specMech Communications Guide

Summary

The communications rules for controlling the BOSS spectrograph mechanical system are described. Using printable ASCII characters the SDSS-V actor can command the BOSS spectrographs and read its sensors. A simple command syntax with an NMEA 0183 reply format is amenable to both autonomous computer control and terminal interaction with a human.

Date	Note	Author
2020-09-09	Started	AU OCIS
2020-09-14	Added info on sending commands to the spectrograph, assumed NMEA	AU OCIS
2020-10-10	Modified time format	AU OCIS
2020-10-13	Added checksum	AU OCIS
2020-10-27	Changed the title from BOSS to specMech Communications Guide	AU OCIS
2020-11-11	Minor edits	AU OCIS
2020-11-24	Added response details to the report command	AU OCIS
2020-11-26	Adapted to possible actor implementation	AU OCIS
2021-01-13	Removed the timezone from the date-time examples	AU OCIS
2021-01-25	Updated examples to include command id/note and timestamps	AU OCIS
2021-01-30	Edited report orientation description	AU OCIS
2021-02-03	rp (report pneumatics) command is working	AU OCIS

Background

The BOSS spectrograph mechanical devices are controlled by an on-board microcontroller. Historically, the microcontroller software was called specMech and we continue to use that name. SpecMech devices include the shutter, Hartmann doors, collimator mirror motors, and sensors for temperature, humidity, CCD dewar vacuum pressure, compressed air, and attitude. It also communicates with the independent liquid nitrogen autofill system.

The CCD detector is read out by an Archon controller that is separate from specMech and is not discussed here.

Commands to and replies from specMech are simple serial data transactions using only the printable ASCII characters (decimal 32 to 126). The hardware connection is tcp/ip over EtherNET using the Telnet protocol.

While normal operations will be with a companion high-level observer's program (the actor), direct interaction through a terminal connection can be useful for debugging and maintenance.

Command Rules

SpecMech receives commands from the observer's software (actor) or terminal connection.

- SpecMech never initiates a conversation. It replies only after detecting a carriage return ('\r') from the sender.
- Only printable ASCII characters are sent by specMech to the actor.[1]
- SpecMech responds to commands by echoing the command on a single line, followed by responses, if necessary. Responses are in NMEA 0183 format.
- After a system reboot, only an exclamation mark character (!) is sent in response to commands. You must acknowledge the reboot by sending an exclamation mark.

Command Syntax

Commands to specMech are built of printable ASCII characters (decimal 32 to 126) in a verb/object/value format.

[1] Note: The Lantronix XPort that we use for Ethernet provides the telnet software layer and by default, carriage returns (\r) sent by specMech are always followed by a NULL (\0). When specMech sends \r\n (as indicated throughout this document), you will receive \r\0\n. End-of-line indicators for telnet terminals are of two varieties: \r\0 or \r\n and telnet will add either the \0 or \n to every carriage return it sends. If you're receiving stream data from specMech, check that the byte following a \r is either a \0 or \n and discard it.

<command> :== <verb>[<object>[<value>]][;note]<terminator>

Verbs are single characters. For example, the letter ‘c’ might be the command verb to close something. These are case sensitive.

Verbs act on objects. Objects are also single characters. For example, ‘s’ is the shutter. Some verbs do not require objects. Objects are case sensitive.

Values may be any character string that modifies an object. These can be used to set a motor position or update the date and time.

A semicolon at the end of the command (;) precedes an optional note. This note tags the command and is repeated in output sentences from that command. Up to 8 characters are available for the note.

The terminator indicates the end of the command string. We use the carriage return (‘\r’) since this is the character sent from a keyboard when the Enter key is pressed. Command processing proceeds after the terminator is seen.

NB: The input buffer is large enough for single commands but we don’t do much to check for overflows. We recommend checking for receipt of the > prompt before sending more commands to avoid overflows.

Command Echo

The first response line from every command is an immediate echo of the command with a timestamp.

