EZ Servo®
Command set and Communications protocol

Command Set For Servo Model: EZSV10

Document Revision: A40   01/15/2017

The following are Trademarks of AllMotion Inc.:
EZStepper ®
EZServo ®
EasyServo ™
EZBLDC ™
EasyBLDC ™
AllMotion ™

EZ Servo®
Command set and Communications protocol

Overview:
This document describes the operation and command set for the EZServo® series of motor drives.

Simple Text Command syntax:
Simple Text Commands to the EZServo are single alpha characters normally followed by a numeric value. The alpha character represents "what to do" and the numeric value represents "how much to do it".

Simple text commands sent to the drive, set values for desired velocities, accelerations, and positions. Commands can be issued one at a time or sent in a group as a long text string, which allows the setting of all move parameters in one command. Text commands can create loops in the strings and cause the EZ Stepper or EZServo to become a stand-alone device that responds to switch inputs. Finally, storing such text strings into the onboard EEPROM allows the EZServo to power up into a mode of your choice, so that it can act with no computer attached.

The Text Commands are simply typed into a Terminal Program such as "Hyperterminal", no special software or compilation is required. The EZServo can even be commanded from a serial enabled smart phone or tablet.

Command Set for V2.81 Firmware:
(Also see examples on page 10 and new commands for V4.71 Firmware on Page 23)

Command (**Case sensitive**)	Operand	Description
		POSITIONING COMMANDS
A	0-(2^32)	**Move Motor to absolute** (units are quadrature encoder ticks- 32 Bit Positioning)**.**
P	0-(2^32)	**Move Motor relative in positive direction.** (units are quadrature encoder ticks) A value of zero will cause an endless forwards move at speed V. (i.e. enter into Velocity Mode) The Velocity can then be changed on the fly by using the "V" command. An endless move can be terminated by issuing a T
D	0-(2^32)	**Move Motor relative in negative direction.** (units are quadrature encoder ticks) (Note: for a finite move, the ending Absolute position must be greater than zero). A value of zero for the operand will cause an endless backwards move at speed V. (i.e. enter into Velocity Mode). The Velocity can then be changed on the fly by using the V command. An endless move can be terminated by issuing a T
B	0-(2^32)	Sets Distance moved in "Bump Jog Mode" see "n" command.

EZ Servo®
Command set and Communications protocol

		HOMING COMMANDS
Z	**0-(2^32)** (+400)	**Home/Initialize Motor.** Motor will turn toward 0 until the home opto sensor (opto #1) is interrupted. If already interrupted it will back out of the opto and come back in until re-interrupted. Current motor position is set to zero. Homing is done at speed V. The Homing distance is the value given +400. **Eg /1Z100000R** Please see Appendix 2 for details
z	**0-(2^32)**	**Change current position without moving.** Sets both Encoder and Commanded position to value given, without moving the motor. **Eg /1z3000R**
r	**0 or 1** (0)	**Recover Servo to current encoder position** Turns the Servo on or off to current encoder position. This is useful in the case where an overload error has occurred and the servo has automatically shut down, but the encoder count is still valid. Operation may be resumed without re-homing. **Eg /1r1R**
f	**0 or 1** (0)	**Home Flag polarity.** Sets polarity of home sensor, default value is 0. **Eg /1f1R** See Appendix 2 for more details
		SET VELOCITY / ACCELERATION COMMANDS
V	**1- (2^32)** (3,000,000)	**In Position Mode (N1, N2) Set Max Speed of motor** For Servo version with an encoder, sets (encoder ticks/32.768) per second. (Eg Value of 33 will cause the motor to spin at approx 1 encoder tick/second)
V	**1-100** (0)	**In Velocity Only Mode (N0) Set Spin Speed Of Motor** For motors with no encoder, allows Open Loop speed control by setting the voltage to the motor as a percentage 0-100 of the supply voltage. Set m= 100 when using this mode.
L	**0-65000** (100)	Sets Acceleration: Acceleration in encoder ticks/second squared = ((L Value) x (4000000)) / (65536) Eg with V=1000000 and L=10 and "A" set very large at 10^8. Issue command /1V1000000L10A10000000R Wait for motor to reach a steady speed. Type /1T and measure the time to decelerate to a stop. Time to decelerate = V/L = (1000000/32.768) / ((10*4000000)/65536) = 50 seconds
		WAIT COMMAND
M	**0-32000** (0)	Waits for M milliseconds prior to executing the next command.

