

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

**Artisan Technology Group**
(217) 352-9330 | sales@artisantg.com | artisantg.com

**Full-service, independent repair center**
with experienced engineers and technicians on staff.

**We buy your excess, underutilized, and idle equipment**
along with credit for buybacks and trade-ins.

**Custom engineering**
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

**Expert team | Trust guarantee | 100% satisfaction**

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

**Visit our website - *Click HERE***

# 1. Introduction:

The MAC 2002 Communication Interface No. 73002040 is an interface between a terminal or host computer and the controller. The communication is accomplished with an RS-232 serial connection. Programming protocol is with text of ASCII alpha-numeric characters. Also included are some control characters such as carriage return, backspacing, and tab.

There is a set of built-in commands with unique names. These commands can be executed by simply sending the command name with some parameters if required. The reply received will be in a preset format and may include the result required.

For example: Reading the position of motor X takes the form:

command: Where X

reply: :A 120000

## 1.1. Basic Rules:

Characters sent from a host computer are stored in a buffer and executed after receiving a carriage return. Each line should be started with a command name and terminated with a carriage return character. Each line can only contain one command name, such as CALIB or HALT, but more then one parameter can be appended to a command, by respecting the command format. Each command is executed when a carriage return is received. Commands and parameters should be separated by at least one space or tab character. When a parameter with a number is involved, one or more spaces or tabs are permitted between them.

For example: The following formats are all the same:

where x=123 <cr>
where x =123<cr>
where x = 123 <cr>

The command termination character specified here as <cr> has the value of 13 decimal, 0d hexadecimal and represented in C programming language by "\r" (carriage return).



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 1.2. Reply:

A reply is sent back from controller to the host upon reception of a command. Every reply starts with an ASCII colon character (":") and terminates by a character having value of 10 decimal,0A hexadecimal and represented as "\n" in C programming language. Reception of a reply means controller is ready to accept the next command from the host computer. The replies can be divided to two groups regarded as positive and negative. A positive reply is sent back if there are no errors encountered within the command structure. A positive reply character is an ASCII ("A") following the reply start character. Internally every command received is assigned a reference number which is sent to host with the positive reply. A negative reply is sent back if there are errors within the command structure. A negative reply character is an ASCII ("N") following the reply start character. Sending a command that does not exist, not respecting the command formats or trying to execute commands with no corresponding modules installed, are some of the reasons that will cause negative replies.

## 1.3. Positive Reply Formats:

Positive reply without any other parameters.
**Move X**
**:A**

Positive reply with a value requested.
**Where X Y**
**:A -2000 1000**
In this case inquired X position is -2000 and Y is 1000.

A positive reply may have error code replacing the value requested.
**Where X Y**
**:A -2000 N-2**
Inquired X position is -2000, but Y axis is not installed.

## 1.4. Negative Reply Formats:

An error code is added to the negative replies:
**Xyxter**
**:N -1**
The command Xyxter is unknown to the controller.

**Move X**
**:N -2**
The axis X is not installed.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 1.5. Error Code Table:

-1 Unknown command
-2 Illegal point type or axis, or module not installed
-3 Not enough parameters (eg. move r= )
-4 Parameter out of range
-21 Process aborted by HALT command



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 2. Terminology:

Each module or function of a module is identified with a single alpha character referred to here as **Modul-Id.** Signed integer values can be stored in **Points** for later usage or retrieval. A Point is a basic memory unit which is identified with a number appended to the Modul-Id. Every point for different Modul-Id's is referred to here as **Point-Id**. For example, X3 is a Point-Id for Modul-Id X and Point No. 3. The largest value a point can hold is 4 bytes long. Points may also be used to read and write ports. Certain Point-Id's are used by the interface and should not be used for other purposes. For example, if **CALIB** command uses Point-Id number 99 for calibration speed, Point-Id's X99 and Y99 should only be written when the calibration speed has to be changed.

The following table shows the standard settings for a basic controller. The numbers between [] are optional and can be omitted.

| Modul-Id | Address | Label | Description |
|---|---|---|---|
| X | 1 | EMOT | stage X axis |
| Y | 2 | EMOT | stage Y axis |
| B | 3 | EMOT | aux axis |
| R | 4 | EMOT | aux axis |
| C | 5 | EMOT | aux axis |
| Z | 6 | EMOT | aux axis |
| T | 7 | EMOT | aux axis |
| I | 9 & 8 | EDAIO | digital in ports |
| O | 9 & 8 | EDAIO | digital out ports |
| F | 11 | EAFC | auto focus finder |
| P[1] | 12 | HPHCD | photometer No. 1 |
| P2 | 13 | HPHCD | photometer No. 2 |
| P3 | 14 | HPHCD | photometer No. 3 |
| P4 | 15 | HPHCD | photometer No. 4 |
| P5 | 16 | HPHCD | photometer No. 5 |
| S[1] | 17 | EFILS | filter shutter No. 1 |
| S2 | 18 | EFILS | filter shutter No. 2 |
| S3 | 19 | EFILS | filter shutter No. 3 |
| S4 | 20 | EFILS | filter shutter No. 4 |
| S5 | 21 | EFILS | filter shutter No. 5 |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 3. Remote Reset:

**Command:** **REMRES**

Format: **Remres**

This command is used to reset the whole controller. Upon receiving this command the interface will restart from a power up condition. Which it will reset the other modules. This command accomplishes the same task as if interface reset button is pressed.

Reply: There is no reply sent to host computer. All modules including the interface are reset.

Example: **Remres**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 4. Remote Key Readings:

**Command:** **REMKEY**

Format: **Remkey**

There are four optional switches in the interface. This command is used to read status of this switches. The command will always return a value in a one character string form. The values range from 0 to 4. If there is no switch closure is detected previous to this command the string character "0" is returned. Otherwise string characters from "1" to "4" is returned, representing the pressed switch number. The switch closures are buffered in a 10 byte long FIFO circular buffer. Switches should closed and then opened in order to detect them.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution. Return value is included in the reply.

Example: **Remkey**

:A 0 will mean no switch pressed since last inquiry.
:A 2 switch number 2 is detected.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 5. Interface Remote Control Byte:

**Command:** **ISTAT**
**Istat parameter**

Format: **Istat [parameter]**

This command is implemented to read and write one byte value to interface. The use of this command is for future use. There are no assigned bits at the present time. The user may write any value to interface and read it back. When parameter is omitted the value is returned from interface. On power up default value is 0.

The parameter value is expressed as one byte. It can have values from 0 to 255.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply also means the end of execution. Return value is included in the reply if it exists.

Example: **Istat 200**
write 200 to interface status byte.

**Istat**
read the interface status byte.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 6. Transmission Delay:

**Command:** **TRXDEL**
**Trxdel parameter**

Format: **Trxdel [parameter]**

This command is used to add delays between bytes, transmitted from controller to host computer. When parameter is omitted the value is returned from interface.

The parameter is expressed as one byte and it can have values from 1 to 255. Unit of measurement is expressed as 0.5 millisecond. On power up the default value is 4 which is 2 millisecond delay between transmitted bytes.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution.

Example: **Trxdel 100**
write 100 to interface transmission delay value, which is equal to 50 millisecond.

**trxdel**
read the interface transmission delay value.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 7. Open Photometer Shutters:

**Command:** **OPEN P**

Format: **Open P[device-number] [shutter-number]**
Device-number and shutter-number are equal to 1 when omitted.

Open photometer shutters. Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device- number], there is no space between the board identifier P and the device-number (P2 or P3 etc.). The range of [d] is from 1 to 5. When this number is equal to 1, 1 may be omitted.

Each board has two shutters, which are numbered 1 and 2. The shutters are addressed with the number specified with [shutter-number]. The value of [shutter-number] is either [1] or [2]. When this number is equal to 1, 1 may be omitted. Opening Shutter 1 enables backlight illumination, and Opening Shutter 2 enables light to the photometer head. To take a measurement, normally Shutter 1 should be closed and Shutter 2 should be opened.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution.

Example: **Open P**
Open shutter 1 of P1.

**Open P2**
Open shutter 1 of P2.

**Open P 2**
Open shutter 2 of P1.

**Open P2 2**
Open shutter 2 of P2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 8. Close Photometer Shutter:

**Command:** **CLOSE P**

Format: **Close P[device-number] [shutter-number]**
Device-number and shutter-number are equal to 1 when omitted.

Close photometer shutters. This command has the same format as the Open command (see Open Photometer Shutters). Closing Shutter 1 disables backlight illumination, and closing Shutter 2 disables light to the photometer head. To take a measurement, normally Shutter 1 should be closed and Shutter 2 should be opened.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution.

Example: **Close P**
Close Shutter 1 of P1.

**Close P2**
Close Shutter 1 of P2.

**Close P 2**
Close Shutter 2 of P1.

**Close P2 2**
Close Shutter 2 of P2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 9. Photometer Data:

**Command:** **SIG P**

Format: **Sig P[device-number]**

Read photometer signal channel. The value received is photometer reading and normalized to from -32767 to +32767. Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device- number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply with a value is sent back if a photometer board is installed. A negative reply is sent otherwise.

Example: **Sig P**
Read photometer value of photometer number 1.

**Sig P2**
Read photometer value of photometer number 2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 10. Photometer Reference Data:

**Command:** **REF P**

Format: **Ref P[device-number]**

Read photometer reference channel. The value received is reference reading and normalized to from -32767 to +32767. Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply with a value is sent back if a photometer board is installed. A negative reply is sent otherwise.

Example: **Ref P**
Read reference value of photometer number 1.

**Ref P2**
Read reference value of photometer number 2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 11. Autofocus Data:

**Command:** **SIG F**

Format: **Sig F**

Read auto-focus signal channel. The value returned is focus signal reading.

Reply: A positive reply with a value is sent back if a auto-focus board is installed. A negative reply is sent otherwise.

Example: **Sig F**
Read focus signal value.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 12. Back Light Illumination:

**Command:** **BIL P**

Format: **Bil P[device-number]**
**or**
**Bil P[device-number] value**

Read or write back light illumination duty cycle value. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 100. It represents the percentage of the full illumination value. The value of (0) will turn off back light illumination and (100) will turn on the back light illumination with the full intensity.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The intensity of the back light will change after reception of a command.

Example: **Bil P 50**
Turn the back light illumination to half of its full intensity ( 50% duty cycle).

**Bil P 0**
Turn the back light illumination off.

**Bil P**
Read the back light illumination duty cycle.
The reply will be:
:A 0



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 13. Analog Output:

**Command:** **DAC P**

Format: **Dac P[device-number]**
**or**
**Dac P[device-number] value**

Read or write auxiliary analog output value. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 10 volts. It represents the analog output voltage. This output may be used as an extra DC power supply for lamps, etc.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The output voltage will change after reception of a command.

Example: **Dac P 5**
Make voltage to be 5 volt at the output.

**Dac P**
Read the last output voltage written.
The reply will be:
:A 5



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 14. Binary Analog Output:

**Command:** **BDAC P**

Format: **Bdac P[device-number]**
**or**
**Bdac P[device-number] value**

Read or write auxiliary analog output binary value. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 32767. This command is similar to DAC command described earlier. Internally, precision of the DAC output is 0.000153 Volt, but due to incapability of loading floating numbers as a command parameter, output voltage can only be programmed in increments of 1 volt. In order correct this inconvenience, this command is added. The following equation should be helpful to compute the correct value to be loaded:

value = ( 32767 / 5 ) * output voltage

With the same equation output voltage can be obtained from the value read as follows:

output voltage = ( 5 / 32767 ) * value

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The output voltage will change after reception of a command.

Example: **Bdac P 5**
Make voltage to be 5 volt at the output.

**Bdac P**
Read the last output voltage written.
The reply will be:
:A 5



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 15. Intensity Sampling Rate:

**Command:** **AVER P**

Format: **Aver P[device-number]**
**or**
**Aver P[device-number] value**

Read or write intensity sampling rate. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 255. When a SIG command is executed, photometer device will sample the detector for the number of times specified with the sampling value. The sampling rate is internally fixed to be 1, 2, 4, 8, 16, 32, 64, 128 and 256. Although the sampling rate may be specified from 0 to 255, it is normalized to the nearest power of two. For example, if value is zero, then sampling rate is 256; if it is 9, it would be 16, etc.

The value read with SIG command is the result of the accumulations of the signals divided by the sampling rate.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Aver P 4**
Make photometer take 4 samples when reading signal intensities.

**Aver P**
Read the last number of samples written.
The reply will be:
:A 4



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 16. Photometer Gain Settings:

**Command:** **GAIN P**

Format: **Gain P[device-number]**
**or**
**Gain P[device-number] value**

Read or write Gain settings. When the value is omitted, then reply received will contain the value stored in photometer device. The purpose is to change the gain to amplify the photometer signal. The amplifier gain can only have one of the following set values:

1 10 100 1000 (power up default is 1)

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The output gain will change after reception of a command.

Example: **Gain P 10**
Gain will change to 10 times.

**Gain P**
Read the last output gain written.
The reply will be:
:A 10



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 17. Photometer Data Step:

**Command:** **STEP P**

Format: **Step P[device-number]**
**or**
**Step P[device-number] value**

Read or write Data collection step interval. When the value is omitted, then the reply received will contain the value stored in photometer device. Unit of the value is specified in motor pulses, and the range is from 1 to 65535. When data collection is done with SYNC ON mode, the interval is governed with the number of pulses coming from motor driver. Photometer data is collected when the number of pulses counted is equal to number of pulses programmed with this command. Data collection stops when incoming motor pulses are stopped.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number] , there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Step P 1000**
Data collection is every 1000 pulses.

**Step P**
Read the last programmed step per data collection.
The reply will be:
:A 1000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 18. Photometer Data Time Interval:

**Command:** **INTER P**

Format: **Inter P[device-number]**
**or**
**Inter P[device-number] value**

Read or write Data collection time interval. When the value is omitted, then the reply received will contain the value stored in photometer device. Unit of the value is in milliseconds, and the range is from 1 to 65535. When data collection is done with SYNC OFF mode, spacing is governed by time. Execution of this command stops when total amount of collection is reached to the value programmed by COLQTY P command.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Inter P 100**
Collect data with 100 millisecond intervals.

**Inter P**
Read the last programmed time interval per data collection.
The reply will be:
:A 100



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 19. Photometer Data Amount:

**Command:** **COLQTY P**

Format: **Colqty P[device-number]**
**or**
**Colqty P[device-number] value**

Read or write Data collection amount. When the value is omitted, then the reply received will contain the value stored in photometer device. The range of the value is from 1 to 65535. When data collection is done with SYNC OFF mode, collection of the data is stopped when the amount of data collected reaches the value programmed by this command.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Colqty P 100**
Collect data 100 times.

**Colqty P**
Read the last programmed amount per data collection.
The reply will be:
:A 100



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 20. Photometer Memory Read:

**Command:** **RDMEMO**

Format: **Rdmemo P[device-number] value**

Read internal photometer memory. The range of the value is from 1 to 255. The unit of this value is 256 bytes of memory. After collecting data, contents of the internal photometer memory may be read by the host. This command reads memory in 256 bytes long blocks. The minimum number of bytes which can be read is 256 and maximum is 65280. It should be noted that the reply of this command does not conform to the other read commands. After receiving the command, the controller sends bytes of information in binary format. In other words, each byte received will have a range of 0 to 255. The format of received data is explained in the code table for photometer "HPHCD" module. The amount of bytes received is equal to 256 times the number of blocks loaded with this command.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: The reply will only contain the actual bytes in photometer memory. There are no leading characters (e.g. ":") or no ending characters ( <cr> ). The host computer should count incoming bytes in order to figure the end of reception.

Example: **Rdmemo P 2**
read memory 512 bytes long.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 21. Photometer Control Word:

**Command:** **CW P**

Format: **Cw P[device-number]**
**or**
**Cw P[device-number] value**

Read or write control word of photometer device. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 255. In order to have access to internal control word of a photometer board, this command is used. The value read shows the last status of the control word, and with the same command the control word can be written in order to directly control some functions of the photometer, such as gain, opening or closing, shutters, etc. For the functions where there are no high-level commands available, this is the only way to execute them.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The following table is the bit map of the Control Word:

**7 6 5 4 3 2 1 0 bit numbers**

**| | | | | | 0 0 gain 1**
**| | | | | | 0 1 gain 10**
**| | | | | | 1 0 gain 100**
**| | | | | | 1 1 gain 1000**
**| | | | | 0 shutter 1 close**
**| | | | | 1 shutter 1 open**
**| | | | 0 shutter 2 close**
**| | | | 1 shutter 2 open**
**| | | 0 sync off**
**| | | 1 sync on**
**| | 0 single beam measurement. Only photometer channel is collected.**
**| | 1 dual beam measurement. Both channels are collected.**
**| 0 direction 0 disabled**
**| 1 direction 0 enabled**
**0 direction 1 disabled**
**1 direction 1 enabled**

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Cw P 12**
Clear all bits of Control Word to zero except bits 2 and 3.

**Cw P**
Read the last Control Word written.
The reply will be:
:A 12



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 22. High Voltage Monitoring:

**Command:** **HV P**

Format: **Hv P[device-number]**
**or**
**Hv P[device-number] value**

Read or write High Volt output value. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 1000. It represents the High Voltage output. This output is used as a power supply to photometer head. Internally, this value is limited to 1000 Volt.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The output voltage will change after reception of a command.

Example: **Hv P 500**

Output will be 500 volt at the output.

**Hv P**
Read the last output voltage written.
The reply will be:
:A 500



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 23. High Voltage Output Control:

**Command:** **BHV P**

Format: **Bhv P[device-number]**
**or**
**Bhv P[device-number] value**

Read or write Binary High Volt output value. When the value is omitted, then the reply received will contain the value stored in photometer device. The value's range is from 0 to 32767. This command is similar to HV command described earlier. Internal precision of the High Voltage is 0.030 Volt, but due to the incapability of loading floating numbers as a command parameter, High Voltage can only be programmed in increments of 1 volt. In order to correct this inconvenience, this command is added. The following equation should be helpful in computing the correct value to be loaded:

value = ( 32767 / 1000 ) * output voltage

With the same equation, output voltage can be obtained from the value read as follows:

output voltage = ( 1000 / 32767 ) * value

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line. The output voltage will change after reception of a command.

Example: **Bhv P 16383**

Output will be 500 volt at the high voltage output.

**Bhv P**
Read the last output voltage written.
The reply will be:
:A 16383



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 24. Start Photometer:

**Command:** **START P**

Format: **Start P[device-number]**

Start the photometer collection. When this command is sent the board is put into data collection mode. Actual collection depends on one of the following two modes:

If sync mode is ON, then the interval of the collection is governed by the pulses and direction coming from motor driver. The number of steps are written with the **STEP P** command. The desired direction can be enabled or disabled with writing the **CW** command. At the end of the cycle when the motor drivers are finished, then a **STOP P** command should be sent. A normal working command sequence in this mode should be first programming the photometer parameters, starting the photometer, and starting the motor drivers (**Move command**).

If sync mode is OFF then the interval of the collection will depend on time and number of collection. In this mode the command **INTER P** will write the interval time and **COLQTY P** will write the number of collection. There is no need to send a **STOP P** command at the end of cycle.

Each controller may have 5 photometer boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error.

Example: **Start P**
**Start P2**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 25. Stop Photometer:

**Command:** **STOP**

Format: **Stop P[device-number]**

Stop the photometer collection. When this command is sent, photometer board is taken out of collection mode. This command also depends on one of the following modes, and it may not even be necessary in one mode.

In sync ON mode, this command has to be sent at the end of the motor driver sweep in order to take out the photometer from collection mode.

In sync OFF mode, this command is not necessary unless process has to be interrupted.

Format: **Stop S[device-number]**

Stop the Filter Wheel rotation.

Each controller may have 5 photometer and filter boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier P and the board number (P2 or P3 etc.). The range of [device- number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error.

Example: **Stop P**
**Stop S**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 26. Rotate Filter Wheel:

**Command:** **ROTAT**

Format: **Rotat S[device-number] Wheel-select Filter-select**

Rotate specified filter wheel. This command will be used to rotate the filter wheels. This command takes 3 parameters as explained below:

First parameter is the modul-id which is always an (S) optionally appended with a device-number.

Second parameter is the wheel selection. Every board can support two filter wheel devices. Wheel-select parameter is one of the following characters:

**M** for main filter wheel.
**A** for secondary filter wheel.

Third parameter is one of the filter selections. It may be expressed with one the following characters:

**N** rotate to next filter.
**P** rotate to previous filter.
**H** search for number ONE filter.
**1** rotate to number ONE filter.
**2** rotate to number TWO filter.
**3** rotate to number THREE filter.
**4** rotate to number FOUR filter.
**5** rotate to number FIVE filter.
**6** rotate to number SIX filter.

Each controller may have 5 filter wheel boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier S and the board number (S2 or S3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply does not mean the end of action taken. A (STATUS S) command should be used to find out the end of the rotation.

Example: **Rotat S M H**
Rotate the Main filter wheel to its number ONE filter. This command normally should be used if the filter wheel is misaligned.

**Rotat S A 4**
Rotate the Auxiliary filter wheel to its number FOUR filter.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 27. Open Shutter:

**Command:** **OPEN S**

Format: **Open S[device-number] [shutter-number]**
Device-number and shutter-number are equal to 1 when omitted.

Open filter wheel Shutters. Each controller may have 5 filter wheel boards installed. Where each board is addressed with the number specified with [device- number], there is no space between the board identifier S and the board number (S2 or S3 etc.). The range of [device-number] is from 1 to 5. When this number is equal to 1, 1 may be omitted.

Each board has two filter wheels and three shutters, which are numbered 1 to 3. Normally, Shutter 1 is on the main wheel, Shutter 2 is on the auxiliary wheel, and Shutter 3 is an extra shutter. The shutters are addressed with the number specified with [shutter-number]. The value of [shutter-number] is either [1], [2] or [3]. When this number is equal to 1, 1 may be omitted.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution.

Example: **Open S**
Open shutter 1 of S1.

**Open S2**
Open shutter 1 of S2.

**Open S 2**
Open shutter 2 of S1.

**Open S2 2**
Open shutter 2 of S2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 28. Close Shutter:

**Command:** **CLOSE S**

Close filter wheel shutters. This command closes the shutters. It has the same format as the Open command (see Open Shutter).

