Serial Data Protocol between PC-based software and the MACH-DSP Servo Driver
(Mach DSP firmware version 12.1)

The protocol is based on transmitting a four byte "**command**" from the PC to the servo, and then receiving a four-byte "**response**" sent by the servo back to the PC. Only one **command** (Read Virtual Scope Data) returns more than four bytes to the PC. Moreover, **responses** are never sent to the PC unsolicited.

For **command** data sent by the PC and received by the servo amplifier, the four bytes are designed to be unique and identifiable, so that if serial communication begins (or is interrupted and resumes) in the middle of a stream, the **commands** can be identified and serial communication can be synchronized. For **response** data sent by the servo amplifier and received by the PC, four bytes are also used, and the sequence is generally similar.

To establish confidence in the message that is received by the servo, the first byte of a **command** always has its most significant bit set. The following two or three bytes have their most significant bits arranged into a combination that is guaranteed to be unique and identifiable, even if the communication is started in the middle of a stream.

The four-byte **command** indicates whether a read or write happens (read from servo to PC, or write from PC to servo), indicates 256 items which can be read or written, and communicates up to 15 bits of data written or read.

The most significant bit combination of **commands** whose item number is 0 through 127 will be 1, 0, 0, x, meaning the most significant bit is set in the first byte, cleared in the next two bytes, and may or may not be cleared in the last byte.

The most significant bit combination of **commands** whose item number is 128 through 255 will be 1, 1, 1, 0, meaning the most significant bit is set in the first three bytes, and cleared in the last byte. This means item numbers 128 through 255 can only be used to communicate 7-bit data. These item numbers are used only to communicate simple integer information, usually well-under 7-bits. Item numbers 128 through 255 are indicated with blue text in this document.

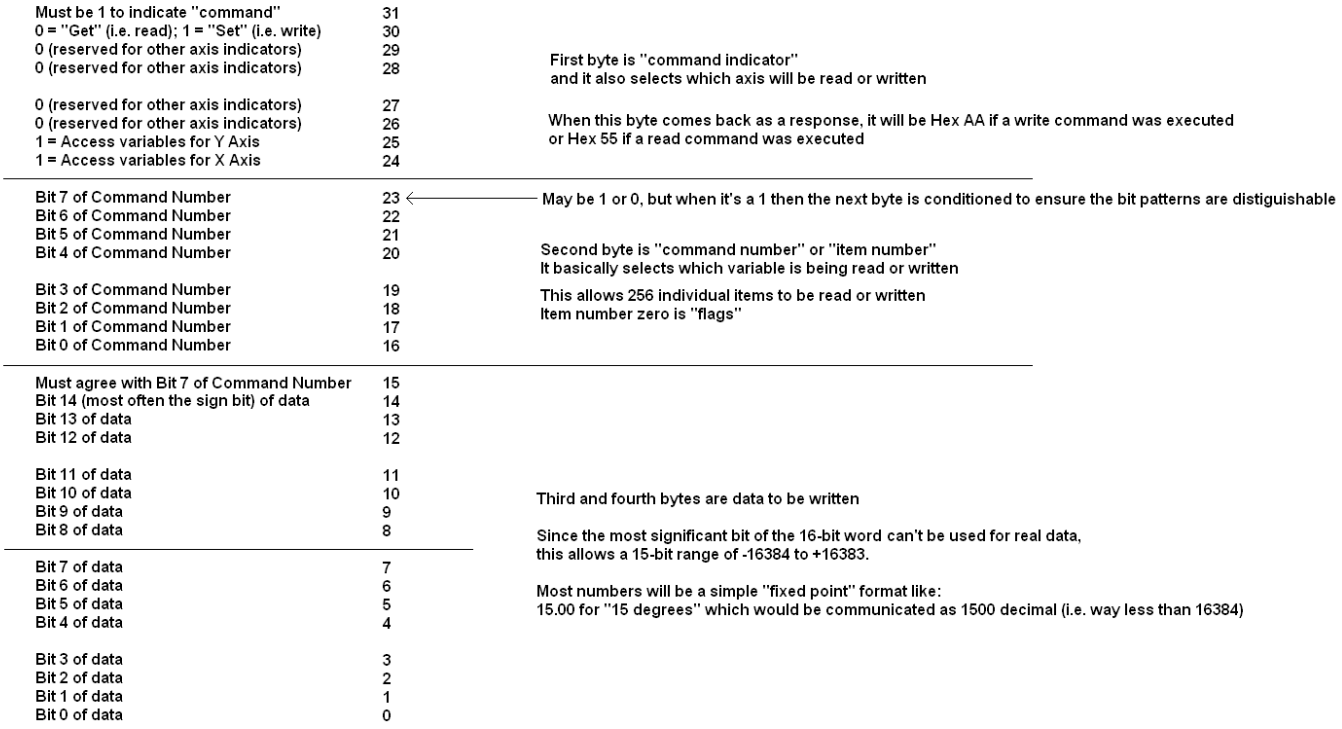
This means the pattern of most significant bits in the four-byte command must be either 1000, 1001 or 1110.

To illustrate how this is unique and identifiable, consider the case where serial communication is initiated and only the last byte of the four-byte **command** is received, followed by another four-byte **command**. In this case, if the most significant bit of the last byte was set, then the next byte will have its most significant bit set, and next two bytes will have their most significant bit cleared, giving a received pattern of 1, 1, 0, 0. However, this is not a valid bit pattern for commands and so this can be rejected.

The four-byte **response** sent by the servo to the PC is formatted very similarly to the original **command** that initiated the **response**. The main difference between **commands** and **responses**, lies in the first byte (although in some cases, other bytes may also be different). The first byte of a **response** will be Hex AA if a write operation was performed, and will be Hex 55 if a read operation was performed. These are deemed unique, and although either AA or 55 might appear in the last byte of a four-byte **command**, certainly the next three bytes won't coincidentally echo those of a command sent, and so this is believed to be a pretty reliable protocol. Another differences is that the third byte of a response always has its most significant bit cleared.

As mentioned above, the protocol allows for the communication of up to 15 bits of information per item. The servo itself is fully floating point and many variables communicated to and from the board will need to communicate a number having precision greater than integers can convey. In most cases this is accomplished by establishing a fixed decimal place position, i.e. by multiplying an internal number by 100 or 1000. For example the number 1.23 may be sent and received by this protocol as simply 123 thereby fixing the decimal point at two places.

Structure of commands written from the PC to the DSP board



The communication is facilitated through the commonplace asynchronous method, with one start bit, 8 data bits, one stop bit, and no parity. The baud rate used to communicate the protocol data is 256 k-bits per second.

Note that although the following pages use hexadecimal notation, this is only for the convenience of the reader, and to avoid spelling out 32 ones-and-zeros for each command. Nevertheless, it is not human-readable hex data encoded in ASCII that is sent over the serial port. Rather all communication is facilitated through binary data.

General Status and Control

The following commands provide general information about the Mach DSP, as well as the ability to control overall aspects of the servo driver. These commands affect the overall system, not aspects of a single axis.

Command: 8000 0000 Read Servo Status Flags **00**

Response: 5500 yyxx

yy = eight status flags for Y axis
xx = eight status flags for X axis

Status flags are explained below:
Bit 15 = Always Zero
Bit 14 = Y Servo Ready
Bit 13 = Y Power Supply Fault
Bit 12 = Y AGC Fault
Bit 11 = Y Scanner body or coil temperature Exceeded
Bit 10 = Y Position Exceeded
Bit 9 = Y Output Limiter Active
Bit 8 = Y Slew Rate Limiter Active

Bit 7 = Watchdog Timer Tripped
Bit 6 = X Servo Ready
Bit 5 = X Power Supply Fault
Bit 4 = X AGC Fault
Bit 3 = X Scanner body or coil temperature Exceeded
Bit 2 = X Position Exceeded
Bit 1 = X Output Limiter Active
Bit 0 = X Slew Rate Limiter Active

Command: 8x01 0y00 Read power supply voltage levels and CPU clock rate **01**
Response: 5501 zzzz

x = Axis indicator, but this is ignored since the power supply is shared between X and Y axis and we have dedicated numbers for reading X and Y voltages
y = A number indicating the voltage you want to read, as explained below