EZ Servo®
Command set and Communications protocol

		LOOPING AND BRANCHING COMMANDS
g		**Beginning of a repeat loop** marker. See examples below on how to set up a loop.
G	0-30000	**End of a repeat loop** marker. Loops can be nested up to 4 levels. A value of 0 causes the loop to be infinite. (Requires T command to Terminate). If no value is specified 0 is assumed. **Eg /1gA1000A0G5R**
H	01 11 02 12 03 13 04 14	**Halt** current string and wait until condition specified. Wait for low on input 1 (Switch 1) Wait for high on input 1 (Switch 1) Wait for low on input 2 (Switch 2) Wait for high on input 2 (Switch 2) Wait for low on input 3 (Opto 1) Wait for high on input 3 (Opto 1) Wait for low on input 4 (Opto 2) Wait for high on input 4 (Opto 2) If Halted operation can also be resumed by typing /1R Also see "S" command for I/O dependant program execution. If an edge detect is desired, a look for Low and a look for Hi can be placed adjacent to each other eg H01H11 is a rising edge detect. **Eg /1gH01H11A1000A0GR** will do the A1000 A0 combination of commands on every rising edge of swith 1 input.
S	01 11 02 12 03 13 04 14	**Skip next instruction depending on status of switch.** Skip next instruction if low on input 1 (Switch 1) Skip next instruction if high on input 1 (Switch 1) Skip next instruction if low on input 2 (Switch 2) Skip next instruction if high on input 2 (Switch 2) Skip next instruction if low on input 3 (Opto 1) Skip next instruction if high on input 3 (Opto 1) Skip next instruction if low on input 4 (Opto 2) Skip next instruction if high on input 4 (Opto 2) Program branching to a complex subroutine can be implemented by making the next instruction a stored string execution. (See examples). Loops can be escaped by branching to a stored string with no commands. Also see "H" command for I/O dependant program execution.

EZ Servo®
Command set and Communications protocol

			PROGRAM STORAGE AND RECALL
s	0-15		**Stores a program** 0-3 or 0-15 depending on model, Program 0 is executed on power up. (25 full commands max per string). Note: This command takes approx 1 Second to write to the EEPROM.
e	0-15		**Executes Stored Program** 0-15.
			PROGRAM EXECUTION
R			**Run** the command string that is currently in the execution buffer.
X			**Repeat Run** the current command string
			SET MAX MOVE CURRENT / PID CONSTANTS / TORQUE MODE / HOLD CURRENT
m	0-100 (50)		Sets Max current allowed during a move. 100% = 2A peak for EZSV10 .
h	0-50 (0)		Sets Current in Torque mode. 100% = 2A peak for EZSV10 . When a value is written in here the servo automatically enters Torque mode.
u	1-25000 (25000)		**(Future Release) Overload Timeout (Milliseconds)** When the Servo detects an overload condition it will shut down after this number of Milliseconds. Default 25000.
w	0-65530 (1000)		**Set Servo Proportional gain.** (Default value is 1000, and is stable with most motors, when equipped with 400-1000 line encoders – 1600-4000 quadrature ticks). Default is 1000
x	0-65530 (0)		**Set Servo Set Integral gain.** (Default value is 1, and is stable with most motors, when equipped with 400-1000 line encoders – 1600-4000 quadrature ticks). Default is 0
y	0-65530 (2500)		**Set Servo Set Differential gain.** (Default value is 2500, and is stable with most motors, when equipped with 400-100 line encoders – 1600-4000 quadrature ticks). Default is 2500

EZ Servo®
Command set and Communications protocol

		SPECIAL MODES COMMANDS
N	0-2 (1)	**Sets Modes – Interpret as combination of Binary Bits** 0 = No Encoder, Velocity Mode control possible only. 1 = Encoder With No Index (Default). Homes to Opto. 2 = Encoder With Index. Homes to Index.
n	0-2047 (0)	**Sets Modes – Interpret as combination of Binary Bits** **Bit0 (LSB)** /1n1R Enable Pulse Jog Mode. Jog distance is given by "B" command. The Switch Inputs become the Jog Inputs. **Bit1 :** /1n2R Enable Limits. (The two optos become limits switches). The polarity of the limits is set by the "f" command **Bit2 :** /1n4R Enable Continuous Jog Mode. Continuous run of motor while switch is depressed. Note that the jog mode allows moves below zero, which will be interpreted by any subsequent "A" command as a large positive number. If this is undesirable, please use the "z" command to define zero position to be some positive number so that underflow will not occur.
K	0-1023	**Optional Output Pin** Set Optional PWM output value. LED 2 pin can be wired a s a PWM output. The value set here changes a PWM between 0 (always off) and 1023 (always on) and any value in between at a 20KHz Rate. This modification requires swapping two resistors on the board and can be done at the factory or by a skilled user.
b	9600 19200 38400 57600 (9600)	**Adjustable baud rate** Eg /1b19200R This command will usually be stored as program zero and execute on power up. Default baud rate is 9600.
		DEVICE RESPONSE PACKET
		See Appendix 7 for detailed description of device response to commands.
		HARDWARE PROTOCOL
		The EZ Servo communicates over the RS485 bus at 9600 baud, 1 Stop bit, No Parity, No flow control.