Format: **Close S[device-number] [shutter-number]**
Device-number and shutter-number are equal to 1 when omitted.

Reply: A positive reply is sent back when command is received correctly. Reception of the reply means the end of execution.

Example: **Close S**
Close shutter 1 of S1.

**Close S2**
Close shutter 1 of S2.

**Close S 2**
Close shutter 2 of S1.

**Close S2 2**
Close shutter 2 of S2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 29. Shutter Control:

**Command:** **EXPn S**

Format:
**Exp1 S[device-number]**
**or**
**Exp1 S[device-number] value**
and
**Exp2 S[device-number]**
**or**
**Exp2 S[device-number] value**

Do shutter exposure or load the exposure time. The value's range is from 1 to 65535 and is given in units of millisecond. The (n) next to the command can only be 1 or 2, specifying the shutter number. It takes the form EXP1 for shutter 1 and EXP2 for shutter 2. There is no space between EXP and the (n). When the value is omitted in the command, execution of the exposure takes place. The corresponding shutter will open and stay open for the time specified with the value.

Each controller may have 5 filter shutter boards installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier S and the board number (S2 or S3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply does not mean the end of exposure.

Example:
**Exp1 S 1000**
Exposure time of shutter 1 will be 1 second.

**Exp1 S**
Expose shutter 1.

**Exp2 S**
Expose shutter 2.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 30. Front Panel Switching Control:

**Command:** **PANEL**

Format: **Panel S [+] | [-]**
**Panel F [+] | [-]**

This command is used to enable and disable the front panel switches of modules specified. It will disable the front panel if (-) option is used and will enable it if (+) option is used.

The following table shows which modul-ids are used:

| | |
|---|---|
| S | Filter shutter module. |
| F | Focus module. |

Reply: A positive reply is sent if there are no errors.

Example: **Panel F +**
Enable front panel switches of focus module.

**Panel F -**
Disable front panel switches of focus module.

**Panel S +**
Enable front panel switches of filter module.

**Panel S -**
Disable front panel switches of filter module.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 31. Status Byte Read:

**Command:** **RDSTAT**

Format: **Rdstat S[device-number]**
**or**
**Rdstat F**

Read status byte from modules. This command will reply with the status byte of the specified module. Please consult the code tables for individual boards included in this manual. For example, in the case of the filter module, the position of the shutters can be read with this command.

Each controller may have 5 filter wheel and 1 focus controller board installed. Where each board is addressed with the number specified with [device-number], there is no space between the board identifier S and the board number (S2 or S3 etc.). The range of [device-number] is from 1 to 5 and may be omitted when it is equal to 1.

Reply: A positive reply is sent if no error. The reply will contain a value requested if there is no value specified within the command line.

Example: **Rdstat S**
:A 64

**Rdstat F**
:A 120



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 32. Focus Window Control:

**Command:** **WINDOW**

Format: **Window [+]**
**or**
**Window -**

Turn window on the screen on and off. This command is only related to focus drive module. When command is optionally used with (+), the window on the viewing screen is turned ON, and it is turned OFF when it is used with the (-).

Reply: A positive reply is sent if no error.

Example: **Window**
:A
Window is turned on.

**Window -**
:A
Window is turned off.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 33. Focus Frame Delay:

**Command:** **FDELAY**

Format: **Fdelay**
**or**
**Fdelay value**

To read and write the framing delay count. When focusing, a delay can be added before the actual data reading. This is done to stabilize the focusing motor which is moving for every data point. The framing delay amount is expressed as number of Video Frames to wait before taking data. Every Video Frame is approximately 16.6 millisecond. The value's range is from 1 to 250, and it is equal to 1 on power up. By omitting the value previously written framing delay can be read.

Reply: A positive reply is sent if no error.

Example: **Fdelay 100**
:A

**Fdelay**
:A 100



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 34. Focus Scan Speed:

**Command:** **FSPEED**

Format: **Fspeed**
**or**
**Fspeed value**

To read and write the focusing scan speed. The focusing scan speed is the speed with the focus drive moves between collecting data points, which is different than the speed when focus drive is used as a motor driver. The value is expressed in pulse per second. The range is from 84 to 33000 pps. Power-up focus scan speed is 20000 pps. By omitting the value previously written focusing speed can be read.

Reply: A positive reply is sent if no error.

Example: **Fspeed 10000**
:A

**Fspeed**
:A 10000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 35. Focus Data Step:

**Command:** **FSTEP**

Format: **Fstep**
**or**
**Fstep value**

To read and write the focusing data collection amount. Focusing is done by moving the focusing motor for a certain distance and by collecting intensity data coming through the viewing camera. The intensity will change with the changing focusing point. The focus point, therefore, will be where the maximum intensity was. This is accomplished by moving the focus motor for a given number of pulses and reading the intensity levels. This procedure is repeated the number of times given with this command. The distance for a focusing sweep can be calculated with the following:

distance = Fstep value * profile distance

Fstep value is given with this command and profile distance is given with the LDPRDIST which is explained later.

By omitting the value previously written focusing step can be read.

Reply: A positive reply is sent if no error.

Example: **Fstep 100**
:A

**Fstep**
:A 100



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 36. Profile Distance:

### 36.1. Profile Distance Load:

**Command:** **LDPRDIST**

Format: **Ldprdist**

To load the profile distances. When focusing is executed, the module will make a fine scan first. If a valid focus point is not found in this first phase, then it makes a coarse scan. If a valid focus is not found in this second phase then action is aborted. Otherwise another fine scan is executed around the focusing point found in the second coarse scan.

The distance covered by these scans is a product of scan steps and step distances. The number of scan steps was explained with the FSTEP command. With this command the step distances are loaded into focus module. By modifying these values, the distance covered with fine or coarse scan can be adjusted to different environments.

Loading of these distances is done with two phases. The first phase uses the (WRITE Fnn) command to write the distances into CPU memory point-ids. The second phase loads these memories into focusing module. The actual distances in CPU memory do not go into any effect unless loaded with this command.

Since there are two scan phases - fine and coarse - and three focusing modes - high, medium, and low - there are six profile distances to load. Once written into CPU, these are permanent even if the controller is turned off.

The following table shows the names of the ranges and the point-ids used:

| Point-ids | Range |
|---|---|
| F91 | Coarse High |
| F92 | Coarse Medium |
| F93 | Coarse Low |
| F94 | Fine High |
| F95 | Fine Medium |
| F96 | Fine Low |

The (READ) and (WRITE) commands are used to load these values. It is not necessary to rewrite the unmodified values.

Reply: A positive reply is sent if no error.

Example: **Ldprdist**
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 36.2. Profile Distance Read:

**Command:** **RDPRDIST**

Format: **Rdprdist**

To read the profile distances. Use this command when profile distances in the focusing module have to be read. It should be noted that the values read do not come from the CPU memory but directly from focusing module. The six distances are read from the reply message. The order of placement is as follows:

:A coarse-high coarse-medium coarse-low
fine-high fine-medium fine-low

Reply: A positive reply is sent if no error.

Example: **Rdprdist**
The following reply contains the default profile distance.
:A 48 192 700 8 32 128

coarse high distance is 48
coarse medium distance is 192
coarse low distance is 700
fine high distance is 8
fine medium distance is 32
fine low distance is 128



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 37. Focusing:

**Command:** **FOCUS**

Format: **Focus**
**Focus H**
**Focus M**
**Focus L**

To execute the focusing action. With this command, actual focusing takes place. It can be specified alone or with one of the three options. If it is without any options, the front panel setting will take effect in focusing range. Otherwise H, M, and L stand for high, medium, and low ranges. With these options, different ranges of profile distances can be used for different environments.

Reply: A positive reply is sent if no error. The reply does not signal the end of action.

Example: **Focus H**
Focus by selecting the high range distances.
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 38. Focus Intensity Data:

**Command:** **SIG F**

Format: **Sig F**

Read Focus intensity signal. This command will read the signals read from camera. It can be used for test purposes or fine adjustments, etc. A lower value will mean poor focus and higher values will signify clearer focus.

Reply: A positive reply with a value is sent back if a focus board is installed. A negative reply is sent otherwise.

Example: **Sig F**
:A 4000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 39. Move Motor Absolute:

**Command:** **MOVE**

Format: **Move point-id [point-id ][ ...]**
**or**
**Move motorid=position [motorid=position] [ ... ]**

Move one or more motors to an **absolute** position. A point id and/or an absolute number can be specified in the same line. If a Motor-Id is specified which does not exist in the controller, the controller will ignore the command for that specific Motor-Id.

Reply: A positive reply is sent back when the command is received correctly. Reception of the reply does not mean the end of execution, and another command (see Status Request) should be executed to determine the end of execution.

Example: **MOVE r1**

Move motor R to the absolute position stored in Point-Id R1.

**MOVE x=10000 y10**

Move the X motor to the 10000 position and the Y motor to position stored already in Y10. If Y10 contains the value 2500, the Y motor will move to that position. Both motors will move simultaneously.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 40. Motor Move Vector:

**Command:** **VMOVE**

Format: **Vmove point-id [point-id]**
**or**
**Vmove motorid=position [motorid=position]**

Move one or two motors to an **absolute** position. The purpose here is to run the motors so that they will plot a straight line. The speed is not specified for each motor axis but for the plotting. This running speed is programmed by writing to **point-id X97**, which is in pulse per second unit. There is also a starting speed which is programmed with **point-id X96**. It should be noted that the starting speeds and running speeds of the individual axes are internally calculated and they are not updated to their original values at the end.

Reply: A positive reply is sent back when the command is received correctly. Reception of the reply does not mean the end of execution, and another command (see Status Request) should be executed to determine the end of execution.

Example: **Vmove x=100000 y=50000**
The both motors will stop at the same time.

**Vmove x=300000**
In this case, there is one axis. The running speed should be equal to vector speed programmed.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 41. Motor Move Relative:

**Command:** **MOVREL**

Format: **Movrel point-id [point-id ] [ ...]**
**or**
**Movrel motorid=distance [motorid=distance] [ .. ]**

Move one or more motors a distance relative to present position. The format of this command is very similar to the MOVE command. The number of steps the motors will move are specified with distance.

Reply: Same reply format is applied as to MOVE command.

Example: **MOVREL r1 x=100**

Move the motor R the number of steps stored in Point-Id R1. The motor X will move 100 steps. All motors will move simultaneously.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 42. Encoder Motor Move:

**Command:** **MOVEI**

Format: **Movei point-id [point-id ] [ ...]**
**or**
**Movei motorid=distance [motorid=distance] [ .. ]**

Move one or more motors a distance relative to present position. The format of this command is similar to the MOVE command. This command is significant if motors are used with encoders. The purpose here is to move the motors between encoder pulses. Otherwise with other MOVE and MOVREL commands motors will move in increments of encoder pulses. It should be noted that there is no acceleration and deceleration when this command is executed, and the maximum speed is limited to 25000 pulses per second. Although distance specified can be a number 3 bytes long, this command will function more effectively if used for short distances.

Reply: Same reply format is applied as to MOVE command.

Example: **MOVEI r1 x=100**

Move the motor R the number of steps stored in Point-Id R1. The motor X will move 100 steps. All motors will move simultaneously.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 43. Motor Spin:

**Command:** **SPIN**

Format: **Spin motor-id=speed [motor-id=speed] [ ...]**

Spin motor with the speed specified in pulse-per-second units. This command is reloadable while motor is already rotating. Speed is a signed decimal number. If speed is positive, motor is rotated toward ascending numbers, and if it is negative, motor is rotated toward descending numbers. Ramping up or down is governed by the number loaded with the ACCEL command (see Accelleration Control), and it is reloadable while rotating. The motor will spin without stopping. The direction can be changed without first stopping motors. In such case motor is ramp down to starting speed and then reversed direction up to programmed speed with internal firmware.

Reply: If there are no errors, a positive reply is sent back. This reply does not signal the end of rotation. The status of motor can be determined by reading the status byte (see Motor Status Byte Read).

Example:
**Spin R=10000 T=-20000**
:A
or
**Spin x=10000**
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 44. Centering Stage:

**Command:** **CENTER**

Format: **Center motor-id=speed [motor-id=speed ] [ ...]**

The syntax of this command is similar to the SPIN command described previously with an extra function. It will be used to center the motor in the middle of the axis or at the center pulse location if one is provided. To use this command the motor should be on an axis with two switches located at the ends of the axis. If center pulse is not provided motor is located in the center of the two endlimits. If center pulse is provided the motor is located at the center pulse location.

Reply: If there are no errors, a positive reply is sent back. This reply does not signal the end of centering process. The status of motor can be determined by reading the status byte (see Motor Status Byte Read).

Example:
**center R=10000 T=-20000**
:A
or
**center x=10000**
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 45. Motor Position Read:

**Command:** **WHERE**

Format: **Where motor-id [motor-id ] [ ...]**
**or**
**Where point-id [point-id ] [ ...]**

Return to the current position of motors. With the second format, this value can be stored into Point-Id.

Reply: If there are no errors, a positive reply with values is sent back for either format.

Example: **WHERE R T Z**
:A 100 200 300
or
**WHERE RTZ**
:A 100 200 300

**WHERE R1**
:A 1000
The Point-Id R1 will be equal to 1000.

**WHERE RTZ**
If T is not selected correctly or it does not exist, then the reply should be as follows:

:A 1000 N-2 10000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 46. Motor Position Set:

**Command:** **HERE**

Format: **Here motor-id=position [motor-id=position] [ ... ]**

Write the current position of motors. The value specified in position will be written to the specified motor module as the current position.

Reply: If there are no errors, a positive reply is sent back.

Example: **HERE R=1000 t=2 z=0**
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 47. Speed Control:

**Command:** **SPEED**

This command is used to programmed to top speed of acceleration.

The default value on power up is 25,000 hz.

Format: **Speed motor-id [motor-id ] [ ...]**
Read motor speeds for given motor-ids.
or
**Speed motor-id=value [motor-id=value][ ...]**
Write speed to the given motor-ids.

The value is given in pulse-per-second form.

The maximum value for speed is 2764800 pulses per second.
The minimum value for speed is 85 pulses per second.

The maximum value specified is a computed value and depends on motor type. Typical maximum speed for LEP stages is 300,000 to 400,000 pulses per second.

Reply: If there are no errors, a positive reply is sent.

Example: **SPEED R=100000 T=200000 Z=5000**
:A
**SPEED R T Z**
:A 100000 200000 5000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 48.Starting Speed Control:

**Command:** **STSPEED**

This command is similar to SPEED command with the exception of being the bottom speed of acceleration. When motors are moved they will start moving with programmed starting speed, accelerate until programmed top speed is reached.

The default value on power up is 5,000 hz.

Format: **Stspeed motor-id [motor-id ] [ ...]**
Read motor starting speeds for given motor-ids.
or
**Speed motor-id=value [motor-id=value][ ...]**
Write motor top speed to the given motor-ids.

The value is given in pulse-per-second form.

The maximum value for speed is 2764800 pulses per second.
The minimum value for speed is 1000 pulses per second.

It should be noted that maximum value is a computed value. And this value depends on the external factors like load , motor type etc. . It should be programmed with a speed that will not cause motor stalling.

Reply: If there are no errors, a positive reply is sent.

Example: **SPEED R=10000 T=20000 Z=5000**
:A
**SPEED R T Z**
:A 10000 20000 5000



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 49. Accelleration Control:

**Command:** **ACCEL**

Format: **Accel motor-id [motor-id ] [ ...]**
or
**Accel motor-id=value [motor-id=value][ ...]**

This command is similar to Speed command. Acceleration number can have values from 1 to 255, which controls the acceleration curve of the motors. This ramp value is inversely proportional to the ramping time. The smaller the ACCEL value, the shorter the ramp time.

Reply: If there are no errors, a positive reply is sent.

Example: **ACCEL R=100 T=60 Z=10**
:A

**ACCEL R T Z**
:A 100 60 10



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 50. Home Motors:

**Command:** **HOME**

Format: **Home motor-id [motor-id ] [ ...]**

Move the specified motors toward the endlimit switch. The endlimit is reached by running the motor to a large negative position. The motor will rest on the endlimit. The motors stay on the end limit switch. The speed of the motor is the last programmed speed.

Reply: If there are no errors, a positive reply is sent back.

Example: **HOME R T z**
:A
Motor-id R, T, and Z are homed. Reply is sent back when all done.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 51. Joystick Control:

**Command:** **JOYSTICK**

Format: **Joystick motor-id flag [motor-id flag] [...]**

Enables or disables the joy-sticks. Flag can only be (+) to enable joy-sticks ON and (-) to disable joy-sticks OFF.

Reply: If there are no errors, a positive reply is sent back.

Example: **JOYSTICK R+ X- T-**
Enables motor R and disables motors X and T.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 52. Servo Control:

**Command:** **SERVO**

This command is used for motor drivers with encoders installed. It will turn the servo on or off, program the servo activation distance, or return the programmed servo activation distance value.

Servo on condition is true if joystick is turned off. When motor is not running and the motor shaft is moved externally, internal software will try to move the motor back to original position.

The activation distance is the minimum steps which the motor driver has to be moved externally in order for the internal servo to be activated. For example, if the distance is programmed to be 10, then motor should be turned at least 10 encoder steps before the servo system activates.

Format: **SERVO motor-id=value**
Reply : :A
Load the servo activation distance value, in encoder steps.
Example: Servo r=4 t=5

Format: **SERVO motor-id**
Reply : :A value [value .. ]
Read the servo activation distance value.
Example: Servo r t
:A 4 5

Format: **SERVO motor-id +**
Reply : :A
Turn servo activation ON.
Example: Servo r+ t+

Format: **SERVO motor-id -**
Reply : :A
Turn servo activation OFF.
Example: Servo r- t-

Any of the above formats can be mixed freely not exceeding the maximum number of characters a command line can hold (100).

Example: **SERVO R=40 r+ r z = 10 z z+ t -**
**:A 40 10**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 53. Motor Status Byte Read:

**Command:** **RDSTAT**

Format: **Rdstat motor-id**

Read status byte from motor modules. This command will reply with the status byte of the specified module. One motor-id may be specified as a parameter. Refer to motor status byte format explained elsewhere in this manual, low-level binary section **Error! Reference source not found.**.

Reply: A positive reply is sent if no error. The reply will contain a value requested.

Example: **Rdstat X**
:A 64

**Rdstat Y**
:A 120



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 54. Stop Activity:

**Command:** **HALT**

Format: **Halt**

This command will stop all active motors.

Reply: If there are no errors, a positive reply is sent back.

Example: **HALT**
:A



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 55. Read:

The following READ commands read the memory points or digital input/output.
See also Write Point-id command.

### 55.1. Read Point-id:

**Command:** **READ**

Format: **Read point-id [point-id ] [ ...]**

Read current value of Point-Id stored in memory.

Reply: If there are no errors, a positive reply with values is sent back.

Example: **READ R0 T1 Z99**
:A 100 200 300

**READ r0t1z99**
:A 100 200 300



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 55.2. Read Input/Output:

**Command:** **READ**

Format: **Read I**
**Read I n**
**Read o**
**Read o n**

This command will read the input/output ports of EDAIO board. Each controller can have two EDAIO boards. Each board has 8 inputs and 8 output ports. The ports can be read and written as a whole one 2 byte value or individually. The board with higher address is the lower byte, and the board with lower address is the higher byte.

Reply: If there are no errors, a positive reply with values is sent back.

Example: **READ I**
This format reads all the inputs in a value. The value ranges from 0 to 65535. Each EDAIO device can have 8 inputs and 8 outputs. The EDAIO with higher address is the lower byte, and the EDAIO with lower address is the higher byte.

**READ I 0 [I1] [In]**
This format reads one or more input bits. N can have values from 0 to 15. The EDAIO with higher address holds the input bits 0 to 7, and EDAIO with lower address holds the input bits 8 to 15.

**READ o**
This format reads all the output latches. The value read is the value written by the last WRITE O command. See also Write Point-id command.

**READ o0 [on]**
This format reads the output bits. N can have values 0 to 15.

For all digital Read commands, bits corresponding to missing module are replaced with zeros.

See also the Read Point-id command.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 56. Write:

### 56.1. Write Point-id:

**Command:** **WRITE**

Format: **Write point-id=value [point-id=value ] [ ...]**

Write current value of Point-Id stored in memory.

Reply: If there are no errors, a positive reply is sent back.

Example: **WRITE R1=0 T3=221 Z99=333**
:A

### 56.2. Write Input/Output:

**Command:** **WRITE**

Format: **Write o=value**
**Write on=[0]/[1]**

Reply: If there are no errors, a positive reply is sent back.

Example: **WRITE o=255**
This format writes all the outputs. The value ranges from 0 to 65535. Each EDAIO device can have 8 inputs and 8 outputs. EDAIO with higher address is the lower byte, and EDAIO with lower address is the higher byte.

**WRITE on=0 or write on=1**
**write o0=0**
**Write o1=1**
This format writes one or more input bits. N can have values from 0 to 15. This format is used to set or reset the individual output ports of the module. The EDAIO with higher address holds the input bits 0 to 7, and EDAIO with lower address holds the input bits 8 to 15.

For all digital Write commands, bits corresponding to missing module are ignored.
See also Write Point-id command.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 57. Analog Output Read/write(volt):

**Command:** **VOLT**

Format: **Volt I**
**or**
**Volt I=value**

Read or write the analog output of DAIO module. Value is expressed in millivolt and the range is from 0 to 10000. By using this command resolution is 1 millivolt. To use this command a hardware modification is also required to the DAIO module.

Reply: If there are no errors, a positive reply is sent back. If a reading is expected the value is also included in the reply.

Example: **Volt I=1000**
:A
set output voltage to 1000 millivolt or 1 volt.

**Volt I**
:A 1000
last output voltage set is 1000 millivolt.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 58. Analog Output Read/write(number):

**Command:** **BVOLT**

Format: **Bvolt I**
**or**
**Bvolt I=value**

Read or write the analog output of DAIO module. Value is a decimal number ranging from 0 to 65535. The resolution of the output in this special 16 bit mode is 153 microvolt (1volt/65535). Since the resolution of the previous command is only 1 millivolt, this command may be used if the higher resolution is required. The following equations may be used in order convert the units.

