TG-1000 W Commands

The TG-1000 controller can be interfaced through High Level commands(HL cmds) and W Commands(W cmds). HL cmds use ASCII characters, human readable, very verbose, easy to construct. W cmds are binary and ideal for hardware to hardware communication. In fact all internal communication between Tiger Comm card and Device cards are done in W cmds. No special operation is required to switch from HL cmds to W cmds, Tiger Comm automatically detects and processes them accordingly.

This document describes the packet structures of W commands and replies, gives examples, and lists the command set in detail. A complete and functional W command includes a card address and a W command packet.

Document Conventions

In this document, the terms *device* and *card* are defined such that devices are mechanical objects connected to cards by cables. Thus one or more devices, e.g., filterwheels or stage axis motors, may be connected to each card.
Non-readable data characters are represented in this document as hexadecimal values in the formats 0xFF and **`#FF`**, may be delimited by a following space when part of a character string. The space character is represented by its byte value: **`#20`**. Readable characters appear as themselves or enclosed in single quotes, e.g., 'A'. For example, the following string includes the data characters 0xFF, '1', 'A', and 0x0D:
`#FF 1A#0D` or **`#FF1A#0D`**
This string may also be represented this way:
`#FF #31 #41 #0D` or **`#FF#31#41#0D`**

W Command and Reply Structure

All W command packets are the structure below. They contain an Address byte, Command set ID, Command ID, Argument length. Argument optional and depend on the command itself.

Table 1 W Command Structure

Card Address 1 byte (0x30 for Comm, 0xFE for broadcast, 0x31.. for device cards)	Command Set ID 1 byte 0xD7 for W cmd	Command ID 1 byte	Argument length 1 byte	Argument 0-251 bytes

Table 2 TG-1000 addresses

Addressee	Usage	Value
TG-1000 Comm	Hard coded, re-assignable	0x30 ('0')
Stage/ FW/Shutter	Unique Address	0x31 to 0x39, then 0x81 to 0xF5

Stage Broadcast	Recognized by all stage controllers	0xF6
Filterwheel Broadcast	Recognized by all FW controllers	0xF7
Shutter Broadcast	Recognized by all shutter controllers	0xF8
LCD Broadcast	Recognized by all LCD controllers	0xF9
Broadcast	Recognized by all cards	0xFD
Broadcast except Comm	Recognized by all cards except TG-1000 Comm	0xFE

Address **0x30** to **0x39** and then **0x81** to **0xF5** are unique addresses. At any given time, only one card in the controller can have them. When a W command is sent, all cards receive it and parse it, but will not act unless the address matches. If the user likes to address multiple cards with the same Command, then he can use the Broadcast command.

Argument length denotes the number of bytes to follow in the packet. W command receivers use the argument length byte to count the remaining incoming characters. When that many characters have been received, processing begins.

Reply Structure

The W reply packet has two elements: the outcome byte and the reply data. The outcome byte is one of the values specified in the table below.

Outcome	Reply data
1 byte	0...∞ bytes

Table 3 W Reply Structure

Character	Hex	Description
ENQ	0x05	The argument length byte does not match the specified value for that command.
ACK	0x06	The command is well-formed, and execution has begun.
BEL	0x07	The argument length byte value exceeds the capacity of TG-1000's input buffers.
NAK	0x15	One of the following has occurred: the cmd id byte is undefined; or an argument is not in the specified value range, or the command is not recognized by the addressed device.
CAN	0x18	An intercharacter timeout (2ms) expired before the expected number of bytes was received.

Command Set

Table 4 W user Commands and Replies describes all W commands with Command Set ID **0xD7**. The commands described in Table 5 W Internal Command and Replies are for use on the backplane bus and not recommended for transmission from an external host. Table 6 Planned W commands, not implemented.

Table 4 W user Commands and Replies

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x01	Move Axis Absolute	1	Stage	Move axis to given position. On the XY controller, axis[0] is the X axis. On the ZF controller, axis[0] is the Z axis.	Argument: 5 bytes. Byte 1: axis selector 0..3. Byte 2: destination position given in 1/10 microns, an IEEE-754 single precision floating point number. Reply: ACK or NAK for out of range argument Example: **#31#D7#01#05#00#46#40#E4#01** ***#06*** Moves first axis on card #1 by 1234.5 microns
0x02	Move Axis Relative	1	Stage	Move axis to given position relative to its current position.	Argument: 5 bytes. Byte 1: axis selector 0...3. Bytes 2-5: destination position given in 1/10 microns, an IEEE-754 single precision floating point number. Reply: ACK or NAK for out of range argument Example: **#31#D7#02#05#01#C6#40#E4#01** ***#06*** Rel moves 2nd axis on card #1 by -1234.5 microns

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x03	Spin Axis	1	Stage	Apply motor power to an axis	Argument: 2 bytes Byte 1: axis selector 0..3 Byte 2: motor power, a signed character in range –128 to 127, where 0=no power. Format is 2s complement. Reply: ACK or NAK for out of range argument Example: **#31#D7#03#02#00#32** ***#06*** Rel moves 1st axis on card #1 by +50 **#31#D7#03#02#00#CE** ***#06*** Rel moves 1st axis on card #1 by -50
0x04	Set Axis Position	1	Stage	Coerces current axis position	Argument: 5 bytes. Byte 1: axis selector 0..3 Bytes 2-5: the position given in 1/10 microns, an IEEE-754 single precision floating point number. Reply: ACK or NAK for out of range argument Example: **#31#D7#04#05#00#46#40#E4#01** ***#06*** Sets first axis on card #1 position as 1234.5 microns
0x08	Halt	0	Stages	Halt all movement of all axes.	Argument: none Reply data: none Since there is no reply, this command can be broadcast to all stages. Example: **#31#D7#08#00** Halts all movement on all axis in just card #1 **#FE#D7#08#00** Halts all movement on all stage class cards in controller.