0 = +24V supply voltage * 100
1 = -24V supply voltage * 100
2 = +15V supply voltage * 100
3 = -15V supply voltage * 100
4 = +3.3V supply voltage * 100
5 = +1.1V supply voltage * 100
6 = Actual CPU Clock rate / 1,000,000 (not the rate requested by command F6)
7 = X-axis AGC voltage * 100
8 = Y-axis AGC voltage * 100
9 = Instantaneous X-axis Position (may be used for external test equipment such as PD linearity testers)
10 = Instantaneous Y-axis Position

11=CPU Load
12=+24V supply current * 100
13=-24V supply current * 100
14=
15=

zzz = voltage level * 100 for voltages (for example, 2400 is the default value for the +24V supply)
zzz = current level * 100 for currents (for example, 20 is the default value for the +24V supply)
zzz = clock rate / 1,000,000 for clock rate (for example 400 for 400MHz)
zzzz = Full 16-bit signed Position from ADC

Command:	8x02 0y00	Read firmware information	**02**
Response:	5502 0zzz		

x = Axis indicator, but this is ignored since the power supply is shared between X and Y axis
y = Firmware information selector

1 = Firmware major (version) – This would be “1” if the version/revision was “1.5”
2 = Firmware minor (revision) – This would be “5” if the version/revision was “1.5”
3 = Firmware build year – This would be “2014” if the firmware was built in 2014
4 = Firmware build month – This would be “1” if the firmware was built in January
5 = Firmware build day – This would be “15” if the firmware was built on the 15th day
6 = Mach-DSP Serial Number
7 = Mach-DSP Additional Serial Number

zzz = portion of firmware information requested

Command: 8x03 0y00 Read Performance Metric Data **03**
Response: 5503 zzzz

x = Axis indicator – axis for which to return the performance metric data
y = Performance Metric to return

0 = Step response time (measured in samples)
1 = Overshoot amount
2 = Undershoot amount
3 = Final error amount

zzzz = Performance metric to return

When 0 is selected as the performance metric, zzzz will be the time in samples. With the nominal sample rate of 160kHz, each sample represents 6.25 microseconds. ILDA 30K is represented by 300 microsecond step response, so zzzz would return a value of 48 (300 microsecond step time / 6.25 microseconds per sample).

Command:	8x04 0000	Read Number Of Times Per Minute to update voltage status	**04**
Response:	5504 0yyy		

Command:	Cx04 0yyy	Write Number Of Times Per Minute to update voltage status (stored in Flash)
Response:	AA04 0yyy	

x = Axis indicator, but this is ignored since this is a system-level variable

yyy = Times per minute (must be positive and between 0 and 600)

300 is the default value

Command:	8x05 0000	Read Master Clock Divider	**05**
Response:	5505 00yy		

Command:	Cx05 00yy	Write Master Clock Divider
Response:	AA05 00yy	

x = Axis indicator, but this is ignored since the Master Clock Divider is the same for X and Y axis

yy = Master Clock Divider (must be positive and between 10 and 20)

20 is the default value

Command: 8x06 0000 Read Servo Driver Circuit Board temperature **06**
Response: 5506 yyyy

x = Axis indicator, but this is ignored since the XY mount is shared between X and Y axis

yyyy = Temperature * 100

4000 is the default value

Command:	8x07 0000	Read LDAC Timing	**07**
Response:	5507 00yy		

Command:	Cx07 00yy	Write LDAC Timing
Response:	AA07 00yy	

x = Axis indicator, but this is ignored since the LDAC Timing is the same for X and Y axis

yy = LDAC Timing (must be positive and between 0 and 128)

0 is the default value

Command:	8x08 0000	Read Desired Sample Rate (in K samples per second)	**08**
Response:	5508 0yyy		

Command:	Cx08 0yyy	Write Desired Sample Rate (in K samples per second
Response:	AA08 0yyy	

x = Axis indicator, but this is ignored since the sample rate is the same for X and Y axis

yyy = Desired Kilo Samples Per Second (must be positive and between 30 and 400)

160 is the default value.

Command:	8x08 0001	Read **Actual** Sample Rate (in samples per second * 10)
Response:	5508 zzzz	

zzzz = Actual Samples Per Second multiplied by 10

16,129 is the default value, which means an actual sample rate of 161,290 samples per second.

Command: 8x09 0000 Read Coil Temperature **09**
Response: 5509 yyyy

x = Axis indicator (must be either X or Y axis)

yyyy = temperature * 100

3000 is the default value

Command: 8x0A 0000 Read Thermistor Temperature, normally attached to XY Mount **0A**
Response: 550A yyyy

x = Axis indicator (X or Y axis or neither, in which case the temperature of the XY Mount is returned)

yyyy = temperature * 100

if yyyy=-1, then Thermistor is not connected.

3000 is the default value

The Thermistor interface is only capable of reading temperatures between +5C and +125C.

Scope Test Point Outputs (those outputs that come from DACs on the board)

Command:	8x0B 0y00	Read Test Point Number for Scope Output	**0B**
Response:	550B 0yzz		

Command:	Cx0B 0yzz	Write Test Point Number for Scope Output
Response:	AA0B 0yzz	

x = Axis indicator, but this is ignored since the scope outputs are shared between X and Y axis

y = Scope Output (1 through 6)

zz = Test Point Number (0 through 255)

Virtual Scope Test Points (those test points returned to the PC)

Command:	8x0C 0y00	Read Test Point Number for Virtual Scope Output	**0C**
Response:	550C 0yzz		

Command:	Cx0C 0yzz	Write Test Point Number for Virtual Scope Output
Response:	AA0C 0yzz	

x = Axis indicator, but this is ignored since the scope outputs are shared between X and Y axis

y = Virtual Scope Output (1 through 6)

zz = Test Point Number (0 through 255)

Virtual Scope Sample Rate

Command:	8x0D 0000	Read Virtual Scope Samples Per Point	**0D**
Response:	550D 00yy		
Command:	Cx0D 00yy	Write Virtual Scope Samples Per Point	
Response:	AA0D 00yy		

x = Axis indicator, but this is ignored since the sample rate is the same for X and Y axis

yy = Number of scanner samples per point collected by the Virtual Scope (1-100)

1 is the default value

Virtual Scope Trigger Mode

Command:	8x0E 0000	Read Virtual Scope Trigger Mode	**0E**
Response:	550E 000y		

Command:	Cx0E 000y	Write Virtual Scope Trigger Mode
Response:	AA0E 000y	

x = Axis indicator, but this is ignored since the scope outputs are shared between X and Y axis

y = Trigger mode: 0= Freerun; 1= rising edge of A; 2=falling edge of A; 3=rising edge of B; 4=falling edge of B, 5=rising edge of C, 6=falling edge of C; 7=rising edge of D; 8=falling edge of D; 9=rising edge of X input command; 10=falling edge of X input command; 11=rising edge of Y input command; 12=falling edge of Y input command.

20 is the default value (2.0 amps per volt)

Read Virtual Scope Data

Command: 8x0F 0y00 Read Virtual Scope Data **0F**
Response: 550F 0y0z followed by 500 bytes of data

x = Axis indicator, but this is ignored since the scope outputs are shared between X and Y axis

yy = Virtual Scope Output (1 through 6)
z = Buffer number used to return the data (used to detect whether or not the scope has been triggered)

Note: Virtual Scope buffer #5 is the X-axis frequency sweep results, and #6 is the Y-axis results
500 points will be returned although only the first 382 are valid.

