current commanded position**

“<CR>”          Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.



**Command:** ?2    **Report Current Maximum Speed**

**Type:**            System Status Command

**Syntax:**         ?2

**Range:**            2

**Description:** Returns the current Slew/Max speed for Position mode of the commanded motor.

**Example:**        /1?2<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“?2”              **Report current running speed**

“<CR>”          Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** ?4 **Report Input Status**

**Type:** System Status Command

**Syntax:** ?4

**Range:** 0 to 15

**Description:** Returns the status of all four inputs in 4 bit binary format, 0 to15, of the commanded motor.

Input	Contribution to ?4 response if active
Switch 1	1
Switch 2	2
Opto 1	4
Opto 2	8

**Example:** /1?4<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“?4” **Report current input status**

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** ?6    **Report Current Microstep Size**

**Type:**            System Status Command

**Syntax:**         ?6

**Range:**           1, 2, 4, 8 microsteps per full step

**Description:** Returns the current microstep size of the commanded motor.

**Example:**        /1?6<CR>

“/”                Is the start character. Lets Accuriss know that a command string will follow.

“1”                Is the device address, (this is preset at the factory).

“?6”              **Report the current microstep size**

“<CR>”          Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** ?9 Erase EEPROM

**Type:** System Control Command

**Syntax:** ?9

**Range:** 9

**Description:** Erases all commands in the EEPROM of the commanded motor.

**Example:** /1?9<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“?9” ***Erases the system’s EEPROM***

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

---

**Command:**   **&**       **Return Firmware Version**

**Type:**        System Status Command

**Syntax:**     &

**Range:**       None

**Description:** Requests firmware version of the commanded motor.

**Example:**    /1&<CR>

“/”            Is the start character. Lets Accuriss know that a command string will follow.

“1”            Is the device address, (this is preset at the factory).

“&”            ***Requests firmware revision***

“<CR>”        Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

**Command:** Q      **Status Request**

**Type:** System Status Command

**Syntax:** Q

**Range:** 0 to 3

0: No error

1: Initialization error

2: Invalid Command

3: Operand out of range

**Description:** Query the Accuriss for its current status.

Returns the Ready/Busy status as well as any error conditions in the “Status” byte of the return string. The return string consists of the start character (/), the master address (0) and the status byte. Bit 5 of the status byte is set when the Accuriss is ready to accept commands. It is cleared when the Accuriss is busy. The least significant four bits of the status byte contain the completion code.

Errors in OpCode will be returned immediately, when errors in Operand range will be returned only when the next command is issued. See Appendix B

**Example:** /1Q<CR>

“/” Is the start character. Lets Accuriss know that a command string will follow.

“1” Is the device address, (this is preset at the factory).

“Q” **Requests Accuriss status**

“<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

## Examples

### ***Example #1 (Move to an Absolute Position of 12345 steps)***

`/1A12345R<CR>`

This breaks down to:

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “A12345” Causes the motor turn to an Absolute position 12345 steps
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

Note: HyperTerminal issues each character as you type it in. Therefore, it is not possible to cut and paste in HyperTerminal. Backspace is allowed only up to the address character. If backspace is used, all characters “backspaced” must be retyped in. If a typing error is made, typically hit enter and type it all in again – what was typed in will be overwritten as long as the R command at the end was not present.

### ***Example #2 (Loop absolute moves with a wait in between)***

`/1gA1000M500A0M500G10R<CR>`

This breaks down to:

- “/” Is the start character. Lets Accuriss know that a command string will follow.
- “1” Is the device address, (this is preset at the factory).
- “g” Indicates the start of a loop block
- “A1000” Causes the motor move to the Absolute position of 1000 steps
- “M500” Causes the Accuriss to wait for 500 Milliseconds.
- “A0” Makes the motor turn to Absolute position 0.
- “M500” Is another wait command of 500 Milliseconds.
- “G10” Will make all commands between here and starting location of the small “g” repeat 10 times
- “R” Instructs Accuriss to run the command(s) that are in the communication buffer.
- “<CR>” Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

To terminate the above loop while in progress type “/1T”.

### **Example #3 (Program Storage and Recall)**

This example stores a command string to storage location #2 for later execution:

```
/1s2gA10000M500A0M500G10R<CR>
```

This breaks down to:

“/”	Is the start character. Lets Accuriss know that a command string will follow.
“1”	Is the device address, (this is preset at the factory).
“s2”	Tells Accuriss to store the command string that follows to storage locations #2
“g”	Indicates the start of a loop block
“A1000”	Causes the motor move to the Absolute position of 1000 steps
“M500”	Causes the Accuriss to wait for 500 Milliseconds.
“A0”	Makes the motor turn to Absolute position 0.
“M500”	Is another wait command of 500 Milliseconds.
“G10”	Will make all commands between here and starting location of the small “g” repeat 10 times
“R”	Instructs Accuriss to run the command(s) that are in the communication buffer.
“<CR>”	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

(Note: program 0 is always executed on power up, if we had used storage location 0 instead of storage location 2 in the above example, then this program would execute automatically on power up).