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x0a	Get Status and Position	6	Stage	Asks stage controller for status byte (see TG-*1000 Programming Guide* section on RDSBYTE), and axis position.	Argument: 1 byte Byte 1: axis selector 0..3 Reply data: 6 bytes Byte 1: ACK Byte 2: Status (1 byte. See *MS-2000 Programming Guide* section on RDSBYTE for bit definitions, reprinted below, and other information. Note: The following excerpt from *MS-2000 Programming Guide* may be out of date.) Bit 0: 0 = No commanded move is in progress. 1 = A commanded move is in progress. This bit is synonymous with the STATUS command. If the bit is set, then STATUS returns 'B', otherwise STATUS returns 'N'. (Note that this bit is not axis-specific. If either axis is busy, 'B' will be returned.) Bit 1: 0 = The axis is disabled. It can be renabled by one of the following: High Level command MC <axis>+, cycling the clutch switch for the Z axis, Low Level StartMotor command (hex 47), or a system reset. This feature is available in versions 6.2c and later; 1 = The axis is enabled. Bit 2: 0 = Motor is active (on); 1 = Motor is inactive (off). Bit 3: 0 = Joystick/Knob disabled, 1 = Joystick/Knob enabled Bit 4: 0 = Motor not ramping, 1 = Motor ramping Bit 5: 0 = Ramping up, 1= Ramping down Bit 6: Upper limit switch: 0 = open, 1 = closed Bit 7: Lower limit switch: 0 = open, 1 = closed Bytes 3-6: Position given in 1/10 Microns, an IEEE-754 single precision floating point number. It's a function of the axis unit multiplier, set with the HL cmd "UM", default value is 10,000 or mm/10000. Note: Pre v2.7, units were millimeters. Example: Card#1's 1st axis when idle **#31#D7#0A#01#00** ***#06#0A#00#00#00#00*** Card#1's 1st axis while active **#31#D7#0A#01#00** ***#06#0F#46#40#E3#B4***

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x0c	Get Status N/B	1	Stage	Same as STATUS command.	Argument: None Reply data: 'N' or 'B' Example: Card #1 when idle **#31#D7#0C#00** ***#4E*** (0x4E hex for N) Card#1 when active **#31#D7#0C#00** ***#42*** (0x42 hex for B)
0x0d	Set Resolution	1	Stage	Selects decimal point of WHERE command in high level command set.	Argument: 1 byte Byte 1: 0..3 for numer of decimal points. Reply: ACK or others Default setting is tenths of a micron resolution. This command replaces its counterparts described in the section "SETUP CONTROL COMMANDS" in the *TG-1000 Programming Guide*. Example: **#31#D7#0D#01#03** ***#06*** **w x** ***:A 12344.700*** ***#0A***

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x0e	Get Axis Names	4 to 6	Stage	Requests names of the axes supported by a stage card. Also returns number of valid axes, given by Byte 1 of the reply.	Argument: none Reply data: 4 to 6 bytes Byte 1: ACK Byte 2: Number of valid axis names to follow Byte 3: Axis 0 name Byte 4: Axis 1 name Byte 5: Axis 2 name Byte 6: Axis 3 name Example: STD XY card on address 1 **#31#D7#0E#00** ***#06#02#58#59*** (0x58 is X and 0x59 is Y) A 4ch MicroMirror card on address 2 **#32#D7#0E#00** ***#06#04#50#51#52#53*** (0x50 to 0x53 is P,Q,R and S)
0x0f	Get Single Axis Position	4	Stage	Requests position of one axis in millimeters	Argument: 1 byte Byte 1: axis selector 0..3 Reply data: 4 bytes Bytes 1-4: Axis position given in **1/10 microns**, an IEEE-754 single precision floating point number. It's a function of the axis unit multiplier, set with the HL cmd "UM", default value is 10,000 or mm/10000. Note: Pre v2.7, units were millimeters. Example: Card#1 X is at 1234.4 microns and Y is at -1234.5 microns **#31#D7#0F#01#00** ***#46#40#E3#B4*** **(12344.92578125)** **#31#D7#0F#01#01** ***#C6#40#E2#D2*** (-12344.705078125)

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x14	Get Device Class	2	Any	Queries each card directly, about what class it is. Note: Piezo drive cards, micro mirror drive cards are also classed as stage cards.	Argument: none Reply data: 2 byte Byte 1: ACK Byte 1: device type: 0=Comm; 1=Stage; 2=Filterwheel; 3=Shutter; 4=LCD; 255=no device (reply timeout) Note: If no device is present at the given bus address, then no reply is sent. Example: In a Controller with 2 stage cards and 1 comm card **#30#D7#14#00** ***#06#30*** **#32#D7#14#00** ***#06#31*** **#31#D7#14#00** ***#06#31*** **#33#D7#14#00** ***(no reply)***

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x16	Get Device Map Element	3	Comm	Queries the Comm card for its list of cards and their class present in the controller. Each time the command is give, Comm card moves down the list, prints the card address and its class. When it runs out of cards, it starts back at the top. Note: Piezo drive cards, micro mirror drive cards are also classed as stage cards.	Argument: none Reply data: 3 bytes Byte 1: ACK Byte 2: bus address Byte 3: device type (‘0’ (0x30)=Comm; ‘1’ (0x31)=Axis; ‘2’ (0x32)=Filterwheel; ‘3’ (0x33)=Shutter; ‘4’ (0x34)=LCD) Example: In a Controller with 2 stage cards and 1 comm card **#30#D7#16#00** ***#06#30#30*** (first time, reports itself) **#30#D7#16#00** (note cmd is always directed to 9omm.. Card) ***#06#31#31*** (second time card#1 details are sent) **#30#D7#16#00** ***#06#32#31*** (3rd time card#2 details are sent) **#30#D7#16#00** ***#06#30#30*** (starts back at the top) **#30#D7#16#00** ***#06#31#31*** **#30#D7#16#00** ***#06#32#31*** **#30#D7#16#00** ***#06#30#30***
0x17	Get Number of Devices	2	Comm	Host command queries the Comm card for total number of card of all classes present in the controller.	Argument: none Reply data: 2 Byte 0: ACK Byte 1: number of devices listed in the device map. Example: In a Controller with 2 stage cards and 1 comm card **#30#D7#17#00** ***#06#03*** (Answer 3 cards)