Response Syntax

Data Format (NMEA 0183)

We use the NMEA 0183 message structure[2] for BOSS spectrograph data sent *from* specMech *to* the actor. It’s easy to create and parse, readable at a terminal, and is already well-documented. Here’s an example:

```
$S2XYZ,DATA1,DATA2,...*CS
>
```

It always starts with a dollar sign ($) followed by two characters identifying the sender, in this case “S2,” the BOSS spectrograph #2. The next three characters, XYZ, is the sentence identifier, which identifies the data format. Comma separated data fields follow, ending in an

[2] A clear description is given in section 3 of this document.

asterisk (*), and finished with a two-digit hexadecimal checksum,[3] indicated by "CS" above.

The following is a GPS position recorded from a terminal session. With a well-written description, it's easy to see what you have.

```
=~=~=~=~=~=~=~=~=~=~=~=~= PuTTY log 2020.09.07 14:37:06 =~=~=~=~=~=~=~=~=~=~=~=~=
$GPRMC,213650.00,A,3408.56593,N,11808.17222,W,0.034,,070920,,,D*68
$GPVTG,,T,,M,0.034,N,0.063,K,D*24
$GPGGA,213650.00,3408.56593,N,11808.17222,W,2,07,1.29,251.3,M,-32.4,M,,0000*69
$GPGSA,A,3,10,18,21,11,08,32,24,,,,,,2.06,1.29,1.61*0E
$GPGSV,3,1,10,08,31,300,36,10,62,008,42,11,11,317,33,18,29,122,35*7D
$GPGSV,3,2,10,21,55,320,20,23,43,052,37,24,19,062,34,32,63,206,31*73
$GPGSV,3,3,10,46,49,199,31,51,49,161,41*7E
$GPGLL,3408.56593,N,11808.17222,W,213650.00,A,D*7A
```

For example, the highlighted $GPGGA line is described here:

GGA Global Positioning System Fix Data. Time, Position and fix related data for a GPS receiver

```
                                                  11
       1         2       3 4        5 6 7  8   9  10 |  12 13  14   15
       |         |       | |        | | |  |   |   | |   | |   |    |
$--GGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh

 1) Time (UTC)
 2) Latitude
 3) N or S (North or South)
 4) Longitude
 5) E or W (East or West)
 6) GPS Quality Indicator,
    0 - fix not available,
    1 - GPS fix,
    2 - Differential GPS fix
 7) Number of satellites in view, 00 - 12
 8) Horizontal Dilution of precision
 9) Antenna Altitude above/below mean-sea-level (geoid)
10) Units of antenna altitude, meters
11) Geoidal separation, the difference between the WGS-84 earth
    ellipsoid and mean-sea-level (geoid), "-" means mean-sea-level below ellipsoid
12) Units of geoidal separation, meters
13) Age of differential GPS data, time in seconds since last SC104
    type 1 or 9 update, null field when DGPS is not used
14) Differential reference station ID, 0000-1023
15) Checksum
```

Deviations from NEMA 0183

[3] The NMEA checksum is the XOR of all the bytes between the $ and the *, not including the $ or *. Not exactly foolproof but since we're working with short strings, it's not too bad.

We extend the definition by allowing sentences longer than 80 characters.

GPS data are sent in a continuous stream but specMech goes quiet after the requested information is delivered. A single line containing a > character indicates that specMech is waiting for a new command.

Command List

Commands are listed below.

c - Close

Close the shutter or Hartmann masks. Possible objects are:

l - left Hartmann mask
r - right Hartmann mask
b - both Hartmann masks
s - shutter

Example:

```
>cs\r (close the shutter)
$S2CMD,2022-05-09T12:23:17,cs*65\r\n (Command echo)
>
```

m - Move

Move a motor. Possible objects are:

a - Motor A, relative motion in μm.
b - Motor B, relative motion in μm.
c - Motor C, relative motion in μm.
d - All motors, relative motion in μm (collimator piston motion)
A - Motor A, go to absolute position in μm.
B - Motor B, go to absolute position in μm.
C - Motor C, go to absolute position in μm.