This example executes the command string stored in the above example.

```
/1e2R<CR>
```

This breaks down to:

“/”	Is the start character. Lets Accuriss know that a command string will follow.
“1”	Is the device address, (this is preset at the factory).
“e2”	Instructs the Accuriss to execute the command string in storage location #2.
“R”	Instructs Accuriss to run the command(s) that are in the communication buffer.
“<CR>”	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.



**Example #4 (Set move and hold Current values, Wait for Switch 2 closure then Home to Opto #1)**

```
/1s0m75h10gJ3M500J0M500G10HZ10000A1000A0R<CR>
```

This breaks down to:

"/	Is the start character. Lets Accuriss know that a command string will follow.
"1"	Is the device address, (this is preset at the factory).
"s0"	Stores the command string that follows in storage location #0 (command strings stored in storage location #0 are executed on power up).
"m75"	Sets the move current to 75% of max
"h10"	Sets the hold current to 10% of max
"g"	Start a loop block
"J3"	Turns ON both driver outputs.
"M500"	Wait 500 msec
"J0"	Turn OFF both driver outputs.
"M500"	Wait 500 msec
"G10"	Will make all commands between here and starting location of the small "g" repeat 10 times
"H"	Waits for a switch 2 closure. (Switch 2 is automatically selected since no number was specified)
"Z10000"	Home the stepping motor to opto #1. Maximum steps allowed to find opto #1 = 10000 steps.
"A1000"	Move to absolute position 1000
"A0"	Move to absolute position 0
"R"	Instructs Accuriss to run the command(s) that are in the communication buffer.
"<CR>"	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

Note: This program string will abort after the "Z10000" command if it does not find a flag.

**Example #5 (Nested loop example)**

`/1gA100A1000gA100A10G10G100R<CR>`

This breaks down to:

<code>"/</code>	Is the start character. Lets Accuriss know that a command string will follow.
<code>"1"</code>	Is the device address, (this is preset at the factory).
<code>"g"</code>	Start an outer loop block
<code>"A100"</code>	Go to Absolute position 100
<code>"A1000"</code>	Go to Absolute position 1000.
<code>"g"</code>	Start an inner loop block.
<code>"A100"</code>	Go to Absolute position 100.
<code>"A10"</code>	Go to Absolute position 10.
<code>"G10"</code>	Repeat inner loop block 10 times. (End of Inner Loop)
<code>G100</code>	Repeat outer loop block 100 times. (End of outer loop)
<code>"R"</code>	Instructs Accuriss to run the command(s) that are in the communication buffer.
<code>"&lt;CR&gt;"</code>	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

To terminate either of the above loops while in progress type `"/1T"`.

### Example #6 (Skip / Branch instruction)

`/1s0gA0A1000S13e1G0R<CR>`

`/1s1gA0A100S03e0G0R<CR>`

Two “Programs” are stored in storage locations 0 and 1, respectively, and the code in these programs switches from one program to the other depending on the state of input #3. In the example below, the code will cycle the motor between position A0 and A1000 if input #3 is High and between A0 and A100 if input #3 is Low.

Stored command string 0: `/1s0gA0A1000S13e1G0R<CR>`

This breaks down to:

“/”	Is the start character. Lets Accuriss know that a command string will follow.
“1”	Is the device address, (this is preset at the factory).
“s0”	Store command string that follows to storage location #0 (executed on power up).
“g”	Start loop block
“A0”	Go to Absolute position 0
“A1000”	Go to Absolute position 1000.
“S13”	Skip next instruction if input # 3 is High
“e1”	Execute (Jump to) command string in storage location #1
“G0”	End of loop block (infinite loop since value is 0).
“R”	Instructs Accuriss to run the command(s) that are in the communication buffer.
“<CR>”	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

Stored command string 1: `/1s1gA0A100S03e0G0R<CR>`

This breaks down to:

“/”	Is the start character. Lets Accuriss know that a command string will follow.
“1”	Is the device address, (this is preset at the factory).
“g”	Start loop block
“A0”	Go to Absolute position 0
“A1000”	Go to Absolute position 1000.
“S03”	Skip next instruction if input 3 is Low
“e0”	Execute (Jump to) command string in storage location #0
“G0”	End of loop block (infinite loop since value is 0).
“R”	Instructs Accuriss to run the command(s) that are in the communication buffer.
“<CR>”	Tells Accuriss that the command string is complete and causes the command to be executed since an R is included in the command string.