To convert the given volt to decimal value:
**value=65535*(Volt/10)**

To convert the value read from the command to volt:
**volt=(value*10)/65535**

Reply: If there are no errors, a positive reply is sent back. If a reading is expected the value is also included in the reply.

Example: **Volt I=100**
:A
set output voltage to 15.millivolts

**Volt I**
:A 100
last output voltage set is 15.3 millivolt.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 59. Center Stage:

**Command:** **CALIB S**

**WARNING!:** **This command cannot be repeated with an empty line.**

It will center the stage and set stage position to zero. The only commands which can be downloaded while calibrating are HALT and WHERE. The other commands will cause the ":N BUSY" message to reply. When the calibration is over the ":A" reply is sent to host signaling the end of operation. The speed of the motors can be programmed by using the command WRITE X99 and Y99. Original speed values are replaced after completion of the command.

Reply: If there are no errors, a positive reply is sent back when the stage is centered.

Example: **CALIB S**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 60. Status Request:

**Command:** **STATUS**

**STATUS device-id**

This command is used to inquire if motor movements are finished or not. When it is used without the device-id parameter reply will report the status of the group of the move commands. When it is used with device-id parameter reply will only report the busy status of the module specified with the device-id. It should be noted that this command takes only ONE device-id as parameter.

Supposing that 3 motors are moving simultaneously, the "STATUS" command will reply with a "N" if all 3 motors are stopped. If any them are still running the reply will be "B".

If any of the single modules status is desired that modules device-id should be used as a parameter.

Format: **Status**

Send motor status. To inquire whether or not any of the motors are active.

Format: **Status F**

Send FOCUS (EAFC) board status. It checks if focusing is done, after one of the focus command is executed.

It should be noted that this command's reply does not follow the common rules. To speed up communication between controllers, the reply is **ONLY ONE CHARACTER WITHOUT** any new line character.

Reply: The positive reply takes two forms:

| | |
|---|---|
| N | if no motors are running. |
| B | if one or more motors are running. |

Example:
```
MOVE X=1000 Y=22000
:A
STATUS
B       any of the X and Y motor is still running
STATUS
N       both X and Y are not running
```



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 65. Current Version Number:

**Command:** **VER**

Format: **Ver**

This command will return the version number of the interface module only.

Reply: Following reply is sent from interface to host:

**Version no. : 6.300**
**:A**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 66. Read Configuration:

**Command:** **RCONFIG**

Format: **Rconfig**

Send configuration report. This command is implemented for monitoring purposes. It will send back a list of motor-id's, labels, and device numbers with descriptions.

Reply: A typical reply is displayed below:

Beginning of reply:

**Configuration Report**

| Dev Address | Label | Id | Description |
|---|---|---|---|
| 1 | EMOT | X | X axis stage MCMSE |
| 9 | EDAIO | I | O digital i/o EDAIO |
| 10 | FFIND | -- | Not supported |
| 18 | EFILS | S2 | Filter Shutter FWSHC |

**:A**

End of reply:

The report shows only the devices installed. If device address does not match the label then id and description will show the correct device label and id with a warning. The devices installed but not supported with high level language are described as Not supported. Such devices may be controlled with low level languages. The labels displayed in the last column can be compared to the labels on the front panel of devices.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 67. POINT-IDS:

Values can be stored in **Points** for later usage or retrieval. A Point is a basic memory unit which is identified with a number appended to the Modul-Id. Each point for different Modul-Id's is referred to here as **Point-Id**. For example, R3 is a Point-Id for Modul-Id R and point No. 3. Some of the point-ids hold values to be used by specific commands, and they are reserved for this purpose. For example, CALIB S command uses point-id 99 to load the calibration speed. If another calibration speed is desired other than default this point-id should be modified.

The range of identification number is from 0 to 99.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## 68. Reserved Motor-ids and Point-ids:

Some of the **Motor-ids** and **Point-ids** are reserved for specific purposes. In order to execute certain commands properly, these reserved motors should be used and their respective point-ids values must be preserved. For example, CALIB S command uses motor-ids X and Y. All prescribed point-ids values for this command should be preserved.

### 68.1. Reserved Motor-ids:

The following motor-ids are reserved for Stage movements:

| | |
|---|---|
| X | Stage movement X axis. |
| Y | Stage movement Y axis. |

The following motor-ids are reserved for various axis:

| | |
|---|---|
| F | Focus finder or stage vertical movement. |
| R | Auxiliary axis. |
| T | Auxiliary axis. |
| Z | Auxiliary axis. |
| B | Auxiliary axis. |
| C | Auxiliary axis. |

### 68.2. Reserved Point-ids:

| | |
|---|---|
| **X99** | X axis calibration speed of stage. |
| **Y99** | Y axis calibration speed of stage. |

Point 99 for the X and Y motor-ids are reserved for calibration speed. Original running speed is replaced with the calibration speed before the command is executed. Original running speed is restored upon completion of the command.

| | |
|---|---|
| **X97** | XY axis vector speed used with VMOVE command. |
| **X96** | XY axis vector starting speed used with VMOVE command. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Following point-ids are reserved for the focus module. Their purpose is to program six scan ranges:

| Point-ids | Range |
|---|---|
| F91 | Coarse High |
| F92 | Coarse Medium |
| F93 | Coarse Low |
| F94 | Fine High |
| F95 | Fine Medium |
| F96 | Fine Low |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

# MAC 2000 COMMUNICATION INTERFACE NO. 73002040

**RS 232 SERIAL INTERFACE**

Section II

## General Description:

The RS 232 Interface Module No. 73002040 is an interface between a host computer and the modules installed in the controller. This module will receive, analyze and execute the commands coming from a host computer. Some of the commands will be used by the interface itself. The communication is realized with an RS 232 connection, with variable baud rates, parity, and stop bits. Currently there are two RS 232 channels available.

Generally a command in Low Level Format consists of a one byte device address, one byte instruction code, one byte data length, one or more bytes of data , and a one byte end of command byte. Each group is later described in detail. Some commands are meant to program the interface itself which do not follow this rules.

The byte length is 8 bits, with either 1- or 2-stop bits. Parity check and parity mode are switch selectable.

The byte values specified in this manual are in decimal form, unless otherwise specified as hexadecimal (as 0xff) or in ASCII string character specified between "" (as "B").

## RS-232 Time-out:

When downloading of a command is not completed within a certain time, that command is discarded by the interface. This time is specified as 10 seconds for High-Level mode and 2 seconds for low-level mode. This time-out will prevent the interface hanging on an unfinished command. The time-out may occur due to errors or dropped bytes in transmission. When the controller's power is on, some bytes may enter the interface caused by turning an external host computer on, which is connected to the controller by RS232 interface. This problem may be avoided most of the time by this time-out feature.

LEP MAC 2000 Controllers are shipped with the following default settings:

Baud rate: 9600
Parity: Disabled
Data bit: 8
Stop bit: 2
Format: Low Level

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## Hardware Switch Description:

Switches are numbered from **1** to **8.**

**Switch No. 8:** Open = Low Level Language (Binary mode )
Closed = High Level Language ( ASCII mode )

**Switch Nos. 1 to 3** set the baud rate.

| Baud Rate | Switch No. 1 | Switch No. 2 | Switch No. 3 |
|---|---|---|---|
| 19200 | open | open | open |
| 9600 | open | open | closed |
| 4800 | closed | open | open |
| 2400 | closed | open | closed |
| 1200 | open | closed | open |
| 600 | open | closed | closed |
| 300 | closed | closed | open |
| 150 | closed | closed | closed |

**Switch Nos. 4** and **5** set parity select and parity check enable.

**Switch No. 4** (parity select):
open = odd parity
closed = even parity

**Switch No. 5** (parity check enable/disable):
open = parity check enabled
closed = parity check disabled (Switch no. 4 is ignored)

REMARK: When parity check disabled 2-stop bits must be used.

**Switch No. 6** add delay between transmitted bytes.

For more details of the function of this switch and how to change the delay by software see Transmission Delay**:** of this section .

| Delay Value | Switch No. 6 |
|---|---|
| No delay | closed |
| 1.5 millisec.(power up) | open |

**Switch No. 7**: Aux channel option:
open = IEEE488 installed
closed = IEEE488 **NOT** installed

If IEEE488 option is installed and selected then switch 1 to 5 becomes device number.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## RS-232 Interface Control Commands:

While the commands described for each module later in this manual control the modules installed in the controller, there are a few other commands which control the RS 232 interface controller itself. Two of these commands, which are used to switch modes, are already explained at the beginning of the manual.

An RS 232 interface control command consists of two or more bytes, always starting with 255 decimal(0xff). These commands directly control the interface, and they do not affect the other modules. They can be sent to the interface regardless of the communication mode the interface is set. It should be noted that these commands can not interrupt a command already being downloaded.

Below is a list of these commands followed by their description:

### Switch to High-Level Mode:

byte 1: 255
byte 2: 65

After sending this control command, the interface will switch to High-Level mode and will accept commands only in a High-Level format.

### Switch to Low-Level Mode:

byte 1: 255
byte 2: 66

After sending this control command, the interface will switch to low-level mode and accept commands only in a low-level format.

### Remote Reset Interface:

byte 1: 255
byte 2: 82

This command will cause the interface to restart, resulting in a hard reset to the other modules. This command can be used to restart the controller if a malfunction is detected.

### Remote Switch Scanning:

byte 1: 255
byte 2: 75

There may be up to four manual switches installed in the controller. This command is used reading which switch is pressed. The interface module will scan these switches and will store the switch values in a first in first out buffer. The interface will respond with one byte to each command. The response is either a '0' (48 dec) meaning no switch closure, or a number from "1" to "4" corresponding to the number of the switch stored in FIFO buffer. The number is expressed in ASCII form.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## **Transmission Delay**:

byte 1: 255
byte 2: 68
byte 3: delay value

This command has an effect if "add transmit delay" switch is open on the interface module. The interface controller transmits available bytes with no delays as soon as transmitter is empty. This may create problems if the host computer cannot read the received byte before the next one arrives. In order to avoid this problem, this command can be used in order to add delays between transmitted bytes from interface controller. On power up and if the delay is enabled by the switch, transmission interval is equal to 2 millisecond.

Apply the following equation in order to convert a delay given in millisecond to the byte to be loaded as the delay value:

byte 3 = delay in millisecond / 0.627

The result should be rounded to a integer having a range from 0 to 255. A value of 0 will also mean no delays.

For example:

If a delay of 5 millisecond is wanted then the byte 3 will be equal to 8.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## RS 232 Serial Command Format:

Commands are generally broken into five groups. In three special cases, the number of data bytes group is omitted to speed up the communication process. These are also explained later in this manual.

For each module existing in a controller, an instruction table is provided, which contains the codes for instructions available and the data length. If the data length is zero, then the command does not contain the data.

Data values are broken into 8-bit bytes for the data length times, and then each byte is sent out through serial channel to the interface, from LSB to MSB.

The ASCII colon (":") character is defined as the end of command code and used to terminate the command loading sequence.

If parity check is enabled, a parity error does not create a syntax error if serial byte received is correct.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## Terminology:

The basic unit of a controller is defined here as a module. For a specific function a different kind of module is designed. A controller is a group of modules for a specific purpose. A controller may perform a function by using its own modules. A controller, for example, can be specified to hold ten modules. Each module may have one or more boards. The function of a module gives a identification to a module. Here we call this module identification as the device-id for short. Every controller can be installed with the same device-ids or different device-ids. Every module has its own selection number referred here as device address. While there may be multiple modules with the same device-ids, none of them may have the same device address.

.

**Group No. 1:**

This is the one byte device address. The device address is set by hardware switches for each module, and they should be exclusive for each module installed in a controller.. Since each module's address selection switch block contains eight switches, this value ranges from 0 to 254 in decimal.

The module address 255 can not be used.

Example:

Supposing the hardware switches for a module are set have the value of four. Then the device address is for that module is four. Any command string intended for this module will have this device address as the group one.

**Group No. 2:**

This group will hold the one byte instruction code. There is a set of instruction codes for each device. These listings explains in more detail their purpose. Each code's value can be found under the Code: header.

Example:

Loading present location counter for a motor driver, code is decimal 65.
Reading the same value back code is decimal 97.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**Group No. 3:**

This group will hold the number of data bytes included with the instruction. While data can be of any amount of bytes , the number of data bytes is a one byte character. This value is also found from tables provided under the 'Data Length:' heading and is fixed for every instruction. The range is 0 to 255 decimal. If the command is a Block Read Command, then this value represent number of blocks to be read, in which case each block is 256 bytes long. If there is no data following the code the value is zero and must be included with the command. There are some exceptions to this rule which is explained later.

Example:

Consider the command of loading the base of a motor driver. Instruction code is decimal (65), and number of data bytes is decimal (3).

| code: | Data Length |
|---|---|
| 65 | 3 |

From what we know so far we can say the following: If device address is four then first byte is four ,second byte is decimal (65) and third byte is (3) for number of bytes following. But there is more to follow.

**Exceptions:**

There are some exceptions where this group is omitted: Request device status, stop and start commands. There are no data bytes following for these commands and the data length byte is always zero and to speed up communications they are omitted.

Example:

All numbers are expressed in decimal.

| Bytes Sent | Action |
|---|---|
| 0,63,58 | request status for device # 0 |
| 15,63,58 | request status for device # 15 |
| 0,71,58 | start module # 0 |
| 10,66,58 | stop module # 10 |

These special command formats are explained with details later in the manual.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**Group No. 4:**

In this group the data value is gathered together. The length of data is not fixed and can be from 0 to 255. The data length expressed within the code table will govern the number of bytes which data is sent and received. Generally data is transferred starting from Least Significant Byte(LSB) to Most Significant Byte(MSB). The actual conversion can be done in several ways. In C programming, for example, if the actual data is a long number , it can be put into a long member of a union and then can be sent by using the character member of the same union.

If the data length is zero then there is no data value load.

Example:

Assuming that number to be loaded is decimal 123456. The following table shows the data bytes for different data lengths. This number fits into at least three bytes. If a number lower than three is used for data length the MSBs are lost.

| Data Length | Bytes to send | |
|---|---|---|
| 1 | 64 | (2 MSBs are dropped) |
| 2 | 226,64 | (1 MSB is dropped) |
| 3 | 1,226,64 | |
| 4 | 0,1,226,64 | |
| 5 | 0,0,1,226,64 | |

**Group No. 5:**

This is the one byte end of command character. It is a fix predetermined character and set to be decimal (58) value. This byte will end the command. Until this byte is received the interface will not execute the command but will keep the bytes in a buffer.

**Reading Values from Modules:**

The command format for reading values from controller is similar to the loading values format. The same rules applies to reading values with exception of the data direction. The data itself comes from controller

The data will start flowing from interface right after the data length is received by the interface. The number of characters sent is equal to the data length specified with code tables.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

## Special Command Formats:

As mentioned earlier there are some commands which do not follow the rules explained. For these commands the data length is assumed always zero. To speed up communications the data length for these commands are dropped. It should be noted that even the data length is zero there might be data coming from controller.

Request Device Status:

This command consists only of device number, instruction code, and end of command code to be sent to the interface. The interface will always will respond with one byte of data.

Instruction code for this command is decimal (63).

Upon receiving the command, interface will send one of the following bytes, depending on whether device is busy or not.

98("b") = if device is not busy
66("B") = if device is busy

Example:

The following bytes are sent to interface:

2 ,63, 58

Meaning:

Check whether or not Device No. 2 is busy. If the value of the byte received is 66, then device is busy with executing a command. It should be noted that if a device which is addressed does not exist the response to this command will always be the device busy code(66). If the value of the byte received is (98), then device is not busy.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

### Start Function:

This command consists only of device number, instruction code, and end of command code. There is no reply coming from interface. Depending on the module this
command will start different functions. For example, it may start a motor driver or start a wafer alignment.

Instruction code for this command is decimal (71).

### Stop Function:

This command consist only of device number, instruction code, and end of command code. There is no reply coming from interface.

Instruction code for this command is (66).

This command is sent when a specific module is required to stop execution of a function. Since every module has a different function to execute, means of stopping the function is also different. For example, this command may stop the stepping motors of a motor driver module if it were already running.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| | | This card is intended to provide the functionality of the previous 73000500 and '503 card set. The differences are as follows:<br>Only one card slot is needed per axis.<br>Trackball performance is improved.<br>Motor resolution is selectable either by dip switch or by software.<br>Motor resolution may be read back by host. See load motor resolution, read motor resolution (on page 16).<br>Configuration dip switch, previously 6 switches, is now 8 switches.<br>Motor current, previously an 8 switch dip, is now a 4 switch dip. See Configuration section. |

## Motion Control:

| | | |
|---|---|---|
| 71 | 0 | **Start motor motions**. It will start motor to reach target position if motor is not running already. It will send busy code while moving the motor if status is requested. |
| 66 | 0 | **Stop motor motions**. If motor is running, it will come to a stop after ramping down. It will respond with a not busy code after a complete stop if status is requested. Power up default. |
| 60 | 0 | **Motor power on**. Motor power should be turned on before running motors. Power up default. |
| 61 | 0 | **Motor power off**. To turn the power off to the motor phases. Motor will not run if motor power is turned off with this command. |
| 39 | 0 | **Goto endlimit**. Motor is activated until an end limit is reached. Motor is positioned at the endlimit. Direction of search is fixed and depends on the reverse direction switch. |

## Position Parameters:

| | | |
|---|---|---|
| 65 | 3 | **Write motor position counter**. This is a two's complement 3 byte number. This parameter is loaded only once on power-up and reloaded again if adjustments are needed. |
| 108 | 4 | **Read motor position counter with status byte**. The first three bytes is the two's complement motor position counter. And the next byte is the status byte. The counter is updated with every movement of the motor. See "Error! Reference source not found." explained later for the format of the status byte. |
| 97 | 3 | **Read motor position counter**. This is the two's complement motor position counter. The counter is updated with every movement of the motor. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| 84 | 3 | **Write target position counter**. This is a two's complement 3 byte number. This parameter is loaded every time motor should be moved to different position. |
| 116 | 3 | **Read target position counter**. This is a two's complement 3 byte number for the target counter. |

## Incremental Movements:

| | | |
|---|---|---|
| 43 | 0 | **Incremental move upwards**. This command will add the increment buffer to present position buffer and load this position to target and then move the motors. It is used when motor movement distances are same. It combines the loading new target and move commands in a single byte command. |
| 45 | 0 | **Incremental move downwards**. Same command as above except that direction is reversed. |
| 68 | 3 | **Write Increment value**. This is a two's complement 3 byte number for the increment counter. Use this value to program the distance for incremental moves. |
| 100 | 3 | **Read Increment value**. This is a two's complement 3 byte number for the increment counter. |
| 46 | 3 | **Incremental up and down moves between encoder pulses**. This command is only significant if motor is used with the encoder option. The motor will move the amount specified by the parameter. In this case, the amount is expressed in motor steps, not encoder pulses. The parameter specified is in two's complement 3 byte number. The motor may be moved forward or backward, depending on the sign of the parameter. The movement is relative to the present position. This command is similar to the incremental moves described earlier with a few differences. First, increment value is included within the command. Then the motor is started and stopped without any acceleration or deceleration. This command also uses the programmed run speed with a lesser maximum speed. |

The maximum speed = 25000 pulse per second. The minimum speed = 84 pulse per second.

The most important difference is that this command will move the motor the same amount of steps regardless of whether or not encoders are installed. When used with an encoder installed, the motor can be positioned between encoder pulses regardless of the resolution of the encoders.

The position read after using this command will not change until the motor is moved to the full encoder step.

For example:



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| Code: | Length: | Description: |
|---|---|---|

Suppose we have an encoder with five (5) motor steps per one (1) encoder pulse. If you want to move the motor only three (3) motor steps forward, use this command with the parameter being equal to three (3).

## Variable Speed Rotating:

| | | |
|---|---|---|
| 47 | 3 | **Write rotating speed to motor driver**. This command will rotate the motors without specifying any other target or speed. This number is a 3 byte signed number which specifies the direction and the speed of the rotation. When this number is positive motor will rotate towards bigger numbers and when it is negative will rotate towards smaller numbers. This command is reloadable. If the motor is already rotating it will ramp up or down depending the last speed loaded. There is no target number to stop the rotation of the motor. The ramp value may also be reloaded to reflect the new ramping time. While motor is rotating run flag is set in the motor status byte. |

Motor will stop if one of the following event is occurred:
1. A stop command is received.
2. An end limit switch is closed.
3. The same command is received with the speed equal to zero.

The following equation should be used in order to convert speed given in pulse per second unit to a binary value to be loaded with this command:

output value= 8,388,608-(5,529,600/speed)

The speed is given in pulse per second and it is positive or negative depending the direction of the rotation.

The programmable speed limits are:
minimum speed = 0.659 pulse per second.
maximum speed = 5,529,600 pulse per second.

It should be noted that the maximum speed specified above is the internal calculation limit and does not necessarily means the running speed of the motor installed in the system. It is left to the user to limit the actual maximum running speed.

There is no reading back command the last speed written with this command at the present time.

## Centering The Axis:

| | | |
|---|---|---|
| 48 | 3 | **Center or search for home pulse command**. This command's parameter value is similar to the "Variable Speed Rotating" command explained earlier, but command's function is different. |

This command may be used for two different purposes. First one is to locate the motor to a specific position. And the second one is to center the motor on the axis between two endlimit switches.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|

The motor will travel between two endlimit switches. While traveling a positive going edge of a pulse will stop the motor where the pulse was received. This pulse should be at least 5 microsecond of duration. In case this pulse is not detected or it is not present at all, then motor is positioned in the center of the axis. The length of the axis is defined with the endlimit switch positions.