Scope and Virtual Scope Test Point Scale Factor

Command:	8x10 0y00	Read Scale Factor for scope and virtual scope test points	**10**
Response:	5510 0yzz		
Command:	Cx10 0yzz	Write Scale Factor for scope and virtual scope test points	
Response:	AA10 0yzz		

x = Axis indicator, but this is ignored since the scope outputs are shared between X and Y axis
y = Scale factor select

y = 0 for current-related test points. In this case zz is 1 to 100 representing tenth-amps per division
y = 1 for temperature-related test points. In this case zz is 1 to 100 representing degrees C per division
y = 2 for position-related test points. In this case zz is 1 to 100 representing tenths of a degree per division
y = 3 for error-related test points. In this case zz is 1 to 100 representing tenths of a degree per division
y = 4 for integral-related test points. In this case zz is 1 to 120 representing degrees (not tenths) per division
y = 5 for velocity-related test points. In this case zz is 1 to 100 representing counts per division
y = 6 for output-related test points. In this case zz is 1 to 100 representing percent per division
y = 7 for voltage-related test points. In this case zz is 1 to 100 representing tenth-volts per division

20 is the default value for current-related test points (2.0 amps per volt)
20 is the default value for temperature-related test points (20 degrees C per volt)
30 is the default value for position-related test points (3.0 degrees per volt)
30 is the default value for error-related test points (3.0 degrees per volt)
80 is the default value for integrator-related test points (8.0 degrees per volt)
10 is the default value for velocity-related test points (10 counts per volt)
20 is the default value for output-related test points (20 percent per volt)
50 is the default value for voltage-related test points (5.0 volts per volt)

TTL Input and Output behaviors

Command:	8x11 0000	Read TTL Input 1 Polarity	**11**
Response:	5511 000y		

Command:	Cx11 000y	Write TTL Input 1 Polarity
Response:	AA11 000y	

x = Axis indicator, but this is ignored since the TTL Inputs are shared between X and Y axis

y = Polarity: 0= Inactive (no current flowing through opto-isolator); 1= Active (current flowing through opto-isolator).

Command:	8x12 0000	Read TTL Input 1 Action	**12**
Response:	5512 00yy		

Command:	Cx12 00yy	Write TTL Input 1 Action
Response:	AA12 00yy	

x = Axis indicator, but this is ignored since the TTL Inputs are shared between X and Y axis

yy = Action: 0= Ignored; 1=Deactivate X-axis Servo; 2=Deactivate Y-axis Servo; 3=Deactivate both X- and Y-axis Servo; 4=Mute X-axis Input Command; 5=Mute Y-axis Input Command; 6=Mute both X- and Y-axis Input Command; 7=Select between Tuning 0 and 1; 8=Select between Tuning 2 and 3; 9=Select between Tuning 1-2-3-4

Command:	8x13 0000	Read TTL Input 2 Polarity	**13**
Response:	5513 000y		

Command:	Cx13 000y	Write TTL Input 2 Polarity
Response:	AA13 000y	

x = Axis indicator, but this is ignored since the TTL Inputs are shared between X and Y axis

y = Polarity: 0= Inactive (no current flowing through opto-isolator); 1= Active (current flowing through opto-isolator).

Command:	8x14 0000	Read TTL Input 2 Action	**14**
Response:	5514 00yy		

Command:	Cx14 00yy	Write TTL Input 2 Action
Response:	AA14 00yy	

x = Axis indicator, but this is ignored since the TTL Inputs are shared between X and Y axis

yy = Action: 0= Ignored; 1=Deactivate X-axis Servo; 2=Deactivate Y-axis Servo; 3=Deactivate both X- and Y-axis Servo; 4=Mute X-axis Input Command; 5=Mute Y-axis Input Command; 6=Mute both X- and Y-axis Input Command; 7=Select between Tuning 1-2-3-4

Command:	8x15 0000	Read TTL Output 1 Qualifier	**15**
Response:	5515 yyyy		

Command:	Cx15 yyyy	Write TTL Output 1 Qualifier
Response:	AA15 yyyy	

x = Axis indicator, but this is ignored since the TTL Outputs are shared by X and Y axis

yyyy = -10000 to 10000, which is scaled to mean -100.00 to 100.00

0 is the default value

Command: 8x16 0000 Read TTL Output 1 Dwell Time **16**
Response: 5516 yyyy

Command: Cx16 yyyy Write TTL Output 1 Dwell Time
Response: AA16 yyyy

x = Axis indicator, but this is ignored since the TTL Outputs are shared by X and Y axis

yyyy = Dwell time in milliseconds (0 to 15,000)

0 is the default value

Command:	8x18 0000	Read TTL Output 2 Qualifier	**18**
Response:	5518 yyyy		

Command:	Cx18 yyyy	Write TTL Output 2 Qualifier
Response:	AA18 yyyy	

x = Axis indicator, but this is ignored since the TTL Outputs are shared by X and Y axis

yyyy = -10000 to 10000, which is scaled to mean -100.00 to 100.00

0 is the default value

Command:	8x19 0000	Read TTL Output 2 Dwell Time	**19**
Response:	5519 yyyy		

Command:	Cx19 yyyy	Write TTL Output 2 Dwell Time
Response:	AA19 yyyy	

x = Axis indicator, but this is ignored since the TTL Outputs are shared by X and Y axis

yyyy = Dwell time in milliseconds (0 to 15,000)

0 is the default value

Dynamic Signal Analyzer

Command:	801B 0000	Read Dynamic Signal Analyzer Channel	**1B**
Response:	551B 000y		

Command:	C01B 000y	Write Dynamic Signal Analyzer Channel
Response:	AA1B 000y	

y = Enable Dynamic Signal Analyzer for channel. 0 = disabled, 1 = X axis, 2 = Y axis

0 is the default value

Function Generator

Command:	8x1D 0000	Read Function Generator Frequency	**1D**
Response:	551D yyyy		

Command:	Cx1D yyyy	Write Function Generator Frequency
Response:	AA1D yyyy	

x = Axis indicator but this is ignored

yyyy = Frequency in Hz (0 to 16,000)

0 is the default value

Note: When Function Generator Frequency is zero and Function Generator Amplitude is non-zero, a frequency sweep is performed on the axis indicated above by Command 1B. The results of the frequency sweep are stored in the Virtual Scope Buffer #5 for the X Axis and Virtual Scope Buffer #6 for the Y axis.

While the sweep is in progress, you can periodically read the frequency that is being used in the sweep by executing Command 1D. However since the protocol can normally only return numbers between –16383 and + 16383, it means that the normal Command 1D could only be used to monitor frequencies below 16383 Hz. Since the sweep occurs up to 60,000 Hz, a modified version of Command 1D is needed. This is described below:

Command:	8x1D 0001	Read Function Generator Frequency **(Divided By 10)**
Response:	551D zzzz	

zzzz = Frequency in Hz divided by 10

Command:	8x1E 0000	Read Function Generator Amplitude	**1E**
Response:	551E yyyy		

Command:	Cx1E yyyy	Write Function Generator Amplitude
Response:	AA1E yyyy	

x = Axis indicator, but this is ignored since the function generator is shared by X and Y axis

yyyy = Centi-Degrees Peak-to-Peak (0 to 4000)

0 is the default value

Note: When Function Generator Amplitude is non-zero, servo input is ignored and servo input comes from the function generator

Command:	8x1F 0000	Read Function Generator Waveform	**1F**
Response:	551F 000y		

Command:	Cx1F 000y	Write Function Generator Amplitude
Response:	AA1F 000y	

x = Axis indicator, but this is ignored since the function generator is shared by X and Y axis

y = 0 for Quadrature Sinewave; 1 for Quadrature Squarewave; 2 for In-phase Squarewave; 3 for Bearing Exercise Move; 4 for Arbitrary Waveform 1 (presently Triangle-wave); 5 for Arbitrary Waveform 2.

0 is the default value

Servo Settings

The following commands provide control of each axis, and provide the ability to communicate 15-bits of data.

Command:	8x20 0000	Read Current ADC Scale Factor	**20**
Response:	5520 yyyy		

Command:	Cx20 yyyy	Write Current ADC Scale Factor
Response:	AA20yyyy	

x = Axis indicator

yyyy = Number of centi-amps for internal full scale reading (100 to 2000)

1000 is the default value (which means 10.00 amps)

Command:	8x21 0000	Read Current Offset	***(OBSOLETE)***	**21**
Response:	5521 yyyy			

Command:	Cx21 yyyy	Write Current Offset
Response:	AA21 yyyy	

x = Axis indicator

yyyy = Number of milliamps of offset (-100 to +100)

0 is the default value

Command:	8x22 0000	Read Current Averaging Filer Constant	**22**
Response:	5522 yyyy		

Command:	Cx22 yyyy	Write Current Averaging Filter Constant
Response:	AA22 yyyy	

x = Axis indicator

yyyy = Filter constant (1 to 10000)

500 is the default value

Command: 8x23 0000 Read Magnet Temperature Averaging Filer Constant **(OBSOLETE)** **23**
Response: 5523 yyyy

Command: Cx23 yyyy Write Magnet Temperature Averaging Filter Constant
Response: AA23 yyyy

x = Axis indicator

yyyy = Filter constant (1 to 10000)

10 is the default value

Command: 8x24 0000 Read Voltage ADC Scale Factor **24**
Response: 5524 yyyy

Command: Cx24 yyyy Write Voltage ADC Scale Factor
Response: AA24yyyy

x = Axis indicator

yyyy = Number of volts for internal full scale reading (100 to 6000)

4000 is the default value (which means 40.00 volts)

Command:	8x25 0000	Read Voltage Offset	***(OBSOLETE)***	**25**
Response:	5525 yyyy			

Command:	Cx25 yyyy	Write Voltage Offset
Response:	AA25 yyyy	

x = Axis indicator

yyyy = Number of millivolts of offset (-100 to +100)

0 is the default value

Command:	8x26 0000	Read Magnetic Spring Return current	**26**
Response:	5526 yyyy		
Command:	Cx26 yyyy	Write Magnetic Spring Return current	
Response:	AA26 yyyy		

x = Axis indicator

yyyy = Milliamps at 10 degrees mechanical (-300 to +300)

0 is the default value

Command:	8x27 0000	Read Maximum Operating Coil Temperature	**27**
Response:	5527 0yyy		

Command:	Cx27 0yyy	Write Maximum Operating Coil Temperature
Response:	AA27 0yyy	

x = Axis indicator

yy = Degrees C (25 to 150)

100 is the default value (meaning 100 degrees C)

Command:	8x28 0000	Read Coil Temperature for Servo shutdown	**28**
Response:	5528 0yyy		

Command:	Cx28 0yyy	Write Coil Temperature for Servo shutdown
Response:	AA28 0yyy	

x = Axis indicator

yy = Degrees C (25 to 150)

130 is the default value (meaning 130 degrees C)

Command:	8x29 0000	Read Coil Temperature Servo Gain	**29**
Response:	5529 yyyy		

Command:	Cx29 yyyy	Write Coil Temperature Servo Gain
Response:	AA29 yyyy	

x = Axis indicator

yyyy = Servo Gain Constant (10 to 10000)

400 is the default value

Command:	8x2A 0000	Read Coil Temperature Servo Offset	**2A**
Response:	552A yyyy		

Command:	Cx2A yyyy	Write Coil Temperature Servo Offset
Response:	AA2A yyyy	

x = Axis indicator

yyyy = Servo Offset (10 to 10000)

1000 is the default value

Command:	8x2B 0000	Read Coil Inductance	**2B**
Response:	552B yyyy		

Command:	Cx2B yyyy	Write Coil Inductance
Response:	AA2B yyyy	

x = Axis indicator

yyyy = Coil inductance (10 to 10000, corresponding to 10uH to 10mH)

100 is the default value

Command:	8x2C 0000	Read Back EMF voltage	**2C**
Response:	552C yyyy		

Command:	Cx2C yyyy	Write Back EMF voltage
Response:	AA2C yyyy	

x = Axis indicator

yyyy = Back EMF voltage (1 to 1000)

50 is the default value

Command:	8x2F 0000	Read Command Input Source	**2F**
Response:	552F yyyy		

Command:	Cx2F yyyy	Write Command Input Source
Response:	AA2F yyyy	

x = Axis indicator

yyyy = Input Source: 00 = Analog; 1 = XY2-100-compatible digital; 2 = FB4-compatible digital

0 is the default value

Command:	8x30 0000	Read Input ADC Scale Factor
Response:	5530 yyyy	

Command:	Cx30 yyyy	Write Input ADC Scale Factor
Response:	AA30 yyyy	

x = Axis indicator

yyyy = Number of Centi-Degrees mechanical that corresponds to maximum input voltage (-15000 to 15000)

1500 is the default value (which means 15.00 degrees)

Command:	8x31 0000	Read Input Offset
Response:	5531 yyyy	

Command:	Cx31 yyyy	Write Input Offset
Response:	AA31 yyyy	

x = Axis indicator

yyyy = Centi-Degrees of offset (-2000 to 2000)

0 is the default value

Command:	8x32 0000	Read Input Shear Angle	**32**
Response:	5532 yyyy		

Command:	Cx32 yyyy	Write Input Shear Angle
Response:	AA32 yyyy	

x = Axis indicator

yyyy = Centi-Degrees of shear (-4500 to 4500)

0 is the default value

Command:	8x33 0000	Read Positive Slew Rate Limit	**33**
Response:	5533 yyyy		

Command:	Cx33 yyyy	Write Positive Slew Rate Limit
Response:	AA33 yyyy	

x = Axis indicator

yyyy = Tenths of a degrees per millisecond (0 to 3000)

1000 is the default value which means 10.00 degrees per millisecond; 0 means “disabled”

Command:	8x34 0000	Read Negative Slew Rate Limit	**34**
Response:	5534 yyyy		

Command:	Cx34 yyyy	Write Negative Slew Rate Limit
Response:	AA34 yyyy	

x = Axis indicator

yyyy = Tenths of a degrees per millisecond (0 to 3000)

1000 is the default value which means 10.00 degrees per millisecond; 0 means “disabled”

1500 is the default value; 0 means “disabled”

Command:	8x35 0000	Read Slew Rate Logarithmic Factor	**35**
Response:	5535 00yy		

Command:	Cx35 00yy	Write Slew Rate Logarithmic Factor
Response:	AA35 00yy	

x = Axis indicator

yy = Log Factor (0 to 100)

1 is the default value

Command: 8x36 0000 Read Command Signal Positive Clamp **36**
Response: 5536 yyyy

Command: Cx36 yyyy Write Command Signal Positive Clamp
Response: AA36 yyyy

x = Axis indicator

yyyy = Centi-Degrees (-3000 to +3000)

1500 is the default value

Command:	8x37 0000	Read Command Signal Negative Clamp	**37**
Response:	5537 yyyy		

Command:	Cx37 yyyy	Write Command Signal Negative Clamp
Response:	AA37 yyyy	

x = Axis indicator

yyyy = Centi-Degrees (-2000 to +2000)

-1500 is the default value

Command:	8x38 0000	Read Position Sensor Linearity	**38**
Response:	5538 00yy		

Command:	Cx38 00yy	Write Position Sensor Linearity
Response:	AA38 00yy	

x = Axis indicator

yy = Linearity (-127 to +127)

0 is the default value

Command:	8x39 0000	Read Position ADC Scale Factor	**39**
Response:	5539 yyyy		
Command:	Cx39 yyyy	Write Position ADC Scale Factor	
Response:	AA39 yyyy		

x = Axis indicator

yyyy = Number of Centi-Degrees that corresponds to maximum detectable angle (200 to 4000)

2000 is the default value

Command: 8x3A 0000 Read Position Offset **3A**
Response: 553A yyyy

Command: Cx3A yyyy Write Position Offset
Response: AA3A yyyy

x = Axis indicator

yyyy = Centi-Degrees of offset (-2000 to +2000)

0 is the default value

Command:	8x3B 0000	Read Position Maximum Positive Angle Before Servo Cutoff	**3B**
Response:	553B yyyy		

Command:	Cx3B yyyy	Write Position Maximum Positive Angle Before Servo Cutoff
Response:	AA3B yyyy	

x = Axis indicator

yyyy = Centi-Degrees (100 to 4000)

+2000 is the default value

Command:	8x3C 0000	Read Position Maximum Negative Angle before Servo Cutoff	**3C**
Response:	553C yyyy		

Command:	Cx3C yyyy	Write Position Maximum Negative Angle before Servo Cutoff
Response:	AA3C yyyy	

x = Axis indicator

yyyy = Centi-Degrees (-100 to -4000)

-2000 is the default value

Command:	8x3D 0000	Read Magnetic Center Position Offset	**3D**
Response:	553D yyyy		

Command:	Cx3D yyyy	Write Magnetic Center Position Offset
Response:	AA3D yyyy	

x = Axis indicator

yyyy = Centi-Degrees of offset (-1000 to +1000)

0 is the default value

Command:	8x40 0000	Read Error Modifier Table Index	**40**
Response:	5540 00yy		

Command:	Cx40 00yy	Write Error Modifier Table Index
Response:	AA40 00yy	

x = Axis indicator but this is ignored because there is only one Error Modifier index

yy = Table Index (0 to 41 with each index representing a mechanical degree from –40 to +40)

Command:	8x41 0000	Read Error Modifier Table Value	**41**
Response:	5541 yyyy		

Command:	Cx41 yyyy	Write Error Modifier Table Value
Response:	AA41 yyyy	

x = Axis indicator

yyyy = Centi-Degrees (-2000 to +2000)

Note: It is expected that you would do a Write to Command 40 with a value of zero, to reset the table index. Each time you do Command 41, it retrieves the value at the index and automatically increments the index. Therefore you can do Command 40 followed by 41 calls to Command 41 to read and write the entire table. Since there is only one index per axis, it is recommended that you read or write all X table values, then read or write all Y table values.