## Appendix A

### Homing Algorithm in Detail

The “Z” command is used to move the motor to a known position by seeking a home sensor. When issued, the motor will turn toward the 0 position until the home opto-sensor is interrupted. If the home sensor is already interrupted, the motor will back off of the sensor in the opposite direction and then come back in until the home sensor is re-interrupted. The motor position counter is set to zero. The homing motion is done at the current speed “V”. The maximum number of steps allowed to go towards home without encountering the sensor is defined by the Z command operand (n) + 400 steps. The maximum number of steps away from home (while sensor is cut) is 10000 steps.

To set up the home flags:

First ensure that a positive move e.g. /1P100R moves away from home and the home flag. If motor does not go away from home, flip the connections to only one of the windings of the stepper.

The Default condition expects the output of the Home flag to be low when away from the home sensor (as is the case in an opto). If the Home flag is high when away from the home sensor (as in the case of a “Normally Open” switch) then issue the command /1f1R to reverse the polarity that is expected of the home flag.

Issue the command e.g. /1Z100000R or /1f1Z100000R as necessary.

Homing should be done at a slow speed, especially if homing to a narrow Index pulse on an encoder, which may be missed at high speeds.

The home opto sensor and flag should be set up to be unambiguous, i.e. when motor is all the way at one end of travel, the flag should interrupt the home opto sensor. When at other end of travel, the flag should not interrupt the home opto sensor. There should only be one High to Low transition possible in the whole range of travel. A Homing routine can be done to either an opto sensor (N1 Mode) or to an Index Pulse (N2 Mode).

The Main axis homes to opto1 (input #3). This opto is also the lower limit. The Main axis uses opto 2 (input #4) as its upper limit.

Note that limits are engaged by the command “/1n2R “ (lower case “n”). The default (n0) mode does not check the limits when moving.

Default “f” mode (f0) expects the inputs to be Low when the axis is away from the limits/home. The command “/1f1R” reverses this.

Also the opto sensors and switches are interchangeable. If four opto sensors are desired, power for the extra opto sensors can be drawn from the 5V supply on the encoder connector. These

extra opto sensors may require an external resistor in series with the LED. When connecting any external switches, connect them between any input and ground.

#### ***Main Axis Homing Details:***

There are four full steps in a single electrical cycle that moves the stepping motor. (A+, B+, A-, B-). For repeatability in homing, the home position is set to first step in that cycle (A+) that occurs after the flag edge has been seen. (This means that the home position is defined some ways beyond the middle of the flag).

However, there is a small but finite chance that an ambiguity in home position may occur in the rare case that the exact point of switching into A+ occurs at the same point at which the flag is interrupted. In the case where a 4 step ambiguity in home position may exist because the flag may sometimes be interrupted just before and sometimes just after, the procedure below describes a method by which the ambiguity can be removed. However, this procedure need not be followed if a 4 step inaccuracy in Home position is acceptable.

To eliminate the home position ambiguity, first issue the Z command, allow the motor to home. Then move 2 full steps (in any direction), now mechanically move the flag edge (or sensor) such that it trips in the middle of the sensor by adjusting it while watching the status LED on the board which shows the status of the home sensor. This will ensure that the flag trips at A- and thus the motor will home to a unique position of A+.

Another way to do this, if it is not hazardous, is to put the motor in an endless homing loop “/1gZ10000GR”, then move the flag/opto around while the motor is homing. It will be noticed that the motor will home to two distinct positions that are 4 steps apart. Make sure the High to Low transition point of the Opto sensor is NOT near these positions (exact position does not matter as long as it is not near the place where it homes to).

#### ***Manual homing:***

Motors can be manually homed to any input by the use of a polling loop such as:

```
/1s1z0R Set the current position to zero and store to storage location #1  
/1z10000gD1S04e1GR Move backwards in an endless loop until input #4 goes High .
```

#### ***Homing to a hard stop:***

It is possible to send the motor against a hard stop and then force the position counter to zero or some other value as necessary. The command string “/1z0R” zeros the position counter. The command string “/1z333R” will set the position to 333, etc. So “/1aM1z0R” zeros the counter.

## Appendix B

### Device Response Packet

The Accuriss responds to commands by sending messages addressed to the “Master Device”. The Master Device (which for example is a PC) is assumed to always have Address zero. The master device should parse the communications on the bus continuously for responses starting with /0. (Do NOT, for example, look for the next character coming back after issuing a command because glitches on the bus when the bus reverses direction can sometimes be interpreted as characters)

After the /0, the next piece of information is the “Status Character” which is actually a collection of 8 bits.

These Bits are:

Bit7 ... Reserved

Bit6 ... Always Set

Bit5 ... Ready Bit - Set when Accuriss is ready to accept a command.