The travel speed between endlimits is programmed with the command parameter but the final centering speed is the programmed running speed.

| | | |
|---|---|---|
| 30 | 0 | **Enable position counter capture**. This command will enable reception of the pulse which normally used as the home pulse. This command should be used before the center command if centering is desired with the pulse. If pulse is disabled centering is done by moving to the center of axis by detecting the two end limit switches. |
| 31 | 0 | **Disable position counter capture**. Power up default state. Recommended use of this command will be after centering an axis. |

**Load Center Position:**

| | | |
|---|---|---|
| 91 | 3 | **Load the capture center position**. |

**Read Center Position:**

| | | |
|---|---|---|
| 123 | 3 | **Read the captured center position.** |

The last two commands may be used with the centering command explained earlier(see "Centering The Axis"). If a high precision positioning required a stepping motor may be installed with a center pulse generator. When motor is passed through the position a center pulse is generated and the position is recorded by the module. These commands may be used to determine if centering of an axis is done by centering pulse or by internal calculation. First load the center position with a large number presenting over the limits. Then center the axis. At the end of centering use the Read Center Position command. Compare the numbers loaded and read. If they are the same then the centering pulse is not received. If they are different then the center pulse is received properly.

Centering pulse also may be used as to move the motor until an event occurs. If a center command is used then motor will stop when a center pulse is generated. By reading the motor position, the location of the event can be determined.

**Ramping Time:**

| | | |
|---|---|---|
| 81 | 1 | **Write ramping time.** This is a one byte straight binary number, which will change the acceleration value. It is related to the time where speed is changed. Smaller values are equal to bigger accelerations. Power up default is 20. |
| 113 | 1 | **Read ramping time**. Same as above to read back the value. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|

## Motor Speed Write Commands:

| | | |
|---|---|---|
| 82 | 2 | **Write motor starting speed**. This is a 2 byte straight binary number. This is the speed the motor will start ramp up. If this value is bigger than top speed, then there is no ramping up. The equation below can be used to convert the speed given in pulse per second. |
| 83 | 2 | **Write motor top speed**. This is a 2 byte straight binary number. When motor is moved, this will be the running speed after ramping up. The equation below can be used to convert the speed given in pulse per second. |

The following equation can be used to convert a given speed in pulse per second units to a binary number. This is the number which is used with write starting speed and write top speed commands.

binary output value = 65536-(5529600/speed)

Speed is the desired speed in pulse per second units.

The maximum binary output value = 65534
The minimum binary output value = 1

The maximum speed = 2764800 pulse per second.
The minimum speed = 84.375 pulse per second.

## Motor Speed Read Commands:

| | | |
|---|---|---|
| 114 | 2 | **Read motor starting speed**. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |
| 115 | 2 | **Read motor top speed**. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |

The following equation can be used to convert the binary number read back to speed in pulse per second units. speed(pulse/sec) = 5529600/(65536-binary value)

## Joystick Speed Write Commands:

| | | |
|---|---|---|
| 69 | 1 | **Write joystick normal top speed.** This is a one byte straight binary number. This will affect the joy- stick top speed when it is deflected to either side without pressing the fast joystick key. |
| 70 | 1 | **Write joystick faster top speed**. This has the same function as above except that it will affect the maximum speed when fast key is pressed with joy-stick deflected either side. How to calculate these two numbers is explained at the end of this manual. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|

The following equation can be used to convert a given speed in pulse per second units to a binary number. This is the one byte number which is used with write joystick speeds.

binary output value = 1843000/speed

Speed is the desired speed in pulse per second units.

The maximum binary output value = 255
The minimum binary output value = 1

**Joystick Speed Read Commands:**

| | | |
|---|---|---|
| 101 | 1 | **Read joystick normal top speed**. This is a one byte straight binary number. |
| 102 | 1 | **Read joystick faster top speed**. This is a one byte straight binary number. |

The following equation can be used to convert the binary number read back, to speed in pulse per second units. speed(pulse/sec) = 1843000/binary value

**Joystick Control:**

| | | |
|---|---|---|
| 74 | 0 | **Joystick enabled**. To enable the joystick or trackball function when motor is not moving. Power up default. |
| 75 | 0 | **Joystick disabled**. To disable the joystick or trackball function when motor is not moving. |
| 72 | 0 | **Make joystick fast speed to ramp up**. This command will affect how the joystick fast key will act when pressed with joystick deflected either side. With this command when joystick fast key is pressed, motor will ramp up to faster then joystick speed. Power up default. |
| 76 | 0 | **Make joystick fast speed to ramp down**. This is the opposite of the above command. When joystick fast key is pressed, motor will ramp down to a slower speed than joystick speed. |
| 122 | 1 | **Read joystick deflection values**. The value read will be the deflection values read from analog input port for joystick. The value is ranged from 0 to 255. Mid position has the value of around 128. |

**Open and closed loop constants**:

For the closed loop operation and double counting mode, there are two counters installed. One of these counters is defined as open loop counter and the other is defined as the closed loop counter. The purpose of the open loop counter, which is defined in this manual as the first counter, is to count the pulses generated by the motor driver in order to move the stepping motor. The purpose of the closed loop counter, which is defined in this manual as the second counter, is to count the pulses returned from an encoder attached



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| Code: | Length: | Description: |
|---|---|---|

mechanically to the axis of the stepping motor. The relation between the two counters is defined below with an equation.

Counter Ratio = open loop constant / closed loop constant

For the proper operation of motor movements this ratio has to be known internally. This is accomplished by writing the two constant values on the right side of the equation. The two constants being only one byte long have the range of from 1 to 255. Open loop constant may be loaded with a value of zero in order to activate the automatic calculation of both constants. In case the automatic calculation is not satisfactory and the constants are unknown, they can be determined by a simple procedure. Before starting the procedure both counters should be cleared. Then motor should be moved for some amount of steps. The larger amount of steps will result in the better precision of calculation. This may be accomplished easily with a joystick if it is installed. Then both open and closed loop counters should be read. Each of the two numbers should be divided by a constant number until they are both smaller then maximum valid number of 255. The divisions should result with a remainder of zero.

For example: Clear both counters by loading zeros. Pulse the motor 1000 steps. Read the open loop and closed loop counters. Supposing open loop count is 1000 and closed loop count is 200, open loop count should be divided until it is smaller then 255. If both numbers are divided by 100 the result will be 10 and 2. These two values can be used as the two constants or each further be divided by 2 to obtain 5 for open loop and 1 for closed loop constant. This numbers in this example means that encoder pulses are 5 times smaller then motor stepping pulses.

**Load Open Loop Constant:**

| | | |
|---|---|---|
| 92 | 1 | An unsigned one byte binary number. This number represents the open loop constant explained above. The valid range of open loop constant is from 1 to 255. On power up default value is 5. In a special case when loaded number is equal to zero an internal routine is executed to calculate both constants. However the precision of the calculation is limited and the result may not be correct. For this reason when motor is no longer busy, both closed and open loop encoder constants should be examined and corrected by the host system. |

**Read Open Loop Constant:**

| | | |
|---|---|---|
| 124 | 1 | This command will read the open loop counter constant programmed. |

**Load Closed Loop Constant:**

| | | |
|---|---|---|
| 32 | 1 | An unsigned one byte binary number. This number represents the closed loop constant explained above. The valid range of values for this number is from 1 to 255. On power up default value is 1. |

**Read Closed Loop Constant:**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| 131 | 1 | This command will read the closed loop counter ratio programmed.<br><br>An example:<br>In the case of power up default values used, open loop counter readings will be 5 times of the closed loop counter reading. If motor is pulsed 1000 steps, closed loop counter reading should be 200. |

## Settling time:

In the closed loop system when double counting mode is selected, at the end of a motor run closed loop counts are compared internally to open loop counts. Since closed loop counts are real time counts they will reflect any movements implied externally. For example motor axis may be vibrating due to sharp deceleration, when the internal calculation takes place. Which in turn may affect the comparison and furthermore add delays to reach to the right target. A settling time may be programmed externally. This time will be used as a delay between when motor is actually stopped and the calculation takes place. This delay is expressed in units of millisecond.

| | | |
|---|---|---|
| 95 | 1 | **Write settling time value**. This one byte unsigned value may range from 0 to 255. The following actions are taken depending on the value.<br><br>0 = No settling time. Power up default.<br>1 to 254 = Settling time is expressed as the value.<br>255 = Automatic settling time. Settling time will vary according the severity of the vibrations. When a certain lowest variation of closed loop readings is reached settling delay is terminated. |
| 130 | 1 | **Read settling time value.** |

## Servo Control:

| | | |
|---|---|---|
| 77 | 0 | **Turn servo checking on.** If the encoder mode is installed with this command, motor movements are checked against encoder increments. If no movement is detected, like stalling motor, driver will stop and ramp up again until it reaches the target. This mode with joystick disabled will track the external motor movements through encoder and will check against last position counter. If they are not the same, motor is moved back to last position stopped. Power up default up to and including version 2.0. |
| 78 | 0 | **Turn servo checking off**. This command cancels the above one. Power up default starting from version 2.1. |
| 79 | 3 | **Load servo activation distance**. Programmed in units of encoder steps, this is the minimum number of steps the motor should skip before internal software will start moving it back to the original position. Default value is 2 and implemented starting from version 2.1 and up. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| 111 | 3 | **Read servo activation distance**. |

**Servo Speed Write Command:**

| | | |
|---|---|---|
| 93 | 2 | **Write motor servo speed.** The servo movements are done with the speed programmed with this command. The speed equation is the same as explained with motor speeds. The minimum and maximum speeds are different then the motor speeds. Default value on power up set to 300 pulse per second. |

The same equation to calculate the starting and top speed can be used for this command. If speed loaded is outside the range the correct speed limit is replaced by internal software.

binary output value = 65536-(5529600/speed)

Speed is expressed in pulse per second units for the stepping motor.

The maximum speed = 1250 pulse per second.
The minimum speed = 100 pulse per second.

**Servo Speed Read Command:**

| | | |
|---|---|---|
| 129 | 2 | **Incremental move downwards** The returned value is an binary unsigned integer representing a speed factor used internally. |

The following equation can be used to convert the binary number back to speed in pulse per second units. speed = 5529600/(65536-binary value)

**Read Status Byte :**

| | | |
|---|---|---|
| 126 | 1 | **Read status byte from motor driver**. This value will reflect various internal status of motor driver. The format is explained below. |

| bit | description |
|---|---|
| 0 | 0=motor not running.<br>1=motor is running. |
| 1 | 0=servo is OFF.<br>1=servo is ON. |
| 2 | 0=motor phases are turned OFF.<br>1=motor phases are turned ON. |
| 3 | 0=joystick is turned OFF.<br>1=joystick is turned ON. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| | | 4 0=motor is not ramping.<br>1=motor is ramping up or down. |
| | | 5 0=ramping down.<br>1=ramping up. |
| | | 6* this bit is the soft copy of the CW end limit switch. |
| | | 7* this bit is the soft copy of the CCW end limit switch. |

* Note: *changed 10/23/98 to* reverse bits 6 and 7. The corrected values are now shown.

## Read Second Status Byte :

| | | |
|---|---|---|
| 128 | 1 | **Read status byte number two**. This value will reflect various internal status of motor driver. The format is explained below. |

| bit | description |
|---|---|
| 0 | 0=motor is not stalled.<br>1=motor is stalled. |
| 1 | 0=motor is NOT servo aligning when at idle.<br>1= motor is servo aligning when at idle . |
| 2 | 0=NO EEPROM is installed.<br>1= EEPROM is installed. |

## Read Move Status Byte :

| | | |
|---|---|---|
| 136 | 1 | **Read move status byte.** This value will reflect various possible move quality and terminations of all moves, (Commands 71,43,45,46,47,29,39,48). This register is cleared to 0x00 on a call to move the motor. This register can be read at any time. Bits 1-7 are valid after Bit 0/7 is Set. The format is explained below. |

| bit | description |
|---|---|
| Bit 0/7 | System Busy bit. Set when the move is completed. When this bit is set, this register will indicate the status of the last move made.<br>Cleared (0) = Busy, Set (1) = Move Done, Not Busy. |
| Bit 1/7 | Set when the move has been completed normally. Normally means that the move completed by what the command intended to do. For example: if command is goto endlimit and endlimit is hit that is considered as completed normally and this bit is set. Since command is sent with intention of finding endlimit. If move to target is the command and endlimit is hit then this bit is NOT set because it is not a normal completion. |
| Bit 2/7 | Set if the completed move position is less than the target position +/- the servo activation distance (#79). This bit is updated in both servo mode and non-servo mode. This bit is not used with the rotate command 47 |
| Bit 3/7 | Set if the move was terminated by user stop command (#66). |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|

Bit 4/7 Set if a stall. Valid if in encoder mode. In non encoder mode never set to 1.

Bit 5/7 Set on invalid parameter, system busy, or motor power off.
An invalid parameter is a parameter that is out of range. ei Acceleration of zero.

Bit 6/7 Set if CW limit terminated the move.
See Note below.

Bit 7/7 Set if CCW limit terminated the move.
Note: If the Capture Home is enabled (#30) and the Home limit is found both Bits 6/7 and 7/7 will be set.

## Read Identification:

105 6 Read identification from device. First 5 characters read are the device ID EMOT. The sixth character is a binary value which reflects the state of the configuration switches according to the following table:

| Msb | | | | | | | Lsb |
|---|---|---|---|---|---|---|---|
| bit7 | bit 6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| sw8 | sw7 | X | sw6 | sw5 | sw4 | X | X |

The same code may be used consecutively in order to get a version information from device. When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

## Read Version No.:

127 6 **Read date and version**. This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented.

Request Device Status:



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| 63 | 0 | **Request Status**. Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.<br><br>66 = If device is busy.<br>98 = If device is not busy.<br><br>Parameters should only be loaded when a device is not busy, but they can be read anytime. |

**Load Motor Resolution:**

| | | |
|---|---|---|
| 40 | 1 | **Load motor resolution.** Configuration dip switches 1-3 must be in the open position to use this code. Default at power up is 10,000 steps/revolution. Resolution is selected according to following table: |

| Data | Steps/ Revolution (1.8deg. motor) | SW3 | SW2 | SW1 |
|---|---|---|---|---|
| 0 | 40,000 | 0 | 0 | 0 |
| 1 | 20,000 | 0 | 0 | 1 |
| 2 | 10,000 | 0 | 1 | 0 |
| 3 | 5,000 | 0 | 1 | 1 |
| 4 | 2,000 | 1 | 0 | 0 |
| 5 | 1,000 | 1 | 0 | 1 |
| 6 | 500 | 1 | 1 | 0 |
| 7 | Soft Select | 1 | 1 | 1 |

| | | |
|---|---|---|
| 160 | 1 | **Read motor resolution**. The current motor resolution is read back. If the resolution dip switches are used to hard set the resolution, this code will read back the switch value. |

**Soft Limits** are added to supplement the hard end limit stop. As of version 8.2,

The soft and hard limits are logically ANDed. If any of them is true then motor is stopped. Soft limits can be activated and deactivated externally. The motor will not move if present position is smaller then low limit and if present position is bigger then high limit.
There are three new codes are added regarding these functions.

| | | |
|---|---|---|
| 16 | 1 | **Activate or deactivate soft limit detection**. The parameter has the following values:<br>parameter = 0 deactivate soft limit detection. POWER UP DEFAULT.<br>parameter <>0 activate soft limit detection. The programmed soft limits will be used as low high limits. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: |
|---|---|---|
| 17 | 6 | Write soft limit values. First three byte is one limit and second three byte is the other limit. Internally soft limits are sorted and smaller number becomes the low limit and bigger number becomes the high limit. Each 3 byte group has the following order:<br>Byte 0:LSB of the limit 1<br>Byte 1:mid byte of the limit 1<br>Byte 2:MSB of the limit 1<br><br>Byte 3:LSB of the limit 2<br>Byte 4:mid byte of the limit 2<br>Byte 5:MSB of the limit 2<br><br>Byte 0 is the byte following the data length byte in the command stream. |
| 210 | 6 | Read soft limit values. First three byte is the high limit and second three byte is the low limit. Each 3 byte group has the following order:<br>Byte 0:LSB of the limit high<br>Byte 1:mid byte of the limit high<br>Byte 2:MSB of the limit high<br><br>Byte 3:LSB of the limit low<br>Byte 4:mid byte of the limit low<br>Byte 5:MSB of the limit low<br><br>Byte 0 is the first byte received in the data stream. |

ON power up both soft limits are set to zeros.
Home does not function with soft limits enabled. Unless capture home is enabled and home input is detected.

## Configuration:

There three (3) DIP switches installed on the board. S2, with eight (8) switches is used for device address selection, S1, with eight (8) switches is used to set device function. and S3, with four (4) switches sets motor drive current.

Each device can be configured to have addresses from 0 to 254. This can be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 |
|---|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: MAC2002 Single Card Microstep controller/driver
Device ID: "**EMOT**"

| Code: | Length: | Description: | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**Function Switch Settings:**

**Switch 1-3:** Motor resolution select. Motor counts/revolution is set according to table in load motor resolution section. SW1-3 open allows resolution selection by computer. This is the factory setting and at power-up, 10K ppr is default.

**Switch 4:** This switch reverses the joystick direction in relation to the motor direction. Factory setting= closed

**Switch 5:** The end limit detection level.
0 = end limit should switch to ground to stop the motor. factory setting
1 = end limit should switch to high to stop the motor.

**Switch 6:** This switch controls the motor direction relative to the position counter. If a different direction is desired when counter is counting up or down, this switch should be reversed. factory setting= closed

**Switch 7:** Closed=Bi-level drive (factory setting)
Open= Boost voltage always on. For use with high impedance motors.

**Switch 8:**
0 = Non-encoder mode or open loop counting. The position is counted with motor pulses and motor direction. Encoder pulses are not used for position counting even if one is installed.
1 = Encoder mode or closed loop counting. The position is counted with encoder pulses and motor direction. In this mode, the encoder is used to determine the position of the motor. If no encoder is installed, the motor will not run.

**Motor Current Program:**

Current is set according to how many of the 4 switches are closed according to the formula:
Peak Motor current = 0.2 + 0.2(number of switches closed) Amps.
ex: all switches open = 0.2 amps or 3 switches closed = 0.8 amps (typical setting)



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

## DC Motor Controller Commands:

**54** **10** **Load DC controller parameter values**. Data format:

| | |
|---|---|
| Byte 1: | DSP Filter Gain. Range 0-255. (Static gain. See Cmd 18) |
| Byte 2 | DSP Filter Zero. Range 0-255. |
| Byte 3: | DSP Filter Pole. Range 0-255. |
| Byte 4: | Counts per Revolution LSB. |
| Byte 5: | Counts per Revolution MSB. |
| Byte 6: | DSP Filter Acceleration LSB in CPR/sampletime$^2$. |
| Byte 7: | DSP Filter Acceleration MSB in CPR/sampletime$^2$. |
| Byte 8: | Trip Current. Value = Max Motor current(Amps) x 42.5. |
| Byte 9: | Sample Time. Value = (Samples in secs x 125000) $^{(-1)}$ |
| Bytes 10: | Current record number. Read Only. |

**154** **10** **Read DC controller values**. Format same as above.

**NOTES:**

1) Parameter presets. DSP filter boot-up values are dependent on DIP switches S1:4-6. If all three configuration DIP switches are closed, the system boots up with the last record parameters saved through command 38. See Command 38 for further details. Records 1 through 7 are factory presets and correlate to one or more LEP products. See table below.

| DIP SW1:4-6 | Record number | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Closed = 0 | **0** (Default) | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Open = 1 Sw4=lsb | Last Record stored | Stage1 3x2, 4x4 | Stage2 6x6, 8x8 | Robot Arm | Robot Rotation | Robot Lift | Focus Drive | GearHead Motor |

2) Changing Counts per Revolution or Sample time causes system to recalculate internal system parameters, of which takes about half a second. Therefore check busy status before writing a new command to systems or wait more than half a second.

3) DSP filter parameters effect the system with the following formula and table below. For more information see Hewlett Packard's HCTL-1100 Application Note 1032. http://www.hp.com/HP-COMP/motion/app_index.html.

$$MC_n = (Gain/4) * X_n - [ ((Pole/256) * MC_{n-1}) + ((Gain/4) * (Zero /256) * X_{n-1}) ]$$

n = current sample time
n-1 = previous sample time
$MC_n$ = Present Motor Command Output.
$MC_{n-1}$ = Previous Motor Command Output.
$X_n$ = Present error between the Actual Pos. and Command Pos.
$X_{n-1}$ = Previous error between the Actual Pos. and Command Pos.

| Increase in Parameter | Stability | Response | Stiffness (1/dead band) |
|---|---|---|---|
| Zero | Better | Faster | Decreases |
| Pole | Slightly Better | Faster | Decreases |
| Gain | Worse | Faster | Increases |
| Sample time | Worse | Slower | Decreases |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

4) The smaller the sample time the faster the overall system speed, but the higher the slowest speed. Minimum sample time is 0x0F. Default is 0x40.
5) Current is in Amps and has a range of 0 to 6 amps. Programmed value = Max Amps * 42.5. If the motor exceeds this value for more than 0.8 seconds the motor driver will be shut down and the over current error bit is set. See command 155 and 136. If set to 0xFF the over current sensor is disabled. Default = 0xFA ( 5.88 Amps).
6) Current Record number is the last record number that was loaded. There are 20 record numbers (0 - 19). See Command 38. This field is read only and is ignored on Load DC Controller Parameter Values, command number 154.
7) DSP Filter Acceleration is the actual acceleration the DSP system uses. The value maybe read or set through commands 54 or 154. The range is 0-0x7FFF the LSB is 256 resolution fraction, the units are in CPR/sampletime$^2$. The old command read/set ramp time (commands 113 and 81) may still be used, but has limited programmable range. To Convert from rpm/sec to CPR/sampletime$^2$ use the following equation and then multiply by 256 to include the fraction, then write it to the register.
CPR/sampletime$^2$ = RPM/s * CPR * Sampetime(sec) * (0.01667/rpm-sec).
8) See command 18 for more information on DC DSP gains.