Command:	8x42 0000	Read Torque Rolloff Table Index	**42**
Response:	5542 00yy		

Command:	Cx42 00yy	Write Torque Rollorr Table Index
Response:	AA42 00yy	

x = Axis indicator but this is ignored because there is only one Torque Rolloff index

yy = Table Index (0 to 41 with each index representing a mechanical degree from –40 to +40)

Command:	8x43 0000	Read Torque Rolloff Table Value	**43**
Response:	5543 yyyy		
Command:	Cx43 yyyy	Write Torque Rolloff Table Value	
Response:	AA43 yyyy		

x = Axis indicator

yyyy = Percent * 10 (+700 to +1000; where 1000 = +1.000)

Note: It is expected that you would do a Write to Command 42 with a value of zero, to reset the table index. Each time you do Command 43, it retrieves the value at the index and automatically increments the index. Therefore you can do Command 42 followed by 41 calls to Command 43 to read and write the entire table. Since there is only one index per axis, it is recommended that you read or write all X table values, then read or write all Y table values.

Command:	8x44 0000	Read Magnetic Spring Return Table Index	**44**
Response:	5544 00yy		
Command:	Cx44 00yy	Write Magnetic Spring Return Table Index	
Response:	AA44 00yy		

x = Axis indicator but this is ignored because there is only Magnetic Spring Return index

yy = Table Index (0 to 41 with each index representing a mechanical degree from –40 to +40)

Command:	8x45 0000	Read Magnetic Spring Return Table Value	**45**
Response:	5545 yyyy		
Command:	Cx45 yyyy	Write Magnetic Spring Return Table Value	
Response:	AA45 yyyy		

x = Axis indicator

yyyy = Milliamps * 10 (-2000 to +2000; where 2000 = +200.0 milliamps)

Command:	8x50 0000	Read Exerciser Type **** RESERVED FOR FUTURE ****	**50**
Response:	5550 yyyy		

Command:	Cx50 yyyy	Write Exerciser Type
Response:	AA50 yyyy	

x = Axis indicator

yyyy = Exerciser Type (currently the only exercise method is spinning the scanner…)

Command:	8x51 0000	Read Exerciser Start Pulse Amplitude	**51**
Response:	5551 yyyy		

Command:	Cx51 yyyy	Write Exerciser Start Pulse Amplitude
Response:	AA51 yyyy	

x = Axis indicator

yyyy = Start Pulse Current (100 = 1 amp peak)

Command: 8x52 0000 Read Exerciser Start Pulse Phase **52**
Response: 5552 yyyy

Command: Cx52 yyyy Write Exerciser Start Pulse Phase
Response: AA52 yyyy

x = Axis indicator

yyyy = Portion of the first sinewave to output Start Pulse instead of normal amplitude
(250 = 90 degrees of the first sinewave; 500 = 180 degrees, etc.)

Command: 8x53 0000 Read Exerciser Start Pulse Amplitude **53**
Response: 5553 yyyy

Command: Cx53 yyyy Write Exerciser Start Pulse Amplitude
Response: AA53 yyyy

x = Axis indicator

yyyy = Start Pulse Current (100 = 1 amp peak)

Command:	8x54 0000	Read Exerciser Start Pulse Phase	**54**
Response:	5554 yyyy		

Command:	Cx54 yyyy	Write Exerciser Start Pulse Phase
Response:	AA54 yyyy	

x = Axis indicator

yyyy = Portion of the first sinewave to output Stop Pulse instead of normal amplitude
(750 = the final 90 degrees of the last sinewave sinewave; 500 = the final 180 degrees, etc.)

Command:	8x55 0000	Read Exerciser Initial Running Amplitude	**55**
Response:	5555 yyyy		

Command:	Cx55 yyyy	Write Exerciser Initial Running Amplitude
Response:	AA55 yyyy	

x = Axis indicator

yyyy = Running Current (100 = 1 amp peak)

Command:	8x56 0000	Read Exerciser Initial Running Frequency	**56**
Response:	5556 yyyy		

Command:	Cx56 yyyy	Write Exerciser Initial Running Frequency
Response:	AA56 yyyy	

x = Axis indicator

yyyy = Running Frequency in Hertz. (RPM = Frequency * 60)

Command:	8x57 0000	Read Exerciser Final Running Amplitude	**57**
Response:	5557 yyyy		

Command:	Cx57 yyyy	Write Exerciser Final Running Amplitude
Response:	AA57 yyyy	

x = Axis indicator

yyyy = Running Current (100 = 1 amp peak)

Command:	8x58 0000	Read Exerciser Final Running Frequency	**58**
Response:	5558 yyyy		
Command:	Cx58 yyyy	Write Exerciser Final Running Frequency	
Response:	AA58 yyyy		

x = Axis indicator

yyyy = Running Frequency in Hertz. (RPM = Frequency * 60)

Command:	8x59 0000	Read Exerciser Ramp Time	**59**
Response:	5559 yyyy		

Command:	Cx59 yyyy	Write Exerciser Ramp Time
Response:	AA59 yyyy	

x = Axis indicator

yyyy = Time to ramp from Initial Frequency / Amplitude to Final Frequency / Amplitude

Command:	8x5A 0000	Read Exerciser Hold Time	**5A**
Response:	555A yyyy		

Command:	Cx5A yyyy	Write Exerciser Hold Time
Response:	AA5A yyyy	

x = Axis indicator

yyyy = Time to hold at the final running Frequency / Amplitude

Command:	8x5B 0000	Read Exerciser Status	**5B**
Response:	555B yyyy		

Command:	Cx5B yyyy	Write Exerciser Status
Response:	AA5B yyyy	

x = Axis indicator

yyyy = Exerciser Status: 0 for “stop exercising”; 1 for “start” or “running”

Command:	8x68 0000	Read Sliding Hysteresis Amount	**68**
Response:	5568 yyyy		

Command:	Cx68 yyyy	Write Sliding Hysteresis Amount
Response:	AA68 yyyy	

x = Axis indicator

yyyy = Thousandths of a degree (0 to 1,000)

+200 is the default value

Command:	8x69 0000	Read Sliding Hysteresis Gain	**69**
Response:	5569 yyyy		
Command:	Cx69 yyyy	Write Sliding Hysteresis Gain	
Response:	AA69 yyyy		

x = Axis indicator

yyyy = Current Gain (0 to 2,000)

0 is the default value

Command:	8x6A 0000	Read Error Integral Positive Clamp	**6A**
Response:	556A yyyy		

Command:	Cx6A yyyy	Write Error Integral Positive Clamp
Response:	AA6A yyyy	

x = Axis indicator

yyyy = Degrees (1 to 15000)

+1000 is the default value

Command:	8x6B 0000	Read Error Integral Negative Clamp	**6B**
Response:	556B yyyy		

Command:	Cx6B yyyy	Write Error Integral Negative Clamp
Response:	AA6B yyyy	

x = Axis indicator

yyyy = Degrees (-1 to -15000)

-1000 is the default value

Command:	8x6C 0000	Read Final Torque Limit	**6C**
Response:	556C yyyy		

Command:	Cx6C yyyy	Write Final Torque Limit
Response:	AA6C yyyy	

x = Axis indicator

yyyy = Number of centi-amps for internal full scale reading (100 to 5000)

2000 is the default value (which means 20.00 amps)

Command:	8x6D 0000	Read Initial Torque Limit	**6D**
Response:	556D yyyy		

Command:	Cx6D yyyy	Write Initial Torque Limit
Response:	AA6D yyyy	

x = Axis indicator

yyyy = Number of centi-amps for internal full scale reading (100 to 5000)

2000 is the default value (which means 20.00 amps)