EZ Servo®
Command set and Communications protocol

Commands Below are "Immediate" Commands, and cannot be cascaded in strings or stored. These commands execute while others commands are running.

		IMMEDIATE QUERY COMMANDS
		An R at the end is not required for the commands below.
T		Terminate current command or loop. (example: /1T)
?	0	Returns the current Commanded motor position
?	1	Reserved
?	2	Returns the current Slew/Max Speed for Position mode
?	3	Reserved
?	4	Returns the status of all four inputs, 0-15 representing a 4 bit binary pattern. Bit 0 = Switch1 , Bit 1 = Switch2 Bit 2 = Opto 1, Bit 3 = Opto 2
?	5 , 6, 7	Reserved
?	8	Returns Encoder Position. (can be zeroed by "z" command)
?	9	Erases all stored commands in EEPROM.
?	w	Returns Proportional Gain Value
?	x	Returns Integral Gain Value
?	y	Returns Differential Gain Value
?	L	Returns Acceleration Value
?	m	Returns Max Current Value.
?	G	Returns Current Loop counter value.
?	B	Returns the Bump Jog Delta Value.
&		Returns current code version.
Q		Query current status of EZServo Returns the Ready/Busy status as well as any error conditions in the "Status" byte of the return string. **The Return string consists of the start character (/), the master address (0) and the status byte.** Bit 5 of the status byte is set when the EZServo is ready to accept commands. It is cleared when the EZServo is busy. The least significant four bits of the Status byte contain the completion code. The list of the codes is: 0 = No Error 1 = Initialization error 2 = Bad Command 3 = Operand out of range Errors in OpCode will be returned immediately, while Errors in Operand range will be returned only when the next command is issued.

EZ Servo®
Command set and Communications protocol

Examples of a command strings in DT protocol are:
Please first see Appendix 5 and 6 to ensure correct wiring and stability of motor.

Example #1 (A Move to Absolute Position)

/1A12345A0R<CR>

This breaks down to:
1. "/" is the start character. It lets the EZ Servos know that a command is coming in.
2. "**1**" is the device address, (set on address switch on device).
3. "**A12345**" makes the motor turn to **A**bsolute position **12345**
4. "**A0**" makes the motor turn to **A**bsolute position **0**
5. "**R**" Tells the EZ Servo to **R**un the command that it has received.
<CR> is a carriage return (ie hit "Enter" button on the keyboard) that tells the EZ Servo that the command string is complete and should be parsed.

Note: Hyperterminal issues each character as you type it in. Therefore it is not possible to cut and paste in Hyperterminal. Backspace is allowed only upto the address character. If backspace is used, all characters "backspaced" must be retyped in. If a typing error is made, typically hit enter and type it all in again – what was typed in will be overwritten as long as the R command at the end was not present.

Example #2 (Move loop with waits)

 /1gA10000M500A0M500G10R<CR>

This breaks down to:
1. "/" is the start character. It lets the EZ Servos know that a command is coming in.
2. "**1**" is the device address, (set on address switch on device).
3. "**g**" is the start of a repeat loop
4. "**A10000**" makes the motor turn to **A**bsolute position **10000**
5. "**M500**" causes the EZ Servo to wait for **500 M**illiseconds.
6. "**A0**" makes the motor turn to **A**bsolute position **0**.
7. "**M500**" is another wait command for **500 M**illiseconds.
8. "**G10**" will make the string repeat 10 times starting from the location of the small "g"
9. "**R**" Tells the EZ Servo to **R**un the command that it has received.
10. <CR> is a carriage return (ie hit "Enter" button on the keyboard) that tells the EZ Servo that the command string is complete and should be parsed.

To Terminate the above loop while in progress type **/1T**

EZ Servo®
Command set and Communications protocol

Example #3 (Program Storage and Recall)

An example of a storing a command string for later execution:

/1s2gA10000M500A0M500G10R<CR>

The program outlined in the prior example is stored as Program 2

/1e2R<CR>

Will execute the previously stored program #2. (Note: program 0 is always executed on power up, if we use 0 instead of 2 in the above example then this program would execute automatically on power up).

