Spaceball® 2003/3003 Protocol

Revision: 1.5
Created: September 10, 1997
Revised: April, 25th 2000
Author: John Hilton

1 Introduction

The Spaceball α Protocol was first implemented in the Spaceball Model 1003 released in August of 1988. This was a very comprehensive protocol which simplified interfacing to various machines of the time. The Spaceball Model 2003, released in June 1991, was plug compatible with the Model 1003 by only implementing a commonly used subset, called the α2 protocol. Similarly the Spaceball Model 3003 supported the α2 protocol. Now, as the technology has evolved, the α3 protocol is defined here which has some minor variations to the α2 protocol related to the sensitization packets.

For a quick interfacing summary refer to the section entitled *Interfacing Summary*.

2 Communications' Settings

The Spaceball communicates using an RS232-C interface at 9600 baud, 8 data bits, no parity and one stop bit. XON/XOFF flow control is employed. Only three lines are used; transmit, receive and signal ground. Other signals may be used for power.

3 Packet Structure

Commands and data passing to and from the Spaceball are contained in packets consisting of a header byte, zero or more data bytes and a terminating CR (0D hex) or CR+LF (0D,0A hex). The header byte is a printable character. Data bytes may be passed as either binary data or printable data depending upon the Communications Mode.

There are two types of packets; data and request. Data packets consist of a single non-lowercase character followed by zero or more data bytes. Request packets consist of a single lowercase character. For example:

data packet	'PW\@h',<CR>
	(50 57 5C 40 68 0D hex)
request packet	'p',<CR>
	(70 0D hex)

To change a Spaceball setting or perform an operation a data packet is sent to the Spaceball. To request data from the Spaceball the corresponding lowercase request is sent to the Spaceball which then responds with the requested data packet. The Spaceball does not generate request packets.

Packets sent to the Spaceball are limited to 15 characters, excluding the terminator. When the Spaceball's input buffer fills to 16 characters an XOFF will be generated. Any more characters arriving after the XOFF will each result in an additional XOFF from the Spaceball. Up to four more characters can be received before overflowing the input buffer. Once an overflow occurs an error is generated and the packet will likely be seen as invalid.

16 bit data elements are transmitted high byte first.

The Spaceball is guaranteed not to generate packets longer than 60 characters, excluding the terminator.

4 Communications Mode

The data portion of a Spaceball packet can be in one of two forms; a binary form or a printable ASCII form. Refer to the Communications mode data "C" packet in the *Packet Reference* section. The binary mode is more compact and easier to parse but some serial port drivers act upon various control characters which can occur as data in the binary mode. In these situations the printable ASCII mode ensures only printable characters are used for transmitting data.

5 Binary Mode

Binary mode transfers data unchanged except for the special characters XON, XOFF, CR and "^" (caret - 9E hex). The caret is used to indicate one of these special characters where the character following the caret must be "Q", "S", "M" or "^" to specify XON, XOFF, CR or "^" respectively. For example,

```
Binary packets
        "D" 40 11 00 22 7F F3 F2 87 00 00 00 00 00 00
        "D" 3F 28 99 13 13 13 40 33 11 12 5E 49 67 20
RS232 Data
        44 40 5E 51 00 22 7F F3 F2 87 00 00 00 00 00 00 0D
        44 3F 28 99 5E 53 5E 53 5E 53 40 33 5E 51 12 5E 5E 49 67 20 0D
RS232 Text
        D@^Q<NUL>"<DEL>sr<BEL><NUL><NUL><NUL><NUL><NUL><NUL><CR>
        D ?(<EM>^S^S^S@3^Q<DC2>^^Ig<SP><CR>
```

Note the conventions used above where "D" is the first byte which indicates the packet type. The RS232 data is in hex format to avoid ambiguity and the textual data form is also given for clarity where <> indicates control codes and an underline indicates the top bit is set.