Command:	8x6E 0000	Read Torque Limit Filter Constant	**6E**
Response:	556E yyyy		

Command:	Cx6E yyyy	Write Torque Limit Filter Constant
Response:	AA6E yyyy	

x = Axis indicator

yyyy = Filter constant (1 to 15000)

80 is the default value

Command:	8x6F 0000	Read Velocity to Position Scale factor	***(OBSOLETE)***	**6F**
Response:	556F yyyy			

Command:	Cx6F yyyy	Velocity to Position Scale factor
Response:	AA6F yyyy	

x = Axis indicator

yyyy = Scale factor (10 to 10000)

260 is the default value

Command:	8x70 0000	Read Error Proportional (Servo Gain) Constant	**70**
Response:	5570 yyyy		

Command:	Cx70 yyyy	Write Error Proportional (Servo Gain) Constant
Response:	AA70 yyyy	

x = Axis indicator

yyyy = Gain Constant (0 to 10000)

0 is the default value

Command:	8x71 0000	Read Error Integral Gain Constant	**71**
Response:	5571 yyyy		

Command:	Cx71 yyyy	Write Error Integral Gain Constant
Response:	AA71 yyyy	

x = Axis indicator

yyyy = Gain (0 to 10000)

0 is the default value

Command:	8x72 0000	Read Position Proportional Gain Constant	**72**
Response:	5572 yyyy		

Command:	Cx72 yyyy	Write Position Proportional Gain Constant
Response:	AA72 yyyy	

x = Axis indicator

yyyy = Gain (-10000 to 10000)

0 is the default value

Command:	8x73 0000	Read Low Frequency Damping Gain Constant	**73**
Response:	5573 yyyy		
Command:	Cx73 yyyy	Write Low Frequency Damping Gain Constant	
Response:	AA73 yyyy		

x = Axis indicator

yyyy = Gain (0 to 10000)

0 is the default value

Command:	8x74 0000	Read High Frequency Damping Gain Constant	**74**
Response:	5574 yyyy		
Command:	Cx74 yyyy	Write High Frequency Damping Gain Constant	
Response:	AA74 yyyy		

x = Axis indicator

yyyy = High Frequency Damping Gain Constant (0 to 10000)

0 is the default value

Command:	8x75 0000	Read Torque To Inertia Ratio
Response:	5575 yyyy	

Command:	Cx75 yyyy	Write Torque To Inertia Ratio
Response:	AA75 yyyy	

x = Axis indicator

yyyy = Torque To Inertia Ratio (1 to 10000)

1160 is the default value

Command:	8x76 0000	Read Observer Gain Constant
Response:	5576 yyyy	

Command:	Cx76 yyyy	Write Observer Gain Constant
Response:	AA76 yyyy	

x = Axis indicator

yyyy = Observer Gain Constant (0 to 100)

0 is the default value

Command: 8x77 0000 Read Viscous Damping **77**
Response: 5577 00yy

Command: Cx77 00yy Write Viscous Damping
Response: AA77 00yy

x = Axis indicator

yy = Viscous Damping (0 to 100)

1 is the default value

Command: 8x78 0000 **OBSOLETE** (was Read Position Filter Frequency) **78**
Response: 5578 0yyy

Command: Cx78 0yyy **OBSOLETE** (was Write Position Filter Frequency)
Response: AA78 0yyy

x = Axis indicator

yyy = Frequency in Hertz divided by 10 (500 to 5000)

1000 is the default value (which means 10,000 Hz)

Command:	8x79 0000	Read Low Frequency Damping Filter Frequency	**79**
Response:	5579 0yyy		

Command:	Cx79 0yyy	Write Low Frequency Damping Filter Frequency
Response:	AA79 0yyy	

x = Axis indicator

yyy = Frequency in Hertz divided by 10 (10 to 5000, corresponding to 100Hz to 50kHz)

600 is the default value (which means 6,000 Hz)

Command:	8x7A 0000	Read High Frequency Damping Filter Frequency	**7A**
Response:	557A 0yyy		

Command:	Cx7A 0yyy	Write High Frequency Damping Filter Frequency
Response:	AA7A 0yyy	

x = Axis indicator

yyy = Frequency in Hertz divided by 10 (10 to 5000, corresponding to 100Hz to 50kHz)

600 is the default value (which means 6,000 Hz)

Command:	8x7B 0000	Read Output Filter 1 Frequency	**7B**
Response:	557B 0yyy		

Command:	Cx7B 0yyy	Write Output Filter 1 Frequency
Response:	AA7B 0yyy	

x = Axis indicator

yyy = Frequency in Hertz divided by 10 (10 to 6000, corresponding to 100Hz to 60kHz)

2500 is the default value (which means 26,000 Hz)

Command:	8x7C 0000	Read Output Filter 2 Frequency	**7C**
Response:	557C 0yyy		

Command:	Cx7C 0yyy	Write Output Filter 2 Frequency
Response:	AA7C 0yyy	

x = Axis indicator

yyyy = Frequency in Hertz divided by 10 (10 to 6000, corresponding to 100Hz to 60kHz)

3700 is the default value (which means 37,000 Hz)

Command:	8x7D 0000	Read Output Filter 2 Frequency	**7D**
Response:	557D 0yyy		

Command:	Cx7D 0yyy	Write Output Filter 2 Frequency
Response:	AA7D 0yyy	

x = Axis indicator

yyyy = Frequency in Hertz divided by 10 (10 to 6000, corresponding to 100Hz to 60kHz)

550 is the default value (which means 5,500 Hz)

Command: 8x7C 0000 Write Serial Number to Memory and Flash **7E**
Response: 557C yyyy

x = Axis indicator but this is used to control which of two serial number parts gets written

- If x indicates X Axis, then Serial Number Supplement data is written to internal memory
- If x indicates Y Axis, then Serial Number data is written to internal memory, and both Serial Number and Serial Number Supplement data is written to Flash.

yyyy = Serial number (or Serial Number Supplement) you want written into the board

Command:	8x7F 0000	Read Drive Gain Constant	**7F**
Response:	557F 0yyy		

Command:	Cx7F 0yyy	Write Drive Gain Constant
Response:	AA7F 0yyy	

x = Axis indicator

yyy = Final Gain Constant (10 to 1000)

400 is the default value

The following commands provide control of each axis. However, since the command number is 80 or higher, the protocol limits these commands such that they can only communicate 7-bits of data.

Command:	8x81 8000	Read Resolution Enhancement FilterType	**81**
Response:	5581 000y		

Command:	Cx81 800y	Write Resolution Enhancement Filter Type
Response:	AA81 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “0.5 bits”; 2 for “1.0 bits”; 3 for “1.5 bits”; 4 for “1.8 bits”

0 is the default value

Command:	8x82 8000	OBSOLETE (was Read Position Filter Q factor)	**(OBSOLETE)**	**82**
Response:	5582 00yy			

Command:	Cx82 80yy	OBSOLETE (was Write Position Filter Q factor)
Response:	AA82 00yy	

x = Axis indicator

yy = Q factor divided by **100** (50 to 100)

57 is the default value (which means 0.57)

Note: Q factor is only effective when filter type = 2

Command:	8x84 8000	Read Low Frequency Damping Filter Type	**84**
Response:	5584 000y		

Command:	Cx84 800y	Write Low Frequency Damping Filter Type
Response:	AA84 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “single-pole low pass”; 2 for “two-pole low pass”

1 is the default value

Command:	8x85 8000	Read Low Frequency Damping Filter Q factor	**85**
Response:	5585 00yy		

Command:	Cx85 80yy	Write Low Frequency Damping Filter Q factor
Response:	AA85 00yy	

x = Axis indicator

yy = Q factor divided by **100** (50 to 100)

57 is the default value (which means 0.57)

Note: Q factor is only effective when filter type = 2

Command:	8x87 8000	Read High Frequency Damping Filter Type	**87**
Response:	5587 000y		

Command:	Cx87 800y	Write High Frequency Damping Filter Type
Response:	AA87 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “single-pole high pass”; 2 for “two-pole high pass” ←pay attention, filter types

1 is the default value

Command:	8x88 8000	Read High Frequency Damping Filter Q factor	**88**
Response:	5588 00yy		

Command:	Cx88 80yy	Write High Frequency Damping Filter Q factor
Response:	AA88 00yy	

x = Axis indicator

yy = Q factor divided by **100** (50 to 100)

57 is the default value (which means 0.57)

Note: Q factor is only effective when filter type = 2

Command:	8x8A 8000	Read Output Filter 1 Type	**8A**
Response:	558A 000y		

Command:	Cx8A 800y	Write Output Filter 1 Type
Response:	AA8A 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “single-pole low pass”; 2 for “two-pole low pass”; 3 for “notch filter”; 4 for “phase”

0 is the default value

Command:	8x8B 8000	Read Output Filter 1 Q factor	**8B**
Response:	558B 00yy		

Command:	Cx8B 80yy	Write Output Filter 1 Q factor
Response:	AA8B 00yy	

x = Axis indicator

yy = Q factor divided by **10** (5 to 50)

20 is the default value (which means 2.0)

Note: Q factor is only effective when filter type >= 2
Also note that Q factor for the Output Filter is divided by 10, while Q factor for other filters is divided by 100.