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x19	Get Stage Axis Settings	23	Stage	Host command to stage controller. Gets Speed (S), Backlash (B), Drift error (E), Finish error (PC), Ramp time (AC), Mouse controls (X, Y, or Scroll) and encoder polarity	Argument: 1 byte Byte 1: axis selector 0..3 Outcome: NAK for argument out of range 0…*n*-1, where *n* = the number of axes supported by the installed firmware build type. See Get Number of Axes command. Reply data: 23 bytes Byte 1: ACK Bytes 2-5: max speed in mm/sec, an IEEE-754 single precision floating point number. Bytes 6-9: backlash in mm, an IEEE-754 single precision floating point number. Bytes 10-13: drift error in mm, an IEEE-754 single precision floating point number. Bytes 14-17: finish error in mm, an IEEE-754 single precision floating point number. Bytes 18-19: ramp time in ms, a 16-bit unsigned integer. Byte 20: 1=pointing device X movement controls this axis; 0=pointing device X movement does not control this axis. Byte 21: 1=pointing device Y movement controls this axis; 0=pointing device Y movement does not control this axis. Byte 22: 1=pointing device scroll wheel movement controls this axis; 0=pointing device scroll wheel movement does not control this axis. Byte 23: Encoder polarity. 0=negative (left side Z); 1=positive (right side Z) Example: **#31#D7#19#01#00** ***#06#40#B7#DE#93#3D#23#D7#0A#39#D1#B7#17#37#CB #42#4B#00#64#00#00#00#01*** **#31#D7#19#01#01** ***#06#40#B7#DE#93#3D#23#D7#0A#39#D1#B7#17#37#CB #42#4B#00#64#00#00#00#01*** **#31#D7#19#01#02** ***#06#15*** (NACK as no 3rd axis) #06 (ACK) #40#B7#DE#93 (5.74591970443726 mm/s) #3D#23#D7#0A (0.0399999991059303 mm) #39#D1#B7#17 (0.00039999998989515 mm) #37#CB#42#4B (2.42303558479762e-05 mm) #00#64 (100 ms) #00 #00 #00 #01

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x1a	Move Filterwheel	1	Filterwheel	Host to Filterwheel command	Argument: 2 bytes Byte 1: Wheel selector. Value: 0…1 Byte 2: Filter selector. Values 0…7 Reply data: ACK Example: **#32#D7#1A#02#00#05** ***#06*** Moves FW #0 to position 5
0x1b	Move Shutter	1	Shutter	Host to shutter command	Argument: 2 bytes Byte 1: Shutter selector. Value = 0…1 Byte 2: Energize shutter=1, De-energize shutter = 0 Reply data: ACK
0x1c	Display Filterwheel Address	1	Filterwheel	Host to Filterwheel. Causes Filterwheel front panel 7 segment display to display device's bus address.	Argument: none Reply data: ACK Example: **#FE#D7#1C#00** Causes all FW class cards to display their Address
0x1d	Restore Filterwheel Display	1	Filterwheel	Host to Filterwheel. Used after Display Filterwheel Address command. Causes Filterwheel to restore its panel 7 segment display to the state it was in before the Display Filterwheel Address command was issued.	Argument: none Reply data: ACK
0x1e	Get Number of Axes	2	Stage	Host to stage controller. Returns number of axes supported on this card.	Argument: none Reply data: 2 byte Byte 1: ACK Byte 2: number of axes supported on this card Example: **#31#D7#1E#00** ***#06#02*** (2 axes)
0x1f	Save Filterwheel Settings	1	Filterwheel	Host to Filterwheel. Causes Filterwheel controller to its write current settings to non-volatile memory.	Argument: none Reply data: ACK

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x20	Write Filterwheel Settings to RAM	1	Filterwheel	Host to Filterwheel. Writes some current Filterwheel settings (see argument) to volatile memory.	Argument:12 bytes defined the same as the Reply to the command Read Filterwheel Settings from RAM. Reply data: ACK Bytes 1-4: Channel 0 Filterwheel offset, a signed long integer. Byte 5: Channel 0 speed value Bytes 6-9: Channel 1 Filterwheel offset, a signed long integer. Byte 10: Channel 1 speed value Byte 11-12: Shutter normal state
0x21	Read Filterwheel Settings from RAM	13	Filterwheel	Host to Filterwheel. Transmits settings from RAM to host.	Argument: none Reply data: 13 bytes Bytes 0: ACK Bytes 1-4: Channel 0 Filterwheel offset, a signed long integer. Byte 5: Channel 0 speed value Bytes 6-9: Channel 1 Filterwheel offset, a signed long integer. Byte 10: Channel 1 speed value Byte 11: Shutter normal state Byte 12: number of currently operable wheels attached Shutter normal state bits are defined as follows: Bit 0: 0=Shutter 0 normally open; 1=Shutter 0 normally closed Bit 1: 0=Shutter 1 normally open; 1=Shutter 1 normally closed Bits 2-3: not used Bit 4: 1=Shutter controller (SH2 card) is connected at this address; 0=No SH2 card is connected at this address. Bits 5-7: not used
0x24	Confirm Halt	1	Internal	Comm to Stage Confirms that Halt command was executed successfully and stage is stopped. This command is invoked automatically when the H command HALT is transmitted from Host to Comm. If the result indicates that an axis failed to halt, then the system is automatically reset and all motors are shut down.	Argument: none Reply data: 1 byte Byte 1: ACK = all axes halted, none were in motion when Halt command was given, or this is not the first Confirm Halt command issued since the most recent Halt command, or no Halt commands have been issued since the last system reset; NAK = at least one axis failed to halt; ENQ = a Halt command was issued prior to this command, and an axis was in motion when that Halt command was issued.

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x25	Zero Axis	1	Stage	Comm to Stage Duplicates H command ZERO and MS-2000 Zero button function. Causes readjustment of limits.	Argument: 1 byte Byte 1: axis selector 0..3 Reply data: ACK Example: **#31#D7#25#01#00** Zeros just 1st axis on card#1 **#FE#D7#25#01#00** Zeros all 1st axes on all cards of stage class.
0x26	Get Axis Types	3	Stage	Comm to Stage Reports whether each axis is an XY, motor-driven focus, or piezo-driven focus axis. Users may also want to look at 0x4A Get Axis Kinds too. It is better implemented.	Argument: none Reply data: 3 bytes Byte 1: ACK Byte 2: Axis 0 type, where 0=no axis present; 1=XY; 2=motor-driven focus; 3=piezo-driven focus; 4=motor-driven zoom; 5=theta Byte 3: Axis 1 type, defined same as Byte 1.

<table>
<tr><th>ID hex</th><th>Name</th><th>Reply length</th><th>Normal recipient</th><th>Description</th><th>Argument(s)/Reply</th></tr>
<tr><td>0x4A</td><td>Get Axis Kinds</td><td>4 to 6</td><td>Stage</td><td>Queries the Device card. Replies with total axes present, followed by ASCII codes for each axis representing which kind of axis.</td><td>Argument: none
Reply data: 4 to 6 bytes
Byte 1: ACK
Byte 2: Total axis on card (and number of bytes to follow)
Byte 2: Axis 0 type
Byte 3: Axis 1 type
Byte 4: Axis 2 type
Byte 5: Axis 3 type

Example:
#31#D7#4A#00
#06#02#78#78 (0x78 is x for xymotor)
#32#D7#4A#00
#06#04#75#75#75#75 (0x75 is u for MicroMirror)

<table>
<tr><th>Axis Type Short</th><th>Description</th></tr>
<tr><td>x</td><td>XY stage</td></tr>
<tr><td>z</td><td>Z focus motor drive. LS50s, Z scopes etc</td></tr>
<tr><td>p</td><td>Piezo Focus. ASIs ADEPT, Piezo DAC etc</td></tr>
<tr><td>o</td><td>Objective Turret</td></tr>
<tr><td>f</td><td>Filter Changer</td></tr>
<tr><td>t</td><td>Theta Stage</td></tr>
<tr><td>l</td><td>Generic linear motorized stage, TIRF, SISKIYOU etc</td></tr>
<tr><td>a</td><td>Generic linear piezo stage</td></tr>
<tr><td>m</td><td>Zoom magnification motor axis</td></tr>
<tr><td>u</td><td>Micro Mirror, Scanner 75 etc</td></tr>
<tr><td>w</td><td>Filter Wheel</td></tr>
<tr><td>s</td><td>Shutter</td></tr>
<tr><td>u</td><td>Unknown axis type</td></tr>
</table>
</td></tr>
</table>

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x4B	Get Axis Props	4 to 6	Stage	Queries the Device card. Replies with total axes present, followed by byte for each axis representing any special properties or capabilities (usually would be firmware module) such as CRISP or RING BUFFER.	Argument: none Reply data: 4 to 6 bytes Byte 1: ACK Byte 2: Total axis on card or number of bytes to follow Byte 2: Axis 0 properties Byte 3: Axis 1 properties Byte 4: Axis 2 properties Byte 5: Axis 3 properties Example: **#34#D7#4B#00** ***#06#02#0A#0A*** (xystage with RING BUFFER and ARRAY) **#33#D7#4B#00** ***#06#04#10#10#10#10*** (Micromirror with SPIM) Bit 0: CRISP auto-focus firmware Bit 1: RING BUFFER firmware Bit 2: SCAN firmware Bit 3: ARRAY firmware Bit 4: SPIM firmware Bit 5: SINGLEAXIS and/or MULTIAXIS firmware Bits 6-7: reserved

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x27	Set Stage Axis Settings	1	Stage	Comm to Stage. Host command to stage controller. Counterpart to Get Axis Settings command. Sets Speed (S), Backlash (B), Drift error (E), Finish error (PC), Max lim (SU), Min lim (SL), Ramp time (AC), Mouse controls (X, Y, or Scroll), and Encoder polarity (EPOL). This command is designed for the Settings window of TG-1000.exe.	Argument: 23 bytes Byte 1: axis selector 0..1 Bytes 2-5: max speed in mm/sec, an IEEE-754 single precision floating point number. Bytes 6-9: backlash in mm, an IEEE-754 single precision floating point number. Bytes 10-13: drift error in mm, an IEEE-754 single precision floating point number. Bytes 14-17: finish error in mm, an IEEE-754 single precision floating point number. Bytes 18-19: ramp time in ms, a 16-bit unsigned integer. Byte 20: 1=pointing device X movement controls this axis; 0=pointing device X movement does not control this axis. Byte 21: 1=pointing device Y movement controls this axis; 0=pointing device Y movement does not control this axis. Byte 22: 1=pointing device scroll wheel movement controls this axis; 0=pointing device scroll wheel movement does not control this axis. Byte 23: Encoder polarity. 0=negative (left side Z); 1=positive (right side Z) Reply data: ACK Example: To set 1st axis to 2mm/sec **#31#D7#27#17#00#40#00#00#00#3D#23#D7#0A#39#D1 #B7#17#37#CB#42#4B#00#64#00#00#00#01** ***#06***
0x28	Save Settings Stage	1	Stage	Writes current stage settings to non-volatile memory.	Argument: none Reply data: ACK Example: **#31#D7#28#00** saves settings of just card #1 to non volatile memory **#FE#D7#28#00** saves settings of all cards in controller to nonvolatile memory.
0x29	Get Saved Settings Stage	1	Stage	Reads stage settings from non-volatile memory into stage volatile memory, overwriting current settings.	Argument: none Reply data: ACK

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x2A	Restore Stage Defaults	1	Stage	Marks stage non-volatile memory as unsaved. Next stage reset, the settings will be the original factory defaults.	Argument: none Reply data: ACK
0x2B	Restore Filterwheel Defaults to RAM	1	Filterwheel	Writes default settings to Filterwheel RAM.	Argument: none Reply data: ACK
0x2C	Read Filterwheel Settings to RAM	1	Filterwheel	Host to Filterwheel. Reads settings from non-volatile memory to Filterwheel RAM.	Argument: none Reply data: ACK
0x2D	Reset Stage	1	Stage	Host to stage. Invokes stage software reset.	Argument: none Reply data: ACK **#31#D7#2D#00** resets just card #1. **#FE#D7#2D#00** resets all stage class cards in the controller
0x2F	Ping	1	All	Replies ACK. Used to verify that sender has sent a command readable to the receiver. May be used to seek serial baud rate match.	Argument: none Reply data: ACK
0x31	Set Clutch	1	Stage	Engages or disengages clutch.	Argument: 1 byte Byte 1: 0=disengage; 1=engage. Reply data: ACK.
0x32	Get Stage Settings And Flags	9	Stage	Return states of system flags.	Argument: none Reply data: 9 bytes Byte 1: ACK Bytes 2: XY pitch Byte 3: Z pitch Some common pitches: PITCH_A_FINE = 'A', (0x41) PITCH_B_COARSE = 'B', (0x42) PITCH_C_ULTRA_COARSE = 'C', (0x43) PITCH_25NM = 'H', (0x48) PITCH_NORM_Z = 'N', (0x4E) PITCH_D_ULTRA_FINE = 'U', (0x55) SD_ACTUATOR = 'X', (0x58) PITCH_ZEISS_Z = 'Z', (0x5A)

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
					GTS_A_FINE = 'a', (0x61) GTS_B_COARSE = 'b', (0x62) GTS_C_ULTRA_COARSE = 'c', (0x63) Byte 4: Where command format Byte 5: X & Y encoder flag, where 'L' = linear, 'R' = rotary Byte 5: Clutch engaged Byte 6: 1st Axis, X or Z axis profile Byte 7: 2st Axis, Y or F axis profile Some common profiles STANDARD_XY 0x00 STANDARD_Z 0x01 STD_CP_ROT 0x02 STD_FP_ROT 0x03 STD_CP_LIN 0x04 STD_FP_LIN 0x05 UCP_ROT 0x06 UUCP_ROT 0x07 UFP_ROT 0x08 UUFP_ROT 0x12 UFP_LIN 0x14 UCP_LIN 0x22 UUCP_LIN 0x23 SCOPE_STD_Z 0x0a SCOPE_LIN_Z 0x0b SD_XLATE 0x0f PIEZO_PROFILE 0x10 MM_PROFILE 0x2b Byte 9: Knob speed Example: **#31#D7#32#00** ***#06#42#46#97#52#00#02#02#05*** #06 (ACK)#42(B pitch)#46(ignore, no z)#97#52(R for rotary)#00#02(#2 profile, std xy)#02(#2 profile, std xy)#05

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x35	Set Joystick/Mouse Speeds	1	Stage	Sets slow and fast joystick speed for XY. This command does the same things as the H cmd JS.	Argument: 3 bytes Byte 1: Slow joystick speed Byte 2: Fast joystick speed Byte 3: Blank (used to be knob speed, however knob speed is handled differently now) Reply: ACK or others Example: To set joystick slow at 20% and fast at 80% **#31#D7#35#03#14#50#00** ***#06*** include a 3rd byte leave it at 0x00 to appease the controller
0x36	Get Mouse Speeds	4	Stage	Returns settings made by Set Stage Mouse Speeds command.	Argument: none Reply data: 4 bytes. See Set Mouse Speeds command. Byte 1: ACK Byte 2: Slow joystick speed Byte 3: Fast joystick speed Byte 4: Blank (used to be knob speed, however knob speed is handled differently now) Example: **#31#D7#36#00** ***#06#14#50#00*** (slow at 20% and fast at 80%)
0x37	Set Encoder Polarity	1	Stage	Sets encoder polarity. Left- or right-hand Z drives need this setting.	Argument: 2 bytes Byte 1: axis selector 0..3 Byte 2: values are 1 and -1. (2s complement) Default value is 1 for left-hand Z drive. Replay data: ACK Example: **#31#D7#37#02#00#FF** ***#06*** (sets card#11st axis as epol as -1) **#31#D7#37#02#00#01** ***#06*** (sets card#11st axis as epol as 1) Note: Save settings to nonvolatile memory(0x28) and restart controller for settings to take affect.

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x38	Get Encoder Polarity	2	Stage	Returns encoder polarity setting (see Set Encoder Polarity command)	Argument: 1 Byte 1: axis selector 0..3 Reply data: 2 bytes Byte 1: ACK Byte 2: 0xFF for -1 , 0x01 for 1 Example: **#31#D7#38#01#00** ***#06#FF*** (card#1 1st axis has negative encoder polarity)
0x39	Set Encoder Type	1	Stage	Selects linear or rotary encoder.	Argument: 1 byte Byte 1: NUL =Rotary; anything else =Linear Reply: ACK or NAK for out of range argument This setting informs the firmware about the hardware configuration with respect to encoders. Then turn the controller OFF/ON for settings to take effect. Example: **#31#D7#39#01#01** ***#06*** Sets all axis on card#1 to linear enc mode **#31#D7#39#01#00** ***#06*** Sets all axis on card#1 to rotary enc mode Note: Restart controller, for settings to take affect.
0x3A	Get Encoder Type	2	Stage	Gets encoder type—linear or rotary	Argument: None Reply data: 2 bytes Byte 1: ACK Byte 2: 0x52 , R for Rotary , 0x4C, L for Linear Example: **#31#D7#3A#00** ***#06#52***(card#1 is in rotary encoder mode)
0x3D	Home Filterwheel	1	Filterwheel	Corresponds to Filterwheel HO command.	Argument: 1 byte Byte 1: 0 or 1, Filterwheel selector Reply: ACK or others

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x3F	Get Firmware Version	Varies	Any	Returns a String containing the version number	Argument: none Reply data: Varies Example: On a stage card **#31#D7#3F#00** ***#76#32#2E#37*** (in ASCII is "v2.7") On a filter wheel card **#32#D7#3F#00** ***#56#65#72#73#69#6F#6E#3A#20#76#31#2E#32#0A*** (in ascii is "Version: v1.2<CR>")
0x40	Set Default Manual Input Device	1	Stage	Sets type of device to be used for manual movements, i.e., analog joystick or Knobs. Writes settings to nonvolatile memory. *Note: For safety, this command disables all movement. Also appropriate firmware modules must also be present on the card*	Argument: 2 bytes Byte 1: axis selector 0..3 Byte 2: device selector, Some important ones 0x00 = NONE 0x02 = Joystick – X deflection 0x03 = Joystick – Y deflection 0x05 = X-Wheel 0x06 = Y-Wheel 0x09 = JX and X-wheel combo 0x0A = JY and Y-wheel combo 0x16 = Z-Wheel 0x17 = F-Wheel Reply: ACK or NACK Example: So switch axis to X and Y knobs. **#31#D7#40#02#00#05** ***#06*** **#31#D7#40#02#01#06** ***#06***

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x41	Get Default Manual Input Device	2	Stage	Returns which kind of manual input device is used, i.e., analog joystick or knobs etc	Argument: 1 byte Byte 1: axis selector 0..3 Reply data: 2 byte Byte 1: ACK Byte 2: see Set Manual Input Device Example: **#31#D7#41#01#00** ***#06#02*** (1st axis is joystick x) **#31#D7#41#01#01** ***#06#03*** (2nd axis is joystick y)
0x43	Set Axis Speed	1	Stage	Sets speed of a single axis	Argument: 5 bytes Byte 1: axis selector 0..3 Bytes 2-5: max speed in mm/sec, an IEEE-754 single precision floating point number. Reply data: ACK or others Example : set ist axis to 2mm/sec **#31#D7#43#05#00#40#00#00#00** ***#06***
0x44	Set Encoder Counts Per Mm	1	Stage	Sets both axes' encoder counts per millimeter.	Argument: 8 bytes Bytes 1-4: for axis 0, counts per millimeter, an IEEE-754 single precision floating point number Bytes 5-8: for axis 1, counts per millimeter, an IEEE-754 single precision floating point number Reply : ACK or others Example: Set Card#1 1st and 2nd axis to 60000 counts/mm **#31#D7#44#08#47#6A#60#00#47#6A#60#00** ***#06***
0x45	Get Encoder Counts Per Mm	9	Stage	Gets encoder counts per millimeter for the specified axis	Argument: none Reply data:9 bytes Byte 1: ACK Byte 2-5: encoder counts for axis 0, in IEEE 754 format Byte 6-9: encoder counts for axis 1, in IEEE 754 format Example: **#31#D7#45#00** ***#06#47#6A#60#00#47#6A#60#00*** (#47#6A#60#00 is 60,000)

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x46	Joystick XY data	0	Stage	Sends current joystick X & Y values from Comm to Stage. Each value denotes the deflection of the joystick relative to its position at rest (center). Behaves like the spin command. Handy to simulate joystick/button in UI.	Argument: 2 bytes Byte 1: X joystick value, -127 to 127 Byte 2: Y joystick value, -127 to 127 Reply : no reply. Example: **#FE#D7#46#02#40#40** All axis in controller that have joystick as manual input, start moving. **#FE#D7#46#02#CE#CE** All axis in controller that have joystick as manual input, now move in opposite direction **#FE#D7#46#02#00#00** All axis in controller that have joystick as manual input, stop moving.
0x47	Button data	0	Stage	Sends current button state from Comm to Stage. Command has to be sent two twice, Once indicating press and the release. Based on the time interval between press and release commands, the stage card will conclude if it's a short press, or long press etc. Note: For broadcast used Address 0xF6	Argument: 2 bytes Byte 1: Button byte with bits defined as follows, its active low Bits 0-3: undefined Bit 4: Zero button state Bit 5: Home button state Bit 6: At (@) button state Bit 7: Joystick button (fast/slow) Byte 2: Clutch byte with bits defined as follows: Bits 0-5: undefined Bit 6: Clutch switch state Bit 7: undefined Reply : none Example: Zero button press and release for just Card #1 **#31#D7#47#02#E0#00** #31#D7#47#02#F0#00 Home button press for all stage cards in controller **#F6#D7#47#02#D0#00** #F6#D7#47#02#F0#00

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x48	Knob data	0	Stage	Sends left and right knob rotation values from Comm to Stage.	Argument: 4 bytes Bytes 1-2: Signed integer left knob value. Bytes 3-4: Signed integer right knob value. Reply: None Example: **#FE#D7#48#04#00#FF#00#FF** Moves all axis that respond to knobs.
0x49	Get Tiger Banner	Varies	All	Gets TIGER_BANNER string from device and relays it to host.	Argument: none Reply: TIGER_BANNER string followed by ETX message terminator. Example: **#32#D7#49#00** ***#41#74#20#33#32#3A#20#5A#3A#5A#4D#6F#74#6F#72 #2C#46#3A#5A#4D#6F#74#6F#72#20#76#32#2E#37#20 #53#54#44#5F#5A#46#20#4A#75#6C#20#33#30#20#32 #30#31#33#3A#31#36#3A#30#39#3A#35#31#03*** in ASCII its "At 32: Z:Zmotor,F:Zmotor v2.7 STD_ZF Jul 30 2013:16:09:51<ETX>"
0x4C	Set Axis Direction	1	Stage	Sets Axis direction. Setting is automatically saved into non volatile memory. Does not need a system reset. Default is 1 or positive direction.	Argument: 2 bytes Byte 1: axis selector 0..3 Byte 2: values are 1 and -1. (2s complement) Default value is 1. Replay data: ACK Example: **#31#D7#4C#02#00#FF** ***#06*** (sets card#11st axis direction as -1or negative) **#31#D7#4C#02#00#01** ***#06*** (sets card#11st axis direction as positive or 1)

ID hex	Name	Reply length	Normal recipient	Description	Argument(s)/Reply
0x4D	Get Axis Direction	2	Stage	Returns axis direction setting .	Argument: 1 Byte 1: axis selector 0..3 Reply data: 2 bytes Byte 1: ACK Byte 2: 0xFF for -1 , 0x01 for 1 Example: **#31#D7#4D#01#00** ***#06#FF*** (card#1 1st axis direction is negative)

Table 5 W Internal Command and Replies

ID	Name	Reply length	Recipient	Description	Argument(s)/Reply
0xFA	Set Filterwheel Number	0	Filterwheel	After internally assigning each filterwheel its number for FW0, FW1,… FW*n* commands and before sending Get Tiger Banner command to filterwheel, Comm sends this command so that the filterwheel can reply to Get Tiger Banner command showing its assigned number.	Argument: 2 bytes Byte 1: Unsigned char, the assigned number for wheel 0 Byte 2: Unsigned char, the assigned number for wheel 1
0xFC	H Cmd	0	Internal	Internal ASI command: copies TG-1000 Comm parser state to TG-1000 stage card.	Argument: 11 bytes Bytes 1-2: gCmd [**0x12 Move, 0x21 Zero, etc]** Bytes 3: gOp **[0x00 none, 0x01 read, 0x02 write, 0x06 plus, 0x07 minus]** Bytes 4: gSubCmd **[0x01 entry, 0x02 exit, 0x03 pseudoaxis,0x04 non rad, 0x05 read]** Bytes 5:gAxisChar **[0x58 X, 0x59 Y,etc]** Bytes 6-9: gNum , IEEE 754 format 4byte float Bytes 10: do_what **[0 GET_ACK, 1 GNUM,2 GSTRING,3 LONG REPLY, 4 Do nothing]** Bytes 11:any_axis **[0x00 or 0x01]** example: m x=123 generates **#31#D7#FC#0B#00#12#02#04#58#42#F6#00#00#04#01**
0xFD	H Get gNum	4	Unused	Internal ASI command: request gNum to be copies from TG-1000 stage card to TG-1000 command card	

ID	Name	Reply length	Recipient	Description	Argument(s)/Reply
0x50	Get Gerror	1	Stage/Internal	Gets an error code following an High Level command. For inhouse use only	

Table 6 Planned W commands, not implemented

0x05	AutoNotify	1	Comm	Supported only by the TG-1000 Comm card, this command causes the TG-1000 Comm card to notify the host when all stage and focus controllers are "not busy," that is, not processing any of the following commands: • W Absolute Move • W Relative Move The purpose of this command is to make it unnecessary to poll the system for move completion. This command would be issued immediately after one or more of the above commands.	Argument: none Reply data: 1 byte. A single "not busy" reply message is transmitted to the host when all stages have completed any and all pending commanded movements. The reply is 'N', that is, **`#4E`**. If a system fault prevents a device from responding as expected during the transaction, then the reply is **`#17`** (ASCII DC1). Host driver software should treat such an event as cause to reset the system. If any new command is issued by the host before transmission of the "not busy" reply to the host is initiated, then TG-1000 cancels the pending AutoNotify transaction and processes the new command. In this event, no reply will ever be sent to the host in response to the pending AutoNotify command. Up to a 7 ms delay in the processing of the new command is possible, depending on the internal state of TG-1000 when the new command is received. It is possible to inadvertently configure a stage controller so that it can never become "not busy" after initiating a commanded move. In this event, TG-1000 would wait forever before replying. Issuing any new command prevents this kind of lockup.
0x07	Set Joystick Fast/Slow	0	Comm	Supported by the X/Y card, this command selects fast or slow stage movement in response to 'J' commands.	Argument: 1 byte Byte 1: 'F' = fast, 'S' = slow. Outcome: NAK for out of range argument Reply data: none This setting may be changed repeatedly during a session. This setting can be saved in nonvolatile memory (see HL command "SS Z").
0x09	Set Pitch	0	Stage	Selects coarse or fine pitch lead screw.	Argument: 1 byte Byte 1: 'C' = coarse; 'F' = fine Outcome: NAK for out of range argument Reply data: none This setting informs the firmware about the hardware configuration. It should be changed only if the hardware is changed.
0x0b	Z Speed	0	Stage	Selects fast or slow Z motor speed.	Argument: 1 byte