Bit4 ... Reserved

#### Bits 3 thru 0 form an error code from 0-15

0 = No Error

1 = InitError

2 = Bad Command (illegal command was sent)

3 = Bad Operand (Out of range operand value)

4 = N/A

5 = Communications Error (Internal communications error)

6 = N/A

7 = Not Initialized (Controller was not initialized before attempting a move)

8 = N/A

9 = Overload Error (Physical system could not keep up with commanded position)

10 = N/A

11 = Move Not Allowed

12 = N/A

13 = N/A

14 = N/A

15 = Command Overflow (unit was already executing a command when another command was received)

Note that RS485 Bus devices must respond right away, after the master sends a command, before the success or failure of the execution of the command is known. Due to this reason some error messages that come back are for the previous command, for example, “failure to find home”.

A program that receives these responses must continuously parse for /0 and take the response from the bytes that follow the /0. The first Character that comes back may be corrupted due to line turn around transients, and should not be used as a “timing mark”.

**Example Initialization Error Response:**

Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)  
 An initialization error has response has 1 in the lower Nibble. So the response is 41 Hex or 61 Hex, which corresponds to ASCII character upper case “A” or lower case “a”, depending on if the device is busy or not.

**Example Invalid Command Response:**

Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)  
 An invalid command has response has 2 in the lower Nibble. So the response is 42 Hex or 62 Hex, which corresponds to ASCII character upper case “B” or lower case “b”, depending on if the device is busy or not.

**Example Operand Out of range Response:**

Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)  
 An operand out of range has response has 3 in the lower Nibble. So the response is 43 Hex or 63 Hex, which corresponds to ASCII character upper case “C” or lower case “c”, depending on if the device is busy or not.

**Example Overload Error Response:**

Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)  
 An overload error has response has 7 in the lower Nibble. So the response is 47 Hex or 67 Hex, which corresponds to ASCII character upper case “I” or lower case “i”, depending on if the device is busy or not.

**Example Response to command /1?4**

- FFh: RS485 line turn around character. It's transmitted at the beginning of a message.
- 2Fh: ASCII “/” Start character. The DT protocol uses the ‘/’ for this.
- 30h: ASCII “0” This is the address of the recipient for the message.  
 In this case ASCII zero (30h) represents the master controller.
- 60h: This is the status character (as explained above
- 31h:
- 31h: These two bytes are the actual answer in ASCII.  
 This is an eleven which represents the status of the four inputs.  
 The inputs form a four bit value. The weighting of the bits is:  
 Bit 0 = Switch 1  
 Bit 1 = Switch 2  
 Bit 2 = Opto 1  
 Bit 3 = Opto 2
- 03h: This is the ETX or end of text character. It is at the end of the answer string.
- 0Dh: This is the carriage return...
- 0Ah: ...and line feed.

### Command Response Modification

Requires Firmware Revision 7.08+

Note that it is possible to change the response from the device to include device address and other parameters such as current position.

/1Q1<CR> will modify every subsequent response to include the device address

/1Q2<CR> will modify every subsequent response to include the current position

If more than one command is used then both parameters will be reported back with a comma delimited response.



## Appendix C

### Heat Dissipation

Most stepper applications require intermittent moving of the motor. In the Accuriss, the current is increased to the “move” current, the move is performed, and the current is then reduced to the “hold” current automatically. The heat dissipation in the drive is proportional to the current flowing in the drive and, therefore, the dissipation occurs primarily during the “move”.

When the Drive generates heat, the heat first warms the circuit board and housing.

Only then does the heat transfer to the surroundings. For intermittent moves that are less than one minute in duration, the Drive primarily cools using this thermal inertia of the board and housing, and not by steady state dissipation to the surrounding ambient.

The electronics for the Accuriss are fully capable of running at the rated voltage and current. However, due to the small size of the boards, which limits the steady state heat transfer to the ambient, care must be taken when the drive is used in high duty cycle and/or high current applications. For conservative operation, it is recommended that the duty cycle be reduced linearly, from 100% duty at 50% of rated current, to 25% duty at 100% of rated current. (Duty cycle means the percentage of the time that the drive is moving the load – averaged over 5 minutes). Conservatively, the maximum continuous run at 100% current is about 1 minute. An on board thermal cutout typically trips after about 2 minutes at 100% current. (This cutout is self-resetting when the drive cools). Of course, at 50% of current, the drive will run continuously with no time limit.

Most “intermittent move” applications will NOT require derating of the drive.

In case the Accuriss is required to run 100% of the time at 100% current, forced air cooling or mounting to an aluminum structure is required.

Accuriss drives are designed with parts rated at 85°C or better. This means the PCB copper temperature must remain below 85°C. The ambient air temperature allowed depends on the airflow conditions.

**MTBF is 20000Hrs at 85°C PCB copper temperature and doubles for every 10°C lower than 85°C.**