Command:	8x8C 8000	Read Power Amp Filter Type	**8C**
Response:	558C 000y		

Command:	Cx8C 800y	Write Power Amp Filter Type
Response:	AA8C 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “single-pole low pass”; 2 for “two-pole low pass”

0 is the default value

Command:	8x8D 8000	Read Output Filter 2 Type	**8D**
Response:	558D 000y		

Command:	Cx8D 800y	Write Output Filter 2 Type
Response:	AA8D 000y	

x = Axis indicator

y = 0 for “bypass”; 1 for “single-pole low pass”; 2 for “two-pole low pass”; 3 for “notch filter”; 4 for “phase”

0 is the default value

Command:	8x8E 8000	Read Output Filter 2 Q factor	**8E**
Response:	558E 00yy		

Command:	Cx8E 80yy	Write Output Filter 2 Q factor
Response:	AA8E 00yy	

x = Axis indicator

yy = Q factor divided by **10** (5 to 50)

20 is the default value (which means 2.0)

Note: Q factor is only effective when filter type >= 2
Also note that Q factor for the Output Filter is divided by 10, while Q factor for other filters is divided by 100.

Command:	8x8F 8000	Read Power Amp Filter Q factor	**8F**
Response:	558F 00yy		

Command:	Cx8F 80yy	Write Power Amp Filter Q factor
Response:	AA8F 00yy	

x = Axis indicator

yy = Q factor divided by **10** (5 to 50)

6 is the default value (which means 0.6)

Note: Q factor is only effective when filter type = 2
Also note that Q factor for the Output Filter is divided by 10, while Q factor for other filters is divided by 100.

Command: 8x90 8000 Read Synchronized Sampling
Response: 5590 000y

Command: Cx90 800y Write Synchronized Sampling
Response: AA90 000y

x = Axis indicator (ignored)

y = 1 for Servo Sampling should be synchronized with the serial data frame rate
0 for Servo Sampling should run asynchronously

0 is the default value

Command: 8x92 8000 Read Drive Polarity **92**
Response: 5592 000y

Command: Cx92 800y Write Drive Polarity
Response: AA92 000y

x = Axis indicator

y = 1 for positive; 0 for negative

0 is the default value

Command:	8x93 8000	Read Centering Pulse Current
Response:	5593 00yy	

Command:	Cx93 80yy	Write Centering Pulse Current
Response:	AA93 00yy	

x = Axis indicator

yy = Centering Pulse Current Tenth Amps (1 to 100)

0 is the default value in version 4 firmware (meaning 0.0 amps)

Comments:

When the Mach DSP starts up, and when a new tuning file is selected or loaded afterwards, the Mach DSP can force the mirror to become centered. This can be helpful for systems where the scanners do not have bumpers on the mirrors, because the “Centering Pulse” will force the mirror to flip into the correct orientation.

Note that for scanners that DO use mirror bumpers, excessive startup currents may cause the mirror to smack up against the mirror bumper of the scanner/mirror orientation is not correct. In a worst-case scenario, this may damage the mirror or the bond between the moving rotor magnet and the output rotor shaft.

Therefore it is a good idea to test the scanner system with various levels of startup currents, and set this feature to the minimum current necessary to reliably flip scanners to the correct orientation.

Command: 8x94 8000 Read Centering Pulse Time **94**
Response: 5594 00yy

Command: Cx94 80yy Write Centering Pulse Time
Response: AA94 00yy

x = Axis indicator

yy = Centering Pulse Time in Milliseconds

10 is the default value (meaning 10 milliseconds)

Command: 8x95 8000 Read TTL Output 1 Polarity **95**
Response: 5595 000y

Command: Cx95 800y Write TTL Output 1 Polarity
Response: AA95 000y

x = Axis indicator, but this is ignored since the TTL Outputs are shared between X and Y axis

y = Polarity: 0= Inactive (no current flowing through opto-isolator); 1= Active (current flowing through opto-isolator).

Command: 8x96 8000 Read TTL Output 1 Condition **96**
Response: 5596 00yy

Command: Cx96 80yy Write TTL Output 1 Condition
Response: AA96 00yy

x = Axis indicator, but this is ignored since the TTL Outputs are shared between X and Y axis

yy = Condition:
0= Never;
1=Always;
2=If X-axis Servo is Ready;
3= If Y-axis Servo is Ready;
4= If both X- and Y-axis Servo are ready;
5= If either X- or Y-axis Servo are ready;
6=If Power Amplifier is enabled;
7=If Servo Board Temperature is greater than qualifier;
8=If X-axis Coil Temperature is greater than qualifier;
9=If Y-axis Coil Temperature is greater than qualifier;
10=If either X- or Y-axis Coil Temperature is greater than qualifier;
11=If X-axis is being exercised;
12=If X-axis is being exercised;
13=If either X- or Y-axis is being exercised;
(Conditions 0 through 13 are only checked around 20 times per second.)

14=If Absolute Value of X-axis Position Error is greater than qualifier;
15=If Absolute Value of Y-axis Position Error is greater than qualifier;
16=If Absolute Value of either X- or Y-axis Position Error is greater than qualifier;
17=If Absolute Value of X-axis Velocity is greater than qualifier;
18=If Absolute Value of Y-axis Velocity is greater than qualifier;
19=If Absolute Value of either X- or Y-axis Velocity is greater than qualifier.
20=If X-axis Position is greater than qualifier;
21=If Y-axis Position is greater than qualifier;
22=If Absolute Value of X-axis Position is greater than qualifier;
23=If Absolute Value of Y-axis Position is greater than qualifier;
24=If Absolute Value of either X- or Y-axis Position is greater than qualifier.
(Conditions 14 and higher are processed in real time.)

Command:	8x98 8000	Read TTL Output 2 Polarity	**98**
Response:	5598 000y		

Command:	Cx98 800y	Write TTL Output 2 Polarity
Response:	AA98 000y	

x = Axis indicator, but this is ignored since the TTL Outputs are shared between X and Y axis

y = Polarity: 0= Inactive (no current flowing through opto-isolator); 1= Active (current flowing through opto-isolator).

Command: 8x99 8000 Read TTL Output 2 Condition **99**
Response: 5599 00yy

Command: Cx99 80yy Write TTL Output 2 Condition
Response: AA99 00yy

x = Axis indicator, but this is ignored since the TTL Outputs are shared between X and Y axis

0= Never;
1=Always;
2=If X-axis Servo is Ready;
3= If Y-axis Servo is Ready;
4= If both X- and Y-axis Servo are ready;
5= If either X- or Y-axis Servo are ready;
6=If Power Amplifier is enabled;
7=If Servo Board Temperature is greater than qualifier;
8=If X-axis Coil Temperature is greater than qualifier;
9=If Y-axis Coil Temperature is greater than qualifier;
10=If either X- or Y-axis Coil Temperature is greater than qualifier;
11=If X-axis is being exercised;
12=If X-axis is being exercised;
13=If either X- or Y-axis is being exercised;
(Conditions 0 through 13 are only checked around 20 times per second.)

14=If Absolute Value of X-axis Position Error is greater than qualifier;
15=If Absolute Value of Y-axis Position Error is greater than qualifier;
16=If Absolute Value of either X- or Y-axis Position Error is greater than qualifier;
17=If Absolute Value of X-axis Velocity is greater than qualifier;
18=If Absolute Value of Y-axis Velocity is greater than qualifier;
19=If Absolute Value of either X- or Y-axis Velocity is greater than qualifier.
20=If X-axis Position is greater than qualifier;
21=If Y-axis Position is greater than qualifier;
22=If Absolute Value of X-axis Position is greater than qualifier;
23=If Absolute Value of Y-axis Position is greater than qualifier;
24=If Absolute Value of either X- or Y-axis Position is greater than qualifier.
(Conditions 14 and higher are processed in real time.)

Command:	8xA0 8000	Read Coil Resistance	**A0**
Response:	55A0 00yy		

Command:	CxA0 80yy	Write Coil Resistance
Response:	AAA0 00yy	

x = Axis indicator

yy = Coil Resistance in Tenth Ohms (1 to 120)

22 is the default value (meaning 2.2 ohms)

Command:	8xA1 8000	Read Thermal Conductivity Coil to Case	**A1**
Response:	55A1 00yy		

Command:	CxA1 80yy	Write Thermal Conductivity Coil to Case
Response:	AAA1 00yy	

x = Axis indicator

yy = Thermal Conductivity in Tenth Degrees C per Watt (1 to 120)

15 is the default value (meaning 1.5 degrees C per watt)

Command:	8xA2 8000	Read Thermal Conductivity Magnet to Case	**A2**
Response:	55A2 00yy		

Command:	CxA2 80yy	Write Thermal Conductivity Magnet to Case
Response:	AAA2 00yy	

x = Axis indicator

yy = Thermal Conductivity in Tenth Degrees C per Watt (1 to 120)

10 is the default value(meaning 1.0 degrees C per watt)

Command:	8xA3 8000	Read MeasureCoilTemperature flag	**A3**
Response:	55A3 000y		

Command:	CxA3 800y	Write MeasureCoilTemperature flag
Response:	AAA3 000y	

x = Axis indicator

yy = 0 for calculate coil temperature by means of heat; 1 for measure coil resistance directly

1 is the default value

Command:	8xA4 8000	Read “OtherThanMotor” resistance	**A4**
Response:	55A4 00yy		

Command:	CxA4 80yy	Write “OtherThanMotor” resistance
Response:	AAA4 00yy	

x = Axis indicator

yy = hundredths of an ohm resistance (10 to 120)

20 is the default value (meaning 0.2 ohms)

Command:	8xA5 8000	Read Nominal Scanner Body Temperature	**A5**
Response:	55A5 00yy		

Command:	CxA5 80yy	Write Nominal Scanner Body Temperature
Response:	AAA5 00yy	

x = Axis indicator but it is ignored because this applies to both axis

yy = degrees C

30 is the default value

Command:	8xB1 8000	Read Error Integrator Anti-Windup Type	**B1**
Response:	55B1 000y		

Command:	CxB1 800y	Write Error Integrator Anti-windup Type
Response:	AAB1 000y	

x = Axis indicator

y = 0 for Position Error Integrator; Anti-windup deactivated
y = 1 for Position Error Integrator; Always limit
y = 2 for Position Error Integrator; Limit when Error and Torque are the same polarity
y = 3 for Position Error Integrator; Limit when Error, Integral and Torque are the same polarity
y = 4 for Position Error Integrator; Limit in an amount proportional to overdrive
y=5 for “Gated Integrator” based on fraction of a degree of Position Error (the fraction specified by command B2)
y=6 for Velocity Error Integrator; Anti-windup deactivated

2 is the default value

Command:	8xB2 8000	Read Error Integrator Anti-Windup Integration Rate	**B2**
Response:	55B2 00yy		

Command:	CxB2 80yy	Write Error Integrator Anti-windup Integration Rate
Response:	AAB2 00yy	

x = Axis indicator

yy = Percentage of Integration Rate during Windup (0 to 100)

70 is the default value

Special Functions

The following commands provide control of the flash memory, and of features stored in flash memory including the actual firmware.

The servo amplifier has four built-in tuning memories which can be activated at any time

Command:	8xF1 8000	Read Current Tuning Memory Number	**F1**
Response:	55F1 0000		

Command:	CxF1 800y	Activate Tuning Memory Number (and store it in Flash)
Response:	AAF1 000y	

x = Axis indicator, but this is ignored since the tuning memory number is shared by X and Y axis

y = 1 through 4

1 is the default value

When “Activating” a tuning memory number, the system first validates that the data in memory is valid, and then selects the new tuning and activates it. If the data in the memory it not valid, then tuning number 1 is selected.

This means after you use Activate Tuning Memory Number, you should always check the returned result. If you try to activate Tuning Memory Number 3, and it returns Tuning Memory Number 1, then you will know an error occurred.

Command: CxF2 8000y Save current settings into tuning flash memory **F2**
Response: AAF2 000y

x = Axis indicator, but this is ignored since the tuning memory number is shared by X and Y axis

y = 1 through 4

Command: CxF3 800y Erase a segment of flash memory **F3**
Response: AAF3 000y

x = Axis indicator, but this is ignored since the tuning memory number is shared by X and Y axis

0 = “Startup Data” including Tuning number, Watchdog, 450MHz, and TTL Input data
1 = Tuning memory number 1
2 = Tuning memory number 2
3 = Tuning memory number 3
4 = Tuning memory number 4

Command:	FFF4 FF7F	Request Firmware Update	**F4**
Response:	FFF4 0000	(if the request did not work)	

This is a special command which sets an indication into Flash Memory that you want to do a firmware upgrade, and then it resets the processor. As such, there will not be a response if this command is successful.

A system boot can take up to 5 seconds. After 5 seconds, when the Flash memory discovers an update has been requested, it goes into a loop, sending out a single byte called “start loading” (currently 0x01). This is an indication from firmware to the receiving program that the firmware file should be sent over the serial port.

In response to the single byte (0x01), the firmware updater program sends four bytes, which correspond to the length of the firmware file. After each byte, the DSP will send “length ack” (0x03) to indicate that each length byte was received. The bytes are sent little-endian format (least significant byte first).

After that, the firmware updater program sends four bytes, which correspond to the address (“offset” within flash memory where the file should be loaded. The boot firmware is loaded into 0x0000 0000. The “fail-safe firmware” (used incase of a firmware update failure) is loaded into 0x0004 0000. The main application firmware is loaded into 0x0007 0000. The firmware updater program sends these four bytes and the DSP acknowledges each byte with a “offset ack” (0x04).

After that, the firmware updater program sends the actual data bytes from the loader file. As the DSP receives these bytes, it acknowledges each one with a “byte received” (0x02) and immediately writes this into Flash memory.

Once the entire file has been received and programmed into Flash memory, the DSP sends out one more “byte received” and then re-boots, using the newly loaded file.

Right now the implementation is essentially just a “proof of concept”. As such, it is pretty crude – using simple bytes like 0x01 for “start loading” and 0x02 for “byte received” and 0x03 for “length ack” and “0x04” for “offset ack”. There is no error checking, and no attempt to obfuscate the contents of the file.

Future versions of the firmware loader will become more robust, for example using a different ack for each length byte, a different ack for each offset byte, perhaps some kind of “rolling ack” (or echo the received data byte with all bits inverted) for actual data bytes, and of course some form of obfuscation/encryption, because we don’t want people discovering that this is an Analog Devices DSP loader file, which they can then disassemble and/or use on their own devices.

One neat thing about this is that this updater concept is capable of modifying literally any part of the flash memory, including the boot sector itself as well as user tunings stored in flash memory. This may be used to implement a “007” mechanism – if the Mach DSP application discovers a serial number of stolen systems or the like. We can force update of the Flash memory, at address zero, and blow away the board’s ability to operate from that point forward, forcing the customer to return the system to Pangolin for repair, and allowing us to recover the stolen merchandise.

Command:	8FF5 FF00	Read status of watchdog timer	**F5**
Response:	55F5 00xx		

Command:	CFF5 FFxx	Write status of watchdog timer (also stored in Flash)
Response:	AAF5 00xx	

xx = Watchdog status indicator; either 00 for “disabled” or 0xAD for “enabled”

Command:	8FF6 FF00	Read Master CPU Clock rate	**F6**
Response:	55F6 00xx		

Command:	CFF6 FFxx	Write status of watchdog timer (also stored in Flash)
Response:	AAF6 00xx	

xx = Processor clock rate; either 0x45 for “450MHz” or 0x40 for “400MHz”