6 Printable ASCII Mode

Printable ASCII mode transfers data unchanged until a non-printable character or a "^" is to be sent. When one of these characters is encountered a "^" is sent to indicate a change to a packed mode. The following characters are then packed six bits to a character with the top two bits of each character being 01 unless the packed character is "?" (3F hex) which is sent unchanged. This avoids the DEL character (7F hex). The final byte is padded with 0s. For example,

```
Binary packet
        "D" 40 11 00 22 7F F3 F2 87 00 00 00 00 00 55
in binary form from the first non-printable character, 11, and separating into six bit sets
        0001 00:01  0000:0000  00:10 0010 : 0111 11:11  1111:0011  11:11 0010 :
        1000 01:11  0000:0000  00:00 0000 : 0000 00:00  0000:0000  00:00 0000 :
        0101 01:01
splitting and preceding with 01 except when the value is 3F
        44    50    40    62    5F    3F    4F    72
        51    70    40    40    40    40    40    40
        55    48
RS232 data, 5E ="^" indicates change to packed mode
        44 40 5E 44 50 40 62 5F 3F 4F 72 51 70 40 40 40 40 40 40 55 48 0D
RS232 Text
```

```
D@^DP@b_?OrQp@@@@@@UH<CR>
```

The following sections provide an overview of the operation of the Spaceball. For examples and more information refer to the packet definitions in the *Packet Reference* section.

7 Power-up & Reset Operation

When the Spaceball is powered up it performs the following operations:
- calibration EEPROM checksum
- set the default settings
- rezero (set the ball's "rest" position as the current position)
- wait 1 second then send <XON><CR> and the reset data "@" packets

If no faults were detected the Spaceball will beep twice. If faults were detected a Morse code "SOS" is beeped repetitively until a button is pressed and released. At least one complete SOS will occur irregardless of how quickly a button is pressed and released. To inform the system of the fault(s) an error data "E" packet will be generated following the reset data "@" packet.

The Spaceball can be reset to its power-on default state by sending it a reset request "@" packet. The Spaceball will respond by sending an XOFF then performing the power-up operations with the exception of the 1 second delay.

Note that the reset operation includes a rezero so the ball should be in its rest position (i.e. should not be touched) during the reset.

Refer to the relevant packet descriptions in the *Packet Reference* section for detailed information.

8 Ball Data

Information of the force and torque being applied to the ball is returned by the Spaceball in a ball data "D" data packet. This information consists of seven values; a period value, three components of force and three components of torque.

Sensing of the force and torque is done *relative* to a rest, or zero, position which normally is just the weight of the ball. There are three situations when the rest position is sensed; when the unit is powered up, when the user presses the rezero button or when instructed by an empty rezero data "Z" packet.

Ball data "D" packets are generated whenever the ball mode is “SS” and
1. the ball is not at rest and an internal MaxPulse period has expired since the last ball data “D” packet.
2. the ball has just moved from a rest state and an internal MinPulse period has expired.
3. a ball data request “d” packet has been received and MinPulse has expired.

This allows a Spaceball to be operated in either a polled mode or a streaming mode. The “pulse” period variables consist of two 12 bit unsigned integers each defining a number of milliseconds (a maximum value of 4.096 sec). Upon power-up and at reset the pulse values are set to a default of 20 msec and 1500 msec.

In order to receive Data, the Spaceball has to be switched in the “SS” mode with the mode switch command. The correct sequence for this is “MSS<CR>” (see packet reference).

Releasing the ball and allowing it to return to its rest position results in the applied force and torque returning to zero. Only one packet containing zero force and torque components is sent. The Spaceball will not generate any more ball data "D" packets while the ball is in its rest position unless a mode data "M" packet is received. Very soon after the ball is once again pushed and/or twisted, a non-zero ball data "D" packet will be generated. The zero ball data packets are very useful for determining when the ball has been released.

A ball data "D" packet can be requested with a ball request "d" packet. The Spaceball responds immediately (provided MinPulse has elapsed) with the ball data "D" packet unless the ball is in the rest position and a zero ball data packet has been sent. In this situation the Spaceball waits for the ball to be touched then automatically generates the ball data "D" packet. Additional ball request "d" packets received during the time the ball is at rest are ignored. If multiple ball request "d" packets are received by the Spaceball during a lengthy operation they will result in only one ball data "D" packet being generated.

To eliminate spurious ball data near the rest position the Spaceball employs a *null region* for the force and torque. Anything smaller than the null region is forced to zero. The null region can be set using the null region data "N" packet and interrogated using the null region request "n" packet. It is recommended that the null region be left at its default power-up value.

For diagnostic purposes the rest or zero position of the ball can be set and interrogated using the zero data "Z" packet and the zero request "z" packet. An empty zero data "Z" packet consisting of only the "Z" results in a rezero operation being performed. The Spaceball does not generate a zero data "Z" packet in response to an empty rezero data "Z" packet. A zero request "z" packet results in the generation of a zero data "Z" packet containing 12 data bytes whose format is unspecified.

Ball force and torque data is typically used to control velocity. A linear relationship between force/torque and translate/rotate velocities results in poor operation. By default the Spaceball applies a sensitization function to the force and torque vectors to produce output values more appropriate for velocity control. The default sensitization function scales each vector by a power of 2.45. i.e. more than quadratic but less than cubic. Previous α protocols provided “feel” settings which allowed the sensitization function to be set from linear to cubic. The α3 protocol only supports two feel settings of ‘p’ or ‘@’ corresponding to the default power of 2.45 or linear. The force and torque feel settings must also be the same. Internally this is implemented as a single flag that is either true or false.

Refer to the relevant packet descriptions in the *Packet Reference* section for detailed information.

9 Button Data

There are three buttons on the Spaceball Model 3003C (left, right and rezero) and 9 buttons on the Spaceball Model 2003C (1-8 and rezero). The rezero button is a special button used to instigate a rezero operation. Its state is returned for diagnostic purposes and it is not normally used by applications.

Whenever a Spaceball button is pressed or released a button data "K" packet is generated. Button status can be interrogated using the button request "k" packet which results in the generation of a button data "K" packet.

When the rezero button is pressed a rezero operation is instigated where the current readings are taken and thereafter used as the zero or rest position. The beeper is enabled while the rezero button is depressed. When the rezero button is released the EEPROM values are reloaded into RAM overwriting any current values.

Refer to the relevant packet descriptions in the *Packet Reference* section for detailed information.

10 Beeper Operation

The Spaceball beeper is operated by sending a beeper data "B" packet to the Spaceball containing a beep sequence. A beep sequence is made up of the letters A-O and a-o (with the top bit set). Uppercase letters are pauses, lowercase letters are beeps. The lower five bits specify the duration in 1/30ths second. For example:

"a" is a 1/30th second beep
"o" is a 15/30ths (or 1/2) second beep
"A" is a 1/30th second pause
"o" is a 15/30ths (or 1/2) second pause

The Spaceball beep queue can hold up to 15 bytes. An error is generated if the beeper queue overflows.

There are two additional special beep sequence characters; "@" and "`" (60 hex). "@" is used to immediately stop the beeper and clear the beeper queue. "`" is used to have the Spaceball generate a beeper data "B" packet when the "`" reaches the head of the beep queue. This would allow, for whatever reason, for beep sequences longer than the limit of 15 characters to be implemented.

Refer to the relevant packet descriptions in the *Packet Reference* section for detailed information.

11 Error & Help Packets

If an error occurs an error data "E" packet is generated containing one or more error codes. The error codes are given in Appendix ???.

If the Spaceball receives one or more invalid packets it will generate an *Invalid Packet* error.

The firmware version and date can be obtained by sending a help version request “hv” packet. The Spaceball will respond with a help packet of the form

```
Hv Firmware version 2.xx created on dd-mmm-yy
```

Refer to the relevant packet descriptions in the *Packet Reference* section for detailed information.

12 Other Packets

The Spaceball ignores packets beginning with a <SPACE> or a <TAB>.

13 XOFF Time-out

To avoid a deadlock situation the Spaceball will automatically resume transmission 1.5 seconds after an XOFF is received. To keep the Spaceball from transmitting a series of XOFFs would be required at, say, every second. Spaceball drivers may also employ this XON/XOFF scheme as the Spaceball should never XOFF the host for more than a fraction of a second except during a reset. In normal operation the Spaceball will only XOFF the host during a reset.

Packet Reference

"<SPACE>" and "<TAB>" Packets

The Spaceball will ignore any packets beginning with a <SPACE> or <TAB>.

"%" Echo Packet

A packet with a "%" header will be echoed back with the "%" replaced with a <SPACE>. This permits synchronization of mode changes and detection of the presence of a Spaceball.

"@" Reset Data/Request Packets

To reset the Spaceball send the string:
    @RESET
The Spaceball responds by sending an XOFF, performing an internal reset, then sending:
    <XON><CR><LF>
    @1 Spaceball alive and well after a <reset type> reset.<CR><LF>
    @2 Firmware version x.xx created on dd-mmm-yy<CR><LF>
where
    x.xx is the version. e.g. "2.41"
    dd-mmm-yy is the firmware creation date. e.g. "01-Jan-97"
If any errors occurred an error data "E" packet will be generated.

All internal parameters will be set to their defaults and a rezero operation will be performed to set the ball's rest position. The parameter defaults are:

zero position	current ball position
pulse	1500/20 msec "PW\@h"
beeper queue	"dDdE" (two beeps)
Communications mode	B

"B"/"b" Beeper Data/Request Packets

The Spaceball beeper is operated by sending a beeper data "B" packet to the Spaceball containing a beep sequence. A beep sequence is made up of one or more of the letters A-O and a-o (with the top bit set). Uppercase letters are pauses, lowercase letters are beeps. The lower five bits specify the duration in 1/30ths second. For example:
    "a" is a 1/30th second beep
    "o" is a 15/30ths (or 1/2) second beep
    "A" is a 1/30th second pause
    "o" is a 15/30ths (or 1/2) second pause
The Spaceball beep queue can hold up to 15 bytes.

A beeper data "B" packet consists of:
    B<beep sequence>
where
    <beep sequence> is defined above.

There are two additional special beep sequence characters; "@" and "`" (60 hex). "@" is used to immediately stop the beeper and clear the beeper queue. "`" is used to have the Spaceball generate a beeper data "B" packet when the "`" reaches the head of the beep queue. This would allow, for whatever reason, for beep sequences longer than the limit of 15 characters to be implemented.

"C"/"c" Communications Mode Data/Request Packets

The communications mode data "C" packet has the forms:

Cm

Cmt

where

m is one of

B	binary mode, <CR> packet terminator
b	binary mode, <CR><LF> packet terminator
P	printable ASCII mode, <CR> packet terminator
p	printable ASCII mode, <CR><LF> packet terminator

t is the XOFF timeout period in tenths of a second, the top two bits being ignored. If the timeout is 0 the timeout feature is disabled and an XON must be sent to the Spaceball to resume transmission following an XOFF.

"D"/"d" Ball Data/Request Packets

The ball data "D" packet has the form:

DppxxyyzzXXYYZZ

where

pp is the period

xx,yy,zz are the sensitized force vector components

XX,YY,ZZ are the sensitized torque vector components

Refer to the Ball Mode “M” packet for enabling/disabling the Ball Data “D” packet.

"E" Error Data Packet

If one or more errors occur the Spaceball generates an error packet of the form:

E<error code list>

where

<error code list> are one or more error codes, each code being an uppercase alphabetic character.
Each error data "E" packet will contain no more than seven error codes.

"F"/"f" Feel Data/Request Packets

The sensed force and torque vectors are commonly used to control translational and rotational velocity. To achieve a nice feel the raw linear vectors need to be scaled in a non-linear way. This is called “sensitizing” the vectors. There are two internal settings in previous α protocols, one for translation and one for rotation. Each determine how the raw vector is scaled in length from a linear through to a cubic function. The parameters are six bit values. A setting of 00 being linear and 3F hex being cubic. The default setting is 30 hex which corresponds to a power of 2.45. The α3 protocol only supports three settings, 00, 30 and 3F.

The feel “F” packet has the forms

FT<feel>

FB<feel>

FR<feel>

where

(feel & 0x3F) is either

0x00 - linear

0x01 to 0x38 - overridden to 0x30 which is a power of 2.45 (default)

0x39 to 0x3f - overridden to 0x3f which is a power of 3.0

NOTE: ‘@’& 0x3F = 0x00, ‘p’& 0x3F = 0x30, ‘?’& 0x3F = 0x3F

"H"/"h" Help Data/Request Packets

Help data "H" packets provide textual information on Spaceball specifications and can also explain error codes. The α3 protocol has only two help packets, the firmware version and the sensing range.

The firmware version help data "H" packet is requested using:
    hv
and has the form:
    HvVx.xx dd-mmm-yy
where
    x.xx is the firmware version
    dd-mm-yy is the firmware creation date

The sensing range can be used by more versatile implementations to allow Spaceball with different force and torque sensing ranges to be cleanly supported. Future Spaceball products may have different sensing ranges. Without the sensing range information the user would be forced to adjust the sensitivity for each different Spaceball.

To request the sensing range use:
    hs
The Spaceball will respond with:
    Hssff.ffN t.ttttNm nnbit
where
    ff.ff is a textual number specifying the force in Newton (N)
    t.tttt is a textual number specifying the torque in Newton-meter (Nm)
    nn is a textual number specifying the number of bits of resolution
The force and torque values correspond to the maximum value for each x, y and z component which can be generated. Use the C sscanf( ) format string, "`Hss%gN %gNm %dbit`", when reading the data. The SpaceController responds with "Hss20.48N 0.5632Nm 10bit"; resolutions of 40mN and 1.1mNm.