**38** **1**

**Load or saves a data record**. A data record contains all the internal parameters the system needs to configure the DSP. Local parameters can be edited using commands 54 and 154, and then saved with this command. Command syntax's: The 7 lower bits of the data hold the record number n. The most significant bit of the data, instructs the system to load or save record number n. A high 8$^{th}$ bit will save the current parameters and a low 8$^{th}$ bit will load the current parameters. Available record numbers are 0 - 19. See table below. Record numbers 1-6 are read only and are factory set. Records 0,7-15 are read/write and are stored in nonvolatile memory. Record numbers 16-19 are stored in volatile Ram and are not preset. Therefore the user must save to this area before loading from this area. The advantage of this area is access time. A typical use for this area would be to change more than one parameter quickly between two different types of moves.

| Record Number | Access | Mem Type | Preset | Selectable |
|---|---|---|---|---|
| 0 | Read/Write | NonVolatile | Yes | Software |
| 1-6 | Read Only | NonVolatile | Yes | Software and DIP SW |
| 7 | Read/Write | NonVolatile | Yes | Software and DIP SW |
| 8-15 | Read/Write | NonVolatile | Yes | Software |
| 16-19 | Read/Write | Volatile | No | Software |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

Command 38 also loads/saves a few global parameters. These global parameters are stored in a common non-volatile area. Each load or save record will also load or save these common parameters. See table for a list of record parameters and common parameters.

| Items saved/loaded to RECORD # area. | Related Command #'s | Items saved/loaded to COMMON area. | Related Command #'s |
|---|---|---|---|
| Static Gain | 54,154,18,138 | Last Record Saved | 38 |
| Dynamic Gain | 18,138 | Joy Stick Fast Speed | 70,102 |
| Boost Gain | 18,138 | Joy Stick Slow Speed | 69,101 |
| DSP Filter Zero | 54,154 | Servo Settle Time | 95,130 |
| DSP Filter Pole | 54,154 | Servo Timeout Count | 18,138 |
| Counts Per Revolution | 54,154 | Servo Activation Dist. | 79,111 |
| Acceleration | 54,154,81,113 | Servo Mode On/Off | 77,78 |
| Max Trip Current | 54,154 | Wafer Width | 53,119 |
| Sample Time | 54,154 | | |
| Top Speed | 83,115 | | |

**155** **1** **Read system error byte**. Byte cleared when read. Active High.

| Msb | | | | | | | Lsb |
|---|---|---|---|---|---|---|---|
| bit7 | bit 6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| Control Loop Error | Bus Cmd Error | Xicor Error | Float Point Error | Bus Cmd Full | Serial Rx Full | Over Temp | Over Amp |

Over Amp Set when current remains above the trip level set by command 54 byte 8 for longer than 0.8 seconds.
Over Temp Set when motor driver chip temperature reaches 145 C.
Motor driver chip shuts off if temperature raises over 170 C.
Serial Rx Full Set when serial buffer input overflows.
Bus Cmd Full Set when bus command input buffer overflows.
Float Point Error Set on floating point math error.
Xicor Error Set on error reading xicor chip or xicor chip not present.
Bus Cmd Error Set on bus command error. Such as unknown command, invalid handshaking, data lost, or incorrect data structure.
Control Loop Error: Set on boot up if encoder loop is unstable, Reversed or Missing.
The control loop error can only be clear by rebooting.


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| **Motion Control:** | | |
| **71** | **0** | **Start motor motions**. Initiates a motor move command towards the target position (See Command 84). The busy bit will be set while the motor is moving. The busy bit will be reset at the end of the move profile. The status/quality/termination of the move may be verified with command 136. Note that actual position should be read (see command 97) to verify system is indeed at desired position. |
| **66** | **0** | **Stop motor motions**. If motor is running, it will come to a stop after ramping down. It will respond with a not busy code after a complete stop if status is requested. **Power up default.** |
| **60** | **0** | **Motor power on**. Motor power should be turned on before running motors. **Power up default.** |
| **61** | **0** | **Motor power off**. To turn the power off to the motor phases. Motor will not run if motor power is turned off with this command. |
| **39** | **0** | **Goto negative endlimit**. Motor is activated in the negative direction until an end limit is reached, then repositioned on the limit. The status/quality/termination of this move may be verified with command 136. This move uses the top speed (command 83) to do its search for the end limit. Setting a slower top speed will result is more accurate/repeatable end limit capture. |
| 29 | **0** | **Goto positive endlimit**. Motor is activated in the positive direction until an end limit is reached, then repositioned on the limit. The status/quality/termination of this move may be verified with command 136. This move uses the top speed (command 83) to do its search for the end limit. Setting a slower top speed will result is more accurate/repeatable end limit capture. |
| **Position Parameters:** | | |
| **65** | **3** | **Write motor position counter**. This is a two's complement 3 byte number. This parameter is loaded only once on power-up and reloaded again if adjustments are needed. |
| **108** | **4** | **Read motor position counter with status byte**. The first three bytes is the two's complement motor position counter. And the next byte is the status byte. The counter is updated with every movement of the motor. See "**Read Status Byte"** explained later for the format of the status byte. |
| **64** | **3** | **Write second motor position counter**. This is a two's complement 3 byte number. This parameter is loaded only once on power-up and reloaded again if adjustments are needed. Usually this counter is used for open loop counting. |
| **103** | **4** | **Read motor second position counter with status byte**. The first three bytes is the two's complement motor position counter. And the next byte is the status byte. |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| | | The counter is updated with every movement of the motor. See "**Read Second Status Byte "** explained later for the format of the status byte. |
| **97** | **3** | **Read motor position counter**. This is the two's complement motor position counter. The counter is updated with every movement of the motor. |
| **84** | **3** | **Write target position counter**. This is a two's complement 3 byte number. This parameter is loaded every time motor should be moved to different position. |
| **116** | **3** | **Read target position counter**. This is a two's complement 3 byte number for the target counter. |
| **Incremental Movements:** | | |
| **43** | **0** | **Incremental move upwards**. This command will add the increment buffer to present position buffer and load this position to target and then move the motors. It is used when motor movement distances are same. It combines the loading new target and move commands in a single byte command. |
| **45** | **0** | **Incremental move downwards**. Same command as above except that direction is reversed. |
| **68** | **3** | **Write Increment value**. This is a two's complement 3 byte number for the increment counter. Use this value to program the distance for incremental moves. |
| **100** | **3** | **Read Increment value**. This is a two's complement 3 byte number for the increment counter. |
| **46** | **3** | **Incremental up and down relative moves**. The parameter specified is in two's complement 3 byte number. The motor may be moved forward or backward, depending on the sign of the parameter. The movement is relative to the present position. |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Variable Speed Rotating:**

**47** **3**

**Write rotating speed to motor driver**. This command will rotate the motors without specifying any other target or speed. This number is a 3 byte signed number which specifies the direction and the speed of the rotation. When this number is positive motor will rotate towards bigger numbers and when it is negative will rotate towards smaller numbers. This command is reloadable. If the motor is already rotating it will ramp up or down depending the last speed loaded. There is no target number to stop the rotation of the motor. The ramp value may also be reloaded to reflect the new ramping time. While motor is rotating run flag is set in the motor status byte. The status/quality/termination of this move may be verified with command 136.

Motor will stop if one of the following event is occurred:
1. A stop command is received.
2. An end limit switch is closed.
3. The same command is received with the speed equal to zero.

The following equation should be used in order to convert speed given in pulse per second unit to a binary value to be loaded with this command:

output value= 8,388,608-(5,529,600/speed)

The speed is given in pulse per second and it is positive or negative depending the direction of the rotation.

The programmable speed limits are:
minimum speed = 0.659 pulse per second.
maximum speed = 5,529,600 pulse per second.

It should be noted that the maximum speed specified above is the internal calculation limit and does not necessarily means the running speed of the motor installed in the system. It is left to the user to limit the actual maximum running speed.

There is no reading back command the last speed written with this command at the present time.

**Centering The Axis:**

**48** **3**

**Center or search for home pulse command**. This command's parameter value is similar to the "**Variable Speed Rotating**" command explained earlier, but command's function is different. The status/quality/termination of this move may be verified with command 136.

This command may be used for two different purposes. First one is to locate the motor to a specific position. And the second one is to center the motor on the axis between two endlimit switches.

The motor will travel between two endlimit switches. While traveling a positive going edge of a pulse will stop the motor where the pulse was received. This pulse


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

should be at least 5 microsecond of duration. In case this pulse is not detected or it is not present at all, then motor is positioned in the center of the axis. The length of the axis is defined with the endlimit switch positions.

The travel speed between endlimits is programmed with the command parameter but the final centering speed is the programmed running speed.

| | | |
|---|---|---|
| **30** | **0** | **Enable position counter capture**. This command will enable reception of the pulse which normally used as the home pulse. This command should be used before the center command if centering is desired with the pulse. If pulse is disabled centering is done by moving to the center of axis by detecting the two end limit switches. |
| **31** | **0** | **Disable position counter capture**. Power up default state. Recommended use of this command will be after centering an axis. |

**Load Center Position:**

| | | |
|---|---|---|
| **91** | **3** | **Load the capture center position.** |

**Read Center Position:**

| | | |
|---|---|---|
| **123** | **3** | **Read the captured center position**. |

The last two commands may be used with the centering command explained earlier(see "**Centering The Axis**"). If a high precision positioning required a stepping motor may be installed with a center pulse generator. When motor is passed through the position a center pulse is generated and the position is recorded by the module. These commands may be used to determine if centering of an axis is done by centering pulse or by internal calculation. First load the center position with a large number presenting over the limits. Then center the axis. At the end of centering use the Read Center Position command. Compare the numbers loaded and read. If they are the same then the centering pulse is not received. If they are different then the center pulse is received properly.

Centering pulse also may be used as to move the motor until an event occurs. If a center command is used then motor will stop when a center pulse is generated. By reading the motor position, the location of the event can be determined.

**Ramping Time:**

| | | |
|---|---|---|
| **81** | **1** | **Write ramping tim**e. This is a one byte straight binary number, which will change the acceleration value. It is related to the time where speed is changed. Smaller values are equal to bigger accelerations. **Power up default is 20.** |


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| **113** | **1** | **Read ramping time**. Same as above to read back the value. |

**Motor Speed Write Commands:**

| | | |
|---|---|---|
| **82** | **2** | **Write motor starting speed**. This is a 2 byte straight binary number. This is the speed the motor will start ramp up. If this value is bigger than top speed, then there is no ramping up. The equation below can be used to convert the speed given in pulse per second. |
| **83** | **2** | **Write motor top speed**. This is a 2 byte straight binary number. When motor is moved, this will be the running speed after ramping up. The equation below can be used to convert the speed given in pulse per second.<br><br>The following equation can be used to convert a given speed in pulse per second units to a binary number. This is the number which is used with write starting speed and write top speed commands.<br><br>**binary output value = 65536-(5529600/speed)**<br><br>Speed is the desired speed in pulse per second units.<br><br>The maximum binary output value = 65534<br>The default binary output value = 65259<br>The minimum binary output value = 1<br><br>The maximum speed = 2764800 pulse per second.<br>The default speed = 20000 pulse per second.<br>The minimum speed = 84.375 pulse per second. |

**Motor Speed Read Commands:**

| | | |
|---|---|---|
| **114** | **2** | **Read motor starting speed**. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |
| **115** | **2** | **Read motor top speed**. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |

The following equation can be used to convert the binary number read back to speed in pulse per second units.

**speed(pulse/sec) = 5529600/(65536-binary value)**

**Joystick Speed Write Commands:**

| | | |
|---|---|---|
| **69** | **1** | **Write joystick normal top speed**. This is a one byte straight binary number. This will affect the joy- stick top speed when it is deflected to either side without pressing the fast joystick key. |


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| **70** | **1** | **Write joystick faster top speed.** This has the same function as above except that it will affect the maximum speed when fast key is pressed with joy-stick deflected either side. How to calculate these two numbers is explained at the end of this manual. |

The following equation can be used to convert a given speed in pulse per second units to a binary number. This is the one byte number which is used with write joystick speeds.

**binary output value = 1843000/speed**

Speed is the desired speed in pulse per second units.

The maximum binary output value = 255 7.227KHz
The minimum binary output value = 1 1.843MHz

**Joystick Speed Read Commands:**

| | | |
|---|---|---|
| **101** | **1** | **Read joystick normal top speed.** This is a one byte straight binary number. |
| **102** | **1** | **Read joystick faster top speed.** This is a one byte straight binary number. |

The following equation can be used to convert the binary number read back, to speed in pulse per second units.

**speed(pulse/sec) = 1843000/binary value**

**Joystick Control:**

| | | |
|---|---|---|
| **74** | **0** | **Joystick enabled.** To enable the joystick or trackball function when motor is not moving. **Power up default.** |
| **75** | **0** | **Joystick disabled.** To disable the joystick or trackball function when motor is not moving. |
| **72** | **0** | **Make joystick fast speed to ramp up.** This command will affect how the joystick fast key will act when pressed with joystick deflected either side. With this command when joystick fast key is pressed, motor will ramp up to faster then joystick speed. **Power up default.** |
| **76** | **0** | **Make joystick fast speed to ramp down.** This is the opposite of the above command. When joystick fast key is pressed, motor will ramp down to a slower speed than joystick speed. |

**Joystick simulations**:

| | | |
|---|---|---|
| **44** | **1** | **Load soft joystick values.** This value when is equal to 128 will have no affect on joystick movements. Any other value will simulate the joystick deflections. In other words joystick deflections can be downloaded from a external computer. Mid point of joystick value is 128 which will stop the joystick movements. Bigger values will |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| | | move motor in one direction and smaller values will move motor in the other direction proportional to the value loaded. |
| **122** | **1** | **Read joystick deflection values.** The value read will be the deflection values read from analog input port for joystick. The value is ranged from 0 to 255. Mid position has the value of around 128. |

**Load Open Loop Constant:**

| | | |
|---|---|---|
| **92** | **1** | **An unsigned one byte binary number.** This number represents the open loop constant explained above. The valid range of open loop constant is from 1 to 255. On power up default value is 5. In a special case when loaded number is equal to zero an internal routine is executed to calculate both constants. However the precision of the calculation is limited and the result may not be correct. For this reason when motor is no longer busy, both closed and open loop encoder constants should be examined and corrected by the host system.<br>**Not Implemented. For backwards compatibility.** |

**Read Open Loop Constant:**

| | | |
|---|---|---|
| **124** | **1** | **Read the open loop counter constant** programmed.<br>**Not Implemented. For backwards compatibility.** |

**Load Closed Loop Constant:**

| | | |
|---|---|---|
| **32** | **1** | **An unsigned one byte binary number**. This number represents the closed loop constant explained above. The valid range of values for this number is from 1 to 255. On power up default value is 1.<br>**Not Implemented**. **For backwards compatibility.** |

**Read Closed Loop Constant:**

| | | |
|---|---|---|
| **131** | **1** | **Read the closed loop counter ratio** programmed.<br>**Not Implemented. For backwards compatibility.** |


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| **Servo Control**: | | |
| **77** | **0** | **Turn servo checking on.** This mode will load a boost gain to the motor after the completion of a move command (Command 71). The intention of the boost gain is to force the mechanical system closer to it's target position. This boost gain will generally be higher than the static or dynamic gains (see Command 18). The system will remain at this boost gain until either; the error between the actual position and the target position are less than servo activation distance (see command 79), or the system has times out (see Command 95). |
| **78** | **0** | **Turn servo checking off.** This command cancels the above one. |
| **95** | **1** | **Write settling time value**. This one byte unsigned value may range from 0 to 255. This value is used to add a delay between the two position samples used in servo mode. The larger the number the more time the system takes to acquisition actual position, of which is compared to the target position. Larger numbers increase position accuracy, but increase overall system move time. Certain number may match system resonant frequency and cause system to oscillate. |
| **130** | **1** | **Read settling time value.** |
| **79** | **3** | **Load servo activation distance**. Programmed in units of encoder steps, this number represents the minimum allowable position error. |
| **111** | **3** | **Read servo activation distance.** |
| **18** | **4** | **Write Dynamic gain control.** Used for high resolution scales which tend to cause instability when a high static gain is used. The individual gains can be tailored to give high stability and accuracy. If in servo mode and after a move is complete, gain is increased to the boost level and held until actual position is within ±(servo activation distance) of the target position. Until this command is used all three gains will be loaded with the static gain value when one uses the load DSP command (Command 54). After this command is used only static gain will be load with the load DSP command. Also included in this command is the servo mode time out value. This value will allow the servo mode loop to cycle n times before it times out. This code is added starting at version 2.3.<br><br>Data format:<br>Byte 1: **Static gain** used when motor is stationary.<br>Byte 2: **Dynamic gain** when motor is moving.<br>Byte 3: **Boost gain** used to minimize error at end of move.<br>Byte 4: **Servo Mode Time Out** range 0-255. |
| **138** | **4** | **Read Dynamic gain control.** Reads 4 bytes, data format the same as command 18. |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Servo Speed Write Command:**

**93** **2**

**Write motor servo speed.** Not Currently implemented in the EMOTD.
**T**he servo movements are done with the speed programmed with this command. The speed equation is the same as explained with motor speeds. The minimum and maximum speeds are different then the motor speeds. Default value on power up set to 300 pulse per second.

The same equation to calculate the starting and top speed can be used for this command. If speed loaded is outside the range the correct speed limit is replaced by internal software.

**binary output value = 65536-(5529600/speed)**

Speed is expressed in pulse per second units for the stepping motor.

The maximum speed = 1250 pulse per second.
The minimum speed = 100 pulse per second.

**Servo Speed Read Command:**

**129** **2**

**Read motor servo speed.** The returned value is an binary unsigned integer representing a speed factor used internally.
**Not Implemented. For backwards compatibility.**

The following equation can be used to convert the binary number back to speed in pulse per second units.

**speed = 5529600/(65536-binary value)**

**Read Status Byte :**

**126** **1**

**Read status byte from motor driver.** This value will reflect various internal status of motor driver. The format is explained below.

| bit | description |
|---|---|
| 0 | 0=motor not running.<br>1=motor is running. |
| 1 | 0=servo is OFF.<br>1=servo is ON. |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|
| 2 | | 0=motor phases are turned OFF.<br>1=motor phases are turned ON. |
| 3 | | 0=joystick is turned OFF.<br>1=joystick is turned ON. |
| 4 | | 0=motor is not ramping.<br>1=motor is ramping up or down. |
| 5 | | 0=ramping down.<br>1=ramping up. |
| 6* | | this bit is the soft copy of the CW end limit switch. |
| 7* | | this bit is the soft copy of the CCW end limit switch. |

* Note: *changed 10/23/98 to reverse bits 6 and 7. The corrected values are now shown*

**Read Second Status Byte :**

| | | |
|---|---|---|
| **128** | **1** | **Read status byte number two.** This value will reflect various internal status of motor driver. This byte also appended with the read second position counter command. The format is explained below. |

| bit | description |
|---|---|
| 0 | 0=motor is not stalled.<br>1=motor is stalled. |
| 1 | 0=motor is NOT servo aligning when at idle.<br>1= motor is servo aligning when at idle . |
| 2 | 0=NO EEPROM(xicor) is installed.<br>1= EEPROM(xicor) is installed. |
| 3 | Software copy of Home limit switch. |
| 4 | Software copy of Pre limit switch. |
| 5 | Target reached bit.<br>Set (High) when move command (Cmd 71) is called.<br>Cleared (low) when motor has reached its target position ± activation distance (Cmd 79)<br>If a limit, stop command or external obstruction stops the motor, this bit remains Set (High). |


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Move Status Byte :**

**136** **1** **Read move status byte.** This value will reflect various possible move quality and terminations of all moves, (Commands 71,43,45,46,47,29,39,48). This register is cleared to 0x00 on a call to move the motor. This register can be read at any time. Bits 1-7 are valid after Bit 0/7 is Set. The format is explained below.

| bit | description |
|---|---|
| Bit 0/7 | System Busy bit. Set when the move is completed. When this bit is set, this register will indicate the status of the last move made.<br>Cleared (0) = Busy, Set (1) = Move Done, Not Busy. |
| Bit 1/7 | Set when the move has been completed normally. Normally means that the move has ramped up and ramped down as defined by the DSP parameters. It does not necessary mean that the motor is at the target position. How close this position is, depends on how well the system is tuned. With a rotate move, normally means the rotate move was terminated with a rotate (zero) command. |
| Bit 2/7 | Set if the completed move position is less than the target position +/- the servo activation distance (#79). This bit is updated in both servo mode and non-servo mode. This bit is not used with the rotate command 47 |
| Bit 3/7 | Set if the move was terminated by user stop command (#66). |
| Bit 4/7 | Set if a stall, crash, over current, or over temp is found. |
| Bit 5/7 | Set on invalid parameter, system busy, or motor power off.<br>An invalid parameter is a parameter that is out of range. ei Acceleration of zero. |
| Bit 6/7 | Set if CW limit terminated the move.<br>See Note below. |
| Bit 7/7 | Set if CCW limit terminated the move.<br>Note: If the Capture Home is enabled (#30) and the Home limit is found both Bits 6/7 and 7/7 will be set. |

**PWM Commands :**

**34** **2** **Read the current motor PWM value and the current.** The PWM value is a signed char value. With a range of –100 to +100. The current is updated every 30ms and must to divide by 42.5 to convert in to Amps.

1**34** **2** **Sets the Motor PWM Value.** This command allows the motor to run in constant torque mode. The first byte is a signed char (+/-100), who's value times 24 represents the voltage across the motor. The second byte of this command is not currently used and is revered for future uses, a zero should be write to it for future compatibility. In order to use this feature you must turn off the motor power first. Then you may write successively to this register. On exit of this mode you MUST write zero to the PWM register and than re-enable power.


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Identification:**

**105** **6** **Read identification from device.** First 5 characters read are the device ID **EMOT**. The sixth character is a binary value which reflects the state of the configuration switches according to the following table:

| Msb | | | | | | | Lsb |
|---|---|---|---|---|---|---|---|
| bit7 | bit 6 | Bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| sw6 | sw5 | Sw4 | sw3 | sw2 | sw1 | X | X |

Switch open = 1
X denotes a don't care bit.

**Version 5.8 and newer:**

The read identification code will have the following function starting with version 5.8 EMOT devices:

The same code may be used consecutively in order to get a version information from device.
When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

**Read Version No.:**

**127** **6** **Read date and version.** This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented.


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Request Device Status:**

| | | |
|---|---|---|
| **63** | **0** | **Request Status.** Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.<br><br>66 = If device is busy.<br>98 = If device is not busy. |

Parameters should only be loaded when a device is not busy, but they can be read anytime.