All values are positive or negative integers. Software restricts the position to within about ±1 mm of the center value of 1500 μm unless unsafe mode is set (see set command).

Motor commands are sent to the controller for execution. We do not wait for the motor motion to complete, so it is up to you to check that the new motor position is where you want it to be before starting an exposure.

Examples:

```
>mB1500\r (Move motor B to absolute position 1500 μm)
```

```
$S2CMD,2022-05-09T13:02:32,mB1500*5B\r\n
>
```

```
>mb100\r (Move motor B 100 µm forward)
$S2CMD,2022-05-09T13:06:10,mb100*4A\r\n (Command echo)
>
```

Note on motor position: We use a PD (proportional-derivative) loop to control the motor movement, leaving out the integral term. This means that we accrue an offset from the target position. The error can be a few µm. You can do a relative move that overshoots the error a bit if it bothers you.

o - Open

Open the shutter or Hartmann masks. Possible objects are:

l - left Hartmann mask
r - right Hartmann mask
b - both Hartmann masks
s - shutter

Example:

```
>ob\r (Open both Hartmann masks)
$S2CMD,2022-05-09T12:23:20,ob*7C\r\n (Command echo)
>
```

s - Set

Set parameters. Possible objects are:

t - time. Set the date/time clock on the specMech board. The value is the ISO time format to the nearest second, without the time zone (YYYY-MM-DDTHH:MM:SS). For example, 2022-05-08T10:26:33.

s - safe. Set the collimator motor range inside the limit switch positions.

u - unsafe. Set the collimator motor range to allow hitting the limit switches. This is used when setting the collimator motor home position.

Examples:

```
>st2022-05-08T08:37:00\r
$S2CMD,2000-01-01T00:02:53,st2022-05-08T08:37:00*29\r\n>
```

Set the time to 2022-05-08T08:37:00. In this instance, the command line echo timestamp

date is 2000-01-01, which is the power-up date if a backup battery is not installed.

```
>ss\r
$S2CMD,2022-05-08T08:44:54,ss*79\r\n>
```

Sets safe mode, where the collimator motors are software-restricted to a zone that should not hit a limit switch.

```
>su\r
$S2CMD,2022-05-08T08:44:48,su*72\r\n>
```

Sets unsafe mode where the collimator motors are allowed to hit the limit switches. This mode is used after the collimator mirror has been removed and requires a new home position setting.

r - Report

The report command returns data in NMEA 0183 sentence format followed by a greater than ('>') symbol on a separate line. This command has many objects that we describe in detail here.

a, b, c, or d - Motor Information

Commanding ra, rb, or rc will return the status of the individual motor while sending rd will return information for all three motors. Example:

```
>rd\r (1)
$S2CMD,2022-05-08T08:37:15,rd*6E\r\n (2)
$S2MTR,2022-05-08T08:37:15,a,2001,um,0,um/s,0,mA,?,dir,?,lim,*50\r\n (3)
$S2MTR,2022-05-08T08:37:15,b,2001,um,0,um/s,0,mA,?,dir,?,lim,*53\r\n
$S2MTR,2022-05-08T08:37:15,c,2002,um,0,um/s,0,mA,?,dir,?,lim,*51\r\n
>
```

Line 1 is the command you sent to specMech, asking for a report on all three collimator motors. Line 2 is the command line echo. Line3 and the following lines show the motor status.

```
 1 2   3                   4 5    6  7 8    9 10 11 12 13 14  15   16
 | |   |                   | |    |  | |    | |  |  |  |  |   |     |
$S2MTR,2022-05-08T08:37:15,a,2001,um,0,um/s,0,mA,?,dir,?,lim,*50\r\n>
```

Where:

1. S2 indicates spectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. MTR indicates this a motor sentence
3. Timestamp for the command
4. Indicates this is information for motor A