Example #4 (Set Current , Move upon Switch2 closure)

/1s0m50Z20000gH02P1000G10R<CR>

/1s0	Store following program in motor number 1 stored string 0 (string 0 is executed on power up).
m50	Set move current to 50% of max
Z20000	Home to opto with max move of 20000 encoder ticks to find opto.
g	Start a loop
H02	Wait for Zero on Switch2
P1000	Advance 1000 Encoder Ticks
G10	Repeat loop above 10 times.
R	Run

Note the above loop will terminate after the Z command if a Home Flag is not found.
Type /1e0R to execute.
To Terminate the above loop while in progress type **/1T**

EZ Servo®
Command set and Communications protocol

Example #5 (Nested loop example)

/1gA10A1000gA10A100G10G100R<CR>

/1	Talk to motor number 1.
g	Start outer loop
A10	Goto Absolute position 10
A1000	Goto Absolute position 1000.
g	Start inner loop.
A10	Goto Absolute position 10.
A1000	Goto Absolute position 100.
G10	Do inner loop 10 times. (End of Inner Loop)
G100	Do outer loop 100 times. (End of outer loop)
R	Run.

To Terminate the above loop while in progress type **/1T**

Example #6 (Skip / Branch instruction)

/1s0gA0A10000S13e1G0R<CR>
/1s1gA0A1000S03e0G0R<CR>

Two "Programs" are stored in string0 and string1 and the code switches from one
Program to the other depending on the state of input3. In the example given the code
will cycle the motor between position A0 and A10000 if input3 is High and between A0
and A1000 if input 3 is Low.

Stored string 0:

/1	Talk to motor 1
s0	Store following in string0 (executed on power up).
g	Start loop
A0	Goto Absolute position 0
A10000	Goto Absolute position 10000.
S13	Skip next instruction if 1 (hi) on input 3
e1	Jump to string1
G0	End of loop (infinite loop).
R	Run.

EZ Servo®
Command set and Communications protocol

Stored string 1:

/1	Talk to motor 1
s0	Store following in string0 (executed on power up).
g	Start loop
A0	Goto Absolute position 0
A1000	Goto Absolute position 10000.
S03	Skip next instruction if 0 (low) on input 3
e0	Jump to string0
G0	End of loop (infinite loop).
R	Run.

Example #7 (Monitor 4 Switches and execute 4 different Programs depending on which switch input is pushed)

/1s0gS11e1S12e2S13e3S14e4G0R<CR>
/1s1A1000e0R<CR>
/1s2A2000e0R<CR>
/1s3A3000e0R<CR>
/1s4A4000e0R<CR>

Five program strings are stored. Upon power up String 0 automatically executes and loops around sampling the switches one by one, and skipping around the subsequent instruction if it is not depressed. Then for example when Switch1 is depressed stored String 1 is executed, which moves the Servo to position 1000. Execution then returns to Stored String 0, due to the e0 command at the end of the other stored strings. If the switch is still depressed it will jump back to String 1 again, but since it is already at that position there will be no visible motion.

To Terminate the above endless loop type **/1T**
Note that using an "e" command to go to another program is a more of a "GOTO" rather than a "GOSUB" since execution does not automatically return to the original departure point.

Example #8 (Move 1000 Steps forwards on every rising edge of Switch2)

/1gH02H12P1000G0R

The endless loop first waits for a 0 level on switch1 then waits for a "1" level on Input2. Then A relative move of 1000 Steps is issued, and the program returns to the beginning to look for another rising edge.

To Terminate the above endless loop type **/1T**

EZ Servo®
Command set and Communications protocol

Coordinated motion between multiple axes

For the simple case of motors 1-9, the EZ Servos are addressed as /1, /2, etc. as shown above.

Up to 16 motors can be addressed individually or in banks of 2, 4 ,or "All", increasing versatility and ease of programming. Synchronized motion is possible by issuing commands addressed to individual EZ Servos without the "R" (Run) command, which sets up the command without executing it. At the proper time, the "R" command is sent to a bank of motors to start several actions in concert.

Addressing motors 10-16
Use the ASCII characters that are the ones above 1-9, which are
10 = ":" (colon)
11 = ";" (semi colon)
12 = "<" (less than)
13 = "=" (equals)
14 = ">" (greater than)
15 = "?" (question mark)
16 = "@" (at sign) – use setting zero on the address switch for this.

For example /=A1000R moves Servo #13 to position 1000.

Addressing banks of motors:
Global addressing of more than one motor is also possible.
The same command can be issued to a bank, or different commands issued to the motors individually, (minus a "Run" at the end) and then a global "Run" command issued to a bank or to All.

EZ Servo®
Command set and Communications protocol

The Banks of two are:
Motors 1 and 2 : "A"
Motors 3 and 4 : "C"
Motors 5 and 6 : "E"
Motors 7 and 8 : "G"
Motors 9 and 10 : "I"
Motors 11 and 12 : "K"
Motors 13 and 14 : "M"
Motors 15 and 16: "O"
The Banks of four are:
Motors 1,2,3, and 4 : "Q"
Motors 5,6,7, and 8 : "U"
Motors 9,10,11, and 12: "Y"
Motors 13,14,15 and 16 : "]"- (close bracket)

For All motors:
Use the Global address "_"   (underscore).

Example #9 Coordinated Motion with axes doing same motion.

/_A10000R<CR>

/_   (Slash then Underscore) Talk to all 15 Motors.
A1000   Goto Absolute position 1000.
R   Run. All motors will go to Absolute position 1000

Example #10 Coordinated Motion with axes doing different motions

/1A10000<CR>
/2A200<CR>
/AR<CR>

/1A10000<CR> Set up motor 1 command buffer to go to **A**bsolute position **10000.**
/2A200<CR> Set up motor 2 command buffer to go to **A**bsolute position **200.**
/AR   Execute current commands in buffer for **Bank Address "A"** which is
                   motors 1 and 2. (The "A" here is an Address of a Bank of motors 1&2
                   because it comes after the slash and should not be confused with the "A"
                   that means absolute position.) Both moves will start at the same time, and
                   complete at a time determined by the Velocity set for each axis.

EZ Servo®
Command set and Communications protocol

SERVO MOTOR ELECTRICAL SPECIFICATION

The EZ Servo® will work with most Servo motors, however the performance achieved will be a function of the motor used.

Select the following:
Kv: Back EMF constant. Chose a motor such that the back EMF at the max speed required is about half of the supply voltage. This will allow for good controllability at the top speed because some voltage is "left over" for the controller to effect control. Eg use a 12V Motor with a 24V supply. (The EZServo is a "chopper drive" and regulates the current using the "m" command, so the motor will not be damaged.)

L: Motor Inductance: The EZSV10 will work with motors of inductance > 0.1mH.

R: Motor Resistance. In most cases when Kv and L is selected as stated above, this value will be acceptable.

EZ Servo®
Command set and Communications protocol
APPENDIX 2
HOMING ALGORITHM IN DETAIL

The "Z" command is used to initialize the motor to a known position. When issued the Motor will turn toward 0 until the home opto sensor is interrupted. If already interrupted it will back out of the opto and come back in until re-interrupted. Current motor position is set to zero. The Homing is done at a speed set by "V".

The maximum number of steps allowed to go towards home is defined by the Z command operand + 400. The maximum number of steps away from home (while sensor is cut) is 10000 for V2.81 Firmware, (and is set by the Zcommand oprand +400 for V 4.71+ Firmware release). If motor goes away from home, switch the A and B encoder wires and also switch the Motor power leads – This switching of both encoder and power leads will switch the direction the motor considers positive.

To set up home
   (1) Ensure that /1P1000R moves away from the home flag. If it does not then switch the A and B encoder wires and also switch the Motor power leads – This switching of both encoder and power leads will switch the direction the motor considers positive. This is independent of any home flag connection and should be done first prior to doing steps below.
   (2) Issue /1Z10000R , the motor should turn towards home, if it does not then probably the polarity of the home flag is opposite to the default expected value, which is high when at home and low when away from home.
   (3) To home to a home flag which has opposite logic (low when at home and high when away from) home issue /1f1Z10000R

Opto and flag should be set up to be unambiguous, i.e. when motor is all the way at one end of travel, flag should cut the opto, when at other end of travel flag should not cut opto. There should only be one black to white transition possible in the whole range of travel.

If homing to an Index Pulse from the encoder (N2 Mode), then it is necessary to travel slowly at say V1000 or so. (Due to the Narrow nature of the Index Pulse). /1N2f0Z10000R if index is a high pulse or /1N2f1Z10000R if index is a low going pulse.

The "f" command is used to set the polarity of the flags. "f0" (default) expects the home / limits to be low when the device is away from home and high when the device is at its limits / home.

EZ Servo®
Command set and Communications protocol

Most applications require intermittent moving of the motor. In the EZ Servo, the current is increased while a move is performed, and the current is then reduced at the end of the move. The dissipation in the drive is proportional to the current flowing in the drive, and therefore the dissipation occurs primarily during the "move". The one exception to this is the rare instance where the drive has to fight a constant load such as gravity.

When the Drive generates heat, the heat first warms the circuit board and heatfin (if any). Only then does the heat transfer to the surroundings. For intermittent moves that are less than one minute in duration, the Drive primarily cools using this thermal inertia of the board and heatfin, and not by steady state dissipation to the surrounding ambient.

The EZServos are designed to work beyond the voltage and current that they are rated for, however the small size of the boards limit their ability to dissipate heat in steady state. It is recommended that the drive be derated linearly from 100% Duty Operation at 50% current to 25% Operation at 100% of rated current. (Ie if an application requires 100% current during the entire move, moves should only performed about 25% of the time – average over about 5 minutes). Note that setting m=100%, means the drive is allowed to go up to 100% of rated current, not that 100% current is always used. The drive uses whatever current is needed to follow the trajectory and will typically only draw significant current during acceleration or deceleration.

Most applications will not require derating of the drive.

The above duty cycle guidelines are arrived at by the fundamental fact that the drive will work upto a PCB copper temperature of 85C, however for long life keep the temperature should be kept below 55C for most of the time.

EZ Servo®
Command set and Communications protocol

The Protocol described in the majority of this manual is DT (Data Terminal) protocol. There is however a more robust protocol known as OEM protocol that includes checksums. AllMotion Drives work transparently in both protocols. And switch between the protocols depending on the start transmission character seen.

The OEM protocol uses 02 hex (Ctrl B) as the start character and 03 Hex (Ctrl C) as the stop character. The 02 Hex Start Character is equivalent to the / character in DT protocol.

OEM PROTOL EXAMPLE1:
/1A12345R(Enter) in DT Protocol is equivalent to
(CtrlB)11A12345R(Ctrl C)# in OEM protocol

Explanation	Typed	Hex
Start Character	Ctrl B	02
Address	1	31
Sequence	1	31
Command	A	41
Operand	1	31
Operand	2	32
Operand	3	33
Operand	4	34
Operand	5	35
Run	R	52
End Character	Ctrl C	03
Check Sum	#	23

The Check Sum is the binary 8 bit XOR of every character typed from the start character to the end character, including the start and end character. (The Sequence character should be kept at 1 when experimenting for the first time.) Note that there is no need to issue a Carriage return in OEM protocol.

Note that some earlier models require the first command issued after power up to be a DT protocol command. Subsequent commands can be in DT protocol or in OEM protocol. Very early models do not have OEM Protocol implemented at all.

<u>OEM PROTOL EXAMPLE 2:</u>
/1gA1000M500A0M500G10R(Return) in DT Protocol is equivalent to
(CtrlB)11gA1000M500A0M500G10R(CtrlC)C in OEM protocol

The C at the end is Hex 43 which is the checksum (Binary XOR of all preceding Bytes).

<u>Sequence Character:</u>
The Sequence Character comes into effect if a response to a command is not received from the Drive. In this instance the same command can be resent with bit 3 (repeat bit) of the sequence byte set, and bits 0-2 representing the sequence number.

When the repeat bit is set consecutive commands received by the drive must have a different sequence number in order to get executed. (Only the sequence number is looked at – not the command itself)

This covers both possibilities that (a) the Drive didn't receive the command and (b) The Drive received the command but the response was not received.

The sequence number can take the following values.
31-37 without the repeat bit set or 39-3F with the repeat bit set.
(The upper nibble of the sequence byte is always 3.)

EZ Servo®
Command set and Communications protocol

APPENDIX 5   DC MOTOR WIRING

The procedure below describes how to figure out the phasing of the encoder. However, please note that AllMotion can perform this process for no extra charge. Just ship us a motor and we will ship it back fully wired with a board

First ensure that the encoder works by issuing /1?8 while spinning the motor by hand, the numbers should increase/decrease.

Wire the motor and encoder as shown in the wiring diagram. Then issue the command /1A1000R. The motor should move 1000 Encoder ticks and then stop. If the motor spins endlessly, and then shuts off, the motor has been wired with positive feedback on the encoder. Switch the A and B encoder channels and try again.

If it is desired to change the direction the motor considers positive, switch the A and B encoder wires and also switch the Motor power leads – This switching of both encoder and power leads simultaneously will maintains negative feedback while changing the direction the motor considers positive.

EZ Servo®
Command set and Communications protocol

<u>**APPENDIX 6   DC MOTOR TUNING**</u>

Typically (in 99% of cases) the motor will be stable with the default constants that are loaded on power up.

Motor may be unstable with default gain values if a high (>4000 line) encoder is used.
If the motor Vibrates or oscillates on power up, try issuing /1w250y500R to reduce gain.
The motor should be stable but the response will be somewhat slow.
Increasing both P and D values in proportion will "stiffen" the servo.

If zero following error is needed then the Integrator will need to be turned on:
(eg /1x100R) , however the system will become more unstable and harder to compensate when the integrator is turned on. Try to achieve the best possible result with the P and D values only and then turn on just a little I.
Eg /1w700x100y30000R

The use of a current limited lab supply is recommended while tuning the motor. Large currents may be drawn if oscillations occur.

<u>**Motor Overload:**</u>
If motor gives up half way through a move, and gives an overload error (Upper or lower case I when queried with /1Q after error) this is due to the fact that the motor could not keep up with the commanded trajectory. Typically increase the value of the move current "m" to allow the motor to move faster, or increase the supply voltage, or reduce L, V.

EZ Servo®
Command set and Communications protocol
APPENDIX 7
DEVICE RESPONSE PACKET

EZ Servos and EZ Servos respond to commands by sending messages addressed to the "Master Device". The Master Device (which for example is a PC) is assumed always has Address zero. The master device should parse the communications on the bus continuously for responses starting with /0. (Do NOT for example look for the next character coming back after issuing a command because glitches on the bus when the bus reverses direction can sometimes be interpreted as characters)

After the /0 the next is the "Status Character" which is actually a collection of 8 bits.
These Bits are:
Bit7 … Reserved
Bit6 … Always Set
Bit5 … Ready Bit - Set When EZ Servo is ready to accept a command.
Bit4 … Reserved

Bits 3 thru 0 form an error code from 0-15
0 = No Error
1 = InitError
2 = Bad Command (illegal command was sent)
3 = Bad Operand (Out of range operand value)
4 = N/A
5 = Communications Error (Internal communications error)
6 = N/A
7 = Not Initialized (Controller was not initialized before attempting a move)
8 = N/A
9 = Overload Error (Physical system could not keep up with commanded position)
10 = N/A
11 = Move Not Allowed
12 = N/A
13 = N/A
14 = N/A
15 = Command Overflow (unit was already executing a command when another command was received)

Example Initialization Error Response:
Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)
An initialization error has response has 1 in the lower Nibble. So the response is 41 Hex or 61 Hex   which corresponds to ASCII character upper case "A" or lower case "a", depending on if the device is busy or not.

EZ Servo®
Command set and Communications protocol

Example Invalid Command Response:
Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)
An invalid command has response has 2 in the lower Nibble. So the response is 42 Hex
or 62 Hex   which corresponds to ASCII character upper case "B" or lower case "b",
depending on if the device is busy or not.

Example Operand Out of range Response:
Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)
An operand out of range has response has 3 in the lower Nibble. So the response is 43
Hex or 63 Hex   which corresponds to ASCII character upper case "C" or lower case "c",
depending on if the device is busy or not.

Example Overload Error Response:
Note that the Upper Nibble only typically takes on values of 4 or 6 (Hex)
An overload error has response has 7 in the lower Nibble. So the response is 47 Hex or
67 Hex   which corresponds to ASCII character upper case "I" or lower case "i",
depending on if the device is busy or not.

Example Response to command   /1?4

FFh: RS485 line turn around character. It's transmitted at the beginning of a message.
2Fh: ASCII "/"Start character. The DT protocol uses the '/' for this.
30h: ASCII "0" This is the address of the recipient for the message.
         In this case ASCII zero (30h) represents the master controller.
60h: This is the status character (as explained above
31h:
31h: These two bytes are the actual answer in ASCII.
         This is an eleven which represents the status of the four inputs.
         The inputs form a four bit value. The weighting of the bits is:
         Bit 0 = Switch 1
         Bit 1 = Switch 2
         Bit 2 = Opto 1
         Bit 3 = Opto 2
03h: This is the ETX or end of text character. It is at the end of the answer string.
0Dh: This is the carriage return…
0Ah: ...and line feed.

A program that receives these responses must **continuously parse for /0** and take the
response from the bytes that follow /0. The first Character that comes back may be
corrupted due to line turn around transients, and should not be used as a "timing mark".

EZ Servo®
Command set and Communications protocol

.

Command (Case sensitive)	Operand	Description
		POSITIONING COMMANDS
A	0 to +/- (2^31)	**Move Motor to absolute** V4.71 Firmware enhanced to allow negative numbers. eg /1A-1000R
P	0 to +/- (2^31)	**Move Motor relative in positive direction.** V4.71 Firmware enhanced to allow negative numbers eg /1P-1000R will actually go negative not positive
D	0 to +/- (2^31)	**Move Motor relative in negative direction.** V4.71 Firmware enhanced to allow negative numbers eg /1D-1000R will actually go positive not negative
U	1- (2^32) (3,000,000)	**Same as V command for Position Mode except sets velocity in negative direction.** This must be set after setting V, any subsequent setting of V will set both U and V to be the same. Eg /1V1000U3000R

V4.71 Firmware Enhancements continued on next page……

EZ Servo®
Command set and Communications protocol

V4.71 Firmware Enhancements continued……

		HOMING COMMANDS
Z	**0 to +/- (2^31) (400)**	**Home/Initialize Motor.** A new homing mode suitable for rotating axes with an index pulse has been introduced. This mode allows an axis to turn in the positive direction until the index pulse is triggered. (in addition to the normal homing in the negative direction) This mode is made active by placing a negative number in front of the command value Eg /1Z-5000R So the various modes are: /1f0N1Z5000R – Home in Negative direction, to the Opto1 input, that is High when at home. /1f1N1Z5000R – Home in Negative direction, to the Opto1 input, that is Low when at home. /1f0N1Z-5000R – Home in Positive direction, to the Opto1 input, that is High when at home. /1f0N1Z-5000R – Home in Positive direction, to the Opto1 input, that is High when at home. /1f0N2Z5000R – Home in Negative direction, to the Index input, that is High when at home. /1f1N2Z5000R – Home in Negative direction, to the Index input, that is Low when at home. /1f0N2Z-5000R – Home in Positive direction, to the Index input, that is High when at home. /1f0N2Z-5000R – Home in Positive direction, to the Index input, that is High when at home.

EZ Servo®
Command set and Communications protocol

V4.71 Firmware Enhancements continued……

		MISCELLANEOUS COMMANDS
G	XX	**End of a repeat loop** In V4.71 the End of loop can be skipped by using the S command /1ggA1000A0S01GP10000GR Will skip out of the inner loop to the outer loop when switch 1 input is low.
S	XX	**Skip next instruction depending on status of switch.** In V4.71 the End of loop can be skipped by using the S command /1ggA1000A0S01GP10000GR Will skip out of the inner loop to the outer loop when switch 1 input is low.
u	1-25000 (25000)	**Overload Timeout (Milliseconds)** When the Servo detects an overload condition (ie max current set by "m" value is exceeded by feedback calculations) it will shut down after this number of Milliseconds. Default 25000.
n	0-2047 (0)	**Sets Modes – Interpret as combination of Binary Bits** **Bit1 :** /1n2R Enable Limits. In V4.71 Firmware this mode is active for "N0" Velocity mode and "h" torque mode in addition to the .
/1?21 **/1?22** **/1?23** **/1?24**		**Position Capture** As a back ground task, the software trajectory generator position, and encoder position are always captured and stored on a switch 1 or switch 2 closure. These can be read back /1?21 Encoder capture on Switch1 /1?22 Encoder capture on Switch2 /1?23 Trajectory generator capture on Switch1 /1?24 Trajectory generator capture on Switch2

EZ Servo®
Command set and Communications protocol

V4.71 Firmware Enhancements continued……

		ANALOG TO DIGITAL CONVRTER COMMANDS
		All 4 Inputs are ADC inputs and can be read and acted upon by the program.
?aa		Reads back all 4 Input ADC Values **E.g. /1?aa<CR>** The Readback order is channels 4:3:2:1 (/1?aa worked in V2.81 and is included here for completeness, however commands **/1at** and **/1?at** are new to V4.71 Firmware)
at	**100000 to 116368 200000 to 216368 300000 to 316368 400000 to 416368 (6144)**	The "at" command sets the threshold, upon which a "one" or "zero" is called for each of the 4 input channels (as read back by the /1?4 command) The Number represents the channel number followed by a 5 digit number from 00000-16368 which represents the threshold on a scale from a 0-3.3V. The default values are 6144 for all 4 channels which represents 1.24V. 1.24 = 3.3 * (6144/16384) Changing the threshold allows the H and S commands to work on a variable analog input value which essentially allows the program to act upon an analog level. This can be used for example to regulate pressure to a given level, by turning a motor on/off at a given voltage. **Eg /1at106144R** sets the threshold of channel 1 to 6144, Note that leading zeros are required for the threshold value which is always 5 digits plus the channel number.
?at		Reads back the thresholds for all 4 channels. The Readback order is channels 4:3:2:1 **Eg /1?at<CR>**
		ADDRESS BANKING TO SELECT >16 MOTORS
aB		Use this command <u>ONLY</u> if you have more than 16 motors on the bus. Else <u>DO NOT</u> use it. Allows upto 48 drives to be addressed by adding 16 or 32 to the value of the address switch.
	aB49520	Sets address bank to 0-15 (normal default mode)
	aB49521	Sets address bank to 16-32 (uses ascii 61-70 hex for addressing). So **issue /1aB49521 then /a?** (a=ascii 61)
	aB49522	Sets address bank to 33-47 (uses ascii 71-7f hex for addressing) If the bank is unknown issue /_aB49520 to globally set all addresses to bank zero.