**Soft Limits Commands:** Soft limits can be programmed by the user to limit the overall travel of a position move (see command 71). If enabled, the system will automatically readjust target position so that they fall within the soft limit range, set up by commands 17 and 210. Note that if the system is outside the soft limit range and soft limits are enabled, the system will ignore position command moves.

| | | |
|---|---|---|
| **16** | **1** | **Activate or deactivate soft limit detection.**<br>Data = 0 : Deactivate soft limit detection. Power up default.<br>Data <> 0 : Activates soft limit detection. |
| **17** | **6** | **Write Soft Limit Data:** Setups up the soft limits. First three bytes are one limit and the second three bytes are the second limit. The smaller limit will be saved as the negative or lower limit and higher will be saved as the positive or higher limit. Syntax is as follows:<br>**Byte 0:** LSB of Soft Limit 1.<br>**Byte 1:** MIB of Soft Limit 1<br>**Byte 2:** MSB of Soft Limit 1<br>**Byte 3:** LSB of Soft Limit 2<br>**Byte 4:** MIB of Soft Limit 2<br>**Byte 5:** MSB of Soft Limit 2 |
| **210** | **6** | **Read Soft Limit Data:** Reads the soft limit data .First three byte are the higher limit. Second three byte hold the lower number. Byte order is LSB,MIB, and MSB.<br>**Byte 0:** LSB of Higher Soft Limit.<br>**Byte 1:** MIB of Higher Soft Limit.<br>**Byte 2:** MSB of Higher Soft Limit<br>**Byte 3:** LSB of Lower Soft Limit.<br>**Byte 4:** MIB of Lower Soft Limit.<br>**Byte 5:** MSB of Lower Soft Limit |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

## Access Commands:

NOTE: The following peek and poke commands allow direct access to system resources. Unintelligent poking will cause system to become unstable.

| | | |
|---|---|---|
| **6** | **4** | **Poke/Peek setup function**. Depending on Address space (byte 3) this command either writes data to AddressSpace:Address or sets up the peek function pointer to AddressSpace:Address.<br>Format:<br>Byte 1: Address LSB.<br>Byte 2: Address MSB.<br>Byte 3: Address space.<br>Where 1= Write to SFR, 2= Write to Idata, 3= Write to Xdata,<br>5= Point to SFR, 6= Point to Idata, 7= Point to Xdata,<br>8= Point to Code Space.<br>Byte 4: Data to be written or length of bytes returned from the peek function.<br>Default is 1. |
| **196** | **n** | **Peek Function**. Returns n bytes of data at AddressSpace:Address pointer. Increments pointer by n when completed. Range 1-255. Default is 1.<br>Default address is Xdata:0000.<br>Format:<br>Byte 1-255: Data. |

## Undocumented Commands: ***Preliminary***

| | | |
|---|---|---|
| 57 | 0 | **Sets Limit Trap On**. If enabled, the system will set the stall bit (see Command 128) if a limit is found during a position move or the target position is not reached +/- the servo activation distances (#79). |
| 59 | 0 | **Sets Limit Trap Off.** Cancels the above mode. Default power up. |
| 41 | 3 | Loads ramp offset. |
| 107 | 3 | Reads ramp offset. |
| 94 | 2 | Loads stop Speed. |
| 98 | 2 | Reads stop speed. |


© 1998 LEP

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **73002051 DC Servo Motor Controller** **Document Revision 1.6**
Device ID: **EMOTD** **Document Date: 3/11/99**

| Code: | Data Length: | Description: |
|---|---|---|

**Configuration:**

There two (2) DIP switches installed on the board. One with eight (8) switches is used for device address selection, and one with six (6) switches is used to control different functions.

Each device can be configured to have addresses from 0 to 254. This can be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 |
|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**Function Switch Settings:**

Switch 1: This switch reverses the joystick direction in relation to the motor direction.
Switch 2: The end limit detection level.
0 = end limit should switch to ground to stop the motor.
1 = end limit should switch to high to stop the motor.
Switch 3: This switch controls the motor direction relative to the position counter. If a different direction is desired when counter is counting up or down, this switch should be reversed.

Switch 4-6: Parameter presets. DSP filter boot-up values are dependent on DIP switches S1:4-6. If all three configuration DIP switches are closed, the system boots up in the last parameter record that was saved or loaded through command 38. See Command 38 for further details. Records 1 through 7 are factory preset and correlate to one or more LEP products. See table below.

| DIP SW1:4-6 | Record number | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Closed = 0 | **0** (Default) | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Open = 1 | Last Record | Stage1 | Stage2 | Robot | Robot | Robot | Focus | GearHead |
| Sw4=lsb | stored | 3x2,4x4 | 6x6,8x8 | Arm | Rotation | Lift | Drive | Motor |


© 1998 LEP
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Digital Analog IO Controller**
Device ID: **EDAIO**

| Code: | Data Length: | Description: | |
|---|---|---|---|
| **Digital Input**: | | | |
| 97 | 1 | Read digital input. | Input logic=True. Input pullups cause open input to read logic 1. |
| 129 | 1 | Read latched input | For use with switch closure inputs, transitions from 1 to 0 on digital input lines will cause a 1 to be loaded into the corresponding bit position for this command. Reading this byte will reset the latch. This code is supported in firmware DAIO ver 4.00 1/18/99. |
| **Digital Output:** | All digital outputs are open collector and floating at power up. | | |
| 68 | 1 | Load digital output. | |
| 100 | 1 | Read digital output. | Read Back of last value loaded with code 68. |
| **Analog Input:** | All analog inputs have 10k pull-down resistors. Range is 0 to 5.00 volt for 0 to 255 value read. | | |
| 104 | 1 | Read analog input channel 0. | |
| 106 | 1 | Read analog input channel 1. | |
| 107 | 1 | Read analog input channel 2. | |
| 108 | 1 | Read analog input channel 3. | |
| **Analog Input with 10 bit resolution:** | Extended resolution inputs similar 8 bit for commands explained above. Raw value read range is from 0 to 1023 counts. These codes are supported in firmware DAIO ver 4.00 1/18/99. | | |
| 140 | 2 | Read analog input channel 0. | |
| 141 | 2 | Read analog input channel 1. | |
| 142 | 2 | Read analog input channel 2. | |
| 143 | 2 | Read analog input channel 3. | |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Digital Analog IO Controller**
Device ID: **EDAIO**

| Code: | Data Length: | Description: |
|---|---|---|

**Analog Output Read:** Returns value loaded with the Analog Output Load code described below.

| | | |
|---|---|---|
| 109 | 1 | Read analog output channel 0. |
| 110 | 1 | Read analog output channel 1. |
| 111 | 1 | Read analog output channel 2. |
| 112 | 1 | Read analog output channel 3. |

**Analog Output Load:** Analog output range is 0 to 10 volt for 0 to 255 byte value. On power up, all analog outputs 0 volt.

| | | |
|---|---|---|
| 77 | 1 | Load analog output channel 0. |
| 78 | 1 | Load analog output channel 1. |
| 79 | 1 | Load analog output channel 2. |
| 80 | 1 | Load analog output channel 3. |

**Extended Output Load:** The following two codes are used to write a 16 bit value into the analog output channel 0. Analog output channel 1 is cascaded to channel 0 and can not be used in this mode. For this code output range is also 0 to 10 volts, but has a higher resolution.

| | | |
|---|---|---|
| 69 | 2 | Load analog output channel 0. |
| 101 | 2 | Read analog output channel 0. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Digital Analog IO Controller**
Device ID: **EDAIO**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Identification:**

| | | |
|---|---|---|
| 105 | 6 | Read identification from device. First 5 characters should match the device ID shown at the top of this page. |

The same code may be used consecutively in order to get a version information from device.
When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

**Read Version No.:**

| | | |
|---|---|---|
| 127 | 6 | Read date and version. This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented. |

Parameters should only be loaded when a device is not busy, but they can be read anytime.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Digital Analog IO Controller**
Device ID: **EDAIO**

**Configuration:**

There are two (2) DIP switches installed on the board. One with eight (8) switches is used for device address selection, and one with four (4) switches is used to control different functions.

Each device can be configured to have addresses from 0 to 254. This can be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary number. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 |
|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**Function Switch Settings:**

Switch 1: 0 = multiboard interface mode. When serial interface is done with RS 232 board.
1 = Single board interface mode. If serial interface is done to the individual boards.

Switch 2: Not used.

Switch 3: Not used.

Switch 4: 0 = Analog outputs initialized to 0 volts on power-up.
1 = Analog outputs initialized to mid scale on power-up.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|

Description: The MAC2002 Flat Aligner, 73002075 is a single card which integrates the functions of a microstepping motor drive along with the sensing and control functions. Both flat and notch find modes are software selectable. The operating codes are backwards compatible with Mac2000 with exceptions (noted by * in code column).

**Flat Find Control**:

| | | |
|---|---|---|
| 32* | 0 | **Align Wafer**. This command will align wafer with a flat or a notch depending upon how front panel the manual switch on the front panel. |
| 33 | 0 | **Find Flat** on a wafer. This command code will align wafer by looking for a flat. Wafers with one or two flats can be aligned. In case of wafers with two flats, the larger flat is aligned with the sensor. |
| 34 | 0 | **Find Notch** on a wafer. This command code will align wafer by looking for a notch. If calculated eccentricity values will be used, a wafer should have ONLY one notch cut out. |
| 37 | 0 | **Go to original position.** It will rotate the wafer to the original position where it was first put on the wafer chuck. When it is desirable to rotate the wafer to the original position, this code should be used because the position counter is cleared to zero when wafer is aligned; therefore, it was not possible to keep the original load position. This code can also be used if wafer is not aligned or a new position counter is loaded. |
| 136 | 0 | **Reset original position.** If original position is used to re-position the wafer to the original placement then this command should be used when a new wafer is placed on the aligner. The flat aligner will not remember the original position if wafer is not rotated to the original position. This command must be used before any alignment command is used for a new wafer placed on the aligner. |
| 110 | 3 | **Read original position.** The value read is the original position where the wafer is placed on the aligner before any align command is executed. This value is the same position used by the Go to original position command. This value may be used externally to position the wafer to its original position even if it is located another inspection station. |
| 51 | 0 | **Get a coarse edge profile with flat.** The profile of the edge is collected and eccentricity is calculated. The profile is taken with rotating the wafer through the light detecting sensor and measuring the light intensity at periodic intervals. Upon completion of this code, the center of the wafer, the center of the wafer chuck, and the robot hand are aligned on the same axis, if the proper value is loaded with offset code. In other words the wafer is rotated to the position where |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|
| | | sensor is covered to the maximum point. If the wafer is moved up vertically and wafer center is moved in horizontally towards the chuck center for the amount of steps in the eccentricity value, the distance between the wafer center the chuck center will be shorter. This function may be repeated until eccentricity values will drop below the threshold value set by the user. When wafer and chuck centers are aligned use the FIND FLAT code immediately after this code, in order to align the wafer. |
| 52 | 0 | **Find flat.** This code should only be used after using the code above. Otherwise it will have no meaning. This code will align flat side with sensor without taking another coarse profile. The coarse profile taken with the previously explained code will be used by the internal firmware in order to locate the flat of the wafer. A fine profile is then taken in order to find the exact peak of the flat. It should be noted that the larger flat is selected internally if more then one flat exist in a wafer. |
| 53 | 0 | **Get a coarse edge profile with notch.** This code will execute the same function as get coarse profile for flat code (see code 51). The only difference is that the wafer should have a notch, not a flat. |
| 54 | 0 | **Find notch.** This code will execute the same function as the code 52. |

All of the above wafer alignment commands will end with clearing motor position counter.

| | | |
|---|---|---|
| 36 | 0 | **Rotate Wafer to the Offset value.** It may be used after wafer alignment commands. It will rotate the wafer additional steps from the alignment point. The additional steps are set by Load offset value command. This code is discarded starting from version 2.0. |
| 86* | 0 | Disable front panel switch. This command code will disable the front panel switch. Manual flat finding is not possible until enabled again. |
| 85* | 0 | Enable front panel switch. Power up default. |
| 90 | 0 | **Calibrate intensity of light source.** The ‘2075 board relies on external manual calibration. On power up light source off. Executing this code reads the sensor type. If a laser sensor is used, the beam is turned off. If a filament type sensor is used, the lamp will remain on. |
| 122 | 2 | **Read single sensor value**. A single read of the flat sensor value. This command is used to read sensor for manual calibration. The sensor is calibrated unblocked for the following values: |

| Sensor | Laser type | Filament type |
|---|---|---|
| Open (gain) | 990±10 | 990±10 |
| Blocked (zero) | 200 | 0 |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|

**Light Beam Sensor Angle:**
For theta mounted on stage with stage correction:
In order to calculate the eccentricity of a wafer, the position of the light sensor used for aligning the wafers has to be known relative to the aligner module. This value is an angle given in degrees between the sensor and stage Y axis. The motor reverse direction switch position is taken into account when calculating eccentricity values and stage is moved accordingly. The angle is not relative to the motor reverse direction switch.

To correctly determine the angle, the following guides have to be kept in mind:

Consider the center of stage at (0,0) coordinates.

X AXIS:
Reverse motor direction is closed (=0):
Stage should move LEFT for positive numbers and move RIGHT for negative numbers.

If reverse motor direction is open (=1):
Stage should move LEFT for negative numbers and move RIGHT for positive numbers.

Y AXIS:
If reverse motor direction is closed (=0):
Stage should move UP for positive numbers and move DOWN for negative numbers.

If reverse motor direction is open (=1):
Stage should move UP for negative numbers and move DOWN for positive numbers.

If the stage is performing according to the given conditions above, then determine the angle from the following:

The angle of the sensor is at:

0 degrees if reverse direction of Y AXIS is closed and stage moves toward the sensor for positive numbers or if reverse direction of Y AXIS is open and stage moves away from the sensor for positive numbers.

90 degrees if reverse direction of X AXIS is closed and stage moves away from the sensor for positive numbers or if reverse direction of X AXIS is open and stage moves toward the sensor.

The 180 and 270 degrees are also determined by taking the opposite sides of the X and Y axis. Any other angle should be determined by taking interim values of these angles.

| | | |
|---|---|---|
| 80 | 2 | **Load sensor angle.** The range is from 0 to 360 degrees. The angle of sensor location will be used for eccentricity computations. This value is only important if eccentricity values are used. Power up |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|
| | | default value is zero (0). Starting from version 2.0, this value has to be set to zero if a stationary stage is used or stage is not moved in order to align the centers of wafer and chuck, in which case OFFSET value has to be programmed. See code 79. |
| 112 | 2 | **Read sensor angle**. |
| 117 | 3 | **Read eccentricity value of X.** |
| 118 | 3 | **Read eccentricity value of Y.** |

The above two values are only significant after an alignment is performed successfully. The values represent the distance between center of the wafer and center of rotating table. The unit of measurement is in motor steps. They are binary numbers in two's complement form.

| | | |
|---|---|---|
| 67 | 2 | **Load the conversion factor** of beam intensity to motor steps of the stage. This value is only important if eccentricity values are used. Conversion factor is a number used by internal firmware to convert the measured light intensities to the number of motor steps when calculating eccentricities. This converted number is used by other means of locating the wafer on the wafer chuck. This value is relative to the resolution of the stage motor. For example if this value is 200, it is necessary to move the wafer 200 steps in or out in order to detect one unit of light measurement. Power up default value is 35. |
| 99 | 2 | **Read the conversion factor** of beam intensity to motor steps. |
| 79 | 3 | **Load offset value.** this value represents the angle, in units of steps, between the sensor and the pickup arm. For applications where stage is kept stationary, and robot hand is used to center the wafer against the chuck center, this value should be programmed, otherwise it should be set to zero. (Ref code 80)<br>Power up default value is zero (0). |
| 111 | 3 | **Read offset value.** |

**Status Byte:**

| | | |
|---|---|---|
| 126 | 1 | **Read the status byte.** The following table defines bits used in the status byte. The status bits have meaning when an alignment command is executed. They are valid until a new alignment command is used. |

| | | | |
|---|---|---|---|
| bit | 7 | : | 1=alignment in progress.<br>0=alignment is finished. |
| bit | 6* | : | 1=manual alignment enabled.<br>0=manual alignment disabled. |
| bit | 5 | : | 1=light beam sensor is calibrated.<br>0=light beam sensor is NOT calibrated. (Power up state) |
| bit | 1 | : | 1=alignment is interrupted by a break |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| | Code: | Length: | Description |
|---|---|---|---|
| | | | code. |
| | | | 0=normal end. |
| bit | 0 | : | 1 = unsuccessful alignment. |
| | | | 0 = successful alignment. |

**Profile Reading:** The following codes allow access to the flat/notch measurement data.

| | | |
|---|---|---|
| 49 | 0 | **Select coarse scan profile.** This will set up the reading of coarse scan profile from memory. Power up default. |
| 50 | 0 | **Select fine scan profile.** This will set up the reading of fine scan profile from memory. |
| 120 | X | **Read image memory.** This command does not have a fixed data length. Data length marked here as an 'X' may have values from 1 to 255. Data length specified with this command will mean memory blocks, each block being 256 bytes. This command primarily used for customized applications where reading of internal scan profile is needed. Before using this command either of the coarse or the fine profiles should be selected with one of the commands described previously. Once selected the same profile data is read until the other is selected. For example when data length is (2) then 512 bytes has to be read. The number of blocks however has to be calculated since the number of data points is variable. At the present time 350 data points are needed for a coarse profile. That means 700 bytes for data points and 6 bytes for header information. From this calculation it is necessary to read at least 3 blocks of data. |

Read Image Memory Format:

Byte No. 1: LSB of number of data points.
Byte No. 2: MSB of number of data points.

First two bytes are the number of data points in the data set which makes the profile selected. Each data point is a measurement of the light intensity of the aligner sensor. When the aligner is rotated wafer goes through the sensor. The amount of light emitted is measured by the sensor. The measurements are taken with a certain intervals. These measurements makes up the data set called the profile.

Byte No. 3: LSB of number of steps between data points.
Byte No. 4: MSB of number of steps between data points.

These next 2 bytes are the number of steps wafer aligner is rotated before the next light measurement is taken. This value is significant for calculating the angle where the flat points are located. The number of steps for the full 360 degree rotation is a fix 60,000 steps. If for example 200 steps are taken between data points, the number of data points for the full circle will be equal to 300 (60,000/200).



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|

Byte No. 5: LSB of data point No. 1.
Byte No. 6: MSB of data point No. 1.

The rest of the bytes read are the light measurements collected at certain intervals described above. Two bytes will make one data point. This collection of the data points will make the profile data set.

## Stepper Motor Control:

### Motion Control:

| | | |
|---|---|---|
| 71 | 0 | **Start motor motions.** It will start motor to reach target position if motor is not already running. It will send busy code while moving the motor if status is requested. |
| 66 | 0 | **Stop motor motions**. If motor is running, it will come to a stop after ramping down. Response to the status request will be a Busy code until a complete stop. Power up default. |

### Position Parameters:

| | | |
|---|---|---|
| 65 | 3 | **Write motor position counter.** This is a 24 bit signed integer in two's complement form. This counter is increased or decreased with every step of the motor. This parameter is loaded only once on power-up and reloaded again if adjustments are needed. |
| 108 | 4 | **Read motor position counter plus the status byte.** The first three bytes are the two's complement signed integer motor position counter. The next byte is the status byte. The counter is updated with every movement of the motor. |
| 97 | 3 | **Read motor position counter.** The motor position counter is a number in two's complement form. The counter is updated with every movement of the motor. |
| 84 | 3 | **Write target position counter.** The target position counter is a number in two's complement form. When a run motor command is executed, the number of steps is equal to motor position counter minus target position counter. |
| 116 | 3 | **Read target position counter.** The target position counter is a 3 byte number in two's complement form. |

### Home alignment:

| | | |
|---|---|---|
| 38 | 0 | **Align with home sensor.** The wafer holder will be aligned with the home sensor. The aligner should be equipped with a sensor. |

### Incremental movements:



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|
| 43 | 0 | **Incremental move upwards.** When this command is executed, the number of steps moved is equal to the number loaded into the increment counter. The direction of the motor is determined by the internal firmware such that contents of the position counter will be increased. |
| 45 | 0 | **Incremental move downwards.** This command is the opposite of the incremental move upwards command where the contents of the position counter will be decreased. |
| 68 | 3 | **Write Increment counter.** This is a two's complement 3 byte number for the increment counter used by incremental movements. The number of steps moved will be equal to the number loaded into the counter with this command. The direction of the motors are determined by using one of the two incremental commands. |
| 100 | 3 | **Read Increment value.** This is a two's complement 3 byte number read from the increment counter. |

**Ramping Time:**

| | | |
|---|---|---|
| 81 | 1 | **Write ramping time.** This is a one byte straight binary number, which will change the ramping time when moving the motor. The smaller the number, the shorter the ramping up or down time. Power up default is set to 5. |
| 113 | 1 | **Read ramping time.** Same as above to read back the value. |

**Motor Speed Write Commands:**

| | | |
|---|---|---|
| 82 | 2 | **Write motor starting speed.** This is a 2 byte straight binary number. This is the speed the motor will start ramp up. If this value is bigger than top speed, then there is no ramping up. The equation below can be used to convert the speed given in pulse per second. |
| 83 | 2 | **Write motor top speed.** This is a 2 byte straight binary number. When motor is moved, this will be the running speed after ramping up. If distance to run is shorter the motor may not reach the top speed programmed. The equation below can be used to convert the speed given in pulse per second. Power up default is set to 60Khz. |

The following equation can be used to convert a given speed in pulse-per-second units to a binary number. This is the number which is used with write starting speed and top speed commands.

binary output value=65536-(5529600/speed)

Speed is the desired speed in pulse-per-second units.

The maximum binary output value = 65534



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|

The minimum binary output value = 1

The maximum speed = 2764800 pulse per second.
The minimum speed = 84.375 pulse per second.

### Motor Speed Read Commands:

| | | |
|---|---|---|
| 114 | 2 | **Read motor starting speed.** This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |
| 115 | 2 | **Read motor top speed.** This is a 2 byte straight binary number. |

The following equation can be used to convert the binary number read back to speed in pulse-per-second units. speed(pulse/sec)=5529600/(65536-binary value)

## Joystick Control:

### Joystick Speed Write Commands:

| | | |
|---|---|---|
| 69 | 1 | **Write joystick normal top speed.** This is a one byte straight binary number. This will affect the joystick top speed when it is deflected to either side without pressing the fast joystick key. |
| 70 | 1 | **Write joystick faster top speed.** This has the same function as above except that it will affect the maximum speed when fast key is pressed with joystick deflected either side. How to calculate these two numbers is explained below. |

The following equation can be used to convert a given speed in pulses per second units to a binary number. This one byte number is used with write joystick speed commands.

binary output value = 1843000/speed
Speed is the desired speed in pulses per second.