5. Motor A is at position 2001 μm
6. um indicates that the previous field was in μm units
7. 0 is the motor speed in μm/s
8. um/s are the motor speed units
9. 0 is the motor current
10. mA units for the motor current
11. ? indicates the direction that the motor was headed in its most recent commanded motion (the options are F, R, and ? for forward, reverse, and unknown). This is unknown if the specMech controller was just booted and the motor hasn't been commanded to move a non-zero amount.
12. dir describes the previous field (motor direction)
13. ? tells you if a limit switch was triggered (the options are Y and ?)
14. lim tells you that the previous field was the limit switch indicator
15. 50 is the checksum
16. > indicates end-of-data and terminal prompt

Note on motor current: The motor current is measured to the nearest 10 mA. The controller's sampling scheme doesn't always read the current accurately and our motors sometimes draw only a small amount of current so a reading of 0 mA does not mean the motor has stopped. Check the motor position twice in succession to tell you if it has stopped moving.

Note on limit switch state: Our limit switches do not toggle sharply, that is, they don't have a positive snap-action when they close or open but rather have a small region where the contact resistance rises or lowers, sometimes bumpily. The controller is designed to move only in the "safe" direction after a limit strike, but because of the soft contact and switch bounce it can get confused about which direction is safe. The limit state is cleared immediately by the controller upon moving in the safe direction but the switches have hysteresis of over 100 μm. This means that a small motion in the safe direction clears the limit from the controller's viewpoint but does not clear the physical switch state and you can move in the wrong direction without limit switch protection. The unstick command should be used to get out of a limit switch hit.

A, B, or C - Motor Controller Parameters

The motor controller has stored parameters tuned for our particular motors and requirements. You can read these parameters for each motor by using the upper case version of the motor name. This command returns multiple sentences:

```
[GET NEW VERSIONS]
>rC\r
$S2CMD,2022-05-08T08:44:19,rC*41\r\n
$S2ETI,2022-05-08T08:44:19,MtrC,23.8,V,26.2,C,2022-05-08T08:44:16,encSaveTime,*
74\r\n
```

```
$S2MTC,2022-05-08T08:44:19,MtrC,2000,mA,0x02,S4,*47\r\n
$S2PID,2022-05-08T08:44:19,MtrC,15.50,P,0.000,I,66.20,D,0,maxInt,*24\r\n
$S2DMM,2022-05-08T08:44:19,MtrC,15,dead,85000,minP,800000,maxP,150000,qpps,*77\
r\n
>
```

The MET sentence:

```
 1 2   3                  4 5    6 7    8 9                  10          11    12
 | |   |                  | |    | |    | |                  |           |     |
$S2MET,2022-05-08T08:44:19,C,23.8,V,26.2,C,2022-05-08T08:44:16,encSaveTime,*74\r\n>
```

Where:

1. S2 indicates spectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. MET indicates controller voltage, temperature, and encoder save time format
3. Timestamp
4. C indicates this is information for motor C
5. 23.8 is the controller power supply voltage
6. V indicates that the previous field was volts
7. 26.2 is the controller temperature
8. C is the temperature unit (Celsius)
9. Last time the encoder position was saved to non-volatile memory
10. encSaveTime describes the previous field
11. Checksum
12. > end of data and terminal prompt.

The MIL sentence:

```
 1 2   3                   4 5    6  7    8   9    10
 | |   |                   | |    |  |    |   |     |
$S2MIL,2022-05-08T08:44:19,C,2000,mA,0x02,S4,*47\r\n>
```

Where:

1. S2 indicates spectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. MIL indicates a motor, current, and sense switch status format
3. Timestamp
4. C indicates this is information for motor C
5. 2000 is the controller maximum allowed current
6. mA are the maximum current units
7. 0x02 is the control input mode
8. S4 indicates the previous field is for the S4 input terminal
9. Checksum
10. > end of data and terminal prompt.