					Byte 1: 'F' = fast; 'S' = slow Outcome: NAK for out of range argument Reply data: none This setting informs the firmware about the hardware configuration. It should be changed only if the hardware is changed.
0x10	Get status byte	1	Stage	Request status byte	Argument: 1 byte Byte 1: axis selector 0..1 Outcome: NAK for out of range argument Reply data: 1 byte Byte 1: status byte for specified axis
0x11	Set PS2 device type	0	Stage	Host to stage, type of PS/2 device in use Argument data type defined in PS2 module header PS2.h	Argument: 1 Byte Example: Host to stage 3 address ('5'), set device to Mouse **5#D7#88#01#01** Byte 1: PS2 device type where 0=None, 1=Mouse. Reply data: none
0x12	Set PS2 data modifiers	0	Stage	Host to stage, modifies responses to PS/2 device	Argument: 4 bytes, interpreted as signed char Example: Host to stage 0 address ('1') **#31#D7#89#04#xx#xx#xx#xx** Byte 1 : X multiplier Byte 2 : Y multiplier Byte 3 : Z multiplier Byte 4: Button mask, where bit values 0=disable, 1=enable, bits defined as follows: Bits 7-5: not used Bit 4: 5th button Bit 3: 4th button Bit 2: Middle button Bit 1: Right button Bit 0: Left button Note: Default value of multipliers is 1. Value 0 can be used to disable an axis. Other values can be used to scale and/or change direction of movement. Default value of Button mask is 0xFF. Reply data: none
0x13	Set Axis Names	0	Stage	Assigns axis names to a stage card. If a card supports fewer than three axes, the unused axis names in the argument are ignored.	Argument: 2 bytes Byte 1: Axis 0 name Byte 2: Axis 1 name Reply data: none

0x18	Get Serial Number	20	Comm	Host command for TG-1000 Comm to read a serial number from non-volatile memory	Argument: none Reply data: 20 bytes, the serial number given for the Set Serial Number command.
0x22	Set Shutter Normal State	0	Shutter	Host to shutter command, sets whether normally open or normally closed	Argument: 2 bytes Byte 1: Shutter selector. Value = 0…1 Byte 2: Normally open=0; normally closed=1. Reply data: none
0x23	available, not used				
0x2E	Shutter Invert	0	Filterwheel	Inverts normally open/closed bit in selected shutter	Argument: 1 byte Byte 1: 0=shutter 0; 1=shutter 1 Reply data: none
0x30	Set Baud Rate	0	Comm	Writes new baud rate selector to volatile memory. The host serial connection operates at the new baud rate after the next reset. Saves setting in non-volatile memory.	Argument: 1 byte Byte 1: baud rate selector where 0=115200 baud; 1=57600 baud; 2=19200 baud; 3=9600 baud. Reply data: none
0x33	Set Stage Settings And Flags	1	Stage	Sets states of system flags and writes to non-volatile memory.	Argument: 8 bytes Reply: ACK See get Stage Settings And Flags command.
0x34	Set Stage Setup Control	0	Stage	Supports the functions provided by the MS-2000 Setup Control Command described in the SETUP CONTROL COMMAND section of the Low Level Format document.	Argument: 1 byte Byte 1: ‘A’, ‘B’, ‘R’, ‘H’, ‘T’, or ‘E’ Reply data: none
0x3B	Get Clutch State	2	Stage	Gets state of clutch. (Clutch may not be implemented in Tg-1000)	Argument: none Reply data: 2 byte. Byte 1: ACK Byte 2:See Set Clutch command.
0x3C	Set Knob Speed	0	Stage	Corresponds to JS Z=??, i.e., JS_KNOB setting. (Knob setting needs to point to XY_KOBS and ZF_KNOBS speed)	Argument: 1 byte Byte 1: value range is 0…127. Reply data: none
0x3E	Get PS2 Data Modifiers	4	Stage	Returns PS2 data modifiers (PS2 stuff needs more testing)	Argument: none Reply data: 4 bytes See Set PS2 data modifiers command (0x12).
0x42	Get Manual Input Device	1	Stage	Returns current manual input device. At times during a session, the current manual input device is a different type from the default manual input device.	Argument: 1 byte Byte 1: axis selector 0..1 Reply: 1 byte

					Byte 1: see Set Manual Input Device
0xFB	Set Serial Number	0	Comm	Host command for TG-1000 Comm to write a serial number to non-volatile memory	Argument: 20 bytes Bytes 1-20: Serial number ASCII string, left-justified, padded with spaces. Reply data: none
0xfe	PS2 Data	0	Internal	Comm to device(s) Argument data types defined in PS2 module header PS2.h, derived from the following document: http://www.computer-engineering.org/ps2mouse/ Requiring no reply, this message is meant to be broadcast. PS2/ mouse stuff need to be tested more	Argument: 5 Bytes Byte 1: PS2 type, where 0=Standard, 1=Scroll Wheel, 2=Five-button. Scroll Wheel and Five-Button are Microsoft Intellimouse extensions of PS/2. Byte 2: Status byte, bits defined as follows for all types: Bit 7: Y overflow Bit 6: X overflow Bit 5: Y sign Bit 4: X sign Bit 3: Always 1 Bit 2: Middle Button Bit 1: Right Button Bit 0: Left Button Byte 3: X Movement Byte 4: Y Movement Byte 5: Standard, Scroll wheel: Bits 7-0: Z Movement Five-Button: Bits 7-6: Always 0 Bit 5: 5th button Bit 4: 4th button Bits 3-0: Z movement Reply data: none

Table 7 Change log

8/2/2013	Branched from old TG-1000 software manual, and updated to reflect TigerComm v1.8 correctly	Vik
8/20/2013	Commands 0x0A and 0x0F reply in units of 1/10 microns or function of axis unit multiplier.	Vik
9/20/2013	Added command 0x4B to get axis properties	Jon

10/14/2013	Expanded on 0xFC command	Vik
2/28/2014	Added new command 0x4C and 0x4D to set and get axis direction	Vik