The maximum binary output value = 255
The minimum binary output value = 1

### Joystick Speed Read Commands:

| | | |
|---|---|---|
| 101 | 1 | **Read joystick normal top speed.** This is a one byte straight binary number. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|
| 102 | 1 | **Read joystick faster top speed.** This is a one byte straight binary number. |

The following equation can be used to convert the binary number read back, to speed in pulse per second units. speed(pulse/sec) = 1843000/binary value

| | | |
|---|---|---|
| 72 | 0 | **Make joystick fast speed to ramp up.** This command will affect how the joystick fast key will act when pressed with joystick deflected either side. With this command when joystick fast key is pressed, motor will ramp up to faster speed. Power up default. |
| 76 | 0 | **Make joystick fast speed to ramp down**. This is the opposite of the above command. When joystick fast key is pressed, motor will ramp down to a slower speed. |
| 74 | 0 | **Joystick enabled.** To enable the joystick or trackball function when motor is not moving. Power up default. |
| 75 | 0 | Joystick disabled. To disable the joystick or trackball function when motor is not moving. |

**Read Identification:**

| | | |
|---|---|---|
| 105 | 6 | **Read Device identification.** First 5 characters should match the device ID shown at the top of this page. The last (sixth) byte read is a representation of the configuration dip switch. |

The same code may be used consecutively in order to get a version information from device. When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

| | | |
|---|---|---|
| Byte 0 | : | 48 decimal ("0") |
| Byte 1 | : | month of the year in binary form. |
| Byte 2 | : | day of the month in binary form. |
| Byte 3 | : | year MOD 100 in binary form. |
| Byte 4 | : | current version number multiplied by 10 in binary form. |
| Byte 5 | : | Not used. |

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

**Read Version No.:**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|
| 127 | 6 | **Read date and version.** This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented. |

**Request Device Status:**

| | | |
|---|---|---|
| 63 | 0 | **Request Status.** Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.<br><br>66 = If device is busy.<br>98 = If device is not busy. |

Parameters should only be loaded when a device is not busy, but they can be read anytime.

**Configuration:**

There are two (2) DIP switches installed on the board. The switch bank closest to the bus connector is used for device address selection, The other is used for configuration functions.

Each board can be set to respond to an addresses from 0 to 254. This is be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary number. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 | |
|---|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | (Factory default) |

**Function Switch Settings:**

Switch 1-3: Motor resolution select. All open allows resolution selection by computer. (Factory setting – SW1,2,3 always open)

Switch 4: This switch reverses the joystick direction in relation to the motor direction. Factory setting = closed

Switch 5: Mechanical drive select
0 = 60K motor steps per revolution of chuck. (motor=10Kppr X 6:1 gearing)
1 = 20K motor steps per revolution of chuck. (motor=20Kppr direct drive)

Switch 6: Not used

Switch 7: 0 = Bilevel drive. (for low impedance motors)



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name:MAC2002 Flat/Notch aligner control and drive #73002075
Device ID: FFIND

| Code: | Length: | Description |
|---|---|---|

1 = Motor High power always on. (for High impedance motors)

Switch 8: 0 = Non-encoder mode or open loop counting. The position is counted with motor pulses and motor direction. Encoder pulses are not used for position counting even if one is installed.
1 = Encoder mode or closed loop counting. The position is counted with encoder pulses and motor direction. In this mode, the encoder is used to determine the position of the motor. If no encoder is installed, the motor will not run.

**Motor Current Program:**

Current is set according to how many of the 4 switches are closed according to the formula:
**Peak Motor current = 0.2 + 0.2(number of switches closed) Amps.**
ex: all switches open = 0.2 amps or 3 switches closed = 0.8 amps (typical setting)

Typical settings for low impedance motors is 800mA and 200mA for high impedance motors.

**Hardware compatibility:** The 73002075 is comparable to the MAC 2000 flat finder two card set; 73000805/73000503 but has the following requirements.

- The '2075 must be used in a mac2002 frame (orange) and with a 73-ca02011 harness.
- When used with a laser sensor, 73000920, operation s the same as the '805/'503 combo.
- When using with a lamp type sensor, 73000908**A**, the 'A' version sensor must be used. The '2075 is externally calibrated and the 'A' version sensor has a control built into the connector shell which sets the lamp brightness to achieve a full scale reading.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|
| **Filter Shutter Controller** | | **"EFILS"** |
| **Main Filter Wheel:** | | |
| 72 | 0 | Seek to HOME (Filter No. 1) |
| 70 | 0 | Rotate wheel to next filter |
| 78 | 0 | Rotate wheel to previous filter |
| 49 | 0 | Rotate wheel to Filter No. 1 |
| 50 | 0 | Rotate wheel to Filter No. 2 |
| 51 | 0 | Rotate wheel to Filter No. 3 |
| 52 | 0 | Rotate wheel to Filter No. 4 |
| 53 | 0 | Rotate wheel to Filter No. 5 |
| 54 | 0 | Rotate wheel to Filter No. 6 |
| 97 | 1 | Read filter wheel position. |
| **Auxiliary Filter Wheel:** | | |
| 71 | 0 | Seek to HOME (Filter No. 1) |
| 81 | 0 | Rotate wheel to next filter |
| 82 | 0 | Rotate wheel to previous filter |
| 33 | 0 | Rotate wheel to Filter No. 1 |
| 64 | 0 | Rotate wheel to Filter No. 2 |
| 35 | 0 | Rotate wheel to Filter No. 3 |
| 36 | 0 | Rotate wheel to Filter No. 4 |
| 37 | 0 | Rotate wheel to Filter No. 5 |
| 94 | 0 | Rotate wheel to Filter No. 6 |
| 98 | 1 | Read filter wheel position. |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Dual Filter Wheel Control:**

30 2

Move Wheels Simultaneously. This command should be used if both motors are to be ran together. They do not have to run the same distance, and can be ran opposite directions. The first data byte(LSB) will move the main filter wheel and second byte(MSB) the auxiliary wheel. The following list describes the data byte values. These values are valid for both wheels and given in ASCII character form.

**P** move to previous filter.
**N** move to next filter.

**A numeric value (n)** move to the filter wheel number n.

Example: consider you want to move main filter to next position and auxiliary filter to position 5. The data bytes should have the following values:
Data byte 1 = 'N'
Data byte 2 = '5'

**Read Status:**

115 1

Read status byte. The byte read shows the last shutter positions and timer status. The following table shows each bits functions. Shutter status is correct when used with software commands. Front panel operation will not affect shutter status bits.

| | | | |
|---|---|---|---|
| bit 0 | timer 1 status | 1=on | 0=off |
| bit 1 | timer 2 status | 1=on | 0=off |
| bit 2 | shutter 1 status | 1=open | 0=closed |
| bit 3 | shutter 2 status | 1=open | 0=closed |
| bit 4 | shutter 3 status | 1=open | 0=closed |
| bit 5 | internal use | | |
| bit 6 | front panel switch status | 1=enabled | 0=disabled |
| bit 7 | timer 3 status | 1=on | 0=off |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: | |
|---|---|---|---|
| **Shutter Control:** | | | |
| **Shutter No. 1:** | | | |
| 74 | 0 | Open shutter. | |
| 75 | 0 | Close shutter. | |
| 84 | 2 | Load exposure time for shutter. | |
| Exposure time resolution is 1 millisecond. The range is from 1 to 65,535 milliseconds. | | | |
| 88 | 0 | Expose shutter. | |
| **Shutter No. 2:** | | | |
| 76 | 0 | Open shutter. | |
| 77 | 0 | Close shutter. | |
| 85 | 2 | Load exposure time for shutter. | |
| Exposure time resolution is 1 millisecond. The range is from 1 to 65,535 milliseconds. | | | |
| 89 | 0 | Expose shutter. | |
| **Shutter No. 3:** | | | |
| 79 | 0 | Open shutter. | |
| 80 | 0 | Close shutter. | |
| **Front Panel Control:** | | | |
| 68 | 0 | Disable front panel switches. | |
| 69 | 0 | Enable front panel switches. | (Default at power on) |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Automatic Shutter Control:**

| | | |
|---|---|---|
| 87 | 1 | Write control byte. **On power this byte is cleared to zero.** |

The specified bits of the parameter may be set or cleared to control the different function.

The control byte:

Bit 0: Main shutter 1=Auto mode 0=Manual mode
Bit 1: Aux shutter 1=Auto mode 0=Manual mode

Setting bits 0 and 1 of the control byte sets shutter 1 and 2 to the automatic mode. In manual mode shutters do not change positions while filter wheels are moving. In automatic mode shutters are closed for the duration of the filter change and opened again when filter wheel is stopped.

Bit 2: 1=Auto expose shutter main.
Bit 3: 1=Auto expose shutter aux.

When these bits are set shutters are opened and closed for the specified time at the end of the filter movement.

Bit 4: External trigger level. 1=Active high 0=Active low
This bit will regulate external shutter active trigger level. Set by Switch 3 on power up and overwritten by this code.

Bit 5: Macro sequencer. 1=ON 0=OFF
When this bit is set to 1, external triggers will act as a sequence advancer. The sequence table may be written with a different code and when this bit is set external trigger will advance the table index and move to filter wheels contained in the table pointed by the index.

Bit 6: Sync output active level. 1=Active high 0=Active low
This bit will regulate the active level of sync output. Set by Switch 4 on power up and overwritten by this code.

| | | |
|---|---|---|
| 119 | 1 | Read control byte. The control byte described above is read. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|
| **High Speed Sync Mode:** | | |
| 44 | 4 | Image capturing. After loading the 4 bytes filter wheel will rotate with the specified speed and will pulse the sync output when the filter position and bit positions are matched. Bit positions are specified with byte 3 and byte 4. The break command will stop the rotation at the next wheel position.<br><br>byte 1: speed of main . 0=no rotation 1 to 255 rotation speed.<br>byte 2: speed of aux.<br>byte 3: bit0 for main wheel number 1. bit1 for main wheel number 2 and etc. When bit is set to one and wheel is in position sync output is pulsed.<br>byte 4: same as byte 3 for aux. wheel. |
| **Sequencer**: | | |
| 43 | 6 | Loading of sequence bytes. After loading the six bytes auto sequence mode is initiated. In this mode a internal index will increment automatically when a trigger input is received. The sequence values pointed by the index will be used as the next target wheel positions. A zero value for both wheels will set the index to beginning of table. The high nibble of the byte is the main wheel position and the low nibble is the aux. wheel position.<br><br>For example: byte 1 is 26, byte 2 is 41 and byte 3 is 00. The first trigger will move main wheel to 2 and aux to 6. The index will be incremented. The next trigger will move main to 4 and aux. to 1. Since next byte is 00, index will be set to the value of zero that will cause to same sequence to repeat. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|
| **Speed Control:** | | |
| 90 | 1 | Write speed constant. The wheel rotation speed is programmable. This speed is the top running speed. Smaller the number faster the rotation. The default running speeds are loaded with switch settings. But later may be modified by using this code. The following equation may be used:<br><br>speed is expressed in pulse per second.<br>speed constant=1,000,000 /(speed*54) |
| 122 | 1 | Read speed constant.<br><br>The following equation can be used to convert constant read to speed pulse per second.<br><br>speed=1,000,000 /(speed const * 54) |
| **Request Device Status:** | | |
| 63 | 0 | Request Status. Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.<br><br>66 = If device is busy.<br>98 = If device is not busy. |

Parameters should only be loaded when a device is not busy, but they can be read anytime.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Identification:**

105 6 Read identification from device. First 5 characters read are the device ID **EFILS**. The sixth character is a binary value which reflects the state of the configuration switches according to the following table:

| Msb | | | | | | | Lsb |
|---|---|---|---|---|---|---|---|
| bit7 | bit 6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| sw6 | sw5 | sw4 | sw3 | sw2 | sw1 | X | X |

Switch open = 1
X denotes a don't care bit.

The same code may be used consecutively in order to get a version information from device.
When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

**Read Version No.:**

127 6 Read date and version. This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **MAC2002 Filter Shutter Controller 73002080**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Configuration:**

There two (2) DIP switches installed on the board. One with eight (8) switches is used for device address selection, and one with six (6) switches is used to control different functions.

Each device can be configured to have addresses from 0 to 254. This can be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 |
|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**Function Switch Settings:** Factory setting

| | | |
|---|---|---|
| Switch 1: | Motor speed. LSB | Closed |
| Switch 2: | Motor speed. MSB | Closed |
| Switch 3: | Power up shutter ext trigger input level. (see code=87) may be overriden by software control. | Closed |
| Switch 4: | Power up sync output true level. closed=pulse low may be overridden by software. | Closed |
| Switch 5: | Filter wheel type select 0 = BioPoint 1 = MAC2000 | X |
| Switch 6: | 0 = parallel mode.<br>1 = serial mode. | Open |

**Hardware compatibility:** The 73002080 filter wheel control card is made to work with the MAC2002 controller. It can not be directly retrofitted to a MAC2000 frame. The rear panel harness and motherboard layout is different. In order to use this card in a MAC200 frame, the filter wheel harness(es) must be exchanged and pin 17 of U6 must be cut. Contact LEP Tech service for additional information.

**Version 4.2** HOME routine is modified to find the center of the sensor rather than the edge. This will be more immune to effects of magnet strength and temperature.

**32mm wheel** Higher inertia of 32mm filter wheel needs lower start speed. 32mm Mac2 wheel speed should be selected by S2=open.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

## Filter Shutter Controller "EFILS"

**Main Filter Wheel:**

| | | |
|---|---|---|
| 72 | 0 | Seek to HOME (Filter No. 1) |
| 70 | 0 | Rotate wheel to next filter |
| 78 | 0 | Rotate wheel to previous filter |
| 49 | 0 | Rotate wheel to Filter No. 1 |
| 50 | 0 | Rotate wheel to Filter No. 2 |
| 51 | 0 | Rotate wheel to Filter No. 3 |
| 52 | 0 | Rotate wheel to Filter No. 4 |
| 53 | 0 | Rotate wheel to Filter No. 5 |
| 54 | 0 | Rotate wheel to Filter No. 6 |
| 97 | 1 | Read filter wheel position. |

**Auxiliary Filter Wheel:**

| | | |
|---|---|---|
| 71 | 0 | Seek to HOME (Filter No. 1) |
| 81 | 0 | Rotate wheel to next filter |
| 82 | 0 | Rotate wheel to previous filter |
| 33 | 0 | Rotate wheel to Filter No. 1 |
| 64 | 0 | Rotate wheel to Filter No. 2 |
| 35 | 0 | Rotate wheel to Filter No. 3 |
| 36 | 0 | Rotate wheel to Filter No. 4 |
| 37 | 0 | Rotate wheel to Filter No. 5 |
| 94 | 0 | Rotate wheel to Filter No. 6 |
| 98 | 1 | Read filter wheel position. |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Dual Filter Wheel Control:**

| | | |
|---|---|---|
| 30 | 2 | Move Wheels Simultaneously. This command should be used if both motors are to be ran together. They do not have to run the same distance, and can be ran opposite directions. The first data byte(LSB) will move the main filter wheel and second byte(MSB) the auxiliary wheel. The following list describes the data byte values. These values are valid for both wheels and given in ASCII character form.<br><br>**P** move to previous filter.<br>**N** move to next filter.<br><br>**A numeric value (n)** move to the filter wheel number n.<br><br>Example: consider you want to move main filter to next position and auxiliary filter to position 5. The data bytes should have the following values:<br>Data byte 1 = 'N'<br>Data byte 2 = '5' |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: | |
|---|---|---|---|
| **Shutter Control:** | | | |
| **Shutter No. 1:** | | | |
| 74 | 0 | Open shutter. | |
| 75 | 0 | Close shutter. | |
| 84 | 2 | Load exposure time for shutter. | |

Exposure time resolution is 1 millisecond. The range is from 1 to 65,535 mllisecond.

| | | | |
|---|---|---|---|
| 88 | 0 | Expose shutter. | |
| **Shutter No. 2:** | | | |
| 76 | 0 | Open shutter. | |
| 77 | 0 | Close shutter. | |
| 85 | 2 | Load exposure time for shutter. | |

Exposure time resolution is 1 millisecond. The range is from 1 to 65,535 mllisecond.

| | | | |
|---|---|---|---|
| 89 | 0 | Expose shutter. | |
| **Shutter No. 3:** | | | |
| 79 | 0 | Open shutter. | |
| 80 | 0 | Close shutter. | |
| **Front Panel Control:** | | | |
| 68 | 0 | Disable front panel switches. | |
| 69 | 0 | Enable front panel switches. | (Default at power on) |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Status:**

115 1

Read status byte. The byte read shows the last shutter positions and timer status. The following table shows each bits functions. Shutter status is correct when used with software commands. Front panel operation will not affect shutter status bits.

| | | | |
|---|---|---|---|
| bit 0 | timer 1 status | 1=on | 0=off |
| bit 1 | timer 2 status | 1=on | 0=off |
| bit 2 | shutter 1 status | 1=open | 0=closed |
| bit 3 | shutter 2 status | 1=open | 0=closed |
| bit 4 | shutter 3 status | 1=open | 0=closed |

**Automatic Shutter Control:**

87 1

Write control byte. Shutter 1 and 2 can be set to different modes of operation with this command. In manual mode shutters do not change positions while filter wheels are moving. In automatic mode the shutter is closed whilst the filter wheel rotates to a new position and then reopens when move is completed. If both automatic mode and automatic exposure are enabled then shutters opens after a filter change for a predefined time set by commands 84 and 85 and then closed.
The macro mode bit enables the use of the inputs lines to sequence through a series of filter positions, see command 43.

| The control byte: | | --Defaults-- |
|---|---|---|
| Bit 0: Main shutter | 1=Auto mode | 0=Manual mode |
| Bit 1: Aux shutter | 1=Auto mode | 0=Manual mode |
| Bit 2: Auto Exposure | 1=Auto Exposure | 0=Manual Exposure |
| Bit 3: Active Level | 1=Active High | 0=Active Low |
| Bit 4: Macro Mode | 1=Macro Enabled | 0=Macro Disabled |

119 1

Read control byte. The control byte described above is read.

**Speed Control:**

90 1

Write DC speed control. The speed control is only valid for DC type motors. On power up default DC motor is run full speed. Valid range is from 1-127. Uints are in Quadrature Counts / sample time.
Speed value = Velocity(rpm) * Quadrature counts * Sample Time(secs)*0.01667

122 1

Read DC speed control.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|
| 43 | 6 | Setup Macro command. Allows the user to enter a six positions macro sequence for both main and aux filter wheels. Each byte represents a new filter wheel move. Format is as follow: The most significant nibble controls the Main wheel and least significant nibble controls the Aux wheel. A value of one through six is a position move and a zero is a no move. Unused sequence steps must be set to zero. In Macro mode Input #2 is redefined as follows. Inputs # 2 becomes the hardware trigger of the Macro mode sequencer. The active level is controlled by command 87. Each time the Input #2 line is triggered the filter wheels goes to the next position in the sequence. If Macro Mode is disabled then Input #2 behaves like Input #1 where it behaves as a hardware trigger for the Main and Aux shutters. |
| 55 | 10 | Load DC controller parameter values. PRILIMINARY<br>Data format:<br>Byte 1: Idle Gain. Range 0-255.<br>Byte 2: Dynamic Gain. Range 0-255.<br>Byte 3: DSP Filter Zero. Range 0-255.<br>Byte 4: DSP Filter Pole. Range 0-255.<br>Byte 5: DSP Filter Acceleration LSB in rps/s.<br>Byte 6: DSP Filter Acceleration MSB in rps/s.<br>Byte 7: DSP Sample Time. Value=(Sample in secs X 125000)$^{(-1)}$ |
| 154 | 10 | Read DC controller values. Format same as above. |


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Read Identification:**

105 6

Read identification from device. First 5 characters read are the device ID **EFILS**. The sixth character is a binary value which reflects the state of the configuration switches according to the following table:

| Msb | | | | | | | Lsb |
|---|---|---|---|---|---|---|---|
| bit7 | bit 6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| sw6 | sw5 | sw4 | sw3 | sw2 | sw1 | X | X |

Switch open = 1
X denotes a don't care bit.

The same code may be used consecutively in order to get a version information from device.
When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".

**Read Version No.:**

127 6

Read date and version. This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented.

**Request Device Status:**

63 0

Request Status. Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.

66 = If device is busy.
98 = If device is not busy.

Parameters should only be loaded when a device is not busy, but they can be read anytime.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Device Name: **Filter Shutter Controller**
Device ID: **EFILS**

| Code: | Data Length: | Description: |
|---|---|---|

**Configuration:**

There two (2) DIP switches installed on the board. One with eight (8) switches is used for device address selection, and one with six (6) switches is used to control different functions.

Each device can be configured to have addresses from 0 to 254. This can be done with the first DIP switch next to the edge connector. The address is straight 8-bit binary. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | s8 | s7 | s6 | s5 | s4 | s3 | s2 | s1 |
|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

**Function Switch Settings:**

Switch 1: 0 = Stepping motors 1 = DC motors

switch 2: Stepping motor speed. 0 = fast 1 = slow

Switch 3: Reverses the main filter rotation direction.

Switch 4: Reverses the auxiliary filter rotation direction.

Switch 5:
- 0 = multiboard interface mode. Select when serial interface is done with RS 232 board. (See also Switch 6)
- 1 = Single board interface mode. Select when serial interface is done to the individual boards. When in this mode the serial communication is set to 9600 baud rate, no parity, 8 bit data and 2 stop bits. (See also Switch 6)

Switch 6:
- 0 = parallel mode. Switch 5 setting is ignored.
- 1 = serial mode.