The PID sentence:

```
 1 2   3                 4 5    6  7   8 9  10 11 12    13   14
 | |   |                 | |    |  |   | |    | | |     |    |
$S2PID,2022-05-08T08:44:19,C,15.50,P,0.000,I,66.20,D,0,maxInt,*24\r\n>
```

Where

1. S2 indicates spectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. MET indicates controller voltage, temperature, and encoder save time format
3. Timestamp
4. C indicates this is information for motor C
5. 15.50 the proportional gain
6. P indicates that the previous field was for the proportional gain
7. 0.000 is the integral gain
8. I indicates the previous field was the integral gain
9. 66.20 is the derivative gain
10. D indicates the previous field was for the differential gain
11. 0 is the maximum integral windup (maxI)
12. maxInt indicates the previous field was the maximum windup
13. Checksum
14. > end of data and terminal prompt

The DMM sentence:

```
 1 2   3                 4 5  6    7     8    9     10   11     12    13    14
 | |   |                 | |  |    |     |    |     |    |      |     |     |
$S2DMM,2022-05-08T08:44:19,C,15,dead,85000,minP,800000,maxP,150000,qpps,*77\r\n>
```

Where:

1. S2 indicates sectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. DMM indicates dead range, min and max position format
3. Timestamp
4. C indicates this is information for motor C
5. 15 is the position dead band in encoder units
6. dead indicates that the previous field was the dead band
7. 85000 is the minimum encoder position (encoder units)
8. minP indicates the previous field was the minimum encoder position
9. 800000 is the maximum encoder position (encoder units)
10. maxP indicates the previous field was the maximum encoder position
11. 150000 is the maximum quadrature pulses per second (qpps)
12. qpps indicates that the previous field was for qpps
13. Checksum
14. > end of data and terminal prompt

e - Environment

The temperature and humidity of three places inside the spectrograph and the temperature of the inside of the specMech box is reported.

```
>re\r (1)
$S2CMD,2022-05-20T08:15:25,re*66\r\n (2)
$S2ENV,2022-05-20T08:15:26,-666.0,C,-666,%,18.7,C,68,%,-666.0,C,-666,%,18.8,C,*41\r\n> (3)
```

Lines 1 and 2 are the command you sent and the command echo. Line 3 is interpreted as:

```
 1 2   3                   4      5 6    7 8    9 10 11 12   13  14 15 16  17 18    19
 | |   |                   |      | |    | |    | |  |  |    |   |  |  |   |  |     |
$S2ENV,2022-05-20T08:15:26,-666.0,C,-666,%,18.7,C,68,%,-666.0,C,-666,%,18.8,C,*41\r\n>
```

Where:

1. S2 indicates this is spectrograph#2
2. ENV says this is an environment report sentence
3. Timestamp
4. Temperature near the blue camera (no sensor was installed at the time)
5. Indicates the units (Celsius) for the previous field
6. Humidity near the blue camera (no sensor was installed at the time)
7. Units (%) for the previous field
8. Temperature near the red camera (18.7 C)
9. Indicates the units (Celsius) for the previous field
10. Humidity near the red camera (68%)
11. Units (%) for the previous field
12. Temperature near the collimator (no sensor was installed at the time)
13. Indicates the units (Celsius) for the previous field
14. Humidity near the collimator (no sensor was installed at the time)
15. Units (%) for the previous field
16. specMech box temperature
17. Units (Celsius) for the previous field
18. Checksum
19. End of data and terminal prompt

o - Orientation

These are the x, y, and z accelerations with respect to the gravity vector in cm/s^2. The x axis is vertical, zenith pointing. The other two axes are fixed in the spectrograph such that ? is along the collimator optical axis in the direction from the fiber slit towards the collimator. Right hand rule. [need to get the signs right after telescope deployment]

The device measures acceleration so reading the orientation while the telescope is ramping

up or down from a slew motion will give erroneous results.

```
>ro\r (1)
$S2CMD,2022-05-20T08:17:11,ro*69\r\n (2)
$S2ORI,2022-05-20T08:17:11,-962.9,1.2,-5.7,*6D\r\n> (3)
```

Lines 1 and 2 are the command you sent and the command echo, respectively. Line 3 is interpreted as follows:

```
 1 2   3                     4       5   6     7     8
 | |   |                     |       |   |     |     |
$S2ORI,2022-05-20T08:17:11,-962.9,1.2,-5.7,*6D\r\n>
```

Where:

1. S2 indicates this is spectrograph #2
2. ORI says this is an orientation sentence
3. Timestamp for the command
4. Zenith direction gravity component
5. Collimator axis component (check this)
6. Blue camera axis component (check this)
7. Checksum
8. End of data and terminal prompt

p - Pneumatics

This command prints the state of the sensors on the pneumatic cylinders.

```
>rp\r (1)
$S2CMD,2022-05-20T08:15:41,rp*71\r\n (2)
$S2PNU,2022-05-20T08:15:41,o,shutter,c,left,c,right,1,air,*54\r\n> (3)
```

Line 1 is the command you sent. Line 2 is the command echo. Line 3 is interpreted as:

```
 1 2   3                   4 5       6 7    8 9    10 11   12   13
 | |   |                   | |       | |    | |     | |    |     |
$S2PNU,2022-05-20T08:15:41,o,shutter,c,left,c,right,1,air,*54\r\n>
```

where:

1. S2 indicates this is spectrograph#2
2. PNU says this is a pneumatic report sentence
3. Timestamp for the command
4. State of the pneumatic cylinder, where o means open, c means closed, t means cylinder is in transit (between open and closed), and x means an error (both sensors

active).
5. shutter means the previous report is for the shutter.
6. State of the left Hartmann mask
7. Left refers to the previous entry
8. State of the right Hartmann mask
9. Right refers to the previous entry
10. State of the air pressure sensor, where 0 means no pressure is sensed and 1 means sufficient pressure is available.
11. air indicates that the previous item was the air pressure sensor
12. Checksum
13. End of report indicator

t - Time

Prints the current time on the specMech date/time clock. Example:

```
>rt\r (1)
$S2CMD,2022-05-20T08:16:04,rt*77\r\n (2)
$S2TIM,2022-05-20T08:16:04,2022-05-08T08:37:00,set,2022-05-20T08:14:15,boot,*10\r\n (3)
>
```

Line 1 is the command you sent. Line 2 is the command echo. Line 3 shows the current time as 2022-05-20T08:16:04 followed by the last time that the clock was set, 2022–05-08T08:37:00. The last time listed, 2022-05-20T08:14:15 shows the time specMech system was booted.

The format:

```
 1 2   3                   4                   5   6                   7     8     9
 | |   |                   |                   |   |                   |     |     |
$S2TIM,2022-05-20T08:16:04,2022-05-08T08:37:00,set,2022-05-20T08:14:15,boot,*10\r\n>
```

where:

1. S2 indicates spectrograph #2 (S2 for Las Campanas; this would be S1 at APO)
2. TIM indicates this a time sentence
3. Timestamp for the command
4. Time that the time was last set using the set command
5. set indicates that the previous item was the set time
6. Time that the specMech controller was booted
7. boot indicates that the previous field was the boot time
8. 10 is the checksum
9. > indicates end-of-data and terminal prompt

v - Vacuum

This returns the ion pump vacuum pressure in the red and blue dewars. The values are

log10(P) where the pressure P is in Pascals. 1 mbar = 100 Pa.

```
>rv\r (1)
$S2CMD,2022-05-20T08:15:57,rv*70\r\n (2)
$S2VAC,2022-05-20T08:15:57,-6.86,redvac,-6.86,bluevac,*07\r\n> (3)
```

As before, lines 1 and 2 are the sent command and command echo. Line 3 is interpreted as:

```
 1 2   3                    4      5       6      7         8
 | |   |                    |      |       |      |         |
$S2VAC,2022-05-20T08:15:57,-6.86,redvac,-6.86,bluevac,*07\r\n>
```

where:

1. S2 denotes spectrograph #2
2. VAC says this is a vacuum measurement sentence
3. Timestamp for the command
4. Red dewar vacuum, log10(P) where P is in Pascals. -6.86 is the value reported when the ion pumps are turned off.
5. redvac means the previous number was for the red dewar vacuum
6. Blue dewar vacuum.
7. bluevac means the previous number was for the blue dewar vacuum.
8. Checksum

V - Version

Reports the specMech version.

```
>rV (1)
$S2CMD,2022-05-20T08:16:03,rV*52\r\n (2)
$S2VER,2022-05-20T08:16:03,2022-05-18,*5F\r\n> (3)
```

The version is just the date that the firmware was compiled, in this case, 2022-05-18. Lines 1 and 2 are the sent command and the command echo, and the line 3 interpretation follows from previous examples.

Reboot

The R command attempts to reboot the specMech controller. If a motor is moving, the controller will not reboot to avoid saving an incorrect motor position. If the motors are not moving, the motor positions are saved before the reboot sequence starts.

```
>R\r
```

This command does not return information unless a motor is moving, in which case it will

send some error messages.

Z - Zero

Set the selected motor encoder position to 0. This is used for setting the collimator mirror home position (see the BOSS technical manual for the procedure) and shouldn't be used during normal operations. Possible objects are:

a - Motor A
b - Motor B
c - Motor C

Example:

```
>Zb\r\n
<command echo>
```

This sets motor b's position to 0.

Verb	Object	Value
m (move motor)	a, b, c, or d for relative moves in μm (d moves all at once). A, B, or C for moves to an absolute position in μm.	Integer
s (set)	t (time) u (unsafe; expands permissible motion to allow hitting the limit switches) s (safe; sets permissible position to inside limit switch boundaries)	YYYY-MM-DDThh:mm:ss[4]
r=report	a, b, or c (individual collimator motor) e (environment: T and RH) o (orientation wrt gravity) p (pneumatic sensors & air pressure) s (status; this list) t (current day/time) v ion pump vacuum V (version, a date like YYYY-MM-DD)	None
R=reboot	Reboots the controller	None

[4] The first two digits of the year are fixed to be "20." That is, you can't specify a year outside the range of 2000 to 2099. Someone besides me will need to change the code in 2100.

Some specific examples. Your commands are in **red**, the specMech response is in **blue**.

Action	Example Commands & Responses
Set day/time clock	>st2022-05-08T08:37:00\r $S2CMD,2000-01-01T00:02:53,st2022-05-08T08:37:00*29
Close both Hartmann doors	>cb\r \r\n>
Open shutter	os\r \r\n>
Report pneumatics	>rp\r $S2PNU,2020-10-27T18:28:15,c,shutter,c,left,t,right,1,air,0b10*13\r\n>
Reboot	>R\r >
Reboot acknowledge	!!\r \r\n>
ms (an unrecognized command)	ms\r $S2ERR*24\r\n>

Reboot Command

The R command attempts to reboot the specMech controller. If a motor is moving, the controller will not reboot to avoid saving an incorrect motor position. If the motors are not moving, the motor positions are saved before the reboot sequence starts.

To be fixed: The Lantronix XPort controller is not rebooted with the reboot command. A future hardware modification should allow this to happen.

Reboot Acknowledge

Immediately after a reboot, specMech will respond to any command except '!\r' with another exclamation point (!) and not do anything else. You *must* respond by sending an exclamation point command (!\r) to acknowledge that you understand the system has been rebooted.

The system reboots when it powers up from an unpowered state, when you send the reboot

command (R\r), or when you push the reset button on the specMech board. If none of these occurred and the system rebooted, you have a problem.

The exclamation point character will never appear anywhere else in specMech output.

Error Codes

Error codes are reported in NMEA sentences flagged with ERR. For example:

```
S2ERR,101,Can't get current time*21
```

Details for each error code follow.

Error Code	Description
101	The date-time clock, a DS3231, on the specMech circuit board is not responding to commands. It is also possible that the TWI bus is corrupted by some other device failure.

101

The date-time clock on the specMech circuit board is not responding.