"K"/"k" Button Data/Request Packets

Whenever a Spaceball button is pressed or released a button data "K" packet is generated:
    Kbb
where
    bb are two bytes of the form
        01ZR 0000   01LR 0000
    where
        Z is the rezero button
        L is the left button
        R is the right button
The "K" stands for "Key".

The rezero button status is provided for diagnostic purposes.

Button status can be queried at any time by sending a button request "k" packet:
    k
to which the Spaceball responds by sending a button data "K" packet.

"M" Mode Switch Command

Use this command to switch the Spaceball into “automatically send Data command”:

M

With the mode as a parameter:

"SS"

Where SS is the mode you want to switch the Spaceball into (e.g. "MSS<CR>").

Once the Spaceball is switched into this mode, a "D" packet is send every time the ball is manipulated until the user releases the ball and he returns into his rest position. Once the ball reached the rest position, one final data packet with all zero's is send to indicate that the ball is no longer manipulated.

"N"/"n" Null Region Data/Request Packets

To eliminate spurious ball data near the rest position the Spaceball employs a *null region* for the force and the torque. A null region defines the magnitude of the smallest vector. Anything smaller than the null region is forced to zero. The null region can be set using the null region data "N" packet and interrogated using the null region request "n" packet. It is recommended that the null region be left at its default power-up value. To prevent old drivers from overriding the factory set null region the packet has been modified from the previous specifications to require three characters.

A Spaceball will respond to a null region request "n" packet:

    n

with a null region data "N" packet consisting of:

    N<null region>!

where

    <null region> is a number of unspecified dimension.

To set the null region send a null region data "N" packet to the Spaceball:

    N<null region><pad>

where

    <pad> is a padding character that is ignored.

For backwards compatibility the two character form will simply be echoed back:

    N<null region>

"P"/"p" Pulse Data/Request Packets

Ball data "D" packets are generated whenever the ball mode is "SS" and

1. the ball is not at rest and an internal MaxPulse period has expired since the last ball data "D" packet.
2. the ball has just moved from a rest state and an internal MinPulse period has expired.
3. a ball data request "d" packet has been received and MinPulse has expired.

This allows a Spaceball to be operated in either a polled mode or a streaming mode. The "pulse" period variables consist of two 12 bit unsigned integers each defining a number of milliseconds (a maximum value of 4.096 sec). Upon power-up and at reset the pulse values are set to a default of 20 msec and 1500 msec.

MinPulse and MaxPulse can be set using the pulse data "P" packet and interrogated using the pulse data request "p" packet.

A Spaceball will respond to a pulse request "p" packet:

    p

with a pulse data "P" packet consisting of:

    PMMmm

where

    MM is the MaxPulse packed as two 6 bit values high byte first
    mm is the MinPulse packed as two 6 bit values high byte first

To set the pulse send a pulse data "P" packet to the Spaceball.

"Z"/"z" Rezero Data/Request Packets

A Spaceball senses force and torque *relative* to a rest, or zero, position which normally is just the weight of the ball. There are three situations when the rest position is sensed; when the unit is powered up, when the user presses the rezero button or when instructed by an empty rezero data "Z" packet.

To rezero a Spaceball send an empty rezero data "Z" packet:

```
Z
```

The Spaceball will not respond with a rezero data packet.

To obtain the rezero data "Z" packet use the rezero request "z" packet:

```
z
```

The rezero data "Z" packet has the form:

```
Z<zero position>
```

where

<zero position> is 12 bytes of data. The format is unspecified and should only be used by diagnostic programs.

Appendix A
Error Codes

@	Hardware error
A	Calibration checksum error
B	Hardware error
C	Hardware error
D	Transmit timeout
E	Receive overflow (host did not stop for XOFF)
F	Receive error
G	Beeper queue overflow
H	Packet too long

Appendix B
Revision History

10SEP97	JAH	1.0	Created
15SEP97	JAH	1.1	Changed ‘Q’ packet to ‘F’ packet
18SEP97	JAH	1.2	Added ‘H’ error code
08OCT97	JAH	1.3	Changed ‘N’ packet to three characters
09OCT97	JAH	1.4	Modified the feel ‘F’ packet to include cubic sensitivity
24APR00	AK	1.5	Added mode switch command “M”.