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

The MAC2002 AutoFocus consists of a two-card set, using a 73002050 Microstep motor controller/driver in conjunction with the 73002085 AutoFocus processor module. Backward compatibility with MAC2000 is achieved by setting the focus processor to address 11 and the motor controller/drive to address 3. All commands are processed by the focus processor and, if needed, transparently sent to the motor controller/drive. Normally, there is no reason to separately address the Z motor controller/drive.
Note: The 73002050 module must have a PC board revision of D or higher.
Codes marked with an asterisk (*) are new to the mac2002.

## Focus Controls:

| | | |
|---|---|---|
| 32 | 0 | Start focusing. Front panel magnification switch position will control the focus range while focusing. It is a software substitute for the manual focus switch on the front panel. |
| 33 | 0 | Start focusing with *HIGH* magnification. Using this mode, the front panel magnification switch is ignored. |
| 34 | 0 | Start focusing with *Medium* magnification. Using this mode, the front panel magnification switch is ignored. |
| 35 | 0 | Start focusing with *Low* magnification. Using this mode, the front panel magnification switch is ignored. |
| 73 * | 1 | Set focus mode<br>0= step and measure mode (similar to MAC2000)<br>1= continuous profile mode.<br>Power up default is read from configuration SW1. |
| 85 | 0 | Enable front panel manual focus switch. Default setting on power up. |
| 86 | 0 | Disable front panel manual focus switch. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

## Window Commands:

| | | |
|---|---|---|
| 90 | 0 | Turn window off. |
| 87 | 0 | Turn window on. Default setting on power up. |
| 55* | 5 | Set Local Window Size. 8bit.<br>Syntax: 55 5 WINn Vo Vw Ho Hw where:<br>WINn= 1,2 or 3 for desired window.<br>Vo= Vertical Offset minimum=10<br>Vw= Vertical Width minimum= 0<br>Ho= Horizontal Offset minimum=10<br>Hw= Horizontal Width minimum= 0<br>Sum of Vo+ Vw should be less than 262 for RS170.<br>Sum of Ho+ Hw should be less than 229 for RS170. |
| 25* | 9 | Set Local Window Size 16 bit.<br>Syntax: 55 5 WINn Vo Vw Ho Hw where:<br>WINn= 1,2 or 3 for desired window. (1Byte)<br>Vo= Vertical Offset (2Byte) minimum=10<br>Vw= Vertical Width (2Byte) minimum= 0<br>Ho= Horizontal Offset (2Byte) minimum=10<br>Hw= Horizontal Width (2Byte) minimum= 0 |
| 56* | 2 | Set Window ON / OFF. Syntax: 56 2 WINn Mode where:<br>WINn= 1,2 or 3 for desired window.<br>Mode= 0 or 1 for **off** or **on** respectively. |
| 57* | 2 | Set Window Color. Syntax: 57 2 WINn Mode where:<br>WINn= 1,2 or 3 for desired window.<br>Mode= 0 or 1 for **black** or **white** respectively. |
| 59* | 1 | Set Window Border Visable. Syntax: 59 1 Mode where:<br>Mode= 0 or 1 for **off** or **on** respectively. This command effects all active windows. |
| 50* | 1 | Set Window Fill Attribute. Syntax: 50 1 Mode where:<br>Mode= 0 or 1 for **no fill** or **fill** respectively. Fill color is set by code 57. |
| 135* | 12 | Read Three Window Sizes from local. 8Bit. Data is read back as:<br>byte 1 Window AVo byte 7 Window BHo<br>byte 2 Window AVw byte 8 Window BHw<br>byte 3 Window AHo byte 9 Window CVo<br>byte 4 Window AHw byte 10 Window CVw<br>byte 5 Window BVo byte 11 Window CHo<br>byte 6 Window BVw byte 12 Window CHw |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|
| 145* | 24 | Read Three Window Sizes from local.16Bit. Data is read back as: |

| | | | |
|---|---|---|---|
| byte 1 | Window Avo LSB | byte 13 | Window Bho LSB |
| byte 2 | Window Avo MSB | byte 14 | Window Bho MSB |
| byte 3 | Window Avw LSB | byte 15 | Window BHw LSB |
| byte 4 | Window Avw MSB | byte 16 | Window BHw MSB |
| byte 5 | Window Aho LSB | byte 17 | Window Cvo LSB |
| byte 6 | Window Aho MSB | byte 18 | Window Cvo MSB |
| byte 7 | Window Ahw LSB | byte 19 | Window CVw LSB |
| byte 8 | Window Ahw MSB | byte 20 | Window CVw MSB |
| byte 9 | Window Bvo LSB | byte 21 | Window Cho LSB |
| byte 10 | Window Bvo MSB | byte 22 | Window Cho MSB |
| byte 11 | Window BVw LSB | byte 23 | Window CHw LSB |
| byte 12 | Window BVw MSB | byte 24 | Window CHw MSB |

**Focus Parameters:**

| | | |
|---|---|---|
| 79 | 12 | Load coarse and fine profile distances between measurements in motor steps. Focusing is accomplished by collecting a set of light intensity data received by the camera. The focus position is where a maximum value is read. The first set of data collected is with a fine stepping. If a valid peak is not detected then a wider stepping is used. The number of steps for fine and coarse sweeping may be programmed as shown below. *Used in step and measure mode.* |

byte 1 LSB of Coarse scan for MEDIUM range.
byte 2 MSB of Coarse scan for MEDIUM range.

byte 3 LSB of Coarse scan for HIGH range.
byte 4 MSB of Coarse scan for HIGH range.

byte 5 LSB of Coarse scan for LOW range.
byte 6 MSB of Coarse scan for LOW range.

byte 7 LSB of Fine scan for MEDIUM range.
byte 8 MSB of Fine scan for MEDIUM range.

byte 9 LSB of Fine scan for HIGH range.
byte 10 MSB of Fine scan for HIGH range.

byte 11 LSB of Fine scan for LOW range.
byte 12 MSB of Fine scan for LOW range.

| | | |
|---|---|---|
| 111 | 12 | Read coarse and fine profile distances between measurements in motor steps. Same format as loading. *Used for step and measure mode.* |

Default coarse and fine values:

Coarse for MEDIUM : 192 steps
Coarse for HIGH : 48 steps



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

Coarse for LOW : 700 steps

Fine for MEDIUM : 32 steps
Fine for HIGH : 8 steps
Fine for LOW : 128 steps

These values always represent number of motor steps whether open or closed loop.

| | | |
|---|---|---|
| 93 | 1 | Load number of data points in the set taken when focusing. This value times the increment per measurement is the focus search distance. Default value is 35 and the range is from 1 to 255. |
| 125 | 1 | Read number of data points in the set. Reads the number written by command above. |
| 67 | 1 | Load number of frames to count before taking an intensity measurement. Each frame count is 16.6 millisecond. By increasing this value, delays are added between measurements to assure a more stable scan. This value should be increased as magnification increases. A typical value is between 1 and 10. Default is 1 and range is from 1 to 250. |
| 99 | 1 | Read number of frames to count before taking intensity measurements. |
| 80 | 2 | Focus scan speed. Used in discrete mode only. This is the scan speed when focusing, which is different from motor speed. With focusing commands, two different speeds are used. Focus scan speed will be used when searching for focus point, and motor speed will be used to position the motors to beginning of scan. For application of the proper equation please see "Motor Speed Write Commands" of this section on page 6. Also note the maximum focus scan speed is limited to 33khz for the discrete focus algorithm. Default focus scan speed is 20khz. |
| 112 | 2 | Read focus scan speed. |
| 122 | 3 | Read focus light intensity. The value is unsigned straight 3 byte binary number. The number is proportional to the light intensity. When properly focused this value is at peak value. |
| 130 | 6 | Read Three Window Contrast Values. Data is read back as:<br>byte 1 Window A lsb<br>byte 2 Window A msb<br>byte 3 Window B lsb<br>byte 4 Window B msb<br>byte 5 Window C lsb<br>byte 6 Window C msb |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|
| 126 | 1 | Read focus status byte. This byte is a collection of focus status bits. The following table represents the status byte configuration: |

| Bit #/7 | Meaning when High(1) | Meaning when Low (0) | When Valid |
|---|---|---|---|
| 0/7 | Focus NOT Found | Focus Found | Valid after focus command goes not busy |
| 1/7 | Focus Found | Focus NOT Found | Valid after focus command goes not busy |
| 2/7 | Contrast Data Found | No Contrast Data Found | Valid after focus command goes not busy |
| 3-6/7 | | | |
| 7/7 | System BUSY | System NOT Busy | Valid always |

**Continuous mode focus ranges:**

Position range is a 4 byte integer and represents open loop motor steps. Focus speed is a 2 byte integer in Garbis units. (see motor speed command). Valid for firmware ver 0.3 and up.

| | | |
|---|---|---|
| 52* | 6 | Write *LOW* Search Range and Speed in *continuous mode.* Syntax: 52 6 [4 byte position range] [2 byte Speed value]. |
| 53* | 6 | Write *MED* Search Range and Speed in c*ontinuous mode.* Syntax: 53 6 [4 byte position range] [2 byte Speed value]. |
| 54* | 6 | Write *HIGH* Search Range and Speed in *continuous mode.* Syntax: 54 6 [4 byte position range] [2 byte Speed value]. |
| 137* | 6 | Read *LOW* Search Range and Speed in *continuous mode*. Returned Syntax: [ 4 bytes position range ] [2 byte Speed] Default Low Range is 37500. Default Low Speed is 45000Hz. |
| 138* | 6 | Reads *MED* Search Range and Speed in *continuous mode.* Returned Syntax: [ 4 bytes position range ] [2 byte Speed] Default Med Range is 15000. Default Med Speed is 15000Hz. |
| 139* | 6 | Read *HIGH* Search Range and Speed in *continuous mode*. Returned Syntax: [ 4 bytes position range ] [2 byte Speed] Default High Range is 5000. Default High Speed is 1000Hz. |

**Focus Options:**

| | | |
|---|---|---|
| 51* | 1 | Anti-backlash correction enable bit. A nonzero parameter enables the anti-backlash correction, a parameter of zero disables it. The anti-backlash correction algorithm will approach the focus position from the same side it was scanned. Thereby canceling the mechanical backlash of the system. Anti-backlash correction requires one extra move, therefore requires more time to find focus. Default value is OFF. This option is available in both discrete and continuous modes. |

**Analog output:**



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|
| 62* | 3 | Output from four cannel, 12 bit DAC. Command sequence is as follows: 62 <channel> <data><br>Where channel is a byte with value of 0 to 3.<br>Data is a 2 byte straight binary value where the most significant 12 bits are used (LSB first).<br>Channel 0 is the dedicated output for the Piezo focus. Channels 1-3 are not presently defined.<br>Full scale output is presently 0 to +10.00V. |

**Motion Control:**

| | | |
|---|---|---|
| 71 | 0 | Start motor moving. It will start motor to reach target position if motor is not running already. It will send busy code while moving the motor if status is requested. |
| 66 | 0 | Stop motor moving. If motor is running, it will come to a stop after ramping down. It will respond with a not busy code after a complete stop if status is requested. Power up default. |
| 60 | 0 | Motor power on. Motor power should be turned on before running motors. Power up default. |
| 61 | 0 | Motor power off. To turn the power off to the motor phases. Motor will not run if motor power is turned off with this command. |

**Position Parameters:**

| | | |
|---|---|---|
| 65 | 3 | Write motor position counter. This is a two's complement 3 byte number. This parameter is loaded only once on power-up and reloaded again if adjustments are needed. |
| 108 | 4 | Read motor position counter with status byte. The first three bytes is the two's complement motor position counter. The next byte is the status byte. The counter is updated with every movement of the motor.<br><br>The following is the status byte configuration:<br><br>bit 7 = 0 if cw end-limit is closed.<br>bit 6 = 0 if ccw end-limit is closed.<br><br>Front panel select position:<br><br>Low Medium High<br>Bit 4 = 1 1 0<br>Bit 5 = 0 1 1<br><br>bit 3 = 0 joystick disabled<br>bit 2 = 0 motor power is off<br>bit 1 = 0 servo off<br>bit 0 = 0 motor not running |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|
| 97 | 3 | Read motor position counter. This is the two's complement motor position counter. The counter is updated with every movement of the motor. |
| 84 | 3 | Write target position counter. This is a two's complement 3 byte number. This parameter should be loaded before Start Motor command. (see "Motion Control" on page 4 ). |
| 116 | 3 | Read target position counter. This is a two's complement 3 byte number for the target counter. |

**Incremental movements:**

| | | |
|---|---|---|
| 43 | 0 | Incremental move upwards. When this command is executed, the number of steps moved is equal to the number loaded into the increment counter. The direction of the motor is determined by the internal firmware such that position counter will be incrementing. |
| 45 | 0 | Incremental move downwards. This command is the opposite of the incremental move upwards command where the position counters will be decreasing. |
| 68 | 3 | Write Increment counter. This is a two's complement 3 byte number for the increment counter used by incremental movements. The number of steps moved will be equal to the number loaded into the counter with this command. The direction of the motors are determined by using one of the two incremental commands. |
| 100 | 3 | Read Increment value. This is a two's complement 3 byte number read from the increment counter. |
| 46 | 3 | Incremental up and down moves between encoder pulses. This command is only significant if motor is used with the encoder option. The motor will move the amount specified by the parameter. In this case, the amount is expressed in motor steps, not encoder steps. The parameter specified is in two's complement 3 byte number. The motor may be moved forward or backward, depending on the sign of the parameter. The movement is relative to the present position. This command is similar to the incremental moves described earlier with a few differences. First, increment value is included within the command. Then the motor is started and stopped without any acceleration or deceleration. This command also uses the programmed run speed with a lesser maximum speed.<br><br>The maximum speed = 25000 pulse per second.<br>The minimum speed = 84 pulse per second.<br><br>*The most important difference is that this command will move the motor the same amount of steps regardless of whether or not encoders are installed. When used with an encoder installed, the* |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|
| | | *motor can be positioned between encoder pulses regardless of the resolution of the encoders*.<br>*The position read after using this command will not change until the motor is moved to the full encoder step.*<br><br>For example :<br>Suppose we have an encoder with five (5) motor steps per one (1) encoder pulse. If you want to move the motor only three (3) motor steps forward, use this command with the parameter being equal to three (3). |
| 92 | 1 | Load encoder to motor steps ratio. If closed loop mode is selected, on power up encoder to motor steps ratio is determined by internal software. This ratio is described as the number of motor pulses required per encoder pulses received through the closed loop. The value is an unsigned binary number. Default is determined automatically on power up by moving motor 1000 steps down and up then reading position. |
| 124 | 1 | Read encoder to motor steps ratio. The value is an unsigned 1 byte binary number. |
| **Ramping Time:** | | |
| 81 | 1 | Write ramping time. This is a one byte straight binary number, which will change the acceleration value. It is related to the time where speed is changed. Smaller values are equal to bigger accelerations. Power up default is 5. |
| 113 | 1 | Read ramping time. Same as above to read back the value. |
| **Motor Speed Write Commands:** | | |
| 82 | 2 | Write motor starting speed. This is a 2 byte straight binary number. This is the speed the motor will start ramp up. If this value is bigger than top speed, then there is no ramping up. The equation below can be used to convert the speed given in pulse per second. |
| 83 | 2 | Write motor top speed. This is a 2 byte straight binary number. When motor is moved, this will be the running speed after ramping up. The equation below can be used to convert the speed given in pulse per second. |
| | | The following equation can be used to convert a given speed in pulse per second units to a binary number. This is the number which is used with write starting speed and write top speed commands. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

CODE: LENGTH: DESCRIPTION:

binary output value = 65536-(5529600/speed)

Speed is the desired speed in pulse per second units.

The maximum binary output value = 65534
The minimum binary output value = 1

The maximum speed = 2764800 pulse per second.
The minimum speed = 84.375 pulse per second.

**Motor Speed Read Commands:**

| | | |
|---|---|---|
| 114 | 2 | Read motor starting speed. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |
| 115 | 2 | Read motor top speed. This is a 2 byte straight binary number. The equation below can be used to convert the binary number. |

The following equation can be used to convert the binary number read back to speed in pulse-per-second units.

speed(pulse/sec) = 5529600/(65536-binary value)

**Joystick Speed Write Commands:**

| | | |
|---|---|---|
| 69 | 1 | Write joystick normal top speed. This is a one byte straight binary number. This will affect the joystick top speed when it is deflected to either side without pressing the fast joystick key. |
| 70 | 1 | Write joystick faster top speed. This has the same function as above except that it will affect the maximum speed when fast key is pressed with joy-stick deflected either side. How to calculate these two numbers is explained at the end of this manual. |

The following equation can be used to convert a given speed in pulse-per-second units to a binary number. This is the one byte number which is used with write joystick speeds.

binary output value = 1843000/speed

Speed is the desired speed in pulse-per-second units.

The maximum binary output value = 255
The minimum binary output value = 1



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

## Joystick Speed Read Commands:

| | | |
|---|---|---|
| 101 | 1 | Read joystick normal top speed. This is a one byte straight binary number. |
| 102 | 1 | Read joystick faster top speed. This is a one byte straight binary number. |

The following equation can be used to convert the binary number read back to speed in pulse-per-second units.

speed(pulse/sec) = 1843000/binary value

## Joystick Control:

| | | |
|---|---|---|
| 74 | 0 | Joystick enabled. To enable the joystick or trackball function when motor is not moving. Power up default. |
| 75 | 0 | Joystick disabled. To disable the joystick or trackball function when motor is not moving. |
| 72 | 0 | Make joystick fast speed to accelerate. This command will affect how the joystick fast key will act when pressed with joystick deflected either side. When joystick fast key is pressed, motor will accelerate. Power up default . |
| 76 | 0 | Make joystick fast speed to decelerate. This is the opposite of the above command. When joystick fast key is pressed, motor will slow down. |

## Servo Control:

| | | |
|---|---|---|
| 77 | 0 | Turn servo checking on. This command is used when encoders are installed. In a closed loop system feedback is received from encoders. If no movement is detected, motor driver will stop the pulses to the motor and retry by accelerating from starting speed until it reaches the target. This mode with joystick disabled and in idle state will keep the motor in the same position if it is moved externally. Power up default . |
| 78 | 0 | Turn servo checking off. This command cancels the above one. |
| 91 | 3 | Load servo activation distance. Programmed in units of encoder steps, this is the minimum number of steps the motor should skip before internal software will start moving it back to the original position. Default value is 2. |
| 128 | 3 | Read servo activation distance. |



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

## Read Identification:

105 6 Read identification from device. First 5 characters are the device ID EAFC. The sixth byte is a binary representation of the configuration dip switch as shown in the following table.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | Sw5 | Sw4 | Sw3 | Sw2 | Sw1 |

The same code may be used consecutively in order to get a version information from device.

When first used, the reply will contain the module ID in the code manuals given for each module. When the same code is immediately repeated, the following information is sent back to host:

Byte 0 : 48 decimal ("0")
Byte 1 : month of the year in binary form.
Byte 2 : day of the month in binary form.
Byte 3 : year MOD 100 in binary form.
Byte 4 : current version number multiplied by 10 in binary form.
Byte 5 : Not used.

In order to use version information this command should be used twice without any other commands in between. If other commands are used after the first read id command the next read id command will reply with module identification.

It could also be determined that the reply with version string procedure is not yet implemented when the first character of the expected version string is not a "0".



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

| CODE: | LENGTH: | DESCRIPTION: |
|---|---|---|

## Read Version No.:

| | | |
|---|---|---|
| 127 | 6 | Read date and version. This code will read date and version information explained above. Where read id twice scheme explained above may be used with all versions, this code may be used with newer versions where these code is implemented. |

## Request Device Status:

| | | |
|---|---|---|
| 63 | 0 | Request Status. Request whether or not device is busy. One of the following codes is sent back to the host depending on the status of device.<br><br>66 = If device is busy.<br>98 = If device is not busy. |

Parameters should only be loaded when a device is not busy, but they can be read anytime.

**Notes on new Continuous focus algorithm:** (AN104)

### Abstract

73002085 Focus firmware version 0.3 and above have the new focus search algorithms. The new algorithms are continuous in motion as opposed to the discrete or step focus algorithm of the past. The 73002085 focus board requires a 73002050 stepper motor controller/driver with a firmware release of 8.4 or above.

### Advantages

The advantage of the continuous algorithms are; lower total search time, faster search speeds and larger search range. The new algorithm scans the video while moving, as soon as it sees a peak it stops. It then does a fine scan, stop again and goes to the focus position if found, else to start position.
Use of the continuous algorithm requires a fast response camera.

### Compatibility

The system is backwards compatible with the discrete system. A dip switch is used to select between the focus mode. The dip switch is read on boot up once. The focus mode can also be overridden via the serial port



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**73002085 Preliminary** **Version 1.8 05/07/99**

Device Name: **Auto Focus Controller**
Device ID: **EAFC_**

CODE: LENGTH: DESCRIPTION:

**Configuration:**

There are two (2) DIP switches installed on the board. One with eight (8) switches is used for device address selection, and one with six (6) switches is used to control different functions.

The first DIP switch next to the edge connector. The address is straight 8-bit binary. Switch 1 is for the least significant bit, and Switch 8 is for the most significant bit.

Switch positions have the following binary values:

0 = close 1 = open

Example: The following table is switch settings for a given address:

| Address | S8 | S7 | S6 | S5 | S4 | S3 | S2 | S1 |
|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 11 (factory setting) | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |

**Function Switch Settings:**

Switch 1: Fmode * Used to default the focus mode on boot up.
0= discrete mode (step and measure) similar to MAC2000.
1= continuous profile while moving at constant speed.
Mode may be over-written through comport.

Switch 2: UseLimits * Allows the **low** continuous focus algorithm to search from top limit to bottom limit.
Limits must be present for this mode.
The **low** focus range is ignored if enabled.
0= use Low focus range. *Factory default*
1= use end limits.
Note: this switch only overrides the LOW focus range.

Switch 3: Vtype Closed (default) : Single Window Mode.
Open : Triple Window Mode. Note: Requires special hardware.

Switch 4: Meter Closed (default) : focus signal bar graph off.
Open : display focus signal bar graph at top of screen.

Switch 5: Mancontr Closed : Front panel pots enabled
Open : Front panel pots locked out.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

**Artisan Technology Group** is an independent supplier of quality pre-owned equipment

## Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

## We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

## Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP